

THESE

En vue de l'obtention du : **DOCTORAT**

Centre de Recherche : Centre de Recherches Mathématiques et Applications de Rabat.

Structure de Recherche : Laboratoire de Mathématiques, Informatique et Applications-Sécurité de l'Information (LABMIA-SI).

Discipline : Mathématiques Appliquées.

Spécialité : Optimisation et Science des Données.

Présentée et soutenue le 20/07/2022 par :

Ahmed BIR-JMEL

**Méthodes hybrides pour l'optimisation combinatoire:
Application au problème de classification de cancer.**

JURY

Awatef SAYAH
Abdeslam KADRANI

PES, Université Mohammed V, Faculté des Sciences de Rabat.
PES, Institut National de Statistique et d'Économie
Appliquée, Rabat.

Présidente/Rapportrice
Examineur/Rapporteur.

Halima LAKHBAB
Badreddine BENYACOB

PH, Université HASSAN II, Faculté des Sciences AIN CHOCK.
PA, Institut National de Statistique et d'Économie
Appliquée, Rabat.

Examinatrice/Rapportrice.
Invité

Souad EL BERNOUSSI
Sidi Mohamed DOURI

PES, Université Mohammed V, Faculté des Sciences de Rabat.
PH, Université Mohammed V, Faculté des Sciences de Rabat.

Co-Directrice.
Directeur

Année Universitaire : 2021-2022

*À la mémoire de mon Cher Grand-Père –El Haj Abdellah-.
À mes parents.
À tous ceux et toutes celles
qui m'ont accompagné et soutenu
durant ma formation.*

Ahmed BIR-JMEL.

*If you don't go after what you want,
you'll never have it. If you don't ask, the
answer is always no. If you don't step
forward, you're always in the same place.*

Nora Roberts

Remerciements

Les travaux présentés dans cette thèse ont été effectués au Département de Mathématique de la Faculté des Sciences de Rabat, Université Mohammed V. Cette thèse a été réalisée au sein du Laboratoire de Mathématiques, Informatique et Applications, Sécurité de l'Information (LABMIA-SI), dirigé par Madame Souad EL BERNOUSSI, Professeur de l'enseignement supérieur à la Faculté des Sciences de Rabat.

Je tiens tout d'abord à exprimer mes sincères remerciements, ma reconnaissance et ma gratitude à mon directeur de thèse bienaimé Monsieur Sidi Mohamed DOUIRI (PH), à la faculté des sciences de Rabat, pour la direction avisée et exigeante, ses qualités humaines et scientifiques et son soutien constant tout au long de cette thèse. C'est grâce à lui que j'ai pu poursuivre avec bonheur mes recherches.

Ma sincère gratitude va également à ma co-directrice de thèse bienaimée Madame Souad EL BERNOUSSI (PES), à la faculté des sciences de Rabat, pour son aide précieuse, pour la qualité de ses conseils, pour avoir toujours été là pour m'encourager, pour me soutenir, et pour m'avoir permis de reprendre confiance et m'avoir laissé la liberté nécessaire à l'accomplissement de mes travaux.

Je tiens à remercier également Madame Awatef SAYAH (PES), à la faculté des sciences de Rabat, qui a, de par son intérêt pour ma recherche, accepté de présider mon jury et de rapporter le travail de ma thèse.

Je tiens à exprimer ma gratitude à Monsieur Abdeslam KADRANI (PES), à l'Institut National de Statistique et d'Économie Appliquée pour avoir accepté de rapporter ma thèse, en tant que rapporteur externe et pour son intérêt porté à ma recherche.

Je remercie Madame Halima LAKHABAB (PH), à la faculté des Sciences AIN CHOCK, Université HASSAN II de Casablanca, d'avoir accepté rapporter ma thèse.

Je voudrais également remercier Monsieur Badreddine BENYACOUB (PA), à l'Institut National de Statistique et d'Économie Appliquée pour avoir accepté d'être invité à examiner mes travaux. J'adresse un grand merci à tous les membres de mon jury, pour avoir ainsi marqué leur intérêt pour mon travail, et pour les remarques qu'ils ont apportées durant la relecture et la soutenance de ma thèse.

Je remercie mon cher Professeur Jilali Mikram (PES), à la faculté des Sciences de Rabat, pour son aide, son encouragement, sa disponibilité et son soutien.

Je souhaite également exprimer ma reconnaissance envers tous les membres de LABMIA-SI, pour m'avoir accueilli chaleureusement, et pour toutes les conversations scientifiques ou non que l'on a pu avoir.

Un grand merci à tous mes professeurs qui ont, tout au long de ma carrière scolaire et académique, ont largement contribué à ma formation grâce à leur soutien, leur conseil et leur dévouement pour leur travail.

Je remercie tous les cadres professionnels de la faculté des sciences de Rabat, qui m'ont aidé à réaliser ce travail.

J'adresse pareillement un grand merci à mes ami(e)s avec qui j'ai partagé de grands moments lors de la rédaction de la thèse. Ils m'ont beaucoup soutenu durant ce parcours. Ils m'ont motivé et soutenu pendant les moments de stress et de fatigue. Mille mercis !

Je dédie ce travail à mon père, qui m'avait fait aimer le calcul dès mon jeune âge, à ma mère que dieu la bénisse et la garde, à mes très chers : frère, sœurs, nièces et neveux, je vous aime beaucoup !

Je témoigne ma reconnaissance à tous ceux qui ont contribué de près ou de loin à la réalisation de ce modeste travail.

Je n'oublie pas, en-fin, de prier pour l'âme de ma sœur qui nous a quittés tôt, durant les années de préparation de ce travail.

Résumé

DANS cette thèse, nous proposons des méthodes innovantes pour réduire la taille initiale des données de puces à ADN et sélectionner des ensembles de gènes pertinents qui assurent une bonne précision dans l'identification des tumeurs cancéreuses. Premièrement, nous avons proposé plusieurs méthodes de sélection de gènes, basées sur les Métaheuristiques BPSO (optimisation par essaim de particules version binaire), ACO (optimisation par colonies de fourmis) et VNS (recherche de voisinage variable).

En effet :

- Trois approches filtres multivariées ont été proposées : MWIS (ensemble indépendant de poids maximum), VNSMI et VNSCorr. Les deux dernières méthodes sont basées sur la combinaison de l'algorithme MRMR (redondance minimal-pertinence maximale) et d'une recherche intelligente via l'algorithme de VNS.
- D'autres approches hybrides ont été proposées telles que : FBPSO, RBPSO et MWIS-ACO-LS et ont montré des performances élevées, mais leur temps d'exécution est aussi important à cause de l'utilisation des classifieurs dans l'étape de sélection.

La qualité des sous ensembles de gènes dans ces méthodes est mesurée à l'aide des classifieurs SVM (Machine à vecteurs de support) et KNN (K plus proches voisins). Enfin, dans le but d'améliorer les méthodes intégrées, nous avons formulé le problème de sélection de gènes comme un problème de régression logistique (maximum de log-vraisemblance) avec une pénalité de type Fused LASSO généralisé. Pour résoudre ce problème d'optimisation, nous avons utilisé l'algorithme de sous-gradient. L'application de ce dernier nécessite la validation de certaines conditions de convergence. Dans notre travail, nous avons analysé théoriquement et numériquement cette méthode et vérifié que notre fonction objectif remplit toutes ces conditions.

Les différentes expérimentations que nous avons menées montrent que nos approches sont performantes et compétitives et produisent des sous-ensembles de petite taille avec une haute performance pour la classification des cancers.

Mots-clefs : Sélection d'attributs ; Classification supervisée, puces à ADN ; Optimisation par Essaim de Particules ; Optimisation par Colonies de Fourmis ; Recherche de Voisinage Variable ; Ensemble Indépendant de Poids Maximum ; Régression Logistique Pénalisée ; Generalized Fused LASSO.

Abstract

IN this thesis, we propose innovative methods to reduce the initial dimension of microarray data, and select relevant gene sets that ensure good accuracy in cancer tumor identification : First, we proposed several gene selection methods, based on the metaheuristics BPSO (Binary Particle Swarm Optimization), ACO (Ant Colony Optimization) and (VNS Variable Neighborhood Search), indeed :

- Three multivariate filter approaches have been proposed : MWIS (Maximum Weight Independent Set), VNSMI and VNSCorr. The last two methods are based on the combination of the MRMR (minimum redundancy-maximum relevance) algorithm and an intelligent search via the VNS algorithm.
- Other hybrid approaches have been proposed such as : FBPSO, RBPSO and MWIS-ACOLS and have shown high performances, but their execution time is also important because of the use of classifiers in the selection step.

The quality of the gene subsets in these methods is measured using SVM (Support Vector Machine) and KNN (K-Nearest Neighbors) classifiers. And finally, in order to improve the integrated methods, we formulated the gene selection problem as a logistic regression problem (maximum log-likelihood) with a generalized Fused LASSO penalty. To solve this optimization problem, we used the subgradient algorithm. The application of the latter requires the validation of some convergence conditions. In our work, we have analyzed theoretically and numerically this method and verified that our objective function meets all these conditions.

The different experiments we have conducted show that our approaches are efficient and competitive and produce small subsets with high performance for cancer classification.

Keywords : Feature selection ; Supervised classification ; DNA microarray ; Particle Swarm Optimization, Ant Colony Optimization, Variable Neighborhood Search ; Maximum Weight Independent Set ; Penalized Logistic Regression ; Generalized Fused LASSO.

Liste des Symboles

<i>KNN</i>	K-Nearest Neighbors.
<i>SVM</i>	support vector machine.
<i>LR</i>	Régression logistique.
<i>LOOCV</i>	Leave-One-Out-Cross-Validation.
<i>SFS</i>	Sequential Forward Selection.
<i>SBS</i>	Sequential Backward Selection.
<i>SFFS</i>	Sequential Forward Floating Selection.
<i>SBFS</i>	Sequential Backward Floating Selection.
<i>VNS</i>	Recherche à Voisinage Variable.
<i>VNSMI</i>	Méthode basée sur VNS et l'information mutuelle.
<i>VNSCorr</i>	Méthode basée sur VNS et le coefficient de corrélation de Pearson.
<i>GA</i>	Algorithmes Génétiques (AGs).
<i>TS</i>	la recherche Tabu.
<i>PSO</i>	Optimisation par Essaims Particulaires.
<i>BPSO</i>	Optimisation par Essaims Particulaires version binaire.
<i>ACO</i>	Algorithmes de Colonies de Fourmis.
<i>IG</i>	Information Gain.
<i>LASSO</i>	Least Absolute Shrinkage and Selection Operator.
<i>MRMR</i>	Minimum Redundancy Maximum Relevance.
<i>FRBPSO</i>	A Fuzzy Rule Based Binary PSO.
<i>IBPSO</i>	An improved binary PSO.
<i>TS – BPSO</i>	A combination of tabu search and BPSO.
<i>EPSO</i>	An enhancement of binary particle swarm optimization.
<i>MWIS</i>	Maximum Weight Independent Set.
<i>FBPSO</i>	Méthode basée sur le score de Fisher et BPSO.
<i>RBPSO</i>	Méthode basée sur l'algorithme de ReliefF et BPSO.
<i>MWIS – ACO – LS</i>	Méthode basée sur MWIS, ACO et nouvel algorithme de recherche locale.
<i>SGD</i>	Algorithme du gradient stochastique.
<i>MI</i>	Mutual information.
<i>NI</i>	Normalized Mutual information.
<i>SLS</i>	Stochastic Local Search.
<i>PLR</i>	Régression Logistique Périalisée.
<i>FGLASSO</i>	Fused LASSO Généralisé.
<i>ALASSO</i>	Adaptive LASSO.

Table des matières

Introduction Générale	12
Contexte de travail	12
Organisation du rapport	20
1 Technologie des puces à ADN et Classification	22
1.1 Technologie des puces à ADN	22
1.2 Classification supervisée	24
1.2.1 Formalisation mathématique du problème de classification supervisée et Notations :	25
1.2.2 Techniques de la classification supervisée	25
1.2.3 Estimation des performances du classifieur et sur-apprentissage	35
1.3 Conclusion	36
2 Sélection d'attributs	38
2.1 Notions de pertinence et redondance	41
2.1.1 Pertinence d'attributs	42
2.1.2 Redondance d'attributs	42
2.2 Présentation formelle du problème de sélection	44
2.3 Processus global de la sélection d'attributs	45
2.3.1 Directions et stratégies de recherche	46
2.3.2 Critère d'évaluation	48
2.3.3 Critère d'arrêt	52
2.3.4 Validation des résultats	53
2.4 Conclusion	54
3 Méthodes Hybrides pour la sélection de gènes pour la classification du cancer	56
3.1 Introduction	56
3.2 Jeux de données	58
3.3 Sélection de gènes via BPSO et une recherche locale de type (backward) pour la classification du cancer	60

3.3.1	Approche proposée	60
3.3.2	Expérimentations et résultats	71
3.4	Sélection de gènes via un nouvel algorithme hybride d'optimisation par colonies de fourmis pour la classification du cancer	84
3.4.1	Méthodes	84
3.4.2	Expérimentations et résultats	98
3.5	Recherche de voisinage variable (VNS) pour la sélection de gènes	111
3.5.1	Approche proposée	111
3.5.2	VNS pour la sélection de gènes	115
3.5.3	Expérimentations et résultats	120
3.6	Conclusion	126
4	Méthode intégrée pour la sélection de gènes (Fused-LASSO "Fused Least Ab- solute Shrinkage Selection Operator")	128
4.1	Approche proposée	131
4.1.1	Méthode du sous-gradient	131
4.1.2	Régression logistique pénalisée (PLR)	143
4.2	Méthode proposée (FGLASSO-LR)	146
4.2.1	Sous-gradient pour l'estimation de $\hat{\beta}_{\text{FGLASSO},\lambda_1,\lambda_2}$	148
4.3	Expérimentations et résultats	154
4.3.1	Jeux de données	155
4.3.2	Paramètres	155
4.3.3	Résultats et comparaisons	161
4.4	Conclusion	165
	Conclusion et perspectives	167
	Appendices	169
A	Théorie des graphes	170
A.1	Notations de base	170
A.2	Opérations sur les graphes	171
B	Notions sur la complexité	172
B.1	Ordre de grandeur et temps d'exécution	172
B.2	Différents types de complexité	173
	Bibliographie	175

Liste des figures

1	Flux de nos contributions proposées pour la sélection de sous-ensembles de gènes dans les données de puces à ADN.	17
1.1	Méthode des K plus proches voisins. Le nouvel échantillon (étoile) est affecté à la classe majoritaire parmi ses K (ici K=3) plus proches voisins, ici la classe 2 [49] . .	26
1.2	Exemple d'un arbre de décisions.	27
1.3	Hyperplan optimal de séparation [87].	30
1.4	Hyperplans séparateurs dans le cas des exemples non linéairement séparables [87]. .	31
1.5	Stratégie un contre tous.	32
1.6	Stratégie un contre un.	32
1.7	Graphique de la fonction sigmoïde	34
2.1	Les trois figures rouges (la ligne, le carré et le cube) ont une longueur, une surface ou un volume correspondant à la moitié des figures noires qui les contiennent. Plus la dimensionnalité augmente, plus le côté de la figure rouge doit grandir pour que la proportion reste 1/2.[2]	39
2.2	Le fléau de la dimensionnalité est bien illustré par un voisinage hyper-cubique pour des données uniformes dans un cube unité. La figure de droite montre la longueur de côté de l'hypercube nécessaire pour couvrir une fraction r du volume des données, pour différentes dimensions p. En dix dimensions, nous devons couvrir 80 % de la plage de chaque coordonnée afin de capturer 10% des données.[1].	40
2.3	La sélection des caractéristiques et le fléau de la dimensionnalité.	40
2.4	Catégorisation des attributs	44
2.5	Sous-ensembles possibles à partir d'un ensemble de 4 attributs	45
2.6	Processus de sélection d'attributs	46
2.7	Méthodes de sélection de gènes	48
3.1	Déplacement d'une particule [61].	61
3.2	Graphe d'influence d'un essaim de particules. (à gauche) Graphe complètement connecté, (à droite) Graphe d'information circulaire [58].	63
3.3	Exemple illustré d'un sous-ensemble généré.	65

3.4	Un exemple illustre la simulation de deux itérations de l'approche en arrière ; g_1 et g_6 ne sont pas pertinents.	66
3.5	Organigramme de nos approches proposées pour la sélection de sous-ensembles de gènes dans les données de puces à ADN.	69
3.6	Schéma général de FBPSO-SVM.. . . .	71
3.7	Étude paramétrique pour le poids d'inertie w	72
3.8	Comparaison des taux d'exactitudes de classification entre les quatre méthodes (Best).	75
3.9	Comparaison des temps d'exécution moyens en heures.	76
3.10	Comparaison de l'évolution du nombre de gènes utilisés pour "9_Tumor1".	77
3.11	Comparaison de l'évolution du taux d'exactitude de la Classification pour "9_Tumors".	77
3.12	Comparaison de l'évolution du nombre de gènes utilisés pour "Brain_Tumor1".	77
3.13	Comparaison de l'évolution du taux d'exactitude de la Classification pour "Brain_Tumor1".	77
3.14	Comparaison de l'évolution du nombre de gènes utilisés pour "Brain_Tumor2".	78
3.15	Comparaison de l'évolution du taux d'exactitude de la Classification pour "Brain_Tumor2".	78
3.16	Le résultat du test de kruskal-wallis entre les approches proposées, SVM et 1NN sur les jeux de données	79
3.17	Comparaison du temps nécessaire pour la construction du classifieur 1NN avant et après la sélection à l'aide de RBPSO.	80
3.18	Comparaison du temps nécessaire pour la construction du classifieur SVM avant et après la sélection à l'aide de FBPSO.	80
3.19	Un exemple de graphe de similarité entre les gènes.	87
3.20	Exemple d'ensemble indépendant de poids maximum	89
3.21	Un exemple illustré avec un sous-ensemble généré et une représentation de chemin.	90
3.22	La procédure de sélection de gènes dans l'ACO modifié.	91
3.23	Organigramme de notre approche de recherche locale.	94
3.24	Organigramme de nos approches proposées pour la sélection de sous-ensembles de gènes dans les données de puces à ADN.	96
3.25	Comparaison de l'exactitude de classification entre MWIS, MWIS-ACO et MWIS-ACO-LS (pour MWIS-ACO et MWIS-ACO-LS nous prenons la solution moyenne).	102
3.26	Comparaison du nombre de gènes utilisés dans la classification entre MWIS, MWIS-ACO et MWIS-ACO-LS (pour MWIS-ACO et MWIS-ACO-LS nous prenons la solution moyenne).	103
3.27	Comparaison de l'exactitude de classification entre les cinq méthodes (pour MWIS, MWIS-ACO-LS et $LR - L_1$ nous prenons la solution moyenne).	103
3.28	Comparaison de l'évolution du nombre de gènes utilisés pour "9_Tumors".	105
3.29	Comparaison de l'évolution de l'exactitude de la classification pour "9_Tumors".	105
3.30	Comparaison de l'évolution du nombre de gènes utilisés pour "Brain_Tumor1".	105
3.31	Comparaison de l'évolution de l'exactitude de la classification pour "Brain_Tumor1".	105
3.32	Comparaison de l'évolution du nombre de gènes utilisés pour "Brain_Tumor2".	106
3.33	Comparaison de l'évolution de l'exactitude de la classification pour "Brain_Tumor2".	106
3.34	Comparaison de l'évolution du nombre de gènes utilisés pour "Leukimia1".	106
3.35	Comparaison de l'évolution de l'exactitude de la classification pour "Leukimia1".	106

3.36	Le résultat du test de kruskal-wallis entre le MWIS-ACO-LS, 1NN et MWIS sur les jeux de données (exactitude de la classification).	106
3.37	Le résultat du test de kruskal-wallis entre le MWIS-ACO, 1NN et MWIS sur les jeux de données (exactitude de la classification).	107
3.38	La relation entre l'entropie et l'information mutuelle	112
3.39	Organigramme de VNS basique.	114
3.40	Un exemple illustré avec un sous-ensemble généré et une représentation de solution.	116
3.41	Organigramme de l'approche proposée (<i>VNSMI</i>) pour la sélection de gènes dans les données de puces à ADN.	119
3.42	Comparaison de l'exactitude moyenne des approches proposées en utilisant (1NN).	124
3.43	Comparaison de l'exactitude moyenne des approches proposées en utilisant (SVM).	124
3.44	Comparaison de l'évolution du nombre de gènes utilisés pour "Colon" à l'aide de VNSMI.	125
3.45	Comparaison de l'évolution du nombre de gènes utilisés pour "Colon" à l'aide de VNSCor.	125
3.46	Comparaison de l'évolution de $\mathcal{J}$ pour "Colon" en utilisant VNSMI.	125
3.47	Comparaison de l'évolution de $\mathcal{J}$ pour "Colon" en utilisant VNSCor.	125
4.1	Ensemble convexe et ensemble non convexe	132
4.2	Fonction convexe et fonction non convexe.	132
4.3	Épigraphe d'une fonction convexe et d'une fonction non convexe.	133
4.4	Sous-différentiel de la fonction valeur absolue à l'origine.	135
4.5	Sous-différentiel de la fonction valeur absolue.	135
4.6	Sous-différentiel de la fonction norme 1, pour le cas $n = 2$.	137
4.7	Organigramme de l'approche proposée (<i>FGLASSO-LR</i>) pour la sélection de gènes dans les données de puces à ADN.	148
4.8	Comparaison entre les pas α_k et α'_k pour minimiser la fonction h_{λ_1, λ_2} .	156
4.9	Comparaison entre les pas α'_k avec des différents γ pour la minimisation de h_{λ_1, λ_2} .	158
4.10	Comparaison des taux d'exactitude trouvés à l'aide de différentes valeurs de λ_1 pour "Prostate_Tumor" (Validation croisée d'ordre 5.)	159
4.11	Comparaison des taux d'exactitudes de classification (LASSO-LR et FGLASSO-LR).	162
4.12	Comparaison des taux d'exactitudes de classification (1NN).	163
4.13	Comparaison des taux d'exactitudes de classification (SVM).	164
B.1	Temps nécessaire à l'exécution d'un algorithme en fonction de sa complexité.	173
B.2	Classes de complexité.	174

Liste des tableaux

1.1	Exemple d'une matrice associée à une puce à ADN	23
2.1	La matrice de confusion, un tableau des quatre différentes combinaisons de valeurs de classe prédites (lignes) et réelles (colonnes).	54
3.1	La description des puces à ADN utilisées.	59
3.2	Paramètres utilisés pour nos méthodes.	73
3.3	Comparaison de SVM, 1NN, RBPSO-1NN et FBPSO-SVM (LOOCV).	74
3.4	Comparaison entre nos méthodes (RBPSO-1NN et FBPSO-SVM) et les méthodes de l'état de l'art.	82
3.5	Matrice de corrélation absolue	86
3.6	Matrice d'adjacence	86
3.7	Paramètres utilisés pour les expériences.	100
3.8	Comparaison entre LR, SVM, 1NN, MWIS-1NN et MWIS-ACO-LS (LOOCV).	101
3.9	Liste des gènes sélectionnés à l'aide de MWIS-ACO-LS.	108
3.10	Comparaison entre notre méthode (MWIS-ACO-LS) et des méthodes de l'état de l'art.	109
3.11	Classement de notre méthode comparativement aux méthodes existantes selon l'exactitude moyenne.	110
3.12	Description des jeux de données (ADN microarray) utilisés.	121
3.13	Paramètres utilisés dans VNS.	122
3.14	Comparaison entre SVM, 1NN, VNSMI et VNSCor (LOOCV).	123
4.1	Méthodes de régression pénalisées populaires	145
4.2	Description des jeux de données (ADN microarray) utilisés dans FGLASSO-LR.	155
4.3	Comparaison entre les pas α'_k avec des différents γ pour la minimisation de h_{λ_1, λ_2}	157
4.4	Comparaison entre les λ_1 pour le jeu de données "Prostate_Tumor" (Validation croisée d'ordre 5 avec $\lambda_2 = 0.2$	159
4.5	Comparaison entre les λ_2 pour le jeu de données "Prostate_Tumor" (Validation croisée d'ordre 5 avec $\lambda_1 = 1.6$	160

4.6	Les résultats obtenus sur 50 exécutions, sans et avec sélection via les pénalités LASSO et FG-LASSO	161
4.7	Comparaison entre notre méthode (FGLASSO-LR) et des méthodes de l'état de l'art.	165
B.1	Qualifications usuelles des complexités.	173

Liste des algorithmes

1	Algorithme d'apprentissage générique	28
2	Algorithme de sélection ReliefF	50
3	Algorithme de PSO général	64
4	Backward generation	66
5	Approche proposée	70
6	Construction du graphe de similarité entre les gènes.	87
7	Algorithme glouton pour approximer le MWIS.	89
8	Algorithme de recherche locale pour la sélection de gènes.	95
9	Approche proposée (MWIS-ACO-LS)	97
10	Méthode de perturbation	117
11	Recherche locale stochastique (SLS) pour la sélection de gènes	118
12	Approche de recherche de voisinage variable pour la sélection de gènes.	120
13	Algorithme de sous-gradient pour minimiser h_{λ_1, λ_2}	154

Introduction Générale

Mathematics is the most beautiful and most powerful creation of the human spirit.

Stefan Banach

Contexte de travail

Actuellement, la technologie a atteint un point où des ensembles de données massifs, tels que les données de puces à ADN, se sont imposés comme une partie permanente de l'analyse statistique. Les problèmes découlant des données de grande dimension comprennent des tâches de classification ainsi que l'identification d'un sous-ensemble pertinent de gènes.

Les mutations génétiques sont les principales causes sous-jacentes de nombreux troubles et maladies existants. Ainsi, les activités des gènes et leurs interactions doivent être déterminées afin d'examiner le mécanisme biochimique [5]. Les puces à ADN sont largement impliquées dans diverses disciplines scientifiques de la médecine et de la biologie. De nos jours, les nombreuses études ayant recours aux puces à ADN comprennent la corégulation des gènes, le diagnostic clinique, la découverte de la fonction des gènes, l'expression différentielle des gènes et les modèles d'activité des gènes sous différents traitements chimiques [180].

C'est un fait bien établi que les déviations systématiques des modèles d'expressions géniques au sein d'un type spécifique de cellules sont directement corrélées avec certaines variations biologiques appartenant à un type de cancer spécifique. Cependant, pour utiliser ce fait bien défini dans la lutte contre le cancer, la sélection d'un petit nombre de gènes prédictifs à partir des données de biopuces de grande dimension est primordiale [122].

La technologie des puces à ADN offre donc des opportunités pour démêler les événements cellulaires et remodeler les sciences de la vie. En particulier, dans la recherche sur le cancer, la technologie de puces à ADN est souvent utilisée pour identifier les gènes qui peuvent avoir un rôle fonctionnel important dans la croissance tumorale et pour construire des réseaux de régulation génique pour faire des inférences sur les mécanismes biologiques et pathologiques. Ceci facilitera

la conception d'un remède efficace pour lutter contre le cancer. De plus, l'identification de gènes importants aide à développer des systèmes de prédiction de diagnostic informatisés.

Parallèlement à ces avancées biotechnologiques, le développement de techniques informatiques est d'une importance capitale, car l'analyse informatique systématique des ensembles de données d'expression génique des puces à ADN est une étape vitale et inévitable pour extraire des informations utiles à partir des données d'expression génique. Ces techniques impliquent la classification, la régression, la détection des valeurs aberrantes, l'identification différentielle des gènes, la sélection des gènes, le regroupement et bien d'autres. Les modèles de classification peuvent être globalement divisés en deux catégories : supervisés et non supervisés. Dans l'approche supervisée, le processus de classification consiste à placer les objets dans la catégorie préexistante, dans laquelle le modèle de classification a été préalablement appris à partir des données accompagnant les informations de classes. D'autre part, les approches non supervisées consistent à développer un ensemble de catégories (classes) tout en plaçant les échantillons dans les classes, par exemple, pour différents stades du cancer. Étant donné que les informations sur la classe sont déjà disponibles dans les biopuces, l'analyse des puces à ADN pour la classification du cancer est formulée comme un problème de classification supervisée dans cette recherche.

Dans la classification du cancer à l'aide d'ensembles de données de biopuces, nous devons diagnostiquer si un échantillon provient d'un tissu cancéreux ou bénin, ou distinguer différents types de tumeurs cancéreuses. Les ensembles de données des puces à ADN posent de réels défis aux techniques de calcul en raison des complexités informatiques ainsi que des subtilités biologiques. Par exemple, le coût élevé des expériences sur puces à ADN ne permet de collecter qu'un petit nombre d'échantillons dans une expérience. Cela se traduit par des ensembles de données avec très peu d'échantillons (en centaines), par rapport aux milliers de gènes dont les expressions géniques sont mesurées simultanément. Ces expressions forment un ensemble de données de grande dimension, et par conséquent, les techniques standard souffrent d'instabilités informatiques et de malédiction de la dimensionnalité (curse of dimensionality). Cela fait de la sélection de gènes (également appelée sélection d'attributs, caractéristiques ou variables) un problème important et difficile dans l'analyse des données de puces à ADN. Du point de vue biologique, les scientifiques s'intéressent à un petit ensemble de marqueurs qui peuvent être testés expérimentalement au laboratoire, conduisant finalement à des traitements efficaces pour une maladie particulière. Par conséquent, une approche efficace de sélection de gènes permet une meilleure interprétation de la relation biologique entre les gènes et le résultat clinique considéré, puis une compréhension scientifique adéquate du problème donné. De plus, elle surpasse la malédiction de la dimensionnalité (curse of dimensionality) afin d'améliorer la qualité des classifieurs.

Le problème de la sélection de gènes nécessite alors le développement d'approches particulières puisque le nombre de gènes dépasse de loin celui d'échantillons (expériences ou observations), la sélection d'un sous ensemble de gènes est un problème NP-difficile qui peut être résolu par des méthodes avancées de l'optimisation combinatoire[17, 28].

Bien que plusieurs méthodes de sélection de gènes aient été proposées pour augmenter l'exactitude de la classification, des recherches supplémentaires sont jugées nécessaires pour proposer de nouveaux modèles qui peuvent encore augmenter le pronostic et la précision de la classification de maladies telles que le cancer. Atteindre une grande exactitude de classification est d'une importance cruciale pour la médecine personnalisée, car cela conduirait à des décisions plus judicieuses

de la part des médecins et par la suite sauverait la vie des patients.

Cette recherche a évolué de multiples façons en introduisant plusieurs nouveaux algorithmes de sélection de gènes. Pour démontrer l'efficacité de ces approches, plusieurs tests ont été effectués sur de nombreux jeux de données relatifs aux cancers prélevés sur diverses tumeurs de côlon, de sein, de poumon, de leucémie etc.

Motivation et objectifs

Le diagnostic clinique implique la découverte de gènes informatifs de la maladie et la sélection du bon diagnostic basé sur les gènes pertinents découverts. Actuellement, le principal objectif de recherche pour l'analyse des données des puces à ADN est la formulation de techniques génériques pour la classification des maladies cancéreuses. Ceci est principalement dû au fait que les puces à ADN génèrent des informations biologiques sans précédent concernant la pathologie et la progression de la maladie, les changements au niveau moléculaire, la résistance et la réponse aux thérapies administrées spécifiques [5].

Le diagnostic précis et rapide d'une maladie cancéreuse et la classification de ses sous-types sont les principaux défis auxquels est confronté le domaine médical aujourd'hui [85]. Le diagnostic du cancer est principalement basé sur les propriétés morphologiques des tissus malades/malins. Cependant, les classifications qui reposent uniquement sur ces caractéristiques morphologiques se sont révélées insuffisantes [207]. En effet, les tumeurs morphologiquement similaires peuvent être classées en classes distinctes à l'aide des profils d'expression génique.

La sélection des gènes est à la fois une étape cruciale et un sujet de recherche largement abordé dans l'analyse des puces à ADN. Les gènes dans ces puces peuvent être pertinents ou non pour la classification des échantillons. Les gènes pertinents sont importants pour la classification, alors que les gènes non pertinents peuvent dégrader les performances du classificateur et n'ont aucune pertinence biologique.

Bien qu'une multitude de travaux de recherche aient déjà été consacrés à la sélection des gènes et que chaque méthode ait ses propres avantages et inconvénients, de nombreuses méthodes ne parviennent pas à identifier des biomarqueurs non redondants, reproductibles et véritables pour classer efficacement les tumeurs cancéreuses. Les motivations et les objectifs spécifiques de cette thèse sont les suivants :

1. La plupart des méthodes de filtrage existantes consistent à évaluer chaque attribut, pour lui assigner un score (une mesure) de pertinence, ce score permet un classement des attributs. La sélection d'attributs se fait par le choix des attributs les mieux classés. De plus, la définition de la pertinence d'un attribut est incomplète si les caractéristiques du classificateur ne sont pas considérées [186]. Récemment, de nombreux algorithmes ont été développés afin de minimiser la redondance entre les gènes et de maximiser la pertinence. Cependant, les caractéristiques du classificateur n'ont pas été prises en compte pour le classement des gènes dans les méthodes de filtrage.

Dans cette thèse, nous avons développé des méthodes de sélection de gènes dont l'objectif est d'obtenir un meilleur compromis entre la pertinence et la redondance des gènes ainsi que les

caractéristiques du classificateur afin d’atteindre de meilleures performances de classification avec moins de gènes.

2. Les méthodes de sélection de gènes souffrent d’une faible stabilité ou reproductibilité du classement des gènes et présentent une complexité de calcul élevée. L’utilisation de méthodes de filtrage avec des méthodes enveloppes ou intégrées peut améliorer la stabilité sans compromettre les performances de classification. Nous avons donc combiné une méthode de filtrage avec une méthode enveloppe afin de dériver une méthode de sélection de gènes stable et moins coûteuse en temps de calcul avec de bonnes performances en matière de classification.
3. Dans la majorité des méthodes filtres, la corrélation entre les gènes sélectionnés n’est pas prise en considération. Ceci implique la sélection de sous-ensembles avec un niveau de redondance élevé qui pénalise les performances des classifieurs ; d’autre part, ces méthodes éliminent les gènes avec un score individuel faible, ignorant la possibilité de devenir très pertinents lorsqu’ils sont combinés avec d’autres gènes [196]. Cela nous a motivé à proposer une approche basée sur la théorie de graphes pour surmonter ces problèmes.
4. Afin d’améliorer la stabilité, la diversité et la robustesse des méthodes de sélection de gènes, nous avons proposé et analysé plusieurs méthodes d’optimisation combinatoire tels que : les métaheuristiques PSO, ACO, et VNS, ces derniers algorithmes ont été hybridés avec plusieurs techniques connues à savoir des méthodes de recherches locales et des méthodes inspirées de la théorie des graphes.
5. Les méthodes intégrées diffèrent des autres méthodes de sélection de gènes par la manière dont la sélection et l’apprentissage interagissent. Les méthodes filtres n’intègrent pas l’apprentissage. Les méthodes enveloppes utilisent une machine d’apprentissage pour mesurer la qualité de sous-ensembles de gènes sans incorporer de connaissances sur la structure spécifique de la fonction de classification ou de régression, et peuvent donc être combinées avec n’importe quelle machine d’apprentissage. Contrairement à ces approches, dans les méthodes intégrées, la partie apprentissage et la partie sélection de gènes ne peuvent pas être séparées. Parmi les méthodes intégrées nous trouvons la régression logistique pénalisée, où la sélection s’effectue dans la tâche de construction du classifieur de régression logistique. Dans notre travail, nous avons adapté une pénalité de type Fused LASSO pour le problème de sélection de gènes. Dans la méthode proposée intervient le terme de valeur absolue dans la fonction objectif, ainsi les méthodes classiques d’optimisation des fonctions différentiables ne peuvent pas être appliquées à ce genre de problèmes.
6. Un grand nombre de problèmes survenant dans la résolution de problèmes de programmation mathématique complexes peuvent être réduits à la minimisation de fonctions convexes non différentiables, ces problèmes sont en général très difficiles à résoudre. En particulier, notre méthode de sélection de type Fused LASSO rentre dans cette classe. Afin de surmonter cet obstacle, nous avons proposé un nouvel algorithme de résolution connu sous le nom de sous-gradient, l’application de ce dernier algorithme nécessite la validation de quelques conditions de convergence, en effet, la fonction objectif doit être convexe, lipschitzienne et doit admettre un minimum global. Dans notre travail nous avons analysé théoriquement et

numériquement cette méthode et vérifié que toutes les conditions sont vérifiées par notre fonction objectif.

Contributions

La revue de littérature montre que de nombreux chercheurs sont activement impliqués dans l'analyse des puces à ADN pour le cancer. En effet, les puces à ADN ont représenté un énorme potentiel dans la lutte contre le cancer.

Les expériences sur les biopuces génèrent une quantité volumineuse de données qui sont directement liées aux mutations génétiques résultant d'une maladie cancéreuse spécifique. Ainsi, il est souhaitable de construire des outils de diagnostic fiables basés sur ces puces afin d'accélérer le processus de diagnostic et de classification du cancer.

La réduction de la dimensionnalité a été identifiée comme un problème clé dans la conception d'outils fiables de diagnostic du cancer. Jusqu'à présent, un certain nombre de techniques de sélection de gènes ont été rapportées pour résoudre ce problème. La majorité de ces approches relèvent des catégories de filtres et enveloppes et intégrées. Bien que les approches de filtrage soient efficaces et rapides sur le plan informatique par rapport aux enveloppes, elles sont indépendantes du classificateur et ignorent les interactions possibles entre les gènes au sein des puces à ADN, ce qui les rend inefficaces pour la tâche de classification des biopuces.

D'autre part, les méthodes enveloppes offrent des précisions de classification intéressantes par rapport aux filtres, mais ils ont un coût de calcul élevé résultant d'une convergence lente. Cela rend ces méthodes trop inefficaces pour la tâche de classification des puces à ADN.

Finalement, nous trouvons les méthodes intégrées dans lesquelles les mécanismes de sélection sont intégrés dans l'algorithme d'apprentissage. Ces méthodes sont plus efficaces et plus faciles à gérer que les méthodes enveloppes, tout en maintenant des performances similaires. En effet, les méthodes intégrées évitent l'exécution répétitive du classificateur et l'examen de chaque sous-ensemble de gènes. De plus, ces approches présentent un risque plus faible de surapprentissage par rapport aux méthodes enveloppes. Les méthodes intégrées prennent en compte les dépendances entre les gènes, mais ne sont spécifiques qu'à un algorithme d'apprentissage donné. Cependant, la complexité de calcul est le problème majeur, en particulier quand les données ont de grande taille.

Ainsi, il est évident que ni une approche de filtre unique ni une approche d'enveloppe ou intégrée ne peuvent offrir une réduction de dimensionnalité optimale pour les puces à ADN. Les dernières approches doivent être améliorées ou hybridées pour être efficaces dans la réduction de la dimensionnalité des ensembles de données de biopuces.

Le cœur de cette thèse est la sélection de gènes pour la classification des puces à ADN de grande taille. Comme la sélection de gènes vise à trouver un petit sous-ensemble de gènes pertinents et non redondants qui peut résumer avec succès l'espace de gènes d'origine, notre stratégie de travail adopté pour cette recherche est décrite dans la figure.1 (les contributions remarquables ont la forme d'une ellipse). À partir de la figure.1, cette recherche a apporté des contributions notables, qui sont résumées dans les sous-sections suivantes.

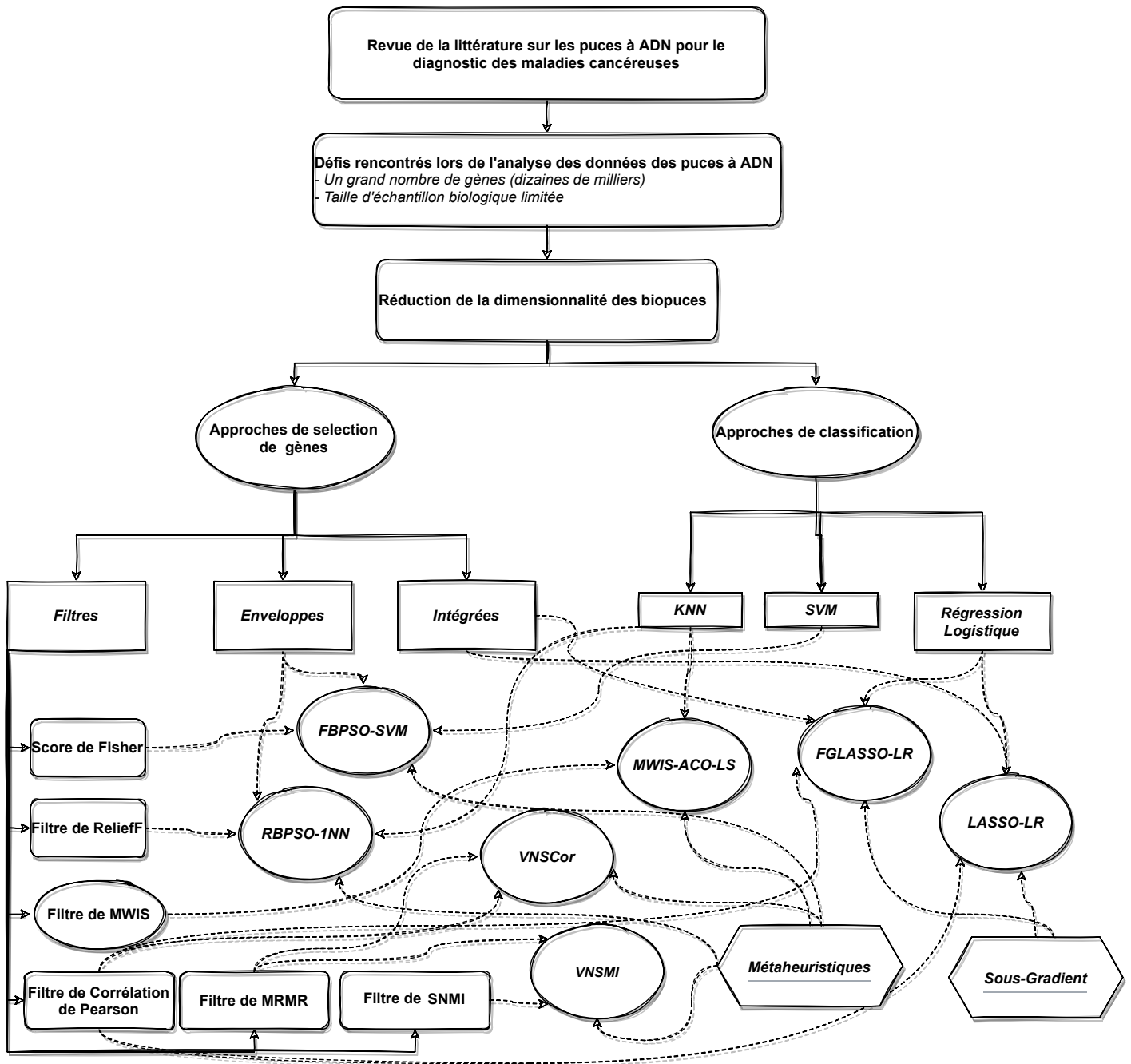


figure. 1: Flux de nos contributions proposées pour la sélection de sous-ensembles de gènes dans les données de puces à ADN.

PSO binaire :

Pour sélectionner un sous-ensemble de gènes informatifs à partir des puces à ADN de grande

taille, deux méthodes hybrides basées sur la métaheuristique de l'optimisation par essaim des particules (BPSO) sont proposées. Les deux approches consistent en une phase de présélection réalisée à l'aide des méthodes de filtrage Fisher et ReliefF pour FBPSO-SVM et RBPSO-1NN respectivement et une phase de recherche qui détermine un bon sous-ensemble de gènes pour la classification. Cette dernière est basée sur deux algorithmes, le BPSO et une méthode de recherche locale (Backward generation) pour surmonter le piégeage des minima locaux du BPSO. Ces approches visent à sélectionner un sous-ensemble optimal de gènes pertinents à partir d'un ensemble de données contenant des données redondantes, bruyantes ou non pertinentes. Les résultats expérimentaux montrent que nos approches se comparent très favorablement aux méthodes de référence en termes d'exactitude de classification et de nombre de gènes sélectionnés.

MWIS-ACO-LS :

Nous remarquons que la majorité des méthodes filtres sont univariées, la corrélation entre les gènes sélectionnés alors n'est pas prise en compte en général. Ceci implique la sélection de sous-ensembles avec un niveau de redondance élevé qui pénalise les performances du classifieur ; en revanche, ces méthodes éliminent les gènes avec un faible score individuel, ignorant ainsi la possibilité qu'ils puissent devenir très pertinents lorsqu'ils sont combinés avec d'autres gènes. Pour pallier à ces problèmes, nous proposons une approche basée sur la théorie de graphes. Dans la première étape de notre méthode, nous considérons le problème de sélection de gènes comme la recherche de l'ensemble indépendant de poids maximum MWIS dans le graphe de similarité entre les gènes. Le choix de ce sous-ensemble est justifié par deux arguments : Premièrement, le terme poids maximum peut être traduit dans le contexte de la sélection de gènes comme la sélection d'un sous-ensemble de gènes avec une pertinence maximale. Deuxièmement, la notion d'indépendance assure le choix d'un sous-ensemble avec une redondance minimale, c'est-à-dire que dans ce sous-ensemble, il n'y a pas deux gènes fortement corrélés. De plus, ce sous-ensemble peut contenir des gènes avec un score faible. Par conséquent, la méthode proposée à ce stade donne un bon sous-ensemble de gènes pour appliquer un algorithme évolutif tel que l'ACO. Ainsi, nous proposons une nouvelle approche hybride (MWIS-ACO-LS) pour résoudre le problème de sélection de gènes basée sur la célèbre métaheuristique de colonies de fourmis (ACO). Cette méthode commence par une phase de présélection où le problème de sélection est vu comme la recherche d'un ensemble indépendant de poids maximum (MWIS) dans un graphe pondéré, par suite dans la deuxième phase : un nouvel ACO couplé à un algorithme de recherche locale (LS) est proposé pour trouver l'ensemble de gènes pertinents, dans cette dernière phase la qualité d'un tel ensemble est mesurée à l'aide du classifieur 1NN. Les résultats expérimentaux montrent que notre approche se compare très favorablement aux méthodes de référence en termes de précision de classification et de nombre de gènes sélectionnés. Bien que les résultats obtenus soient intéressants et encourageants.

VNSMI et VNSCor :

Dans le but d'améliorer les méthodes filtres déjà existantes, nous avons pensé à profiter des avantages de la métaheuristique (Recherche 'à voisinages variables -VNS-). En effet, deux approches filtres multivariées basées sur la notion de MRMR (redondance minimal-pertinence maximale "

Maximum Relevance — Minimum Redundancy") sont présentées (VNSMI et VNSCor).

Notre méthode proposée VNSMI pour la sélection de gènes est basée sur la combinaison entre deux approches filtres (un schéma univarié, et un autre multivarié). Le schéma univarié (SNMI) est basé sur le score de l'information mutuelle normalisée en sélectionnant les p gènes les mieux classés, tandis que le schéma multivarié se fait via un VNS couplé à un algorithme de recherche locale stochastique appelé SLS. La qualité du sous-ensemble final de gènes est mesurée par un classificateur spécifique avec une validation croisée de type : (LOOCV "leave one out cross validation"). Le déroulement général de cette approche se caractérise par un processus séquentiel en deux phases :

Phase 1 : Cette phase de prétraitement consiste à réduire la taille des données initiales des puces à ADN, en filtrant les gènes non informatifs à l'aide du filtre SNMI ; le nombre de gènes présélectionnés est fixé à p , il s'agit d'un schéma univarié.

Phase 2 : Cette étape consiste à appliquer la recherche de voisinage variable où VNS génère des sous-ensembles de gènes candidats à partir des gènes retenus par l'étape précédente selon les concepts VNS, de plus une recherche locale stochastique (SLS) est appliquée pour aider VNS à trouver des bons résultats. La fonction objectif utilisée dans cette étape s'inspire de celle de la méthode classique MRMR. La qualité de l'ensemble de gènes généré par VNSMI est mesurée via les classifieurs 1NN et SVM (LOOCV).

La deuxième approche VNSCor est dérivée de la première en remplaçant l'information mutuelle normalisée par le coefficient de corrélation absolu et le filtre SNMI par le filtre de Corrélation de Pearson. Ces méthodes proposées enrichissent l'état de l'art de la sélection de gènes pour la classification du cancer.

FGLASSO-LR :

La sélection des gènes dans les méthodes intégrées est effectuée pendant l'étape d'entraînement du classifieur. Ces approches sont beaucoup plus efficaces en termes de calcul et moins sujettes au surapprentissage que les méthodes enveloppes avec des performances similaires.

Parmi les techniques intégrées nous trouvons les méthodes de régularisation dans lesquelles la sélection et la classification se font simultanément. Par suite, les modèles de régression logistique pénalisés (PLR) trouvent leur application dans le problème de sélection de gènes pour la classification dans les puces à ADN. Dans le but d'améliorer ces méthodes nous formulons le problème de sélection de sous-ensembles de gènes correspondant comme un problème de régression logistique (maximum de log-vraisemblance) associé à un régularisateur Fused LASSO généralisé, cette dernière pénalité n'est jamais utilisée dans ce type de problèmes.

Le Fused LASSO généralisé se compose de deux pénalités, la première assure la sélection des gènes pertinents tandis que la deuxième sélectionne les gènes voisins qui sont très pertinentes pour la cible (classe) améliorant ainsi le compromis entre la pertinence individuelle de chaque gène et la redondance entre les gènes.

La procédure générale de notre approche peut être décrite par un processus séquentiel en deux étapes qui utilise des techniques complémentaires pour réduire progressivement l'espace de recherche et sélectionner un sous-ensemble pertinent de gènes.

Étape 1 : C'est une étape de pré-traitement qui a pour objectif de filtrer les gènes non perti-

nents, par exemple, les gènes dont les niveaux d'expression sont uniformes quelle que soit la classe. De plus, il y a des milliers de gènes, il est alors inefficace d'adopter un algorithme de nature non-déterministe (comme l'algorithme sous-gradient) directement dans un ensemble de données de biopuces sans filtration. La sortie de ce pré-traitement est un ensemble de gènes classés par ordre de pertinence selon le score de corrélation absolue entre les gènes et la classe cible, Il s'agit d'une étape préliminaire pour réduire la taille des données des puces à ADN, dans laquelle le filtre de corrélation de Pearson sélectionne les 200 gènes les mieux classés.

Étape 2 : Cette étape se résume en une méthode intégrée (embedded), qui s'applique sur les gènes retenus par le filtrage précédent. Dans cette étape la sélection est effectuée durant la phase de construction du classifieur de régression logistique. La construction du modèle de classification consiste à trouver le $\hat{\beta}_{\text{FGLASSO}, \lambda_1, \lambda_2}$ qui résout le problème de minimisation (4.18) et pour se faire, l'algorithme de sous-gradient est appliqué et étudié pour trouver des meilleurs résultats.

L'apparition de la valeur absolue dans la pénalité de type Fused LASSO généralisé, rend difficile la résolution du problème de minimisation associé à cause de la non différentiabilité de la fonction objectif en zéro, ceci complique alors les calculs puisqu'il n'existe pas de solution analytique de ce dernier. Dans ce contexte, nous présentons un algorithme basée sur la notion de sous-différentiel pour résoudre le problème de minimisation associé, dans la suite nous prouvons la convergence de ce dernier algorithme vers une solution optimale à notre problème. En effet, nous montrons que la fonction objectif est convexe, lipschitzienne et admet un minimum global, finalement nous analysons numériquement cette méthode en effectuant une étude paramétrique des paramètres de cette méthode.

Dans l'ensemble, les résultats ont démontré le fait que le FGLASSO-LR est une méthode très compétitive pour analyser avec précision les données des puces à ADN de haute dimension pour la classification du cancer. Pour une utilisation pratique des résultats, le FGLASSO-LR peut être appliqué directement à d'autres types de données de classification de grande dimension liées au domaine médical.

En conclusion, nous avons proposé plusieurs méthodes de sélection de gènes, des méthodes wrappers (BPSO et ACO), des méthodes filtres (VNS et MWIS) et une méthode intégrée (FGLASSO-LR). Les algorithmes présentés sont simples et efficaces pour sélectionner des gènes pertinents aux cancers avec une meilleure performance. En résumé, cette recherche apporte plusieurs contributions nouvelles à l'identification des gènes à partir de données de puces à ADN.

Organisation de la thèse

Cette thèse est principalement concentrée sur l'analyse des données de puces à ADN en combinant des outils statistiques et d'optimisation avancée. Cela commence par le chapitre 1, qui couvre un aperçu sur la technologie des puces à ADN et les méthodes de classification supervisée. Ce chapitre aborde également les bases des arbres de décision (c4.5). Une explication détaillée de plusieurs modèles de classification tels que les séparateurs à vaste marges (SVM), les K plus proches voisins (KNN) et la régression logistique. Nous avons décrit par la suite certaines méthodes permet-

tant d'évaluer les performances d'un classifieur. Dans le chapitre 2 nous introduisons le problème de la sélection des gènes dans les puces à ADN ainsi qu'une revue de littérature sur les travaux antérieurs. Les contributions majeures de la thèse font l'objet des chapitres 3 et 4. Enfin, nous concluons notre thèse avec un rappel sur les principaux résultats obtenus et quelques perspectives.

Technologie des puces à ADN et Classification

You can have data without information,
but you cannot have information without
data.

Daniel Keys Moran

1.1 Technologie des puces à ADN

Ces dernières années, la technologie des puces à ADN est devenue l'une des avancées les plus importantes dans la collecte de données sur les systèmes biologiques, cette technologie a permis aux chercheurs de mesurer simultanément les niveaux d'expressions des dizaines de milliers de gènes dans une seule expérience, ceci dans différentes conditions et au fil du temps, elle a été largement utilisée dans la recherche biomédicale. Depuis leur introduction au début des années 90, de nombreuses applications ont été trouvées, telles que la classification des tumeurs, l'attribution de fonctions à des gènes jusque-là inconnus, le regroupement de gènes en voies fonctionnelles, etc. Quant à la recherche sur le cancer, les technologies des puces à ADN sont souvent utilisées pour identifier les gènes pouvant avoir un rôle fonctionnel déterminant dans la croissance tumorale. Cela facilitera la conception de médicaments efficaces pour combattre le cancer. En outre, l'identification de gènes importants aide à développer des systèmes informatiques de prédiction de diagnostic.

Une puce à ADN est constituée de fragments d'ADN immobilisés sur un support solide, de manière ordonnée. Chaque emplacement de séquence est soigneusement repéré. Un emplacement est souvent appelé spot ou sonde.

L'hybridation de la puce avec un échantillon biologique marqué par une substance radioactive ou fluorescente permet de quantifier l'ensemble de cibles qu'il contient : l'intensité du signal émis est proportionnelle à la quantité de gènes cibles qu'il contient.

Grâce à la richesse des données d'expression génique provenant des puces à ADN (telles que les puces à oligonucléotides à haute densité et les puces à ADN), des techniques de prédiction, de classification et de regroupement sont utilisées pour l'analyse et l'interprétation des données.

Certaines applications concernent, par exemple, la classification moléculaire de la leucémie aiguë [78], l'analyse groupée des tissus tumoraux et normaux du côlon [16], le groupage et la classification des cellules cancéreuses humaines [160]. DL-BCL [15], les cellules épithéliales mammaires humaines et le cancer du sein [150, 151] ainsi que le mélanome cancéreux cutané [27]. Ces techniques ont également permis d'identifier des sous-types de cancer jusqu'alors inconnus [78, 15, 27, 151].

Les données de puces à ADN se présentent de la manière suivante :

Chaque valeur de x_{ij} est la mesure du niveau d'expression du $j^{\text{ème}}$ gène dans le $i^{\text{ème}}$ échantillon, cette matrice contient des valeurs réelles, et il n'y a pas de valeurs manquantes.

Où N est le nombre de gènes et M le nombre d'échantillons.

Table. 1.1: Exemple d'une matrice associée à une puce à ADN

Gène _{id}	Gène ₁	Gène ₂	...	Gène _N	Class label
Échantillon ₁	x_{11}	x_{12}	...	x_{1N}	y_1
Échantillon ₂	x_{21}	x_{22}	...	x_{2N}	y_2
Échantillon ₃	x_{31}	x_{32}	...	x_{3N}	y_3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
Échantillon _M	x_{M1}	x_{M2}	...	x_{MN}	y_M

Comme les informations sur les classes des échantillons sont déjà disponibles, l'analyse des puces à ADN pour la classification des cancers est formulée dans nos travaux comme un problème de classification supervisée.

Dans la classification du cancer à l'aide des puces à ADN, nous devons diagnostiquer si un échantillon provient d'un tissu cancéreux ou bénin, ou différencier les différents types de tumeurs cancéreuses. Les ensembles de données d'expression génétique de ces puces posent des défis considérables aux techniques informatiques compte tenu de leur subtilité et leur complexité biologique. En effet, le coût élevé de la préparation des puces à ADN ne permet de rassembler qu'un petit nombre d'échantillons lors d'une expérience. De plus, l'analyse d'ensembles de données de grande dimension reste l'un des obstacles majeurs au quels se heurtent les techniques d'apprentissage statistique, principalement en raison de l'ampleur du nombre de gènes (N) impliqués, par rapport au nombre d'échantillons (M) : c'est-à-dire $N \gg M$. Une revue détaillée des propriétés des ensembles de données de puces à ADN et la manière de remédier aux difficultés qu'elles relèvent est présentée dans [189].

La classification est l'une des techniques les plus anciennes d'analyse et de traitement de données. Plusieurs définitions ont été proposées par des spécialistes du domaine, citons par exemple :

- "La classification est le fait de mettre en évidence des relations entre des objets, et entre ces derniers et leurs paramètres"[127].
- "La classification consiste à affecter des objets, des candidats, des actions potentielles à des catégories ou des classes prédéfinies"[86].

- "La classification est l'action de regrouper en différentes catégories des objets ayant certains points communs ou faisant partie d'un même concept, sans avoir connaissance de la forme ni de la nature des classes au préalable, nous parlons alors de problème d'apprentissage non supervisé ou de classification automatique, ou l'action d'affecter des objets à des classes pré-définies, on parle dans ce cas d'apprentissage supervisé ou de problème d'affectation"[131].
- "La classification cherche à mettre en évidence les dépendances implicites qui existent entre les objets, les classes entre elles, les classes et les instances. La classification recouvre les processus de reconnaissance de la classe d'un objet, et l'insertion éventuelle d'une classe dans une hiérarchie. Ce mode de raisonnement permet de reconnaître un objet en identifiant ses caractéristiques, relativement à la hiérarchie étudiée. La classification fait intervenir un processus de décision d'appartenance" [30].

Les modèles de classification peuvent être divisés en deux catégories : supervisés et non supervisés (supervised classification et unsupervised classification en anglais). Dans l'approche supervisée, le processus de classification consiste à placer des objets dans une catégorie préexistante, dans laquelle le modèle de classification a été préalablement appris à partir des données accompagnant les informations de classe. D'autre part, les approches non supervisées visent à développer un nouvel ensemble de catégories (classes) tout en plaçant les échantillons dans les classes, par exemple, pour différents stades du cancer.

Les méthodes supervisées utilisent la connaissance à priori sur l'appartenance d'un élément à une classe pour construire un système de reconnaissance de ces classes. Dans la plupart des problèmes bioinformatiques, tels que la classification des cancers, l'information sur la catégorie d'une lignée cellulaire ou d'un tissu particulier existe. Ceci dit, nous nous intéressons particulièrement dans cette thèse aux méthodes de classification supervisée.

1.2 Classification supervisée

Nous rappelons tout d'abord une définition du problème de classification supervisée. L'objectif de la classification est d'identifier les classes auxquelles appartiennent des objets à partir de leurs caractéristiques (attributs) descriptives.

Le fonctionnement de la classification supervisée se décompose en deux phases [167, 130] :

La première est celle de l'apprentissage, tout ce qui est appris par l'algorithme est représenté sous forme des règles de classification que l'on appelle le modèle d'apprentissage.

La seconde phase est celle de la classification proprement dite, dans laquelle les données tests vont être utilisées pour estimer l'exactitude des règles de classification générées pendant la première phase. Si l'exactitude du modèle est considérée comme acceptable, la règle pourra être appliquée à des nouvelles données.

L'objectif de la classification supervisée est de désigner un prédicteur de classe pour distinguer deux ou plusieurs classes d'échantillons (par exemple sains ou cancéreux). Le prédicteur de classe est conçu sur la base des données actuellement disponibles de différentes classes de diagnostic, qui font référence à des échantillons d'apprentissage (training).

1.2.1 Formalisation mathématique du problème de classification supervisée et Notations :

Les données d'entrée consistent en M observations (échantillons), chacune décrite par un ensemble de N caractéristiques (gènes ou attributs). Dans la suite, nous noterons X un ensemble de données (voir la table 1.1) de M échantillons $X = (x_1; x_2; \dots; x_M)$, $x_i \in \mathbb{R}^N$. Nous utilisons $\{g_1; g_2; \dots; g_N\}$, $g_i \in \mathbb{R}^M$ pour désigner les vecteurs de N gènes.

Les données de sortie ou la réponse sont décrites comme un vecteur $\mathbf{Y} = (y_i)_{i=1 \dots M}$. Dans un cadre de classification, la réponse se compose de C classes : $\mathbf{Y} \in \{1 \dots C\}^M$. Dans le cas particulier d'une classification binaire, $\mathbf{Y} \in \{-1, +1\}^M$ ou $\mathbf{Y} \in \{0, 1\}^M$, par exemple, prédire l'issue du cancer du sein est un problème de classification binaire, avec y_i spécifiant le résultat ($y_i = +1$ en cas d'événement métastatique, et -1 sinon).

Soit un ensemble d'exemples de M données étiquetées :

$$\mathbb{S} = \{(x_1, y_1), \dots, (x_M, y_M)\}.$$

Chaque donnée x_i est caractérisée par N attributs et par sa classe y_i , nous cherchons une hypothèse h telle que :

— h satisfait les critères des échantillons :

$$\forall i \in \{1, \dots, M\} \quad h(x_i) = y_i$$

— h possède de bonnes propriétés de généralisation.

Le problème de la classification consiste donc à se baser sur l'ensemble d'exemples pour prédire la classe de toute nouvelle donnée $x \in \mathbb{R}^N$.

1.2.2 Techniques de la classification supervisée

Les problèmes qui nous intéressent dans le cadre de cette thèse font appel à des méthodes de classification supervisée : les données que nous utilisons disposent toutes d'une classe y connue à l'avance, telle que le type de tumeur sur les données puces. Ce type de problème est fréquent en biologie, il est souvent possible d'obtenir l'information sur la classe (par exemple, examen histologique pour déterminer le type de cancer). Dans ce qui suit, nous présentons les principales méthodes de classification utilisées dans le cadre des données de biopuces (puces à ADN).

1.2.2.1 K Plus proches voisins (KNN)

La méthode des K plus proches voisins (K-PPV) (KNN pour K-nearest neighbors) [44] est une méthode basée sur la notion de proximité (voisinage) entre exemples (échantillons), et sur l'idée de raisonner à partir des cas similaires pour prendre une décision. Face à un nouvel échantillon à classer, la méthode des KNN recherche les K échantillons les plus proches dans l'ensemble d'apprentissage ; et fait un vote majoritaire sur leurs classes pour déterminer la classe du nouvel échantillon (figure.1.1). Le seul paramètre du modèle est K , nombre de voisins à considérer, si $K = 1$, alors l'exemple du test est simplement affecté à la classe de son plus proche voisin.

Pour trouver les plus proches voisins d'un nouvel exemple, une métrique de distance (similarité) est utilisée (par exemple, la distance euclidienne). Les éléments clés de l'approche KNN, dont dépendent principalement ses performances, sont la métrique de distance et la valeur choisie de K [75].

En ce qui concerne la métrique pour mesurer la proximité entre l'exemple à classer $\mathbf{x}$ et chacun des exemples de l'ensemble d'apprentissage :

Notons par $x_p = (x_{p1}, x_{p2}, \dots, x_{pN})$ l'exemple de l'entité p , avec N le nombre d'attributs et par x_p et x_q deux exemples à comparer.

Les distances suivantes sont usuellement employées par les classificateurs KNN :

Distance Euclidienne :

$$D(x_p, x_q) = \sqrt{\sum_{i=1}^N (x_{pi} - x_{qi})^2}.$$

Distance de Manhattan :

$$D(x_p, x_q) = \sum_{i=1}^N |x_{pi} - x_{qi}|.$$

Distance de Minkowski pour tout r supérieur ou égal à 1 :

$$D(x_p, x_q) = \sqrt[r]{\sum_{i=1}^N |x_{pi} - x_{qi}|^r}.$$

Distance de Tchebychev :

$$D(x_p, x_q) = \sup_{1 \leq i \leq N} |x_{pi} - x_{qi}|.$$

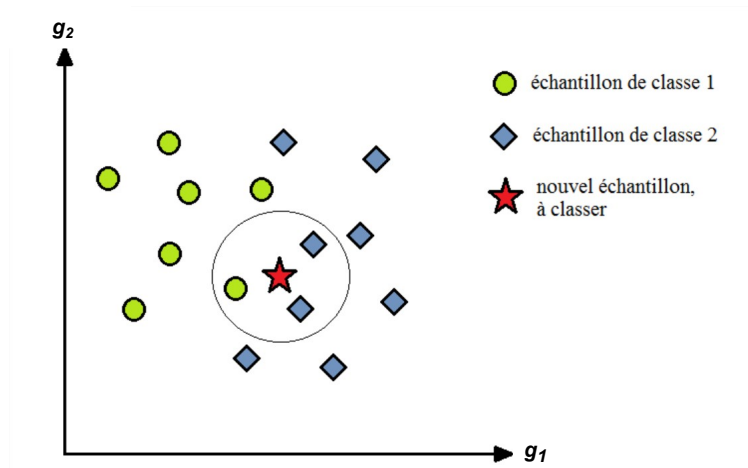


figure. 1.1: Méthode des K plus proches voisins. Le nouvel échantillon (étoile) est affecté à la classe majoritaire parmi ses K (ici K=3) plus proches voisins, ici la classe 2 [49]

À partir d'un ensemble de données, un sous-ensemble d'observations relevant d'un domaine particulier par rapport à l'espace de caractéristiques considéré est illustré à la figure.1.1. Deux

caractéristiques, $'g'_1$ et $'g'_2$, sont représentées respectivement sur les axes horizontal et vertical. Les observations d'apprentissage appartiennent à l'une des deux classes 1 et 2. L'observation de test est assignée à la classe majoritaire de ses voisins, les distances euclidiennes sont mesurées à partir de toutes les observations d'apprentissage. Ensuite, ses K voisins les plus proches sont identifiés en fonction de la distance euclidienne ($K = 3$ dans la figure.1.1). L'observation est assignée à la classe populaire dans ses voisins qui est la classe 2.

