Aide au choix d'une solution optimale sur un front Pareto à l'aide de méthodes de segmentation.

Ilyas RAHHALI
Directeurs : Charles Audet, Jonathan Jalbert

Polytechnique Montréal
Gerad

Montréal, 9 Juillet 2020

Plan

Introduction

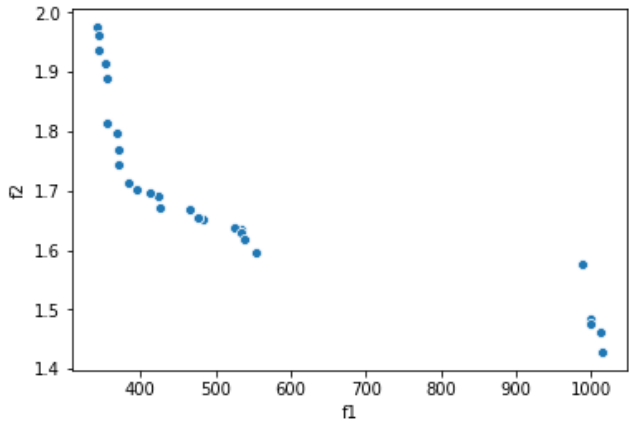

Figure 1 – Solutions optimales d'un problème biobjectif

- Choisir une seule solution parmi celles-ci peut être difficile
- On doit vérifier les caractéristiques des solutions une par une

Comment choisir une solution qui vérifie les critères de l'ingénieur ?

Optimisation biobjectif

Problème biobjectif [1]

Un problème biobjectif se formule de la manière suivante :

$$\min_{x \in \Omega} F(x) = (f_1(x), f_2(x)) \tag{1}$$

Points dominés [1]

Soient $x, y \in \Omega$
On dit que x *domine* y *si et seulement si* $f_i(x) \leq f_i(y)$ *pour* $i \in \{1, 2\}$ *avec au moins l'une des inégalités qui doit être stricte*
On écrit alors $x \prec y$

Points non dominés ou Pareto optimaux [1]

Soit $x \in \Omega$
x *est dit non dominé ou Pareto optimal si et seulement si* $\nexists\ y \in \Omega$ *tel que* $y \prec x$

Définiton de segmentation

Apprentissage non supervisé ou segmentation

- Catégorie de l'apprentissage machine qui tente de découvrir une structure latente à la distribution d'un jeu de données
- Permet d'identifier des groupements de points
- Un groupement est un ensemble de points regroupés à partir de certaines similarités

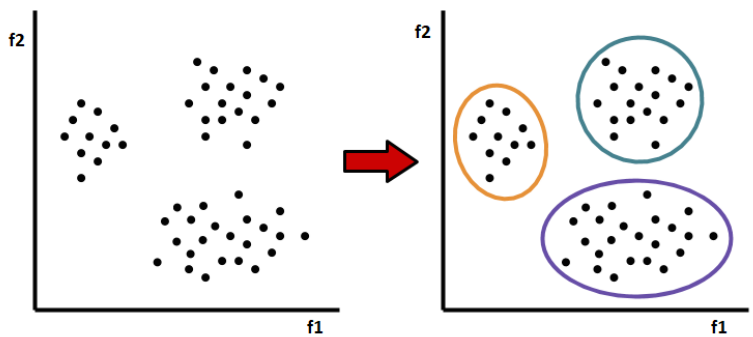

Figure 2 – Exemple de segmentation [2]

Algorithmes de segmentation

kmeans [3]

- Présuppose un nombre de groupements prédéfini K
- Algorithme assignant chaque point au centroïde le plus proche itérativement
- Le but est de minimiser la variance au sein de chaque groupement notée s

Algorithme 1 Kmeans

1: *Trouver un nombre de groupements K*
2: *Initialiser K centroïdes* $\{c_1, .., c_K\}$
3: **faire**
4: $s = \sum_{k=1}^{K} \sum_{i=1}^{N} (x_i - c_k)^2$
5: *Assigner chaque point* x_i *au centroïde* c_k *le plus proche*
6: *Mettre à jour les centroïdes*
7: **tant que** *Changement de centroïdes*

Meanshift [4]

- Aussi appelé algorithme de recherche de mode.
- Considère l'espace de paramètres comme une fonction de densité de probabilité dont les points qui lui sont fournis sont considérés comme échantillons.
- Considère que s'il existe des groupements dans les données, ils correspondent aux modes ou maxima locaux de cette fonction de densité.
- Pas besoin de spécifier un nombre de groupements à l'avance.
- Retourne un groupement pour chaque mode trouvé.

Algorithme 2 Meanshift

1: *Sélectionner une fenêtre F*
2: **pour** chaque point p du jeu de données **faire**
3: **tant que** pas encore convergé **faire**
4: *Calculer la moyenne des points dans une fenêtre F autour du point p*
5: *Déplacer la fenêtre F pour la centrer sur la nouvelle moyenne*
6: **fin tant que**
7: **fin pour**

HDBscan [5]

- Extension de l'algorithme par densité DBscan
- Fait peu d'hypothèses sur la structure des données
- Plutôt que de chercher des groupements d'une certaine forme, il cherche les régions denses de l'espace
- Excellent pour les données bruitées
- Très sensible à ses hyperparamètres

Algorithme 3 HDBscan

1: *Transformer l'espace suivant la densité*
2: *Construire un arbre couvrant minimal à partir du graphe pondéré*
3: *Construire une hiérarchie de segmentation à partir des composants connectés*
4: *Condenser cette hiérarchie en se basant sur la taille de groupement minimale*
5: *Extraire les groupements stables de l'arbre condensé*

Algorithmes de segmentation

Algorithme spectral [6]

- Algorithme n'utilisant pas de mesure de distance mais plutôt la matrice de similarité lui octroyant une robustesse en haute dimension
- Se base sur les valeurs propres de la matrice de similarité pour réduire la dimensionnalité des données avant de performer la segmentation

Algorithme 4 Segmentation spectrale [7]

1: *Construire le graphe de similarité. Soit W sa matrice d'adjascence pondérée*
2: *Calculer le laplacien normalisé L*
3: *Calculer les k premier vecteurs propres* $u_1, ..u_k$ *de L (k étant le nombre de groupements)*
4: *Construire la matrice T à partir des* u_i *comme colonnes puis normaliser ses lignes suivant la norme 1*
5: *Effectuer une segmentation par kmeans sur les lignes de la matrice T*

Méthodologie proposée

Démarche suivie

- Génération et préparation des données
- Choix d'un algorithme de segmentation
- Présentation des groupements et leurs caractéristiques
- Choix de représentant par groupement

Méthodologie proposée

Génération et préparation des données

- Récupérer la cache de l'algorithme d'optimisation biobjectif
- Normaliser les données
- Extraire les points non dominés dans un ensemble noté P
- Étendre P en y ajoutant les points à un certain rayon r de chacun des points dans l'espace (f_1, f_2)

Méthodologie proposée

Génération et préparation des données

Soit C la cache du problème biobjectif définie comme suit :

$$C = \{(x, f(x), c(x)) \in \mathbf{R^n} \times \mathbf{R^2} \times \mathbf{R^m} : x \in \{x^1, x^2, \ldots, x^p\}\}$$

où $x^1, \ldots, x^p$ sont les points visités par l'algorithme.

Algorithme 5 Extraction des points non dominés

Entrée: *cache* : C

1: **fonction** ExtractPareto
2: $P = \emptyset \leftarrow$ *Front Pareto*
3: **pour** $x \in C$ **faire**
4: **si** x est non dominé dans C et $x \in \Omega$ **alors**
5: $P \leftarrow P \cup \{x\}$
6: **fin si**
7: **fin pour**
8: **retourne** P
9: **fin fonction**