1.2.2.2 Arbres de Décisions

Les arbres de décision sont parmi les méthodes les plus connues en classification. Le principe des arbres de décision est de réaliser la classification d'un exemple par une suite de tests sur les attributs qui le décrivent (figure.1.2).

Dans toutes les méthodes de classification basées sur les arbres de décisions, nous trouvons les trois opérateurs suivants :

- Décider si un nœud est terminal.
- Si un nœud n'est pas terminal, sélectionner un test à lui associer.
- Si un nœud est terminal, lui affecter une classe.

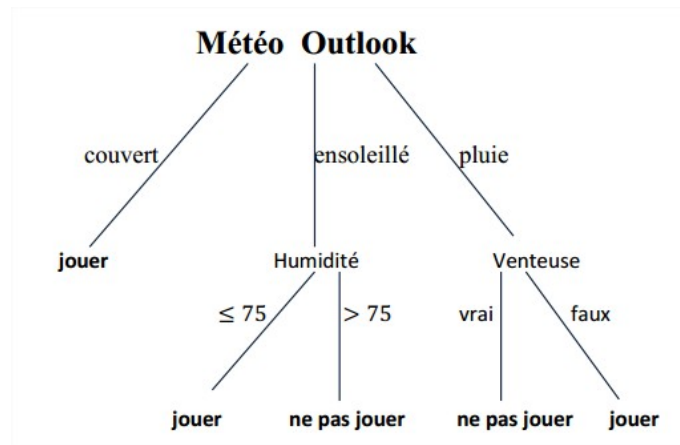


figure. 1.2: Exemple d'un arbre de décisions.

Le schéma général des algorithmes d'apprentissage par les arbres de décisions est le suivant :

Algorithme 1 Algorithme d'apprentissage générique

```

1: Entrées : Ensemble d'apprentissage  $\mathbb{S}$ ;
   Début
2:   Initialiser l'arbre à vide ; la racine est le nœud courant
3:   répéter
4:
5:   Décider si le nœud courant est terminal
6:
7:   if le nœud est terminal then
8:     Lui affecter une classe
9:
10:  else Sélectionner un test et créer autant de nouveaux nœuds fils qu'il y a de réponses
    possibles au test.
11:  end if
12:  Passer au nœud suivant non exploré s'il en existe
13:
14:  Until : obtenir un arbre de décision
   Fin

```

Les principaux algorithmes de construction d'un arbre de décision sont : C4.5 [159] et les arbres de régression. Dans ces algorithmes, à chaque étape de création d'un nœud, un critère de séparation entre les classes est utilisé pour décider quel attribut est le plus pertinent pour la classification

1.2.2.3 C4.5

C'est un algorithme basé essentiellement sur la notion de **gain** qui favorise les attributs contenant les informations nécessaires.

Soient :

- $\mathbb{S}$ un ensemble d'apprentissage.
- g_j un attribut de $\mathbb{S}$.
- $\{C_1, C_2, \dots, C_k\}$: une partition sur g_j .
- c le nombre de classe relatif à $\mathbb{S}$.
- $|\cdot|$ le cardinal de l'ensemble.

On définit le gain associé à l'attribut g_j de la manière suivante :

$$Gain(S, g_j) = Entropie(\mathbb{S}) - \sum_{i=1}^k \frac{|C_i|}{|\mathbb{S}|} Entropie(C_i)$$

Avec : $Entropie(\mathbb{S}) = - \sum_{i=1}^c P_i \log_2(P_i)$, P_i le pourcentage des échantillons de la classe i .

1.2.2.4 Machine à vecteurs de support (SVM)

Les machines à vecteurs de support (ou séparateurs à vaste marge, SVM pour support vector machine) [43] sont basées sur la recherche d'un hyperplan qui sépare les deux classes avec une marge (distance entre l'hyperplan et les observations les plus proches) maximale. Les SVM ont été largement utilisés dans la reconnaissance des formes, la classification, la régression et l'estimation de densité.

1.2.2.4.1 Classifieur SVM linéaire :

Cas linéairement séparable :

Soit $\mathbb{S} = \{(x_1, y_1), \dots, (x_M, y_M)\}$, avec $x_i \in \mathbb{R}^N$ et $y_i \in \{1, -1\}$ un ensemble d'exemples d'apprentissage.

Nous supposons dans un premier temps qu'il existe un hyperplan affine séparateur permettant de distinguer les exemples de la classe 1 des exemples de la classe -1 . La recherche d'un hyperplan séparateur revient à trouver une fonction de décision de la forme $H(x) = w^T x + b$, où w est le vecteur normal à l'hyperplan affine d'équation $H(x) = 0$.

Cette fonction nous permet de classer les nouveaux exemples :

$$Classe(x) = \text{signe}(H(x)).$$

Les points de l'échantillon d'apprentissage sont correctement classés par l'hyperplan affine si, pour $i = 1 \dots M$

$$\begin{cases} w^T x_i + b < 0 & \text{si } y_i = -1 \\ w^T x_i + b > 0 & \text{si } y_i = 1 \end{cases}.$$

Ces deux contraintes se résument comme suit :

$$\text{pour } i = 1 \dots M, \quad y_i(w^T x_i + b) > 0. \quad (1.1)$$

Si les exemples sont linéairement séparables, il sera possible de trouver plusieurs hyperplans satisfaisant (1.1) (figure.1.3). Nous pouvons alors chercher l'hyperplan optimal qui va être le plus éloigné des exemples positifs et des exemples négatifs.

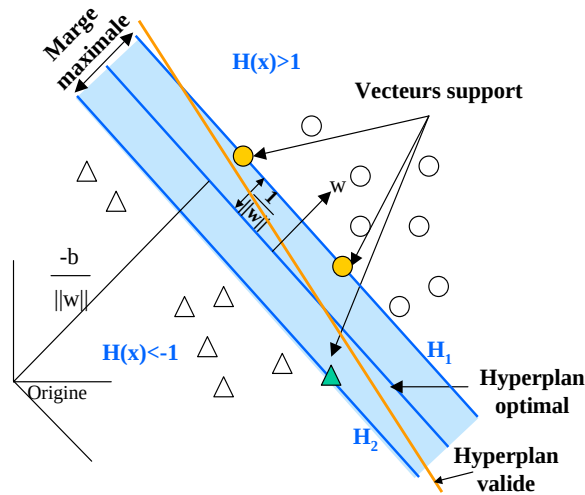


figure. 1.3: Hyperplan optimal de séparation [87].

La distance de l'hyperplan à chacune des classes est appelée marge. Nous cherchons donc l'hyperplan qui assure la plus grande marge.

Nous pouvons normaliser les paramètres w et b pour que les points positifs les plus proches de l'hyperplan (hyperplan H_1) vérifient l'équation $w^T x + b = 1$ et de même les points négatifs les plus proches de l'hyperplan (hyperplan H_2) doivent satisfaire la condition $w^T x + b = -1$.

La distance entre H_1 (resp H_2) et l'hyperplan optimal vaut alors $\frac{1}{\|w\|}$.

En effet :

Rappelons que la distance d'un point x à l'hyperplan H est définie par : $\frac{|w^T x + b|}{\|w\|}$.

Soit $x \in H_1$ donc :

$$d(x, H) = \frac{|w^T x + b|}{\|w\|} = \frac{|w^T x + b - 1 + 1|}{\|w\|}.$$

Comme $x \in H_1$ on aura : $w^T x + b - 1 = 0$.

Par conséquent : $d(x, H) = \frac{1}{\|w\|}$ ceci pour tout $x \in H_1$.

D'où $d(H_1, H) = \frac{1}{\|w\|}$. La marge est ainsi égale à $M = \frac{2}{\|w\|}$ que nous cherchons à maximiser.

Or la maximisation de M est équivalente à la minimisation de $\frac{\|w\|}{2}$.

La détermination de l'hyperplan optimal est donc équivalente à la résolution du problème d'optimisation suivant :

$$\min_{w, b} \frac{\|w\|}{2}$$

sous les contraintes :

$$y_i(w^T x_i + b) \geq 1, \quad i = 1 \dots M.$$

Cette écriture du problème, appelée formulation primale implique le réglage de $N + 1$ paramètres, N étant le nombre d'attributs. Cela est possible avec des méthodes de programmation quadratique pour des valeurs de N assez petites, mais devient inenvisageable pour des valeurs de N assez grandes comme dans notre cas. Heureusement, il existe une transformation de ce problème dans une formulation duale [32], que nous pouvons résoudre en pratique.

Cas non linéairement séparable :

Nous considérons maintenant le cas où les exemples ne sont pas linéairement séparables, dans ce cas, l'hyperplan optimal est celui qui satisfait les contraintes suivantes : d'une part la distance entre les vecteurs support et l'hyperplan optimal doit être maximale, d'autre part la distance entre les exemples mal classés et l'hyperplan optimal doit être minimale. Pour cela, une technique dite des variables d'écart (slack variables) permet de chercher un hyperplan séparateur faisant le moins d'erreurs possible [43] (figure.1.4).

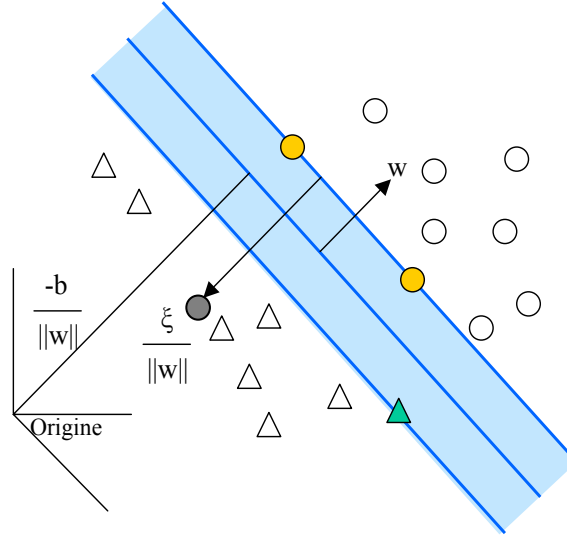


figure. 1.4: Hyperplans séparateurs dans le cas des exemples non linéairement séparables [87].

Pour formuler tout cela, nous utilisons les variables d'écart ξ_i . L'objectif est de résoudre le problème d'optimisation suivant :

$$\min_{w, b, \xi_i} \frac{\|w\|}{2} + C \sum_{i=1}^M \xi_i$$

sous les contraintes :

$$y_i(w^T x_i + b) \geq 1 - \xi_i, \quad i = 1 \dots M$$

Où, ξ_i est la variable de relâchement d'une contrainte et C un coefficient de pénalisation du relâchement.

1.2.2.4.2 Multiclassification via SVM

Les Séparateurs à vaste marge ont été développés pour traiter des problèmes binaires mais ils peuvent être adaptés pour analyser les problèmes multi-classes [80].

Stratégie un contre tous :

L'idée consiste simplement à transformer le problème à c classes en c classifieurs binaires (figure.1.5).

Le classement est donné par le classifieur qui répond le mieux.

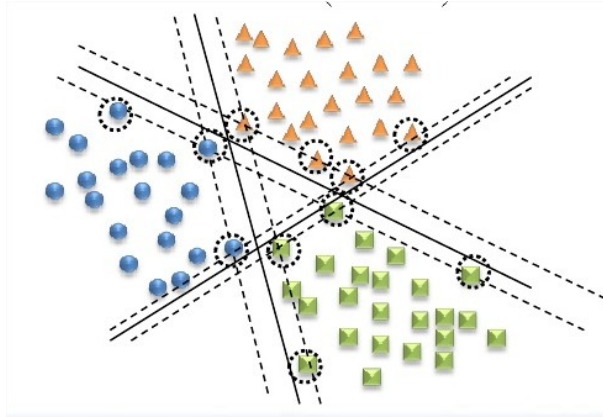


figure. 1.5: Stratégie un contre tous.

Stratégie un contre un :

Cette stratégie consiste à transformer le problème en $\frac{c(c-1)}{2}$ classifieurs binaires, puis à déterminer la classe gagnante.

Le classement est donné par le vote majoritaire ou un graphe acyclique de décision (figure.1.6).

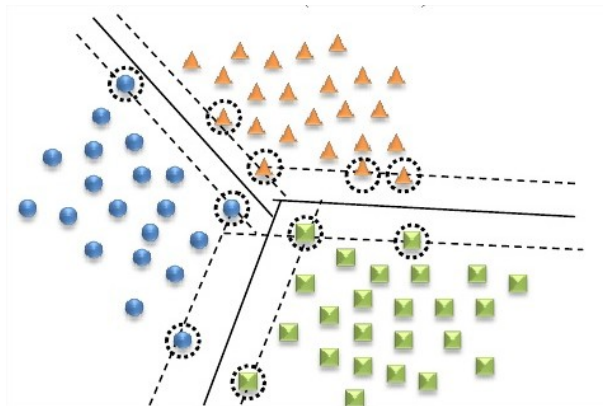


figure. 1.6: Stratégie un contre un.

1.2.2.5 Régression logistique pour la classification

Les modèles linéaires sont composés d'une ou plusieurs variables indépendantes qui décrivent une relation avec une variable de réponse dépendante. La mise en correspondance de caractéristiques (gènes) d'entrée quantitatives avec une variable cible (classe) que l'on tente de prédire, comme des données financières, biologiques ou sociologiques, est appelée apprentissage supervisé dans la terminologie de l'apprentissage automatique si les étiquettes sont connues. L'un des modèles statistiques linéaires les plus couramment utilisés pour l'analyse discriminante est la régression logistique.

En régression logistique binaire, il est usuel de considérer les valeurs de la variable à expliquer y comme appartenant à l'ensemble $\{0, 1\}$, il s'agit d'une classification binaire.

Notons $\Pi_i = \mathbb{P}(y_i = 1/x_i = x_{i1}, \dots, x_{iN})$ la probabilité que la classe de x_i soit 1, pour tout échantillon x_i , $i = 1, \dots, M$, de la matrice des jeux de données.

Nous pouvons déduire alors que $\mathbb{P}(y_i = 0/x_i = x_{i1}, \dots, x_{iN}) = 1 - \Pi_i$. Le modèle de la régression logistique s'écrit [67] :

$$\Pi_i = \frac{\exp(\beta_0 + \beta_1 x_{i1} + \dots + \beta_N x_{iN})}{1 + \exp(\beta_0 + \beta_1 x_{i1} + \dots + \beta_N x_{iN})} = \frac{e^{\beta_0 + \sum_{j=1}^N \beta_j x_{ij}}}{1 + e^{\beta_0 + \sum_{j=1}^N \beta_j x_{ij}}}$$

où $\beta = (\beta_0, \beta_1, \dots, \beta_N)^t$ désigne le vecteur de coefficients de régression incluant une constante β_0 . Ce vecteur sera estimé à partir des données d'apprentissage. Sans perte de généralité, nous supposons que les gènes sont centrés, $\sum_{i=1}^M x_{ij} = 0$ et $\left(n^{-1}\right) \sum_{i=1}^M x_{ij}^2 = 1$, $\forall j \in \{1, 2, \dots, N\}$.

Pour des raisons de simplification, considérons les notations suivantes :

Pour $i = 1, \dots, M$ soient :

— $x'_i = (1, x_i) = (1, x_{i1}, \dots, x_{iN})$, ou tout simplement pour tout vecteur $x = (x_1, \dots, x_N)$ de $\mathbb{R}^N$ nous notons $x' = (1, x) = (1, x_1, \dots, x_N)$

— $x'_i \beta = \beta_0 + \sum_{j=1}^N \beta_j x_{ij}$ il s'agit du produit scalaire usuel.

Nous avons alors, pour $i = 1, \dots, M$:

$$\Pi_i = \frac{e^{x'_i \beta}}{1 + e^{x'_i \beta}} \quad \text{et} \quad \ln \left(\frac{\Pi_i}{1 - \Pi_i} \right) = x'_i \beta$$

La probabilité d'observer la réponse y_i pour un échantillon x_i s'écrit de manière plus compacte :

$$\mathbb{P}(y_i/x_i) = \Pi_i^{y_i} (1 - \Pi_i)^{1-y_i}$$

Notons f la fonction "sigmoïde" ou logistique (figure.1.7) utilisée dans la régression logistique définie par : $f(z) = \frac{1}{1 + e^{-z}}$ pour tout réel z , cette dernière est à valeur dans $]0, 1[$, nous pouvons remarquer alors que $\Pi_i = f(x'_i \beta)$.

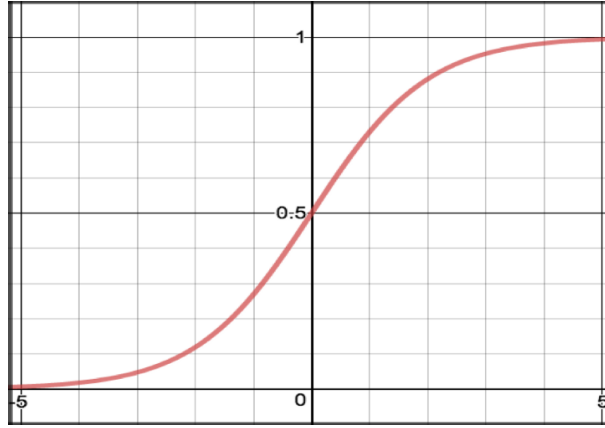


figure. 1.7: Graphique de la fonction sigmoïde

La règle de classification pour un nouvel exemple x via la régression logistique est donc définie par :

$$Classe(x) = \begin{cases} 0 & \text{si } f(x'\beta) < 0.5 \\ 1 & \text{si } f(x'\beta) \geq 0.5 \end{cases}$$

Parmi les méthodes visant à estimer les coefficients de β nous citons la méthode statistique appelée " maximum de vraisemblance " (maximum likelihood estimation) qui permet de déterminer les valeurs du paramètre β du modèle qui rendent la valeur suivante maximale

$$L(\beta) = \prod_{i=1}^M \Pi_i^{y_i} (1 - \Pi_i)^{1-y_i}$$

Ceci est équivalent à trouver β dans $\mathbb{R}^{N+1}$ qui maximise la fonction de log-vraisemblance.

$$\ln(L(\beta)) = \sum_{i=1}^M y_i \ln(\Pi_i) + (1 - y_i) \ln(1 - \Pi_i)$$

Maximiser $\ln(L(\beta))$ revient à minimiser $-\ln(L(\beta))$, notons alors $g(\beta) = -\ln(L(\beta))$ pour tous β de $\mathbb{R}^{N+1}$.

Après des calculs élémentaires nous aboutissons à :

$$g(\beta) = - \sum_{i=1}^M \left(y_i x'_i \beta - \ln(1 + e^{x'_i \beta}) \right)$$

L'étude de cette dernière fonction sera détaillée dans le dernier chapitre.

Les différentes techniques de classification que nous venons de décrire ont été utilisées dans différents travaux concernant le diagnostic à partir de données de puces à ADN.

1.2.3 Estimation des performances du classifieur et sur-apprentissage

Comme nous l'avons vu, dans une tâche de classification, il existe deux grandes étapes à distinguer : l'apprentissage (entraînement) et le test. Dans la première phase, le modèle de classification est construit à l'aide des données d'apprentissage et dans la seconde phase, le modèle est évalué à l'aide de données de test. Il est important de noter que les données de test ne doivent pas être utilisées dans la phase d'apprentissage, sinon le résultat de la validation sera optimiste. L'objectif principal de la construction d'un modèle de classification n'est pas seulement d'avoir de bonnes performances sur les données d'apprentissage, mais aussi d'être capable de généraliser ce modèle pour obtenir de bonnes performances sur les données de test et d'autres données non vues [161]. Le surapprentissage (overfitting) est un phénomène qui se produit lorsqu'un modèle est trop complexe (trop de paramètres) et que ce dernier mémorise les données d'apprentissage plutôt que d'apprendre à généraliser à partir de ces données. autrement dit, lorsqu'il apprend à partir des données mais aussi à partir de patterns (schémas, structures) qui ne sont pas liés au problème, comme du bruit. Par conséquent, si un surapprentissage a lieu, le modèle fonctionne très bien sur les données d'entraînement, cependant, ce modèle aurait un faible pouvoir de prédiction lorsqu'il est appliqué aux données du test en raison du manque de généralisation [118].

Le surapprentissage peut être évité si des méthodes telles que la validation non-croisée (holdout method), la validation croisée à k blocs (k -fold cross-validation) ou la validation croisée d'un contre tous, (leave-one-out cross-validation (LOOCV)) sont mises en œuvre dans le modèle. En général, dans une tâche de validation croisée, les données sont divisées en deux parties, l'apprentissage étant effectué sur une partie et la validation (test) sur l'autre. Ceci dit : le principe de la validation croisée met l'accent sur la séparation d'une partie des données de l'étape d'apprentissage pour valider la performance du modèle sur cette partie de données qui n'est pas vue par le modèle auparavant. La validation croisée est largement acceptée dans la société de l'apprentissage automatique où elle est utilisée pour la sélection de modèles [204].

Dans la validation non-croisée (holdout method), les données sont divisées en deux parties (par exemple, 70 % – 30%). L'apprentissage est généralement effectué sur la partie la plus élevée des données (70% de données), puis le modèle est évalué sur la partie restante (30% de données). Étant donné que la division des données est un processus aléatoire, nous avons généralement tendance à répéter le fractionnement plusieurs fois et, quitte à répéter plusieurs fois l'apprentissage et l'évaluation. Ensuite, il s'agit de rapporter l'exactitude finale comme étant la moyenne des exactitudes moyennes obtenues à partir de toutes les répétitions [197].

La validation croisée à k blocs (k -fold cross-validation) est une technique couramment utilisée pour évaluer les performances de prédiction d'un classificateur. Dans cette méthode, les données sont divisées en k sous ensembles de taille égale, chaque sous ensemble est alors utilisé comme donnée du test, tandis que l'union de tous les autres sous ensembles est utilisée comme base d'apprentissage. Ce processus est répété k fois, où à chaque fois un nouveau ensemble est choisi pour la phase du test et les $k - 1$ ensembles restants pour l'apprentissage. Les tests sont donc effectués sur tous les morceaux séparément. À chaque itération, le taux d'exactitude (accuracy) de la classification est comptabilisé et le taux d'exactitude de la classification final du modèle est déterminé en faisant la moyenne des exactitudes à chaque itération.

Le taux d'exactitude se mesure par la formule suivante :

$$\text{Taux d'exactitude} = \frac{\text{Nombre d'exemples correctement identifiés}}{\text{Nombre d'exemples}}.$$

La validation croisée d'un contre tous, (leave-one-out cross-validation (LOOCV)) est un cas particulier de validation croisée à k -blocs où k est égal au nombre d'échantillons [197]

Étant donné que la classe de chaque échantillon est connue, nous pouvons également produire **la matrice de confusion**, qui recense pour chaque classe réelle ; le nombre d'échantillons que le classifieur a classé dans chaque classe ce qui permet de détecter les éventuelles confusions que pourrait faire le système. Si C est le nombre de classes, la matrice de confusion est une matrice de taille $C \times C$ où la cellule à l'intersection de la ligne i et de la colonne j est le nombre d'éléments de la classe réelle i générés par le système dans la classe j .

Remarque 1.2.1. — *Si k est petit, l'ensemble d'apprentissage sera trop faible et le système de classification ne généralisera pas assez, entraînant ainsi une augmentation du taux d'erreur. Par contre, s'il y a moins de phases d'apprentissage à réaliser l'obtention des résultats est plus rapide. Cette solution est intéressante quand nous disposons de beaucoup de mesures.*
— *Si k est grand, l'ensemble de test sera plus faible par rapport à l'ensemble d'apprentissage et nous risquons le phénomène de sur-apprentissage.*

Nous utiliserons la procédure de validation croisée dans la suite de cette thèse pour évaluer les classifieurs que nous mettrons en oeuvre sur les jeux de données des puces à ADN.

1.3 Conclusion

Dans ce chapitre nous avons introduit la technologie des puces à ADN et les méthodes de classification des biopuces. Ce chapitre aborde également les bases des arbres de décision (c4.5). Une explication détaillée de plusieurs modèles de classification tels que les séparateurs à vaste marges (SVM), les K plus proches voisins (KNN), et la régression logistique. Nous avons décrit par la suite certaines méthodes permettant d'évaluer les performances d'un classifieur.

Sélection d'attributs

Computers are able to see, hear and learn. Welcome to the future.

Dave Waters

La taille des données peut être mesurée selon deux aspects : le nombre d'attributs (gènes) et le nombre d'échantillons, ces deux dimensions peuvent prendre des valeurs très élevées, ce qui peut causer des problèmes lors de l'analyse de ces données.

La réduction de la dimensionnalité se pose alors comme une étape primordiale dans le processus de pré-traitement des données. Une méthode de réduction consiste à trouver une représentation des données initiales dans un espace plus réduit, il existe principalement deux approches différentes de réduction :

- Une réduction basée sur une **sélection de variables** (d'attributs "feature selection") qui consiste à sélectionner les attributs les plus pertinents à partir de l'ensemble de données des variables décrivant le phénomène étudié. L'idée derrière cette sélection est que seules quelques-unes de ces variables sont vraiment liées au processus de classification, toutes les autres sont redondantes et n'ont pas de réel effet explicatif [125].
- Une réduction basée sur **une transformation des données** appelée aussi une extraction de caractéristiques, celle-ci consiste à remplacer l'ensemble initial des données par un nouvel ensemble réduit, construit à partir des combinaisons des attributs initiaux [63, 62].

Du point de vue biologique, il est plus efficace de sélectionner de vrais gènes que de créer des caractéristiques artificielles avec une signification biologique incertaine. Par conséquent, une approche pertinente de sélection des caractéristiques garantit une meilleure interprétation de la relation biologique entre les gènes et le résultat clinique considéré, puis vraisemblable compréhension scientifique de la situation en question. Par suite, dans notre thèse nous nous intéressons au problème de sélection d'attributs ou tout simplement au problème de sélection de gènes dans le contexte de classification des puces à ADN. En effet, lors de l'analyse des puces à ADN, le nombre de gènes est significativement plus élevé que le nombre d'échantillons [19, 36], cela rendra difficile l'obtention d'un niveau élevé d'exactitude pour la classification [76]. Ce concept fait référence au **fléau de la dimension** (curse of dimensionality), terme introduit par Bellman et Kalaba [22, 175]

De façon très intuitive : Supposons qu'on ait n points répartis uniformément dans un hypercube de dimension d ayant un côté de taille 1. On voudrait placer un autre hypercube à l'intérieur de celui-ci de sorte qu'il couvre une fraction r des exemples. Plus on aura de dimensions, plus le côté c de l'hypercube couvrant cette fraction sera grand. En fait, il est facile de voir qu'il faudra un hypercube de côté $c = r^{1/d}$ pour couvrir une fraction r du volume de l'hypercube de côté 1. Les figures 2.1 et 2.2 illustrent ce phénomène. On remarque que lorsque la dimensionnalité augmente, une plus en plus grande proportion d'exemples sera située près des bords. En effet, dans dix dimensions, $c_{10}(0.01) = 0.63$ et $c_{10}(0.1) = 0.80$, alors que l'hypercube tout entier n'a qu'un côté de taille 1. Donc, pour couvrir 1 % ou 10 % des données, nous devons couvrir 33 % ou 80 % de la plage de chaque variable d'entrée. Réduire r n'aide pas beaucoup, car moins nous faisons la moyenne d'observations, plus la variance de notre ajustement est élevée. Ce qui est problématique puisque nous sommes censés extrapoler à partir des exemples du centre, or l'extrapolation est beaucoup moins fiable que l'interpolation.

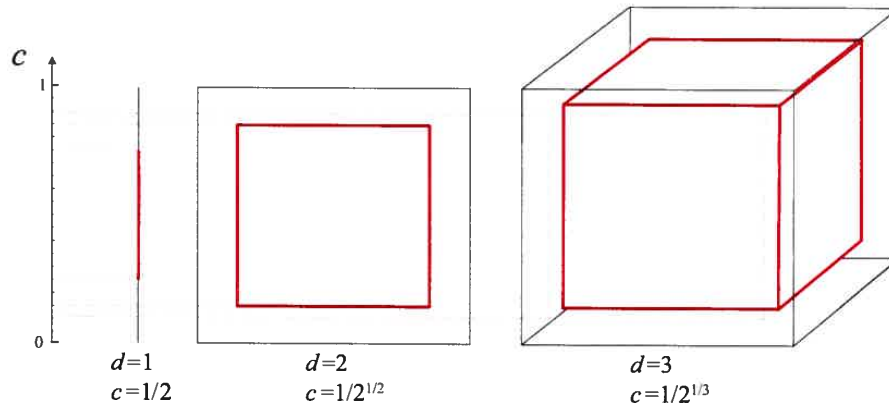


figure. 2.1: Les trois figures rouges (la ligne, le carré et le cube) ont une longueur, une surface ou un volume correspondant à la moitié des figures noires qui les contiennent. Plus la dimensionnalité augmente, plus le côté de la figure rouge doit grandir pour que la proportion reste 1/2.[2]

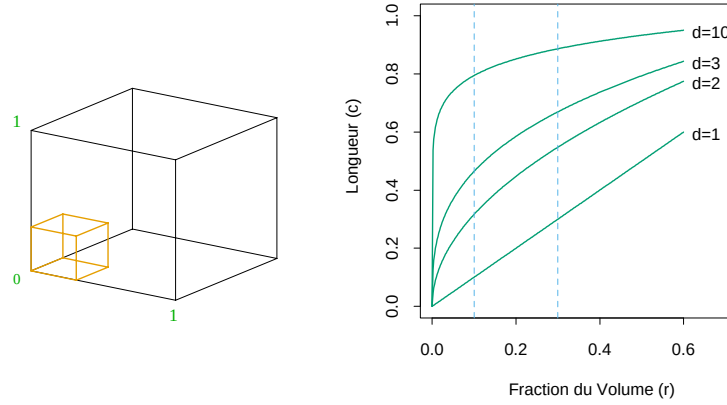


figure. 2.2: Le fléau de la dimensionnalité est bien illustré par un voisinage hyper-cubique pour des données uniformes dans un cube unité. La figure de droite montre la longueur de côté de l'hypercube nécessaire pour couvrir une fraction r du volume des données, pour différentes dimensions p . En dix dimensions, nous devons couvrir 80 % de la plage de chaque coordonnée afin de capturer 10% des données.[1].

Pour un problème de classification ou régression où n est le nombre minimal d'exemples nécessaires dans une dimension, étendre le problème à p dimensions nécessiterait n^p exemples. Cependant, nous ne nous donnons pas plus d'exemples et nous devons généralement continuer avec une complexité $\mathcal{O}(n)$ entre eux. La principale - et pire - conséquence de ceci est le sur-apprentissage (voir Section « Estimation des performances du classifieur et Sur-apprentissage ») : parce que nous avons beaucoup moins de points que ce qui est demandé. Il est facile de trouver des modèles reliant les données à la réponse, mais il y a de fortes chances qu'ils soient accidentels et donc pas du tout généralisables.

En fait, nous supposons que les données se comportent comme le montre la figure.2.3, c'est-à-dire qu'il existe un nombre optimal de caractéristiques maximisant la précision de classification.

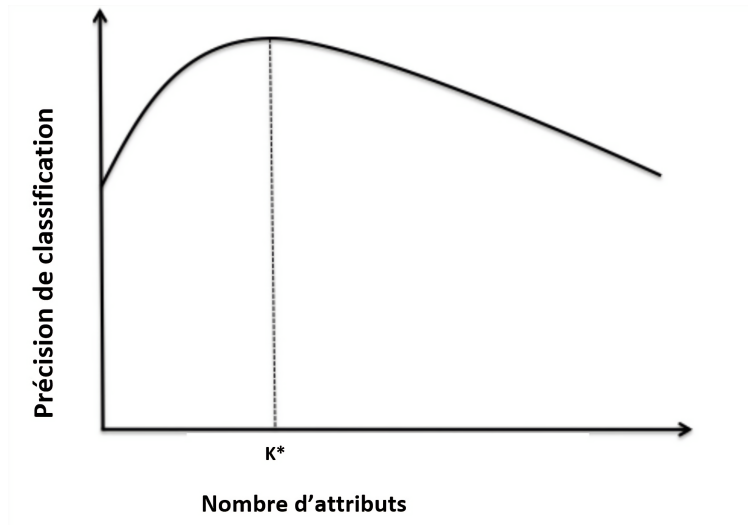


figure. 2.3: La sélection des caractéristiques et le fléau de la dimensionnalité.

Afin de remédier à ce problème, des mécanismes de sélection de gènes ont été introduits, par lesquels seuls les gènes les plus importants sont sélectionnés et utilisés à des fins de classification [81, 123, 182, 25]. Ce processus a plusieurs avantages. En effet, il permet de minimiser le nombre de gènes et de n'en sélectionner que ceux qui sont significatifs et donc les plus prédictifs lors de la classification. En ayant moins de gènes, non seulement le temps de traitement pour la classification est considérablement réduit, mais le risque d'erreur est également restreint. De plus, la saisie d'un nombre élevé de gènes dans le classificateur peut entraîner un sur-apprentissage du classificateur [162]. D'autre part, cette sélection permet une meilleure compréhension des phénomènes étudiés.

L'exemple suivant illustre la notion de sélection d'attributs [202] :

Exemple 2.0.1. (Sous-ensemble optimal) *Pour un problème de classification, nous considérons les attributs booléens $g_1, \dots, g_5$. La fonction de classification est $C = h(g_1, g_2)$ où h est une fonction booléenne, avec $g_2 = \overline{g_3}$ et $g_4 = \overline{g_5}$, (il n'y a que huit possibilités). Afin de déterminer la fonction de classification, g_1 est indispensable ; l'un des g_2 et g_3 peut être éliminé (notons que C peut également être déterminé par $h(g_1, \overline{g_3})$), mais g_4 et g_5 peuvent être éliminés puisqu'ils ne sont pas utiles pour la classification. Ainsi $\{g_1, g_2\}$ ou $\{g_1, g_3\}$ est le sous-ensemble optimal pour la classification. Le but de la sélection d'attributs est de trouver l'un ou l'autre.*

En présence de centaines ou de milliers d'attributs, les chercheurs ont remarqué qu'il est courant qu'un grand nombre de caractéristiques ne soient pas informatives [198, 195], car elles sont soit non pertinentes, soit redondantes par rapport au classifieur. En d'autres termes, l'apprentissage peut être réalisé de manière plus efficace avec uniquement des attributs pertinents et non redondants. Cependant, le nombre de sous-ensembles d'attributs possibles croît de façon exponentielle avec l'augmentation de la dimensionnalité ; en effet, si N est le nombre total d'attributs alors $2^N - 1$ est le nombre de sous-ensembles possibles. Trouver un sous-ensemble optimal est généralement insoluble [105]. Ainsi, de nombreux problèmes liés à la sélection de caractéristiques se sont avérés NP-difficiles [17, 28].

2.1 Notions de pertinence et redondance

Les attributs notamment les gènes dans les puces à ADN peuvent être répartis en deux catégories : **pertinents** et **non pertinents** ceci par rapport à la classification des échantillons. Les gènes pertinents peuvent être classés en [105, 202] fortement pertinents et faiblement pertinents. À leur tour, les gènes fortement pertinents sont absolument nécessaires dans le sous-ensemble de gènes optimal, tandis que les gènes faiblement pertinents sont conditionnellement importants.

Cependant, les gènes faiblement pertinents peuvent également être redondants. Les gènes hautement corrélés sont généralement appelés gènes redondants car ils n'apportent aucune information supplémentaire pour construire un classificateur. L'objectif de toute méthode de sélection de gènes pour l'analyse des puces à ADN consiste à sélectionner un maximum des gènes pertinents tout en éliminant à la fois les gènes redondants et non pertinents.

2.1.1 Pertinence d'attributs

Soient F un ensemble complet d'attributs, g_i un attribut et $S_i = F - \{g_i\}$. Nous supposons que l'on travaille avec un espace probabilisé où la probabilité est notée P . $P(C | S)$ est la probabilité de la classe C connaissant les attributs de l'ensemble S . Nous définissons alors :

Définition 2.1.1. (*Forte pertinence (Strong relevance)*) Un attribut g_i est fortement pertinent si et seulement si

$$P(C | g_i, S_i) \neq P(C | S_i)$$

Définition 2.1.2. (*Pertinence faible (Weak relevance)*) Un attribut g_i est faiblement pertinent si et seulement si

$$P(C | g_i, S_i) = P(C | S_i) \text{ et } \exists S'_i \subset S_i \text{ tel que } P(C | g_i, S'_i) \neq P(C | S'_i).$$

Définition 2.1.3. (*non pertinence (Irrelevance)*) Un attribut g_i est non pertinent si et seulement si

$$\forall S'_i \subseteq S_i \quad P(C | g_i, S'_i) = P(C | S'_i)$$

Un attribut est considéré comme fortement pertinent lorsque sa présence est obligatoire dans tous les sous-ensembles optimaux, en fait, il ne peut pas être supprimé sans affecter la classification finale. Une faible pertinence suggère que l'attribut n'est pas toujours nécessaire mais peut devenir indispensable pour un sous-ensemble optimal dans certaines conditions. La non-pertinence indique quant à elle, que l'on pourrait se passer de l'attribut. D'après ces définitions, il est clair que dans l'exemple 2.0.1, l'attribut g_1 est fortement pertinent, g_2, g_3 sont faiblement pertinents et g_4, g_5 sont non-pertinents. Un sous-ensemble optimal doit inclure : tous les attributs fortement pertinents, aucun des attributs non pertinents, et un sous-ensemble de caractéristiques faiblement pertinentes. Cependant, il n'est pas indiqué dans les définitions quels attributs faiblement pertinentes sélectionnés. Ce qui nous mène à l'introduction de la notion de redondance des attributs.

2.1.2 Redondance d'attributs

La notion de la redondance d'attributs est généralement exprimée en termes de corrélation entre ces derniers. Il est largement admis que deux attributs sont redondants si leurs valeurs sont complètement corrélées (comme c'est le cas des attributs g_2 et g_3 de l'exemple 2.0.1. En réalité, il n'est peut-être pas si simple de généraliser cette notion à un ensemble d'attributs. A présent nous nous contentons de définir formellement la redondance des attributs afin de concevoir une approche permettant d'identifier et d'éliminer explicitement ceux qui sont redondants. Avant de continuer, nous introduisons d'abord la définition de la couverture de Markov (Markov blanket) d'un attribut telle qu'elle a été formulée par Koller et Sahami [106, 202].

Définition 2.1.4. (*Couverture de Markov*) Étant donné un attribut g_i , soit $M_i \subset F$ ($g_i \notin M_i$), M_i est dit être une couverture de Markov pour g_i si et seulement si

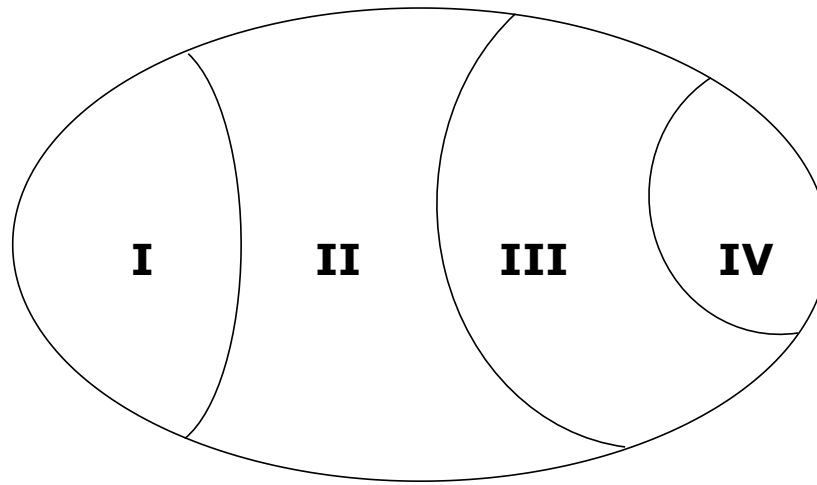
$$P(F - M_i - \{g_i\}, C | g_i, M_i) = P(F - M_i - \{g_i\}, C | M_i).$$

La définition de la couverture de Markov exige que M_i contienne non seulement les informations que g_i apporte sur C , mais également celles qui se rapportent à tous les autres attributs. Il est souligné dans [106] qu'un sous-ensemble optimal peut être obtenu par une procédure d'élimination en arrière (backward elimination procedure), connue sous le nom de « Markov blanket filtering » : Soit G l'ensemble courant d'attributs ($G = F$ au début), à n'importe quelle phase, s'il existe une couverture de Markov pour g_i dans le G actuel, alors g_i est supprimé de G . Il est prouvé que ce processus garantit qu'un attribut enlevé dans une étape antérieure peut toujours avoir une couverture de Markov dans une étape postérieure.

Selon les définitions précédentes de la pertinence des attributs, il est possible d'établir que les attributs fortement pertinents ne peuvent trouver aucune couverture de Markov. Étant donné que les attributs non pertinents doivent être supprimés de toute façon, il n'est donc pas nécessaire de s'y intéresser dans la définition des variables redondantes.

Définition 2.1.5. (*Attribut redondant*) [202]. Soit G l'ensemble actuel des attributs, un attribut est redondant et doit donc être supprimé de G s'il est faiblement pertinent et possède une couverture de Markov M_i dans G .

D'après la propriété de la couverture de Markov, il est facile de voir qu'un attribut redondant supprimé antérieurement reste tout de même redondant lorsque d'autres attributs sont supprimés. La figure.2.4 illustre les relations entre les définitions de la pertinence et la redondance des attributs introduites jusqu'à présent. Il montre qu'un ensemble initial d'attributs peut être conceptuellement divisé en quatre parties disjointes de base (figure.2.4) : attributs non pertinents (I), attributs redondants II (partie d'attributs faiblement pertinents), attributs faiblement pertinents mais non redondants (III) et attributs fortement pertinents (IV). Un sous-ensemble optimal contient essentiellement tous les attributs des parties III et IV. Il convient de souligner que bien que les parties II et III soient disjointes, des partitions différentes de celles-ci peuvent résulter du processus de filtrage par couverture de Markov. Dans l'exemple 2.0.1, l'un des gènes g_2 ou g_3 doit être supprimé comme étant un attribut redondant, mais pas les deux à la fois.



- | | |
|---|--|
| (I) Attributs non pertinents | (III) Attributs faiblement pertinents
mais non redondants |
| (II) Attributs faiblement pertinents
mais redondants | (IV) Attributs fortement pertinents |
| (III+IV) Sous-ensemble optimal | |

figure. 2.4: Catégorisation des attributs

2.2 Présentation formelle du problème de sélection

Le problème de sélection d'un sous ensemble d'attributs peut être perçu comme une sorte de recherche dans un espace d'hypothèses (ensemble de solutions possibles).

Étant donné un ensemble initial $\mathbf{X}$ de N attributs, la sélection du sous ensemble optimal nécessite d'examiner potentiellement $2^N - 1$ sous-ensembles (figure.2.5). La qualité d'un sous-ensemble sélectionné est évaluée selon un critère de performance. Dans le cas d'un problème de classification supervisée, ce critère est souvent l'exactitude d'un classifieur construit à partir de cet ensemble.

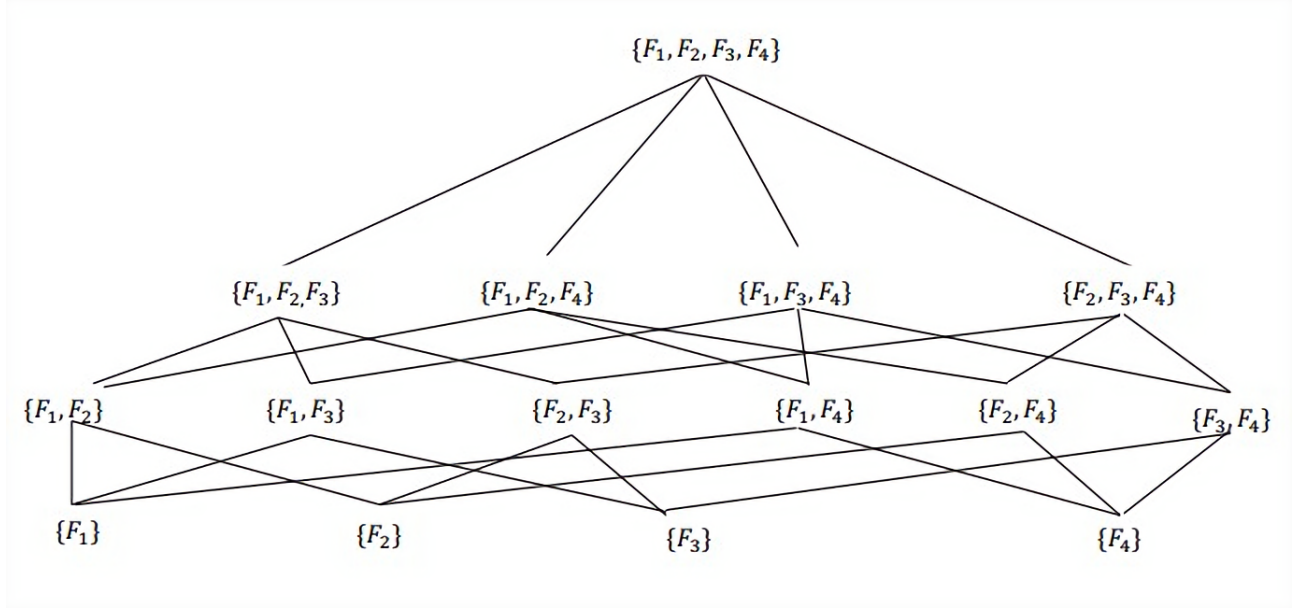


figure. 2.5: Sous-ensembles possibles à partir d'un ensemble de 4 attributs

Définition 2.2.1. (Sélection d'attributs [136]) Soit $J(X')$ une mesure d'évaluation à optimiser (disons maximiser) définie par $J : X' \subseteq X \rightarrow \mathbb{R}$

La sélection d'un sous-ensemble d'attributs peut être vue sous trois considérations :

- Pour un nombre n fixé, avec $n < N$, nous cherchons $|X'| = m < n$, avec $X' \subset X$, tel que $J(X')$ soit maximum.
- Fixer J_o , la valeur minimale tolérée de J , puis trouver $X' \subseteq X$ avec un $|X'|$ plus petit, tel que $J(X') \geq J_o$
- Trouver un compromis entre minimiser $|X'|$ et maximiser $J(X')$ (cas général).

Notons qu'avec ces définitions, un sous-ensemble optimal d'attributs n'est pas nécessairement unique.

2.3 Processus global de la sélection d'attributs

Le processus de sélection d'un sous-ensemble d'attributs pertinents et informatifs à partir de l'ensemble original de ces derniers peut être divisé en quatre étapes principales, comme le montre la figure.2.6. La décision prise à chaque étape influence les performances de sélection des attributs [29, 18] :

1. Déterminer la stratégie de recherche dont dépendra l'exploration de l'espace de recherche.
2. Fixer le critère d'évaluation permettant de donner la qualité des sous-ensembles candidats.
3. Définir des critères d'arrêt
4. Valider le résultat : pour vérifier si l'objectif souhaité est atteint

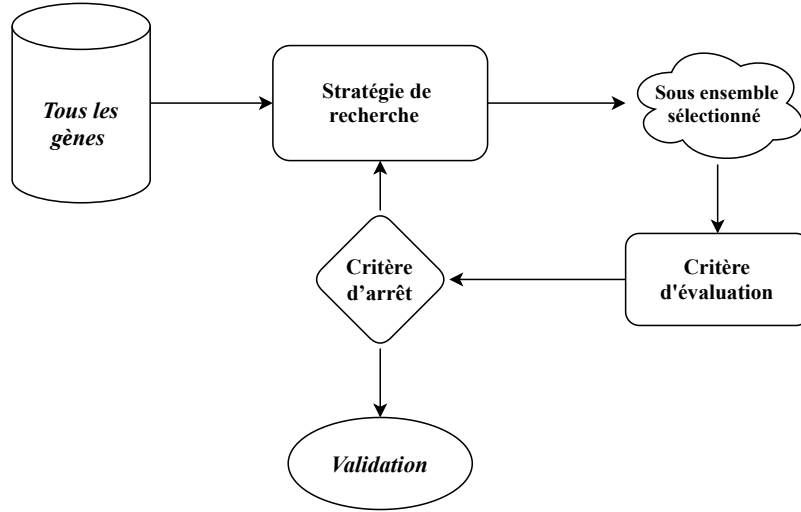


figure. 2.6: Processus de sélection d'attributs

Nous détaillons ci-dessous les étapes importantes de ce schéma.

2.3.1 Directions et stratégies de recherche

Les **directions de recherche** peuvent être de trois types : ajout d'attributs (forward), suppression d'attributs (backward) et bidirectionnelles (bi-directional) :

La stratégie d'ajout d'attributs démarre avec un ensemble vide, puis à chaque itération, nous ajoutons successivement de nouveaux attributs dans l'ensemble. Dans la stratégie de suppression d'attributs nous commençons avec un ensemble complet d'attributs, puis ces derniers sont éliminés consécutivement de l'ensemble à chaque itération. Et finalement la stratégie bidirectionnelle, consiste à en ajouter et en supprimer simultanément des attributs à chaque itération, le processus de recherche peut être entamé par un choix aléatoire d'un sous-ensemble de départ.

Si l'ensemble des données contient N attributs, le nombre total des sous-ensembles candidats concurrents à générer est de $2^N - 1$. C'est un nombre énorme, même pour les N de taille moyenne. Il faudrait alors trouver une stratégie performante de recherche [74], qui devrait garantir à la fois une bonne capacité de recherche globale, une convergence rapide vers une solution presque optimale, une excellente capacité de recherche locale et une efficacité de calcul élevée. Les stratégies de recherche peuvent être classées en trois groupes : complètes, séquentielles, et aléatoires.

Une recherche complète Cette procédure de génération effectue une recherche complète du sous-ensemble optimal selon le critère d'évaluation utilisé. Un exemple de cette stratégie est la recherche exhaustive [50], une recherche qui évalue tous les sous-ensembles possibles pour trouver le sous-ensemble optimal. Cependant, une méthode complète n'est pas forcément exhaustive [165]. En effet, différentes fonctions heuristiques sont utilisées pour réduire la recherche sans compromettre les chances de détecter le sous-ensemble optimal. Par conséquent, bien que l'ordre de l'espace de recherche soit $O(2^N)$, moins de sous-ensembles sont évalués. L'optimalité du sous-ensemble d'attributs, selon le critère d'évaluation, est garantie car la procédure peut faire le retour sur trace ou retour arrière (backtracking). Celui-ci

peut être effectué à l'aide de diverses techniques, L'algorithme le plus classique rencontré en littérature est l'algorithme de séparation et évaluation « Branch and Bound » de Narendra et Fukunaga [139]. Les algorithmes classiques de recherche complète intègrent tous une heuristique les rendant opérationnels, parmi ces dernières figurent : la recherche best-first (littéralement : le meilleur en premier) et recherche de faisceau (beam search.).

Une recherche séquentielle : La recherche séquentielle, également appelée recherche gourmande en escalade (greedy hillclimbing search), ajoute ou supprime un attribut à la fois. Les stratégies séquentielles les plus courantes sont les algorithmes (sequential forward selection « SFS ») et (sequential backward selection SBS). Ces approches sont relativement simples à mettre en œuvre, leur complexité est polynomiale par rapport au nombre d'attributs et elles sont robustes face aux problèmes de multicolinéarité. Cependant, ces méthodes sont sensibles à l'interaction des attributs, de sorte qu'ils peuvent facilement être piégés dans des minima locaux [74]. Les algorithmes de recherche flottante séquentielle : (sequential forward floating selection (SFFS) et sequential backward floating selection (SBFS) [155] ont été développés afin de surmonter ces difficultés en fournissant des mécanismes pour re-sélectionner les attributs supprimés et supprimer ceux qui sont déjà ajoutés. D'autres exemples de stratégie de recherche séquentielle sont la recherche best-first (littéralement : le meilleur en premier) et recherche de faisceau (beam search.)

Une recherche aléatoire : Elle commence par une sélection aléatoire des attributs, puis procède par deux stratégies différentes de recherche. La première utilise la recherche séquentielle ou bidirectionnelle classique, par exemple le recuit simulé [52] et L'escalade stochastique (random hillclimbing) [177]. La deuxième, quant elle, fait appel à des stratégies qui n'ont pas de mouvements réguliers, comme c'est le cas des algorithmes génétiques (AGs), la recherche Tabu, l'optimisation par essaims particulaires (PSO), les algorithmes de colonies de fourmis (ACO), la recherche à voisinage variable (VNS)[115, 39, 168, 6, 24, 25, 182, 70]. dernières stratégies citées peuvent échapper aux optima locaux, mais nous pouvons les améliorer à travers des algorithmes de recherche locale, ces méthodes évaluent généralement moins de nombres de sous-ensembles que les méthodes complètes en définissant un nombre maximal d'itérations possibles. L'optimalité du sous-ensemble sélectionné dépend des ressources disponibles. Chaque procédure de génération aléatoire nécessiterait des valeurs de certains paramètres. L'attribution de valeurs appropriées à ceux-ci est une tâche cruciale pour obtenir de bons résultats.

2.3.2 Critère d'évaluation

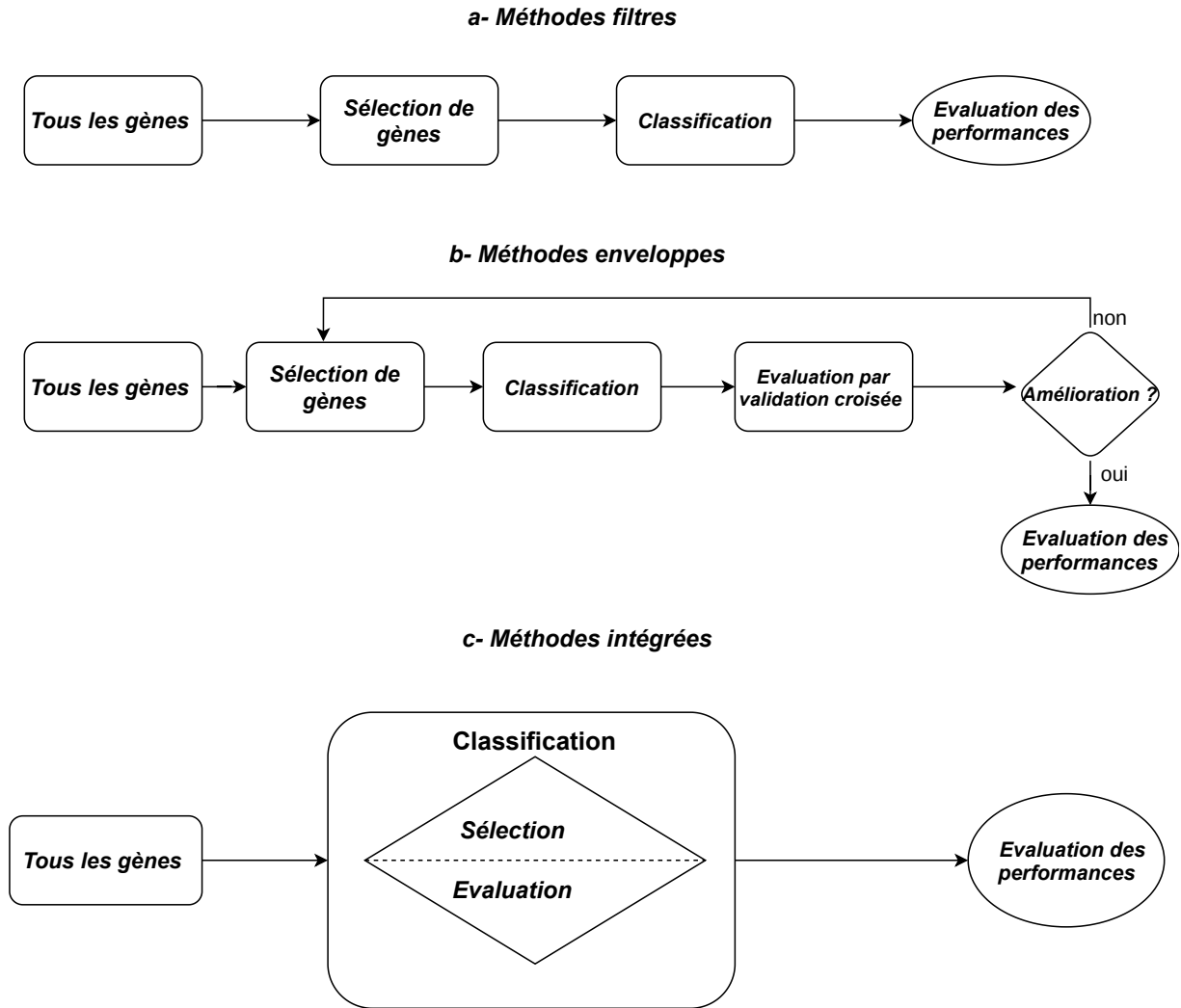


figure. 2.7: Méthodes de sélection de gènes

A cette étape, chacun des sous-ensembles d'attributs candidats est examiné par rapport à son meilleur sous-ensemble précédent par un critère d'évaluation particulier. Si la sortie de l'évaluation confirme que le nouveau sous-ensemble est meilleur que le précédent, le premier remplace le second, nous trouvons trois types de méthodes communes à de nombreuses taxonomies [163] selon le critère d'évaluation et leur interaction avec le classificateur : les méthodes **Filtres** (Filter) [205], **Enveloppes** (Wrapper)[99] et, **Intégrées** (Embedded) [171] (figure.2.7). Les méthodes filtres sont également appelées méthodes indépendantes car elles sont indépendantes de tout algorithme d'apprentissage, tandis que les méthodes enveloppes et intégrées sont qualifiées dépendantes parce qu'elles utilisent les performances d'un algorithme d'apprentissage dans le processus de sélection des attributs [47, 119].

2.3.2.1 Méthodes filtres

Les méthodes filtres (figure.2.7-a) mesurent la qualité des sous-ensembles d'attributs générés en s'appuyant sur les caractéristiques intrinsèques de l'ensemble d'apprentissage des données indépendamment de tout algorithme d'apprentissage. Ces méthodes peuvent facilement s'adapter à des ensembles de données de grande dimension car elles sont simples et rapides en termes de calcul par rapport aux autres approches. Bien que ces méthodes soient efficaces sur le plan informatique, leurs sous-ensembles de caractéristiques sélectionnés peuvent ne pas être optimaux pour le classifieur utilisé. Cela est dû au fait qu'aucun algorithme d'apprentissage spécifique ne guide le processus de sélection des attributs. Généralement, toute méthode filtre consiste à classer les attributs par ordre d'importance par rapport à un critère d'évaluation précis. Ce dernier est ensuite divisé en deux types de schémas :

- Schémas univariés [26] qui consistent à évaluer chaque attribut indépendamment des autres pour lui assigner un score de pertinence. La sélection dans ce type s'effectue en choisissant les attributs les mieux classés. Parmi les inconvénients de ces schémas, nous trouvons l'ignorance des dépendances entre les attributs, cela peut conduire à des performances moins bonnes comparées à d'autres approches.
- Schémas multivariés [47, 146] où les attributs sont sélectionnés en optimisant une fonction de coût particulière qui est souvent définie comme un compromis entre la pertinence maximale et la redondance minimale à l'intérieur du sous-ensemble sélectionné d'attributs /gènes.

Le principal avantage des méthodes filtres reste leur efficacité calculatoire et leur robustesse face au sur-apprentissage. Malheureusement, la plupart des techniques proposées sont univariées. Cela signifie que chaque attribut est considéré séparément, en ignorant ainsi les interactions entre lui et les autres, ce qui peut entraîner une dégradation des performances de classification et sélectionner des attributs comportant de l'information redondante plutôt que complémentaire [82]. Ces approches ont donc du mal à éliminer les attributs qui sont redondants. Par ailleurs, ces méthodes reposent généralement sur le choix d'un seuil pour le critère de pertinence choisi ou d'un nombre d'attributs à sélectionner qui doit être fixé à priori. Le choix de ces paramètres n'est pas facile à réaliser.

Pour mesurer la pertinence d'un attribut nous faisons appel à des outils statistiques classiques ou à d'autres techniques dont nous citons : (l'algorithme ReliefF, le score discriminant BW, le test de Fisher ou le test de Wilcoxon...).

2.3.2.1.1 Le critère de Fisher [79]

Il permet de mesurer le degré de séparabilité des classes à l'aide d'un attribut donné g_i . Il est défini par :

$$F_i = F(g_i) = \frac{\sum_{k=1}^c n_k (\mu_k^i - \mu^i)^2}{\sum_{k=1}^c n_k (\sigma_k^i)^2}$$

Où n_k , μ_k^i et σ_k^i représentent respectivement l'effectif, la moyenne et l'écart type du $i^{\text{ème}}$ attribut au sein de la classe k . μ^i est la moyenne globale du $i^{\text{ème}}$ attribut.

On pourrait dire que la mesure est liée à la variance interclasse de l'attribut.

2.3.2.1.2 Score discriminant BW

Le score discriminant BW a été proposé par Dudoit et al. [60]. Il est basé sur le rapport de dispersion entre les classes et la dispersion intra-classes. Pour un attribut i , ce rapport est obtenu comme suit :

$$BW(g_i) = \frac{\sum_{j=1}^M \sum_{k=1}^c I(y_j = k)(\bar{x}_{ki} - \bar{x}_i)^2}{\sum_{j=1}^M \sum_{k=1}^c I(y_j = k)(x_{ji} - \bar{x}_i)^2}$$

Où $\bar{x}_i$ et $\bar{x}_{ki}$ dénotent, la moyenne d'un attribut i à travers tous les échantillons et à travers les échantillons appartenant uniquement à la classe k .

2.3.2.1.3 ReliefF

Cet algorithme, introduit sous le nom de Relief dans [103] puis amélioré et adapté au cas multi-classes par Kononenko [107] sous le nom de ReliefF ne se contente pas d'éliminer la redondance mais de plus il définit un critère de pertinence. Celui-ci mesure la capacité de chaque caractéristique à regrouper les données de même étiquetée et discriminer celles ayant des étiquettes différentes. L'algorithme est décrit ci-dessous.

Algorithme 2 Algorithme de sélection ReliefF

Début

pour ($i=1 : N$)

 Initialiser les poids de chaque Attribut g_i , $W[g_i] = 0$

3: **Fin pour**

pour ($j=1 : M$)

 trouver les K plus proches voisins de la même classe que x_j (hits)

6: trouver les K plus proches voisins de classe différente de celle de x_j (misses)

pour ($i=1 : N$)

$$W[g_i] = W[g_i] - \sum_{q=1}^k \frac{\text{diff}(x_j, g_i, \text{hits}_q)}{M * k} + \sum_{cl \neq \text{class}(x_j)} \frac{p(cl)}{1 - p(\text{class}(x_j))} \sum_{q=1}^k \frac{\text{diff}(x_j, g_i, \text{misses}_q)}{M * k}$$

9: **Fin pour**

Fin pour

Fin

ReliefF utilise une fonction diff pour calculer la distance entre deux échantillons pour un attribut g_i (discret ou symbolique). Cette distance vaut 1 quand les valeurs de g_i sont différentes et 0 quand elles sont égales.

Le poids d'un attribut est d'autant plus grand que les données issues de la même classe ont des valeurs proches et que les données issues de classes différentes sont bien séparées.

2.3.2.1.4 Gain d'information [114]

Comme son nom l'indique, le filtre Gain d'information est basé sur quelques mesures inspirées de la théorie de l'information de Shannon. Il utilise le concept d'incertitude et évalue les attributs en fonction de leur capacité à réduire l'incertitude vis-à-vis de la classe. Un concept très important qui est conforme à la base de la théorie de l'information est celui de l'entropie d'une variable aléatoire discrète par elle-même ou étant donné (sachant) une autre variable aléatoire discrète. Elles sont représentées dans les équations suivantes :

$$H(X) = - \sum_{x \in X} p(x) \log_2 p(x)$$

$$H(X | Y) = - \sum_{y \in Y} p(y) \sum_{x \in X} p(x | y) \log_2 p(x | y)$$

L'entropie est une mesure de la quantité d'incertitude qu'une variable aléatoire détient, par exemple si X est une variable aléatoire de Bernoulli avec $p = 0.9$, elle aura une entropie de $H(X) \approx 0.47$ mais si la quantité d'incertitude est incrémentée en définissant $p = 0.5$ alors l'entropie augmentera à $H(X) = 1,0$

Cette métrique de gain d'information sert à mesurer le degré de dépendance entre deux variables aléatoires X (un attribut déterminé) et Y (la classe) en utilisant leur entropie et leur entropie conditionnelle. Il s'agit d'un filtre univarié qui produit un classement des attributs en sortie.

$$I(X, Y) = H(X) - H(X | Y)$$

2.3.2.1.5 Redondance minimale Pertinence maximale (MRMR) [51]

La méthode Redondance minimale Pertinence maximale (Minimum Redundancy Maximum Relevance MRMR) est un filtre multivarié qui produit un classement des attributs basé sur leur pertinence des attributs par rapport à la classe tout en pénalisant la redondance. La pertinence d'un attribut repose sur l'information mutuelle qu'elle partage avec la classe et sa redondance est obtenue grâce à l'information mutuelle provenant du reste des attributs sélectionnés.

2.3.2.1.6 Fast Correlation Based Feature Selection [201]

Différent des précédents, le Fast Correlation-Based Filter (FCBF) ne produit pas un classement des attributs mais plutôt un sous-ensemble de ces derniers. C'est aussi un filtre multivarié puisqu'il considère les relations entre les attributs dans le but de réduire la redondance. Le FCBF utilise une mesure de corrélation faisant appel à la théorie de l'information connue sous le nom d'incertitude symétrique donnée par l'équation suivante :

$$SU(X, Y) = 2 \cdot \left[\frac{H(X) - H(X | Y)}{H(Y) + H(X)} \right]$$

Cette mesure permet de détecter des corrélations linéaires et non linéaires entre deux variables aléatoires discrètes étant les attributs ou la classe. Il convient de noter qu'elle mesure est symétrique, c'est-à-dire $SU(X, Y) = SU(Y, X)$. Ceci est utile lors du calcul des associations entre les attributs car aucun d'entre eux ne peut être identifié comme la classe.