Méthodologie proposée

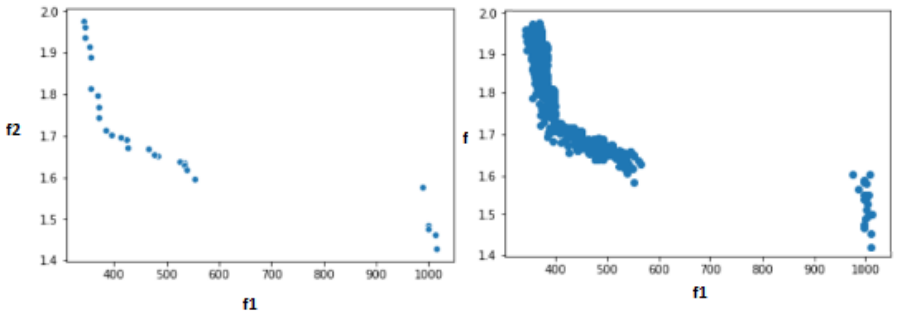

Figure 3 – Relaxation du Front Pareto

Algorithme 6 Relaxation du front Pareto

Entrée: cache C, pareto P, rayon r

fonction RELAXPARETO
    $R = \emptyset \leftarrow$ *front relâché*
    pour x *dans* P **faire**
        pour y *dans* $C \setminus \{x\}$ **faire**
            si $y \in B_r(F(x))$ **alors**
                $R \leftarrow R \cup \{y\}$
            fin si
        fin pour
    fin pour
    retourne R
fin fonction

Méthodologie proposée

Choix d'un algorithme de segmentation

- Comparer la performance des 4 algorithmes de segmentation présentés précédemment
- En choisir un qui a de bonne performances bien dans un cadre général
- Vérifier sur un problème réel

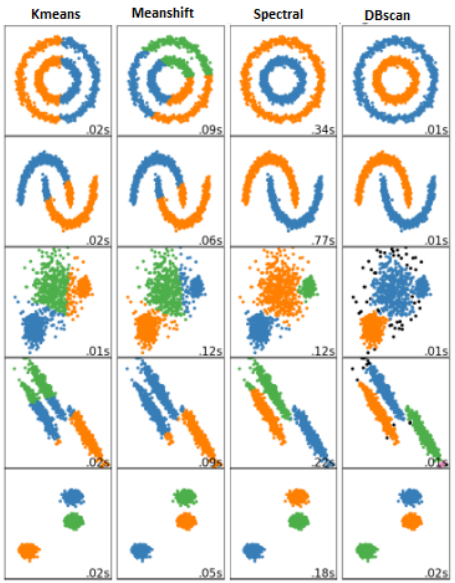

Figure 4 – Comparaison des 4 algos [8]

Méthodologie proposée

Voici un exemple pour illustrer pourquoi Kmeans risque d'avoir de mauvaises performances en hautes dimensions.

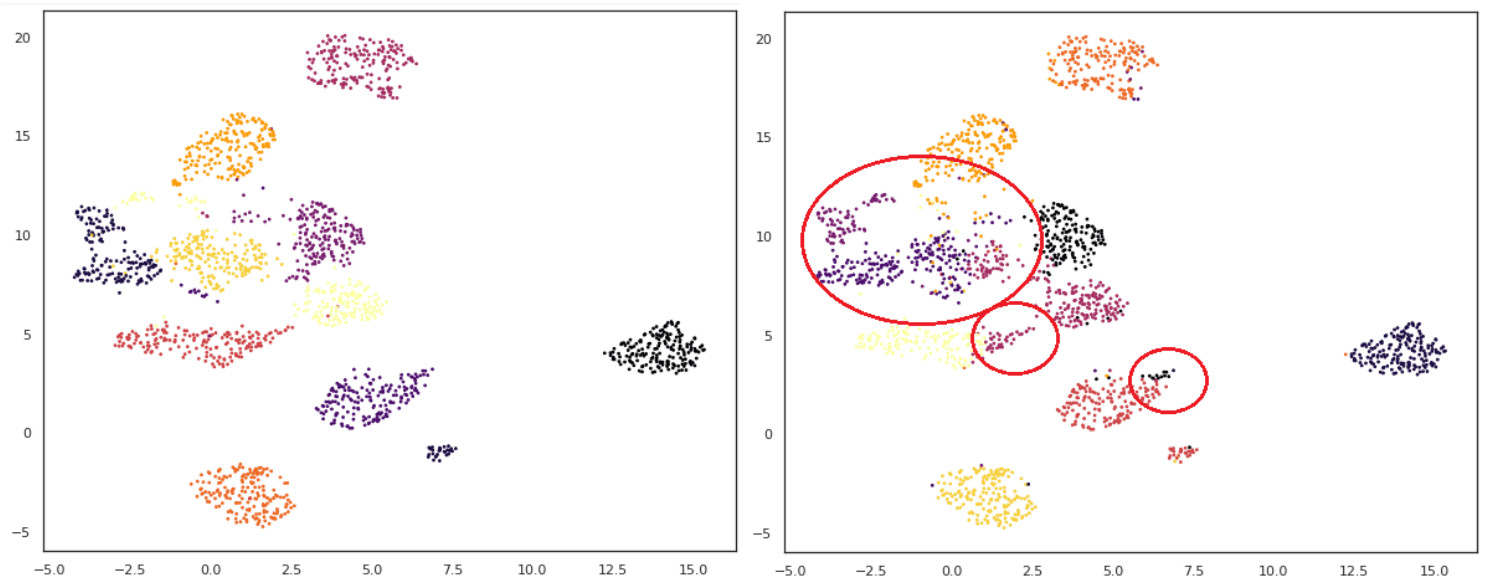

Figure 5 – Mauvaus résultats par Kmeans

Méthodologie proposée

Choix d'un représentant par groupement

Deux méthodes sont proposées, l'utilisateur pourra choisir selon ses besoins

- Choix du point Pareto le plus proche du centroïde de chaque groupement

---

Algorithme 7 choix de représentant

Entrée: Ensemble de groupements : $G = \{G_k\}, k \in \{1..p\}$

1: **fonction** HEURISTIQUE 1
2: **pour** $G_k \in G$ **faire**
3: $c_k = \left(\frac{1}{|G_k|} \sum_{x \in G_k} f_1(x), \frac{1}{|G_k|} \sum_{x \in G_k} f_2(x) \right)$ *dans l'espace* (f_1, f_2)
4: Choisir $x_{repr_k} = \min_{x \in P} d(x - c_k)$
5: **fin pour**
6: **fin fonction**

- Construire une fonction de coût à partir des critères de l'utilisateur qui sera utilisée pour trier ces points, ou effectuer une optimisation mono-objectif dessus

Exemple pratique

Définition du problème

On souhaite synthétiser un fluide caloporteur de température de fusion et d'activité de solution solides minimaux à partir de huits sels. Ce problème peut donc être reformulé sous forme de problème d'optimisation comme suit :

$$\min_{x \in \Omega} \quad F(x) = (f_1(x), f_2(x))$$

$$\text{avec} \quad F : \mathbf{R^8} \longrightarrow \mathbf{R^2} \; et \; x \in \Omega \subseteq \mathbf{R^8}$$

Exemple pratique

Lecture des données :

	LiCl	NaCl	KCl	CaCl_2	MgCl_2	MnCl_2	NiCl_2	CoCl_2	f1	f2
0	33.0	54.0	66.0	49.0	6.0	28.0	9.0	66.0	413.2337	1.111307
1	90.0	85.0	81.0	54.0	93.0	66.0	100.0	16.0	564.5982	0.880445
2	15.0	39.0	51.0	21.0	54.0	61.0	91.0	18.0	658.5661	0.808576
3	66.0	24.0	83.0	35.0	70.0	30.0	19.0	91.0	472.5585	1.086090
4	23.0	20.0	54.0	77.0	65.0	77.0	47.0	57.0	615.7641	0.845349

Figure 6 – Aperçu des variables du problème produit par BIMADS

Exemple pratique

Normalisation des données :

	x1	x2	x3	x4	x5	x6	x7	x8	f1	f2
0	0.106109	0.173633	0.212219	0.157556	0.019293	0.090032	0.028939	0.212219	0.396616	0.806089
1	0.153846	0.145299	0.138462	0.092308	0.158974	0.112821	0.170940	0.027350	0.541894	0.638632
2	0.042857	0.111429	0.145714	0.060000	0.154286	0.174286	0.260000	0.051429	0.632083	0.586502
3	0.157895	0.057416	0.198565	0.083732	0.167464	0.071770	0.045455	0.217703	0.453555	0.787798
4	0.054762	0.047619	0.128571	0.183333	0.154762	0.183333	0.111905	0.135714	0.591002	0.613176