2.3.2.2 Méthodes enveloppes

Les méthodes enveloppes (figure.2.7-b) ont été introduites par Kohavi et John [97]. Ces approches enveloppent la sélection d'attributs autour de l'algorithme d'apprentissage (classifieur). Elles utilisent le taux d'erreur de classification ou la précision des performances comme critère d'évaluation des sous-ensembles d'attributs dont chacun sera utilisé pour entraîner le modèle de classification. Le calcul des performances du modèle fait souvent intervenir la validation croisée.

Il est démontré (de façon empirique) dans [97, 4] que ces méthodes donnent souvent de meilleurs résultats par rapport aux méthodes filtres, car elles prennent en compte la dépendance entre les attributs et le biais intrinsèque de l'algorithme de classification. Cependant, elles sont moins générales que les méthodes filtres car elles doivent être ré-exécutées si un autre algorithme d'apprentissage est utilisé. Il n'y a donc aucune garantie que la solution soit optimale pour d'autres classifieurs. De plus, ces méthodes sont plus sujettes au sur-apprentissage que les filtres car le classificateur est appelé à plusieurs reprises pour évaluer chaque sous-ensemble. La majorité des méthodes enveloppes sont multivariées, elles nécessitent donc un temps de calcul important pour atteindre les convergences et peuvent être insolubles pour les problèmes de grandes tailles.

Les algorithmes génétiques (AGs), la recherche Tabu, l'Optimisation par essaims particuliers (PSO), les algorithmes de colonies de fourmis (ACO), Recherche à voisinage variable (VNS)[115, 39, 168, 6, 24, 25, 182, 70] sont des exemples représentatifs des méthodes enveloppes.

2.3.2.3 Méthodes intégrées

Les méthodes intégrées (figure.2.7-c) sont des mécanismes de sélection d'attributs qui intègrent la sélection dans l'algorithme d'apprentissage et utilisent ses propriétés pour guider l'évaluation des sous-ensembles candidats. Ces méthodes sont plus efficaces et plus faciles à gérer que les méthodes enveloppes, tout en maintenant des performances similaires. En effet, les méthodes intégrées évitent l'exécution répétitive du classificateur et l'examen de chaque sous-ensemble d'attributs. De plus, ces approches présentent un risque plus faible de sur-apprentissage par rapport aux méthodes enveloppes.

Les méthodes intégrées prennent en compte les dépendances entre les attributs, mais ne sont spécifique qu'à un algorithme d'apprentissage donné [147]. Cependant, la complexité de calcul est le problème majeur, en particulier quand les données de grande taille.

Un exemple de cette catégorie est un classificateur basé sur un arbre de classification [34]. Un autre exemple représentatif courant de cette approche est la méthode du « Least Absolute Shrinkage and Selection Operator » (LASSO) [183]. Il construit un modèle qui pénalise les coefficients de régression. Les attributs avec des coefficients de régression non nuls sont sélectionnés par l'algorithme LASSO [9]. Une revue sur diverses méthodes de sélection de caractéristiques pénalisées peut être trouvée dans [124] et dans le chapitre 4.

2.3.3 Critère d'arrêt

Le critère d'arrêt détermine quand le processus de sélection des caractéristiques doit être terminé. En fait, les étapes de génération et d'évaluation sont répétées jusqu'à ce que un critère d'arrêt soit satisfait. Ce dernier peut être déterminé par :

- L'étape de génération de sous-ensemble, où le processus peut être arrêté soit lorsqu'un nombre spécifique d'attributs est sélectionné ou lorsqu'un nombre maximum d'itérations est atteint.
- L'étape d'évaluation, où la fin de la sélection des attributs peut être déterminée par la stabilité des performances, ce qui signifie que l'ajout ou la suppression d'attributs n'imposerait aucun changement ou amélioration des performances. Elle peut également être déterminée en obtenant un sous-ensemble d'attributs suffisamment bon, c'est-à-dire un sous-ensemble qui obtient une mesure d'évaluation qu'est supérieure à un seuil d'acceptation (acceptance threshold).

2.3.4 Validation des résultats

Enfin, le sous-ensemble d'attributs sélectionné (c'est-à-dire considéré comme le meilleur par rapport à tous les autres sous-ensembles générés) doit être validé. En conséquence, la validation des résultats peut être effectuée en évaluant simplement le sous-ensemble sélectionné à l'aide des connaissances préalables du domaine. D'autre part, diverses techniques d'estimation ou de validation d'erreur ont été proposées. La méthode d'estimation d'erreur la plus courante est la validation croisée et la mesure de la performance basée sur une matrice de confusion. En outre, d'autres validations et analyses ont également été utilisées dans des études antérieures. Par exemple : indice Jaccard, Indice de Kuncheva, analyse de variance (ANOVA) [94, 108]. L'article [138] traite l'utilisation de la méthode de validation croisée et de certaines expressions courantes dérivées de la matrice de confusion.

2.3.4.1 Métriques d'évaluation

Il existe plusieurs métriques d'évaluation des performances supervisées qui sont liées à la classification des données, et peuvent être obtenues après avoir appliqué un classificateur sur l'ensemble d'attributs sélectionnés. Trois mesures sont largement utilisées : l'exactitude (accuracy), la précision (precision) et le rappel (recall). Pour bien illustrer ces mesures, nous considérons un problème classification binaire, dont il y a une classe positive et une autre négative, cela nous permet de construire une matrice de confusion (table 2.1). La table 2.1 est composée des différentes combinaisons de valeurs prédites et réelles des deux classes.

Vrai positif : (True Positive TP) le nombre total de prédictions correctes qui sont positives, ce qui signifie qu'elles ont été prédites comme positives sachant qu'elles appartiennent en fait à la classe positive.

Faux positif (False Positive FP) : le nombre total de prédictions incorrectes qui sont positives, ce qui signifie qu'elles ont été prédites comme positives sachant qu'elles appartiennent en fait à la classe négative.

Vrai négatif (True Negative TN) : le nombre total de prédictions correctes qui sont négatives, ce qui signifie qu'elles ont été prédites négatives sachant qu'elles appartiennent en fait à la classe négative.

Faux négatif (False Negative FN) : le nombre total de prédictions incorrectes qui sont négatives, c'est-à-dire qu'elles ont été prédites négatives sachant qu'elles appartiennent réellement à la classe positive.

Table. 2.1: La matrice de confusion, un tableau des quatre différentes combinaisons de valeurs de classe prédites (lignes) et réelles (colonnes).

	Réel Positif	Négatif réel
Prédit positif	TP	FP
Négatif prévu	FN	TN

Exactitude de la classification : une métrique d'évaluation bien connue définie comme un pourcentage de prédictions correctes. Ainsi, il évalue combien de points de données ont été correctement classés en utilisant les caractéristiques sélectionnées par les classificateurs

$$\text{Taux d'exactitude} = \frac{TP + TN}{TP + TN + FP + FN}$$

Précision et rappel : deux autres mesures d'évaluation bien connues généralement utilisées ensemble et pouvant être appliquées dans le cadre de la classification. La précision répond à la question de savoir quelle proportion de prédiction positive était réellement correcte. Alors que le rappel, également appelé sensibilité, répond à la question de savoir quelle proportion de réels positifs a été identifiée correctement.

$$\text{Précision} = \frac{TP}{TP + FP}$$

$$\text{Rappel} = \frac{TP}{TP + FN}$$

2.4 Conclusion

Dans ce chapitre, nous avons traité le domaine de la réduction de la dimensionnalité. Dans un premier temps, une revue du domaine de la sélection des attributs a été présentée. Après avoir exposé les différents composants nécessaires à un algorithme de sélection d'attributs, les alternatives possibles pour leurs mises en œuvre ont été présentées. Un certain nombre d'algorithmes a alors été décrit en fonction de la combinaison de composants utilisée cela en présentant les avantages et les inconvénients des techniques de sélection en général. Finalement, nous avons détaillé les méthodes de validation des résultats obtenus.

La recherche d'un sous ensemble d'attributs optimal à partir d'un ensemble d'attributs de dimension élevée est un problème d'optimisation NP-difficile. Les chapitres suivants sont consacrés à la présentation des nouvelles méthodes innovantes de sélection de gènes pour la classification des cancers.

Méthodes Hybrides pour la sélection de gènes pour la classification du cancer

3.1 Introduction

I have always believed that process is more important than results.

MS Dhoni

Les données des biopuces ont des caractéristiques de dimension élevée, de bruit élevé et de petite taille d'échantillon. Ces données bruitées limitent les performances des classificateurs. Comme nous avons un petit nombre d'échantillons alors que chaque échantillon est décrit par un très grand nombre de gènes, l'un des principaux défis de l'analyse de l'expression génique est de sélectionner un petit sous-ensemble de gènes qui contient les informations nécessaires et pertinentes pour un cancer donné [115].

Dans ce contexte, la sélection d'attributs (gènes) est devenue l'objet qui attire l'attention de nombreux chercheurs ces dernières années, cette sélection permet d'identifier et de supprimer des attributs inutiles, non pertinents et redondants qui pénalisent les performances des classificateurs. Elle facilite également l'interprétation et la compréhension des aspects médicaux et biologiques, ainsi, il défie le fléau de la dimensionnalité (curse of dimensionality) pour améliorer les performances des méthodes de classification. En pratique, il existe trois sous-classes d'algorithmes de sélection d'attributs, à savoir : les méthodes de filtres, les méthodes de enveloppes, et les méthodes intégrées [105].

Dans les approches filtres la sélection est indépendante du classificateur utilisé, le principe de la majorité de ces méthodes consiste à évaluer chaque attribut individuellement, et à lui attribuer un score. La sélection des attributs se fait en choisissant les attributs les mieux classés. Ces méthodes sont faciles à mettre en œuvre. Le principal point négatif de ces méthodes est qu'elles évaluent les caractéristiques individuellement en négligeant les interactions possibles avec les autres attributs. Dans la littérature, il existe plusieurs méthodes individuelles de classement des gènes telles que

le score de Fischer [79], le t-test [95], le rapport signal/bruit [132], le gain d'information [188] et ReliefF [107], etc. Dans les méthodes enveloppes, les attributs sont sélectionnés à l'aide du classificateur. Leur idée générale consiste à générer des sous-ensembles candidats et à les évaluer à l'aide d'un classifieur spécifique, ces méthodes donnent une haute performance, et elles sont simples : il suffit de générer et d'évaluer. Bien qu'ils souffrent généralement d'un coût de calcul accru et d'un surajustement possible [105]. Dans les méthodes enveloppes, les méta-heuristiques sont couramment utilisées pour générer des sous-ensembles d'attributs de haute qualité. Dans ces méthodes, l'évaluation des sous ensembles candidats est effectuée à l'aide d'un classificateur spécifique.

Des exemples de classificateurs utilisés pour évaluer chaque solution candidate incluent le K plus proche voisin (KNN) et séparateurs à vaste marge (SVM). Ils sont répertoriés dans le top 10 des algorithmes d'exploration de données les plus influents dans la communauté des chercheurs [194].

Les premiers travaux sur la classification des données oncologiques ont été publiés vers la fin des années 1990 [15, 78]. Dans [78], par exemple, les données obtenues à partir de puces à ADN ont permis de discriminer deux formes de leucémie. De plus, parmi les 6817 gènes testés, un petit nombre de gènes (environ 50) semblent être très importants pour la reconnaissance des deux formes de la maladie.

La sélection de gènes peut être envisagée dans une telle perspective de recherche combinatoire. En effet, étant donné un ensemble de N gènes candidats, le but est de rechercher, parmi les sous-ensembles possibles, un (petit) sous-ensemble particulier mène à la précision de prédiction la plus élevée. Nous constatons aisément que l'espace de recherche est l'ensemble de tous les sous-ensembles possibles. Clairement, cet espace est hautement combinatoire puisqu'il y a $2^N - 1$ sous-ensembles possibles. Il sera donc illusoire d'essayer de faire une énumération exhaustive de toutes les solutions candidates possibles pour toute valeur raisonnable de N , disons, > 20 . Par conséquent, au lieu de rechercher le sous-ensemble optimal en parcourant tout l'espace de recherche, nous pouvons rechercher le meilleur sous-ensemble possible en explorant, de manière sélective, un nombre limité de bons sous-ensembles candidats. Pour accomplir cette tâche, les métaheuristiques constituent une classe de méthodes importantes [77], qui propose différentes stratégies pour explorer un grand espace de recherche combinatoire.

Plusieurs définitions ont été proposées pour expliquer clairement la notion de métaheuristique, nous citons parmi elles :

« Un processus itératif qui subordonne et qui guide une heuristique, en combinant intelligemment plusieurs concepts pour explorer et exploiter tout l'espace de recherche. Des stratégies d'apprentissage sont utilisées pour structurer l'information afin de trouver efficacement des solutions optimales, ou presque optimales » [141].

Les métaheuristiques constituent une classe de méthodes qui fournissent des solutions de bonne qualité en un temps raisonnable à des problèmes combinatoires et continues réputés difficiles pour lesquels nous ne connaissons pas de méthode classique plus efficace. Elles sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global, c'est à dire l'extremum global d'une fonction en évaluant une certaine fonction objectif. Elles se comportent comme des algorithmes de recherche, tentant d'apprendre les caractéristiques d'un problème afin d'en trouver une approximation de la meilleure solution d'une manière proche des algorithmes d'approximations.

L'objectif commun des algorithmes métaheuristiques est de trouver des solutions presque optimales pour un problème donné, par exemple, les PSO, ACO, VNS et GAs sont considérés comme des métaheuristiques qui peuvent être utilisées pour l'optimisation combinatoire dans laquelle une solution optimale est recherché sur un espace de recherche discret. En particulier le problème de sélection de gènes, en tant que tel, il n'y a pas de meilleur algorithme dans l'exploration de données. Plusieurs chercheurs ont utilisé des méthodes métaheuristiques pour résoudre le problème de sélection de gènes, afin de faciliter la reconnaissance de cellules cancéreuses : l'optimisation par essaims particulaires (particle swarm optimization PSO) [3, 24, 38, 39, 83, 115, 134, 135], l'algorithme de colonies de fourmis (ant colony optimization ACO) [25, 37, 116, 166, 181, 182, 203], la recherche à voisinage variable (variable neighborhood search VNS) [69, 70, 142, 149], Les algorithmes génétiques (GAs) [6, 112, 115, 126, 144], l'algorithme compétitif impérialiste [192], et l'algorithme d'évolution différentielle binaire (Binary Differential Evolution (BDE)) [20], l'optimisation simplifiée en essaim (improved simplified swarm optimization (SSO)) [109, 110].

Dans le cadre de la sélection de gènes pour la classification des cancers, il existe des milliers de gènes, alors il est inefficace d'adopter un algorithme évolutif directement dans un jeu de données de biopuces sans passer par une étape de pré-traitement. Dans cette étape, un sous-ensemble de gènes est sélectionné en utilisant une méthode de sélection par filtre dans la majorité des cas, puis un algorithme évolutif est utilisé pour sélectionner le sous-ensemble final [182]. À titre d'exemples, la statistique du chi carré et un algorithme génétique sont utilisés dans [112], le filtre de gain d'information et un algorithme mémétique sont exposés dans [210], le gain d'information et l'optimisation simplifiée en essaim (improved simplified swarm optimization (SSO)) sont utilisés dans [109]. Zhao et al. ont proposé une approche hybride en combinant le score de Fisher avec un AG et un PSO [208] : ReliefF, MRMR et GAs sont étudiés dans [174]. Ainsi, les métaheuristiques ont été utilisées avec succès pour résoudre le problème de sélection de gènes qui est connu pour être NP-difficile [17, 48].

Dans ce chapitre nous présentons les principales contributions basées sur les méta heuristiques PSO, ACO et VNS pour résoudre le problème de sélection de gènes dans les puces à ADN.

3.2 Jeux de données

Pour évaluer nos approches, nous avons choisi des jeux de données (puces à ADN) qui concernent tous des problèmes de reconnaissance de cancers importés depuis (Gene Expression Model Selector www.gems-system.org) (consulté en janvier 2018) [72].

Nous avons utilisé des jeux de données publics, facilement accessibles et qui sont d'ailleurs exploités dans de nombreux travaux concernant la classification des données.

Le détail des jeux de données est présenté dans la Table 3.1. Le choix de ces derniers repose sur une multitude de spécificités (classe binaire ou multi-classes, nombre de gènes, nombre d'échantillons). Certains jeux de données ont des classes binaires (Prostate_Tumor, DLBCL) alors que d'autres ont des multi-classes (Leukemia1, Lung_Cancer...). Par ailleurs quelques-uns disposent d'un nombre inférieur d'échantillons (Brain_Tumor2, 9_Tumors...), tandis que d'autres en possèdent un nombre supérieur (Lung_Cancer, 11_Tumors...).

Comme nos approches sont basées sur des données de grande dimension tous ces jeux sont caractérisés par des milliers de gènes allant de 2308 à 12600 (high-dimensional data)[72].

Table. 3.1: La description des puces à ADN utilisées.

Jeux de données	Tâche de diagnostic	Nombre		
		d'échantillons	de gènes	de classes
11 Tumors	11 types de tumeurs humaines différents	174	12533	11
9 Tumors	9 types de tumeurs humaines différents	60	5726	9
Brain Tumor1	5 types de tumeurs cérébrales différents	90	5920	5
Brain Tumor2	4 types de gliomes malins	50	10367	4
Leukemia1	3 types de leucémie : AML, ALL(B-cell) et ALL(T-cell)	72	5327	3
Leukemia 2	3 types de leucémie : AML, ALL et MLL.	72	11225	3
Lung Cancer	4 types de cancer du poumon et des tissus normaux	203	12600	5
SRBCT	Small, round blue cell tumors (SRBCT) of childhood	83	2308	4
Prostate Tumor	Tumeur de la prostate et des tissus normaux	102	10509	2
DLBCL	Diffuse large b-cell lymphomas (DLBCL) and follicular lymphomas	77	5469	2

3.3 Sélection de gènes via BPSO et une recherche locale de type (backward) pour la classification du cancer

Dans cette section, nous présentons deux approches hybrides pour faire face au problème de sélection de gènes pour la classification des données de puces à ADN. La première approche est basée sur la combinaison d'une méthode de type Filtre, utilisant le critère de Fisher et une méthode de type Enveloppe (wrapper), basée sur l'algorithme d'optimisation par essaim de particules (BPSO) hybridé par une méthode de recherche locale (méthode descente), en utilisant un classifieur SVM pour l'évaluation des sous-ensembles candidats, pour la deuxième approche nous remplaçons le critère de Fisher par l'algorithme de ReliefF, et le classifieur SVM par le KNN.

Ces approches proposées n'ont pas été étudiées auparavant. De plus, ce travail étudie les performances de nos méthodes, en les appliquant à huit ensembles des jeux de données de la littérature bien connus de dimensions élevées variant de 2308 à 11225 (Table 3.1). Les expérimentations menées sur les différents jeux de données montrent que nos méthodes s'avèrent très compétitives avec les travaux existants.

Cette étude a fait l'objet d'une publication dans le journal indexé (RAIRO-Oper. Res) [24]

3.3.1 Approche proposée

Le problème de sélection de gènes peut être considéré comme un problème d'optimisation, où nous essayons de trouver le meilleur sous-ensemble de gènes qui maximise une fonction objectif f .

3.3.1.1 L'optimisation par essaim de particules (PSO)

L'optimisation par essaim de particules (le PSO en anglais : Particle Swarm Optimazation) est une métaheuristique d'optimisation numérique basé sur l'intelligence en essaim, introduit en 1995 par James Kennedy, un psychologue social, et Russell Eberhart, un ingénieur électricien [101], inspirée du comportement social d'animaux évoluant en essaim, tel que le vol d'oiseaux ou les bancs de poissons pour la résolution de problèmes d'optimisation non linéaires, non convexes ou combinatoires qui se posent dans de nombreux domaines scientifiques et techniques.

De nombreuses espèces d'oiseaux sont sociales et forment des groupes pour diverses raisons. Les troupes (essaims) peuvent être de tailles différentes, se produire à des saisons différentes et peuvent même être composés de différentes espèces qui peuvent bien travailler ensemble dans un groupe. Plus d'yeux et d'oreilles signifient des possibilités accrues de trouver de la nourriture et de meilleures chances de détecter un prédateur à temps. En effet, nous pouvons observer chez ces animaux des dynamiques de déplacement relativement complexes, alors qu'individuellement chaque individu a une intelligence limitée et une connaissance seulement locale de sa situation dans l'essaim.

Pour le développement du modèle PSO, Kennedy et Eberhart ont suivi cinq principes fondamentaux qui déterminent si un groupe d'agents est un essaim ou non [199] :

- Principe de proximité : la population doit être capable d'effectuer des calculs spatiaux et temporels simples.

- Principe de qualité : la population doit être en mesure de réagir aux facteurs de qualité de l'environnement.
- Principe de réponse diversifiée : la population ne doit pas engager son activité dans des canaux trop étroits.
- Principe de stabilité : la population ne doit pas changer de mode de comportement à chaque fois que l'environnement change.
- Principe d'adaptabilité : la population doit pouvoir changer de mode de comportement lorsque cela en vaut la peine.

Considérant ces cinq principes, Kennedy et Eberhart ont développé un modèle PSO pour l'optimisation des fonctions. Dans PSO, la solution est obtenue grâce à une recherche aléatoire équipée d'une intelligence en essaim. En d'autres termes, PSO est un algorithme de recherche intelligent en essaim. Cette recherche se fait par un ensemble de solutions potentielles générées aléatoirement. Cette collection de solutions potentielles est connue sous le nom **d'essaim** et chaque solution potentielle individuelle est connue sous le nom de **particule**.

Contrairement aux autres algorithmes évolutionnaires tel que l'algorithme génétique où la recherche de la solution optimale évolue par compétition entre les individus en utilisant des opérateurs de croisements et de mutations, le PSO utilise plutôt la coopération entre les individus. Les particules de l'essaim représentent des solutions potentielles au problème traité. L'essaim de particules survole l'espace de recherche, en quête de l'optimum global. Le déplacement de chaque particule est influencé par les trois composantes suivantes (figure.3.1) :

- Une composante **d'inertie** : la particule tend à suivre sa direction courante de déplacement.
- Une composante **cognitive** : la particule tend à se diriger vers le meilleur site par lequel elle est déjà passée.
- Une composante **sociale** : la particule tend à se fier à l'expérience de ses congénères et, ainsi, à se diriger vers le meilleur site déjà atteint par ses voisins.

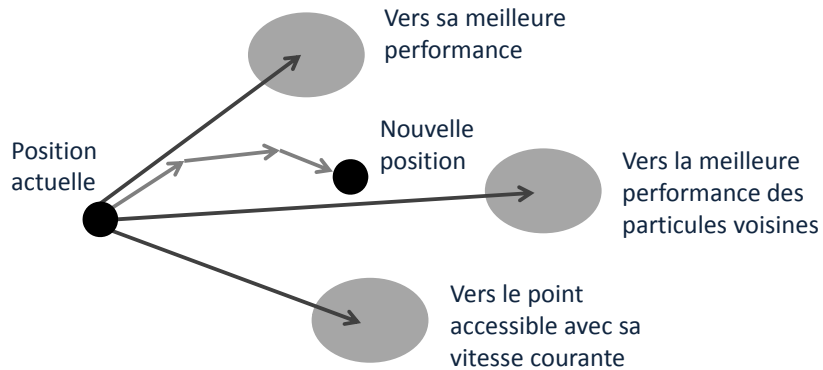


figure. 3.1: Déplacement d'une particule [61].

Chaque particule i de l'essaim de taille m est définie par sa position $S_i = [s_i^1, s_i^2, \dots, s_i^p]$ et sa vitesse de déplacement $V_i = [v_i^1, v_i^2, \dots, v_i^p]$ dans un espace de recherche de dimension p . Cette particule garde en mémoire la meilleure position par laquelle elle est déjà passée et la meilleure position atteinte par ses voisins, notées respectivement : $P_i = [p_i^1, p_i^2, \dots, p_i^p]$ et $P_{vi} = [p_{vi}^1, p_{vi}^2, \dots, p_{vi}^p]$.

Le processus de recherche est basé sur deux règles :

1. Chaque particule est dotée d'une mémoire qui lui permet de mémoriser le meilleur point par lequel elle est déjà passée et elle a tendance à retourner vers ce point.
2. Chaque particule est informée du meilleur point connu au sein de son voisinage et elle a tendance à aller vers ce point.

La particule i se déplace entre les itérations n et $n + 1$, en fonction de sa vitesse et des deux meilleures positions qu'elle connaît (la sienne et celle de son voisinage) suivant les deux formules suivantes [101] :

pour $i = 1, \dots, m$.

$$S_i^{n+1} = S_i^n + V_i^{n+1}. \quad (3.1)$$

$$V_i^{n+1} = wV_i^n + c_1 * rand * (S_i - X_i^n) + c_2 * rand * (P_{vi} - S_i^n). \quad (3.2)$$

Avec :

- m, p , la taille de la population et la dimension de la particule respectivement.
- S_i^{n+1}, S_i^n : la position de la particule i aux temps $n + 1$ et n , respectivement.
- V_i^{n+1}, V_i^n : la vitesse de la particule i aux temps $n+1$ et n , respectivement.
- P_i : la meilleure position obtenue par la particule i au temps n .
- P_{vi} : la meilleure position atteinte par les voisins de la particule i au temps $n-1$.
- w : une constante appelée, coefficient d'inertie, introduite par Shi et Eberhart [169] pour réaliser un équilibre entre la recherche locale (exploitation) et la recherche globale (exploration).
- $rand$: un nombre généré aléatoirement $\in [0, 1]$.
- c_1, c_2 : des constantes qui représentent les coefficients d'accélération; c_1 pondère les tendances de la particule à vouloir suivre son instinct de conservation, et à aller vers sa meilleure position connue et c_2 contrôle l'aptitude sociale de la particule, en se rapprochant plus de la meilleure position de ses voisins.
- wV_i^n : correspond à la composante d'inertie du déplacement.
- $c_1 * rand * (P_i - S_i^n)$: correspond à la composante cognitive du déplacement.
- $c_2 * rand * (P_{vi} - S_i^n)$: correspond à la composante sociale du déplacement.

Pour éviter l'explosion du système (c'est-à-dire que les particules se déplacent trop rapidement d'une région à une autre dans l'espace de recherche), nous définissons v_{min} et v_{max} telle que : $V_i \in [v_{min}, v_{max}]^p$.

Il est à noter que le terme "vitesse" est ici abusif car les vecteurs V_i ne sont pas homogènes à une vitesse. Il serait plus approprié de parler de "direction de déplacement".

Après avoir faire le déplacement des particules, les nouvelles positions sont évaluées et les deux vecteurs P_{vi} et P_i sont mis à jour, la mise à jour se fait de la manière suivante (lorsqu'il s'agit d'un problème de maximisation) :

$$P_i = \begin{cases} S_i^{n+1} & \text{si } fitness(S_i^{n+1}) \geq fitness(P_i). \\ P_i & \text{sinon.} \end{cases} \quad (3.3)$$

$$P_{vi} = \arg \min_{P_j} \min_{j \in V_o(i)} fitness(P_j) \quad (3.4)$$

Avec $V_o(i)$ le voisinage de la particule S_i

Topologie de voisinage

La topologie du voisinage définit avec qui chacune des particules peut communiquer. Nous trouvons plusieurs types de topologie, dans la version classique dite "globale", la meilleure particule P_{vi} est choisie à partir de la population entière. Le graphe d'information est donc complètement connecté. Nous sommes ici en présence d'une topologie statique d'information, car les liens d'information entre les particules sont définis une fois pour toutes.

De même, Eberhart et al. [101] ont défini une version dite "locale" de PSO, qui utilise aussi un graphe d'information statique. Cette version utilise un graphe d'information "circulaire". Les particules de l'essaim sont virtuellement disposées en cercle et numérotées séquentiellement à partir de 1 en parcourant le cercle. La particule n'est donc plus informée par toutes les particules de l'essaim, mais par elle-même et ses deux voisins. Bien qu'elle converge moins rapidement que la version "globale", la version "locale" de PSO donne de meilleurs résultats, car elle est moins sujette à l'attraction par des minima locaux. La figure.3.2 illustre la différence entre un graphe complètement connecté et un graphe circulaire.

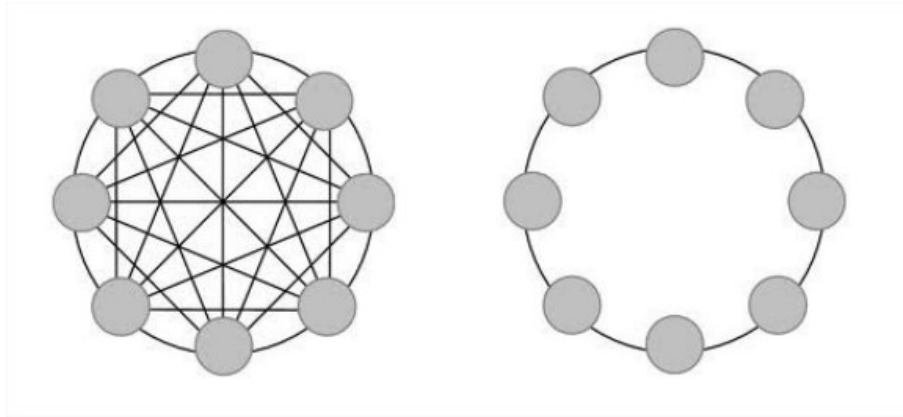


figure. 3.2: Graphe d'influence d'un essaim de particules. (à gauche) Graphe complètement connecté, (à droite) Graphe d'information circulaire [58].

L'algorithme suivant représente un pseudo code de PSO.

Algorithme 3 Algorithme de PSO général

```

1: Entrées :  $n_{max}$ ,  $m, w, v_{max}, v_{min}, c_1, c_2$ ;
2:
3: Sorties :  $S_{best}$ 
   Début
4:   Initialiser  $m$  particules : position et vitesse.
5:
6:   Evaluer les positions des particules.
7:   Pour chaque particule  $i$ ,  $P_i = S_i$ 
8:   Pour chaque particule  $i$  Calculer  $P_{vi}$  selon (3.3 )
9:   Tant que le critère d'arrêt n'est pas satisfait faire
10:    Déplacer les particules selon (3.1) et (3.2 ).
11:    Evaluer les positions des particules.
12:    Mettre à jour  $P_i$  et  $P_{vi}$  selon (3.3) et (3.4)
13:  Fin Tant que
14:  Retourner  $S_{best}$ .
Fin

```

3.3.1.2 version discrète/binaire de l'optimisation par essaim de particules (BPSO) pour la sélection de gènes

Au début, la métaheuristique d'optimisation par essaim de particules a été proposée pour résoudre des problèmes d'optimisation continus.

Toutefois, maints problèmes d'optimisation sont discrets ou discrets/binaires. Par conséquent, une tendance active a été remarquée ces dernières décennies, c'est l'adaptation de l'algorithme PSO pour résoudre des problèmes d'optimisation discrets et discrets/binaires.

La première version de l'algorithme BPSO (l'algorithme standard discret/binaire de l'optimisation par essaim de particules) a été proposée en 1997 par Kennedy et Eberhart [102]. Dans l'algorithme BPSO (de l'anglais : Binary Particle Swarm Optimization), la position de la particule i est représentée par un ensemble de bits.

Dans la sélection de gènes la position de chaque particule i , $S_i = [s_i^1, s_i^2, \dots, s_i^p]$ (codée en binaire) présente une solution candidate (sous-ensemble de gènes) dans un espace p -dimensionnel, où chaque bit de la particule représente un gène, si un bit a une valeur de "1", cela signifie que le gène a été sélectionné. En revanche, une valeur de "0" indique que le gène n'est pas sélectionné dans le sous-ensemble. Dans la figure.3.3 la solution candidate signifie que parmi 6 gènes, les gènes $g_1; g_3; g_4$ et g_6 sont sélectionnés, ainsi le sous ensemble candidat est $S = \{g_1; g_3; g_4; g_6\}$

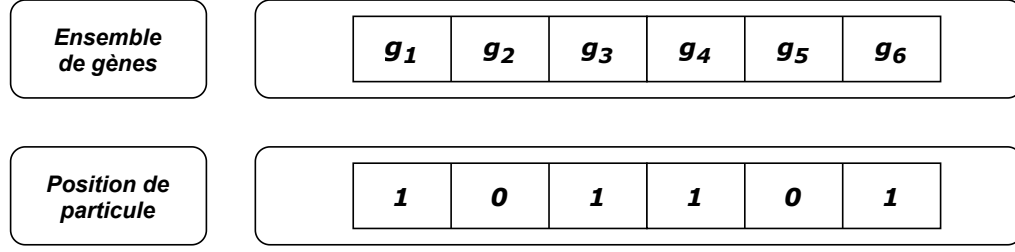


figure. 3.3: Exemple illustré d'un sous-ensemble généré.

Le mouvement d'une particule est influencé par sa vitesse $V_i = [v_i^1, v_i^2, \dots, v_i^p]$ sa meilleure position déjà visitée $P_i = [p_i^1, p_i^2, \dots, p_i^p]$ et la position de son meilleur voisin $P_{vi} = [p_{vi}^1, p_{vi}^2, \dots, p_{vi}^p]$

La vitesse d'une particule i V_i est calculée à partir de l'équation 3.2. V_i est un ensemble de nombres réels qui doivent être transformés en un ensemble de probabilités en utilisant une fonction spécifique.

En particulier la fonction sigmoïde :

$$Sig(t) = \frac{1}{1 + e^{-t}}$$

Où $Sig(v_i^j)$ représente la probabilité du bit s_i^j de prendre la valeur 1.

En se basant sur la vitesse v_i^j de la particule i et sur la valeur de la probabilité $Sig(v_i^j)$, la position de la particule S_i est calculée comme suit :

pour $j=1, \dots, p$

$$s_i^j = \begin{cases} 1 & \text{Si } r < Sig(v_i^j) \\ 0 & \text{sinon} \end{cases} \quad (3.5)$$

avec $r \in [0, 1]$ (généré aléatoirement).

3.3.1.3 La génération en arrière (Backward generation)

La génération en arrière (backward generation) est une méthode de recherche locale pour la sélection de gènes (recherche en escalade ou HillClimbing), dans cet algorithme itératif, nous partons d'un ensemble de gènes et nous essayons de supprimer un gène non pertinent à chaque itération (le gène qui n'influence pas l'exactitude du classifieur doit être supprimé). Le codage des solutions candidates est le même que celui du BPSO (figure.3.4).

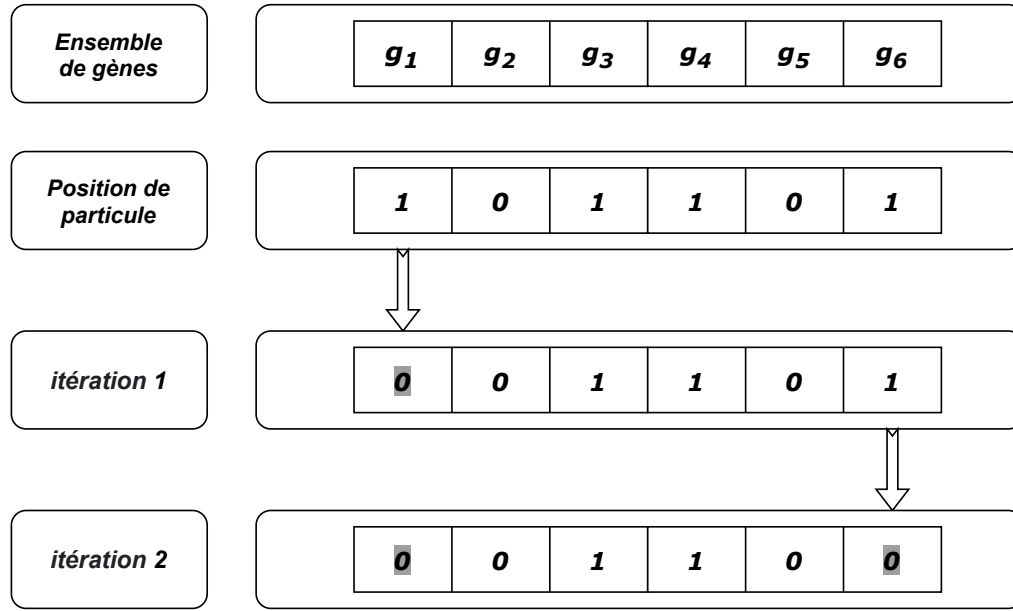


figure. 3.4: Un exemple illustre la simulation de deux itérations de l'approche en arrière; g_1 et g_6 ne sont pas pertinents.

Soit f une fonction qui évalue la qualité d'un sous-ensemble de gènes. L'algorithme suivant présente la génération en arrière (backward generation).

Algorithme 4 Backward generation

```

1: Entrées : Solution candidate  $S$  (ensemble de gènes ), nombre maximum d'itérations  $it_{max}$ 
2: Sortie :  $S_{best}$ 
3: Début
4: for ( $i = 1 : it_{max}$ ) do
5:   Choisir au hasard un gène ( $g_i$ ) de  $S$ 
6:    $S' = S \setminus g_i$ 
7:   if ( $f(S') > f(S)$ ) then
8:      $S = S'$ 
9:   end if
10: end for
11:  $S_{best} = S$ 
12: Fin

```

3.3.1.4 Critère de sélection

Nous avons étudié quelques stratégies de recherche pour trouver le sous-ensemble de gènes optimal. Nous devons maintenant définir "qu'est-ce qu'un bon sous-ensemble de gènes?". Sans

définir une mesure pour l'évaluation de chaque sous-ensemble, nous ne pouvons pas parler du meilleur sous-ensemble de gènes ou le sous-ensemble optimal [121].

Cette évaluation peut être mesurée par une fonction objectif f qui présente un compromis entre le nombre de gènes sélectionnés et l'exactitude (accuracy) du classificateur. Dans le contexte de sélection de gènes nous cherchons à maximiser l'exactitude prédictive et minimiser le nombre de gènes utilisés.

Afin d'estimer l'exactitude, nous avons besoin d'un classificateur qui peut être testé sur les données de test après avoir appris à partir des données d'apprentissage. Dans cette étude, nous utilisons deux classificateurs : SVM linéaire et KNN.

3.3.1.5 Fonction objectif

Pour chaque solution candidate S , la qualité de S est évaluée par une fonction de fitness f selon deux critères : la capacité à obtenir une bonne exactitude (accuracy) de classification avec ce sous-ensemble de gènes et le nombre de gènes contenus dans ce sous-ensemble. Plus formellement, la valeur de fitness est calculée comme suit :

$$f(S) = w_1 * CA_{Classifieur}(S) + (1 - w_1) * (1 - \frac{\#genes}{p}).$$

Où

- w_1 : est un coefficient de pondération en $[0, 1]$ qui contrôle l'agrégation des deux objectifs.
- $\#gènes$: le nombre de gènes sélectionnés dans S et p la dimension du vecteur S .

La fonction de fitness cherche à maximiser l'exactitude de classification ($CA_{Classifieur}$) obtenue avec un classifieur déterminé construit sur le sous-ensemble de gènes S , et calculé en utilisant LOOCV (la validation croisée d'un contre tous, (leave-one-out cross-validation)) (voir le paragraphe 1.2.3, la page 35). Le second terme assure le fait de préférer les sous-ensembles avec moins de gènes. Notre objectif est de maximiser la fonction f . En outre, notre objectif principal est de trouver un sous-ensemble de gènes avec une exactitude maximale, nous choisissons donc $w_1 = 0.98$. En ce qui concerne le **critère d'arrêt**, nous avons défini le critère suivant : atteindre un nombre préfixé de générations.

3.3.1.6 Méthodes proposées

L'originalité de nos méthodes provient de la combinaison de deux approches différentes Filtres (Fisher score et ReliefF) et enveloppes (la métaheuristique BPSO combinée à un algorithme de recherche locale) pour sélectionner un petit nombre de gènes hautement pertinents responsables de la classification du cancer qui facilite le pronostic précoce de la maladie. Ces méthodes proposées enrichissent l'état de l'art de la sélection de gènes.

3.3.1.6.1 Structure générale des approches proposées

La procédure générale de nos approches peut être décrite par un processus séquentiel en trois étapes qui utilise des techniques complémentaires pour réduire progressivement l'espace de recherche et sélectionner un sous-ensemble pertinent de gènes (figure.3.5).

Étape 1 : C'est une étape de pré-traitement qui a pour objectif de filtrer les gènes non pertinents, par exemple, les gènes dont les niveaux d'expression sont uniformes quelle que soit la classe. De plus, il y a des milliers de gènes, il est alors inefficace d'adopter un algorithme évolutif tel qu'une optimisation d'essaim de particules binaires (BPSO) directement dans un ensemble de données de biopuces sans filtration. La sortie de ce pré-traitement est un ensemble de gènes classés par ordre de pertinence selon un score déterminé. Il s'agit d'une étape préliminaire pour réduire la taille des données des puces à ADN. D'après plusieurs tests, le nombre de gènes présélectionnés p a été fixé à 300.

Étape 2 : Cette étape consiste en une méthode enveloppe (Wrapper), où (BPSO) génère des sous-ensembles de gènes évalués par un classificateur spécifique, à partir des gènes retenus par le filtrage précédent.

Étape 3 : Cette étape est basée sur une méthode de génération séquentielle de type **Backward**, appliquée à l'ensemble des gènes sélectionnés à l'étape 2, le but de cette étape est d'améliorer les solutions données par les particules et apporter de bonnes solutions dans un temps raisonnable.

3.3. SÉLECTION DE GÈNES VIA BPSO ET UNE RECHERCHE LOCALE DE TYPE (BACKWARD) POUR LA CLASSIFICATION DU CANCER

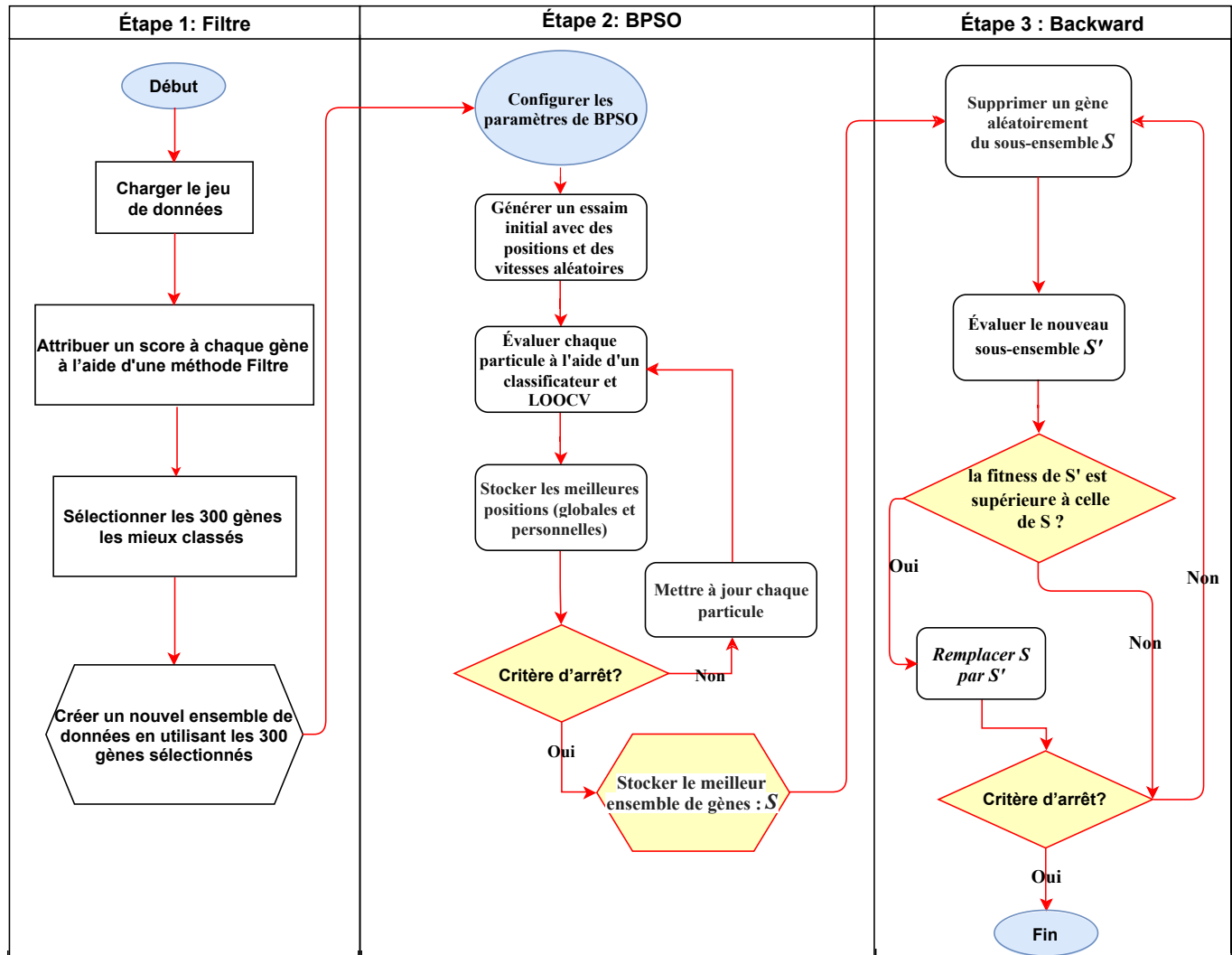


figure. 3.5: Organigramme de nos approches proposées pour la sélection de sous-ensembles de gènes dans les données de puces à ADN.

L'algorithme suivant présente un aperçu des approches proposées.

Algorithme 5 Approche proposée

Entrées : Données de puces à ADN ; n_{max} ; m ; p ; it_{max} ; v_{min} ; v_{max} , w , c_1 ; c_2
Sortie : S_{best}

3: **Début**
Étape 1 :
Appliquer un algorithme de filtrage pour choisir les p gènes les mieux classés.

6: **Étape 2 :**
 $k=0$;
Initialiser aléatoirement l'essaim

9: **Tant que** $k < n_{max}$
| **for** $i = 1$; $i \leq m$; $i++$ **do**
| Évaluer la fitness de la particule S_i par un **Classifieur** spécifique.

12: | | **if** (la fitness de la particule de S_i est supérieure à celle de P_i) **then**
| | $P_i = S_i$
| | **if** (la fitness de S_i est supérieure à celle de P_{vi}) **then** $P_{vi} = S_i$

15: | | **end if**
| **end if**
| Calculer la vitesse V_{i+1} en utilisant (3.2)

18: | | **for** $d = 1$; $d < p$; $d++$ **do**
| | | **if** ($v_{i+1}^d > v_{max}$) **then**
| | | $v_{i+1}^d = v_{max}$

21: | | | **end if**
| | | **if** ($v_{i+1}^d < v_{min}$) **then**
| | | $v_{i+1}^d = v_{min}$

24: | | | **end if**
| | | **if** ($Sig(v_{i+1}^d) > rand()$) **then**
| | | $s_{i+1}^d = 1$

27: | | | **else** $s_{i+1}^d = 0$
| | | **end if**
| **end for**

30: **end for**
 $k=k+1$
Fin Tant que

33: **Étape 3 :**
Appliquer l'algorithme de retour (backward) (Algorithme 4) à P_{vi} .
Retourner S_{best} .

36: **Fin**

3.3.1.7 FBPSO-SVM

Notre première méthode proposée pour la sélection de gènes (FBPSO-SVM) basée sur la combinaison de deux approches pour la sélection de gènes : l'approche Filtre via un score de Fisher

généralisé et l'approche enveloppe via une méthode PSO binaire combinée à un algorithme de recherche locale heuristique appelé (backward generation). Dans cette méthode, la qualité du sous-ensemble candidat de gènes est mesurée à l'aide d'un classificateur SVM multiclasse avec une validation croisée (LOOCV). Le choix du classifieur SVM se justifie par sa robustesse éprouvée pour une donnée de grande dimension.

Le schéma suivant représente la structure générale de notre première approche (figure.3.6) :

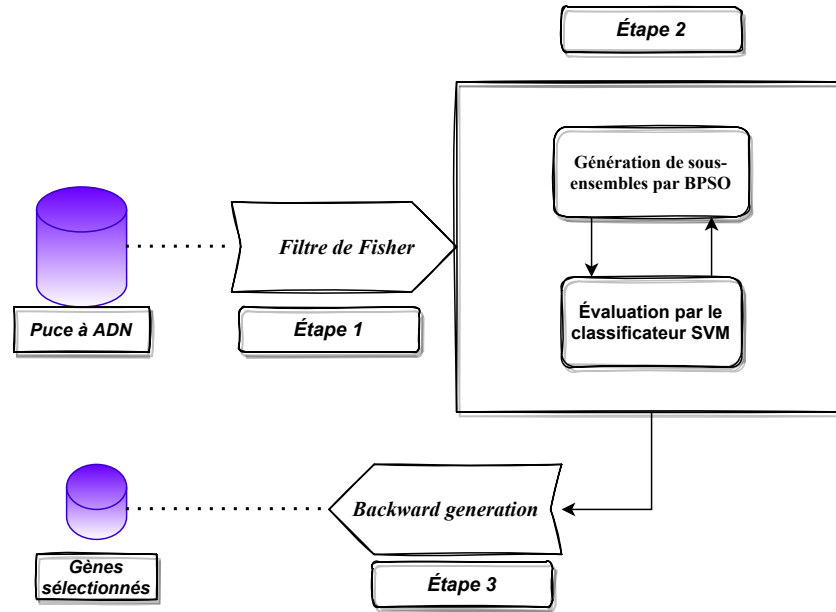


figure. 3.6: Schéma général de FBPSO-SVM..

3.3.1.8 RBPSO-1NN

La seconde approche (RBPSO-1NN) est dérivée de la première (FBPSO-SVM), les deux seules différences sont :

- Dans la première étape (étape de filtrage), le critère de Fisher est remplacé par l'algorithme ReliefF.
- Pour la fonction de fitness utilisée dans l'approche enveloppe, le classificateur 1NN a été choisi, après avoir constaté que KNN convient aux données filtrées par ReliefF (puisque les deux utilisent la notion de voisinage et de distance).

3.3.2 Expérimentations et résultats

L'implémentation des approches proposées (FBPSO-SVM) et (RBPSO-1NN) est faite sous Matlab qui est un langage de programmation de quatrième génération émulé par un environnement de développement du même nom. Nous rappelons au passage que Matlab est développé par la société 'The MathWorks' et qu'il est utilisé essentiellement à des fins de calcul numérique.

En ce qui est du classifieur SVM, nous avons opté pour un classifieur linéaire binaire prédéfini.

Dans le cas où le nombre des classes dépasse deux, nous avons développé un classifieur SVM basé sur la stratégie 'un contre tous'.

Pour le classifieur 1NN et l'algorithme de ReliefF, nous avons eu recours à des fonctions prédéfinies dans Matlab R2017a.

Nous avons choisi les jeux de données issus de (Gene Expression Model Selector www.gems-system.org) pour l'expérimentation (Table 3.1). La suite de la section est dédiée à l'analyse des résultats expérimentaux.

3.3.2.1 Paramètres

Nous notons que nos approches ont été exécutées sur un ordinateur portable Acer Aspire 7750g avec processeur Intel Core I5 2,30 GHz et 8 Go de RAM, sous système exécutant Windows 7 (64 bits).

Dans le cadre de l'étude de nos approches, plusieurs tests ont été réalisés afin d'adopter la meilleure paramétrisation. Plus précisément, nous avons sélectionné la meilleure paramétrisation en fixant d'abord un ensemble de valeurs initiales, puis nous ajustons la valeur d'un paramètre jusqu'à ce que les solutions ne puissent pas être améliorées. Nous répétons le processus jusqu'à ce que l'ajustement de tous les paramètres n'ait pas fourni une amélioration. Ce processus a été testé sur deux ensembles de données de classification du cancer (Brain_tumor1 et Brain_tumor2).

A titre d'exemple pour le poids d'inertie w , la figure.3.7) montre que $w = 1$ est la meilleure valeur de w .

Le nombre de gènes à sélectionner par l'approche filtre (étape 1) est fixé à 300.

Le nombre de voisins pour ReliefF est fixé à 3. D'autre part, le nombre de voisins pour le classifieur KNN est fixé à $K=1$.

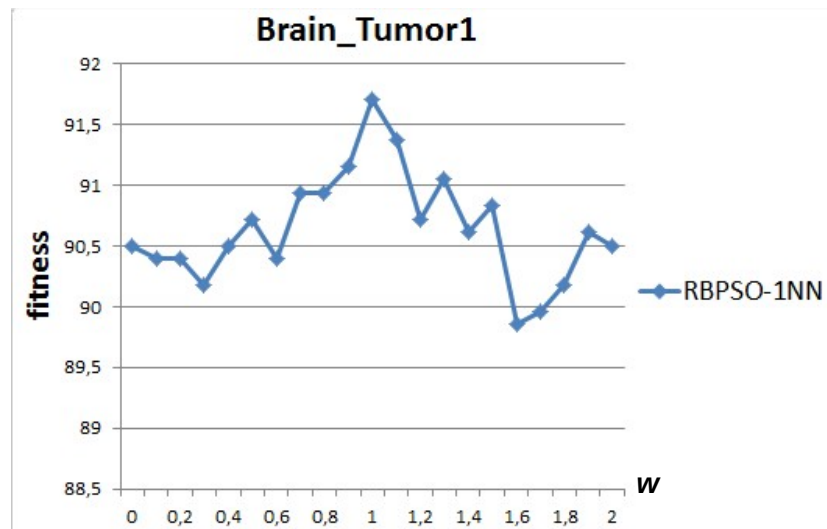


figure. 3.7: Étude paramétrique pour le poids d'inertie w .

Les paramètres que nous avons définis pour explorer la méthode enveloppe sont indiqués dans le tableau suivant (Table 3.2) :

Table. 3.2: Paramètres utilisés pour nos méthodes.

Paramètres	Valeur
Taille d'essaim m	30
Nombre de générations n_{max}	100
$c_1, \quad c_2$	2
w	1
v_{max}	4
v_{min}	-4
Backward generation, it_{max}	200

3.3.2.2 Résultats et comparaisons

Les objets des expérimentations menées sur les huit jeux de données de puces à ADN sont, d'une part, de tester l'effet de la sélection de gènes sur l'amélioration de taux d'exactitude (accuracy) de la classification et le temps de construction de classifieur, et, d'autre part part, pour montrer les performances de nos approches.

3.3. SÉLECTION DE GÈNES VIA BPSO ET UNE RECHERCHE LOCALE DE TYPE (BACKWARD) POUR LA CLASSIFICATION DU CANCER

Table. 3.3: Comparaison de SVM, 1NN, RBPSO-1NN et FBPSO-SVM (LOOCV).

data sets	Performance	SVM	1NN	RBPSO-1NN		FBPSO-SVM	
				Best	Avg	Best	Avg
9_Tumors	Accuracy (%)	38.33	53.33	83.33	81.83	95	92.999
	Gènes	5726	5726	20	29.1	71	45
	Time(h)	-	-	0.68	0.69	4.17	4.22
	Class Time(sec)	26.38	2.21	0.79	0.9	4.89	5.27
Brain_Tumor1	Accuracy (%)	88.89	86.67	94.44	94.00	97,78	97,22
	Gènes	5920	5920	11	24.7	21	22.4
	Time(h)	-	-	1.47	1.48	4,09	4,20
	Class Time(sec)	23.32	3.32	1.10	1.04	4.81	5.38
Brain_Tumor2	Accuracy (%)	70.00	70.00	96.00	92.80	100.00	100.00
	Gènes	10367	10367	15	24.5	12	14.3
	Time(h)	-	-	0.50	0.52	1.84	1.91
	Class Time(sec)	14.28	2.29	0.63	0.59	1.72	1.55
Leukimia1	Accuracy (%)	97.22	87.50	100.00	99.72	100.00	100.00
	Gènes	5327	5327	8	11.7	6	8.4
	Time(h)	-	-	0.76	0.76	1.58	1.60
	Class Time(sec)	10.81	1.96	0.9	0.83	1.68	1.54
Leukemia2	Accuracy (%)	97.22	93.05	100.00	100.00	100.00	100.00
	Gènes	11225	11225	5	13.1	6	8.6
	Time(h)	-	-	0.74	0.84	1.92	1.99
	Class Time(sec)	18.89	3.06	0.85	0.87	1.68	1.71
SRBCT	Accuracy %)	100.00	91.57	100.00	100.00	100.00	100.00
	Gènes	2308	2308	7	11.7	10	12.4
	Time(h)	-	-	0.87	0.85	2.89	3.02
	Class Time(sec)	10.86	2.21	1.02	1.06	3.29	3.01
Prostate_Tumor	Accuracy(%)	92.16	76.47	99.02	98.24	100.00	100.00
	Gènes	10509	10509	9	11.2	6	8.3
	Time(h)	-	-	1.02	1.01	3.04	3.46
	Class Time(sec)	24.48	4.73	1.40	1.29	2.87	2.97
DLBCL	Accuracy (%)	97.40	87.01	100.00	100.00	100.00	100.00
	Gènes	5469	5469	6	12.5	4	6.7
	Time(h)	-	-	1.34	1.30	1.43	1.49
	Class Time(sec)	8.34	2.48	0.97	0.97	1.11	1.20

Note : Les meilleurs résultats sont indiqués en gras.

Accuracy :Le taux d'exactitude de la classification à l'aide de LOOCV.

Gènes : Le nombre de gènes utilisés.

Time : Le temps d'exécution en heures.

Class Time : Le temps pris pour construire le classifieur en secondes.

Best : Le meilleur résultat trouvé dans dix exécutions indépendantes.

Avg : La moyenne des dix expériences.

Compte tenu de la nature non déterministe de nos approches, dix exécutions indépendantes (pour comparer les temps d'exécution) ont été effectuées pour chaque ensemble de données et méthode.

La table 3.3 illustre les taux d'exactitude de classification (LOOCV) pour les huit jeux de données, ces taux sont obtenus à partir des algorithmes SVM et 1NN sans sélection puis avec sélection en utilisant nos approches, ce tableau souligne l'intérêt de la sélection dans un problème de classification :

Nous analysons nos résultats de quatre manières :

1. Le nombre de gènes sélectionnés par les 2 méthodes.
2. Le taux d'exactitude de la classification.
3. Le temps d'exécution.
4. Le temps pris pour construire le classifieur avant et après la sélection.

3.3.2.3 Discussion

Les approches proposées tirent leur efficacité de l'amélioration remarquable de l'exactitude (accuracy) de la classification (indiquée en gras dans le tableau 3.3), pour tous les ensembles de données (figure.3.8)

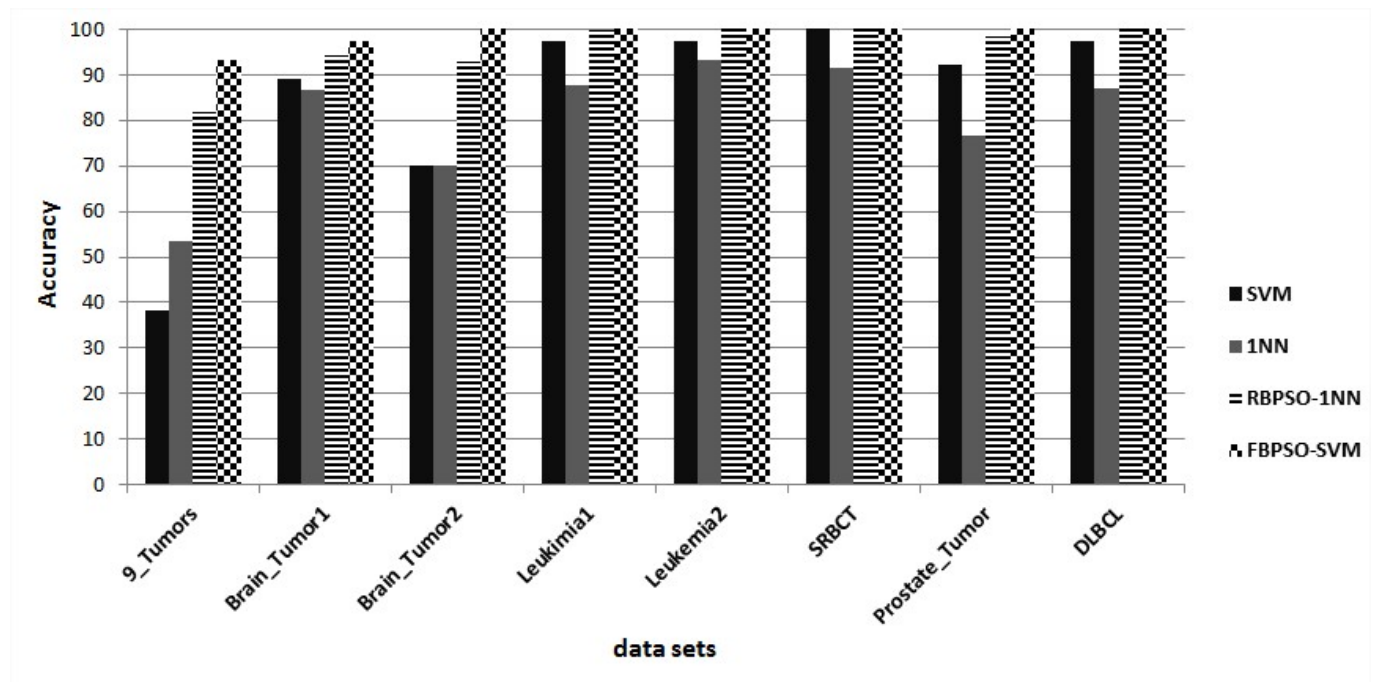


figure. 3.8: Comparaison des taux d'exactitudes de classification entre les quatre méthodes (Best).

Les approches « RBPSO-1NN » et « FBPSO-SVM » réduisent le nombre de gènes par rapport au nombre original de gènes dans les jeux de données, nous parlons d'une sélection qui assure un meilleur taux d'exactitude (accuracy).

De plus, pour l'ensemble de données "Brain_Tumor2", l'approche "FBPSO-SVM" a atteint une exactitude de classification de 100 % en utilisant 0,116 % des gènes d'origine uniquement (12 gènes de 10367). De même, pour l'ensemble de données "Leukimia2", l'approche "RBPSO-1NN" a donné une exactitude parfaite en utilisant seulement 0,044 % des gènes (5 gènes de 11225).

Les résultats montrent tout d'abord que « FBPSO-SVM » et « RBPSO-1NN » réalisent des performances similaires pour la majorité des jeux de données, avec un avantage de FBPSO-SVM au niveau de l'exactitude (en gras table 3.3), à l'exception de "Leukimia2" et SRBCT" où "RBPSO-1NN" est le meilleur selon le nombre de gènes sélectionnés et le temps d'exécution. Cependant la différence majeure entre nos deux approches réside dans le « temps d'exécution », en effet, « RBPSO-1NN » donne de bonnes performances en un temps minimal. Ceci est dû à l'utilisation de 1NN comme classificateur (figure.3.9).

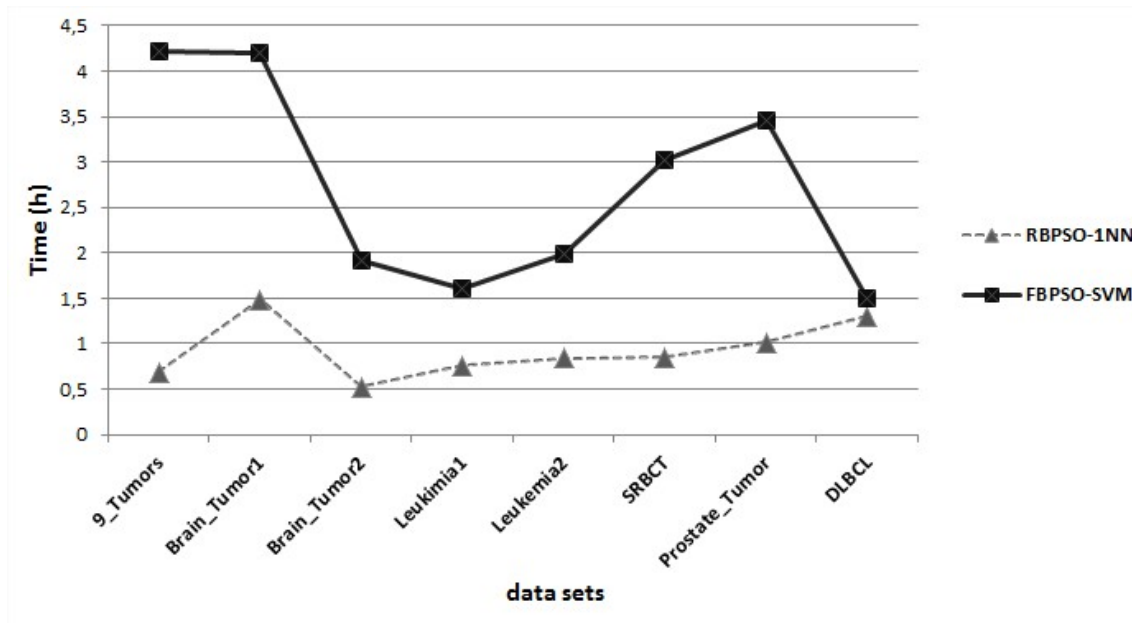


figure. 3.9: Comparaison des temps d'exécution moyens en heures.

Dans les figures (3.11, 3.13 et 3.15), l'axe des abscisses exprime le nombre de générations tandis que l'axe des ordonnées représente l'exactitude de la classification de la meilleure particule dans l'essaim. Ceci est fait pour la solution moyenne trouvée pour les ensembles de données "9_Tumors, Brain_Tumor1 and Brain_Tumor2". Ces chiffres montrent clairement que l'utilisation du BPSO joue un rôle crucial dans l'augmentation du taux d'exactitude de la classification.