Figure 7 – Normalisation des données

Exemple pratique

Distribution des solutions et extraction des points Pareto

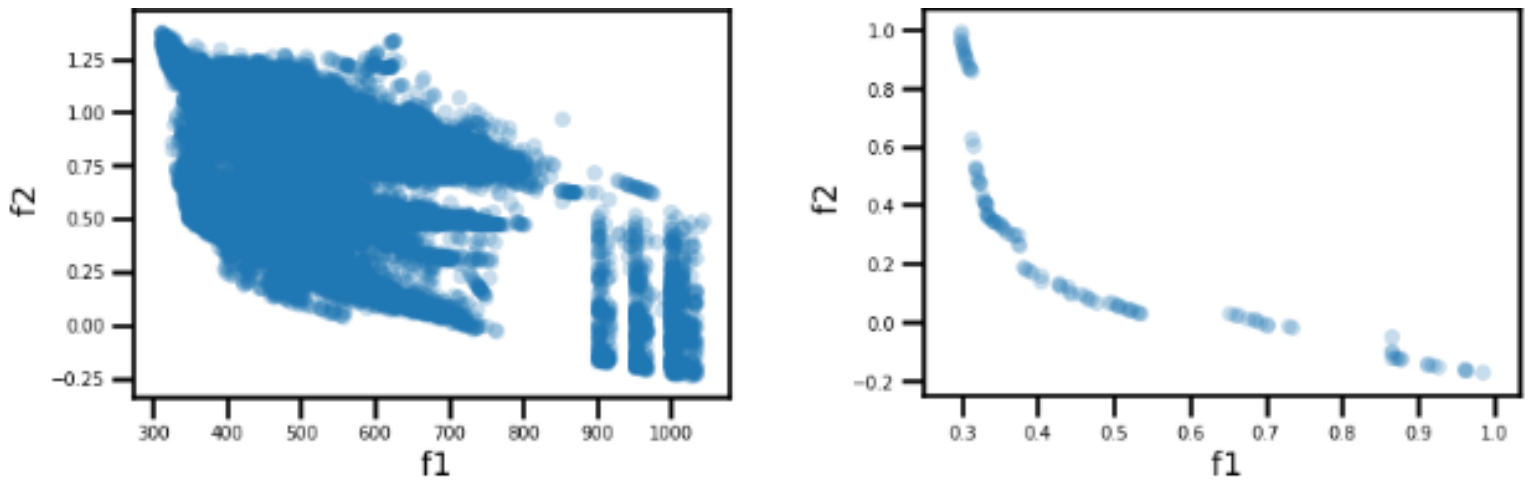

Figure 8 – Solutions dans l'espace (f_1, f_2) *(gauche)*, Points Pareto normalisés *(droite)*

Exemple pratique

Relaxation du front Pareto :