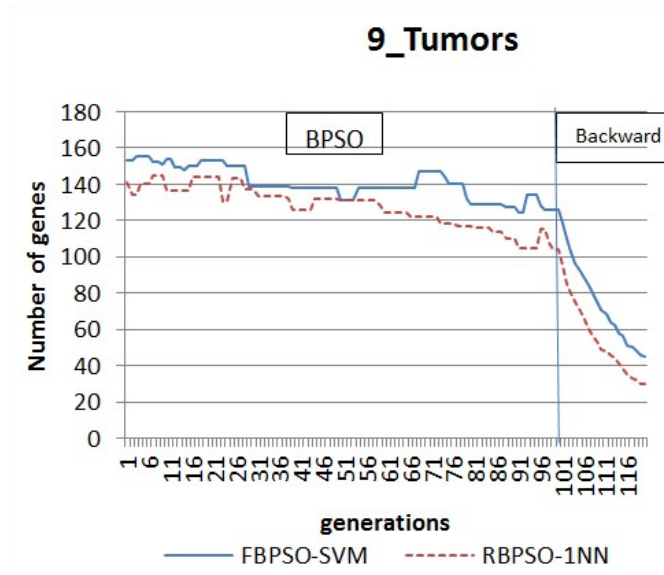


figure. 3.10: Comparaison de l'évolution du nombre de gènes utilisés pour "9_Tumor1".

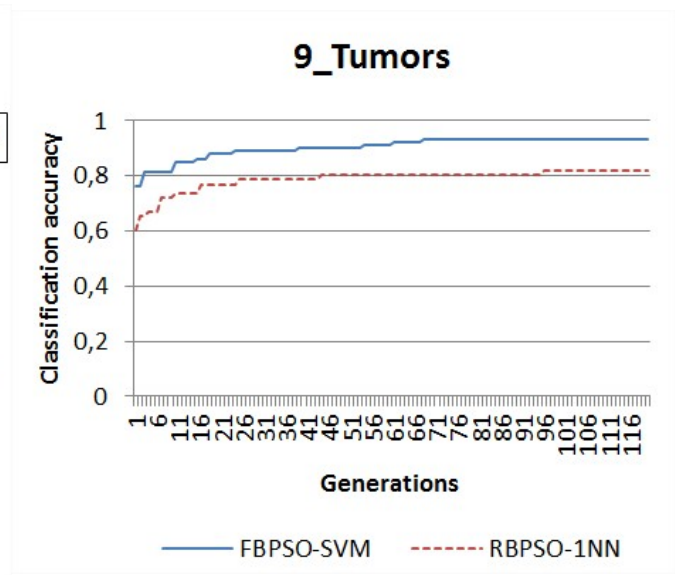


figure. 3.11: Comparaison de l'évolution du taux d'exactitude de la Classification pour "9_Tumors".

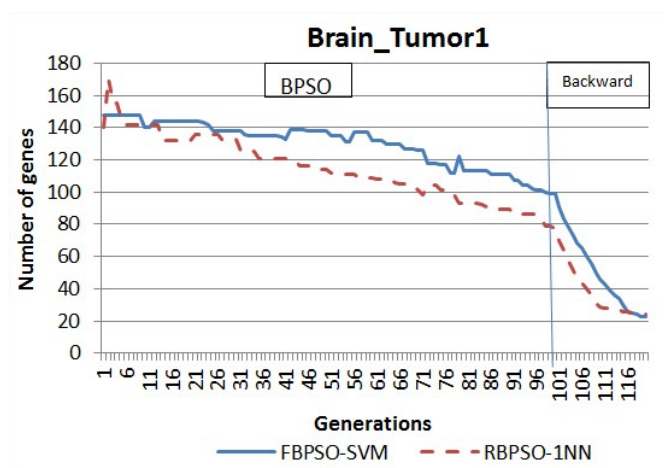


figure. 3.12: Comparaison de l'évolution du nombre de gènes utilisés pour "Brain_Tumor1".

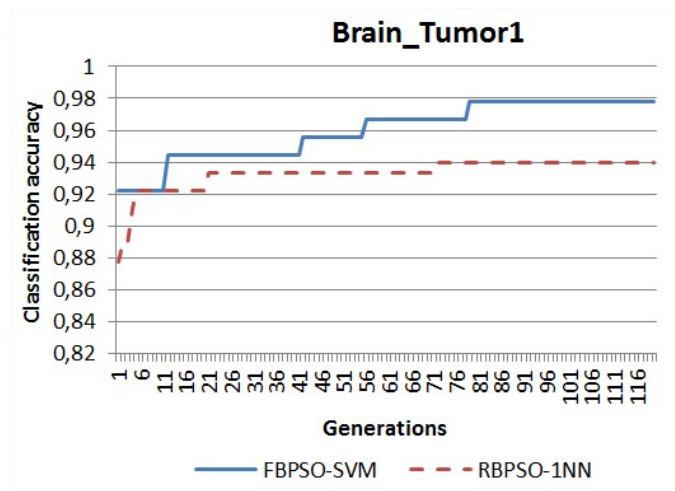


figure. 3.13: Comparaison de l'évolution du taux d'exactitude de la Classification pour "Brain_Tumor1".

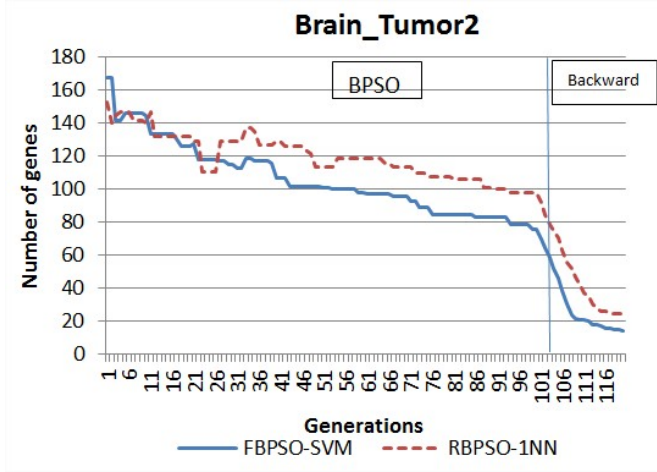


figure. 3.14: Comparaison de l'évolution du nombre de gènes utilisés pour "Brain_Tumor2".

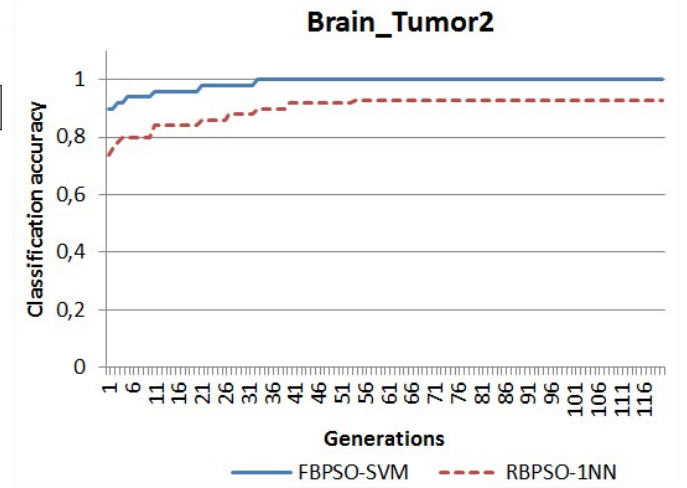


figure. 3.15: Comparaison de l'évolution du taux d'exactitude de la Classification pour "Brain_Tumor2".

Dans les figures (3.10, 3.12 et 3.14), nous montrons l'évolution du nombre de gènes sélectionnés sur l'axe des ordonnées par rapport au nombre de générations (l'axe des abscisses) pour les jeux de données "9_Tumeurs, Brain_Tumeur1 et Brain_Tumeur2". Ces chiffres illustrent le rôle de l'algorithme Backward dans la réduction du nombre de gènes.

De plus, la méthode enveloppe basée sur le BPSO et l'algorithme de génération Backward joue un rôle clé dans l'augmentation du taux d'exactitude de la classification et la réduction du nombre de gènes utilisés. En effet, le BPSO vise à identifier le meilleur sous-ensemble de gènes candidats qui maximise la fonction objectif (un compromis entre le taux d'exactitude et le nombre de gènes utilisés) (figure.3.11, figure.3.13 et figure.3.15). Une fois le sous-ensemble en question trouvé, l'algorithme Backward est appelé pour réduire le nombre de gènes utilisés tout en conservant le taux de classification précédemment fixé (figure.3.10, figure.3.12 et figure.3.14).

Nous avons essayé d'évaluer statistiquement nos résultats à l'aide du test statistique non paramétrique de kruskal-wallis, afin de tester la significativité de la différence dans les résultats (taux d'exactitude) obtenus par nos approches.

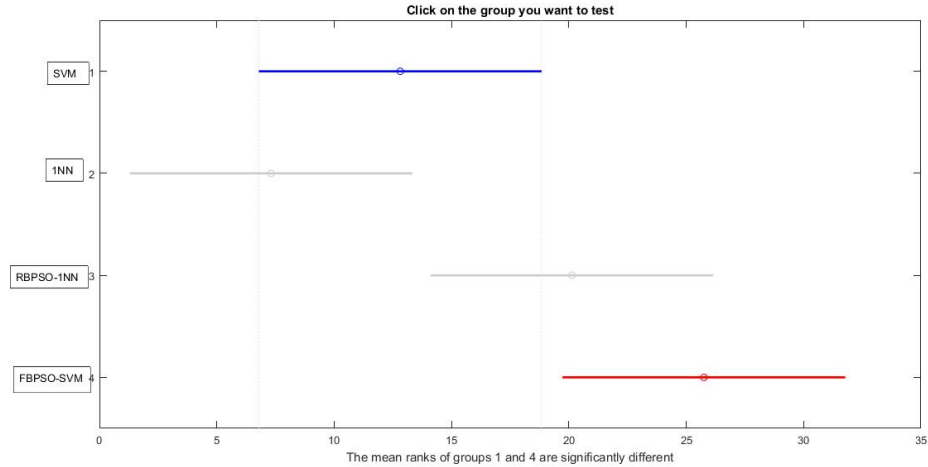


figure. 3.16: Le résultat du test de kruskal-wallis entre les approches proposées, SVM et 1NN sur les jeux de données

Le test statistique de kruskal-wallis présenté dans la figure.3.16 montre une comparaison des résultats obtenus par nos algorithmes proposés, le classificateur SVM et le 1NN. D'après la figure.3.16, les performances de nos approches proposées dépassent celles des classifieurs SVM et 1NN. En termes de significativité statistique des résultats, le test de kruskal-wallis montre que la différence entre le taux d'exactitude en « SVM » et « FBPSO-SVM » est statistiquement significative (grande et remarquable).

Sur la base de ces résultats, nous pouvons dire que nos approches (FBPSO-SVM et RBPSO-1NN) de sélection de gènes sont très bien fondées. En effet, sur les huit jeux de données utilisés, nos approches de sélection ont réussi à améliorer la qualité de la classification. De plus, l'approche FBPSO-SVM a donné un taux de classification égal ou supérieur à 95,00 % pour tous les ensembles de données, avec une classification parfaite de 100 % pour (Brain_Tumor2, Leukimia1, Leukimia2, SRBCT, Prostate_Tumor, DLBCL) utilisant moins de 12 gènes.

En pratique, le choix d'un sous-ensemble optimal de gènes a fait l'objet d'un intérêt croissant dans le domaine biologique, dans le sens où il permet aux biologistes de gagner du temps, puisqu'ils peuvent se référer directement aux gènes sélectionnés qui sont utiles pour le diagnostic du cancer. De plus, un petit sous-ensemble de gènes peut réduire le temps de construction du classifieur, comme indiqué dans (figure.3.17 et figure.3.18).

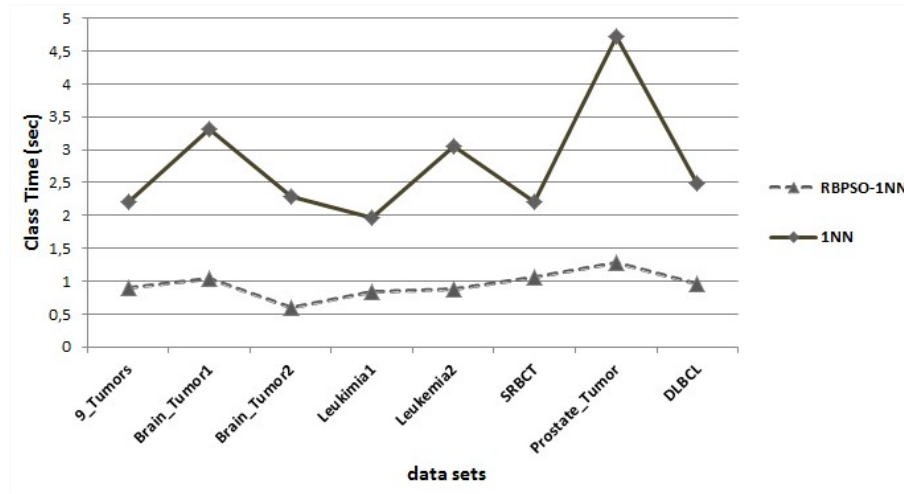


figure. 3.17: Comparaison du temps nécessaire pour la construction du classifieur 1NN avant et après la sélection à l'aide de RBPSO.

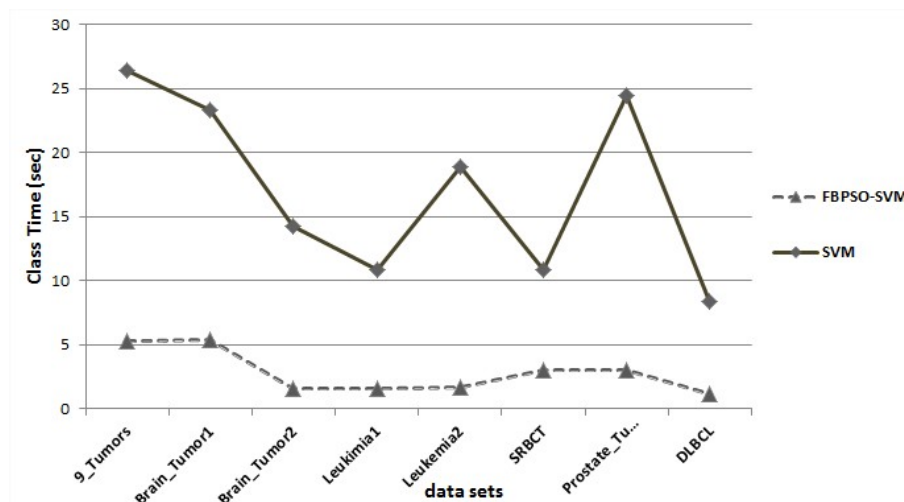


figure. 3.18: Comparaison du temps nécessaire pour la construction du classifieur SVM avant et après la sélection à l'aide de FBPSO.

Dans ce qui suit, nous proposons de faire une comparaison avec d'autres algorithmes d'optimisation sur plusieurs jeux de données de classification de référence.

3.3.2.4 Comparaison avec d'autres approches

Dans cette sous-section, nous comparons nos résultats avec ceux de cinq algorithmes de référence [3, 38, 39, 135, 192]. Pour ce faire, nous réalisons nos expériences en utilisant les mêmes conditions expérimentales utilisées par chaque algorithme de référence. Concrètement, nous exécutons nos approches dix fois, puis nous choisissons la moyenne et le meilleur sous-ensemble de

gènes trouvé.

Tenons compte du fait que les cinq algorithmes de référence utilisent LOOCV pour calculer l'exactitude de la classification.

Nous indiquons que les auteurs de [38, 39] rapportent les meilleurs résultats trouvés sur 10 répétitions.

Le tableau 3.4 illustre l'exactitude de la classification pour les différentes approches ainsi que le nombre de gènes sélectionnés. Nous utilisons le symbole (-) pour indiquer qu'un résultat n'est pas rapporté. Nous constatons que les résultats obtenus par nos approches sont très compétitifs par rapport aux méthodes les plus représentatives des dernières années.

Premièrement, pour l'ensemble de données (9_tumors), nous avons une exactitude de classification de 95 % avec seulement 71 gènes. Nous constatons que la meilleure performance pour cet ensemble de données est la performance de notre approche (FBPSO-SVM). Nous notons que le nombre de gènes rapportés dans d'autres travaux est d'au moins 251 gènes pour avoir une bonne exactitude, et qu'ils ne parviennent pas à fournir des performances élevées.

De même pour les ensembles de données (Brain_Tumor1, Brain_Tumor2 et Prostate_Tumor), nous obtenons les meilleures performances. En utilisant l'approche (FBPSO-SVM), nous avons une classification parfaite (100%) pour (Brain_Tumor2 et Prostate_Tumor) avec moins de 12 gènes, et une précision de 97,78 % pour (Brain_Tumor1) en utilisant seulement 21 gènes. De plus, notre méthode (FBPSO-SVM) a permis d'obtenir des taux d'exactitude plus élevés sur tous les ensembles de données par rapport aux autres méthodes.

Les résultats de cette analyse comparative avec d'autres méthodes proposées pour la sélection de gènes et la classification des données de puces à ADN nous ont permis de savoir à quel point nos approches sont compétitives.

3.3. SÉLECTION DE GÈNES VIA BPSO ET UNE RECHERCHE LOCALE DE TYPE (BACKWARD) POUR LA CLASSIFICATION DU CANCER

Table. 3.4: Comparaison entre nos méthodes (RBPSO-1NN et FBPSO-SVM) et les méthodes de l'état de l'art.

Jeu de données	Méthode	RBPSO-1NN [Ce travail]	FBPSO-SVM [Ce travail]	FRBPSO 2017 [3]	HICATS (2016) [192]	EPSO (2013) [135]	TS-BPSO (2009) [39]	IBPSO (2008) [38]
Performances								
9_Tumors	Best #Acc (%)	83.33	95.00	-	83.33	76.67	81.63	78.33
	Best #Gènes	20	71	-	259	251	2941	1280
	Average #Acc (%)	81.83	92.222	-	78.33	75.00	-	-
	Average #Gènes	29.1	45	-	248.5	247.10	-	-
Brain_Tumor1	Best #Acc (%)	94.44	97,78	-	94.44	93.33	95.89	94.44
	Best #Gènes	11	21	-	6	8	2913	754
	Average #Acc (%)	94.00	97.22	90.67	93.10	92.11	-	-
	Average #Gènes	24.7	22.4	803	8.9	7.5	-	-
Brain_Tumor2	Best #Acc (%)	96.00	100.00	-	94.00	94.00	92.65	94.00
	Best #Gènes	15	12	-	3	4	5086	1197
	Average #Acc (%)	92.80	100.00	87.6	92.60	92.4	-	-
	Average #Gènes	24.5	14.3	662	5.8	6.0	-	-
Leukimia1	Best #Acc (%)	100	100.00	-	100.00	100.00	100.00	100.00
	Best #Gènes	8	6	-	3	2	2577	1034
	Average #Acc (%)	99.72	100.00	98.89	100.00	100.00	-	-
	Average #Gènes	11.7	8.4	825	3	3.2	-	-
Leukemia2	Best #Acc (%)	100	100.00	-	100.00	100.00	100.00	100.00
	Best #Gènes	5	6	-	5	4	5609	1292
	Average #Acc (%)	100.00	100.00	97.50	100.00	100.00	-	-
	Average #Gènes	13.1	8.6	1028	6.80	6.8	-	-
SRBCT	Best #Acc (%)	100.00	100.00	-	100.00	100.00	100.00	100.00
	Best #Gènes	7	10	-	9	7	1084	431
	Average #Acc (%)	100.00	100.00	98.19	100.00	99.64	-	-
	Average #Gènes	11.7	12.4	213	11.7	14.90	-	-
Prostate_Tumor	Best #Acc (%)	99.02	100.00	-	98.04	99.02	95.45	92.61
	Best #Gènes	9	6	-	5	5	5320	1294
	Average #Acc (%)	98.24	100.00	92.43	97.75	97.84	-	-
	Average #Gènes	11.2	8.3	418	7.2	6.6	-	-
DLBCL	Best #Acc (%)	100.00	100.00	-	100.00	100.00	100.00	100.00
	Best #Gènes	6	4	-	3	3	2671	1042
	Average #Acc (%)	100.00	100.00	96.49	100.00	100.00	-	-
	Average #Gènes	12.5	6.7	105	4.10	4.70	-	-

Note : Les meilleurs résultats sont indiqués en gras.

#Acc : Le taux d'exactitude de la classification à l'aide de LOOCV.

#Gènes : Le nombre de gènes utilisés.

Best : Le meilleur résultat trouvé dans dix exécutions indépendantes.

Average : La moyenne des dix expériences.

FRBPSO= A Fuzzy Rule Based Binary PSO.

HICATS=Hybrid Binary Imperialist Competition Algorithm and Tabu Search.

EPSO= An enhancement of binary particle swarm optimization.

TS-BPSO = A combination of tabu search and BPSO.

IBPSO = An improved binary PSO.

Dans cette section, nous avons présenté deux approches hybrides pour la sélection de gènes dans les données de puces à ADN. Les deux approches proposées consistent en une phase de présélection réalisée à l'aide des méthodes de filtrage Fisher et ReliefF pour FBPSO-SVM et RBPSO-1NN respectivement et une phase de recherche qui détermine un bon sous-ensemble de gènes pour la classification. Cette dernière est basée sur deux algorithmes, le BPSO et une méthode de recherche

locale (Backward generation). Ces approches visent à sélectionner un sous-ensemble optimal de gènes pertinents à partir d'un ensemble de données contenant des données redondantes, bruyantes ou non pertinentes.

Les résultats expérimentaux montrent que nos approches se comparent très favorablement aux méthodes de référence en termes d'exactitude de classification et de nombre de gènes sélectionnés.

3.4 Sélection de gènes via un nouvel algorithme hybride d'optimisation par colonies de fourmis pour la classification du cancer

L'algorithme d'optimisation par colonies de fourmis (en abrégé ACO) est une métaheuristique à base de population [41, 54]. Grâce à son efficacité, il a été utilisé pour résoudre plusieurs problèmes d'optimisation dans différents domaines.

Dans l'ACO, chaque fourmi présente une solution candidate au problème, les fourmis construisent des solutions approximatives de manière itérative (étape par étape). Le processus de construction de solutions peut être considéré comme un chemin (entre la maison et la source de nourriture des fourmis) sur un graphe. Le choix du meilleur chemin par les fourmis est influencé par les quantités de phéromones laissées dans ces chemins, et une information heuristique qui indique la qualité de la décision prise par une fourmi. L'ACO a été appliqué avec un grand succès dans le domaine de sélection de gènes [25, 37, 116, 166, 181, 182, 203]. D'autre part, plusieurs chercheurs ont également adopté des techniques basées sur la théorie de graphes pour sélectionner un sous-ensemble presque optimal d'un ensemble d'attributs [46, 170, 206]

Dans cette étude, nous proposons une approche hybride (MWIS-ACO-LS) pour résoudre le problème de sélection de gènes, cette méthode est basée sur la combinaison d'une nouvelle approche inspirée de la théorie de graphes pour la sélection de gènes (MWIS), dans laquelle nous cherchons à minimiser la redondance entre les gènes en considérant la corrélation entre ces derniers, et maximiser les scores de classement des gènes (Fisher), et un ACO modifié couplé à un algorithme de recherche locale (LS) ceci, en utilisant le classificateur $1NN$ pour mesurer de la qualité des sous-ensembles candidats. L'approche proposée n'a pas été étudiée auparavant par les chercheurs précédents. Et afin d'évaluer la méthode proposée, nous avons testé MWIS-ACO-LS sur dix ensembles de données de biopuces (Table 3.1 bien répliqués de dimensions élevées variant de 2308 à 12600 de gènes.

Cette étude a fait l'objet d'une publication dans le journal indexé (Computational and Mathematical Methods in Medicine "CMMM") [25]

3.4.1 Méthodes

3.4.1.1 Nouvelle approche basée sur la théorie des graphes pour la sélection de gènes

3.4.1.1.1 Notations

Dans ce travail, nous utilisons X pour désigner un ensemble de données (voir table 1.1) de M échantillons $X = (x_1; x_2; \dots; x_M)$, $x_i \in \mathbb{R}^N$. Nous utiliserons $\{g_1; g_2; \dots; g_N\}$, $g_i \in \mathbb{R}^M$ pour désigner les vecteurs de N gènes. $Y = (y_1; y_2; \dots; y_M)$ sont les étiquettes des classes.

La théorie des graphes donne un modèle abstrait pour représenter les relations entre deux ou

plusieurs éléments (sommets) dans un système donné. Soit $G = (V, E)$ un graphe non orienté où V est un ensemble fini non vide appelé l'ensemble de ses sommets, et E est l'ensemble des arêtes. Nous définissons un graphe pondéré par les sommets (G, W) comme un graphe G avec une fonction W (fonction de pondération des sommets) telle que $W(u) \in \mathbb{R}^{+*}$ pour tout $u \in V$ [104, 164]. L'ensemble indépendant de poids maximum (The maximum weight independent set « MWIS ») est l'un des problèmes d'optimisation les plus importants grâce à leurs nombreux domaines d'application [143]. En particulier, dans le problème de sélection de gènes, où nous pouvons transformer les données de puces à ADN en un graphe pondéré par les sommets (vertex-weighted graph) ; dans lequel un poids est attribué à chaque sommet.

(**Graphe de similarité entre les gènes** « Gene-similarity Graph »). Dans ce graphe, chaque gène peut être considéré comme un sommet et leur score de Fisher comme un poids de ce sommet. L'ensemble des arêtes représente l'existence d'une corrélation (relation) significative entre ces gènes, cette relation n'est rien d'autre que le degré d'association linéaire (corrélation de Pearson) entre ces derniers. Après avoir transformé les données des puces à ADN, nous essayons de trouver l'ensemble indépendant de poids maximum. Cet ensemble de gènes sera utilisé dans la deuxième étape de notre méthode proposée.

3.4.1.1.2 Construction du graphe de similarité entre les gènes

La construction du graphe de similarité entre les gènes nécessite la définition de quelques notions statistiques :

Nous commençons par le score de Fisher (voir le paragraphe 2.3.2.1.1, la page 49) pour calculer le poids de chaque sommet (gène). Puis, nous définissons la notion de corrélation linéaire entre deux variables, pour bien terminer la construction du graphe en question.

Coefficient de corrélation de Pearson Le coefficient de corrélation de Pearson est un indice reflétant une relation linéaire entre deux variables (gènes).

Soient g_1 et g_2 deux variables aléatoires, le coefficient de corrélation entre g_1 et g_2 est défini par :

$$\rho_{g_1, g_2} = \frac{\text{cov}(g_1, g_2)}{\sigma_{g_1} \sigma_{g_2}} \quad (3.6)$$

où :

$\text{cov}(g_1, g_2)$ est la covariance entre g_1 et g_2 , σ_{g_1} est l'écart type de g_1 et σ_{g_2} l'écart type de g_2 .

Le coefficient de corrélation peut prendre une gamme de valeurs de -1 à $+1$. Notons de plus ($r_{ij} = |\rho_{g_i, g_j}|$) pour designer la valeur absolue de la corrélation entre g_i et g_j .

Maintenant, nous pouvons définir la matrice d'adjacence $A_G = (a_{ij})_{N \times N}$ associée à (G, W) , avec des zéros sur la diagonale.

Où $a_{ij} = 1$ si $(i, j) \in E$ est une arête de G , et $a_{ij} = 0$ si $(i, j) \notin E$. Plus précisément : une valeur de 1 représente l'existence d'une relation entre g_i (ligne i) et g_j (colonne j), tandis qu'une valeur de 0 signifie l'inexistence de cette relation.

La création de la matrice d'adjacence A_G nécessite la définition de la matrice de corrélation absolue $R = (r_{ij})_{N \times N}$. Sur la base de cette matrice, nous remplissons A_G de la manière suivante :

Étant donnée r_0 dans $[0, 1]$, si $r_{ij} > r_0$ alors l'information mutuelle entre g_i et g_j est supposée élevée (c'est-à-dire que les deux sommets correspondants sont adjacents). Plus exactement, la matrice A_G est remplie selon la règle ci-dessous :

pour $i \neq j$:

$$a_{ij} = \begin{cases} 1, & \text{if } r_{ij} > r_0 \\ 0, & \text{sinon} \end{cases} \quad (3.7)$$

Où $r_0 \in [0, 1]$, est la valeur de corrélation minimale pour laquelle on considère deux gènes en relation. L'étude expérimentale réalisée dans notre méthode prouve que ($r_0 = 0.35$ se comporte bien avec les données de grande dimension).

Par exemple, si nous avons un ensemble de données composé de 7 gènes ; $\{g_1; g_2; \dots; g_7\}$, le tableau 3.5 montre la matrice de corrélation absolue correspondante à ces données.

Table. 3.5: Matrice de corrélation absolue

	g_1	g_2	g_3	g_4	g_5	g_6	g_7
g_1	1	0.59	0.19	0.45	0.1	0.24	0.67
g_2	0.59	1	0.36	0.3	0.11	0.07	0.66
g_3	0.19	0.36	1	0.31	0.24	0.06	0.29
g_4	0.45	0.3	0.31	1	0.49	0.81	0.12
g_5	0.1	0.11	0.24	0.49	1	0.72	0.66
g_6	0.24	0.07	0.06	0.81	0.72	1	0.57
g_7	0.67	0.66	0.29	0.12	0.66	0.57	1

Pour $r_0 = 0.35$ la matrice d'adjacence prend la forme (table 3.6) :

Table. 3.6: Matrice d'adjacence

	g_1	g_2	g_3	g_4	g_5	g_6	g_7
g_1	0	1	0	1	0	0	1
g_2	1	0	1	0	0	0	1
g_3	0	1	0	0	0	0	0
g_4	1	0	0	0	1	1	0
g_5	0	0	0	1	0	1	1
g_6	0	0	0	1	1	0	1
g_7	1	1	0	0	1	1	0

On définit le poids d'un sommet i (g_i), en utilisant le score de Fisher : $W(i) = F_i$. Un gène i avec un score élevé dans l'ensemble de données de puces à ADN correspond à un sommet avec un poids élevé en (G, W) . Ce poids donne des informations importantes sur la pertinence du gène pour les données. En effet, lorsqu'il y a deux gènes reliés par une arête dans G , on préfère de choisir le gène avec le grand poids. Sur la base des étapes définies précédemment, nous pouvons transformer une puce à ADN déterminée en un graphe pondéré par les sommets (Algorithme 6).

La figure.3.19 montre le graphe de similarité entre les gènes équivalent à la matrice d'adjacence (Table 3.6), nous associons à chaque gène (sommet) un poids en utilisant le score de Fisher :

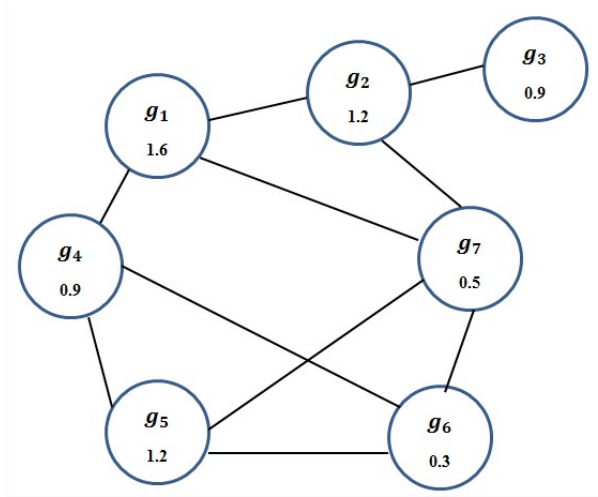


figure. 3.19: Un exemple de graphe de similarité entre les gènes.

Algorithme 6 Construction du graphe de similarité entre les gènes.

Entrées : Données de puces à ADN, r_0

Sortie : Graphe de similitude entre les gènes(G, W).

Début

Calculer le poids de chaque gène $W(g_i) = F_i$ en utilisant le score de Fisher.

Calculer la matrice de corrélation absolue $R = (r_{ij})_{N \times N}$.

Remplir la matrice d'adjacence $A_G = (a_{ij})_{N \times N}$ associée à G.

Créer le graphe de similarité entre les gènes (G, W).

Retourner (G, W).

Fin

Dans le cadre de la sélection de gènes pour la classification des cancers, les jeux de données de biopuces sont caractérisés par un très grand nombre de gènes. Ainsi, l'application d'un algorithme évolutif tel ACO directement sans passer par une étape de pré-traitement est très expansive, c'est là que les méthodes de filtrage deviennent si utiles afin d'extraire un sous-ensemble de gènes éventuellement informatifs, puis la métaheuristique évolutive est appliquée pour sélectionner le sous-ensemble optimal de gènes [182]. À titre d'exemples, le score de Fisher généralisé, le ReliefF et le BPSO sont combinés dans [24], la statistique du chi carré et un algorithme génétique sont utilisés dans [112]. Filtre de gain d'information et un algorithme mémétique dans [210], gain d'information et optimisation simplifiée en essaim (improved simplified swarm optimization (SSO))[109], Zhao et al. ont proposé une approche hybride en combinant le score de Fisher avec un AG et un PSO [208]. ReliefF, mRMR et GA [174]. Afin de pallier les inconvénients des méthodes Filter, nous proposons

une approche efficace basée sur des techniques de théorie des graphes pour sélectionner le premier sous-ensemble, cette méthode prend en compte les interactions possibles entre les gènes.

3.4.1.1.3 Sélection de gènes basée sur l'ensemble indépendant de poids maximum

Soit (G, W) un graphe non orienté pondéré par les sommets, où V est l'ensemble de ses sommets, E l'ensemble des arêtes, et W une fonction de pondération des sommets.

Pour chaque $v \in V$, nous définissons $N_G(v)$ le voisinage de v par :

$$N_G(v) = \{u \in V : u \neq v, (u, v) \in E\}$$

Un sous-ensemble $I \subseteq V$ est un ensemble indépendant de G s'il n'y a pas deux sommets adjacents dans I . Le MWIS est l'ensemble indépendant de poids maximum (le poids d'un sous-ensemble de sommets dans V est défini comme la somme des poids des sommets dans ce sous-ensemble [73]).

Nous remarquons que dans la majorité des méthodes filtres (univariées) la corrélation entre les gènes sélectionnés n'est pas prise en compte. Ceci implique la sélection de sous-ensembles avec un niveau de redondance élevé ce qui pénalise les performances du classifieur, en revanche, ces méthodes éliminent les gènes avec un score individuel faible, ignorant la possibilité qu'ils puissent devenir très pertinents lorsqu'ils sont combinés avec d'autres gènes [196]. Cela nous motive à proposer une approche basée sur la théorie de graphes pour surmonter ces problèmes. Dans la première étape de notre méthode, nous considérons le problème de sélection de gènes comme la recherche de l'ensemble indépendant de poids maximum dans le graphe de similarité entre les gènes (G, W) . Le choix de ce dernier est justifié par deux arguments : Premièrement, le terme poids maximum peut être traduit dans le contexte de la sélection de gènes comme la sélection d'un sous-ensemble avec une pertinence maximale. Deuxièmement, la notion "indépendant" assure le choix d'un sous-ensemble avec une redondance minimale, c'est-à-dire que dans ce sous-ensemble, il n'y a pas deux gènes fortement corrélés. De plus, l'ensemble sélectionné peut contenir des gènes avec un score faible. Par conséquent, la méthode proposée à ce stade donne un bon sous-ensemble de gènes pour appliquer un algorithme évolutif tel que l'ACO.

Le MWIS dans un graphe donné est un problème NP-difficile [71], et puisque dans notre cas le graphe de similarité entre les gènes est de grand taille (plusieurs milliers de sommets et d'arêtes), alors il est impossible de trouver une solution exacte à notre problème dans un temps raisonnable, pour cela nous proposons un algorithme glouton (heuristique) pour obtenir rapidement une solution approchée.

Les grandes lignes de cet algorithme sont présentées dans le pseudo-code suivant (Algorithme 7) :

Algorithme 7 Algorithme glouton pour approximer le MWIS.

Entrée : Graphe de similarité entre les gènes (G_0, W)

Sortie : Approximation du MWIS I .

Début

$I := \emptyset; i := 0; G_i := G;$

3: **Tant que** $V(G_i) \neq \emptyset$

Choisir le meilleur sommet v_i dans G_i , (c'est-à-dire le sommet avec le poids le plus élevé).

$I := I \cup \{v_i\};$

6: $G_{i+1} := G_i;$

$V(G_{i+1}) := V(G_i) \setminus \{N_{G_i}(v_i) \cup \{v_i\}\};$ % $N_{G_i}(v_i)$ est le voisinage de v_i .

$i=i+1;$

9: **Fin Tant que**

Retourner I .

Fin

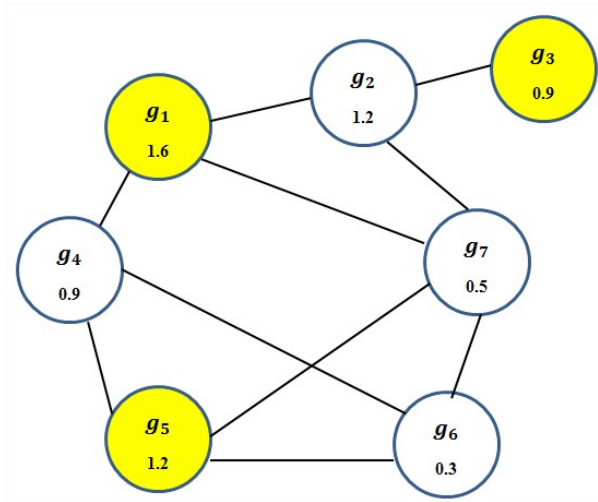


figure. 3.20: Exemple d'ensemble indépendant de poids maximum

Nous illustrons l'exécution de notre algorithme glouton (Algorithme 7) sur le graphe de la figure.3.19, formé par $\{g_1; g_2; \dots; g_7\}$. Dans la première itération, nous sélectionnons le meilleur gène g_1 ($W(g_1) = 1,6$), puis nous supprimons leur voisinage $\{g_2; g_4; g_7\}$, à l'itération suivante nous choisissons le meilleur gène g_5 dans le deuxième graphe composé de $\{g_3; g_5; g_6\}$. Dans la dernière itération, nous n'avons qu'un seul gène à choisir, c'est g_3 . Alors $I = \{g_1, g_3, g_5\}$ (figure.3.20) est un ensemble indépendant de poids maximum approximatif, nous pouvons remarquer que notre algorithme glouton donne le MWIS exact pour cet exemple.

3.4.1.2 Optimisation par colonies de fourmis (ACO) pour la sélection de gènes

Les algorithmes de colonies de fourmis (en anglais, ant colony optimization, ou ACO) sont des algorithmes inspirés du comportement des fourmis, ou d'autres espèces formant un superorganisme, et qui constituent une famille de métaheuristiques d'optimisation.

L'algorithme de l'ACO a été introduit comme méthode de résolution de problèmes d'optimisation au début des années 90 par Colorni, Dorigo et Maniezzo [41, 54] et développé par la suite dans [56, 57]. Initialement, l'ACO a été conçu pour résoudre le problème du voyageur de commerce en proposant le premier algorithme ACO : 'Ant System' (AS) [55]. Par la suite, d'autres applications qui ont été envisagées au début de l'histoire de l'ACO telles que l'affectation quadratique [68], la somme coloration d'un graphe [59], le routage des véhicules [35] la satisfaction des contraintes [178] et sélection de gènes [37, 116, 166, 181, 182, 203].

L'algorithme ACO est inspiré du comportement social des fourmis. Les fourmis artificielles utilisées dans ACO peuvent coopérer entre elles (en échangeant des informations via des phéromones) pour résoudre des problèmes difficiles, cela en construisant des solutions approximatives de manière itérative (étape par étape). Les solutions réalisables peuvent être considérées comme un chemin entre le nid et la source de nourriture des fourmis, la méthode de choix de ce dernier chemin est détaillée dans les paragraphes suivants.

3.4.1.2.1 ACO pour la sélection de gènes

Notons les gènes p comme $\{g_1; g_2; \dots; g_p\}$, pour adapter l'ACO au problème de sélection de gènes, un nouvel ACO est proposé; le chemin de chaque fourmi du nid à la nourriture est codé comme une chaîne binaire taille p où chaque bit est attaché à un gène; la sélection de la voie "1" signifie que le gène correspondant a été choisi. D'autre part, une voie "0" indique que le gène n'est pas sélectionné dans le sous-ensemble final. Pour $p = 6$, le codage de notre ACO modifié est présenté et expliqué dans la figure.3.21, les gènes sélectionnés dans notre exemple sont $g_1; g_3; g_4$ et g_6 .

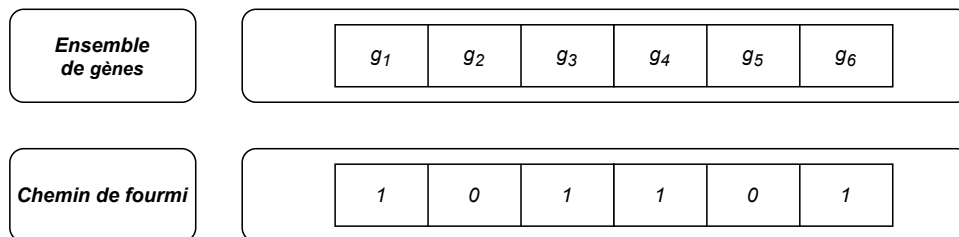


figure. 3.21: Un exemple illustré avec un sous-ensemble généré et une représentation de chemin.

Les fourmis cherchent à trouver le meilleur chemin qui maximise l'exactitude de la classification et minimise le nombre de gènes sélectionnés. La figure.3.22 décrit la procédure de section de gènes proposée sur notre ACO; chaque fourmi part du nid à la source de nourriture dans le but de trouver le meilleur chemin (meilleur sous-ensemble de gènes), la construction de ce chemin se fait étape par étape, à chaque étape i la fourmi décide d'ajouter le gène i au sous-ensemble candidat de gènes ou non, en fonction de la phéromone et des informations heuristiques attribuées à ce

gène (figure.3.22). La fourmi termine sa tournée par p étapes et produit un sous-ensemble de gènes sélectionnés lorsqu'elle atteint la source de nourriture.

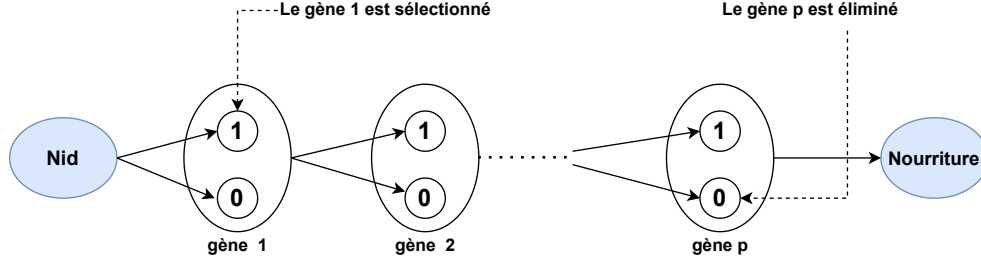


figure. 3.22: La procédure de sélection de gènes dans l'ACO modifié.

Comme indiqué précédemment, la rôle de chaque fourmi est de construire un sous-ensemble candidat de gènes en utilisant des informations heuristiques et des phéromones, ceci via une règle de décision probabiliste ; nous calculons la probabilité de sélectionner une voie comme ci-dessous :

$$p_{ij} = \frac{(\tau_{ij}^\alpha)(\eta_{ij}^\beta)}{(\tau_{i0}^\alpha)(\eta_{i0}^\beta) + (\tau_{i1}^\alpha)(\eta_{i1}^\beta)} \quad (i = 1, 2, \dots, p; j = 1, 0) \quad (3.8)$$

Où i représente le $i^{\text{ème}}$ gène, j prend une valeur de 1 ou 0 pour indiquer si le gène correspondant a été sélectionné ou non, τ_{ij} est l'intensité de la phéromone qui présente l'importance de la sélection du gène $i^{\text{ème}}$, et η_{ij} est l'heuristique reflétant la désirabilité de la sélection de ce gène ou non. α et β sont deux paramètres contrôlant l'importance relative de l'intensité de la phéromone par rapport à la visibilité : avec $\alpha = 0$, seule la visibilité (information heuristique) du gène est prise en compte, les fourmis se basent alors sur η_{ij} pour décider de sélectionner ou non un gène donné, dans ce cas, l'expérience de recherche précédente est perdue, il n'y a donc pas de coopération entre les fourmis. Au contraire, avec $\beta = 0$, seules les traînées de phéromones sont considérées. Pour éviter une convergence trop rapide de l'algorithme ACO, un compromis entre ces deux paramètres est nécessaire pour assurer la diversification et l'intensification de l'espace de recherche.

Information heuristique Le choix d'une bonne heuristique, qui sera utilisée en combinaison avec les informations sur les phéromones pour construire des solutions, est une tâche importante dans l'implémentation de l'ACO [113]. Dans notre ACO, cette information heuristique est utilisée pour indiquer la qualité d'un gène sur la base d'un algorithme de scoring.

Pour une fourmi donnée, l'information heuristique η_{i1} est l'opportunité d'ajouter le gène i au sous-ensemble de gènes sélectionnés. Nous définissons cette quantité à partir du score de Fisher F_i qui mesure la qualité de ce gène, et le nombre de gènes sélectionnés par la fourmi avant d'arriver au $i^{\text{ème}}$ gène N_s . η_{i1} est calculé comme ci-dessous :

$$\eta_{i1} = \frac{F_i}{1 + N_s} \quad (3.9)$$

Pour la valeur de η_{i0} , nous combinons la moyenne des scores de Fisher de tous les gènes et N_s , cela signifie que les fourmis ont tendance à choisir les petits sous-ensembles de gènes qui ont une

pertinence élevée :

$$\eta_{i0} = \frac{1}{p(1 + N_s)} \sum_{k=1}^{k=p} F_i \quad (3.10)$$

Mise à jour de la piste des phéromones Le but de la mise à jour des phéromones est d'augmenter les valeurs de phéromones associées aux bonnes solutions tout en réduisant celles associées aux mauvaises.

Les mises à jour des phéromones se font en deux étapes, une mise à jour locale et une mise à jour globale.

Une fois que la $k^{ème}$ fourmi termine la construction de son chemin, la phéromone dans tous les chemins sera mise à jour, il s'agit d'une mise à jour locale. La formule de ce mise à jour est décrite ci-dessous :

$$\tau_{ij} \leftarrow (1 - \rho_{loc})\tau_{ij} + \rho_{loc}\Delta\tau_{ij} \quad (3.11)$$

Où ρ_{loc} est un coefficient d'évaporation locale de la phéromone (de la traînée) ($0 < \rho_{loc} < 1$), et $\Delta\tau_{ij}$ est la quantité de phéromone déposée par la $k^{ème}$ fourmi donnée par :

$$\Delta\tau_{ij} = \begin{cases} CA_{1NN}(S) * \left(\lambda - \frac{\# \text{ Gènes}}{p} \right) & \text{si la fourmi } k \text{ utilise le chemin } j \text{ lorsqu'elle est arrivée au gène } i \\ 0 & \text{sinon} \end{cases} \quad (3.12)$$

Où S est la solution candidate créée par la $k^{ème}$ fourmi, CA_{1NN} est l'exactitude de classification (calculée à l'aide d'une validation croisée via le classifieur $1NN$) en utilisant l'ensemble S , $\#Gènes$ est le nombre de gènes sélectionnés dans S , et λ est un paramètre indiquant l'importance du nombre de gènes sélectionnés dans S ($1 \leq \lambda$).

À chaque itération T , lorsque toutes les fourmis terminent leur traversée, une mise à jour globale des quantités de phéromones est effectuée pour toutes les voies choisies par la meilleure fourmi (la meilleure solution candidate) au cours de l'itération T .

La mise à jour globale s'effectue comme suit :

$$\tau_{ij} \leftarrow (1 - \rho_{glob})\tau_{ij} + \rho_{glob}\Delta\tau_{ij}(T) \quad (3.13)$$

Où ρ_{glob} est un paramètre global d'évaporation de la phéromone, et $\Delta\tau_{ij}(T)$ est la quantité de phéromone déposée par la meilleure fourmi au cours de l'itération T donné par (3.12).

Pour éviter la stagnation de la recherche, la plage de pistes de phéromones possibles est limitée dans un intervalle $[\tau_{min}; \tau_{max}]$.

3.4.1.2.2 Fonction objectif

Afin de guider notre nouvel ACO vers un sous-ensemble de gènes de haute qualité, nous devons définir une "fonction de fitness" f . La qualité d'un sous-ensemble candidat peut être mesurée en combinant le nombre de gènes dans ce sous-ensemble (taille) et l'exactitude de la classification à l'aide d'un classificateur spécifique. Dans la sélection des gènes, l'objectif est de maximiser l'exactitude et de minimiser le nombre de gènes utilisés.

L'estimation de l'exactitude de la classification est mesurée par un classificateur donné en utilisant la règle de validation croisée (souvent LOOCV). Dans notre méthode, nous utilisons le classificateur $1NN$, qui est un cas particulier de KNN (avec $K = 1$).

Les expressions géniques dans les puces à ADN se représentent dans des échelles différentes, et puisque le classificateur KNN est influencé par la mesure des distances entre les échantillons, nous pensons à modifier notre $1NN$ en normalisant les données d'entraînement pour les transformer en une échelle commune, cette transformation se fait en fonction de la moyenne et de l'écart type de chaque gène, ces dernières valeurs sont utilisées pour la mise à l'échelle des données de test. La valeur de fitness d'une solution candidate S dans notre ACO est calculée comme suit :

$$f(S) = w_1 * CA_{1NN}(S) + (1 - w_1) * (1 - \frac{\#Gènes}{p}). \quad (3.14)$$

Où w_1 est un coefficient de pondération dans $[0, 1]$ qui contrôle l'agrégation des deux objectifs (en maximisant l'exactitude prédictive et en minimisant le nombre de gènes), $\#Gènes$ est le nombre de gènes sélectionnés dans S et p est le nombre total de gènes.

Nous mentionnons que " CA_{1NN} " présente l'exactitude calculée par le classificateur KNN , à l'aide d'une validation croisée de type (LOOCV).

3.4.1.2.3 Recherche locale

La recherche locale est utilisée pour améliorer les solutions trouvées par les fourmis et fournir de bonnes solutions dans un délai raisonnable. Dans ce but, nous nous inspirons du framework proposé dans [209], dans lequel une recherche locale basée sur une méthode filtre de classement est utilisée pour résoudre le problème de sélection de gènes.

Étant donné une solution candidate S générée par une fourmi, nous définissons X et Y comme le sous-ensemble des gènes sélectionnés et éliminés respectivement, X et Y sont tous deux classés selon le score de Fisher. Nous définissons en outre deux opérateurs de notre algorithme de recherche locale :

- **Add** : sélectionne un gène de Y en fonction de son classement et l'ajoute à S .
- **Del** : sélectionne un gène de X en fonction de son classement et le supprime de S .

La sélection d'un gène i de Y pour le déplacer vers S par l'opérateur **Add** est basée sur la Roulette, développée par Holland [91]. Soit $Y = \{g_1; g_2; \dots; g_{n_1}\}$ et $\{F_1; F_2; \dots; F_{n_1}\}$ les scores de Fisher associés. Alors la probabilité P_i de la sélection du gène g_i est définie comme :

$$P_i = \frac{F_i}{\sum_{j=1}^{j=n_1} F_j} \quad (3.15)$$

De même, pour l'opérateur **Del**, on définit la probabilité de sélectionner un gène g_i de $X = \{g_1; g_2; \dots; g_{n_2}\}$ pour le supprimer de S avec une probabilité :

$$P_i = \frac{\overline{F_i}}{\sum_{j=1}^{j=n_2} \overline{F_j}} \quad (3.16)$$

Où $\overline{F_j} = \max(F_1; \dots; F_{n_2}) - F_j$, pour $j = 1; \dots; n_2$, et $\{F_1; F_2; \dots; F_{n_1}\}$ sont les valeurs du score Fisher de $\{g_1; g_2; \dots; g_{n_2}\}$.

Sur la base des probabilités définies précédemment, nous pouvons remarquer que l'opérateur **Add** privilégie l'ajout de gènes avec un score élevé à S , en revanche, l'opérateur **Del** préfère supprimer de S les gènes avec un faible score.

Notre algorithme de Recherche Locale (la figure.3.23 et l'algorithme 8) est caractérisé par un nombre maximal d'opérations **Add** et **Del**, n_{add} et n_{del} respectivement et it_{max} un nombre maximum d'itérations consécutives sans amélioration de la meilleure solution. De plus, cet algorithme de recherche locale est général et efficace, par exemple, si on fixe n_{add} à 0, l'algorithme de recherche locale devient une **backward generation**, dans laquelle on essaie de supprimer les gènes non pertinents à chaque itération.

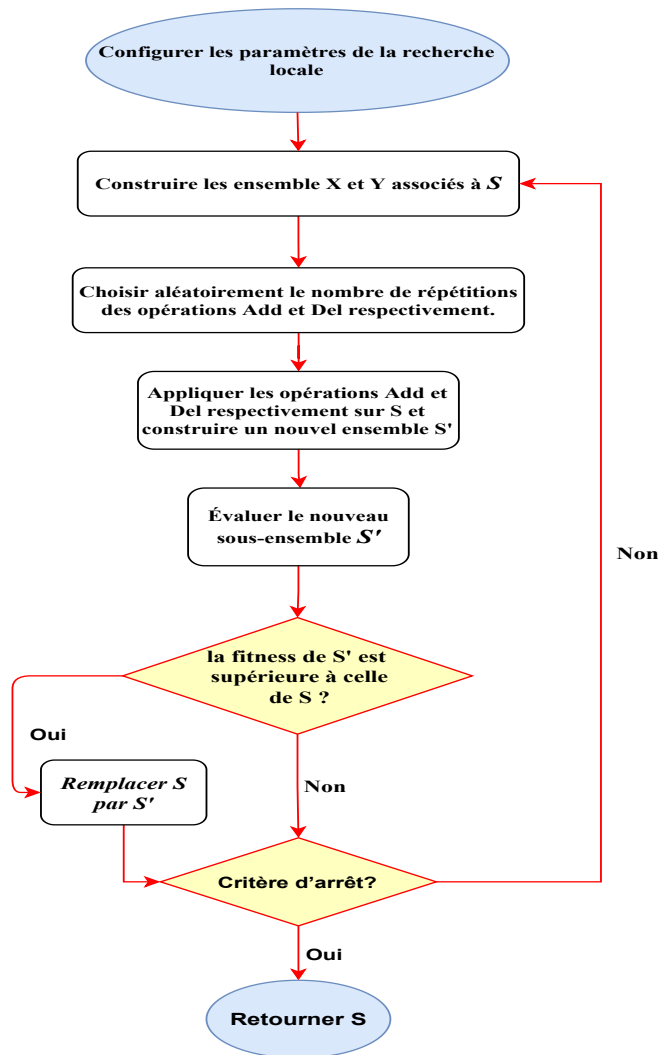


figure. 3.23: Organigramme de notre approche de recherche locale.

Algorithme 8 Algorithme de recherche locale pour la sélection de gènes.

Entrées : Données de puces à ADN ; it_{max} ; S une solution candidate donnée par une fourmi ;
 n_{add} le nombre maximal d'opérations **Add** ;
 n_{del} le nombre maximal d'opérations **Del** ;
Sortie : Une solution candidate S_{best} meilleure que S .

```

Début
k=0 ;
3: Tant que  $k < it_{max}$ 
    Déterminer les ensembles  $X$  et  $Y$ .
     $n_a = \lfloor rand * (n_{add} + 1) \rfloor$  ;  $n_d = \lfloor rand * (n_{del} + 1) \rfloor \% \lfloor . \rfloor$  la partie entière.
6: Répéter  $n_a$  fois l'opération Add à  $S$ .
    Répéter  $n_d$  fois l'opération Del à  $S$ .
    Créer une nouvelle solution candidate  $S'$ 
9: | if  $f(S') > f(S)$  then
    |    $S = S'$ 
    |    $k = -1$ 
12: | end if
    k=k+1
Fin Tant que
15:  $S_{best} = S$ 
    Retourner  $S_{best}$ .
Fin

```

3.4.1.3 Approche proposée pour la sélection de gènes (MWIS-ACO-LS)

Notre méthode hybride pour résoudre le problème de sélection de gènes est basée sur la combinaison d'approches filtres et enveloppes. Ceci en profitant du faible temps de calcul dans les filtres (MWIS) et de la haute qualité des sous-ensembles fournis par les méthodes enveloppes (ACO et la recherche locale (Local Search LS)). Le processus global de MWIS-ACO-LS peut être vu dans la figure.3.24.

Le processus commence par transformer l'ensemble de données initial en un graphe pondéré par les sommets (algorithme 6), où nous recherchons le MWIS, qui est bien connu comme un problème NP-difficile, et afin de trouver un ensemble de sommets presque optimal, nous avons proposé un algorithme glouton (algorithme 7). Le sous-ensemble de gènes sélectionnés à l'étape ultérieure est utilisé dans la deuxième étape de sélection, où un algorithme évolutif (ACO), combiné à un algorithme de recherche locale (LS) essaie de sélectionner le nombre minimum de gènes qui donne une exactitude de classification maximale via 1NN. À ce stade, les fourmis artificielles coopèrent pour construire un sous-ensemble de gènes de haute qualité basé sur les règles de transition déjà présentées. Aussi, une Recherche Locale (algorithme 8) est proposée pour aider les fourmis à obtenir de bons résultats dans un délai raisonnable.

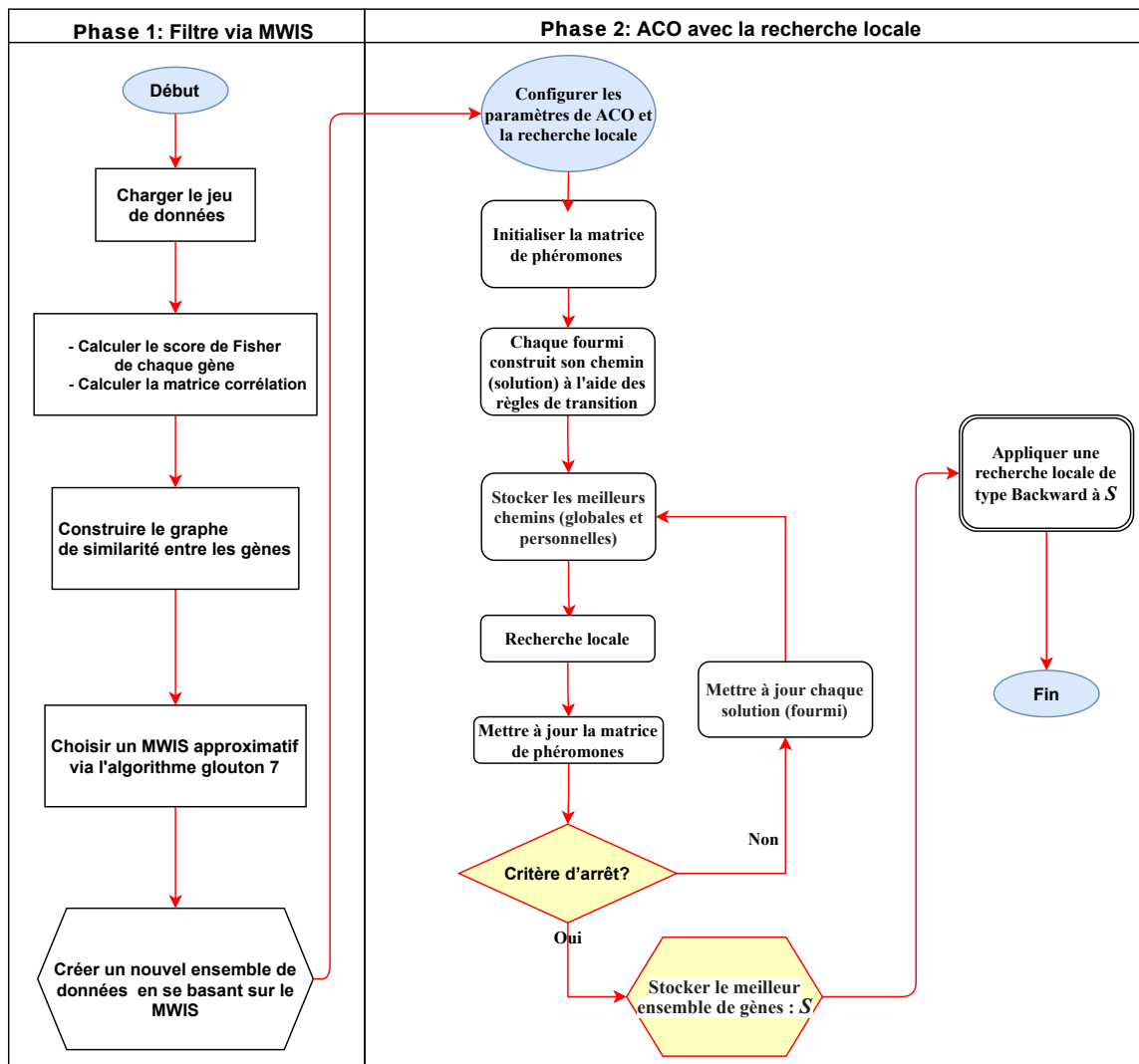


figure. 3.24: Organigramme de nos approches proposées pour la sélection de sous-ensembles de gènes dans les données de puces à ADN.

Le pseudo-code de notre méthode proposée est présenté comme suit :

Algorithme 9 Approche proposée (MWIS-ACO-LS)

Entrées : Données de puces à ADN ; n_{max} ; m ; α ; β ; ρ_{loc} ; ρ_{glob} ; λ ; τ_{i0} ; τ_{i1} ; τ_{min} ; τ_{max} ; w_1 .

Sortie La meilleure solution candidate globale S_{gbest} .

Début

phase 1 : La sélection du premier sous-ensemble de gène

3: **Étape 1 :** Utilisez l'algorithme 6 pour construire le graphe de similarité entre les gènes.

Étape 2 : Appliquer l'algorithme glouton (Algorithm 7) pour sélectionner un sous-ensemble initial de gènes (MWIS).

phase 2 : L'application de l'ACO au sous-ensemble de gène sélectionné dans la première phase

6: **Etape 1 : ACO combiné avec la recherche locale**

Initialise la matrice de phéromones par unités.

for $T = 1$; $T \leq n_{max}$; $T++$ **do**

9: | **for** $i = 1$; $i \leq m$; $i++$ **do**

construit le chemin (solution candidate S) de la fourmi en se basant sur la règle de décision probabiliste.

Calculer la fitness de la solution candidate en utilisant LOOCV.

12: | | **if** $i == 1$ **then**

$S_{best} = S$

| | **end if**

15: | | **if** $f(S_{best}) \leq f(S)$ **then**

$S_{best} = S$

| | **end if**

18: | Faire une mise à jour locale des phéromones basée sur S .

| **end for**

Appliquer la recherche locale (algorithme 8) à S_{best} .

21: Faire une mise à jour globale des phéromones basée sur S_{best} .

end for

Trouver la meilleure solution globale S_{gbest}

24: **Étape 2 : Appliquer une génération en arrière (backward generation) à S_{gbest} .**

Retourner S_{gbest} .

Fin

3.4.1.4 Analyse de complexité de MWIS-ACO-LS

Supposons que N soit le nombre de gènes originaux et M soit le nombre d'échantillons. Notre méthode se divise en trois phases principales :

Phase 1 : Dans la première étape (Algorithme 6) les poids des gènes sont évaluées à l'aide du score de Fisher, ainsi la complexité est de $O(NM)$. De plus, les valeurs de corrélation absolues entre chaque paire de gènes sont calculées, la complexité est donc de $O(N^2M)$. Et enfin, pour le remplissage de la matrice d'adjacence A_G (implicitement la construction du

graphe de similarité entre les gènes), la complexité est de $O(N^2)$. Dans la deuxième étape de cette phase (Algorithme 7), le poids de chaque sommet est déjà défini, alors nous pouvons supposer de manière conservatrice qu'à chaque itération nous supprimons uniquement le sommet lui-même pour obtenir une complexité de $O(N)$. Par conséquent, la complexité temporelle globale de cette phase (MWIS) est de $O(NM + N^2M + N^2 + N) = O(N^2M)$.

Phase 2 : Pour la deuxième phase, l'ACO et l'algorithme de recherche locale fonctionnaient en fonction de p le nombre de gènes sélectionnés dans la première étape généralement ($p \ll N$). À cette étape, la fitness de chaque sous-ensemble de gènes candidats est calculée en utilisant LOOCV (Leave-One-Out-Cross-Validation) et le 1NN comme classificateur. Analysons maintenant la complexité du calcul de fitness en utilisant 1NN (LOOCV) : nous calculons la distance entre l'unique échantillon de l'ensemble de test et chaque échantillon de l'ensemble d'apprentissage, nécessitant $O(p(M - 1))$, ce processus est répété M fois, donc le calcul de fitness nécessite $O(p(M - 1)M)$.

A chaque itération de notre ACO, chaque fourmi des m fourmis commence de g_1 à g_p en passant par tous les p gènes, puis la construction d'un chemin par une fourmi a $O(p)$, et chaque chemin est évalué par LOOCV, ce processus est répété n_{max} fois par les m fourmis, de plus, la mise à jour des valeurs de phéromone a $O(pn_{max})$, donc, la complexité de calcul globale de ACO sans recherche locale est : $O(pn_{max} + pn_{max}mp(M - 1)M) = O(n_{max}mp^2(M - 1)M)$. Concernant l'algorithme de recherche locale ; pour le LS utilisé dans la deuxième phase (ligne 20 Algorithme 9) ; répété n_{max} fois, le temps de complexité est : $O((n_{add} + n_{del})it_{max}n_{max}p(M - 1)M)$.

Généralement, $(n_{add} + n_{del})it_{max} \simeq m$, donc le temps de complexité total de la deuxième phase est :

$$O(n_{max}mp^2(M - 1)M).$$

Phase 3 : Pour la dernière phase (ligne 24 Algorithme 9), la complexité de la génération arrière (backward generation) est :

$$O(it_{max}n_{del}p(M - 1)M).$$

Par conséquent, la complexité temporelle totale de la méthode proposée MWIS-ACO-LS est :

$$O(it_{max}n_{del}p(M - 1)M + n_{max}mp^2(M - 1)M + N^2M) = O(n_{max}mp^2M^2 + N^2M)$$

3.4.2 Expérimentations et résultats

Cette sous-section présente les performances de notre approche (MWIS-ACO-LS) sur dix ensembles de données de classification d'expression génique bien connus Table 3.1, et nous comparons nos résultats avec ceux de l'état de l'art. Le réglage des paramètres et les résultats numériques seront décrits dans les sous-sections suivantes.

La mise en œuvre de l'approche (MWIS-ACO-LS) est faite sous Matlab R2017a.

En ce qui concerne le classifieur KNN, nous avons choisi une fonction prédéfinie dans Matlab. De même pour le classificateur SVM utilisé dans la comparaison, un classificateur linéaire binaire prédéfini a été choisi. De plus, nous avons développé un classificateur SVM multiclasse basé sur la « stratégie un contre tous ». D'autre part, nous avons régularisé la fonction de coût associée au classificateur de la régression logistique (LR) par deux pénalités, la première est LASSO (L_1)

et la seconde est la régularisation Elastic-net. La minimisation des fonctions de coût utilisées sur $LR - L_1$ et **LR-Elasticnet** est assurée par l'algorithme de descente de gradient stochastique (SGD) implémenté dans le package **Scikit-learn** [145]. Les paramètres initiaux expérimentaux sont donnés dans le tableau 3.7.

De plus, dans cette étude, nous utilisons une validation croisée de type (LOOCV) pour mesurer la qualité des sous-ensembles de gènes candidats et pour comparer nos résultats avec ceux de l'état de l'art.

Pour évaluer notre approche, nous avons choisi dix jeux de données (ADN microarray) concernant la reconnaissance des cancers Table 3.1, accessibles au public, facilement accessibles. De plus, ces ensembles de données sont utilisés dans plusieurs travaux de classification supervisés ; notamment dans les articles de la section " Comparaison avec les algorithmes de l'état de l'art ". Et comme notre méthode est conçue pour les puces à haute dimension, tous ces ensembles de données sont caractérisés par des milliers de gènes allant de 2308 à 12600.

3.4.2.1 Paramètres

Nous notons que nos approches ont été exécutées sur un ordinateur portable Acer Aspire 7750g avec processeur Intel Core I5 2,30 GHz et 8 Go de RAM, sous système exécutant Windows 7 (64 bits).

Plusieurs tests ont été effectués afin d'obtenir un paramétrage approprié, en effet, un ensemble de valeurs initiales pour les paramètres est fixé, puis nous modifions la valeur d'un paramètre pour différentes séries jusqu'à ce que les solutions ne puissent pas être améliorées. Le processus d'ajustement a été répété pour chaque paramètre jusqu'à ce que les solutions ne puissent pas être améliorées. Ce processus est basé sur un ensemble de données de classification du cancer. Le tableau 3.7 représente les paramètres de l'approche proposée.

Table. 3.7: Paramètres utilisés pour les expériences.

Paramètres communs pour MWIS – ACO – LS		Valeur
Paramètres communs pour MWIS		
r_0	la corrélation pour la création d'une arête	0.35
Paramètres communs pour ACO		
m	taille de la population	30
n_{max}	le nombre d'itération	100
α	influence de la phéromone	1.2
β	information heuristique	0.2
ρ_{loc}	évaporation locale de phéromone	0,002
ρ_{glob}	évaporation locale de phéromone	0,06
λ	facteur de mise à jour de la phéromone	1.6
τ_{i0}	la phéromone initiale des voies 0	1.0
τ_{i1}	la phéromone initiale de la voie 1	1.0
τ_{min}	la phéromone inférieure	0,05
τ_{max}	la phéromone supérieure	1.4
w_1	le coefficient de poids dans la fonction fitness	0.99
Paramètre commun pour l'algorithme de recherche locale		
it_{max}	le nombre maximal d'itérations sans amélioration	5
n_{add}	le nombre maximal d'opérations Add	3
n_{del}	le nombre maximal d'opérations Del	5
Paramètre commun pour l'algorithme de génération arrière (Backward)		
it_{max}	le nombre maximal d'itérations sans amélioration	20
n_{add}	le nombre maximal d'opérations Add	0
n_{del}	le nombre maximal d'opérations Del	2
Paramètre commun pour le classificateur KNN		
K	le nombre de voisins	1
la distance utilisée	la distance euclidienne	
Paramètres communs pour le modèle de régression logistique (LR)		
α_{LR}	Le terme de régularisation constant	0,0001
$l1_{ratio}$	Le poids donné à L_1 dans l'Elastic Net	0.5

3.4.2.2 Résultats et comparaisons

Premièrement, afin de limiter l'espace de recherche et d'accélérer la vitesse de convergence de notre approche, le premier sous-ensemble de gènes a été sélectionné sur la base d'un nouvel algorithme glouton de la théorie des graphes pour la sélection de (MWIS), puis un ACO-1NN modifié couplé à un algorithme de recherche locale (LS) a été appliqué pour trouver un sous-ensemble de gènes optimal; la qualité d'un sous-ensemble candidat est mesurée par le classifieur 1NN à l'aide de LOOCV.

Les objectifs des expérimentations menées sur les dix jeux de données de (puces à ADN) sont les suivants : tester l'effet de la sélection de gènes sur l'amélioration de l'exactitude de la classification, valider la méthode proposée et vérifier son efficacité.

3.4. SÉLECTION DE GÈNES VIA UN NOUVEL ALGORITHME HYBRIDE D'OPTIMISATION PAR COLONIES DE FOURMIS POUR LA CLASSIFICATION DU CANCER

Table. 3.8: Comparaison entre LR, SVM, 1NN, MWIS-1NN et MWIS-ACO-LS (LOOCV).

jeu de données	Performances	SVM	LR- L_1		LR-Elasticnet		1NN	MWIS	MWIS-ACO		MWIS-ACO-LS	
			Avg	Best	Avg	Best			Avg	Best	Avg	Best
11_Tumors	Accuracy (%)	85,63	88,22	93,1	86,38	88,22	74,14	67,24	94,90	96	99,14	99,42
	Gènes	12533	-	-	-	-	12533	1308	463,00	460	166,9	101
	Time (min)	-	-	-	-	-	-	0,82	91,33	-	123,2	-
9_Tumors	Accuracy (%)	38,33	35,00	50,00	29,50	38,33	40,00	60,00	98,83	100	100,00	100,00
	Gènes	5726	-	-	-	-	5726	263	90,10	83	51	40
	Time (min)	-	-	-	-	-	-	0,34	21,3	-	34,48	-
Brain_Tumor1	Accuracy (%)	88,89	85,67	88,89	85,44	88,89	85,56	80,00	96,56	100,00	99,22	100,00
	Gènes	5920	-	-	-	-	5920	246	55,90	46	22,9	19
	Time (min)	-	-	-	-	-	-	0,22	29,14	-	45,81	-
Brain_Tumor2	Accuracy (%)	70,00	27,20	32,00	29,20	36,00	60,00	48,00	95,40	100,00	99,40	100,00
	Gènes	10367	-	-	-	-	10367	110	22,40	18	11,1	11
	Time (min)	-	-	-	-	-	-	0,55	17,89	-	27,13	-
Leukemia1	Accuracy (%)	97,22	91,81	94,44	92,64	95,83	83,33	66,67	100,00	100,00	100,00	100,00
	Gènes	5327	-	-	-	-	5327	297	63,00	56	9,4	5
	Time (min)	-	-	-	-	-	-	0,26	25,94	-	43,77	-
Leukemia2	Accuracy (%)	97,22	91,81	95,83	91,67	95,83	86,11	73,61	100,00	100,00	100,00	100,00
	Gènes	11225	-	-	-	-	11225	203	45,80	42	13,9	11
	Time (min)	-	-	-	-	-	-	0,44	23,85	-	37,67	-
Lung_Cancer	Accuracy (%)	95,07	91,67	94,10	91,03	93,60	87,68	90,15	98,42	99,01	98,92	99,51
	Gènes	12600	-	-	-	-	12600	602	180,00	183	34,8	36
	Time (min)	-	-	-	-	-	-	0,82	74,61	-	107,3	-
SRBCT	Accuracy (%)	100,00	97,59	100,00	97,10	98,79	85,54	91,57	100,00	100,00	100,00	100,00
	Gènes	2308	-	-	-	-	2308	109	15,60	15	7,6	6
	Time (min)	-	-	-	-	-	-	0,14	23,48	-	38,24	-
Prostate_Tumor	Accuracy (%)	92,16	81,47	88,23	80,69	83,33	82,35	79,41	98,24	99,04	99,12	100,00
	Gènes	10509	-	-	-	-	10509	193	47,30	43	20,3	21
	Time (min)	-	-	-	-	-	-	0,39	29,2	-	46,59	-
DLBCL	Accuracy (%)	97,40	91,95	97,40	91,43	94,80	84,42	87,01	100,00	100,00	100,00	100,00
	Gènes	5469	-	-	-	-	5469	147	25,7	22	7,2	6
	Time (min)	-	-	-	-	-	-	0,17	27,18	-	43,63	-

Note : Les meilleurs résultats sont indiqués en gras.

Remarque : SVM, 1NN et MWIS étant de nature déterministe, l'exactitude de classification est calculée en une seule exécution.

Accuracy : L'exactitude de la classification utilisant LOOCV.

Gènes : Le nombre de gènes utilisés dans la classification.

Best : Le meilleur résultat trouvé dans les dix exécutions.

Avg : La moyenne des dix exécutions.

Time : Le temps d'exécution en minutes.

SVM : Le classificateur de machines à vecteurs de support utilisant un noyau linéaire.

LR- L_1 : Le classifieur de régression logistique avec la régularisation LASSO.

LR-Elasticnet : Le classifieur de régression logistique avec la régularisation élastique net.

1NN : le classifieur de 1-le plus proche voisin.

MWIS : l'ensemble indépendant de poids maximum pour la sélection des gènes.

MWIS-ACO : Notre méthode de sélection combinant MWIS et ACO sans utiliser la recherche locale (LS).

MWIS-ACO-LS : Notre méthode de sélection améliorée combinant MWIS et ACO et la recherche locale (LS).

Compte tenu de la nature non déterministe de notre approche et du classificateur SGDlogistic, dix exécutions indépendantes ont été effectuées pour chaque ensemble de données afin d'obtenir un résultat plus fiable.

Le tableau 3.8 montre les résultats obtenus en utilisant une nouvelle approche basée sur la théorie des graphes (MWIS) pour la sélection de gènes, puis, en utilisant la méthode MWIS-ACO où nous

appliquons au sous-ensemble sélectionné par MWIS et l'algorithme ACO, et enfin en utilisant notre nouvelle méthode améliorée MWIS- ACO-LS, où l'ACO est couplé à une méthode de recherche locale (LS) ; l'exactitude de la classification dans MWIS, MWIS-ACO et MWIS-ACO-LS est calculée à l'aide du classifieur 1NN, d'autre part, ces méthodes sont comparées aux classifieurs SVM, 1NN et regression logistique pénalisée sans sélection pour démontrer l'utilité de notre approche de sélection. Nous analysons nos résultats de trois manières :

- L'exactitude de la classification.
- Le nombre de gènes utilisés dans la classification.
- Le temps d'exécution.

3.4.2.3 Discussion

Tout d'abord, nous commençons par l'analyse du temps d'exécution de nos méthodes proposées : nous pouvons remarquer que le temps d'exécution est approprié à l'analyse de complexité ; dans l'approche basée sur le filtre MWIS, le temps d'exécution est faible, mais l'exactitude de classification n'est pas bonne, puisque la sélection est indépendante du classifieur. Alors que dans MWIS-ACO et MWIS-ACO-LS, le temps d'exécution est important en raison de la nature de la méthode enveloppe où nous faisons appel au classifieur 1NN à chaque évaluation, mais l'exactitude de la classification est élevée. Passant maintenant à l'analyse des différentes étapes de notre méthode proposée MWIS-ACO-LS, De (figure.3.25, figure.3.26 et Tableau 3.8), nous pouvons remarquer que le rôle de l'ACO est d'améliorer l'exactitude de la classification et de réduire le nombre de gènes utilisés. De plus, la recherche locale a un rôle primordial dans l'affinement des solutions candidates apportées par les fourmis en réduisant le nombre de gènes, tout en conservant l'exactitude de classification prouvée par les fourmis.

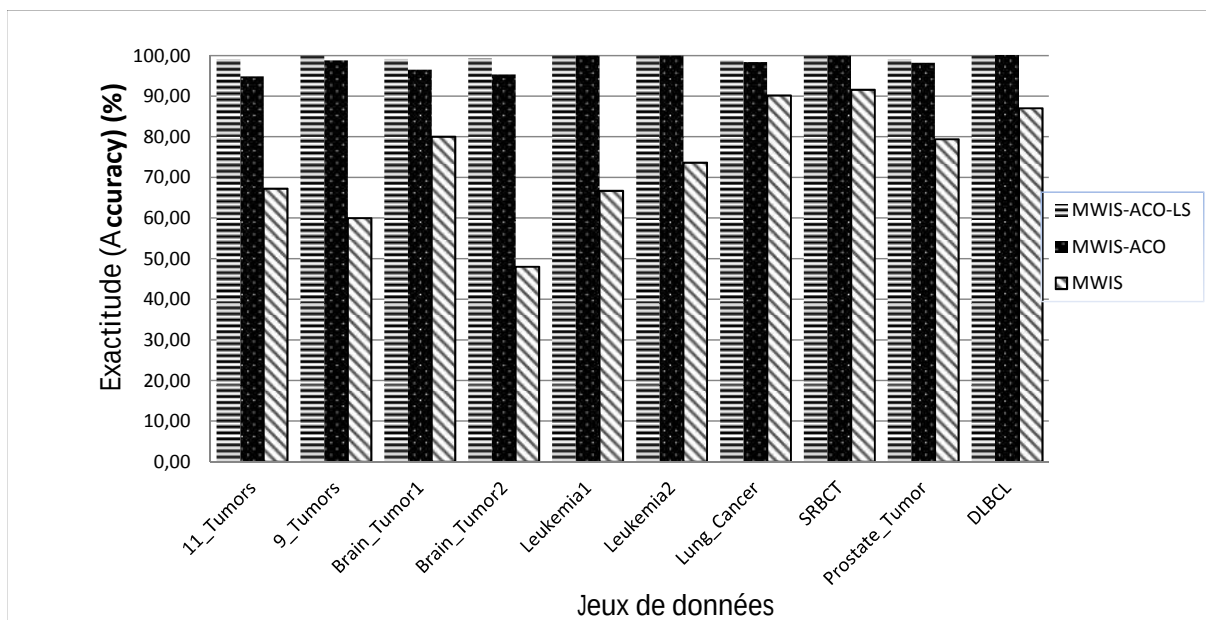


figure. 3.25: Comparaison de l'exactitude de classification entre MWIS, MWIS-ACO et MWIS-ACO-LS (pour MWIS-ACO et MWIS-ACO-LS nous prenons la solution moyenne).

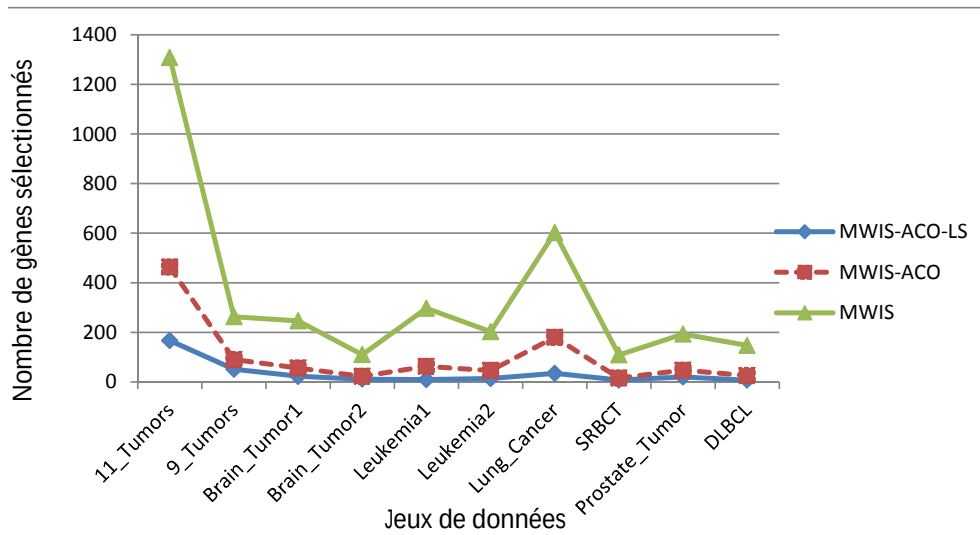


figure. 3.26: Comparaison du nombre de gènes utilisés dans la classification entre MWIS, MWIS-ACO et MWIS-ACO-LS (pour MWIS-ACO et MWIS-ACO-LS nous prenons la solution moyenne).

L'approche proposée (MWIS-ACO-LS) tire son efficacité de l'amélioration remarquable de l'exactitude de la classification et de la réduction du nombre de gènes utilisés dans la classification (indiqués en gras dans le tableau 3.8), dans tous les ensembles de données (la figure. réffigac).

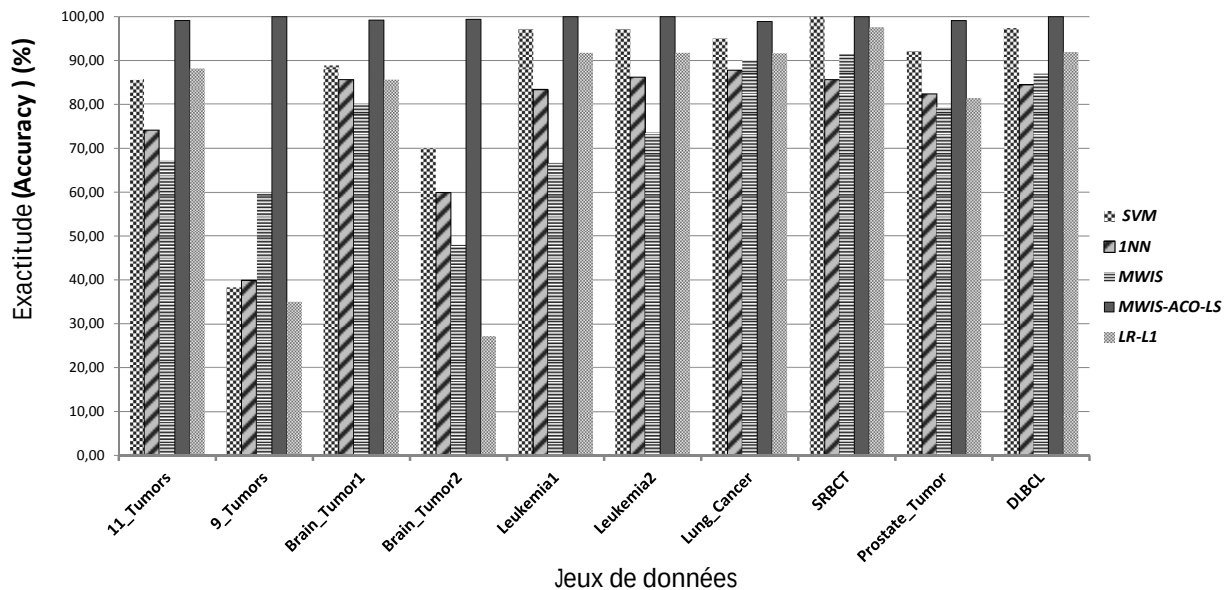


figure. 3.27: Comparaison de l'exactitude de classification entre les cinq méthodes (pour MWIS, MWIS-ACO-LS et $LR - L_1$ nous prenons la solution moyenne).

Les méthodes "MWIS", "MWIS-ACO" et "MWIS-ACO-LS" sélectionnent un sous-ensemble ré-

duit de gènes informatifs par rapport au sous-ensemble original de gènes dans les ensembles de données.

À partir du tableau 3.8 et de la figure.3.27, nous pouvons observer que MWIS surmonte les résultats obtenus par le classificateur 1NN pour les ensembles de données "9_Tumors", "Lung_Cancer", "SRBCT" et "DLBCL" qui est incroyable parce que le rôle de l'algorithme MWIS est juste de trouver un sous-ensemble candidat de gènes pour appliquer notre ACO modifié, ce sous-ensemble peut contenir des gènes faibles et le processus de sélection dans cette méthode est indépendant du classifieur utilisé.

Sur la base des expériences et de l'application de notre approche sur dix ensembles de données concernant la reconnaissance du cancer, nous pouvons observer que la méthode (MWIS-ACO-LS) surpasse les quatre algorithmes en termes d'exactitude de classification et de nombre de gènes utilisés dans la classification. L'amélioration des performances est plus significative pour 9_Tumors; nous sommes passés d'une exactitude de classification inférieure à 60 % à une classification parfaite utilisant seulement 40. Ainsi, nous pouvons conclure à partir du Tableau 3.8 et de la figure.3.27 que MWIS-ACO-LS peut sélectionner avec succès un petit sous-ensemble de gènes qui donne une exactitude de classification élevée. Pour tous les jeux de données, l'approche "MWIS-ACO-LS" a atteint une grande exactitude de classification, plus exactement; une classification supérieure à 99,42% n'utilisant qu'un petit sous-ensemble de gènes. De plus, MWIS-ACO-LS a donné une exactitude parfaite 100 % pour la majorité des ensembles de données :(9_Tumors; Brain_Tumor1; Brain_Tumor2; Leukemia1; Leukemia2; SRBCT; Prostate_Tumor; DLBCL) en utilisant seulement 5 gènes pour Leukimia1 et 6 gènes pour l'ensemble de données SRBCT et DLBCL.

Cependant, en ce qui concerne l'approche MWIS basée sur certains principes de la théorie des graphes, nous remarquons que le sous-ensemble de gènes sélectionné par cette méthode donne une exactitude de classification acceptable selon le nombre de gènes utilisés. Cela renvoie à la procédure utilisée pour la construction de ce sous-ensemble dans lequel nous donnons aux gènes ayant des scores faibles la possibilité d'être sélectionnés. Comme détaillé dans les sous-sections précédentes, notre méthode de deux phases MWIS-ACO-LS commence par sélectionner un petit sous-ensemble initial de gènes contenant les informations principales à l'aide de MWIS, puis, nous appelons notre ACO modifié combiné à un algorithme de recherche local, dans cette deuxième phase, notre algorithme essaie de trouver le plus petit sous-ensemble de gènes qui donne la plus grande exactitude de classification, le tableau 3.8 montre comment la deuxième étape joue un rôle crucial dans l'augmentation de la classification pour tous les ensembles de données, en particulier (Brain_Tumeur2, 9_Tumeurs, 11_Tumeurs et Leukimia1) où les résultats sont significativement différents (grande amélioration d'exactitude de la classification).

Dans les figures (3.29, 3.31, 3.33 et 3.35), l'axe des abscisses exprime le nombre de générations dans la deuxième phase de MWIS-ACO-LS, et l'axe des ordonnées exprime l'exactitude de classification de la meilleure solution candidate à chaque itération. Ceci est fait pour la moyenne de toutes les solutions et la meilleure solution trouvée pour les ensembles de données "9_Tumors, Brain_Tumor1, Brain_Tumor2 and Leukemia1". Ces chiffres montrent clairement que l'utilisation de notre ACO modifié et de l'algorithme de recherche locale jouent un rôle crucial dans l'amé-

lioration de l'exactitude de la classification. Comme nous pouvons le remarquer sur ces figures, la différence entre la meilleure solution et la solution moyenne n'est pas grande. Par conséquent, MWIS-ACO-LS possède une vitesse de convergence plus rapide et atteint rapidement la solution optimale.

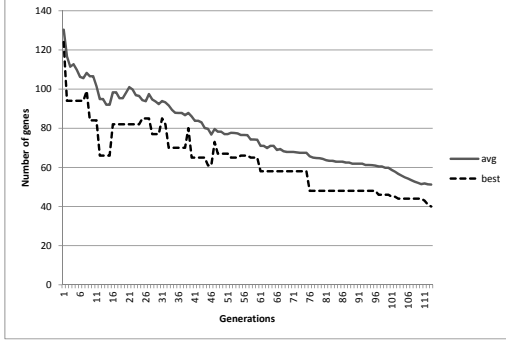


figure. 3.28: Comparaison de l'évolution du nombre de gènes utilisés pour "9_Tumors".

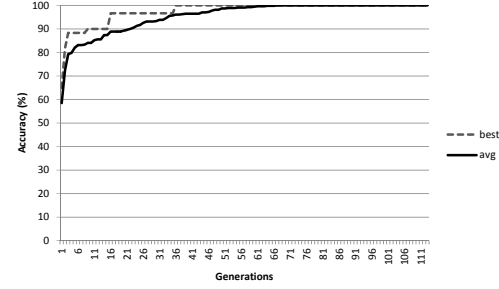


figure. 3.29: Comparaison de l'évolution de l'exactitude de la classification pour "9_Tumors".

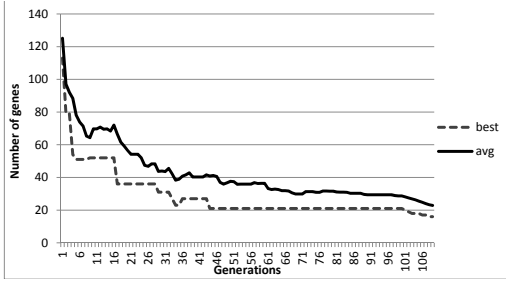


figure. 3.30: Comparaison de l'évolution du nombre de gènes utilisés pour "Brain_Tumor1".

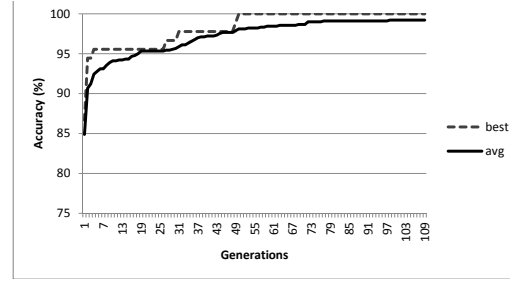


figure. 3.31: Comparaison de l'évolution de l'exactitude de la classification pour "Brain_Tumor1".

Dans les figures (3.28, 3.30, 3.32, et 3.34), nous montrons l'évolution du nombre de gènes sélectionnés sur l'axe des ordonnées relatif au nombre de générations (l'axe des abscisses) pour les ensembles de données "9_Tumors, Brain_Tumor1, Brain_Tumor2 et Leukemia1". Ces chiffres illustrent le rôle de notre algorithme enveloppe basé sur l'ACO dans la réduction du nombre de gènes. De plus, la deuxième phase de notre méthode basée sur l'ACO modifié et l'algorithme de recherche locale joue un rôle clé dans l'augmentation de l'exactitude de la classification et la minimisation du nombre de gènes utilisés. En effet, à chaque itération, l'ACO vise à identifier le sous-ensemble presque optimal de gènes candidats (appelé la meilleure fourmi) qui maximise la fonction objectif et, une fois le sous-ensemble en question trouvé, notre algorithme de recherche locale est appelé pour améliorer l'exactitude ou réduire le nombre de gènes utilisés tout en conservant la qualité de classification trouvée précédemment. Après 100 générations de l'algorithme ACO, nous appliquons un algorithme de recherche locale (sélection en arrière "backward generation") sur la meilleure solution trouvée pour réduire le nombre de gènes utilisés (figures 3.28, 3.30, 3.32, et 3.34).

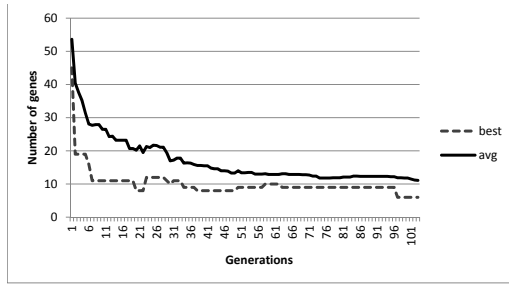


figure. 3.32: Comparaison de l'évolution du nombre de gènes utilisés pour "Brain_Tumor2".

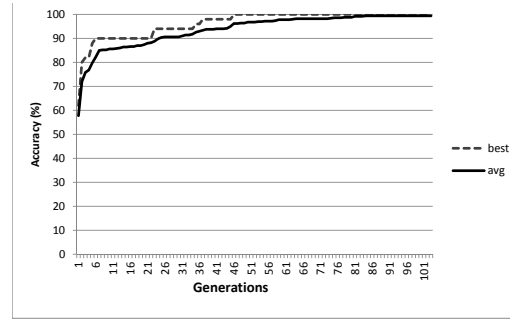


figure. 3.33: Comparaison de l'évolution de l'exactitude de la classification pour "Brain_Tumor2".

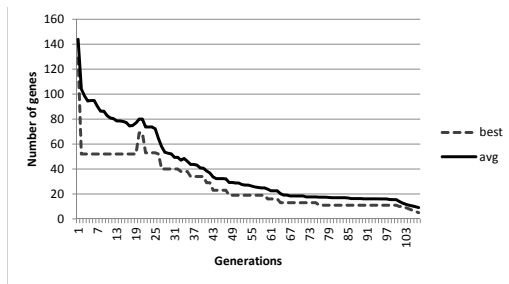


figure. 3.34: Comparaison de l'évolution du nombre de gènes utilisés pour "Leukimia1".

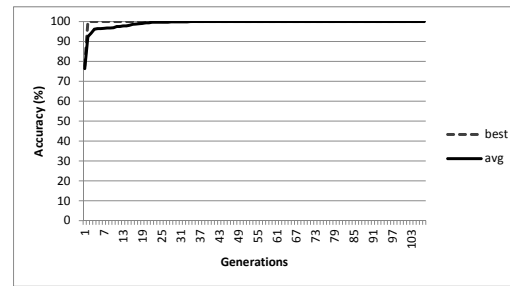


figure. 3.35: Comparaison de l'évolution de l'exactitude de la classification pour "Leukimia1".

Par la suite, une analyse statistique a été réalisée à l'aide du test statistique non paramétrique de Kruskal-Wallis pour évaluer nos résultats et tester la significativité de la différence dans les résultats (exactitude) obtenus par notre approche.

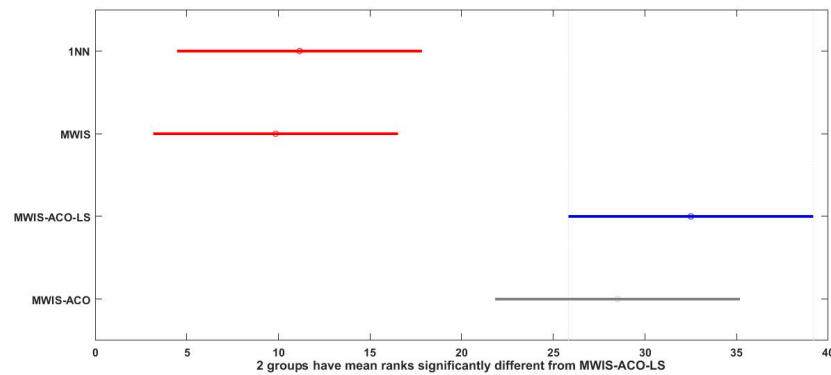


figure. 3.36: Le résultat du test de kruskal-wallis entre le MWIS-ACO-LS, 1NN et MWIS sur les jeux de données (exactitude de la classification).

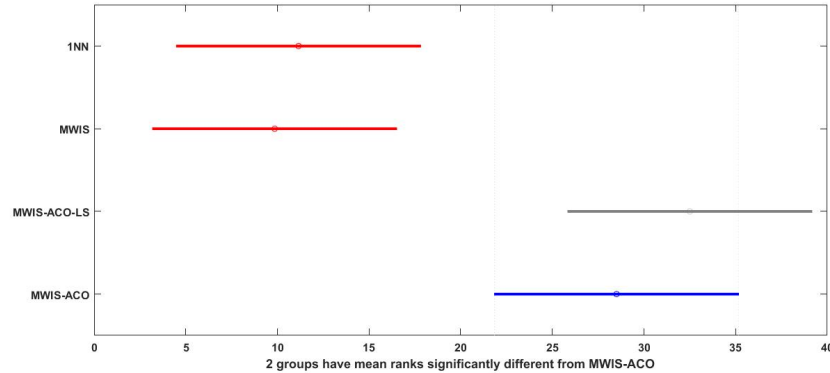


figure. 3.37: Le résultat du test de kruskal-wallis entre leMWIS-ACO, 1NN et MWIS sur les jeux de données (exactitude de la classification).

Le test statistique de kruskal-wallis présenté dans les figures (3.36 et 3.37) montre une comparaison des résultats obtenus par MWIS, MWIS-ACO, MWIS-ACO-LS, et le 1NN classifieur. Selon ces chiffres, les performances des approches MWIS-ACO et MWIS-ACO-LS dépassent celles de la méthode MWIS et du classifieur 1NN. En termes de significativité statistique des résultats (l'exactitude de la classification), ledit test prouve que la différence entre l'exactitude de classification en ("1NN", "MWIS") et "MWIS-ACO-LS" est statistiquement significative (remarquable).

Le tableau 3.9 répertorie le meilleur sous-ensemble de gènes sélectionnés par (MWIS-ACO-LS) pour les ensembles de données dans lesquels nous avons trouvé les meilleures performances par rapport aux autres travaux (9_Tumors, Brain_Tumor1, Brain_Tumor2 et Prostate_Tumeur). Ces gènes sont des biomarqueurs potentiels dans l'identification du cancer.

Sur la base des expériences que nous avons menées, nous pouvons conclure que notre approche de sélection de gènes (MWIS-ACO-LS) est bien fondée. En effet, sur les dix jeux de données utilisés, notre méthode a atteint une grande exactitude de classification. Plus précisément, la méthode proposée a donné une exactitude de classification égale ou supérieure à 99,42 % pour tous les ensembles de données, avec une classification parfaite (100 %) pour (9_Tumors, Brain_Tumor1, Brain_Tumor2, Leukimia1, Leukimia2, SRBCT, Prostate_Tumor, DLBCL) utilisant moins de 40 gènes. L'exactitude élevée de classification trouvée revient à deux éléments, le premier est la combinaison d'une méthode de la théorie des graphes (MWIS) et de la métaheuristique ACO. La seconde est l'utilisation d'un classificateur 1NN modifié où nous normalisons les données d'apprentissage afin de les transformer en une échelle commune.

Table. 3.9: Liste des gènes sélectionnés à l'aide de MWIS-ACO-LS.

Jeu de données	Gènes sélectionnés gene No
9_Tumors	79; 139; 507; 552; 712; 1297; 1364; 1387; 1432; 1444; 1858; 1952; 1967; 1999; 2022; 2058; 2448; 2498; 2989; 3114; 3370; 3473; 3662; 3860; 3876; 3922; 3991; 4018; 4050; 4081; 4525; 4688; 4724; 4783; 4967; 5175; 5265; 5282; 5550; 5698
Brain_Tumor1	26; 318; 868; 895; 925; 1509; 1880; 1950; 2703; 2704; 3789; 3941; 4269; 4690; 5291; 5342; 5404; 5604; 5746
Brain_Tumor2	1026; 3184; 4049; 4304; 4524; 4639; 4649; 5547; 6892; 8134; 8482
Prostate_Tumor	364; 1478; 1936; 2409; 2445; 2968; 3738; 4032; 4413; 4441; 4680; 4823; 4960; 5714; 6773; 7437; 8009; 8531; 8574; 9070; 10108

En pratique, le choix d'un sous-ensemble optimal de gènes fait l'objet d'un intérêt croissant dans le domaine biologique, dans le sens où il permet aux biologistes de gagner du temps puisqu'ils peuvent se référer directement à des gènes sélectionnés qui peuvent servir au diagnostic de cancer. De plus, un petit sous-ensemble de gènes peut réduire le temps de construction des classifieurs.

Dans ce qui suit, nous faisons une comparaison entre notre méthode et certains algorithmes d'optimisation récents en utilisant plusieurs ensembles de données de classification.

3.4.2.4 Comparaison avec d'autres approches

Dans cette sous-section, nous comparons notre méthode avec huit algorithmes récemment référencés dans la littérature [3, 24, 38, 39, 134, 135, 192]. Pour donner un sens à cette comparaison, les expériences sont réalisées dans les mêmes conditions dans chaque algorithme. Concrètement, notre approche est exécutée dix fois sur chaque jeu de données, puis nous choisissons la moyenne et le meilleur sous-ensemble de gènes trouvés. Nous indiquons que les papiers [38, 39] ne rapportent que les meilleurs résultats trouvés.

Le tableau 3.10 résume l'exactitude de la classification et le nombre de gènes sélectionnés (tirés des articles originaux) pour les différentes approches. Le symbole (-) indique que le résultat n'est pas rapporté dans le travail associé. Nous remarquons que les résultats obtenus par notre approche sont très encourageants par rapport aux travaux antérieurs. En effet, pour la plupart des ensembles de données examinés, les exactitudes de classification obtenues par notre méthode de sélection correspondaient ou surpassaient celles obtenues en utilisant d'autres méthodes [3, 24, 38, 39, 134, 135, 192].

3.4. SÉLECTION DE GÈNES VIA UN NOUVEL ALGORITHME HYBRIDE D'OPTIMISATION PAR COLONIES DE FOURMIS POUR LA CLASSIFICATION DU CANCER

Table. 3.10: Comparaison entre notre méthode (MWIS-ACO-LS) et des méthodes de l'état de l'art.

Jeu de données	Méthode	MWIS-ACO-LS [Ce travail]	RBPSO-1NN 2018 [24]	FBPSO-SVM 2018 [24]	FRBPSO 2017 [3]	HICATS (2016) [192]	EPSO (2013) [135]	TS-BPSO (2009) [39]	IBPSO (2008) [38]	IBPSO 2011 [134]
11_Tumors	Performances									
	Best #Acc (%)	99.42	-	-	-	97.70	96.55	97.35	93.10	95.4
	Best #Genes	101	-	-	-	287	243	3206	2948	228
	Average #Acc (%)	99.14 <1>	-	-	-	95.86	95.40	-	-	95.06
9_Tumors	Average #Genes	166.9	-	-	-	307.5	237.70	-	-	240.9
	Best #Acc (%)	100.00	83.33	95.00	-	83.33	76.67	81.63	78.33	78.33
	Best #Genes	40	20	71	-	259	251	2941	1280	248
	Average #Acc (%)	100.00 <1>	81.83	92.222	-	78.33	75.00	-	-	75.5
Brain_Tumor1	Average #Genes	51	29.1	45	-	248.5	247.10	-	-	240.6
	Best #Acc (%)	100.00	94.44	97.78	-	94.44	93.33	95.89	94.44	93.33
	Best #Genes	19	11	21	-	6	8	2913	754	5
	Average #Acc (%)	99.22 <1>	94.00	97.22	90.67	93.10	92.11	-	-	92.56
Brain_Tumor2	Average #Genes	22.9	24.7	22.4	803	8.9	7.5	-	-	11.2
	Best #Acc (%)	100.00	96.00	100.00	-	94.00	94.00	92.65	94.00	94.00
	Best #Genes	11	15	12	-	3	4	5086	1197	4
	Average #Acc (%)	99.40 <2>	92.80	100.00	87.6	92.60	92.4	-	-	91.00
Leukimia1	Average #Genes	11.1	24.5	14.3	662	5.8	6.0	-	-	6.4
	Best #Acc (%)	100.00	100.00	100.00	-	100.00	100.00	100.00	100.00	100.00
	Best #Genes	5	8	6	-	3	2	2577	1034	2
	Average #Acc (%)	100.00 <1>	99.72	100.00	98.89	100.00	100.00	-	-	100.00
Leukemia2	Average #Genes	9.4	11.7	8.4	825	3	3.2	-	-	3.2
	Best #Acc (%)	100.00	100.00	100.00	-	100.00	100.00	100.00	100.00	100.00
	Best #Genes	11	5	6	-	5	4	5609	1292	4
	Average #Acc (%)	100.00 <1>	100.00	100.00	97.50	100.00	100.00	-	-	100.00
Lung_Cancer	Average #Genes	13.9	13.1	8.6	1028	6.80	6.8	-	-	6.7
	Best #Acc (%)	99.51	-	-	-	97.04	96.06	99.52	96.55	96.55
	Best #Genes	36	-	-	-	7	7	6958	1897	10
	Average #Acc (%)	98.92 <1>	-	-	-	96.16	95.67	-	-	95.86
SRBCT	Average #Genes	34.8	-	-	-	7.8	8.3	-	-	14.9
	Best #Acc (%)	100.00	100.00	100.00	-	100.00	100.00	100.00	100.00	100.00
	Best #Genes	6	7	10	-	9	7	1084	431	6
	Average #Acc (%)	100.00 <1>	100.00	100.00	98.19	100.00	99.64	-	-	100.00
Prostate_Tumor	Average #Genes	7.6	11.7	12.4	213	11.7	14.90	-	-	17.5
	Best #Acc (%)	100.00	99.02	100.00	-	98.04	99.02	95.45	92.61	98.04
	Best #Genes	21	9	6	-	5	5	5320	1294	7
	Average #Acc (%)	99.12 <2>	98.24	100.00	92.43	97.75	97.84	-	-	97.94
DLBCL	Average #Genes	20.3	11.2	8.3	418	7.2	6.6	-	-	13.6
	Best #Acc (%)	100.00	100.00	100.00	-	100.00	100.00	100.00	100.00	100.00
	Best #Genes	6	6	4	-	3	3	2671	1042	4
	Average #Acc (%)	100.00 <1>	100.00	100.00	96.49	100.00	100.00	-	-	100.00
	Average #Genes	7.2	12.5	6.7	105	4.10	4.70	-	-	6

< · > : Indique le rang de notre méthode parmi les autres algorithmes au niveau de l'exactitude moyenne pour chaque jeu de données.

RBPSO-1NN= A gene selection method based on the combination of ReliefF, BPSO and 1NN as a classifier.

FBPSO-SVM = A gene selection method based on the combination of Fisher score, BPSO and SVM as a classifier.

FRBPSO= A Fuzzy Rule Based Binary PSO.

HICATS=Hybrid Binary Imperialist Competition Algorithm and Tabu Search.

EPSO= An enhancement of binary particle swarm optimization.

TS-BPSO = A combination of tabu search and BPSO.

IBPSO = An improved binary PSO.

Premièrement, pour l'ensemble de données (9_tumeurs), nous obtenons une classification parfaite avec seulement 40 gènes. Nous constatons que la meilleure performance pour cet ensemble de données est atteinte par notre approche (MWIS-ACO-LS), dépassant le résultat le plus connu de 5% au niveau d'exactitude [24]. Nous notons que le nombre de gènes rapportés dans FBPSO-SVM [24] est de 71 gènes pour avoir une bonne exactitude.

De même, pour les ensembles de données (11_Tumors, Brain_Tumor1, Brain_Tumor2 et Prostate_Tumor), nous obtenons les meilleures performances. De plus, nous avons une classification parfaite (100%) pour Brain_Tumor1, Brain_Tumor2 et Prostate_Tumor avec moins de 21

gènes.

Le tableau 3.11 rapporte le rang de la méthode proposée par rapport aux autres méthodes existantes selon l'exactitude moyenne. Les résultats dans ledit tableau montrent que la méthode proposée a atteint la meilleure exactitude dans la plupart des ensembles de données. En effet, nous voyons clairement que notre méthode est plus adaptée à la sélection de gènes. Comme le montrent les tableaux 3.10 et 3.11, nous égalons ou dépassons les performances de toutes les méthodes de comparaison ; sauf pour les jeux de données « Brain_Tumor2 » et « Prostate_Tumor », dans lesquels notre approche arrive au deuxième rang après l'approche FBPSO-SVM.

Les résultats de cette analyse comparative avec des méthodes antérieures de sélection de gènes dans le contexte de la classification des cancers nous ont permis de conclure que notre méthode d'optimisation inspirée de la nature est utile dans le problème de sélection de gènes.

Table. 3.11: Classement de notre méthode comparativement aux méthodes existantes selon l'exactitude moyenne.

Dataset	Method						
	MWIS-ACO-LS	RBPSO-1NN	FBPSO-SVM	FRBPSO	HICATS	EPSO	IBPSO
	Our proposed method						
11_Tumors	1	-	-	-	2	3	4
9_Tumors	1	3	2	-	4	6	5
Brain_Tumor1	1	3	2	7	4	6	5
Brain_Tumor2	2	3	1	7	4	5	6
Leukemia1	1	3	1	2	1	1	1
Leukemia2	1	1	1	2	1	1	1
Lung_Cancer	1	-	-	-	2	4	3
SRBCT	1	1	1	3	1	2	1
Prostate_Tumor	2	3	1	7	6	5	4
DLBCL	1	1	1	2	1	1	1

Dans cette section, nous avons présenté une approche hybride (MWIS-ACO-LS) pour la sélection de gènes dans les données de puces à ADN. La méthode proposée commence par une phase de présélection où une nouvelle approche de la théorie des graphes est présentée pour sélectionner un premier petit sous-ensemble de gènes, dans cette phase nous modélisons le problème de sélection de gènes comme un problème de MWIS (maximum weight independent set), et nous présentons un algorithme glouton pour approximer le MWIS des gènes, et la méthode termine par une phase de recherche qui détermine un sous-ensemble presque optimal de gènes pour la classification du cancer. Cette dernière est basé sur un ACO modifié et un algorithme de recherche locale (LS). Cette approche vise à sélectionner un petit sous-ensemble de gènes pertinents à partir d'un ensemble de données original qui contient des données redondantes, bruyantes ou non pertinentes. Les résultats expérimentaux montrent que notre approche se compare très favorablement aux méthodes de référence en termes d'exactitude de classification et de nombre de gènes sélectionnés.

3.5 Recherche de voisinage variable (VNS) pour la sélection de gènes

La recherche de voisinage variable (Variable Neighborhood Search en abrégé VNS) [84, 133] est une métaheuristique récente. Grâce à son efficacité, il a été utilisé pour résoudre plusieurs problèmes d'optimisation dans différents domaines, en particulier dans le problème de sélection d'attributs pour la classification [69, 70, 142, 149]. Dans VNS, nous partons d'une solution initiale et nous essayons de l'améliorer en explorant l'espace de recherche en se basant sur des structures de voisinage multiples (procédure de perturbation (shaking)) en utilisant une méthode de recherche locale bien choisie (procédure d'amélioration "improvement procedure") pour échapper aux optima locaux.

Dans cette étude, nous proposons deux approches hybrides de type filtre pour résoudre le problème de sélection de gènes. Nos approches proposées commencent par une première étape dans laquelle un filtre univarié est utilisé pour sélectionner les p gènes les mieux classés en fonction de leur pertinence. Dans la deuxième étape, une méthode hybride basée sur la métaheuristique VNS et un algorithme de recherche locale stochastique est appliquée sur l'ensemble de gènes sélectionnés par le filtre univarié, la première méthode proposée (*VNSMI*) est basée sur certains concepts de la théorie de l'information. Alors que la seconde (*VNSCor*) est basée sur le coefficient de corrélation de Pearson, les approches proposées n'ont pas été étudiées auparavant.

3.5.1 Approche proposée

Dans ce travail, nous utilisons D pour désigner un ensemble de données (voir table 1.1) de M échantillons $X = (x_1; x_2; \dots; x_M)$, $x_i \in \mathbb{R}^N$. Nous utiliserons $\{g_1; g_2; \dots; g_N\}$, $g_i \in \mathbb{R}^M$ pour désigner les N gènes. $Y = (y_1; y_2; \dots; y_M)$ sont les étiquettes des classes.

3.5.1.1 Information mutuelle MI

Notre première méthode *VNSMI* est basée sur la théorie de l'information, dans laquelle l'information mutuelle (Mutual information MI) est une mesure utile largement utilisée dans les problèmes de sélection de gènes pour la classification du cancer [31, 51, 109]. Dans le contexte de la théorie de l'information, les colonnes de notre matrice D (gènes et étiquette de classe) peuvent être considérées comme des variables aléatoires. En effet, l'information mutuelle entre deux variables aléatoires X, Y est la dépendance statistique entre ces variables. Elle est souvent mesurée en bits, et elle est définie comme :

$$\begin{aligned} I(X; Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X, Y) \\ &= H(X, Y) - H(X|Y) - H(Y|X). \end{aligned}$$

Où $H(X)$ et $H(Y)$ sont les mesures d'incertitude (entropie) des variables aléatoires X et Y respectivement. $H(X|Y)$ et $H(Y|X)$ sont les entropies conditionnelles de X sachant Y , et Y sachant X , respectivement, et $H(X, Y)$ est l'entropie conjointe en X et Y .

La relation entre l'entropie et l'information mutuelle peut être interprétée dans la figure.3.38 :

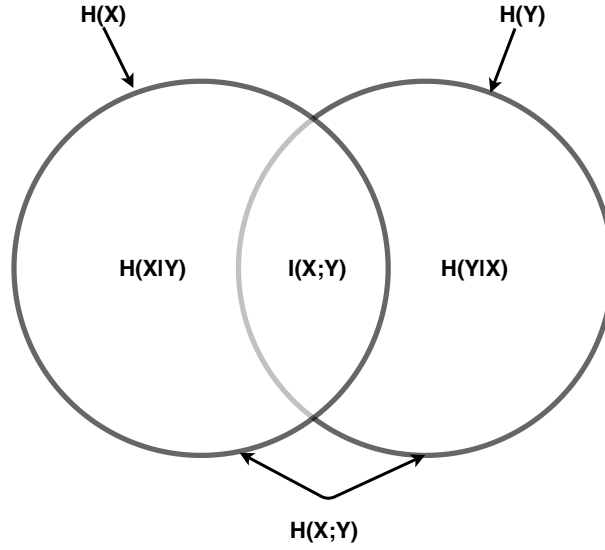


figure. 3.38: La relation entre l'entropie et l'information mutuelle

Selon la définition de Shannon de l'entropie, nous avons dans le cas discret :

$$H(X) = - \sum_{x \in X} p(x) \log(p(x))$$

$$H(X, Y) = - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log(p(x, y))$$

Où $p(x, y)$ représente la distribution probabiliste conjointe sur les variables aléatoires X et Y , et $(p(x), p(y))$ sont les probabilités marginales respectives.

Dans le cas de variables aléatoires continues, la sommation est remplacée par une intégrale double définie :

$$I(X; Y) = \int_Y \int_X p(x, y) \log \left(\frac{p(x, y)}{p(x)p(y)} \right) dx dy$$

L'information mutuelle a les propriétés suivantes :

- $I(X; X) = H(X)$
- $I(X; Y) = 0$ si et seulement si X et Y sont des variables aléatoires indépendantes
- L'information mutuelle est positive ou nulle $I(X; Y) \geq 0$
- L'information mutuelle est symétrique $I(X; Y) = I(Y; X)$

Nous définissons aussi l'information mutuelle normalisée notée (NI) entre g_i et g_s . $NI(g_i; g_s)$ est l'information mutuelle normalisée par l'entropie minimale des deux gènes [64].

$$NI(g_i; g_s) = \frac{I(g_i; g_s)}{\min \{H(g_i), H(g_s)\}}.$$

Ainsi, nous définissons la pertinence d'un gène g_i aux classes C en attribuant un score de pertinence $SNMI_i$ à g_i défini par

$$SNMI_i = \frac{NI(g_i; C)}{\max_{j \in 1, \dots, N} NI(g_j; C)}.$$

Ce score de pertinence est inspiré de l'algorithme "Information Gain (IG)" [109]. Le calcul de l'information mutuelle est basé sur l'estimation des fonctions de densité de probabilité et de probabilités conjointes des variables. Dans notre étude, nous utilisons le package d'information mutuelle de Peng codé sous Matlab [148], afin d'estimer la valeur d'information mutuelle.

3.5.1.2 Coefficient de corrélation de Pearson

Notre deuxième méthode *VNSCor* est basée sur le coefficient de corrélation (déjà définie dans le sous-paragraphe 3.4.1.1.2, la page 85) qui est une mesure de la force de la relation linéaire entre deux variables (gènes).

Sur la base de cette notion, nous pouvons définir une mesure de pertinence SR_i de chaque gène g_i aux classes (C) par

$$SR_i = \frac{|\rho_{g_i, C}|}{\max_{j \in 1, \dots, N} |\rho_{g_j, C}|}.$$

Où $\rho_{g_j, C}$ est la corrélation entre le gène g_j et le vecteur des classes cibles C (dernière colonne de la table 1.1).

3.5.1.3 VNS

La recherche à voisinage variable (Variable Neighborhood Search en abrégé VNS) est une métaheuristique récente, introduit par Mladenović et Hansen en 1997 [84, 133] pour résoudre des problèmes d'optimisation combinatoire. Cette méthode peut être décomposée en deux phases : une phase dans laquelle une recherche locale converge vers un optimum local, et une autre phase permet de s'en échapper.

Cette métaheuristique utilise les concepts suivants :

- Changement systématique de voisinage pour extraire les optimums locaux.
- Passer d'une solution à une autre ne serait-ce que s'il y a une amélioration.
- Utilisation de la recherche locale pour obtenir un optimum local.

Le principal inconvénient des algorithmes de recherche de voisinage est qu'en général, ils fournissent des solutions localement optimales. Par conséquent, ils intègrent généralement des mécanismes qui leur permettent d'échapper aux optima locaux. en changeant systématiquement la structure du voisinage au cours du processus de recherche.

Parmi les travaux récents dans lesquels la métaheuristique VNS a été proposée pour résoudre des problèmes de sélection d'attributs nous trouvons [69, 70, 142, 149].

La recherche à voisinage variable (VNS) commence à partir d'une solution initiale et d'un nombre fini de voisinages prédéfinis $\mathcal{N}_k, k = 1, \dots, k_{\max}$, à chaque itération, une nouvelle solution candidate (Shaking step) est générée à partir du voisinage courant, puis un algorithme de recherche local est appliqué à la solution courante afin de l'affiner et de l'améliorer. Si cette nouvelle solution

s' est meilleure que la première, le processus reprend avec le premier voisinage, dans le cas contraire les mêmes étapes sont répétées mais en passant au voisinage suivant.

3.5.1.3.1 VNS Basique [84] (figure.3.39)

Initialisation. Sélectionnez l'ensemble des structures de voisinage $\mathcal{N}_k, k = 1, \dots, k_{max}$, qui seront utilisés dans la recherche; trouver une solution initiale s ; choisir une condition d'arrêt; Répétez ce qui suit jusqu'à ce que la condition d'arrêt soit remplie :

1. $k \leftarrow k + 1$;
2. Jusqu'à $k = k_{max}$, répétez les étapes suivantes :
 - (a) **Shaking (perturbation).** Générer une solution candidate s' au hasard à partir du $k^{\text{ème}}$ voisinage de s ($s' \in \mathcal{N}_k(s)$);
 - (b) **Recherche locale.** Appliquez une méthode de recherche locale sur s' ; notons par s'' l'optimum local ainsi obtenu;
 - (c) **Se déplacer ou non.** Si cet optimum local est meilleur que s , se déplacer là ($s \leftarrow s''$) et continuer la recherche avec $\mathcal{N}_1(k \leftarrow 1)$; sinon, définissez $k \leftarrow k + 1$;

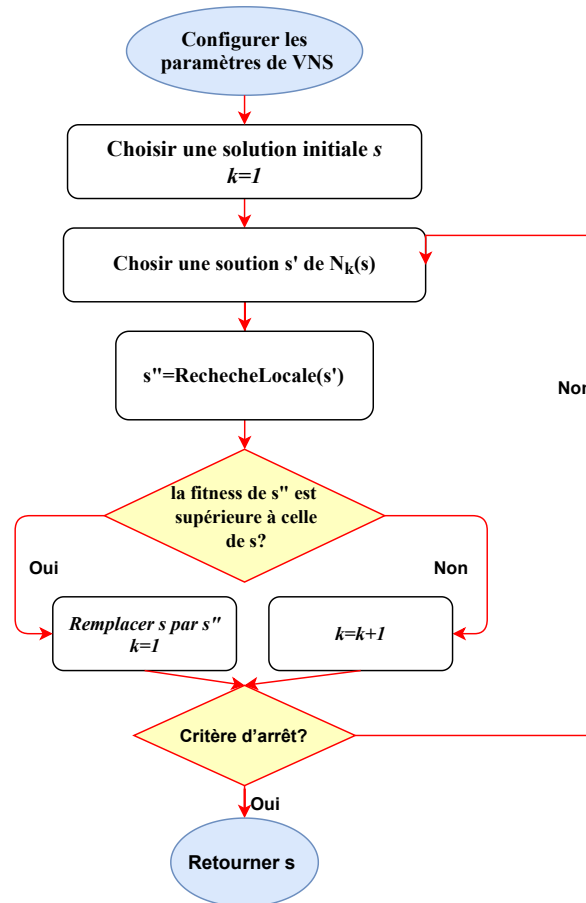


figure. 3.39: Organigramme de VNS basique.

3.5.2 VNS pour la sélection de gènes

Dans cette sous-section, nous présentons une nouvelle méthode de sélection de gènes pour la classification du cancer, dont le but est de choisir un sous-ensemble de gènes pertinents pouvant améliorer les performances d'un modèle d'apprentissage défini. Étant donné que ce problème est NP-difficile [17], les métaheuristiques basées sur le voisinage semblent bien adaptées pour traiter ce problème, qui est modélisé comme un problème d'optimisation combinatoire.

3.5.2.1 Formalisation du problème

L'objectif principal de la sélection de gènes est de trouver un sous-ensemble de gènes pertinents à partir de données de puces à ADN de grande dimension, cela nous permet d'obtenir une classification élevée du cancer, et avoir une meilleure interprétation de la relation biologique entre les gènes et le résultat clinique considéré, puis une meilleure compréhension scientifique d'un problème donné.

L'espace de recherche : défini par les sous-ensembles de gènes sélectionnés possibles ;

La fonction objectif :

Dans ce travail, nous définissons $\mathcal{J}$ comme la valeur de la fonction objectif utilisée pour mesurer la qualité d'un sous-ensemble donné de gènes S . La fonction fitness utilisée $\mathcal{J}$ cherche à minimiser la redondance et à maximiser la pertinence en S en utilisant un nombre minimal de gènes. En effet, nous nous sommes inspirés de la fonction objectif définie dans [51] *MID* :

Une redondance minimale garantit que la sélection d'un ensemble contient des gènes non corrélés pour une meilleure représentation de l'ensemble de données initial. Soit S un sous-ensemble de gènes. La condition de redondance minimale est :

$$\min W_I(S), \quad W_I(S) = \frac{1}{|S|^2} \sum_{g_i, g_j \in S} NI(g_i, g_j) \quad (3.17)$$

D'autre part, la condition de pertinence maximale assure la sélection d'un sous-ensemble de gènes à haute pertinence pour les classes cibles, pour un sous-ensemble S donné nous cherchons à :

$$\max V_I(S), \quad V_I = \frac{1}{|S|} \sum_{g_i \in S} SNMI_i \quad (3.18)$$

Le but de maximiser la pertinence et de minimiser la redondance est d'optimiser simultanément les conditions (3.17) et (3.18), il s'agit alors d'un problème d'optimisation multi-objectif de deux conditions, cela nécessite alors de les combiner en une seule fonction objectif. Nous considérons que le traitement des deux conditions est également important et choisissons la combinaison la plus simple :

$$\max (V_I(S) - W_I(S))$$

La manière de combiner pertinence et redondance conduit au critère de sélection d'un nouveau sous-ensemble de gènes, nommée *MID* : critère de différence d'information mutuelle [51]. Ainsi,

notre fonction objectif est définie de la manière suivante : pour un sous-ensemble donné de gènes S nous avons

$$\mathcal{J}(S) = V_I(S) - W_I(S).$$

Nous pouvons remarquer que ledit problème d'optimisation prend en considération l'interaction possible entre les gènes.

Les données de biopuces de grande dimension rendent inefficace l'application de méthodes évolutives directement pour résoudre le problème de sélection de gènes [24]. Afin de surmonter cet obstacle, nos méthodes proposées commencent par une étape de prétraitement, où nous utilisons une méthode de filtrage. Plus précisément, à ce stade, les p gènes qui ont un score élevé sont sélectionnés. Le sous-ensemble de gènes sélectionné dans la première étape est pris en entrée dans la deuxième étape de sélection, qui utilise un VNS couplé à un algorithme de recherche locale pour sélectionner un petit sous-ensemble de gènes contenant le maximum de pertinence et le minimum de redondance :

$$\max (V_I - W_I)$$

3.5.2.1.1 Représentation de l'espace de recherche

L'espace de recherche doit être représenté sous une forme appropriée pour VNS. Compte tenu d'une nouvelle donnée de puce à ADN générée par l'étape de filtre contenant p gènes seulement. L'ensemble des gènes (c'est-à-dire la solution s) est représenté par une chaîne binaire de longueur p (c'est-à-dire $s = [s_1, s_2, \dots, s_p]$), si un bit s_i dans la solution vaut "1" cela signifie que le gène g_i est sélectionné dans le sous-ensemble S , d'autre part, une valeur de "0" indique que le gène n'est pas sélectionné dans le sous-ensemble S . p est le nombre original de gènes.

Dans la figure.3.40, la solution candidate s signifie que parmi les 6 gènes, g_1, g_3, g_4 et g_6 sont sélectionnés $S = \{g_1; g_3; g_4; g_6\}$.

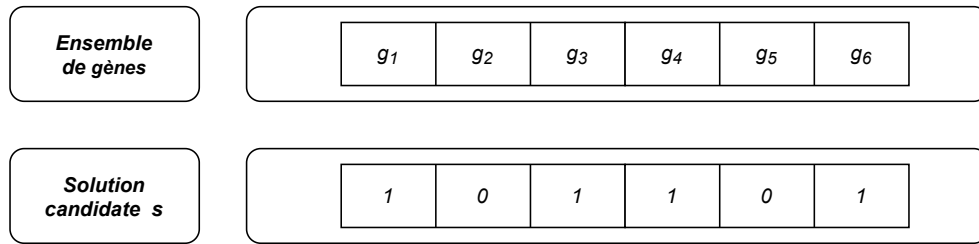


figure. 3.40: Un exemple illustré avec un sous-ensemble généré et une représentation de solution.

3.5.2.1.2 Structure de voisinage (Shaking step)

Le voisinage d'une solution s correspond à l'ensemble des solutions accessibles depuis s à l'aide d'un mouvement élémentaire, ainsi, la Structure de Voisinage est une notion essentielle dans l'algorithme VNS qui nécessite de définir une distance entre deux solutions (ensembles de gènes) candidates, dans notre travail nous utilisons la distance de Hamming.

$$d(s, s') = \sum_{i=1}^p |s_i - s'_i|$$

Où $d(s, s')$ est le nombre de positions où s et s' diffèrent. Donc le voisinage k de s noté $N_k(s)$ est défini comme suit :

$$N_k(s) = \{s' : d(s, s') \leq k\}$$

3.5.2.1.3 Voisinage d'une solution

Soit s une solution courante, on fixe le nombre maximum de voisinages k_{max} $\mathcal{N}_k, k = 1, \dots, k_{max}$ si l'on veut effectuer une perturbation (shaking step) de s pour obtenir une nouvelle solution s' d'ordre m (à partir d'un voisinage donné $m, N_m(s)$) tel que $m \in \{1, \dots, k_{max}\}$, nous répétons le processus suivant m fois : on choisit au hasard un nombre $j \in \{1, 2, \dots, p\}$, puis on échange le bit associé dans s (ie $s_j \rightarrow |s_j - 1|$). L'algorithme suivant montre comment fonctionne la perturbation d'ordre m :

Algorithme 10 Méthode de perturbation

fonction : MéthodeDePerturbation(s,m)

Entrées : $s = [s_1, s_2, \dots, s_p]$: solution candidate ; m :ordre de voisinage **Sortie** : s^m solution perturbée du voisinage $\mathcal{N}_m(s)$.

Début

$s^m = s$;

$k = 0$;

Tant que $k < m$

$j = \lfloor rand * (p + 1) \rfloor \% \lfloor . \rfloor$ la partie entière.

$s_j^m = |s_j^m - 1|$.

$k = k + 1$

Fin Tant que

Retourner s^m .

3.5.2.1.4 Initialisation :

La solution initiale s_{init} est générée de manière complètement aléatoire à partir des p gènes sélectionnés dans l'étape de filtrage. s_{init} est considéré comme une solution initiale à notre VNS de base.

3.5.2.1.5 Recherche locale stochastique (Stochastic Local Search SLS) :

La recherche locale est utilisée pour améliorer la solution construite dans l'étape de perturbation et fournir de bonnes solutions dans un délai raisonnable. Dans ce but, nous utilisons une recherche locale stochastique SLS inspirée de [33], dans cette méthode (SLS) on démarre avec la solution

générée dans la phase de perturbation, ensuite, on effectue un certain nombre d'opérations qui combinent des stratégies de diversification et d'intensification pour aboutir à de bonnes solutions, en effet on définit les deux opérations :

- **La phase de diversification** : sélection aléatoire d'une solution parmi $\mathcal{N}_1$.
- **La phase d'intensification** : sélection de la meilleure solution voisine dans $\mathcal{N}_1$ selon la mesure $\mathcal{J}$.

La phase d'intensification se fait avec une probabilité fixée $ip > 0$ et la phase de diversification avec une probabilité $1 - ip$. Le processus se répète tant que la solution initiale est améliorée. Notre recherche locale stochastique se présente comme suit :

Algorithme 11 Recherche locale stochastique (SLS) pour la sélection de gènes

fonction : SLS(s,ip)

Entrées : s = une solution candidate; ip = la probabilité de faire la phase d'intensification

Sortie : Une solution candidate s_{best} meilleure que s .

Début

k=0;

Tant que $k < 1$

if ($rand < ip$) **then**

$s' =$ choisir la meilleure solution parmi le voisin $\mathcal{N}_1(s)$ (La phase d'intensification).

else

$s' =$ choisir une solution aléatoire du voisin $\mathcal{N}_1(s)$ (La phase de diversification).

end if

if $\mathcal{J}(s') \geq \mathcal{J}(s)$ **then**

$s = s'$

$k = -1$ % % continue la recherche locale.

end if

 k=k+1

Fin Tant que

$s_{best} = s$

Retourner s_{best} .

3.5.2.2 Méthodes proposées pour la sélection de gènes VNS

Nous proposons deux approches filtres pour la sélection de gènes, la première (*VNSMI*) est basée sur l'information mutuelle normalisée NI et la seconde (*VNSCor*) est basée sur le coefficient de corrélation de Pearson. Nous présentons d'abord l'approche *VNSMI*, concernant le *VNSCor* est dérivé de la première juste nous remplaçons l'information mutuelle normalisée par le coefficient de corrélation absolu et le score *SNMI* par le coefficient *SR*. Ces méthodes proposées enrichissent l'état de l'art de la sélection de gènes pour la classification du cancer.