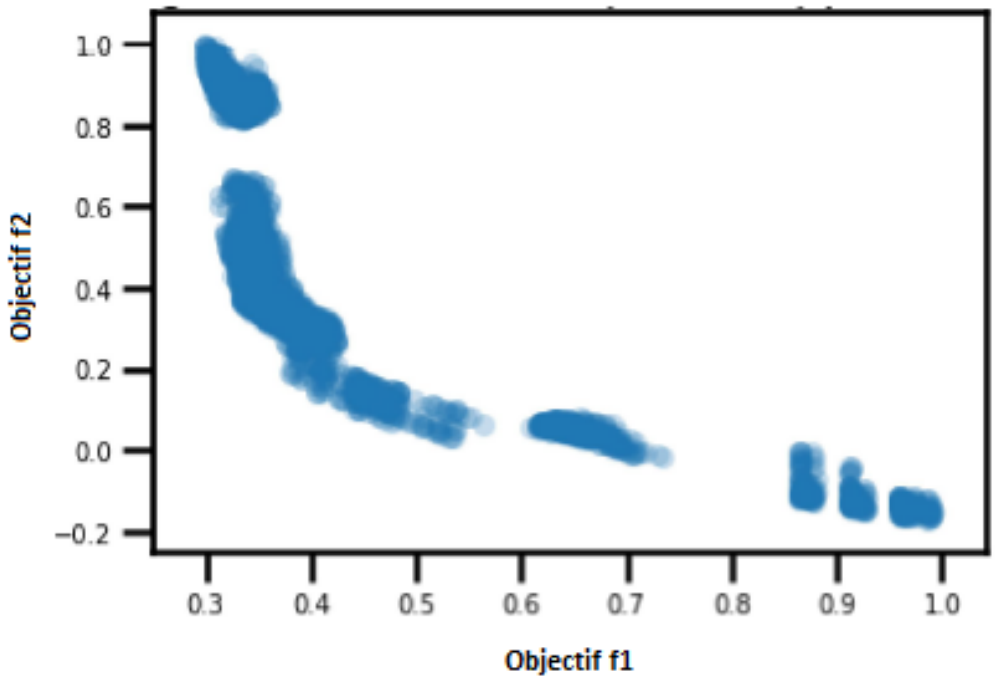

Figure 9 – Points approximés autour des points non dominés

Exemple pratique

Application des algorithmes de segmentation :

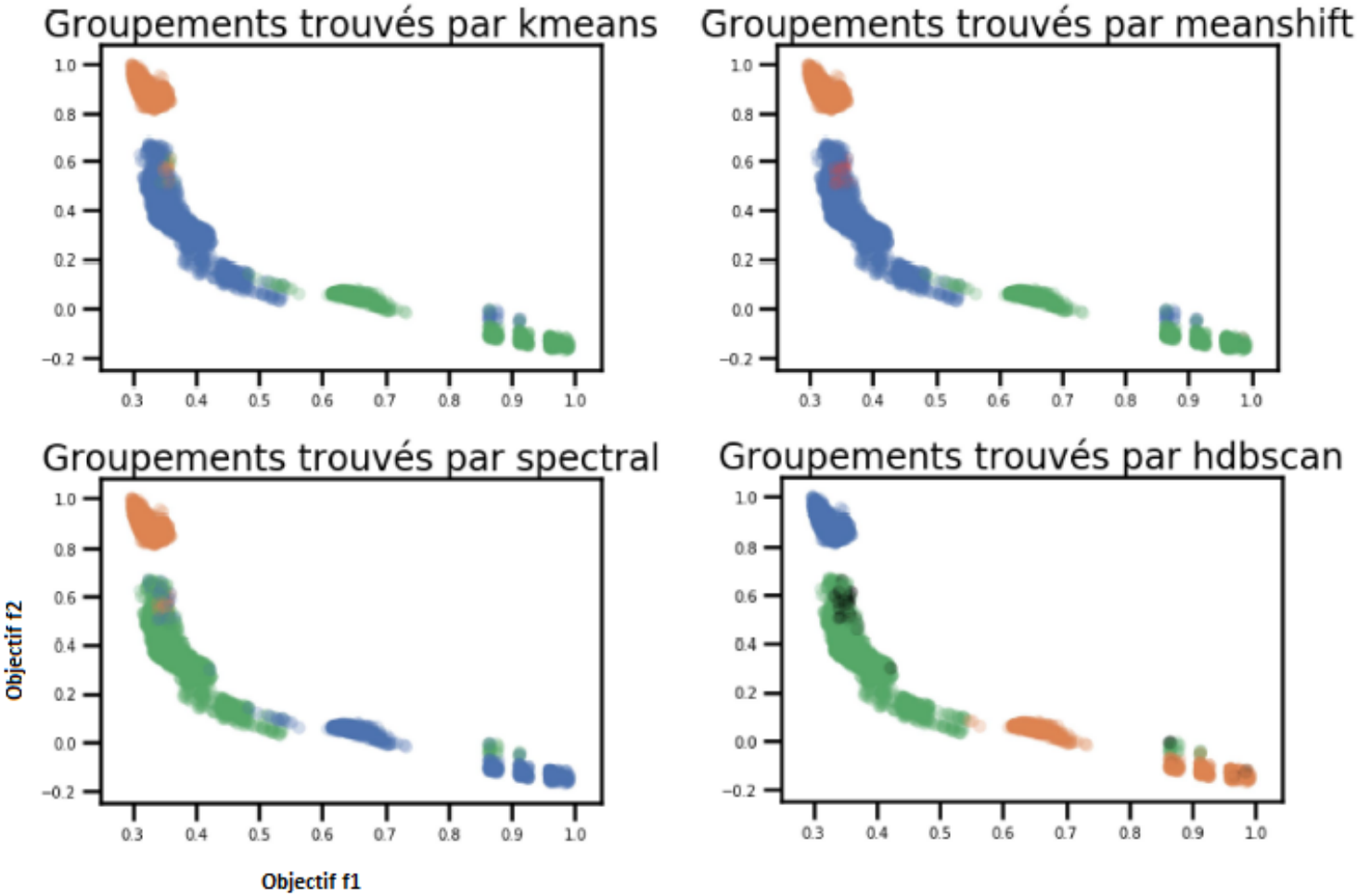

Figure 10 – Groupements trouvés par chaque algorithme

Exemple pratique

Informations sur les groupements des groupements :

La distribution des Xi pour la classe 1 est :

	x1	x2	x3	x4	x5	x6	x7	x8
count	662.000000	662.000000	662.000000	662.000000	662.000000	662.000000	662.000000	662.000000
mean	0.102261	0.003901	0.792250	0.014973	0.011571	0.031722	0.029404	0.013917
std	0.035357	0.009242	0.068488	0.028516	0.015871	0.033431	0.029988	0.037031
min	0.017544	0.000000	0.503448	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.081538	0.000000	0.777778	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.097561	0.000000	0.806452	0.000000	0.000000	0.024096	0.023529	0.000000
75%	0.114754	0.000000	0.830932	0.016598	0.016227	0.046110	0.046875	0.015385
max	0.268293	0.081633	0.925000	0.280899	0.087719	0.265957	0.202703	0.347518

La distribution des Xi pour la classe 2 est :

	x1	x2	x3	x4	x5	x6	x7	x8
count	899.000000	899.000000	899.000000	899.000000	899.000000	899.000000	899.000000	899.000000
mean	0.036920	0.190741	0.189605	0.109073	0.089589	0.161444	0.001913	0.220715
std	0.020052	0.029444	0.049380	0.053933	0.028608	0.042488	0.002709	0.057699
min	0.000000	0.030928	0.086817	0.000000	0.000000	0.016713	0.000000	0.020305
25%	0.024096	0.189024	0.168496	0.056131	0.079832	0.122899	0.000000	0.199203
50%	0.036403	0.196464	0.177515	0.103030	0.088710	0.174419	0.000000	0.206612
75%	0.043668	0.202492	0.202052	0.152720	0.105051	0.192308	0.002941	0.275135
max	0.170103	0.260450	0.500000	0.283828	0.219178	0.365546	0.024064	0.406504

La distribution des Xi pour la classe 3 est :

	x1	x2	x3	x4	x5	x6	x7	x8
count	1181.00000	1181.00000	1181.00000	1181.00000	1181.00000	1181.00000	1181.00000	1181.00000
mean	0.414739	0.048527	0.479904	0.006008	0.011523	0.028545	0.004088	0.006666
std	0.111601	0.039562	0.068510	0.014327	0.018476	0.038971	0.007400	0.015046
min	0.213592	0.000000	0.180451	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.344156	0.005348	0.458333	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.377483	0.053892	0.492611	0.000000	0.005714	0.014815	0.000000	0.000000
75%	0.492857	0.081633	0.520710	0.006494	0.016043	0.033898	0.005988	0.006757
max	0.800000	0.176471	0.644444	0.135338	0.120690	0.191919	0.045977	0.205021

Figure 11 – Statistiques des groupements

Exemple pratique

Sélection d'un représentant par groupement par la première heuristique :