3.5.2.2.1 Structure générale de l'approche proposée (VNSMI)

Notre première méthode proposée *VNSMI* (l'algorithme 12 et la figure.3.41) pour la sélection de gènes est basée sur la combinaison entre deux méthodes filtres (c'est-à-dire le schéma univarié et multivarié). Le schéma univarié basé sur le score *SNMI* en sélectionnant le p gènes les mieux classé, tandis que le schéma multivarié se fait via un VNS couplé à un algorithme de recherche locale stochastique appelé SLS. La qualité du sous-ensemble final de gènes est mesurée par un classificateur spécifique avec une validation croisée "Leave-one-out-cross-validation (LOOCV)". Le déroulement général de cette approche se caractérise par un processus séquentiel en deux phases :

Phase 1 : Cette phase de prétraitement consiste à réduire la taille des données initiales des puces à ADN, en filtrant les gènes non informatifs à l'aide du score *SNMI* ; le nombre de gènes présélectionnés est fixé à p , il s'agit d'un schéma univarié.

Étape 2 : Cette étape consiste à appliquer la recherche de voisinage variable où *VNSMI* génère un sous-ensemble de gènes à partir des gènes retenus par l'étape précédente selon les concepts VNS, de plus une recherche locale stochastique (SLS) est appliquée pour aider VNS à trouver des bons résultats. La fonction objectif utilisée est $\mathcal{J}$.

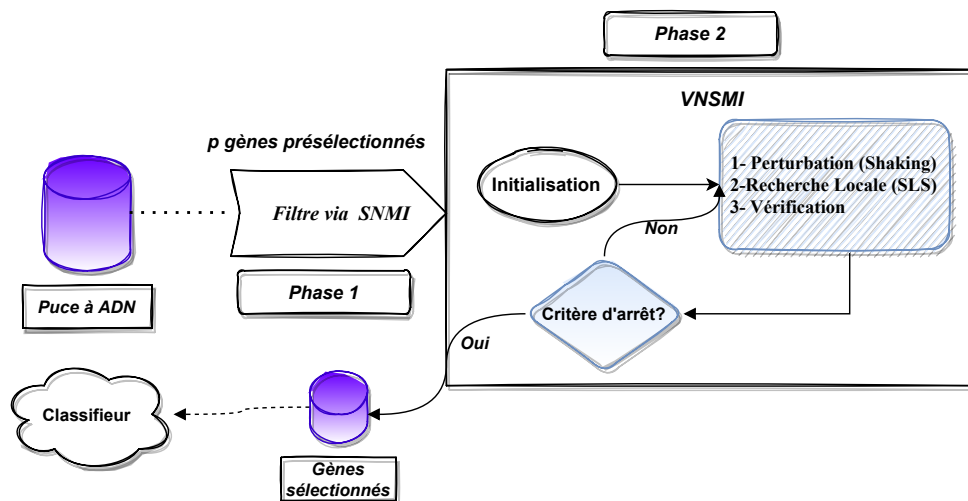


figure. 3.41: Organigramme de l'approche proposée (*VNSMI*) pour la sélection de gènes dans les données de puces à ADN.

L'algorithme 12 montre un aperçu de l'approche proposée *VNSMI*.

Algorithme 12 Approche de recherche de voisinage variable pour la sélection de gènes.

Fonction : $VNSMI(D, k_{max}, it_{max}, p, ip)$

Entrées : X = Données initiales de biopuce, k_{max} : Le nombre total de structures de voisinage, it_{max} : Le nombre maximal d'itérations. p : le nombre de gènes à sélectionner dans la première étape, ip = la probabilité de faire la phase d'intensification.

Sortie : s_{best} : Le meilleur sous-ensemble de gènes.

Phase 1 : La sélection du premier sous-ensemble de gène

Étape 1 : Calculer $SNMI_i$ pour chaque gène g_i .

Étape 2 : Sélectionner les p gènes les mieux classés pour les utiliser dans la **Phase 2** .

Phase 2 : L'application de VNS au sous-ensemble de gène sélectionné dans la première phase.

Calculer l'information mutuelle normalisée entre tous les gènes.

Générer une solution initiale s .

$l=0$;

Tant que $l < it_{max}$

$k = 1$;

Tant que $k \leq k_{max}$

$s' = MéthodeDePerturbation(s, k)$;

$s'' = SLS(s', ip)$;

if $\mathcal{J}(s'') \geq \mathcal{J}(s)$ **then**

$s = s''$;

$k = k_{max} + 1$; % Return to the first neighborhood .

else

$k = k + 1$;

end if

Fin Tant que

$l = l + 1$;

Fin Tant que

$s_{best} = s$

Return s_{best} .

3.5.2.3 (VNSCor)

La seconde approche ($VNSCor$) est dérivée de la première ($VNSMI$), les deux seules différences sont : le score SMI est remplacé par le score SR , et l'information mutuelle normalisée entre les gènes est remplacée par la corrélation absolue.

3.5.3 Expérimentations et résultats

La mise en œuvre des approches proposées se fait sous Matlab R2018a. Afin d'évaluer les méthodes proposées, nous utilisons le classifieur linéaire SVM et le plus proches voisins $1NN$

pour calculer la l'exactitude de la classification (LOOCV) en utilisant le sous-ensemble de gènes de sortie. La suite de cette sous-section est consacrée à la présentation des jeux de données utilisés et à l'analyse des résultats expérimentaux.

3.5.3.1 Environnement et jeux de données

3.5.3.1.1 jeux de données

Pour évaluer nos approches, nous avons choisi des jeux de données (DNA microarray), qui concernent tous des problèmes de reconnaissance des cancers, facilement accessible au public. De plus, ces jeux de données ont été utilisés dans plusieurs travaux de classification supervisés (table 3.12). Ces derniers ensembles de données ont une multitude de caractéristiques distinctives : Le nombre d'échantillons dans certains jeux de données est faible (Colon et Leukimia), tandis que d'autres ont un nombre plus élevé (Ovarian). De plus, tous les jeux de données ont des classes binaires. Et comme notre méthode proposée est conçue pour les puces à ADN de grande dimension. Tous les ensembles de données sont décrits par un nombre élevé de gènes (variant entre 2000 et 15154 gènes).

Table. 3.12: Description des jeux de données (ADN microarray) utilisés.

Jeu de données	# gènes	# échantillons	# classes	source
Colon	2000	62	2	[16]
DLBCL	5469	77	2	[179]
Leukemia	7129	72	2	[78]
Ovarian	15154	253	2	[152]
Prostate Tumor	10509	102	2	[179]
Prostate	12600	102	2	[176]

3.5.3.2 Préparation des jeux de données

Notez que les gènes dans les données d'expressions géniques sont à valeurs continues et avaient des échelles différentes. Par conséquent, nous les avons classifié dans un premier temps directement en utilisant SVM, 1NN, puis nous avons mis à l'échelle les données afin que chaque gène ait une valeur moyenne nulle et une variance unitaire, les données mis en échelle sont utilisées pour la méthode *VNSCor*, nous avons utilisé les données mises à l'échelle. Au contraire, dans *VNSMI* nous devons approximer les valeurs d'information mutuelle entre les gènes, ainsi, les données sont discrétisées selon la méthode suivante : pour chaque gène dans les données, nous calculons les σ (écart-type) et μ (moyenne) respectifs, puis les données du vecteur gène qui ont une valeur supérieure à $\mu + \sigma/2$ ont été transformés en 1 ; les valeurs comprises entre $\mu - \sigma/2$ et $\mu + \sigma/2$ ont été transformées en 0 ; et toutes les valeurs inférieures à $\mu - \sigma/2$ ont été transformées en -1.

3.5.3.3 Paramètres

Nous notons que nos approches ont été exécutées sur un ordinateur portable Acer Aspire 7750g avec processeur Intel Core I5 2,30 GHz et 8 Go de RAM, sous système exécutant Windows 7 (64 bits).

Plusieurs tests ont été effectués afin d'obtenir un paramétrage approprié, le tableau 3.13 représente les paramètres des approches proposées.

Table. 3.13: Paramètres utilisés dans VNS.

paramètre		valeur
p	Le nombre de gènes sélectionnés dans la phase 1 (Filtre)	100
k_{max}	Le nombre total de structures de voisinages (VNS)	15
it_{max}	Le nombre maximum d'itérations (VNS)	200
ip	La probabilité de l'intensification dans SLS	0,5

3.5.3.4 Résultats and comparaisons

Les objectifs des expérimentations réalisées sur les six jeux de données de (Puces à ADN) sont les suivants : tester l'effet de la sélection de gènes sur l'amélioration de l'exactitude de la classification, valider les méthodes proposées et vérifier leur efficacité.

Étant donné la nature non déterministe de nos approches, 50 exécutions indépendantes ont été effectuées pour chaque jeu de données afin d'obtenir des résultats plus fiables.

Le tableau 3.14 montre les résultats obtenus en utilisant *VNSMI* pour la sélection de gènes, puis, en utilisant la méthode *VNSCor* ; l'exactitude de la classification est calculée à l'aide des classificateurs 1NN et SVM linéaire, d'autre part, ces méthodes sont comparées aux classificateurs SVM, 1NN sans sélection pour démontrer l'utilité de nos approches de sélection.

Nous analysons nos résultats de trois manières :

- L'exactitude de la classification.
- Le nombre de gènes utilisés dans la classification.
- Le temps d'exécution.

3.5. RECHERCHE DE VOISINAGE VARIABLE (VNS) POUR LA SÉLECTION DE GÈNES

Table. 3.14: Comparaison entre SVM, 1NN, VNSMI et VNSCor (LOOCV).

<i>datasets</i>	<i>performances</i>	<i>1NN</i>	<i>SVM</i>	<i>VNSMI(1NN)</i>		<i>VNSMI(SVM)</i>		<i>VNSCor(1NN)</i>		<i>VNSCor(SVM)</i>	
				<i>best</i>	<i>avg</i>	<i>best</i>	<i>avg</i>	<i>best</i>	<i>avg</i>	<i>best</i>	<i>avg</i>
<i>Prostate Tumor</i>	<i>Accuracy</i>	82,35	92,16	95,1	95,1	97,06	97,06	95,1	94,04	98,04	96,53
	<i>Genes</i>	10509	10509	9	9	9	9	13	14,38	13	14,38
	<i>Time of execution(s)</i>	-	-		4,89				5,01		
<i>DLBCL</i>	<i>Accuracy</i>	84,42	97,4	96,1	96,1	98,7	98,7	100	100	100	99,97
	<i>Genes</i>	5469	5469	10	10	10	10	13	16,64	13	16,64
	<i>Time of execution</i>	-	-		4,27				4,4		
<i>Leukemia</i>	<i>Accuracy</i>	100	100	100	100	100	100	100	100	100	100
	<i>Genes</i>	7129	7129	11	11	11	11	12	15,44	12	15,44
	<i>Time of execution(s)</i>	-	-		4,41				4,48		
<i>Ovarian</i>	<i>Accuracy</i>	93,28	100	100	100	100	100	100	99,76	100	100
	<i>Genes</i>	15154	15154	20	20,94	20	20,94	8	12,54	8	12,54
	<i>time of execution(s)</i>	-	-		5,49				5,63		
<i>Prostate</i>	<i>Accuracy</i>	83,33	92,16	95,1	95,1	97,06	97,06	95,1	94,04	98,04	96,63
	<i>Genes</i>	12600	12600	9	9	9	9	13	14,56	13	14,56
	<i>Time of execution(s)</i>	-	-		4,82				4,94		
<i>Colon</i>	<i>Accuracy</i>	79,03	79,03	82,26	79,1	79,03	77,45	88,71	85,03	88,71	87,52
	<i>Genes</i>	2000	2000	13	10,06	13	10,06	7	8,06	7	8,06
	<i>Time of execution(s)</i>	-	-		4,41				4,52		

Note : Les meilleurs résultats sont présentés en gras.

Remarque : Les SVM, 1NN étant de nature déterministe, la classification est calculée une fois.

Accuracy : L'exactitude de classification (LOOCV).

Genes : Le nombre de gènes utilisés dans la classification.

Best : Le meilleur résultat trouvé dans les 50 exécutions.

Avg : La moyenne des 50 exécutions.

SVM : Le classificateur de machines à vecteurs de support utilisant un noyau linéaire.

1NN : Le classificateur de plus proche voisins.

VNSCor : La méthode proposée basée sur le coefficient de corrélation.

VNSMI : La méthode proposée basée sur l'information mutuelle.

Time of execution : La moyenne du temps d'exécution en secondes.

L'efficacité des approches proposées est tirée de l'amélioration remarquable de l'exactitude de la classification (Tableau 3.14, figures 3.42 et 3.42). En effet, ces méthodes ("VNSMI" et "VNSCor") réduisent le nombre de gènes et améliorent l'exactitude dans tous les jeux de données. De plus, nos méthodes atteignent une exactitude parfaite pour 3 jeux de données (DLBCL, Leukimia et Ovarian) en utilisant moins de gènes 13, nous parlons donc d'une sélection dans le bon sens.

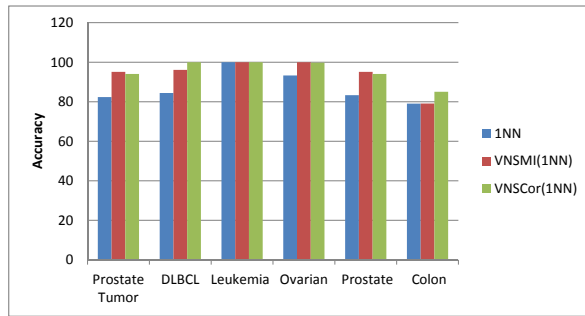


figure. 3.42: Comparaison de l'exactitude moyenne des approches proposées en utilisant (1NN).

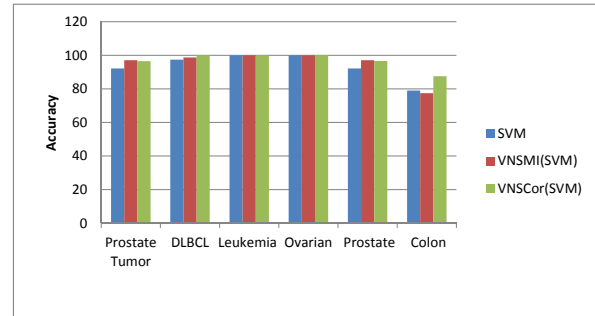


figure. 3.43: Comparaison de l'exactitude moyenne des approches proposées en utilisant (SVM).

Les résultats montrent tout d'abord que "VNSMI" et "VNSCor" ont des performances similaires pour la majorité des jeux de données, avec un avantage de VNSCor" au niveau de l'exactitude. De plus, la combinaison entre nos méthodes proposées et le classificateur SVM surpasse le classificateur 1NN.

Pour tous les ensembles de données, les approches « VNSMI » et « VNSCor » ont atteint une exactitude élevée en utilisant les classificateurs SVM et 1NN, plus précisément ; une classification supérieure à 95,1% utilisant seulement moins de 20 gènes pour cinq jeux de données.

Sur la base du Tableau 3.14, et figures (3.42 et 3.42), nous pouvons remarquer que nos méthodes de sélection de gènes ne sont pas spécifiques à un seul classifieur contrairement aux approches enveloppes. Un autre point qui rend nos méthodes efficaces est le temps d'exécution, nous pouvons trouver un sous ensemble de gènes presque optimal en quelques secondes.

Les figures suivantes (3.44, 3.45, 3.46 et 3.47) montrent l'efficacité des approches proposées, d'une part pour réduire le nombre de gènes et d'autre part pour maximiser la fonction objectif, ceci en se basant sur le jeu de données "Colon", nous pouvons remarquer que les résultats obtenus dans les 50 exécutions sont assez proches (la différence entre la meilleure solution et la solution moyenne n'est pas significative).

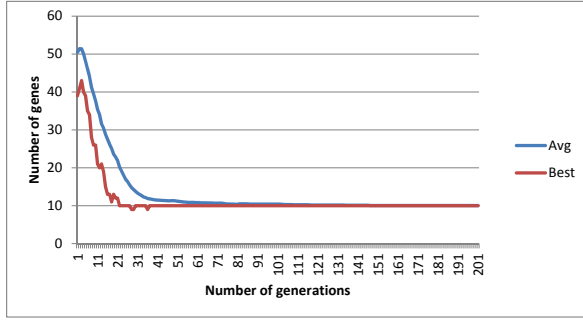


figure. 3.44: Comparaison de l'évolution du nombre de gènes utilisés pour "Colon" à l'aide de VNSMI.

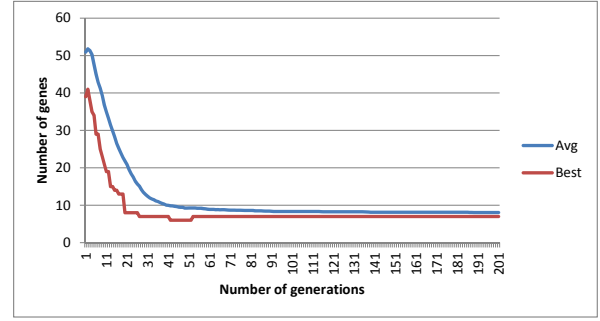


figure. 3.45: Comparaison de l'évolution du nombre de gènes utilisés pour "Colon" à l'aide de VNSCor.

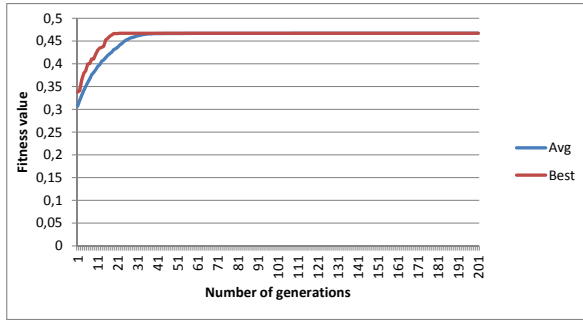


figure. 3.46: Comparaison de l'évolution de $\mathcal{J}$ pour "Colon" en utilisant VNSMI.

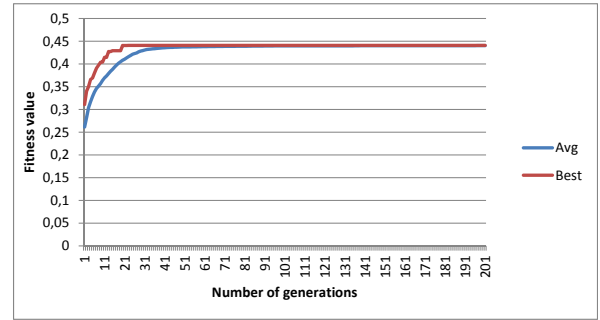


figure. 3.47: Comparaison de l'évolution de $\mathcal{J}$ pour "Colon" en utilisant VNSCor.

Ces figures illustrent également le rôle de nos approches filtres multivariées basées sur VNS dans la réduction du nombre de gènes, tout en conservant des taux d'exactitude élevés, cela revient à la façon dont le nombre de gènes est combiné dans la fonction objectif $\mathcal{J}$.

Sur la base des expériences que nous avons menées, nous pouvons conclure que nos approches de sélection de gènes sont bien fondées. En effet, sur les six jeux de données utilisés, nos méthodes ont atteint une grande exactitude de classification.

Dans cette section, nous avons présenté des approches filtres multivariées pour la sélection de gènes dans les données de puces à ADN. Les approches proposées en deux étapes consistent en une phase de présélection réalisée à l'aide d'un filtre de schéma univarié, puis la métaheuristique VNS couplée à un algorithme de recherche locale stochastique (SLS) est utilisée pour sélectionner le sous-ensemble de gènes final.

Ces approches visent à sélectionner un petit sous-ensemble de gènes pertinents à partir d'un ensemble de données original contenant des données redondantes, bruyantes ou non pertinentes. Les résultats expérimentaux montrent que nos approches sont très efficaces en termes d'exactitude de classification et de nombre de gènes sélectionnés.

3.6 Conclusion

Dans ce chapitre nous avons proposé plusieurs méthodes de sélection de gènes dans les puces à ADN, en utilisant les métaheuristiques BPSO,ACO et VNS, en effet nos contributions sont les suivantes :

Trois approches filtres multivariées ont été proposées, MWIS, VNSMI et VNSCorr, pour la sélection d'un sous ensemble de gènes contenant l'information nécessaire pour le problème de classification de cancer, ces méthodes sont efficaces au niveau d'exactitude de classification et temps d'exécution.

D'autres approches hybrides ont été proposée tel que : FBPSO, RBPSO et MWIS-ACO-LS, ces dernières méthodes montrent des performances élevées, mais leur temps d'exécution est aussi important à cause de l'utilisation des classifieurs dans l'étape de sélection.

La comparaison avec les travaux les plus importants dans le domaine de la sélection et de la classification des données de puces à ADN montre que nos approches sont performantes et compétitives et produisent des sous-ensembles de petite taille avec une haute performance.

Méthode intégrée pour la sélection de gènes (Fused-LASSO "Fused Least Absolute Shrinkage Selection Operator")

"Obvious" is the most dangerous word in
mathematics

Eric Temple Bell

Les méthodes de sélection de gènes sont divisées en trois catégories : filtres, enveloppes et intégrées. Les méthodes filtres sélectionnent les gènes en fonction des informations statistiques des données. Les méthodes enveloppes consistent en un processus de recherche répétitif dans lequel les résultats de l'algorithme d'apprentissage à chaque répétition sont utilisés pour guider le processus de recherche, ces méthodes sont bien détaillées dans le chapitre 2. En ce qui concerne les méthodes intégrées, la sélection de gènes est effectuée pendant l'étape d'entraînement du classifieur. Ces approches sont beaucoup plus efficaces en termes de calcul et moins sujettes au surapprentissage que les méthodes enveloppes avec des performances similaires.

Les méthodes de régularisation sont parmi les techniques intégrées importantes qui effectuent simultanément la classification et la sélection automatique de gènes. Récemment, on s'intéresse de plus en plus à l'application des techniques de régularisation dans les modèles de régression logistique (LR). La régression logistique (LR) est une méthode discriminante puissante qui a une interprétation probabiliste directe qui peut obtenir des probabilités de classification en dehors des informations sur l'étiquette de classe et une interprétation facile des coefficients des gènes [117]. Cependant, la régression logistique n'est ni applicable ni adaptée à la classification des cancers de grandes dimensions [100, 23]. En effet, Albert et al. [8] ont montré que le problème associé n'admet pas nécessairement un minimum, exactement, dans le cas où l'ensemble de données est complètement ou quasi-complètement séparé, les estimateurs du problème de minimisation sont infinis non uniques. Pour cette raison, les modèles de régression logistique pénalisés (PLR) sont utilisés pour surmonter ce problème et améliorer la précision de la classification dans les puces à

ADN.

Récemment, il y a eu un intérêt croissant pour l'application des méthodes pénalisées dans la classification des cancers, en particulier, la régression logistique pénalisée (PLR) a été appliquée avec succès pour s'attaquer simultanément à l'estimation des coefficients et à la sélection de gènes [14, 13, 9, 12, 200, 93, 100, 193]. Dans la régression logistique pénalisée (PLR), nous trouvons différentes pénalités, la pénalité la plus répandue et la plus populaire est la pénalité L_1 ($\lambda_1 \sum_{j=0}^N |\beta_j|$,

où $\lambda_1 > 0$ est le paramètre qui contrôle la régularité de la solution résultante) proposée par Tibshirani [183]. « the least absolute shrinkage selection operator » (LASSO), il s'agit d'une méthode de contraction des coefficients qui cherche à réduire à zéro les coefficients de certains gènes pour obtenir une sélection de gènes, pour cette raison, le LASSO obtient sa popularité dans les données de grande dimension. PLR avec pénalité L_1 (LASSO) donne une solution clairsemée avec une précision de classification élevée.

Bien que LASSO soit capable de faire la sélection de gènes, il présente trois inconvénients [190, 10]. Tout d'abord, lorsque l'ensemble de données est de grande dimension, le nombre de gènes sélectionnés par LASSO est majoré par celui des échantillons M . Deuxièmement, le LASSO ne tient pas compte de l'effet de regroupement lors de la sélection des gènes. LASSO devrait soit sélectionner l'ensemble du groupe de gènes fortement corrélés (s'ils sont liés à la maladie), soit tout laisser (s'ils ne sont pas liés). Cependant, LASSO ne sélectionne qu'un ou quelques membres de chaque groupe de gènes fortement corrélés. Zou et Hastie [211] ont proposé la pénalité Elastec-Net (EN) pour surmonter ces lacunes. La pénalité EN utilise une autre pénalité qui est composée linéairement de L_1 -norm et L_2 -norm. Troisièmement, la méthode LASSO est biaisée dans la sélection de gènes car elle pénalise tous les coefficients de gènes de manière égale [66]. En raison de cette faiblesse, le LASSO n'a pas les propriétés d'oracle, qui font référence à la probabilité de sélectionner le bon ensemble de gènes (avec des coefficients non nuls) converge vers un, et que les estimateurs des coefficients non nuls sont asymptotiquement normaux avec les mêmes moyennes et covariances [65]. Pour résoudre les problèmes dus au manque de propriétés d'oracle, Zou [212] a développé une nouvelle technique de régularisation appelée technique adaptative LASSO (ALASSO). Certains poids ont été utilisés pour pénaliser chaque coefficient à l'intérieur de la pénalité basée sur la norme L_1 .

Une autre façon d'améliorer le LASSO est d'envisager des termes supplémentaires dans la pénalité, plus précisément l'utilisation de pénalités structurées [21] pour tirer profit d'une structure attendue dans les paramètres de PLR, reflétant une certaine structure au niveau des gènes. C'est le cas notamment du fused LASSO [184]. Dans ce cadre, le fused LASSO consiste à ajouter à

la pénalité de type LASSO (L_1) une autre pénalité de type $\left(\lambda_2 \sum_{j=2}^N |\beta_j - \beta_{j-1}| \right)$ (avec $\lambda_2 \geq 0$

le paramètre qui contrôle la régularité de la solution résultante), il s'agit d'une généralisation du LASSO (pour $\lambda_2 = 0$, c'est LASSO classique), les gènes dans cette pénalité sont supposés ordonnés de telle sorte que g_{j-1} et g_j sont connectés (corrélés positivement) dans le sens où ils partagent des effets similaires sur la variable réponse considérée (classe). La pénalité Fused LASSO s'est avérée efficace pour la réduction des dimensions des attributs [96]. Plus précisément, en pénalisant la norme L_1 des coefficients et leurs différences successives, le Fused LASSO encourage

ainsi les solutions $\hat{\beta}(\lambda_1, \lambda_2)$ creuses vérifiant $\hat{\beta}_j(\lambda_1, \lambda_2) = \hat{\beta}_{j-1}(\lambda_1, \lambda_2)$, c'est-à-dire les solutions creuses et constantes par morceaux. Cela indique que bien que Fused LASSO et Elastic Net puissent promouvoir un regroupement des gènes sélectionnés, Elastic Net sélectionne des gènes fortement corrélés en tant que groupe tandis que Fused LASSO est plus efficace pour les gènes qui peuvent être ordonnés de manière significative. Les propriétés théoriques des estimations Fused-LASSO ont été étudiées par Rinaldo [157] et Qian et Jia [156].

L'idée du Fused LASSO a été reprise et généralisée dans le contexte du modèle de régression linéaire (et des modèles linéaires généralisés) [90, 187]. En effet, dans le modèle ci-dessus, il n'est pas nécessaire de restreindre la pénalité λ_2 aux coefficients voisins. Pour permettre une plus grande flexibilité, les auteurs de [90] ont pénalisé toutes les différences de coefficients qui correspondent à une arête dans un graphe. Plus précisément ; pour $V = \{1, \dots, N\}$ un ensemble de sommets (gènes dans une puce à ADN), dont chacun correspond à l'un des coefficients. Les auteurs de [90] ont construit un graphe $\mathcal{G} = (V, E)$ avec un ensemble d'arêtes E et pénalisé toutes les différences $|\beta_k - \beta_l|$ pour lesquelles $(k, l) \in E$. Puis Ils ont associé des poids différents pour chacune des arêtes (qui présente le degré d'association entre les gènes) afin d'obtenir la pénalité associée à la méthode Fused LASSO généralisé (FGLASSO) :

$$\lambda_1 \sum_{k=0}^N |\beta_k| + \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\beta_k - \beta_l| \quad (4.1)$$

Le Fused LASSO généralisé encourage donc les vecteurs solution $\hat{\beta}(\lambda_1, \lambda_2)$ avec une double structure : ces vecteurs auront tendance à être à la fois creux et avec des composantes non nulles égales entre elles ($\hat{\beta}_k = \hat{\beta}_l$) pour certains gènes connectés dans le graphe. Une étude théorique de propriétés de Fused LASSO généralisé est effectuée dans [187].

De plus, l'apparition de la valeur absolue dans les pénalités où il y a la norme L_1 rend la résolution du problème de minimisation associé difficile à cause de la non différentiabilité de la fonction objectif en zéro, ceci complique les calculs puisqu'il n'existe pas de solution analytique à ce problème. Parmi les premiers algorithmes utilisés pour résoudre ce problème dans le contexte de la régression linéaire nous trouvons l'algorithme (FLSA : Fused LASSO Signal Approximator) [88] et d'autres algorithmes [120, 185, 98], mais dans le contexte de classification où nous utilisons la régression logistique, il n'existe pas beaucoup d'algorithmes, nous citons comme exemple [90], dans lequel les auteurs appliquent itérativement une approximation quadratique à la fonction log-vraisemblance. À notre connaissance, il n'existe actuellement aucun algorithme optimisé disponible ayant traité directement le problème de Fused LASSO dans le cas de régression logistique. Dans cette thèse, nous présenterons un algorithme basé sur les méthodes du sous-gradient qui sont une généralisation naturelle des méthodes du gradient dans le cas où la fonction étudiée n'est pas différentiable. En effet, nous proposons un nouveau Fused LASSO généralisé (FGLASSO), composé de deux étapes le premier consiste en un pré-traitement dans lequel nous choisissons un sous ensemble de gènes initial via une méthode de filtrage uni-variée basée sur le coefficient de corrélation de Pearson, dans la seconde étape nous utilisons un nouveau Fused LASSO généralisé, dans lequel les poids $w_{k,l}$ et la construction du graphe $\mathcal{G} = (V, E)$ dépendent du coefficient de corrélation de Pearson entre les gènes. De plus nous étudions théoriquement la convergence de l'algorithme sous gradient pour résoudre le problème d'optimisation convexe non différentiable

associé. L'objectif principal de cette étude est de mettre en évidence l'efficacité de notre méthode de sélection de gènes dans la classification des cancers de grande dimension.

4.1 Approche proposée

4.1.1 Méthode du sous-gradient

La méthode du sous-gradient est un algorithme très simple pour minimiser une fonction convexe non différentiable. La méthode ressemble beaucoup à la méthode du gradient ordinaire pour les fonctions différentiables, mais avec plusieurs exceptions notables :

- La méthode du sous-gradient s'applique directement aux f non différentiables.
- Les longueurs de pas ne sont pas choisies via une recherche de ligne, comme dans la méthode du gradient ordinaire. Dans les cas les plus courants, les longueurs de pas sont fixées à l'avance.
- Contrairement à la méthode du gradient ordinaire, la méthode du sous-gradient n'est pas une méthode de descente ; la valeur de la fonction peut (et le fait souvent) augmenter.

Ces méthodes ont été introduites dans les années 60 par N.Z. Shor [172] pour la minimisation sans contrainte des fonctions convexes dans le domaine du transport de réseau. Les premiers résultats de convergence sont dus à [153, 154] ou [140]. Parmi les références classiques qui présentent un aperçu général sur les premiers développements des méthodes sous-gradient nous citons les livres [173, 137].

4.1.1.1 Éléments d'analyse convexe

Pour étudier les problèmes d'optimisation non-différentiable, il est nécessaire de recourir à des outils spécifiques dont l'étude est basée sur l'analyse convexe [137]. Dans cette section, nous présentons quelques éléments de l'analyse convexe requis pour l'étude de l'optimisation non-différentiable. Cette section emprunte des notations et différents éléments des références [158, 137].

On se place sur l'espace vectoriel $\mathbb{R}^n$, $n \in \mathbb{N}$. On le munit d'un produit scalaire $\langle \cdot, \cdot \rangle$. La norme associée au produit scalaire est notée $\| \cdot \|_2$. Elle est définie pour tout $x \in \mathbb{R}^n$ par $\|x\| = \sqrt{\langle x, x \rangle}$.

Définition 4.1.1. *Un sous ensemble C de $\mathbb{R}^n$ est dit convexe si : pour tout $(a, b) \in C^2$, le segment $[a, b] \subset C$, c'est à dire que pour tout $\alpha \in [0, 1]$, on a $\alpha a + (1 - \alpha)b \in C$. (figure.4.1)*

Étant donné $\omega_1, \dots, \omega_m \in \mathbb{R}^n$, l'élément $x = \sum_{i=1}^m \lambda_i \omega_i$, où $\sum_{i=1}^m \lambda_i = 1$ et $\lambda_i \geq 0$ pour $m \in \mathbb{N}$, est appelé une combinaison convexe de $\omega_1, \dots, \omega_m$.

Proposition 4.1.1. *Un sous-ensemble Ω de $\mathbb{R}^n$ est convexe si et seulement s'il contient toutes les combinaisons convexes de ses éléments.*

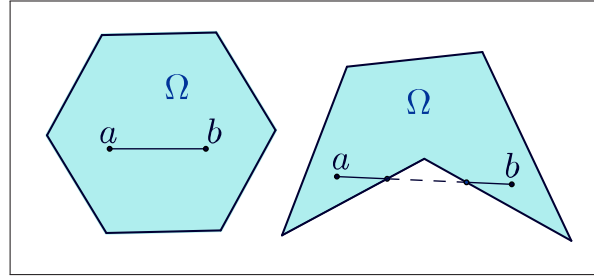


figure. 4.1: Ensemble convexe et ensemble non convexe

Définition 4.1.2. Soit $f : \Omega \rightarrow \mathbb{R}$ une fonction à valeurs réelles définie sur un ensemble convexe $\Omega \subset \mathbb{R}^n$. Alors la fonction f est convexe sur Ω si

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y) \text{ pour tout } x, y \in \Omega \text{ et } \lambda \in [0, 1] \quad (4.2)$$

Si l'inégalité dans (4.2) est stricte pour $x \neq y$ et $\lambda \in]0, 1[$, alors f est **strictement convexe** sur Ω

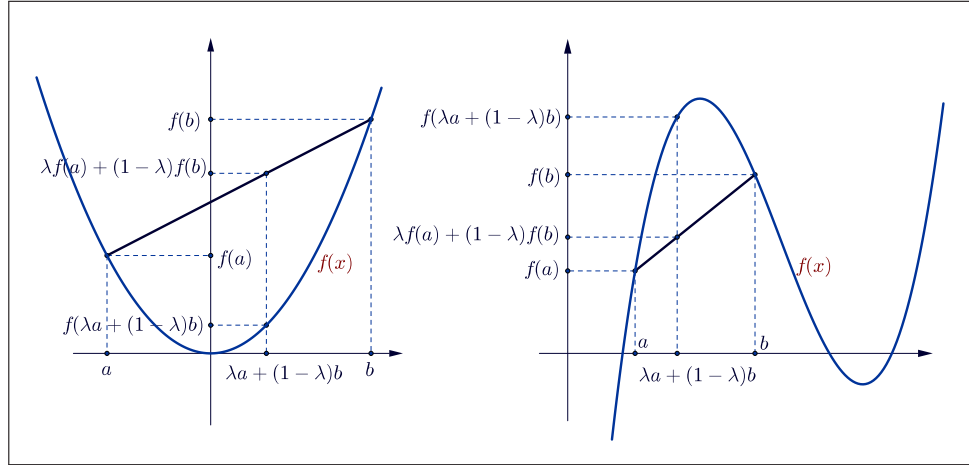


figure. 4.2: Fonction convexe et fonction non convexe.

Exemple 4.1.1. Les fonctions suivantes sont convexes :

1. $f(x) := \langle a, x \rangle + b$ pour $x \in \mathbb{R}^n$, où $a \in \mathbb{R}^n$ et $b \in \mathbb{R}$.
2. $g(x) := \|x\|_2$ pour $x \in \mathbb{R}^n$.

Théorème 4.1.1. Une fonction $f : \mathbb{R}^n \rightarrow \mathbb{R}$ est convexe si et seulement si pour tout $\lambda_i \geq 0$, $i = 1, \dots, m$ avec $\sum_{i=1}^m \lambda_i = 1$ et pour tout élément $x_i \in \mathbb{R}^n$, $i = 1, \dots, m$, nous avons

$$f\left(\sum_{i=1}^m \lambda_i x_i\right) \leq \sum_{i=1}^m \lambda_i f(x_i)$$

Définition 4.1.3. Soit $f : \Omega \rightarrow \mathbb{R}$ une fonction à valeurs réelles définie sur un ensemble convexe $\Omega \subset \mathbb{R}^n$. Alors l'épigraphe de f est défini par

$$\text{epi } f := \{(x, t) \in \Omega \times \mathbb{R} \mid x \in \Omega, t \geq f(x)\} = \{(x, t) \in \mathbb{R}^n \times \mathbb{R} \mid x \in \Omega, t \geq f(x)\}$$

Le théorème suivant donne une caractérisation géométrique de la convexité d'une fonction via la convexité de l'ensemble épigraphique associé.

Théorème 4.1.2. Soit $f : \Omega \rightarrow \mathbb{R}$ une fonction à valeurs réelles définie sur un ensemble convexe $\Omega \subset \mathbb{R}^n$. Alors f est convexe si et seulement si son épigraphe $\text{epi } f$ est un sous-ensemble convexe de l'espace produit $\mathbb{R}^n \times \mathbb{R}$

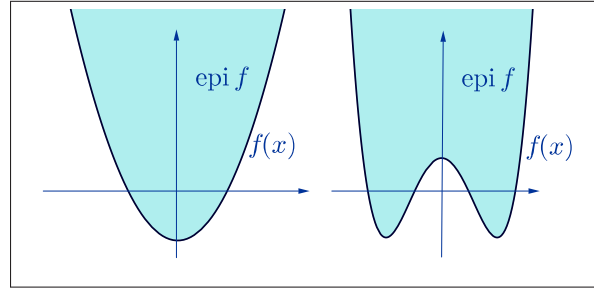


figure. 4.3: Épigraphe d'une fonction convexe et d'une fonction non convexe.

Nous montrons maintenant que la convexité est préservée sous certaines opérations importantes :

Proposition 4.1.2. 1. Soient $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$ des fonctions convexes pour tout $i = 1, \dots, m$.

Alors les fonctions suivantes sont également convexes :

(a) La multiplication par des scalaires λf pour tout $\lambda > 0$.

(b) La fonction somme $\sum_{i=1}^m f_i$.

(c) La fonction $\max_{1 \leq i \leq m} f_i$.

2. Soit $f : \mathbb{R}^n \rightarrow \mathbb{R}$ convexe et soit $\varphi : \mathbb{R} \rightarrow \mathbb{R}$ une fonction croissante et convexe sur un ensemble convexe contenant l'image de f . Alors la composition $\varphi \circ f$ est convexe.

3. Soit $B : \mathbb{R}^n \rightarrow \mathbb{R}^p$ une application affine et soit $f : \mathbb{R}^p \rightarrow \mathbb{R}$ une fonction convexe. Alors la composition $f \circ B$ est convexe.

Donnons maintenant quelques caractérisations de la convexité dans le cas où la fonction étudiée est assez régulière.

Théorème 4.1.3. Supposons que $f : \mathbb{R} \rightarrow \mathbb{R}$ soit dérivable sur son domaine de définition qui est un intervalle ouvert I . Alors f est convexe si et seulement si la dérivée f' est croissante sur I .

Corollaire 4.1.3. Soit $f : \mathbb{R} \rightarrow \mathbb{R}$ deux fois dérivable sur son domaine de définition, qui est un intervalle ouvert I . Alors f est convexe si et seulement si $f''(x) \geq 0$ pour tout $x \in I$.

Exemple 4.1.2. La fonction $f : x \mapsto \ln(1 + e^x)$ est convexe et croissante sur $\mathbb{R}$. En effet : elle est croissante comme composée de deux fonction croissante, de plus, cette fonction est deux fois dérivable sur $\mathbb{R}$ et sa dérivée seconde est $x \mapsto \frac{e^x}{(1 + e^x)^2}$ qui est une fonction positive, donc elle est convexe.

Théorème 4.1.4. Soient Ω un convexe de $\mathbb{R}^n$ et f une fonction convexe de Ω sur $\mathbb{R}$ alors : un minimum local est un minimum global.

Revenant à une fonction numérique $f : \Omega \subset \mathbb{R}^n \rightarrow \mathbb{R}$ différentiable sur l'ouvert Ω , on dit que f est deux fois différentiable en $\bar{x} \in \Omega$ lorsque $\nabla f : \Omega \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ est différentiable en $\bar{x}$. La matrice jacobienne de ∇f en $\bar{x}$ est appelée matrice hessienne de f en $\bar{x}$ et notée $\nabla^2 f(\bar{x})$; il s'agit d'une matrice symétrique de taille n dont le terme (i, j) est $\partial_i(\partial_j f)(\bar{x}) = \partial_{ij}^2 f(\bar{x})$ (dérivée partielle d'ordre 2 en $\bar{x}$ de f par rapport à la i^e et j^e variables).

Nous sommes maintenant prêts à donner la caractérisation du second ordre de la convexité pour des fonctions deux fois différentiable.

Théorème 4.1.5. Soient Ω un ouvert convexe de $\mathbb{R}^n$ et $f : \Omega \subset \mathbb{R}^n \rightarrow \bar{\mathbb{R}}$ une fonction deux fois différentiable sur Ω . Alors f est convexe si et seulement si la matrice hessienne $\nabla^2 f(\bar{x})$ est semi-définie positive pour chaque $\bar{x} \in \Omega$.

4.1.1.2 Calcul sous-différentiel

Dans cette section, nous définissons et étudions le concept de sous-différentiel (ou la collection de sous-gradients) pour les fonctions convexes à valeurs réelles. Cette notion de dérivée généralisée est l'une des plus fondamentales en analyse, avec une variété d'applications importantes. L'idée révolutionnaire derrière le concept sous-différentiel, qui le distingue des autres notions de dérivées généralisées en mathématiques, est sa valeur d'ensemble en tant qu'indication de non régularité (non différentiabilité) d'une fonction autour d'un point de référence. Il offre non seulement divers avantages et flexibilités, mais engendre également certaines difficultés dans le développement de son calcul et de ses applications. Les principales techniques d'analyse sous-différentielle portent sur la séparation convexe des ensembles.

Définition 4.1.4. Soit $f : \Omega \rightarrow \mathbb{R}$ une fonction à valeurs réelles définie sur un ensemble convexe $\Omega \subset \mathbb{R}^n$ et soit $\bar{x} \in \Omega$. Un élément $v \in \mathbb{R}^n$ est appelé un **sous-gradient** de f à $\bar{x}$ si

$$\langle v, x - \bar{x} \rangle \leq f(x) - f(\bar{x}) \text{ pour tout } x \in \Omega$$

L'ensemble de tous les sous-gradients de f à $\bar{x}$ est appelé le **sous-différentiel** de la fonction à ce point et est noté $\partial f(\bar{x})$

L'interprétation géométrique du sous-différentiel est la suivante : il est formé par toutes les directions des hyperplans qui passent par le point $(\bar{x}, f(\bar{x}))$ et restent "sous" le graphe de la fonction f . Ces hyperplans sont appelés hyperplans support ou hyperplans d'appui au graphe de f en $\bar{x}$.

Exemple 4.1.3. Soit $f : x \in \mathbb{R} \mapsto |x|$. (figure 4.4) Calculons les sous-gradients de f en tout point x de $\mathbb{R}$.

Dans $\mathbb{R}^2$, les hyperplans d'appui sont des droites et les sous-gradients associés leurs coefficients directeurs.

$$\partial f(x) = \begin{cases} \{-1\} & \text{si } x < 0 \\ \{+1\} & \text{si } x > 0 \\ [-1, 1] & \text{si } x = 0 \end{cases}$$

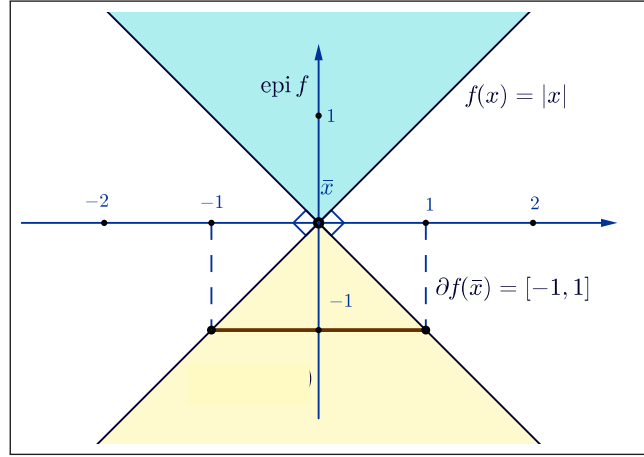


figure. 4.4: Sous-différentiel de la fonction valeur absolue à l'origine.

Ainsi, nous pouvons voir le sous-gradient de la fonction valeur absolue comme suit :

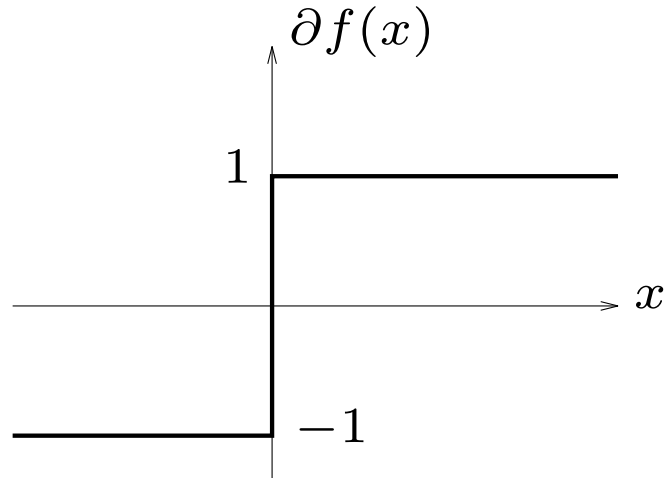


figure. 4.5: Sous-différentiel de la fonction valeur absolue.

Notons que pour f une fonction convexe et $\bar{x} \in \Omega$, on a $\partial f(x_0)$ est un ensemble non vide, convexe, borné, ceci est bien traité ainsi que d'autres résultats similaires dans [158].

Proposition 4.1.4. *Soit Ω un convexe de $\mathbb{R}^n$, $f : \Omega \rightarrow \mathbb{R}$ convexe et soit $\bar{x} \in \Omega$. Alors f atteint son minimum local/global en $\bar{x}$ si et seulement si $0 \in \partial f(\bar{x})$.*

Démonstration. Supposons que f atteigne son minimum global en $\bar{x}$. Puis

$$f(\bar{x}) \leq f(x) \text{ pour tout } x \in \Omega$$

qui peut être réécrit comme

$$0 = \langle 0, x - \bar{x} \rangle \leq f(x) - f(\bar{x}) \text{ pour tout } x \in \Omega$$

La définition du sous-différentiel montre que cela équivaut à $0 \in \partial f(\bar{x})$. □

Nous prouvons maintenant que le sous-différentiel est bien un singleton pour les fonctions différentiables se réduisant à la dérivée/gradient classique au point de référence.

Proposition 4.1.5. *Soient Ω un convexe de $\mathbb{R}^n$ et $f : \Omega \rightarrow \mathbb{R}$ convexe et différentiable en $\bar{x} \in \text{int}(\Omega)$. On a alors $\partial f(\bar{x}) = \{\nabla f(\bar{x})\}$ et*

$$\langle \nabla f(\bar{x}), x - \bar{x} \rangle \leq f(x) - f(\bar{x}) \text{ pour tout } x \in \Omega$$

Exemple 4.1.4. *Soit $p(x) := \|x\|$ la fonction de norme euclidienne sur $\mathbb{R}^n$. Nous avons alors :*

$$\partial p(x) = \begin{cases} IB & \text{si } x = 0 \\ \left\{ \frac{x}{\|x\|} \right\} & \text{sinon} \end{cases}$$

Avec IB la boule unité fermée.

Pour vérifier cela, remarquons d'abord que la fonction de norme euclidienne p est différentiable en tout point non nul avec $\nabla p(x) = \frac{x}{\|x\|}$ puisque $x \neq 0$. Il reste à calculer sa sous-différentielle en $x = 0$. Procédons par la définition du sous-gradient, nous avons $v \in \partial p(0)$ si et seulement si

$$\langle v, x \rangle = \langle v, x - 0 \rangle \leq p(x) - p(0) = \|x\| \text{ pour tout } x \in \mathbb{R}^n$$

$x = v$ nous donne $\langle v, v \rangle \leq \|v\|$, ce qui implique que $\|v\| \leq 1$, c'est-à-dire $v \in IB$. Prenons maintenant $v \in IB$ et déduisons de l'inégalité de Cauchy-Schwarz que

$$\langle v, x - 0 \rangle = \langle v, x \rangle \leq \|v\| \cdot \|x\| \leq \|x\| = p(x) - p(0) \text{ pour tout } x \in \mathbb{R}^n$$

et donc $v \in \partial p(0)$, ce qui montre que $\partial p(0) = IB$.

Théorème 4.1.6. *Soit $f : \mathbb{R}^n \rightarrow \bar{\mathbb{R}}$ une fonction différentiable sur son domaine D , qui est un ensemble convexe ouvert. Alors f est convexe si et seulement si*

$$\langle \nabla f(u), x - u \rangle \leq f(x) - f(u) \text{ pour tout } x, u \in D$$

Proposition 4.1.6. Soit Ω un ensemble convexe $\mathbb{R}^n$, considérons un nombre fini de fonctions convexes $f_i : \Omega \rightarrow \mathbb{R}$ et $\lambda_i > 0$ pour $i = 1, \dots, m$. Soit $\bar{x} \in \Omega$, alors nous avons la règle de somme des sous-différentiels

$$\partial \left(\sum_{i=1}^m \lambda_i f_i \right) (\bar{x}) = \sum_{i=1}^m \lambda_i \partial f_i(\bar{x})$$

Ainsi, le sous-différentiel d'une somme finie de fonctions convexes est la somme des sous-différentiels de ces fonctions.

Proposition 4.1.7. Composition affine :

Soit f une fonction convexe sur $\mathbf{R}^m$. Considérons l'application affine :

$$\mathcal{B}(x) = Ax + b \mid \mathbf{R}^n \rightarrow \mathbf{R}^m$$

Et $\phi(x) = f(\mathcal{B}(x))$. Alors ϕ est convexe sur $\mathbf{R}^n$, et pour tout $x \in \mathbf{R}^n$

$$\partial \phi(x) = A^T \partial f(\mathcal{B}(x))$$

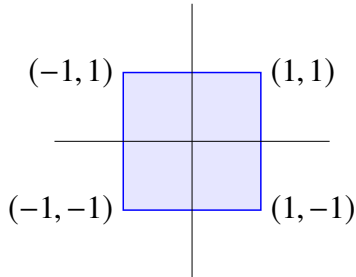
Compte tenu de ces dernières propositions nous pouvons déduire la forme du sous-différentiel de quelques fonctions qui seront utilisées dans notre approche pour la sélection de gènes dans les sous-sections suivantes :

Exemple 4.1.5. La fonction norme 1 (L_1 norm) :

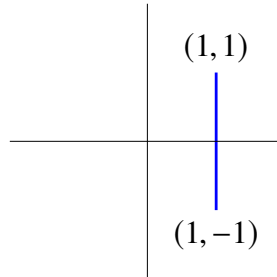
Soit $f : x \in \mathbb{R}^n \mapsto \|x\|_1 = |x_1| + \dots + |x_n|$.

Le sous-différentiel de f est le produit cartésien des intervalles suivants :

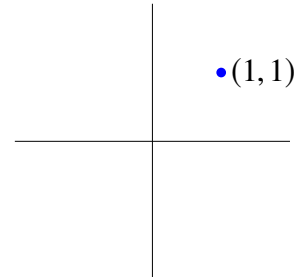
$$\partial f(x) = I_1 \times \dots \times I_n, \quad I_k = \begin{cases} [-1, 1] & x_k = 0 \\ \{1\} & x_k > 0 \\ \{-1\} & x_k < 0 \end{cases}$$



$$\partial f(0,0) = [-1, 1] \times [-1, 1]$$



$$\partial f(1,0) = \{1\} \times [-1, 1]$$



$$\partial f(1,1) = \{(1, 1)\}$$

figure. 4.6: Sous-différentiel de la fonction norme 1, pour le cas $n = 2$.

Exemple 4.1.6. Soit n un entier supérieur ou égal à 2, soit $(k, l) \in \llbracket 1, n \rrbracket^2$ avec $k < l$.

Notons $f_{k,l}$ la fonction définie sur $\mathbb{R}^n$ par $f_{k,l} : x \in \mathbb{R}^n \mapsto |x_k - x_l|$

Le sous-différentiel de $f_{k,l}$ est le produit cartésien des intervalles suivants :

$$\partial f_{k,l}(x) = I_1 \times \cdots \times I_n.$$

Avec :

$$\begin{aligned} &— I_i = \{0\} \text{ pour tout } i \in \llbracket 1, n \rrbracket \setminus \{k, l\} \\ &— I_k = \begin{cases} [-1, 1] & x_k - x_l = 0 \\ \{1\} & x_k - x_l > 0 \\ \{-1\} & x_k - x_l < 0 \end{cases} \\ &— I_l = \begin{cases} [-1, 1] & x_k - x_l = 0 \\ \{1\} & x_k - x_l < 0 \\ \{-1\} & x_k - x_l > 0 \end{cases} \end{aligned}$$

4.1.1.3 Algorithme du sous-gradient

Dans la suite nous présentons la méthode classique du sous-gradient pour les problèmes d'optimisation convexe sans contraintes :

$$\text{minimiser } f(x) \text{ sous réserve } x \in \mathbb{R}^n \quad (4.3)$$

où $f : \mathbb{R}^n \rightarrow \mathbb{R}$ est une fonction convexe. Soit (α_k) avec $k \in \mathbb{N}$ une suite de nombres positifs. L'algorithme du sous-gradient généré par (α_k) est défini comme suit, étant donné un point de départ $x_1 \in \mathbb{R}^n$, considérons la procédure itérative :

$$x_{k+1} := x_k - \alpha_k v_k \text{ avec } v_k \in \partial f(x_k), \quad k \in \mathbb{N} \quad (4.4)$$

Dans cette section, nous supposons que le problème (4.3) admet une solution optimale et que f est convexe et lipschitzienne sur $\mathbb{R}^n$. Notre objectif principal est de trouver des conditions qui assurent la convergence de l'algorithme. Nous divisons la preuve du résultat principal en plusieurs propositions, qui ont leur propre intérêt. La première nous donne une propriété importante du sous-gradient des fonctions lipschitziennes convexes.

Proposition 4.1.8. Soient $\ell \geq 0$ tel que f soit ℓ -lipschitzienne sur $\mathbb{R}^n$. Alors pour tout $x \in \mathbb{R}^n$, nous avons l'estimation du sous-gradient

$$\|v\| \leq \ell \text{ quand } v \in \partial f(x)$$

Démonstration. Soit $x \in \mathbb{R}^n$ et $v \in \partial f(x)$. Il résulte de la définition que

$$\langle v, u - x \rangle \leq f(u) - f(x) \text{ pour tout } u \in \mathbb{R}^n$$

La combinaison avec le fait que f est ℓ -lipschitzienne nous donne

$$\langle v, u - x \rangle \leq \ell \|u - x\|, \quad u \in \mathbb{R}^n$$

par suite

$$\langle v, y + x - x \rangle \leq \ell \|y + x - x\|, \text{ i.e. } \langle v, y \rangle \leq \ell \|y\|, \quad y \in \mathbb{R}^n$$

En utilisant $y := v$, nous arrivons à la conclusion $\|v\| \leq \ell$ □

Proposition 4.1.9. *Soit $\ell \geq 0$ tel que f soit ℓ -lipschitzienne sur $\mathbb{R}^n$. Pour la suite (x_k) générée par l'algorithme (4.4), on a*

$$\|x_{k+1} - x\|^2 \leq \|x_k - x\|^2 - 2\alpha_k (f(x_k) - f(x)) + \alpha_k^2 \ell^2 \text{ pour tous } x \in \mathbb{R}^n, k \in \mathbb{N} \quad (4.5)$$

Démonstration. Soient $x \in \mathbb{R}^n, k \in \mathbb{N}$ de (4.4) on déduit que

$$\begin{aligned} \|x_{k+1} - x\|^2 &= \|x_k - \alpha_k v_k - x\|^2 = \|x_k - x - \alpha_k v_k\|^2 \\ &= \|x_k - x\|^2 - 2\alpha_k \langle v_k, x_k - x \rangle + \alpha_k^2 \|v_k\|^2 \end{aligned}$$

Puisque $v_k \in \partial f(x_k)$ alors par la proposition (4.1.8) on a $\|v_k\| \leq \ell$, il s'ensuit que

$$\langle v_k, x - x_k \rangle \leq f(x) - f(x_k)$$

Ceci nous donne (4.5) et achève la preuve de la proposition. $\square$

La proposition suivante fournit une estimation de la valeur optimale de (4.3) pour une suite arbitraire (α_k) dans l'algorithme (4.4). Notons

$$\bar{V} := \min \{f(x) \mid x \in \mathbb{R}^n\} \text{ et } V_k := \min \{f(x_1), \dots, f(x_k)\}$$

Proposition 4.1.10. *Soit $\ell \geq 0$ tel que f est ℓ -lipschitzienne sur $\mathbb{R}^n$ et soit S l'ensemble des solutions optimales de (4.3). Pour tout $k \in \mathbb{N}$, nous avons*

$$0 \leq V_k - \bar{V} \leq \frac{d(x_1; S)^2 + \ell^2 \sum_{i=1}^k \alpha_i^2}{2 \sum_{i=1}^k \alpha_i} \quad (4.6)$$

Démonstration. Soit $\bar{x} \in S$, par substitution dans (4.5) on trouve

$$\|x_{k+1} - \bar{x}\|^2 \leq \|x_k - \bar{x}\|^2 - 2\alpha_k (f(x_k) - f(\bar{x})) + \alpha_k^2 \ell^2$$

Puisque $V_k \leq f(x_i)$ pour tout $i = 1, \dots, k$ et $\bar{V} = f(\bar{x})$, on a

$$\begin{aligned} \|x_{k+1} - \bar{x}\|^2 &\leq \|x_1 - \bar{x}\|^2 - 2 \sum_{i=1}^k \alpha_i (f(x_i) - f(\bar{x})) + \ell^2 \sum_{i=1}^k \alpha_i^2 \\ &\leq \|x_1 - \bar{x}\|^2 - 2 \sum_{i=1}^k \alpha_i (V_k - \bar{V}) + \ell^2 \sum_{i=1}^k \alpha_i^2 \end{aligned}$$

Il résulte de $\|x_{k+1} - \bar{x}\|^2 \geq 0$ que

$$2 \sum_{i=1}^k \alpha_i (V_k - \bar{V}) \leq \|x_1 - \bar{x}\|^2 + \ell^2 \sum_{i=1}^k \alpha_i^2$$

Puisque $V_k \geq \bar{V}$ pour tout k , on en déduit l'estimation

$$0 \leq V_k - \bar{V} \leq \frac{\|x_1 - \bar{x}\|^2 + \ell^2 \sum_{i=1}^k \alpha_i^2}{2 \sum_{i=1}^k \alpha_i}$$

D'où (4.6) puisque $\bar{x} \in S$ a été choisi arbitrairement. $\square$

Le corollaire suivant montre le nombre de pas permettant d'obtenir une estimation d'erreur appropriée pour la valeur optimale dans le cas de pas constants.

Corollaire 4.1.11. *Supposons que $\alpha_k = \epsilon$ pour tout $k \in \mathbb{N}$ dans le cadre de la proposition (4.1.10). Alors il existe une constante positive C telle que*

$$0 \leq V_k - \bar{V} < \ell^2 \epsilon \text{ lorsque } k > C/\epsilon^2$$

Démonstration. De (4.6) on obtient facilement

$$0 \leq V_k - \bar{V} \leq \frac{a + k\ell^2\epsilon^2}{2k\epsilon} = \frac{a}{2k\epsilon} + \frac{\ell^2\epsilon}{2} \text{ avec } a := d(x_1; S)^2$$

Puisque $a/(2k\epsilon) + \ell^2\epsilon/2 < \ell^2\epsilon$ comme $k > a/(\ell^2\epsilon^2)$, la conclusion tient avec $C := a/\ell^2$. $\square$

Comme conséquence directe de la proposition (4.1.10), nous obtenons notre premier résultat de convergence.

Proposition 4.1.12. *Dans le contexte de la proposition (4.1.10) supposons en plus que*

$$\sum_{k=1}^{\infty} \alpha_k = \infty \text{ et } \lim_{k \rightarrow \infty} \alpha_k = 0 \quad (4.7)$$

Ensuite, donnons la convergence $V_k \rightarrow \bar{V}$ quand $k \rightarrow \infty$.

Démonstration. Soit $\epsilon > 0$, choisissons $K \in \mathbb{N}$ tel que $\alpha_i \ell < \epsilon$ pour tout $i \geq K$.

Pour tout $k > K$, l'utilisation de (4.6) garantit les estimations :

$$\begin{aligned} 0 \leq V_k - \bar{V} &\leq \frac{d(x_1; S)^2 + \ell^2 \sum_{i=1}^K \alpha_i^2}{2 \sum_{i=1}^k \alpha_i} + \frac{\ell^2 \sum_{i=K}^k \alpha_i^2}{2 \sum_{i=1}^k \alpha_i} \\ &\leq \frac{d(x_1; S)^2 + \ell^2 \sum_{i=1}^K \alpha_i^2}{2 \sum_{i=1}^k \alpha_i} + \frac{\epsilon \sum_{i=K}^k \alpha_i}{2 \sum_{i=1}^k \alpha_i} \\ &\leq \frac{d(x_1; S)^2 + \ell^2 \sum_{i=1}^K \alpha_i^2}{2 \sum_{i=1}^k \alpha_i} + \epsilon/2 \end{aligned}$$

L'hypothèse $\sum_{k=1}^{\infty} \alpha_k = \infty$ donne $\lim_{k \rightarrow \infty} \sum_{i=1}^k \alpha_i = \infty$, et puisque la valeur de $d(x_1; S)^2 + \ell^2 \sum_{i=1}^K \alpha_i^2$ est constante par rapport à k , donc

$$\lim_{k \rightarrow \infty} \frac{d(x_1; S)^2 + \ell^2 \sum_{i=1}^K \alpha_i^2}{2 \sum_{i=1}^k \alpha_i} = 0$$

Ainsi, il existe un certain $K_1 \in \mathbb{N}$ tel que $\frac{d(x_1; S)^2 + \ell^2 \sum_{i=1}^K \alpha_i^2}{2 \sum_{i=1}^k \alpha_i} < \frac{\epsilon}{2}$ pour tout $k \geq K_1$

Par suite, on déduit : $\forall k \geq \max(K, K_1); 0 \leq V_k - \bar{V} \leq \epsilon$

Puisque $\epsilon > 0$ est choisi arbitrairement, on a $\lim_{k \rightarrow \infty} (V_k - \bar{V}) = 0$

ou $V_k \rightarrow \bar{V}$ lorsque $k \rightarrow \infty$. $\square$

Le résultat suivant nous donne des informations cruciales sur la convergence des valeurs de l'algorithme de sous-gradient (4.4).

Proposition 4.1.13. *Dans le cadre de la Proposition 4.1.12 nous avons*

$$\liminf_{k \rightarrow \infty} f(x_k) = \bar{V}$$

Démonstration. Puisque $\bar{V}$ est la valeur optimale dans (4.3), nous obtenons $f(x_k) \geq \bar{V}$ pour tout $k \in \mathbb{N}$. Cela donne

$$\liminf_{k \rightarrow \infty} f(x_k) \geq \bar{V}$$

Pour vérifier l'inégalité inverse, supposons par absurde que $\liminf_{k \rightarrow \infty} f(x_k) > \bar{V}$ puis trouvons un petit nombre $\epsilon > 0$ tel que

$$\liminf_{k \rightarrow \infty} f(x_k) - 2\epsilon > \bar{V} = \min_{x \in \mathbb{R}^n} f(x) = f(\bar{x})$$

où $\bar{x}$ est une solution optimale au problème (4.3). Ainsi

$$\liminf_{k \rightarrow \infty} f(x_k) - \epsilon > f(\bar{x}) + \epsilon$$

Dans le cas où $\liminf_{k \rightarrow \infty} f(x_k) < \infty$, par définition de $\liminf$, il existe $k_0 \in \mathbb{N}$ tel que pour chaque $k \geq k_0$ nous avons :

$$f(x_k) > \liminf_{k \rightarrow \infty} f(x_k) - \epsilon$$

et donc $f(x_k) - f(\bar{x}) > \epsilon$ pour chaque $k \geq k_0$. Notons que cette conclusion est également valable si $\liminf_{k \rightarrow \infty} f(x_k) = \infty$. Il résulte de (4.5) avec $x = \bar{x}$ que

$$\|x_{k+1} - \bar{x}\|^2 \leq \|x_k - \bar{x}\|^2 - 2\alpha_k(f(x_k) - f(\bar{x})) + \alpha_k^2 \ell^2 \text{ pour tout } k \geq k_0 \quad (4.8)$$

Soit k_1 tel que $\alpha_k \ell^2 < \epsilon$ pour tout $k \geq k_1$ et soit $\bar{k}_1 := \max\{k_1, k_0\}$. Pour tout $k > \bar{k}_1$, il résulte facilement de l'estimation (4.8) que

$$\|x_{k+1} - \bar{x}\|^2 \leq \|x_k - \bar{x}\|^2 - \alpha_k \epsilon \leq \dots \leq \|x_{\bar{k}_1} - \bar{x}\|^2 - \epsilon \sum_{j=\bar{k}_1}^k \alpha_j$$

ce qui contredit l'hypothèse $\sum_{k=1}^{\infty} \alpha_k = \infty$ et complète ainsi la preuve. $\square$

Nous sommes maintenant en mesure de justifier la convergence de l'algorithme du sous-gradient (4.4).

Théorème 4.1.7. *Supposons que la suite générée (α_k) dans (4.4) satisfait*

$$\sum_{k=1}^{\infty} \alpha_k = \infty \text{ et } \sum_{k=1}^{\infty} \alpha_k^2 < \infty \quad (4.9)$$

Alors la suite (V_k) converge vers la valeur optimale $\bar{V}$ et la suite d'itérations (x_k) dans (4.4) converge vers une solution optimale $\bar{x}$ de (4.3).

Démonstration. Puisque (4.9) implique (4.7), la suite (V_k) converge vers $\bar{V}$ par la proposition 4.1.12.

Il nous reste à prouver que (x_k) converge vers une solution optimale $\bar{x}$ du problème (4.3).

Soit $\ell \geq 0$ une constante de Lipschitz associée à f sur $\mathbb{R}^n$. Puisque l'ensemble S des solutions optimales de (4.3) est supposé non vide, choisissons un $\tilde{x} \in S$ et substituons $x = \tilde{x}$ dans (4.5) de la proposition 4.1.9. Ceci nous donne

$$\|x_{k+1} - \tilde{x}\|^2 \leq \|x_k - \tilde{x}\|^2 - 2\alpha_k (f(x_k) - \bar{V}) + \alpha_k^2 \ell^2 \text{ pour tout } k \in \mathbb{N} \quad (4.10)$$

Puisque $f(x_k) - \bar{V} \geq 0$, il découle de l'inégalité (4.10) que

$$\|x_{k+1} - \tilde{x}\|^2 \leq \|x_k - \tilde{x}\|^2 + \alpha_k^2 \ell^2$$

On obtient alors par induction

$$\|x_{k+1} - \tilde{x}\|^2 \leq \|x_1 - \tilde{x}\|^2 + \ell^2 \sum_{i=1}^k \alpha_i^2 \quad (4.11)$$

Le membre de droite de (4.11) est fini par notre hypothèse stipulant que $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$, et donc

$$\sup_{k \in \mathbb{N}} \|x_{k+1} - \tilde{x}\| < \infty$$

Cela implique que la suite (x_k) est bornée. De plus, la proposition 4.1.13 nous donne

$$\liminf_{k \rightarrow \infty} f(x_k) = \bar{V}$$

Soit (x_{k_j}) une sous-suite de (x_k) telle que

$$\lim_{j \rightarrow \infty} f(x_{k_j}) = \bar{V} \quad (4.12)$$

Alors (x_{k_j}) est également bornée, par le théorème de Bolzano-Weierstrass dans les espaces vectoriels normés de dimension finie, on peut extraire une sous suite convergente $(x_{k_{j_m}})$ de (x_{k_j}) , notons la limite de cette suite $\bar{x}$. On déduit directement de (4.12) et de la continuité de f que $\bar{x}$ est une solution optimale de (4.3). Sans perte de généralité, supposons que $x_{k_j} \rightarrow \bar{x}$ lorsque $j \rightarrow \infty$. Soient $j \in \mathbb{N}$ et $k \geq k_j$. Ainsi nous pouvons prendre $\tilde{x} = \bar{x}$ dans (4.10) et voir que

$$\|x_{k+1} - \bar{x}\|^2 \leq \|x_k - \bar{x}\|^2 + \alpha_k^2 \ell^2 \leq \dots \leq \|x_{k_j} - \bar{x}\|^2 + \ell^2 \sum_{i=k_j}^k \alpha_i^2$$

Considérons d'abord $k \rightarrow \infty$ puis la limite de $j \rightarrow \infty$ dans l'inégalité résultante ci-dessus nous donne l'estimation

$$\limsup_{k \rightarrow \infty} \|x_{k+1} - \bar{x}\|^2 \leq \lim_{j \rightarrow \infty} \|x_{k_j} - \bar{x}\|^2 + \ell^2 \lim_{j \rightarrow \infty} \sum_{i=k_j}^{\infty} \alpha_i^2$$

Puisque $x_{k_j} \rightarrow \bar{x}$ et $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$, cela implique que

$$\limsup_{k \rightarrow \infty} \|x_{k+1} - \bar{x}\|^2 = 0$$

et justifie ainsi la convergence revendiquée de $x_k \rightarrow \bar{x}$ quand $k \rightarrow \infty$. $\square$

4.1.1.4 Choix des pas α_k

L'algorithme de type sous-gradient est une méthode itérative qui génère une suite de points (x_k) où x_0 est un point initial choisi dans $\mathbb{R}^n$ via la suite suivante

$$x_{k+1} := x_k - \alpha_k v_k \text{ avec } v_k \in \partial f(x_k), \quad k \in \mathbb{N} \quad (4.13)$$

Plusieurs approches ont été proposées dans la littérature pour choisir la suite des pas α_k pour que la méthode de type sous-gradient converge. Dans la suite, nous allons discuter certains d'entre eux.

Règle (1) : Pour un pas $\alpha_k = \varepsilon$ où $\varepsilon > 0$ (on a un pas constant indépendant de k) (Corollaire 4.1.11).

Règle (2) : Une règle classique garantissant la convergence théorique de l'algorithme consiste à choisir une série divergente de carré sommable pour le pas qui satisfait : (Théorème 4.1.7)

$$\alpha_k \geq 0, \quad \sum_{k=1}^{\infty} \alpha_k^2 < \infty, \quad \sum_{k=1}^{\infty} \alpha_k = \infty$$

Règle (3) : Il est démontré que la suite V_k converge vers la valeur optimale $\bar{V}$ recherchée si la suite des pas α_k satisfait les conditions suivantes

$$\alpha_k \geq 0, \quad \lim_{k \rightarrow \infty} \alpha_k = 0, \quad \sum_{k=1}^{\infty} \alpha_k = \infty$$

Il s'agit d'un cas particulier de la règle (2)

4.1.2 Régression logistique pénalisée (PLR)

La régression logistique est une méthode statistique pour modéliser un problème de classification binaire. Dans la classification des cancers, la variable de réponse de la régression logistique a deux valeurs soit 1 pour la classe tumorale ou 0 pour la classe normale. En se basant sur les notations de la sous-section 1.2.2.5, l'estimation des coefficients β associés aux gènes est obtenue en minimisant (4.14)

$$\hat{\beta}_{LR} = \arg \min_{\beta \in \mathbb{R}^{N+1}} g(\beta) = \arg \min_{\beta \in \mathbb{R}^{N+1}} - \sum_{i=1}^M \left(y_i x'_i \beta - \ln(1 + e^{x'_i \beta}) \right) \quad (4.14)$$

La régression logistique est couramment utilisée pour modéliser les problèmes de classification et elle est considérée comme une méthode discriminante puissante. Cependant, la régression logistique n'est pas applicable dans la classification des données de grande dimension, car le problème

associé (4.14) n'admet pas nécessairement un minimum, en effet, Albert et al. [8] ont examiné le problème de la minimisation (4.14) en considérant les configurations possibles des M échantillons dans l'espace d'observation R^N . Ces configurations se répartissent essentiellement en trois catégories exhaustives et mutuellement exclusives, séparation complète, séparation quasi-complète et chevauchement (overlap), dans ce dernier papier, les auteurs se sont concentrés sur la finitude de la solution du (4.14) et sur la valeur réelle du minimum de la fonction g associée à (4.14). Nous pouvons résumer les principaux résultats de cet article comme suit :

- Lorsque l'ensemble de données est complètement ou quasi-complètement séparé, les estimateurs de (4.14) sont infinis non uniques.
- Dans le cas contraire (overlap), le minimum de ((4.14) existe et il est fini.

Le nombre de gènes élevé dans les puces à ADN peut conduire à un surajustement et les estimateurs (β) peuvent également souffrir de multicollinéarité. En effet, il n'y a que peu de gènes qui peuvent être réellement associés à la variable de réponse dans les ensembles de données de biopuces. De plus, du point de vue statistique, ces gènes non pertinents peuvent introduire du bruit et dégrader la précision de la classification. En raison de ces problèmes, il est souhaitable de développer des méthodes de sélection de gènes qui peuvent supprimer les gènes non pertinents, améliorant ainsi la précision de la classification. Ceci dit, les modèles de régression logistique pénalisés (PLR) sont appliqués pour améliorer la précision de la classification.

La régression logistique pénalisée ajoute une pénalité positive à la fonction de vraisemblance g , de sorte que certains coefficients se réduisent à zéro. Cela a pour effet de réduire les coefficients des gènes moins contributifs à zéro et ce processus est également connu sous le nom de régularisation. La log-vraisemblance pénalisée peut s'écrire sous la forme :

$$PLR = g(\beta) + \lambda P(\beta) \quad (4.15)$$

$P(\beta)$ est le terme de pénalité, λ est le paramètre de réglage contrôlant l'étendue de la pénalité, pour $\lambda = 0$, on obtient la solution de (4.14) si elle existe. En revanche, pour des valeurs élevées de λ , l'influence du terme de régularisation sur l'estimation du coefficient augmente. Le choix du paramètre de réglage est un élément indispensable de l'ajustement du modèle, dans un problème de classification, si ce paramètre est réglé par validation croisée, le classificateur peut atteindre une précision de classification satisfaisante. Le PLR est minimisé pour trouver une estimation du coefficient β . Ces méthodes pénalisées sont une famille de méthodes de sélection de gènes (méthodes intégrées) qui sont couramment utilisées dans la sélection de gènes et la classification des données de grande dimension [117].

Dans la suite, on note $\hat{\beta}_\lambda$ le vecteur de taille $N + 1$ qui présente une solution du problème de minimisation (4.15), c'est-à-dire :

$$\hat{\beta}_\lambda = \arg \min_{\beta \in \mathbb{R}^{N+1}} g(\beta) + \lambda P(\beta) \quad (4.16)$$

Plusieurs termes de pénalité ont été expliqués et appliqués dans la littérature à savoir (table 4.1) : Ridge (L_2) [89], LASSO (L_1) [183], Elastic net [211], Adaptive LASSO [212], Fused LASSO et Fused LASSO généralisé [184, 88, 185, 187, 96], Relaxed LASSO [129], Group LASSO [128], Random LASSO [190] et bien d'autres. Ces pénalités ainsi que la régression logistique peuvent être

utilisées avec succès pour obtenir des taux classification élevés. Nous nous intéressons uniquement à la pénalité de type Fused LASSO généralisé.

Table. 4.1: Méthodes de régression pénalisées populaires

Méthode	pénalité
Ridge	$\lambda \sum_{j=0}^N \beta_j^2$
LASSO	$\lambda \sum_{j=0}^N \beta_j $
Adaptive LASSO	$\lambda \sum_{j=0}^N w_j \beta_j $
Elastic net	$\lambda_1 \sum_{j=0}^N \beta_j + \lambda_2 \sum_{j=0}^N \beta_j^2$
Fused LASSO	$\lambda_1 \sum_{j=0}^N \beta_j + \lambda_2 \sum_{j=2}^N \beta_j - \beta_{j-1} $
Fused LASSO généralisé	$\lambda_1 \sum_{j=0}^N \beta_j + \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} \beta_k - \beta_l $

4.1.2.0.1 Fused LASSO généralisé

Proposé par Tibshirani et Al [184], le Fused LASSO est une extension du LASSO qui vise à améliorer les propriétés de LASSO dans le cas où les attributs sont supposés ordonnés. Elle consiste à ajouter une autre pénalité à L_1 , celle-ci a pour objectif de rendre la différence des coefficients successifs nulle. L'idée du Fused LASSO a été généralisée dans le contexte du modèle de régression linéaire (et des modèles linéaires généralisés) [90, 187]. Ceci est effectué, en construisant un graphe $\mathcal{G} = (V, E)$ avec $V = \{1, \dots, N\}$ un ensemble de sommets (gènes dans une puce à ADN), dont chacun correspond à l'un des coefficients, et un ensemble d'arêtes E . Dans le Fused LASSO généralisé, on pénalise toutes les différences $|\beta_k - \beta_l|$ pour lesquelles $(k, l) \in E$. De plus, on associe également des poids différents pour chacune des arêtes (qui présente le degré d'association entre les gènes). Ainsi on obtient la fonction objectif suivante :

$$\hat{\beta}_{FGLASSO, \lambda_1, \lambda_2} = \arg \min_{\beta \in \mathbb{R}^{N+1}} g(\beta) + \lambda_1 \sum_{k=0}^N |\beta_k| + \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\beta_k - \beta_l| \quad (4.17)$$

Ces poids $w_{k,l}$ de pénalité sont déterminés généralement à partir des données initiales. Dans (4.17), $\lambda_{k,l} = \lambda_2 \times w_{k,l}$ représente la pénalité appliquée à la différence entre les gènes g_k et g_l . Ainsi, une augmentation de $w_{k,l}$ implique alors une augmentation de la pénalisation appliquée à la différence $|\beta_k - \beta_l|$, par suite, les coefficients β_k et β_l ont plus de chance d'être égaux par cette procédure. Par ailleurs la première pénalité assure le fait d'avoir beaucoup de coefficients β_j nuls. Ainsi, les coefficients de la solution $\hat{\beta}_{FGLASSO, \lambda_1, \lambda_2}$ auront tendance à être à la fois creux et avec des

composantes non nulles égales entre elles pour certains des gènes connectés dans le graphe. Le choix de λ_1 et λ_2 est effectué à l'aide d'une validation croisée bien appropriée [92].

Grâce à la pénalité (4.17), certains coefficients de $\hat{\beta}_{FGLASSO, \lambda_1, \lambda_2}$ sont réduits à exactement zéro : les gènes associés à ces coefficients ne sont pas utilisés dans le modèle. Les paramètres λ_1 et λ_2 permettent de contrôler le degré de pénalisation qui agit directement sur le nombre de coefficients estimés non nuls : plus λ_1 augmente, plus le nombre de composantes nulles dans $\hat{\beta}_{FGLASSO, \lambda_1, \lambda_2}$ augmente. Pour λ_1 et λ_2 , deux paramètres donnés, le vecteur de taille $N + 1$ des coefficients de régression estimés associé sera noté $\hat{\beta}_{FGLASSO, \lambda_1, \lambda_2}$. L'ensemble de gènes sélectionnés est défini comme suit :

$$\mathcal{S}(\lambda_1, \lambda_2) = \left\{ g_j : \hat{\beta}_{FGLASSO, \lambda_1, \lambda_2, j} \neq 0; j = 1, \dots, N \right\}$$

Avec $\hat{\beta}_{FGLASSO, \lambda_1, \lambda_2, j}$ est le coefficient de régression logistique pénalisée (4.17) associé au gène g_j pour les paramètres de pénalité λ_1 et λ_2

4.2 Méthode proposée (FGLASSO-LR)

Dans le domaine de classification des données de grande dimension en particulier les puces à ADN, le LASSO classique présente plusieurs inconvénients [190, 10]. Le premier, c'est qu'il ne tient pas compte de l'effet de regroupement lors de la sélection des gènes, il peut alors sélectionner un ensemble de gènes fortement corrélés [211]. D'autre part, la méthode LASSO est biaisée dans la sélection de gènes car elle pénalise tous les coefficients de gènes de manière égale [66], pour pallier ces difficultés, plusieurs méthodes sont proposées à savoir : Elastic net [211], Adaptive LASSO [212], Fused LASSO et Fused LASSO généralisé [184, 88, 185, 187, 96], Relaxed LASSO [129], Group LASSO [128], Random LASSO [190] et bien d'autres. Nous formulons le problème de sélection des sous-ensembles de gènes correspondant comme un problème de régression logistique (maximum de log-vraisemblance) associé à un régularisateur Fused LASSO généralisé, cette dernière pénalité n'est jamais utilisée dans ce type de problèmes. La sélection est effectuée alors en minimisant la fonction suivante :

$$\hat{\beta}_{FGLASSO, \lambda_1, \lambda_2} = \arg \min_{\beta \in \mathbb{R}^{N+1}} g(\beta) + \lambda_1 \sum_{k=0}^N |\beta_k| + \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\beta_k - \beta_l| \quad (4.18)$$

Tout d'abord, nous illustrons comment concevoir notre graphe $\mathcal{G} = (V, E)$ et le poids $w_{k,l}$ de chaque arête (k, l) . En effet, considérons $V = \{1, \dots, N\}$ l'ensemble des sommets (gènes) associé à notre puce à ADN, compte tenu des notions présentées dans la sous-section (3.4.1.1), la mesure d'interaction entre les gènes est calculée sur la base de coefficient de corrélation de Pearson; nous construisons donc une arête entre deux sommets k et l si et seulement si la corrélation absolue entre les gènes g_k et g_l est supérieure ou égale à un seuil (threshold) r_0 , d'autre part, nous définissons le poids d'une arête (k, l) comme étant la valeur absolue de la corrélation entre g_k et g_l ($w_{k,l} = |\rho_{g_k, g_l}|$), ainsi le graphe $\mathcal{G} = (V, E)$ et les poids $w_{k,l}$ sont créés.

Les raisons d'utiliser Fused LASSO sont les suivantes [45] :

- Lorsque le nombre de gènes est supérieur à celui des échantillons, le nombre maximum de gènes sélectionnés par LASSO ne peut pas dépasser le nombre d'échantillons. Bien qu'Elastic

Net puisse obtenir de meilleures performances que LASSO, il est moins efficace que Fused LASSO lorsqu'il existe une relation entre les gènes.

- S'il existe une relation entre les gènes, Fused LASSO exploite cette relation en régularisant explicitement les différences entre les coefficients voisins à l'aide d'un régularisateur de norme L_1 . Ainsi, les coefficients finaux auront tendance à être à la fois creux et avec des composantes non nulles égales entre elles pour certains gènes connectés dans le graphe. C'est-à-dire que pour la première pénalité LASSO assure la sélection des gènes pertinents, d'autre part le Fused LASSO généralisé sélectionne les gènes voisins qui sont très pertinents pour la cible (classe), et améliore ainsi le compromis entre la pertinence individuelle de chaque gène et la redondance entre les gènes.

4.2.0.0.1 Structure générale de l'approche proposée

La procédure générale de notre approche peut être décrite par un processus séquentiel en deux étapes (4.7) qui utilise des techniques complémentaires pour réduire progressivement l'espace de recherche et sélectionner un sous-ensemble pertinent de gènes.

Étape 1 : C'est une étape de pré-traitement qui a pour objectif d'éliminer les gènes non pertinents, par exemple, les gènes dont les niveaux d'expression sont uniformes quelle que soit la classe. De plus, il y a des milliers de gènes, il est alors inefficace d'adopter un algorithme de nature non-déterministe (comme l'algorithme sous-gradient) directement dans un ensemble de données de biopuces sans filtration. La sortie de ce pré-traitement est un ensemble de gènes classés par ordre de pertinence selon le score de corrélation absolue entre les gènes et la classe cible. Il s'agit d'une étape préliminaire pour réduire la taille des données des puces à ADN, dans laquelle le filtre de corrélation de Pearson sélectionne les 200 gènes les mieux classés.

Étape 2 : Cette étape consiste en une méthode intégrée (embedded), qui s'applique sur les gènes retenus par le filtrage précédent. Dans cette étape la sélection est effectuée durant la phase de construction du classifieur de régression logistique. La construction du modèle de classification consiste à trouver le $\hat{\beta}_{\text{FGLASSO}, \lambda_1, \lambda_2}$ qui résout le problème de minimisation (4.18), et pour ce faire, l'algorithme du sous-gradient est appliqué et étudié pour trouver de meilleurs résultats.

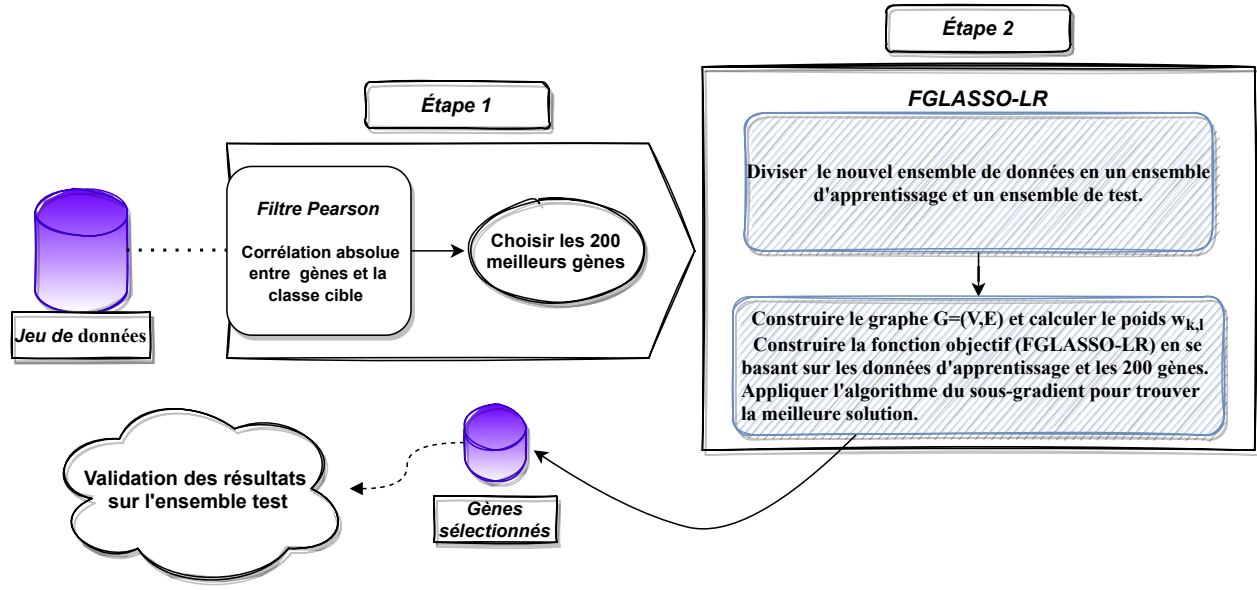


figure. 4.7 : Organigramme de l'approche proposée (*FGLASSO – LR*) pour la sélection de gènes dans les données de puces à ADN.

En ce qui concerne l'estimation de $\hat{\beta}_{\text{FGLASSO},\lambda_1,\lambda_2}$ dans (4.18), puisque la fonction en question n'est pas différentiable à cause de la présence de la valeur absolue, nous avons utilisé l'algorithme du sous-gradient pour résoudre le problème de minimisation (4.18), ce dernier n'a jamais été utilisé pour résoudre le problème de régression logistique avec pénalité de type Fused LASSO, dans la suite nous prouvons la convergence de cet algorithme pour (4.18).

4.2.1 Sous-gradient pour l'estimation de $\hat{\beta}_{\text{FGLASSO},\lambda_1,\lambda_2}$

Nous avons vu dans la sous-section 4.1.1 que l'algorithme du sous-gradient converge vers la solution optimale d'un problème de minimisation convexe en choisissant un pas (α_k) approprié, sous réserve que la fonction en question vérifie les conditions suivantes :

- Elle est convexe.
- Elle admet un minimum global.
- Elle est lipschitzienne.

Revenons à notre problème (4.17), il faudrait d'abord s'assurer que notre fonction vérifie ces trois conditions.

Notons

$$h_{\lambda_1,\lambda_2}(\beta) = g(\beta) + \lambda_1 \sum_{j=0}^N |\beta_j| + \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\beta_k - \beta_l| \quad \forall \beta \in \mathbb{R}^{N+1} \quad (4.19)$$

1. La convexité de h_{λ_1,λ_2} :

Nous avons tout d'abord

$$g(\beta) = -\sum_{i=1}^M \left(y_i x'_i \beta - \ln(1 + e^{x'_i \beta}) \right) = -\sum_{i=1}^M y_i x'_i \beta + \sum_{i=1}^M \ln(1 + e^{x'_i \beta}) \quad \forall \beta \in \mathbb{R}^{N+1}$$

Montrons que g est convexe sur $\mathbb{R}^{N+1}$:

- La fonction $\beta \mapsto -\sum_{i=1}^M y_i x'_i \beta$ est convexe, il s'agit d'une forme linéaire sur $\mathbb{R}^{N+1}$.
- La fonction $x \mapsto \ln(1 + e^x)$ est convexe et croissante sur $\mathbb{R}$ (exemple 4.1.2).
- La fonction $\beta \mapsto x'_i \beta$ est convexe pour tout i de $[[1, M]]$, donc $\beta \mapsto \ln(1 + e^{x'_i \beta})$ est convexe pour tout i de $[[1, M]]$. (c'est la composée d'une fonction croissante convexe sur $\mathbb{R}$ et une fonction convexe sur $\mathbb{R}^{N+1}$ (proposition 4.1.2)).

La fonction g est donc **convexe** sur $\mathbb{R}^{N+1}$, comme somme de $M + 1$ fonctions convexes.

D'autre part, la fonction $\beta \mapsto \lambda_1 \sum_{j=0}^N |\beta_j|$ est convexe comme somme de $N + 1$ fonctions convexes.

Il nous reste alors à montrer que la fonction $\beta \mapsto \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\beta_k - \beta_l|$ est convexe.

En effet : soit $(k, l) \in [[1, N]]^2$ avec $k < l$. La fonction $\beta \mapsto \lambda_2 w_{k,l} |\beta_k - \beta_l| = f_{k,l}(\beta)$ est convexe, ceci revient au fait que la fonction $f_{k,l}$ définie dans l'exemple (4.1.6) est convexe comme composée d'une fonction affine et d'une autre convexe, $\lambda_2 w_{k,l} \geq 0$. Ensuite, la fonction $\beta \mapsto \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\beta_k - \beta_l|$ est convexe comme étant une somme de fonctions convexes.

Finalement, nous pouvons déduire que la fonction h_{λ_1, λ_2} est convexe comme somme de fonctions convexes.

2. L'existence d'un minimum global de h_{λ_1, λ_2} :

En se basant sur les notations de la section 1.2.2.5 nous avons :

$$\forall \beta \in \mathbb{R}^{N+1}, \quad L(\beta) = \prod_{i=1}^M \Pi_i^{y_i} (1 - \Pi_i)^{1-y_i} \quad \text{et} \quad g(\beta) = -\ln(L(\beta))$$

avec $\Pi_i = \frac{e^{x'_i \beta}}{1 + e^{x'_i \beta}}$ pour tout i de $[[1, M]]$.

De plus, nous avons :

$$\forall i \in [[1, M]], 0 < \Pi_i < 1 \text{ et } y_i \in \{0, 1\}$$

Donc

$$0 < L(\beta) < 1 \quad \forall \beta \in \mathbb{R}^{N+1}$$

Par suite

$$g(\beta) \in \mathbb{R}^{+*} \quad \forall \beta \in \mathbb{R}^{N+1}$$

Et puisque le λ_1 dans (4.19) est strictement positif alors :

$$h_{\lambda_1, \lambda_2}(\beta) > 0 \quad \forall \beta \in \mathbb{R}^{N+1}.$$

Notons

$$A = \{h_{\lambda_1, \lambda_2}(\beta), \quad \beta \in \mathbb{R}^{N+1}\}$$

Il s'agit d'une partie non vide de $\mathbb{R}$ minorée par 0, donc elle admet une borne inférieure, notons cette borne $a = \inf A \in \mathbb{R}^+$.

Dans l'étape suivante nous montrons que cette borne est atteinte, c'est-à-dire $\inf A = \min A$, ou tout simplement :

$$\exists \beta^* \in \mathbb{R}^{N+1}, \quad h_{\lambda_1, \lambda_2}(\beta^*) = a.$$

D'après la caractérisation séquentielle de la borne inférieure, il existe une suite $(\gamma_n) \subset \mathbb{R}^{N+1}$ telle que

$$\lim_{n \rightarrow +\infty} h_{\lambda_1, \lambda_2}(\gamma_n) = a$$

Montrons que la suite (γ_n) est bornée avec :

$$\gamma_n = (\gamma_{n,0}, \gamma_{n,1}, \dots, \gamma_{n,N}) \quad \forall n \in \mathbb{N}.$$

Remarquons que la suite $(h_{\lambda_1, \lambda_2}(\gamma_n))$ est convergente donc elle est bornée. c'est-à-dire :

$$\exists M_1 \in \mathbb{R}^+, \quad \forall n \in \mathbb{N} \quad |h_{\lambda_1, \lambda_2}(\gamma_n)| \leq M_1.$$

Ou tout simplement :

$$\exists M_1 \in \mathbb{R}^+, \quad \forall n \in \mathbb{N} \quad 0 \leq h_{\lambda_1, \lambda_2}(\gamma_n) \leq M_1.$$

Examinons maintenant l'expression de $(h_{\lambda_1, \lambda_2}(\gamma_n))$. Vu que la fonction g est positive sur $\mathbb{R}^{N+1}$ alors :

$$M_1 \geq h_{\lambda_1, \lambda_2}(\gamma_n) \geq \lambda \|\gamma_n\|_1.$$

Ainsi

$$\|\gamma_n\|_1 \leq \frac{M_1}{\lambda}.$$

Finalement :

$$\forall n \in \mathbb{N} \quad \|\gamma_n\|_1 \leq \frac{M_1}{\lambda_1}.$$

Nous déduisons alors que la suite (γ_n) est bornée dans $\mathbb{R}^{N+1}$, donc (γ_n) admet une sous suite $(\gamma_{\phi(n)})$ convergente.

Notons la limite de cette dernière sous suite β^* , nous avons alors :

$$\gamma_{\phi(n)} \longrightarrow \beta^* \quad \text{lorsque} \quad n \rightarrow +\infty$$

et par continuité de h_{λ_1, λ_2} sur $\mathbb{R}^{N+1}$ nous aboutissons à :

$$h_{\lambda_1, \lambda_2}(\gamma_{\phi(n)}) \longrightarrow h_{\lambda_1, \lambda_2}(\beta^*) \quad \text{lorsque} \quad n \rightarrow +\infty.$$

Puisque $(h_{\lambda_1, \lambda_2}(\gamma_{\phi(n)}))$ est une sous suite de $(h_{\lambda_1, \lambda_2}(\gamma_n))$ alors :

$$h_{\lambda_1, \lambda_2}(\gamma_{\phi(n)}) \longrightarrow a \quad \text{lorsque} \quad n \rightarrow +\infty.$$

Ainsi, par unicité de la limite nous concluons que :

$$h_{\lambda_1, \lambda_2}(\beta^*) = a.$$

Finalement, nous avons obtenu que la fonction $h_{\lambda_1, \lambda_2}a$ admet **un minimum global** a atteint au point β^* .

3. **La fonction h_{λ_1, λ_2} est ℓ -lipschitzienne sur $\mathbb{R}^{N+1}$:**

Il ne nous reste qu'une seule condition à établir pour montrer que l'algorithme du sous-gradient converge vers un minimum global de h_{λ_1, λ_2} , il s'agit de prouver que h_{λ_1, λ_2} est ℓ -lipschitzienne sur $\mathbb{R}^{N+1}$, c'est-à-dire trouver un ℓ positif tel que :

$$\forall \beta, \gamma \in \mathbb{R}^{N+1}, \quad |h_{\lambda_1, \lambda_2}(\beta) - h_{\lambda_1, \lambda_2}(\gamma)| \leq \ell \|\beta - \gamma\|$$

Notons au passage que la constante de Lipschitz ℓ dépend du choix de la norme sur $\mathbb{R}^{N+1}$. Vu que toutes les normes sont équivalentes sur $\mathbb{R}^{N+1}$, le fait qu'une fonction soit lipschitzienne ou non ne dépend pas de la norme choisie.

Commençons par montrer que la fonction g est lipschitzienne, nous avons tout d'abord la proposition suivante :

Proposition 4.2.1. (Inégalité des accroissements finis)

Soient U un ouvert de $\mathbb{R}^{N+1}$, $f : U \rightarrow \mathbb{R}$ de classe C^1 et β, γ deux points de U tel que le segment $[\beta, \gamma] = \{\beta + t(\gamma - \beta), t \in [0, 1]\}$ soit inclus dans U . Alors

$$|f(\gamma) - f(\beta)| \leq \left(\sup_{x \in [\beta, \gamma]} \|\nabla f(x)\| \right) \|\gamma - \beta\|$$

En particulier si $f : \mathbb{R}^{N+1} \rightarrow \mathbb{R}$ est de classe C^1 , et il existe un certain M_1 positif tel que :

$$\forall x \in \mathbb{R}^{N+1}, \quad \|\nabla f(x)\| \leq M_1$$

Alors f est M_1 -lipschitzienne sur $\mathbb{R}^{N+1}$.

D'une autre part nous avons :

$$g(\beta) = - \sum_{i=1}^M \left(y_i x'_i \beta - \ln(1 + e^{x'_i \beta}) \right) \quad \forall \beta \in \mathbb{R}^{N+1}$$

La fonction g est de classe C^1 sur $\mathbb{R}^{N+1}$. Cherchons les dérivées partielles de g : Soit $\beta \in \mathbb{R}^{N+1}$, nous avons :

$$\forall j \in [0, N], \quad \frac{\partial g(\beta)}{\partial \beta_j} = - \sum_{i=1}^M \left(y_i x'_{ij} - \Pi_i x'_{ij} \right)$$

Par conséquent :

$$\|\nabla g(\beta)\|_1 \leq \sum_{j=0}^N \left(\sum_{i=1}^M (|y_i x'_{ij}| + |x'_{ij}|) \right)$$

Soit $M_1 = \sum_{j=0}^N \left(\sum_{i=1}^M (|y_i x'_{ij}| + |x'_{ij}|) \right)$, nous avons alors :

$$\|\nabla g(\beta)\|_1 \leq M_1$$

Comme nous travaillons sur l'espace vectoriel normé $\mathbb{R}^{N+1}$ de dimension finie, il existe alors un certain M_2 positif tel que :

$$\|\nabla g(\beta)\| \leq M_2$$

Nous pouvons remarquer que le réel M_2 ne dépend pas de β , plus précisément il dépend seulement des données de l'ensemble d'apprentissage, ainsi nous obtenons :

$$\forall \beta \in \mathbb{R}^{N+1}, \quad \|\nabla g(\beta)\| \leq M_2 \quad (4.20)$$

Donc g est une fonction M_2 -lipschitzienne sur $\mathbb{R}^{N+1}$ d'après la proposition 4.2.1.

Dans la suite nous allons prouver que la fonction h_{λ_1, λ_2} est lipschitzienne.

Soient $\beta, \gamma \in \mathbb{R}^{N+1}$, nous avons :

$$|\lambda_1 \sum_{j=0}^N |\beta_j| - \lambda_1 \sum_{j=0}^N |\gamma_j| | = \lambda_1 \left| \sum_{j=0}^N (|\beta_j| - |\gamma_j|) \right| \leq \lambda_1 \sum_{j=0}^N ||\beta_j| - |\gamma_j|| \leq \lambda_1 \|\beta - \gamma\|_1$$

De même, puisque les poids $w_{k,l} \in]0; 1]$:

$$\begin{aligned} |\lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\beta_k - \beta_l| - \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\gamma_k - \gamma_l| | &= \lambda_2 \left| \sum_{(k,l) \in E, k < l} w_{k,l} (|\beta_k - \beta_l| - |\gamma_k - \gamma_l|) \right| \\ &\leq \lambda_2 \sum_{(k,l) \in E, k < l} |(\beta_k - \beta_l) - (\gamma_k - \gamma_l)| \\ &\leq \lambda_2 \frac{N(N-1)}{2} \|\beta - \gamma\|_1 \end{aligned}$$

Ainsi :

$$\forall \beta, \gamma \in \mathbb{R}^{N+1}, \quad \left| \lambda_1 \sum_{j=0}^N |\beta_j| - \lambda_1 \sum_{j=0}^N |\gamma_j| \right| \leq \lambda_1 \|\beta - \gamma\|_1$$

et

$$\forall \beta, \gamma \in \mathbb{R}^{N+1}, \quad \left| \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\beta_k - \beta_l| - \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\gamma_k - \gamma_l| \right| \leq \lambda_2 \frac{N(N-1)}{2} \|\beta - \gamma\|_1$$

Par équivalence des normes, nous pouvons déduire qu'il existe certains M_3 et M_4 positifs tel que :

$$\forall \beta, \gamma \in \mathbb{R}^{N+1}, \quad \left| \lambda_1 \sum_{j=0}^N |\beta_j| - \lambda_1 \sum_{j=0}^N |\gamma_j| \right| \leq M_3 \|\beta - \gamma\| \quad (4.21)$$

$$\forall \beta, \gamma \in \mathbb{R}^{N+1}, \quad \left| \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\beta_k - \beta_l| - \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} |\gamma_k - \gamma_l| \right| \leq M_4 \|\beta - \gamma\| \quad (4.22)$$

Il résulte de (4.20), (4.21) et (4.22) que :

$$\forall \beta, \gamma \in \mathbb{R}^{N+1}, \quad |h_{\lambda_1, \lambda_2}(\beta) - h_{\lambda_1, \lambda_2}(\gamma)| \leq (M_2 + M_3 + M_4) \|\beta - \gamma\|$$

Finalement, il suffit de prendre $\ell := M_2 + M_3 + M_4$ pour déduire que h_{λ_1, λ_2} est ℓ -lipschitzienne sur $\mathbb{R}^{N+1}$.

4.2.1.1 Algorithme de sous-gradient

Après avoir confirmé la convergence de notre algorithme du sous-gradient pour estimer β passons maintenant à détailler la méthode pour la fonction objectif h_λ .

Soit $\beta \in \mathbb{R}^{N+1}$, calculons $\partial h_{\lambda_1, \lambda_2}(\beta)$: Nous avons d'après la proposition 4.1.6 :

$$\partial h_{\lambda_1, \lambda_2}(\beta) = \partial g(\beta) + \lambda_1 \partial \|\beta\|_1 + \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} \partial f_{k,l}(\beta)$$

Avec $f_{k,l}$ est la fonction définie dans l'exemple (4.1.6), or g est différentiable sur $\mathbb{R}^{N+1}$ alors nous avons $\partial g(\beta) = \nabla g(\beta)$ (proposition 4.1.5), d'autre part, les sous différentiels de la fonction norme 1 et L_1 , et $f_{k,l}$ sont donnés respectivement dans les exemples 4.1.5 et 4.1.6 ce qui nous mène à

$$\forall \beta \in \mathbb{R}^{N+1}, \quad \partial h_{\lambda_1, \lambda_2}(\beta) = \nabla g(\beta) + \lambda_1 \partial \|\beta\|_1 + \lambda_2 \sum_{(k,l) \in E, k < l} w_{k,l} \partial f_{k,l}(\beta)$$

Ainsi pour une suite (α_k) bien choisie, l'algorithme de recherche de β optimal pour le problème (4.18) est le suivant :

Algorithme 13 Algorithme de sous-gradient pour minimiser h_{λ_1, λ_2} .

Fonction : $GFLASSO - LR(X, it_{max}, \lambda_1, \lambda_2, r_0, (\alpha_k))$

Entrées : X = Ensemble D'apprentissage, it_{max} : Le nombre maximal d'itérations. F = vecteur de poids de chaque gènes. λ_1 et λ_2 : les coefficients qui contrôlent les pénalités, r_0 le seuil pour la construction d'une arrête dans le graphe $\mathcal{G} = (V, E)$, (α_k) la suite qui contrôle les pas dans l'algorithme sous-gradient.

Sortie : β_{best} : Le meilleur β .

Construire le graphe $\mathcal{G} = (V, E)$, et calculer les poids $w_{k,l}$

$k=1$;

Choisir β^1 aléatoirement dans $\mathbb{R}^{N+1}$

$\beta^* = \beta^1, h_{\lambda_1, \lambda_2}^* = h_{\lambda_1, \lambda_2}(\beta^1)$

$k = 2$

Tant que $k < it_{max}$

$\beta^k = \beta^{k-1} - \alpha_k * v_{k-1}$ avec v_{k-1} un vecteur choisi aléatoirement de $\partial h_{\lambda_1, \lambda_2}(\beta^{k-1})$

if $h_{\lambda_1, \lambda_2}(\beta^k) \leq h_{\lambda_1, \lambda_2}^*$ **then**

$\beta^* = \beta^k, h_{\lambda_1, \lambda_2}^* = h_{\lambda_1, \lambda_2}(\beta^k)$.

end if

$k = k + 1$.

Fin Tant que

Return β^* .

Grâce à la pénalité Fused LASSO généralisé, plusieurs coefficients dans β^* sont mis à zero, ainsi nous avons effectué une sélection de gènes intégrée dans notre modèle de classification. En effet, les gènes associés à des β_j^* nuls n'ont aucun rôle dans la classification d'un nouvel exemple x puisque la règle de classification dans la régression logistique est la suivante :

$$Classe(x) = \begin{cases} 0 & \text{si } \frac{1}{1 + e^{-x' \beta^*}} < 0.5 \\ 1 & \text{si } \frac{1}{1 + e^{-x' \beta^*}} \geq 0.5 \end{cases}$$

4.3 Expérimentations et résultats

Cette section expose les performances de notre approche proposée sur quatre ensembles de données de classification d'expression génique bien connus Table 4.2. Les réglages des paramètres et les résultats numériques seront décrits dans les sous-sections suivantes.

La mise en œuvre de l'approche proposée est faite sous Matlab R2018a.

De plus, dans cette étude, nous divisons chaque jeu de données en un ensemble d'apprentissage et un ensemble de test pour mesurer la qualité des sous-ensembles de gènes sélectionnés par notre approche.

4.3.1 Jeux de données

Pour évaluer nos approches, nous avons choisi des jeux de données (DNA microarray), qui concernent tous des problèmes de reconnaissance des cancers, facilement accessibles au public. De plus, ces jeux de données ont été utilisés dans plusieurs travaux de classification supervisés (table 4.2). Ces derniers ensembles de données ont une multitude de caractéristiques distinctives : Le nombre d'échantillons dans certains jeux de données est faible, tandis que d'autres ont un nombre plus élevé. En outre, tous les jeux de données ont des classes binaires. Et comme notre méthode proposée est conçue pour les puces à ADN de grande dimension, tous les ensembles de données sont décrits par un nombre élevé de gènes (variant entre 2000 et 12600 gènes).

Table. 4.2: Description des jeux de données (ADN microarray) utilisés dans FGLASSO-LR.

Jeu de données	# gènes	# échantillons	# classes	source
Colon	2000	62	2	[16]
DLBCL	5469	77	2	[179]
Prostate Tumor	10509	102	2	[179]
Prostate	12600	102	2	[176]

4.3.2 Paramètres

Dans notre approche, nous commençons par une étape de pré-traitement qui consiste à sélectionner les 200 gènes les mieux classés en utilisant le score de pertinence (coefficient absolu de corrélation), puis, nous divisons notre jeu de données aléatoirement en un ensemble d'apprentissage et un ensemble de test, et par suite, nous construisons notre fonction objectif (4.18) à l'aide de l'ensemble d'apprentissage. Pour minimiser la fonction en question, nous appliquons l'algorithme de sous-gradient (algorithme 13), ce dernier algorithme dépend fortement du choix de pas (α_k), en effet un meilleur choix de (α_k) peut affecter la vitesse de convergence, ainsi que la qualité de l'estimation trouvée.

4.3.2.1 Choix de (α_k)

Le choix d'un pas (α_k) approprié pour l'algorithme de sous gradient, est une étape primordiale dans notre étude, pour cela, nous avons fixé dans un premier temps $\lambda_1 = 1$, $\lambda_2 = 0.2$ et $r_0 = 0.7$, puis le jeu de données "Prostate_Tumor" est appelé pour faire une étude paramétrique de la manière suivante :

Nous avons commencer par choisir les 200 gènes les mieux classés, puis le nouveau jeu de données trouvé après l'étape de filtrage est découpé en deux ensembles disjoints (80 % d'apprentissage et 20 % de test), et par suite la fonction h_{λ_1, λ_2} est construite sur la base de l'ensemble d'apprentissage. Et puisque l'objectif de cette sous-section est de choisir le pas (α_k) qui accélère la convergence de notre algorithme vers le minimum global, nous n'aurons pas besoin de faire du test de classifieur, en

effet, nous nous intéressons juste au comportement de l'algorithme de sous-gradient pour trouver le minimum de h_{λ_1, λ_2} .

Dans un premier temps nous commençons par faire une comparaison entre le pas classique $\alpha_k = \frac{1}{k}$ et le pas donné par Alber et al dans [7], $\alpha'_k = \frac{1}{k} * \frac{1}{\max(1, \|v_{k-1}\|)}$, avec v_{k-1} un vecteur choisi aléatoirement de $\partial h_{\lambda_1, \lambda_2}(\beta^{k-1})$ (algorithme 13), avec un nombre maximal d'itérations de 10000. Nous pouvons vérifier facilement que ces pas assurent la convergence de notre algorithme de sous-gradient vers le minimum global.

D'après la figure 4.8, nous pouvons remarquer que l'utilisation de α'_k accélère la convergence de l'algorithme de sous-gradient, cela revient à l'utilisation du terme v_{k-1} dans α'_k .

Dans le but du choix d'un pas optimal, nous nous inspirons du [7] et nous utilisons des pas de type $\alpha'_k = \alpha_k * \frac{1}{\max(1, \|v_{k-1}\|)}$, avec v_{k-1} un vecteur choisi aléatoirement de $\partial h_{\lambda_1, \lambda_2}(\beta^{k-1})$ (algorithme 13), et (α_k) une suite à termes positifs, vérifiant :

$$\sum_{k=1}^{\infty} \alpha_k = \infty \text{ et } \lim_{k \rightarrow \infty} \alpha_k = 0$$

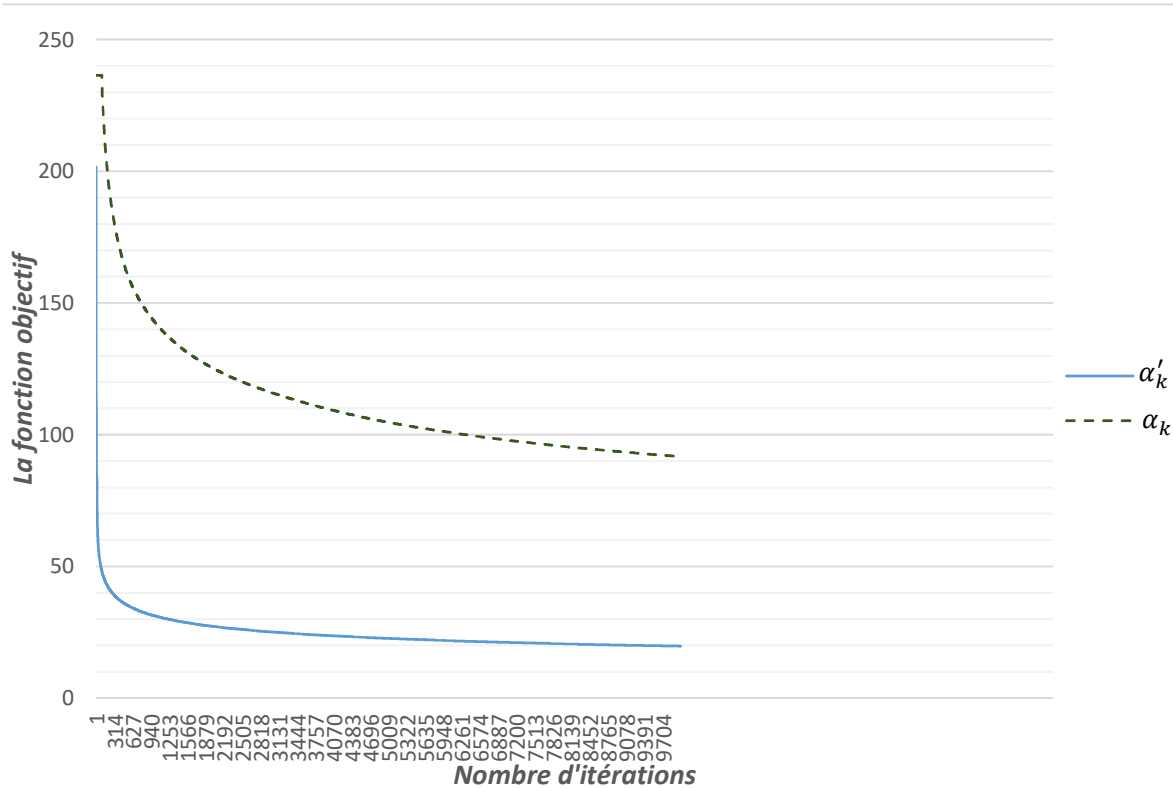


figure. 4.8: Comparaison entre les pas α_k et α'_k pour minimiser la fonction h_{λ_1, λ_2} .

Les derniers pas assurent la convergence de l'algorithme de sous-gradient. Dans la suite nous

comparons différentes formes de (α_k) de types $\frac{1}{k^\gamma}$ avec $\gamma \in [0.05; 1]$. La figure 4.9 présente l'évolution de la fonction objectif en fonction de nombre d'itérations, pour différents γ , et afin de bien illustrer cette évolution, les données sont présentées en échelle logarithmique.

Le tableau suivant exprime les valeurs optimales de la fonction objectif h_{λ_1, λ_2} en utilisant l'algorithme de sous-gradient à l'aide de différents pas α'_k , cela est effectué pour plusieurs valeurs de γ .

Table. 4.3: Comparaison entre les pas α'_k avec des différents γ pour la minimisation de h_{λ_1, λ_2} .

γ	0,05	0,15	0,25	0,35	0,45	0,55	0,65	0,75	0,85	0,95	1
La valeur optimale $h_{\lambda_1, \lambda_2}^*$	12,4911	8,6091	7,3783	7,1253	7,0435	7,0135	7,0093	7,1549	8,0189	15,4161	19,7215

La figure 4.9 et le tableau 4.3 montrent clairement que le pas $\alpha'_k = \frac{1}{k^{0,65}} * \frac{1}{\max(1, \|v_{k-1}\|)}$ assure une convergence plus rapide que les autres pas, et présente les meilleurs résultats.

4.3.2.2 Choix de λ_1 et λ_2

Après avoir fixé le meilleur pas α'_k , nous passons maintenant à choisir les meilleurs paramètres λ_1 et λ_2 , la comparaison entre ces paramètres est effectuée via l'ensemble de gènes sélectionné en minimisant $h_{\lambda_1, \lambda_2}^*$, notre objectif alors est de trouver λ_1 et λ_2 qui maximisent le taux de classification sur l'ensemble test, et minimisent le nombre de gènes utilisé dans la classification.

Les expérimentations sont effectuées à l'aide du jeu de données "Prostate_tumor", et les taux d'exactitudes sont calculés en utilisant une validation croisée d'ordre 5. Pour faire cela, nous avons fixé dans un premier temps $\lambda_2 = 0.2$, et en variant λ_1 , le tableau 4.4 illustre les résultats trouvés suivant différentes valeurs de λ_1 ($\lambda_2 = 0.2$) :

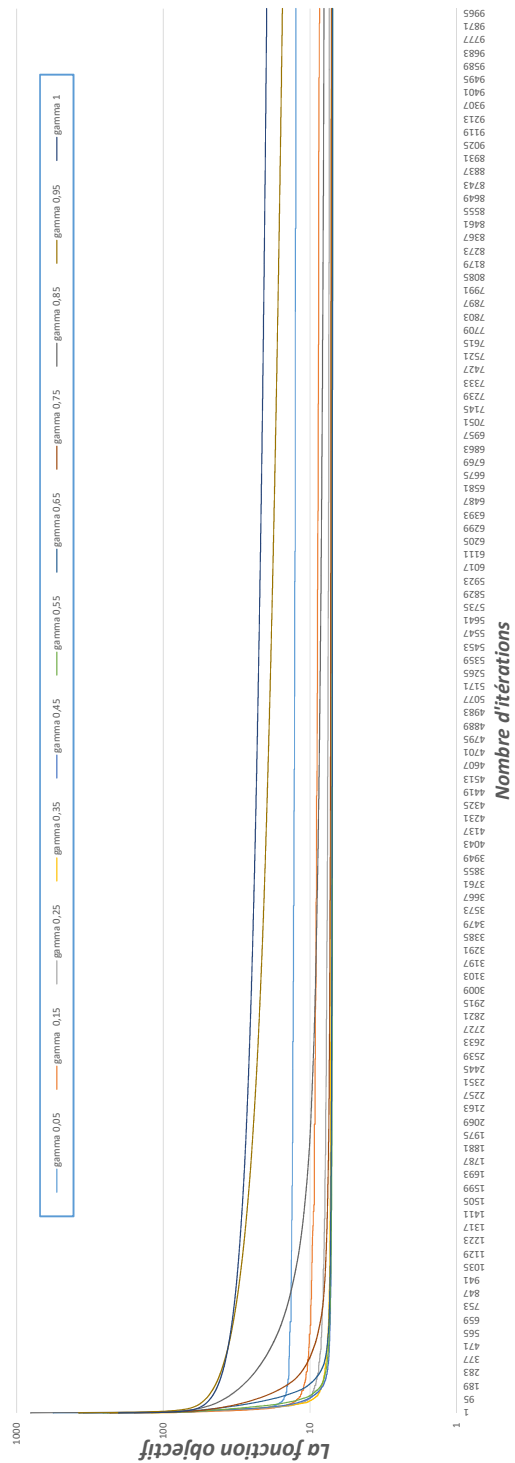


figure. 4.9: Comparaison entre les pas α'_k avec des différents γ pour la minimisation de h_{λ_1, λ_2} .

Table. 4.4: Comparaison entre les λ_1 pour le jeu de données "Prostate_Tumor" (Validation croisée d'ordre 5 avec $\lambda_2 = 0.2$).

λ_1	Taux d'exactitude (Test)	Nombre de gènes
0,2	96,14	55,6
0,4	96,1	36,8
0,6	96,14	32,6
0,8	96,14	29
1	96,14	24
1,2	97,1	28
1,4	96,14	26,2
1,6	97,1	25,4
1,8	96,14	25,6
2	96,14	22,6
2,2	96,14	24,6
2,4	96,14	23,8
2,6	95,14	22,2
2,8	96,14	20,4
3	95,14	20,8
3,2	94,19	20,4
3,4	94,19	20,4
3,6	94,19	20,4
3,8	95,14	19,6
4	94,19	19,2

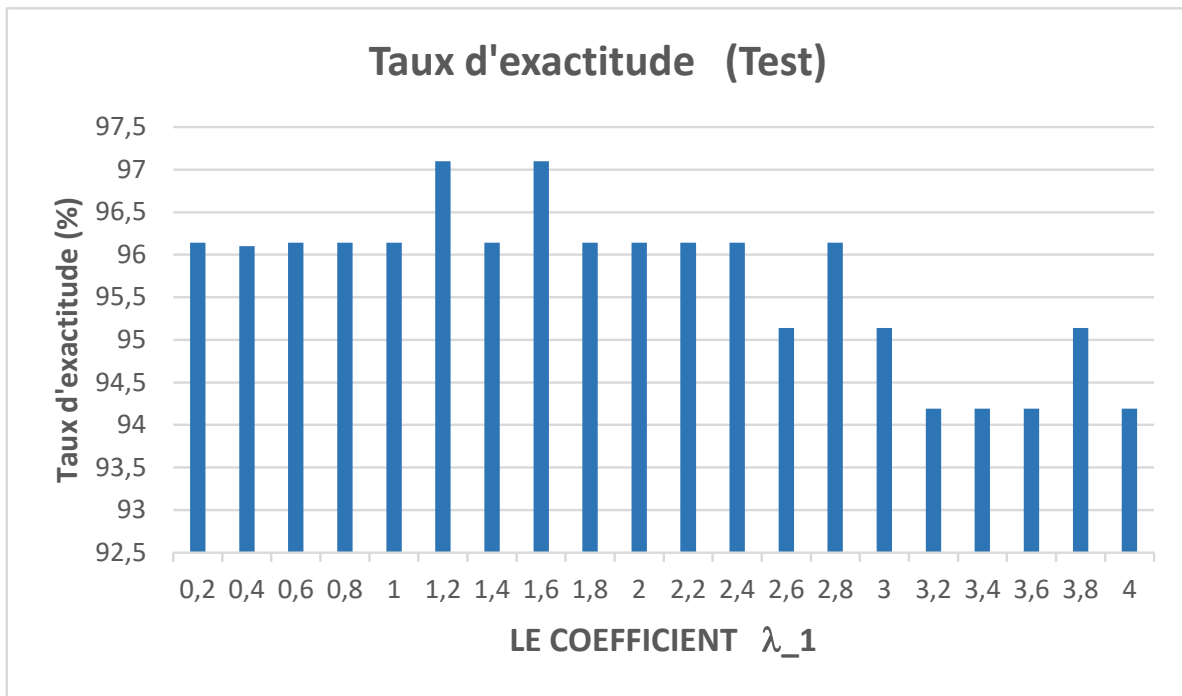


figure. 4.10: Comparaison des taux d'exactitude trouvés à l'aide de différentes valeurs de λ_1 pour "Prostate_Tumor" (Validation croisée d'ordre 5.)

Sur la base du tableau 4.4 et la figure 4.10, nous pouvons remarquer que $\lambda_1 = 1.6$ donne les meilleurs résultats, particulièrement, elle assure une exactitude élevée sur les données de test (97.1%), ceci en utilisant juste 25 gènes, d'autre part nous remarquons que lorsque on augmente le λ_1 le nombre de gènes sélectionnés se diminue, cela due à la pénalité du type L_1 .

Après avoir fixé le premier paramètre à $\lambda_1 = 1.6$, nous passons à choisir le meilleur λ_2 , le tableau suivant présente les résultats trouvés suivant différentes valeurs de λ_2 ($\lambda_1 = 1.6$) :

Table. 4.5: Comparaison entre les λ_2 pour le jeu de données "Prostate_Tumor" (Validation croisée d'ordre 5 avec $\lambda_1 = 1.6$).

λ_2	Taux d'exactitude (Test)	Nombre de gènes
0	94,19	21,6
0,025	96,14	20
0,05	96,14	21,2
0,075	96,14	22,4
0,1	97,1	22
0,125	97,1	23,8
0,15	97,1	24
0,175	96,14	25
0,2	97,1	25,4
0,225	95,14	25,6
0,25	97,1	26,6
0,275	95,14	28,2
0,3	95,19	27,4
0,325	95,14	28,4
0,35	95,14	28,6
0,375	95,19	27,2
0,4	96,14	27,2
0,425	96,14	28,4
0,45	96,14	25,8
0,475	96,14	28,6
0,5	95,19	28,2
0,525	95,1	29,6
0,55	97,1	27
0,575	95,14	30,8
0,6	96,19	29,6
0,625	95,19	28,2
0,65	95,19	32,2
0,675	96,1	31,2
0,7	94,14	26,8
0,725	96,14	29,8
0,75	95,19	29,2

Sur la base du tableau 4.5, nous pouvons remarquer que $\lambda_2 = 0$, présente la méthode LASSO classique, d'autre part $\lambda_2 = 0.1$ assure les meilleurs résultats, particulièrement, elle assure une exactitude élevée sur les données de test (97.1%), ceci en utilisant juste 22 gènes.

Sur la base des expérimentations effectuées nous fixons $\lambda_1 = 1.6$ et $\lambda_2 = 0.1$

4.3.3 Résultats et comparaisons

Premièrement, afin de limiter l'espace de recherche et d'accélérer la vitesse de convergence de notre approche proposée, le premier sous-ensemble de gènes a été sélectionné sur la base du filtre de corrélation de Pearson, puis l'algorithme FGLASSO-LR est utilisé pour sélectionner l'ensemble de gènes optimal, la qualité de ce dernier sous-ensemble est mesurée via le taux d'exactitude de l'ensemble test et le nombre de gènes utilisé.

Compte tenu de la nature non déterministe de notre approche, chaque ensemble de données est divisé aléatoirement en deux : un ensemble d'apprentissage avec 70 % de la taille d'origine et un ensemble de test avec 30 %, ce processus est répété en 50 exécutions indépendantes pour chaque ensemble de données afin d'obtenir un résultat plus fiable.

Les objectifs des expérimentations menées sur les jeux de données de (puces à ADN) sont les suivants : tester l'effet de la sélection de gènes sur l'amélioration de l'exactitude de la classification, valider la méthode proposée et vérifier son efficacité.

Table. 4.6: Les résultats obtenus sur 50 exécutions, sans et avec sélection via les pénalités LASSO et FG-LASSO

Jeu de données	Performances	Sans sélection		LASSO			FG-LASSO		
		1NN	SVM	1NN	SVM	LR	1NN	SVM	LR
Colon	Accuracy (%)	72,33	80,56	78,44	81	83,56	78,67	83,22	83,67
	Nombre de genes	2000		15,4			19,9		
DLBCL	Accuracy (%)	81,04	95,65	93,56	94,09	90,78	92,96	93,56	90,87
	Nombre de genes	5469		21,24			20,4		
Prostate_Tumor	Accuracy (%)	80	91,27	93,67	94	93,73	94,06	94,6	94,6
	Nombre de genes	10509		20,66			21,34		
Prostate	Accuracy (%)	80	91,27	93,67	94	93,8	94,07	94,67	95,27
	Nombre de genes	12600		20,64			20,94		

Accuracy : La moyenne d'exactitude de classification (70% apprentissage et 30 % de test).

Gènes : Le nombre de gènes utilisés dans la classification.

SVM : Le classificateur de machines à vecteurs de support utilisant un noyau linéaire.

LR : Le classifieur de régression logistique avec la régularisation LASSO et FG-LASSO respectivement.

1NN : le classifieur de 1-le plus proche voisin.

Le tableau 4.6 montre les résultats obtenus en utilisant notre méthode enveloppe avec les paramètres suivants :

- $\lambda_1 = 1.6$, $\lambda_2 = 0$, il s'agit donc d'une pénalité de type LASSO-LR
- $\lambda_1 = 1.6$, $\lambda_2 = 0.1$, il s'agit d'une pénalité généralisée de type Fused LASSO FGLASSO-LR.

Et puisque nos méthodes de sélection de gènes sont intégrées dans le processus de construction du classificateur de régression logistique (LR), nous avons pensé tester si cette sélection convient uniquement au classificateur LR, ou peut être utile pour les classificateurs SVM et KNN, les résultats de cette comparaison sont présentés dans le tableau 4.6. D'autre part nous comparons

l'effet des sélections de type LASSO et FGLASSO sur les performances des classifieurs SVM et 1NN. Nous analysons nos résultats selon deux axes :

- L'exactitude de la classification.
- Le nombre de gènes utilisés dans la classification.

Dans les figures (4.11,4.12 et 4.13), l'axe des abscisses exprime les jeux de données alors que l'axe des ordonnées donne la valeur moyenne du taux de classification sur 50 expériences pour les méthodes proposées.

A partir de ces dernières figures et la table 4.6, nous pouvons remarquer que l'approche FGLASSO donne de bonnes performances avec les différents classifieurs sur tous les jeux de données. En particulier, le classifieur de Regression logistique (LR) atteint les meilleurs taux d'exactitudes moyens sur les ensembles de données Colon, Prostate_Tumor et Prostate.

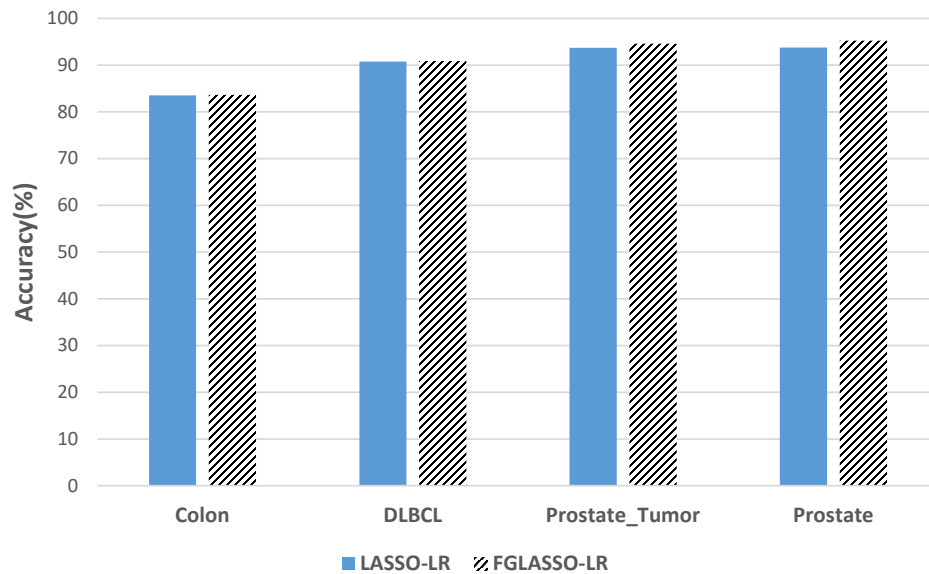


figure. 4.11: Comparaison des taux d'exactitudes de classification (LASSO-LR et FGLASSO-LR)).

À partir du tableau 4.6 et la figure 4.11, nous pouvons clairement observer que la nouvelle méthode FGLASSO-LR surpasse la méthode classique de LASSO-LR au niveau de taux d'exactitude pour tous les jeux de données, ce dépassement est bien remarquable pour les jeux de données (prostate_Tumor et Prostate) en utilisant un nombre de gènes qui varie entre 19 et 21, d'autres part nos méthodes FGLASSO-LR et LASSO-LR assure des exactitudes de classification élevées (supérieur ou égale à 83,57%) ceci en utilisant un faible nombre de gènes par rapport au nombre initial.

Afin de déterminer la fiabilité des gènes sélectionnés, d'autres types de classificateurs (1NN et SVM) ont été examinés selon la même procédure avec les ensembles finaux de gènes sélectionnés via les méthodes LASSO-LR et FGLASSO-LR. Les résultats présentés dans la figure 4.12 et le tableau 4.6 indiquent que nos méthodes de sélection sont efficaces pour le classifieur 1NN, en effet, cette sélection permet d'améliorer les taux d'exactitudes pour tous les jeux de données, plus précisément, pour le jeu de données "Colon", nous obtenons un taux d'exactitude supérieur ou égal

à 78.44% en utilisant moins de 20 gènes, ainsi cette sélection est efficace puisque, ça nous permet d'améliorer la classification (amélioration remarquable plus de 6% de différence) en utilisant moins de gènes, d'autre part l'amélioration la plus remarquable est pour le jeu de données de Prostate (plus de 14 % de différence) en utilisant moins de 21 gènes.

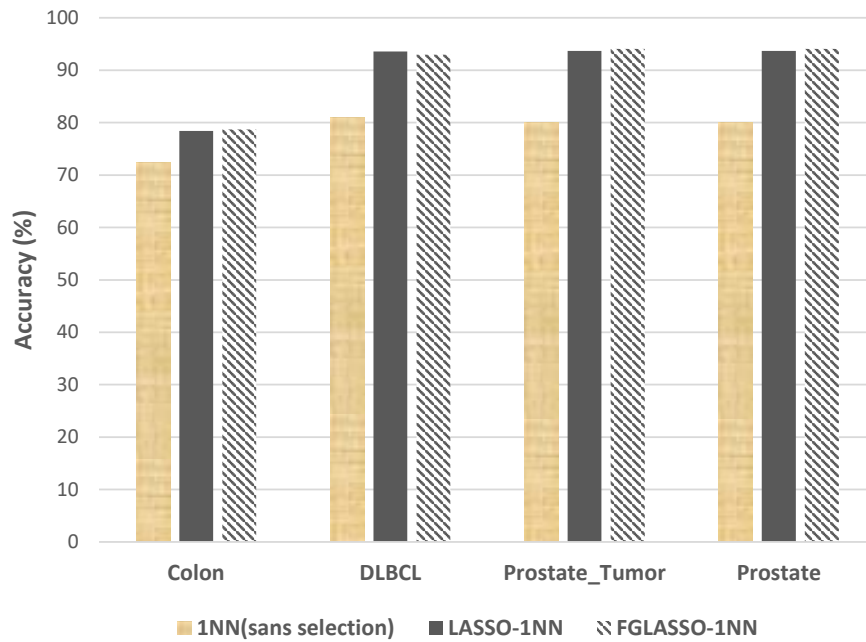


figure. 4.12: Comparaison des taux d'exactitudes de classification (1NN).

Les résultats présentés dans la figure 4.13 et le tableau 4.6 indiquent que nos méthodes de sélection sont parallèlement fiables pour le classifieur SVM, en effet, nous obtenons des améliorations remarquables au niveau des taux d'exactitudes pour les jeux de données (Colon, Prostate_Tumor et Prostate) en utilisant moins de 22 gènes.

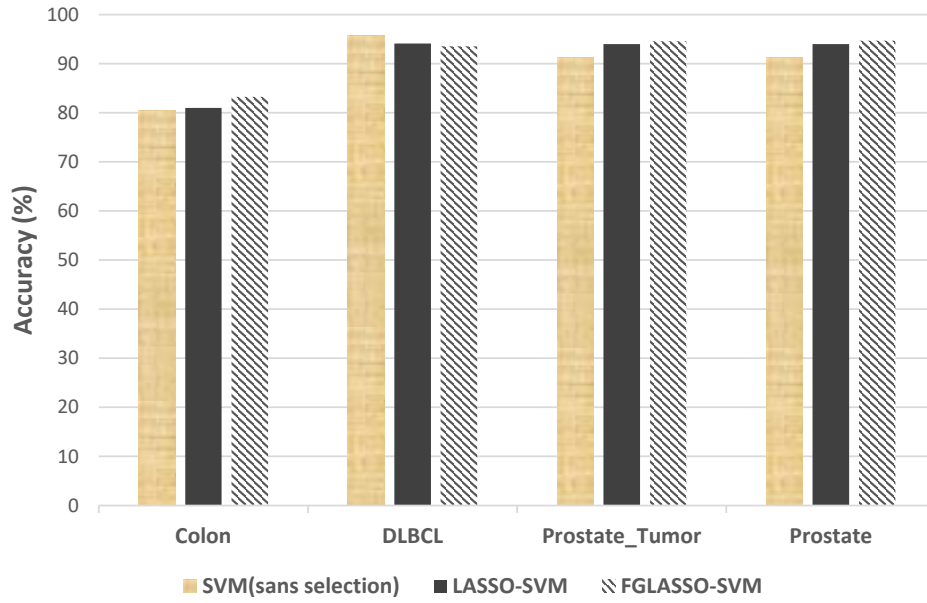


figure. 4.13: Comparaison des taux d'exactitudes de classification (SVM)).

Sur la base des expériences que nous avons menées, nous pouvons conclure que nos approches de sélection de gènes sont bien fondées. En effet, sur les quatre jeux de données utilisés, nos méthodes ont atteint une grande exactitude de classification. D'autre nos méthodes sont assez généralistes, en effet, elles sont appropriées à d'autres classifieurs tels-que : 1NN et SVM.

4.3.3.1 Comparaison avec d'autres approches

Dans cette sous-section, nous comparons notre méthode FGLASSO-LR avec des algorithmes référencés dans la littérature [11, 183, 65, 212, 13, 9], tous ces articles proposent des méthodes intégrées basées sur différents types de pénalités, et les appliquent pour le problème de sélection de gènes. Afin de donner un sens à cette comparaison, les expériences sont réalisées dans les mêmes conditions dans chaque algorithme. Concrètement, notre approche est exécutée 50 fois sur chaque jeu de données, et à chaque exécution, le jeu de données est divisé aléatoirement en deux (70 % d'apprentissage et 30 % de test) puis nous choisissons la moyenne des taux sur l'ensemble test ainsi que le nombre moyen de gènes utilisés dans la classification. Nous indiquons que dans le papier [13] on effectue 100 exécutions.

Le tableau 4.7 résume l'exactitude de la classification et le nombre de gènes sélectionnés (tirés des articles originaux) pour les différentes approches. Le symbole (-) indique que le résultat n'est pas rapporté dans le travail associé. Nous remarquons que les résultats obtenus par notre approche sont très encourageants et compétitifs par rapport aux travaux antérieurs.

Table. 4.7: Comparaison entre notre méthode (FGLASSO-LR) et des méthodes de l'état de l'art.

Jeu de donnees	Performences	FGLASSO-LR (notre méthode)	BLASSO [11]	LASSO [183]	SCAD [65]	ALASSO [212]	Nouvel ALASSO [13]	CBPLR [9]
Colon	Accuracy (%)	83,67 < 2 >	93	79,53	79,51	78,4	82,91	89,5
	Nombre de genes	19,9	11	14	14	14	12	10
DLBCL	Accuracy (%)	90,87 < 3 >	84	88,33	74,02	84,41	91,32	91,7
	Nombre de genes	20,4	17	24	24	24	22	17
Prostate	Accuracy (%)	95,27 < 1 >	-	91,14	60,13	82,11	93,53	93,6
	Nombre de genes	20,94	-	29	28	28	24	16

< · > : Indique le rang de notre méthode parmi les autres algorithmes au niveau de l'exactitude moyenne pour chaque jeu de données.

BLASSO= Bayesian Lasso quantile regression.

LASSO= La méthode LASSO classique.

SCAD= smoothly clipped absolute deviation.

ALASSO= La méthode de LASSO Adaptatif classique.

Nouvel ALASSO= Une nouvelle méthode de LASSO Adaptatif basée sur la distance de Mahalanobis pondérée.

CBPLR= Une nouvelle méthode de LASSO Adaptatif basée sur la corrélation entre les gènes.

Premièrement, pour l'ensemble de données (Colon), nous obtenons un taux d'exactitude de 83,67 % avec seulement 19 gènes. Nous constatons que la meilleure performance pour cet ensemble de données est atteinte par l'approche BLASSO (93 %), notre méthode alors se situe au rang 2 en surpassant les cinq autres approches. De même, pour le jeu de données (DLBCL), nous obtenons la troisième meilleure performance (90,87 %) après les approches « Nouvel ALASSO » et « CBPLR » avec une différence inférieure à 1 %. Finalement, pour le jeu de données « Prostate » nous obtenons la meilleure performance (exactitude de 95,27 %) suivi par la méthode « CBPLR » avec une exactitude de 93,6 %.

Les résultats de cette analyse comparative avec des méthodes intégrées antérieures de sélection de gènes dans le contexte de la classification des cancers nous ont permis de conclure que notre nouvelle approche de pénalisation de type fused LASSO généralisée est utile dans le problème de sélection de gènes.

4.4 Conclusion

Les problèmes de classification de grande dimension associés à l'analyse des données des puces à ADN constituent un domaine de recherche très important dans la classification du cancer. Dans ce chapitre, nous avons proposé et appliqué une nouvelle méthode intégrée FGLASSO-LR, pour estimer simultanément les coefficients des gènes et effectuer la sélection des gènes, puis améliorer les performances de classification à l'aide de données de puces à ADN de haute dimension, d'autre part, nous avons présenté et démontré la convergence de l'algorithme de sous gradient afin de résoudre le problème d'optimisation convexe non différentiel associé.

La méthode proposée a été évaluée en termes de nombre de gènes sélectionnés, et de taux d'exactitude de classification (sur l'ensemble test) en appliquant trois ensembles de données de

classification du cancer de haute dimension. Les résultats de la partie d'expérimentations ont systématiquement indiqué que FGLASSO-LR a la capacité de réduire considérablement la taille des gènes pertinents. De plus, on observe que le FGLASSO-LR a la supériorité en termes d'exactitude par rapport à la méthode classique LASSO.

Dans l'ensemble, les résultats ont démontré le fait que le FGLASSO est une méthode très compétitive pour analyser avec précision les données des puces à ADN de haute dimension pour la classification du cancer. Pour une utilisation pratique des résultats, le FGLASSO peut être appliqué directement à d'autres types de données de classification de grande dimension liées au domaine médical.

Dans les aspects concernant la recherche future, le présent travail peut être étendu pour couvrir les données de cancer de classification multiclassées de grande dimension. Une étude plus approfondie peut être menée en mettant davantage l'accent sur les données de puces à ADN à très haute dimension pour la classification du cancer.

Conclusion et perspectives

La sélection de gènes pertinents et la classification précise d'un échantillon donné sont parmi les sujets de recherches les plus actifs dans le domaine de diagnostic des maladies cancéreuses à l'aide des puces à ADN, ce qui donne aux chercheurs l'opportunité de développer des nouvelles approches de résolution. En effet, la phase de sélection de gènes vise à minimiser le bruit, le temps de calculs et à éviter le surajustement lors de la construction d'un tel modèle. D'autre part, un sous ensemble de gènes optimal améliore le modèle de classification dans le sens où nous pouvons avoir une classification fiable et précise d'un échantillon donné. Ainsi, la sélection des gènes est une étape essentielle dans l'analyse et la classification des ensembles de données des biopuces pour des raisons à la fois biologiques et informatiques. Dans cette thèse, nous nous sommes concentrés sur la recherche d'un sous-ensemble de gènes stable et/ou pertinent qui assure une bonne précision dans l'identification des tumeurs cancéreuses dans un ensemble de données d'expression génique donné. En effet, dans un premier temps, nous avons proposé plusieurs méthodes de sélection de gènes à ADN, basées sur les métaheuristiques BPSO, ACO et VNS :

- Trois approches filtres multivariées ont été proposées, MWIS, VNSMI et VNSCorr, pour la sélection d'un sous ensemble de gènes contenant l'information nécessaire pour le problème de classification de cancer, ces méthodes sont efficaces au niveau d'exactitude de classification et temps d'exécution.
- D'autres approches hybrides ont été proposée tel que : FBPSO, RBPSO et MWIS-ACOLS, ces dernières méthodes montrent des performances élevées, mais leur temps d'exécution est aussi important à cause de l'utilisation des classifieurs dans l'étape de sélection.

Par suite, dans le but d'améliorer les méthodes intégrées nous avons formulé le problème de sélection de gènes comme un problème de régression logistique (maximum de log-vraisemblance) associé à un régularisateur Fused LASSO généralisé, cette dernière pénalité n'est jamais utilisée dans ce type de problèmes. L'apparition de la valeur absolue dans la fonction objectif associée à la pénalité Fused LASSO rend l'utilisation des méthodes classiques d'optimisation difficile. Et afin de surmonter cet obstacle, nous avons proposé un nouvel algorithme de résolution connu sous le nom de sous-gradient, l'application de ce dernier algorithme nécessite la validation de quelques conditions de convergence, en effet, la fonction objectif doit être convexe, lipschitzienne et doit admettre un minimum global. Dans notre travail nous avons analysé théoriquement et numériquement cette méthode et vérifié que toutes les conditions sont remplies par notre fonction objectif.

La comparaison avec les travaux les plus importants dans le domaine de la sélection et de

la classification des données de puces à ADN montre que nos approches sont performantes et compétitives et produisent des sous-ensembles de petite taille avec une haute performance.

Ce domaine de recherche restera toujours actif tant qu'il sera motivé par les progrès des systèmes de collecte et de stockage des données d'une part, et par les exigences de l'oncologie d'autre part. La meilleure approche pour juger cette sélection de gènes est de collaborer avec des experts (biologistes) pour une bonne interprétation des résultats.

Bien que les résultats obtenus soient intéressants et encourageants, il existe de nombreux points et idées qui sont brièvement discutées dans ce présent rapport et qui fournissent des orientations pour les futures recherches :

- Lors de la proposition de la méthode MWIS, nous avons utilisé le coefficient de corrélation de Pearson pour mesurer la dépendance entre les gènes, cependant la mesure de l'information mutuelle peut être une bonne alternative de corrélation de Pearson.
- Faire appel aux méthodes de réseaux de neurones pour améliorer encore les taux de classification.
- Utiliser d'autres méta heuristiques pour enrichir l'état de l'art des méthodes hybrides de sélection de gènes.
- Dans le dernier chapitre nous avons proposé une méthode de sélection intégrée FGLASSO-LR, cette technique est valable juste pour les problèmes de classification binaire. On peut étudier la possibilité d'étendre cette approche pour gérer des situations multi-classes.
- Dans ce travail, toutes les méthodes formulées ont été adoptés uniquement pour les modèles d'apprentissage supervisé pour le diagnostic et la classification des maladies cancéreuses, qui nécessitent que tous les échantillons soient étiquetés. Cependant, il existe un certain nombre d'ensembles de données de biopuces avec des données non étiquetées. Ainsi, il sera important de considérer également les techniques d'apprentissage semi-supervisé.
- Tous les travaux présentés dans cette thèse portent sur les procédures de sélection de gènes dans les puces à ADN et leurs effets sur l'amélioration de l'apprentissage statistique. Appliquer ces travaux sur des données issues d'autres domaines peut également être une autre direction de recherche.

Ce domaine de recherche restera toujours actif tant qu'il est motivé par l'évolution des systèmes de collecte et de stockage des données d'une part, et par les exigences oncologiques d'autre part. La meilleure approche pour juger cette sélection est de collaborer avec des experts (biologistes) pour une bonne interprétation des résultats.

Appendices

Théorie des graphes

Dans cette thèse nous manipulerons des objets mathématiques : les graphes. Nous allons donc donner un bref aperçu de la terminologie utilisée dans cette thèse.

A.1 Notations de base

Définition A.1.1. Graphes non orientés : un graphe non orienté $G = (V, E)$ est constitué de deux ensembles finis : V , l'ensemble des sommets de G (ou nœud), qui est un ensemble non vide d'éléments appelés sommets et $E \subseteq V \times V$ est un ensemble de couples non ordonnés de sommets $\{u, v\} \in V^2$ appelé ensemble d'arêtes. La cardinalité de l'ensemble de sommets V est appelée l'ordre de G , communément désigné par $|V|$. La cardinalité de l'ensemble des arêtes E est la taille de G , notée $|E|$. Un graphe d'ordre n et de taille m est souvent appelé un (n, m) -graphe. Deux sommets distincts u et v sont adjacents (ou voisins) s'il existe une arête $\{u, v\}$ qui les relie. On dit qu'une arête $\{u, v\}$ est incidente aux sommets u et v . Deux arêtes sont adjacentes si elles sont incidentes à un même sommet. Un graphe est simple s'il y a au plus une arête entre deux sommets. Dans cette thèse, les graphes considérés seront des graphes non orientés simples finis. Dans la suite nous confondons $\{u, v\}$ et (u, v) .

Définition A.1.2. Degré et voisinage : Soit v un sommet d'un graphe G . Un voisin de v est un sommet u tel que $(u, v) \in E$.

Le voisinage d'un sommet v dans G , noté $N_G(v)$ ou $N(v)$ s'il n'y a pas d'ambiguïté sur le graphe considéré, est l'ensemble des voisins de v dans G :

$$N_G(v) = \{u \in V, (u, v) \in E\}$$

Le voisinage fermé d'un sommet v dans G , noté $N_G^+(v)$, est le voisinage de v plus le sommet v lui-même : $N_G^+(v) = N_G(v) \cup \{v\}$.

Définition A.1.3. Sous-graphe Étant donné un graphe $G = (V, E)$, un sous-graphe de G est tout graphe $G' = (V', E')$ tel que $V' \subseteq V$ et $E' \subseteq E$.

Étant donné un sous-ensemble V' des nœuds de G , le sous-graphe de G induit par V' est défini

comme $G_{V'} = (V', E(V'))$, où $E(V') = \{(u, v) \in E : u, v \in V'\}$, c'est-à-dire que $G_{V'}$ contient toutes les arêtes de G telles que les deux extrémités de l'arête sont dans V' .

Définition A.1.4. Graphe pondéré *Un graphe pondéré est un graphe dans lequel les arêtes, ou les sommets, ou les deux, sont étiquetés avec un poids, dans le cas où les sommets sont étiquetés par des poids on dit que le graphe est pondéré par les sommets (vertex-weighted graph).*

Définition A.1.5. Ensembles indépendants : *un ensemble indépendant de G est un sous-ensemble de sommets $U \subseteq V$, tel que deux sommets dans U ne sont pas adjacents. Un ensemble indépendant est dit maximal si aucun ensemble indépendant ne le contient strictement. Un ensemble indépendant de cardinalité maximale est appelé un ensemble indépendant maximum.*

Définition A.1.6. Ensemble indépendant de poids maximum : *(The maximum weight independent set « MWIS ») est un ensemble indépendant de G avec un poids total maximum. Le problème de l'ensemble indépendant maximal est le cas particulier dans lequel tous les poids sont un.*

A.2 Opérations sur les graphes

Dans les définitions suivantes, nous détaillons quelques opérations sur les graphes bien connues.

Définition A.2.1. Complément d'un graphe : *si G est un graphe simple avec un ensemble de sommets $V(G)$, son complément $\bar{G}$ est le graphe simple avec un ensemble de sommets $V(G)$ dans lequel deux sommets sont adjacents si et seulement s'ils ne sont pas adjacents dans G .*

Définition A.2.2. Supprimer un sommet : *si v est un sommet du graphe $G = (V, E)$, alors $G - v$ est le sous-graphe de G induit par l'ensemble des sommets $V \setminus \{v\}$.*

Supprimer une arête : *de manière similaire à l'opération précédente, si e est une arête du graphe $G = (V, E)$, alors $G - e$ est le graphe (V, E') , où E' est obtenu en supprimant e de E . Notez que les extrémités de e ne sont pas supprimés de G .*

Union de graphe : *l'union $G = G_1 \cup G_2$ des graphes G_1 et G_2 avec les ensembles de sommets disjoints V_1 et V_2 et les ensembles des arêtes E_1 et E_2 est le graphe avec $V = V_1 \cup V_2$ et $E = E_1 \cup E_2$. Cette opération est parfois également connue explicitement sous le nom d'union disjointe.*

Notions sur la complexité

Dans cette annexe, nous rapportons quelques définitions et notions sur la complexité des problèmes d'optimisation combinatoire qui ont été dans cette thèse. Généralement, le temps d'exécution est le facteur majeur qui détermine l'efficacité d'un algorithme, alors la complexité en temps d'un algorithme est le nombre d'instructions nécessaires (affectation, comparaison, opérations algébriques, lecture et écriture, etc.) que comprend cet algorithme pour une résolution d'un problème quelconque.

Définition B.0.1. Une fonction $f(n)$ est $O(g(n))$ ($f(n)$ est de complexité $g(n)$), s'il existe un réel $c > 0$ et un entier positif n_0 tel que pour tout $n \geq n_0$ on a $|f(n)| \leq c.g(n)$

Définition B.0.2. Un algorithme en temps polynomial est un algorithme dont le temps de la complexité est en $O(p(n))$, où p est une fonction polynomiale et n est la taille de l'instance (ou sa longueur d'entrée).

Si k est le plus grand exposant de ce polynôme en n , le problème correspondant est dit être résoluble en $O(n^k)$ et appartient à la classe P .

B.1 Ordre de grandeur et temps d'exécution

La détermination de la complexité algorithmique ne permet pas d'en déduire le temps d'exécution mais seulement de comparer entre deux algorithmes résolvant le même problème. Cependant, il importe de prendre conscience des différences échelles considérables qui existent entre les ordres de grandeurs usuels que l'on rencontre. En s'appuyant sur une base de 10^9 opérations par seconde, le tableau de la figure B.1 est à cet égard significatif. Il indique en fonction de la taille n de l'entrée ($10^2, 10^3, \dots$) et du nombre d'opérations requis par un algorithme ($\log(n), n, \dots$) le temps d'exécution de ce dernier.

	$\log n$	n	$n \log n$	n^2	n^3	2^n
10^2	7 ns	100 ns	0,7 μ s	10 μ s	1 ms	$4 \cdot 10^{13}$ années
10^3	10 ns	1 μ s	10 μ s	1 ms	1 s	10^{292} années
10^4	13 ns	10 μ s	133 μ s	100 ms	17 s	
10^5	17 ns	100 μ s	2 ms	10 s	11,6 jours	
10^6	20 ns	1 ms	20 ms	17 mn	32 années	

figure. B.1: Temps nécessaire à l'exécution d'un algorithme en fonction de sa complexité.

La lecture de ce tableau est édifiante : on comprend que les algorithmes ayant une complexité supérieure à une complexité quadratique soient en général considérés comme inutilisables en pratique (sauf pour de petites voire très petites valeurs de n).

Table. B.1: Qualifications usuelles des complexités.

$O(\log n)$	logarithmique
$O(n)$	linéaire
$O(n \log n)$	quasi-linéaire
$O(n^2)$	quadratique
$O(n^k) \quad (k \geq 2)$	polynomiale
$O(k^n) \quad (k > 1)$	exponentielle

B.2 Différents types de complexité

Certains algorithmes ont un temps d'exécution qui dépend non seulement de la taille des données mais de ces données elles-mêmes. Dans ce cas on distingue plusieurs types de complexités :

- La complexité dans le pire des cas : c'est un majorant du temps d'exécution possible pour toutes les entrées possibles d'une même taille. On l'exprime en général à l'aide de la notation O .
- La complexité dans le meilleur des cas : c'est un minorant du temps d'exécution possible pour toutes les entrées possibles d'une même taille. On l'exprime en général à l'aide de la notation Ω . Cependant cette notion n'est que rarement utilisée car souvent peu pertinente au regard des complexités dans le pire des cas et en moyenne.
- La complexité en moyenne : c'est une évaluation du temps d'exécution moyen portant sur toutes les entrées possible d'une même taille supposées équiprobables.

La plupart du temps on se contentera d'analyser la complexité dans le pire des cas.

Définition B.2.1. La classe NP contient les problèmes de décision qui peuvent être décidés sur une machine non déterministe en temps polynomial. C'est la classe des problèmes qui admettent un algorithme polynomial capable de tester la validité d'une solution du problème. Intuitivement, les problèmes de cette classe sont les problèmes qui peuvent être résolus en énumérant l'ensemble de solutions possibles et en les testant à l'aide d'un algorithme polynomial. On cite aussi que l'égalité

de P et NP est un problème non encore résolu. Le problème $P=NP$ est considéré comme l'un des problèmes les plus importants de l'informatique théorique, et même des mathématiques en général [42].

Définition B.2.2. On dit qu'un problème de recherche P_1 se réduit polynomialement à un problème de recherche P_2 par la réduction de Turing s'il existe un algorithme A_1 pour résoudre P_1 utilisant comme sous programme un algorithme A_2 résolvant P_2 , de telle sorte que la complexité de A_1 est polynomiale, quand on évalue chaque appel de A_2 par une constante.

Définition B.2.3. La classe NP-complet :

parmi l'ensemble des problèmes appartenant à NP , il en existe un sous ensemble qui contient les problèmes les plus difficiles : on les appelle les problèmes NP-complets. Un problème NP-complet possède la propriété que tout problème dans NP peut être transformé (réduit) en celui-ci en temps polynomial. C'est à dire qu'un problème est NP-complet quand tous les problèmes appartenant à NP lui sont réductibles. Si on trouve un algorithme polynomial pour un problème NP-complet, on trouve alors automatiquement une résolution polynomiale de tous les problèmes de la classe NP .

Définition B.2.4. La classe NP-difficile : un problème est NP-difficile s'il est plus difficile qu'un problème NP-complet, c'est à dire s'il existe un problème NP-complet se réduisant à ce problème par une réduction de Turing.

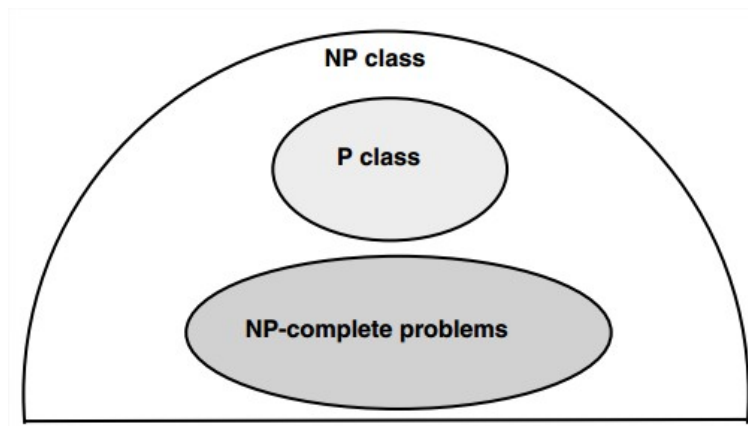


figure. B.2: Classes de complexité.

Bibliographie

- [1] Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). The elements of statistical learning : data mining, inference, and prediction (Vol. 2, pp. 1-758). New York : springer.
- [2] Ouimet, M. (2005). Réduction de dimensionnalité non linéaire et vorace.
- [3] Agarwal, S., Rajesh, R., & Ranjan, P. (2017). FRBPSO : a Fuzzy rule based binary PSO for feature selection. Proceedings of the National Academy of Sciences, India Section A : Physical Sciences, 87(2), 221-233.
- [4] Aha, D. W., & Bankert, R. L. (1996). A comparative evaluation of sequential feature selection algorithms. In Learning from data (pp. 199-206). Springer, New York, NY.
- [5] Ahmed, Z., Zeeshan, S., Mendhe, D., & Dong, X. (2020). Human gene and disease associations for clinical-genomics and precision medicine research. Clinical and translational medicine, 10(1), 297-318.
- [6] Alba, E., Garcia-Nieto, J., Jourdan, L., & Talbi, E. G. (2007, September). Gene selection in cancer classification using PSO/SVM and GA/SVM hybrid algorithms. In 2007 IEEE congress on evolutionary computation (pp. 284-290). IEEE.
- [7] Alber, Y. I., Iusem, A. N., & Solodov, M. V. (1998). On the projected subgradient method for nonsmooth convex optimization in a Hilbert space. Mathematical Programming, 81(1), 23-35.
- [8] Albert, A., & Anderson, J. A. (1984). On the existence of maximum likelihood estimates in logistic regression models. Biometrika, 71(1), 1-10.
- [9] Algamal, Z. Y., & Lee, M. H. (2015). Penalized logistic regression with the adaptive LASSO for gene selection in high-dimensional cancer classification. Expert Systems with Applications, 42(23), 9326-9332.
- [10] Algamal, Z. (2017). Classification of gene expression autism data based on adaptive penalized logistic regression. Electronic Journal of Applied Statistical Analysis, 10(2), 561-571.
- [11] Algamal, Z. Y., Alhamzawi, R., & Ali, H. T. M. (2018). Gene selection for microarray gene expression classification using Bayesian Lasso quantile regression. Computers in biology and medicine, 97, 145-152.
- [12] Algamal, Z. Y., & Lee, M. H. (2019). A two-stage sparse logistic regression for optimal gene selection in high-dimensional microarray data classification. Advances in data analysis and classification, 13(3), 753-771.

- [13] Alharthi, A. M., Lee, M. H., & Algamal, Z. Y. (2020). Weighted L1-norm logistic regression for gene selection of microarray gene expression classification. *International Journal on Advanced Science Engineering Information Technology*, 4, 2088-5334.
- [14] Alharthi, A. M., Lee, M. H., & Algamal, Z. Y. (2021). Gene selection and classification of microarray gene expression data based on a new adaptive L1-norm elastic net penalty. *Informatics in Medicine Unlocked*, 100622.
- [15] Alizadeh, A. A., Eisen, M. B., Davis, R. E., Ma, C., Lossos, I. S., Rosenwald, A.,...& Staudt, L. M. (2000). Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769), 503-511.
- [16] Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., & Levine, A. J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 96(12), 6745-6750.
- [17] Amaldi, E., & Kann, V. (1998). On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209(1-2), 237-260.
- [18] Ang, J. C., Mirzal, A., Haron, H., & Hamed, H. N. A. (2015). Supervised, unsupervised, and semi-supervised feature selection : a review on gene selection. *IEEE/ACM transactions on computational biology and bioinformatics*, 13(5), 971-989.
- [19] Antoniadis, A., Lambert-Lacroix, S., & Leblanc, F. (2003). Effective dimension reduction methods for tumor classification using gene expression data. *Bioinformatics*, 19(5), 563-570.
- [20] Apolloni, J., Leguizamón, G., & Alba, E. (2016). Two hybrid wrapper-filter feature selection algorithms applied to high-dimensional microarray experiments. *Applied Soft Computing*, 38, 922-932.
- [21] Bach, F., Jenatton, R., Mairal, J., & Obozinski, G. (2012). Structured sparsity through convex optimization. *Statistical Science*, 27(4), 450-468.
- [22] Bellman, R., & Kalaba, R. (1959). On adaptive control processes. *IRE Transactions on Automatic Control*, 4(2), 1-9.
- [23] Bielza, C., Robles, V., & Larrañaga, P. (2011). Regularized logistic regression without a penalty term : An application to cancer classification with microarray data. *Expert Systems with Applications*, 38(5), 5110-5118.
- [24] Bir-Jmel, A., Douiri, S. M., & Elbernoussi, S. (2019). Gene selection via BPSO and Backward generation for cancer classification. *RAIRO-Operations Research*, 53(1), 269-288.
- [25] Bir-Jmel, A., Douiri, S. M., & Elbernoussi, S. (2019). Gene selection via a new hybrid ant colony optimization algorithm for cancer classification in high-dimensional data. *Computational and mathematical methods in medicine*, 2019.
- [26] Bishop, C. M. (1996). *Neural networks : a pattern recognition perspective*.
- [27] Bittner, M., Meltzer, P., Chen, Y., Jiang, Y., Seftor, E., Hendrix, M., ... & Trent, J. (2000). Molecular classification of cutaneous malignant melanoma by gene expression profiling. *Nature*, 406(6795), 536-540.

- [28] Blum, A. L., & Rivest, R. L. (1992). Training a 3-node neural network is NP-complete. *Neural Networks*, 5(1), 117-127.
- [29] Blum, A. L., & Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1-2), 245-271.
- [30] Bognar, K. Aspects théoriques de la classification à base de treillis. Université Debrecen : Institut de mathématiques et informatique, 2003.
- [31] Bonev, B., Escolano, F., & Cazorla, M. (2008). Feature selection, mutual information, and the classification of high-dimensional patterns. *Pattern Analysis and Applications*, 11(3), 309-319.
- [32] Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992, July). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory* (pp. 144-152).
- [33] Boughaci, D., & Alkhawaldeh, A. A. S. (2018). Three local search-based methods for feature selection in credit scoring. *Vietnam Journal of Computer Science*, 5(2), 107-121.
- [34] Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). Classification and regression trees. *Wadsworth Int. Group*, 37(15), 237-251.
- [35] Bullnheimer, B., Hartl, R. F., & Strauss, C. (1999). An improved ant System algorithm for the vehicle Routing Problem. *Annals of operations research*, 89, 319-328.
- [36] Cao, J., Zhang, L., Wang, B., Li, F., & Yang, J. (2015). A fast gene selection method for multi-cancer classification using multiple support vector data description. *Journal of biomedical informatics*, 53, 381-389.
- [37] Chiang, Y. M., Chiang, H. M., & Lin, S. Y. (2008, July). The application of ant colony optimization for gene selection in microarray-based cancer classification. In *2008 International Conference on Machine Learning and Cybernetics* (Vol. 7, pp. 4001-4006). IEEE.
- [38] Chuang, L. Y., Chang, H. W., Tu, C. J., & Yang, C. H. (2008). Improved binary PSO for feature selection using gene expression data. *Computational Biology and Chemistry*, 32(1), 29-38.
- [39] Chuang, L. Y., Yang, C. H., & Yang, C. H. (2009). Tabu search and binary particle swarm optimization for feature selection using microarray data. *Journal of computational biology*, 16(12), 1689-1703.
- [40] Colorni, A., Dorigo, M., & Maniezzo, V. (1991, December). Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life* (Vol. 142, pp. 134-142).
- [41] Colorni, A., Dorigo, M., & Maniezzo, V. (1991, December). Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life* (Vol. 142, pp. 134-142).
- [42] Cook, S. (2000). The P versus NP problem. *Clay Mathematics Institute*, 2.
- [43] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273-297.
- [44] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory*, 13, 21-27.

- [45] Cui, L., Bai, L., Wang, Y., Philip, S. Y., & Hancock, E. R. (2021). Fused lasso for feature selection using structural information. *Pattern Recognition*, 108058.
- [46] Das, A. K., Goswami, S., Chakrabarti, A., & Chakraborty, B. (2017). A new hybrid feature selection approach using feature association map for supervised and unsupervised classification. *Expert Systems with Applications*, 88, 81-94.
- [47] Dash, M., & Liu, H. (1997). Feature selection for classification. *Intelligent data analysis*, 1(1-4), 131-156.
- [48] Dashtban, M., Balafar, M., & Suravajhala, P. (2018). Gene selection for tumor classification using a novel bio-inspired multi-objective approach. *Genomics*, 110(1), 10-17.
- [49] Dernoncourt, D. (2014). Stabilité de la sélection de variables sur des données haute dimension : une application à l'expression génique (Doctoral dissertation, Paris 6).
- [50] Devijver, P. A., & Kittler, J. (1982). *Pattern recognition : A statistical approach*. Prentice hall.
- [51] Ding, C., & Peng, H. (2005). Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology*, 3(02), 185-205.
- [52] Doak, J. (1992). An evaluation of feature selection methods and their application to computer security. *Techninal Report CSE-92-18*.
- [53] Dorigo, M., Maniezzo, V., & Coloni, A. (1996). Ant system : optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1), 29-41.
- [54] Dorigo, M., Maniezzo, V., & Coloni, A. (1996). Ant system : optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1), 29-41.
- [55] Dorigo, M., & Gambardella, L. M. (1997). Ant colony system : a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on evolutionary computation*, 1(1), 53-66.
- [56] Dorigo, M., & Di Caro, G. (1999, July). Ant colony optimization : a new meta-heuristic. In *Proceedings of the 1999 congress on evolutionary computation-CEC99* (Cat. No. 99TH8406) (Vol. 2, pp. 1470-1477). IEEE.
- [57] Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE computational intelligence magazine*, 1(4), 28-39.
- [58] Douiri, M., Elbernoussi, S., & Lakhab, H. (2009). Cours des méthodes de résolution exactes heuristiques et métaheuristiques. Université Mohamed V, Faculté des sciences de Rabat, 5-87.
- [59] Douiri, S. M., & Elbernoussi, S. (2012). A new ant colony optimization algorithm for the lower bound of sum coloring problem. *Journal of Mathematical Modelling and Algorithms*, 11(2), 181-192.
- [60] Dudoit, S., Fridlyand, J., & Speed, T. P. (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American statistical association*, 97(457), 77-87.

- [61] El Dor, A. (2012). Perfectionnement des algorithmes d'optimisation par essaim particulaire : applications en segmentation d'images et en électronique (Doctoral dissertation, Université Paris-Est).
- [62] El Moudden, I. Modélisation, Analyse et Classification de données à grande Echelle Basées sur les Méthodes de Réduction de Dimensionnalité. Thèse de doctorat en Mathématiques. Faculté des sciences de Rabat, Université Mohammed V, 2017.
- [63] El Moudden, I., Ouzir, M., Benyacoub, B., & ElBernoussi, S. (2016). Mining human activity using dimensionality reduction and pattern recognition. *Contemporary Engineering Sciences*, 9(21), 1031-1041.
- [64] Estévez, P. A., Tesmer, M., Perez, C. A., & Zurada, J. M. (2009). Normalized mutual information feature selection. *IEEE Transactions on neural networks*, 20(2), 189-201.
- [65] Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456), 1348-1360.
- [66] Fan, J., Fan, Y., & Barut, E. (2014). Adaptive robust variable selection. *Annals of statistics*, 42(1), 324.
- [67] Fisher, R. A., & Yates, F. (1938). Statistical tables for biological, agricultural and medical research. Statistical tables for biological, agricultural and medical research.
- [68] Gambardella, L. M., Taillard, É. D., & Dorigo, M. (1999). Ant colonies for the quadratic assignment problem. *Journal of the operational research society*, 50(2), 167-176.
- [69] Garcia-Torres, M., Garcia-López, F., Melián-Batista, B., Moreno-Pérez, J. A., & Moreno-Vega, J. M. (2004). Solving feature subset selection problem by a hybrid metaheuristic. *Hybrid Metaheuristics*, 59-68.
- [70] García-Torres, M., Gómez-Vela, F., Melián-Batista, B., & Moreno-Vega, J. M. (2016). High-dimensional feature selection via feature grouping : A Variable Neighborhood Search approach. *Information Sciences*, 326, 102-118.
- [71] Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability* (Vol. 174). San Francisco : freeman.
- [72] Gene Expression Model Selector www.gems-system.org (consulté en janvier 2018)
- [73] Gerber, M. U., Hertz, A., & Schindl, D. (2003). P5-free augmenting graphs and the maximum stable set problem. *Discrete Applied Mathematics*, 132(1-3), 109-119.
- [74] Gheyas, I. A., & Smith, L. S. (2010). Feature subset selection in large dimensionality domains. *Pattern recognition*, 43(1), 5-13.
- [75] Ghosh, A. K., Chaudhuri, P., & Murthy, C. A. (2005). On visualization and aggregation of nearest neighbor classifiers. *IEEE transactions on pattern analysis and machine intelligence*, 27(10), 1592-1602.
- [76] Gillies, C. E., Siadat, M. R., Patel, N. V., & Wilson, G. D. (2013). A simulation to analyze feature selection methods utilizing gene ontology for gene expression classification. *Journal of biomedical informatics*, 46(6), 1044-1059.

- [77] Glover, F. W., & Kochenberger, G. A. (Eds.). (2006). Handbook of metaheuristics (Vol. 57). Springer Science & Business Media.
- [78] Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., ... & Lander, E. S. (1999). Molecular classification of cancer : class discovery and class prediction by gene expression monitoring. *science*, 286(5439), 531-537.
- [79] Gu, Q., Li, Z., & Han, J. (2012). Generalized fisher score for feature selection. *arXiv preprint arXiv :1202.3725*.
- [80] Guermeur Y : SVM multiclass, théorie et applications. Habilitation à diriger des recherches, UHP 2007. [In French]
- [81] Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1), 389-422.
- [82] Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar), 1157-1182.
- [83] Habib, M., Aljarah, I., Faris, H., & Mirjalili, S. (2020). Multi-objective particle swarm optimization : theory, literature review, and application in feature selection for medical diagnosis. *Evolutionary Machine Learning Techniques*, 175-201.
- [84] Hansen, P., & Mladenović, N. (2001). Variable neighborhood search : Principles and applications. *European journal of operational research*, 130(3), 449-467.
- [85] Hawkes, N. (2019). Cancer survival data emphasise importance of early diagnosis. « <https://doi.org/10.1136/bmj.l408> »
- [86] Henriot, L. Système d'évaluation et de classification multicritères pour l'aide à la décision, construction de modèles et procédures d'affectation. Thèse de doctorat en science. Université Paris Dauphine, 2000.
- [87] Hernandez, J. C. H. (2008). Algorithmes métaheuristiques hybrides pour la sélection de gènes et la classification de données de biopuces (Doctoral dissertation, Université d'Angers).
- [88] Hoeffling, H. (2010). A path algorithm for the fused lasso signal approximator. *Journal of Computational and Graphical Statistics*, 19(4), 984-1006.
- [89] Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression : Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), 55-67.
- [90] Höfling, H., Binder, H., & Schumacher, M. (2010). A coordinate-wise optimization algorithm for the Fused Lasso. *arXiv preprint arXiv :1011.6409*.
- [91] Holland, J. (1975). Adaptation in artificial and natural systems. Ann Arbor : The University of Michigan Press, 232.
- [92] Huang, J., Ma, S., & Zhang, C. H. (2008). The iterated lasso for high-dimensional logistic regression. The University of Iowa, Department of Statistics and Actuarial Sciences, 1-20.
- [93] Ijaz, M., Asghar, Z., & Gul, A. (2019). Ensemble of penalized logistic models for classification of high-dimensional data. *Communications in Statistics-Simulation and Computation*, 1-17.
- [94] Jaccard, P. (1912). The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2), 37-50.

- [95] Jafari, P., & Azuaje, F. (2006). An assessment of recently published gene expression data analyses : reporting experimental design and statistical factors. *BMC Medical Informatics and Decision Making*, 6(1), 1-8.
- [96] Jang, W., Lim, J., Lazar, N. A., Loh, J. M., & Yu, D. (2015). Some properties of generalized fused lasso and its applications to high dimensional data. *Journal of the Korean Statistical Society*, 44(3), 352-365.
- [97] John, G. H., Kohavi, R., & Pfleger, K. (1994). Irrelevant features and the subset selection problem. In *Machine learning proceedings 1994* (pp. 121-129). Morgan Kaufmann.
- [98] Johnson, N. A. (2013). A dynamic programming algorithm for the fused lasso and l 0-segmentation. *Journal of Computational and Graphical Statistics*, 22(2), 246-260.
- [99] Kabir, M. M., Islam, M. M., & Murase, K. (2010). A new wrapper feature selection approach using neural network. *Neurocomputing*, 73(16-18), 3273-3283.
- [100] Kastrin, A., & Peterlin, B. (2010). Rasch-based high-dimensionality data reduction and class prediction with applications to microarray gene expression data. *Expert Systems with Applications*, 37(7), 5178-5185.
- [101] Kennedy, J., & Eberhart, R. (1995, November). Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks* (Vol. 4, pp. 1942-1948). IEEE.
- [102] Kennedy, J., & Eberhart, R. C. (1997, October). A discrete binary version of the particle swarm algorithm. In *1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation* (Vol. 5, pp. 4104-4108). IEEE.
- [103] Kira, K., & Rendell, L. A. (1992). A practical approach to feature selection. In *Machine learning proceedings 1992* (pp. 249-256). Morgan Kaufmann.
- [104] Klavžar, S., & Gutman, I. (1997). Wiener number of vertex-weighted graphs and a chemical application. *Discrete applied mathematics*, 80(1), 73-81.
- [105] Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2), 273-324.
- [106] Koller, D., & Sahami, M. (1996). Toward optimal feature selection. *Stanford InfoLab*.
- [107] Kononenko, I. (1994, April). Estimating attributes : Analysis and extensions of RELIEF. In *European conference on machine learning* (pp. 171-182). Springer, Berlin, Heidelberg.
- [108] Kuncheva, L. I. (2007, February). A stability index for feature selection. In *Artificial intelligence and applications* (pp. 421-427).
- [109] Lai, C. M., Yeh, W. C., & Chang, C. Y. (2016). Gene selection using information gain and improved simplified swarm optimization. *Neurocomputing*, 218, 331-338.
- [110] Lai, C. M., & Huang, H. P. (2021). A gene selection algorithm using simplified swarm optimization with multi-filter ensemble technique. *Applied Soft Computing*, 100, 106994.
- [111] Lazar, C., Taminau, J., Meganck, S., Steenhoff, D., Coletta, A., Molter, C., ... & Nowe, A. (2012). A survey on filter techniques for feature selection in gene expression microarray analysis. *IEEE/ACM transactions on computational biology and bioinformatics*, 9(4), 1106-1119.

- [112] Lee, C. P., & Leu, Y. (2011). A novel hybrid feature selection method for microarray data analysis. *Applied Soft Computing*, 11(1), 208-213.
- [113] Levine, J., & Ducatelle, F. (2004). Ant colony optimization and local search for bin packing and cutting stock problems. *Journal of the Operational Research society*, 55(7), 705-716.
- [114] Lewis, D. D. (1992). Feature selection and feature extraction for text categorization. In *Speech and Natural Language : Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.
- [115] Li, S., Wu, X., & Hu, X. (2008). Gene selection using genetic algorithm and support vectors machines. *Soft computing*, 12(7), 693-698.
- [116] Li, Y., Wang, G., Chen, H., Shi, L., & Qin, L. (2013). An ant colony optimization based dimension reduction method for high-dimensional datasets. *Journal of Bionic Engineering*, 10(2), 231-241.
- [117] Liang, Y., Liu, C., Luan, X. Z., Leung, K. S., Chan, T. M., Xu, Z. B., & Zhang, H. (2013). Sparse logistic regression with a $L_{1/2}$ penalty for gene selection in cancer classification. *BMC bioinformatics*, 14(1), 1-12.
- [118] Liu, Y., & Khoshgoftaar, T. (2004, March). Reducing overfitting in genetic programming models for software quality classification. In *Eighth IEEE International Symposium on High Assurance Systems Engineering, 2004. Proceedings.* (pp. 56-65). IEEE Computer Society.
- [119] Liu, H., & Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on knowledge and data engineering*, 17(4), 491-502.
- [120] Liu, J., Yuan, L., & Ye, J. (2010, July). An efficient algorithm for a class of fused lasso problems. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 323-332).
- [121] Liu, H., & Motoda, H. (2012). *Feature selection for knowledge discovery and data mining* (Vol. 454). Springer Science & Business Media.
- [122] Lockhart, D. J., Dong, H., Byrne, M. C., Follettie, M. T., Gallo, M. V., Chee, M. S., ... & Brown, E. L. (1996). Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature biotechnology*, 14(13), 1675-1680.
- [123] Luque-Baena, R. M., Urda, D., Claros, M. G., Franco, L., & Jerez, J. M. (2014). Robust gene signatures from microarray data using genetic algorithms enriched with biological pathway keywords. *Journal of biomedical informatics*, 49, 32-44.
- [124] Ma, S., & Huang, J. (2008). Penalized feature selection and classification in bioinformatics. *Briefings in bioinformatics*, 9(5), 392-403.
- [125] Ma, Y., & Zhu, L. (2013). A review on dimension reduction. *International Statistical Review*, 81(1), 134-150.
- [126] Mabarti, I. (2020). Implementation of Minimum Redundancy Maximum Relevance (MRMR) and Genetic Algorithm (GA) for Microarray Data Classification with C4. 5 Decision Tree. *Journal of Data Science and Its Applications*, 3(1), 38-47.
- [127] Mari, J., & Napoli, A. (1996). *Aspects de la classification*. Rapport technique 2909, INRIA, 1996.

-
- [128] Meier, L., Van De Geer, S., & Bühlmann, P. (2008). The group lasso for logistic regression. *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, 70(1), 53-71.
- [129] Meinshausen, N. (2007). Relaxed lasso. *Computational Statistics & Data Analysis*, 52(1), 374-393.
- [130] Miclet, C. A. L., & Cornuéjols, A. (2010). *Apprentissage Artificiel. Concepts et algorithmes*, 2e edition, Eyrolles.
- [131] Michie, D., Spiegelhalter, D., & C.C. . *Machine learning, neural and statistical classification*. New York : Ellis Horwood, 1994.
- [132] Mishra, D., & Sahu, B. (2011). Feature selection for cancer classification : a signal-to-noise ratio approach. *International Journal of Scientific & Engineering Research*, 2(4), 1-7.
- [133] Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & operations research*, 24(11), 1097-1100.
- [134] Mohamad, M. S., Omatu, S., Deris, S., & Yoshioka, M. (2011). A modified binary particle swarm optimization for selecting the small subset of informative genes from gene expression data. *IEEE Transactions on information Technology in Biomedicine*, 15(6), 813-822.
- [135] Mohamad, M. S., Omatu, S., Deris, S., Yoshioka, M., Abdullah, A., & Ibrahim, Z. (2013). An enhancement of binary particle swarm optimization for gene selection in classifying cancer classes. *Algorithms for Molecular Biology*, 8(1), 1-11.
- [136] Molina, L. C., Belanche, L., & Nebot, À. (2002, December). Feature selection algorithms : A survey and experimental evaluation. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.* (pp. 306-313). IEEE.
- [137] Mordukhovich, B. S., & Nam, N. M. (2013). An easy path to convex analysis and applications. *Synthesis Lectures on Mathematics and Statistics*, 6(2), 1-218.
- [138] Muselli, M., Bertoni, A., Frasca, M., Beghini, A., Ruffino, F., & Valentini, G. (2010). A mathematical model for the validation of gene selection methods. *IEEE/ACM transactions on computational biology and bioinformatics*, 8(5), 1385-1392.
- [139] Narendra, P. M., & Fukunaga, K. (1977). A branch and bound algorithm for feature subset selection. *IEEE Transactions on computers*, 26(09), 917-922.
- [140] Nemirovski, A. S., & Yudin, D. B. (1978). Cesari convergence of the gradient method of approximating saddle points of convex-concave functions. In *Doklady Akademii Nauk (Vol. 239, No. 5, pp. 1056-1059)*. Russian Academy of Sciences.
- [141] Osman, I. H., & Laporte, G. (1996). *Metaheuristics : A bibliography*.
- [142] Pacheco, J., Casado, S., & Nunez, L. (2007). Use of VNS and TS in classification : variable selection and determination of the linear discrimination function coefficients. *IMA Journal of Management Mathematics*, 18(2), 191-206.
- [143] Pardalos, P. M., & Xue, J. (1994). The maximum clique problem. *Journal of global Optimization*, 4(3), 301-328.
- [144] Pati, S. K., Das, A. K., & Ghosh, A. (2013, December). Gene selection using multi-objective genetic algorithm integrating cellular automata and rough set theory. In *International conference on swarm, evolutionary, and memetic computing* (pp. 144-155). Springer, Cham.

-
- [145] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn : Machine learning in Python. the Journal of machine Learning research, 12, 2825-2830.
 - [146] Peng, H., Long, F., & Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on pattern analysis and machine intelligence, 27(8), 1226-1238.
 - [147] Peng, Y., Wu, Z., & Jiang, J. (2010). A novel feature selection approach for biomedical data classification. Journal of Biomedical Informatics, 43(1), 15-23.
 - [148] Peng, H. : Mutual information Matlab toolbox, <http://www.mathworks.com/matlabcentral/fileexchange/14888-mutualinformation-computation>
 - [149] Pentelas, A., Sifaleras, A., & Koloniari, G. (2019, October). A reduced variable neighborhood search approach for feature selection in cancer classification. In International Conference on Variable Neighborhood Search (pp. 1-16). Springer, Cham.
 - [150] Perou, C. M., Jeffrey, S. S., Van De Rijn, M., Rees, C. A., Eisen, M. B., Ross, D. T., ... & Botstein, D. (1999). Distinctive gene expression patterns in human mammary epithelial cells and breast cancers. Proceedings of the National Academy of Sciences, 96(16), 9212-9217.
 - [151] Perou, C. M., Sørlie, T., Eisen, M. B., Van De Rijn, M., Jeffrey, S. S., Rees, C. A., ...& Botstein, D. (2000). Molecular portraits of human breast tumours. nature, 406(6797), 747-752.
 - [152] Petricoin III, E. F., Ardekani, A. M., Hitt, B. A., Levine, P. J., Fusaro, V. A., Steinberg, S. M., ... & Liotta, L. A. (2002). Use of proteomic patterns in serum to identify ovarian cancer. The lancet, 359(9306), 572-577.
 - [153] Polyak, B. T. (1967). A general method for solving extremal problems. In Doklady Akademii Nauk (Vol. 174, No. 1, pp. 33-36). Russian Academy of Sciences.
 - [154] Polyak, B. T. (1969). Minimization of unsmooth functionals. USSR Computational Mathematics and Mathematical Physics, 9(3), 14-29.
 - [155] Pudil, P., Novovičová, J., & Kittler, J. (1994). Floating search methods in feature selection. Pattern recognition letters, 15(11), 1119-1125.
 - [156] Qian, J., & Jia, J. (2016). On stepwise pattern recovery of the fused lasso. Computational Statistics & Data Analysis, 94, 221-237.
 - [157] Rinaldo, A. (2009). Properties and refinements of the fused lasso. The Annals of Statistics, 37(5B), 2922-2952.
 - [158] Rockafellar, R. T. (1970). Convex Analysis Princeton University Press. Princeton, NJ.
 - [159] Quinlan, J. R. (2014). C4. 5 : programs for machine learning. Elsevier.
 - [160] Ross, D. T., Scherf, U., Eisen, M. B., Perou, C. M., Rees, C., Spellman, P., ... & Brown, P. O. (2000). Systematic variation in gene expression patterns in human cancer cell lines. Nature genetics, 24(3), 227-235.
 - [161] Ross, K. A. (2009). Shabo (Shvo) A, Russler D, Rocha RA, Russler D, Lussier YA, Chen JL, Russler D, Zaki MJ, Corral A, Vassilakopoulos M, Gunopulos D, Wolfram D, Venkatasubramanian S, Gunopulos D, Vazirgiannis M, Davidson I, Sarawagi S, Peyton L, Hinterberger H,

- Speegle G, Vianu V, Van Gucht D, Etzion O, Etzion O, Curbera F, Ericsson A, Berndtsson M, Mellin J, Aalst WMP, Gray PMD, Trajcevski G, Wolfson O, Scheuermann P, Dorai C, Weiner M, Borgida A, Mylopoulos J, Vossen G, Reuter A, Grahne G, Tannen V, Elnikety S, Fekete A Cross-Validation. In : Liu L, Özsu MT, eds. Encyclopedia of Database Systems. Boston : Springer, 532-538.
- [162] Saeys, Y., Inza, I., & Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19), 2507-2517.
- [163] Saeys, Y., Inza, I., & Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19), 2507-2517.
- [164] Sakai, S., Togasaki, M., & Yamazaki, K. (2003). A note on greedy algorithms for the maximum weighted independent set problem. *Discrete applied mathematics*, 126(2-3), 313-322.
- [165] Schlimmer, J. C. (1993). Efficiently inducing determinations : A complete and systematic search algorithm that uses optimal pruning. In *Proceedings of the 1993 International Conference on Machine Learning* (pp. 284-290).
- [166] Sharbaf, F. V., Mosafer, S., & Moattar, M. H. (2016). A hybrid gene selection approach for microarray data classification using cellular learning automata and ant colony optimization. *Genomics*, 107(6), 231-238.
- [167] Shawe-Taylor, J., & Cristianini, N. (2000). An introduction to support vector machines and other kernel-based learning methods (Vol. 204). Volume.
- [168] Shen, Q., Shi, W. M., & Kong, W. (2008). Hybrid particle swarm optimization and tabu search approach for selecting genes for tumor classification using gene expression data. *Computational biology and chemistry*, 32(1), 53-60.
- [169] Shi, Y., & Eberhart, R. (1998, May). A modified particle swarm optimizer. In *1998 IEEE international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence* (Cat. No. 98TH8360) (pp. 69-73). IEEE.
- [170] Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8), 888-905.
- [171] Shi, C., Ruan, Q., & An, G. (2014). Sparse feature selection based on graph Laplacian for web image annotation. *Image and Vision Computing*, 32(3), 189-201.
- [172] Shor N. Z., (1962) "Application of the gradient-descent method to solution of the network transport problem," in : *Proc. Sci. Seminar on Theor. Appl. Probl. Cybern. Oper. Res.*, Kiev, Nauch. Sovet po Kibern. AN USSR [in Russian], Issue 1 , pp. 9–17
- [173] Shor, N. Z. (2012). *Minimization methods for non-differentiable functions* (Vol. 3). Springer Science & Business Media.
- [174] Shreem, S. S., Abdullah, S., Nazri, M. Z. A., & Alzaqebah, M. A. L. E. K. (2012). Hybridizing ReliefF, MRMR filters and GA wrapper approaches for gene selection. *J. Theor. Appl. Inf. Technol*, 46(2), 1034-1039.
- [175] Shultz, T. R., Fahlman, S. E., Craw, S., Andritsos, P., Tsaparas, P., Silva, R., & Mueen, A. (2011). Curse of Dimensionality. *Encyclopedia of Machine Learning*, 257-258.

- [176] Singh, D., Febbo, P. G., Ross, K., Jackson, D. G., Manola, J., Ladd, C., ... & Lander, E. S. (2002). Gene expression correlates of clinical prostate cancer behavior. *Cancer cell*, 1(2), 203-209.
- [177] Skalak, D. B. (1994). Prototype and feature selection by sampling and random mutation hill climbing algorithms. In *Machine Learning Proceedings 1994* (pp. 293-301). Morgan Kaufmann.
- [178] Solnon, C. (2002). Ants can solve constraint satisfaction problems. *IEEE transactions on evolutionary computation*, 6(4), 347-357.
- [179] Statnikov, A., Aliferis, C. F., Tsamardinos, I., Hardin, D., & Levy, S. (2005). A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics*, 21(5), 631-643.
- [180] Syu, G. D., Dunn, J., & Zhu, H. (2020). Developments and applications of functional protein microarrays. *Molecular & Cellular Proteomics*, 19(6), 916-927.
- [181] Tabakhi, S., Moradi, P., & Akhlaghian, F. (2014). An unsupervised feature selection algorithm based on ant colony optimization. *Engineering Applications of Artificial Intelligence*, 32, 112-123.
- [182] Tabakhi, S., Najafi, A., Ranjbar, R., & Moradi, P. (2015). Gene selection for microarray data classification using a novel ant colony optimization. *Neurocomputing*, 168, 1024-1036.
- [183] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society : Series B (Methodological)*, 58(1), 267-288.
- [184] Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., & Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, 67(1), 91-108.
- [185] Tibshirani, R. J., & Taylor, J. (2011). The solution path of the generalized lasso. *The annals of statistics*, 39(3), 1335-1371.
- [186] Tsamardinos, I., & Aliferis, C. F. (2003, January). Towards principled feature selection : Relevancy, filters and wrappers. In *International Workshop on Artificial Intelligence and Statistics* (pp. 300-307). PMLR.
- [187] Viallon, V., Lambert-Lacroix, S., Hoefling, H., & Picard, F. (2016). On the robustness of the generalized fused lasso to prior specifications. *Statistics and Computing*, 26(1-2), 285-301.
- [188] Wang, Z. (2005). Neuro-fuzzy modeling for microarray cancer gene expression data. First year transfer report, University of Oxford.
- [189] Wang, Y., Miller, D. J., & Clarke, R. (2008). Approaches to working in high-dimensional data spaces : gene expression microarrays. *British journal of cancer*, 98(6), 1023-1028.
- [190] Wang, S., Nan, B., Rosset, S., & Zhu, J. (2011). Random lasso. *The annals of applied statistics*, 5(1), 468.
- [191] Wang, S., Kong, W., Zeng, W., & Hong, X. (2016). Hybrid binary imperialist competition algorithm and tabu search approach for feature selection using gene expression data. *BioMed Research International*, 2016.

- [192] Wang, S., Kong, W., Zeng, W., & Hong, X. (2016). Hybrid binary imperialist competition algorithm and tabu search approach for feature selection using gene expression data. *BioMed Research International*, 2016.
- [193] Wang, Y., Zhang, W., Fan, M., Ge, Q., Qiao, B., Zuo, X., & Jiang, B. (2021). Regression with Adaptive Lasso and Correlation based Penalty. *Applied Mathematical Modelling*.
- [194] Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., ... & Steinberg, D. (2008). Top 10 algorithms in data mining. *Knowledge and information systems*, 14(1), 1-37.
- [195] Xing, E. P., Jordan, M. I., & Karp, R. M. (2001, June). Feature selection for high-dimensional genomic microarray data. In *Icml* (Vol. 1, pp. 601-608).
- [196] Xue, B., Zhang, M., Browne, W. N., & Yao, X. (2015). A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation*, 20(4), 606-626.
- [197] Yadav, S., & Shukla, S. (2016, February). Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification. In *2016 IEEE 6th International conference on advanced computing (IACC)* (pp. 78-83). IEEE.
- [198] Yang, Y., & Pedersen, J. O. (1997, July). A comparative study on feature selection in text categorization. In *Icml* (Vol. 97, No. 412-420, p. 35).
- [199] Yang, B. (2009). Modified particle swarm optimizers and their application to robust design and structural optimization (Doctoral dissertation, Technische Universität München).
- [200] Yang, Z. Y., Liang, Y., Zhang, H., Chai, H., Zhang, B., & Peng, C. (2018). Robust Sparse Logistic Regression With the $L_q(0 < q < 1)$ Regularization for Feature Selection Using Gene Expression Data. *IEEE Access*, 6, 68586-68595.
- [201] Yu, L., & Liu, H. (2003). Feature selection for high-dimensional data : A fast correlation-based filter solution. In *Proceedings of the 20th international conference on machine learning (ICML-03)* (pp. 856-863).
- [202] Yu, L., & Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. *The Journal of Machine Learning Research*, 5, 1205-1224.
- [203] Yu, H., Gu, G., Liu, H., Shen, J., & Zhao, J. (2009). A modified ant colony optimization algorithm for tumor marker gene selection. *Genomics, proteomics & bioinformatics*, 7(4), 200-208.
- [204] Zhang, Y., & Yang, Y. (2015). Cross-validation for selecting a model selection procedure. *Journal of Econometrics*, 187(1), 95-112.
- [205] Zhang, D., Chen, S., & Zhou, Z. H. (2008). Constraint score : A new filter method for feature selection with pairwise constraints. *Pattern Recognition*, 41(5), 1440-1451.
- [206] Zhang, Z., & Hancock, E. R. (2012). Hypergraph based information-theoretic feature selection. *Pattern Recognition Letters*, 33(15), 1991-1999.
- [207] Zhang, H., & Li, S. (2013). DNA Microarray Assay Helps to Identify Functional Genes Specific for Leukemia Stem Cells. *Dataset Papers in Science*, 2013.

- [208] Zhao, W., Wang, G., Wang, H. B., Chen, H. L., Dong, H., & Zhao, Z. D. (2011). A novel framework for gene selection. *Int J Adv Comput Technol*, 3(3), 184-91.
- [209] Zhu, Z., Ong, Y. S., & Dash, M. (2007). Wrapper-filter feature selection algorithm using a memetic framework. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(1), 70-76.
- [210] Zibakhsh, A., & Abadeh, M. S. (2013). Gene selection for cancer tumor detection using a novel memetic algorithm with a multi-view fitness function. *Engineering Applications of Artificial Intelligence*, 26(4), 1274-1281.
- [211] Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society : series B (statistical methodology)*, 67(2), 301-320.
- [212] Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476), 1418-1429.

Résumé

Dans cette thèse, nous proposons des méthodes innovantes pour réduire la taille initiale des données de puces à ADN et sélectionner des ensembles de gènes pertinents qui assurent une bonne précision dans l'identification des tumeurs cancéreuses. Premièrement, nous avons proposé plusieurs méthodes de sélection de gènes, basées sur les Métaheuristiques BPSO (optimisation par essaim de particules version binaire), ACO (optimisation par colonies de fourmis) et VNS (recherche de voisinage variable). En effet:

- Trois approches filtres multivariées ont été proposées: MWIS (ensemble indépendant de poids maximum), VNSMI et VNSCorr. Les deux dernières méthodes sont basées sur la combinaison de l'algorithme MRMR (redondance minimal-pertinence maximale) et d'une recherche intelligente via l'algorithme de VNS.
- D'autres approches hybrides ont été proposées telles que: FBPSO, RBPSO et MWIS-ACO-LS et ont montré des performances élevées, mais leur temps d'exécution est aussi important à cause de l'utilisation des classifieurs dans l'étape de sélection.

La qualité des sous-ensembles de gènes dans ces méthodes est mesurée à l'aide des classifieurs SVM (Machine à vecteurs de support) et KNN (K plus proches voisins). Enfin, dans le but d'améliorer les méthodes intégrées, nous avons formulé le problème de sélection de gènes comme un problème de régression logistique (maximum de log-vraisemblance) avec une pénalité de type Fused LASSO généralisé. Pour résoudre ce problème d'optimisation, nous avons utilisé l'algorithme de sous-gradient. L'application de ce dernier nécessite la validation de certaines conditions de convergence. Dans notre travail, nous avons analysé théoriquement et numériquement cette méthode et vérifié que notre fonction objectif remplit toutes ces conditions.

Les différentes expérimentations que nous avons menées montrent que nos approches sont performantes et compétitives et produisent des sous-ensembles de petite taille avec une haute performance pour la classification des cancers.

Mots-clefs: Sélection d'attributs; Classification supervisée, puces à ADN; Optimisation par Essaim de Particules; Optimisation par Colonies de Fourmis; Recherche de Voisinage Variable; Ensemble Indépendant de Poids Maximum; Régression Logistique Pénalisée; Generalized Fused LASSO.

Abstract

In this thesis, we propose innovative methods to reduce the initial dimension of microarray data, and select relevant gene sets that ensure good accuracy in cancer tumor identification: First, we proposed several gene selection methods, based on the metaheuristics BPSO (Binary Particle Swarm Optimization), ACO (Ant Colony Optimization) and (VNS Variable Neighborhood Search), indeed:

- Three multivariate filter approaches have been proposed: MWIS (Maximum Weight Independent Set), VNSMI and VNSCorr. The last two methods are based on the combination of the MRMR (minimum redundancy-maximum relevance) algorithm and an intelligent search via the VNS algorithm.
- Other hybrid approaches have been proposed such as FBPSO, RBPSO and MWIS-ACO-LS and have shown high performances, but their execution time is also important because of the use of classifiers in the selection step.

The quality of the gene subsets in these methods is measured using SVM (Support Vector Machine) and KNN (K-Nearest Neighbors) classifiers. And finally, in order to improve the integrated methods, we formulated the gene selection problem as a logistic regression problem (maximum log-likelihood) with a generalized Fused LASSO penalty. To solve this optimization problem, we used the sub-gradient algorithm. The application of the latter requires the validation of some convergence conditions. In our work, we have analyzed theoretically and numerically this method and verified that our objective function meets all these conditions.

The different experiments we have conducted show that our approaches are efficient and competitive and produce small subsets with high performance for cancer classification.

Key Words: Feature selection; Supervised classification; DNA microarray; Particle Swarm Optimization, Ant Colony Optimization, Variable Neighborhood Search; Maximum Weight Independent Set; Penalized Logistic Regression; Generalized Fused LASSO.