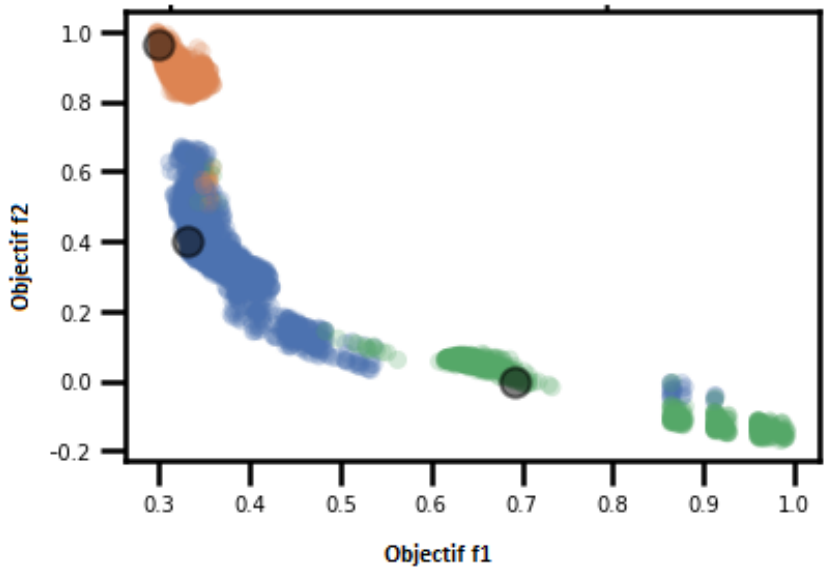

Figure 12 – Représentants par groupement par l'heuristique 1

Exemple pratique

Pour la deuxième heuristique, on définit le coût d'une solution comme suit :

pour $x \in \Omega$

$$C(x) = \frac{\sum p_i * x_i}{\sum p_i}$$

avec p_i : le prix du sel i

Pour les prix du sel, on considère les valeurs suivantes en \$/kg :

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
2745	32.25	105.6	51.9	112.95	157.05	861	2692.5

Exemple pratique

Après calcul des coûts des solutions on obtient l'aperçu suivant des solutions les moins coûteuses pour chaque classe :

	x1	x2	x3	x4	x5	x6	x7	x8	f1	f2	Coût	labels
550	0.152174	0.054348	0.532609	0.141304	0.000000	0.108696	0.010870	0.000000	0.334740	0.634193	0.075386	0
519	0.159091	0.045455	0.534091	0.125000	0.000000	0.125000	0.011364	0.000000	0.328234	0.641776	0.078493	0
706	0.164835	0.054945	0.538462	0.120879	0.000000	0.120879	0.000000	0.000000	0.348575	0.569744	0.079364	0
511	0.170455	0.045455	0.545455	0.125000	0.000000	0.113636	0.000000	0.000000	0.350997	0.560320	0.081574	0
330	0.170455	0.045455	0.534091	0.125000	0.000000	0.125000	0.000000	0.000000	0.327724	0.648465	0.081661	0
9	0.006042	0.178248	0.163142	0.235650	0.151057	0.229607	0.012085	0.024169	0.358238	0.861207	0.026693	1
2733	0.022843	0.157360	0.182741	0.230964	0.177665	0.197970	0.010152	0.020305	0.352394	0.872306	0.031611	1
319	0.000000	0.218341	0.218341	0.218341	0.072052	0.218341	0.000000	0.054585	0.349576	0.890067	0.034155	1
1665	0.000000	0.218818	0.218818	0.218818	0.070022	0.218818	0.000000	0.054705	0.349716	0.906245	0.034193	1
1141	0.000000	0.219780	0.219780	0.219780	0.065934	0.219780	0.000000	0.054945	0.349995	0.899537	0.034270	1
2445	0.037500	0.000000	0.925000	0.000000	0.012500	0.025000	0.000000	0.000000	0.706872	0.016649	0.030475	2
968	0.017544	0.000000	0.842105	0.000000	0.052632	0.000000	0.087719	0.000000	0.686435	0.061866	0.032339	2
409	0.033898	0.000000	0.864407	0.016949	0.033898	0.016949	0.033898	0.000000	0.918039	-0.107324	0.032684	2
2455	0.037037	0.000000	0.876543	0.000000	0.049383	0.000000	0.037037	0.000000	0.691357	0.042695	0.034283	2
2702	0.048387	0.000000	0.887097	0.000000	0.000000	0.064516	0.000000	0.000000	0.690095	0.023490	0.035014	2

Figure 13 – Tableau des solutions triées par coût

Exemple pratique

En utilisant cette deuxième heuristique pour sélectionner les points le point le moins coûteux pour chaque classe, on obtient le résultat suivant :

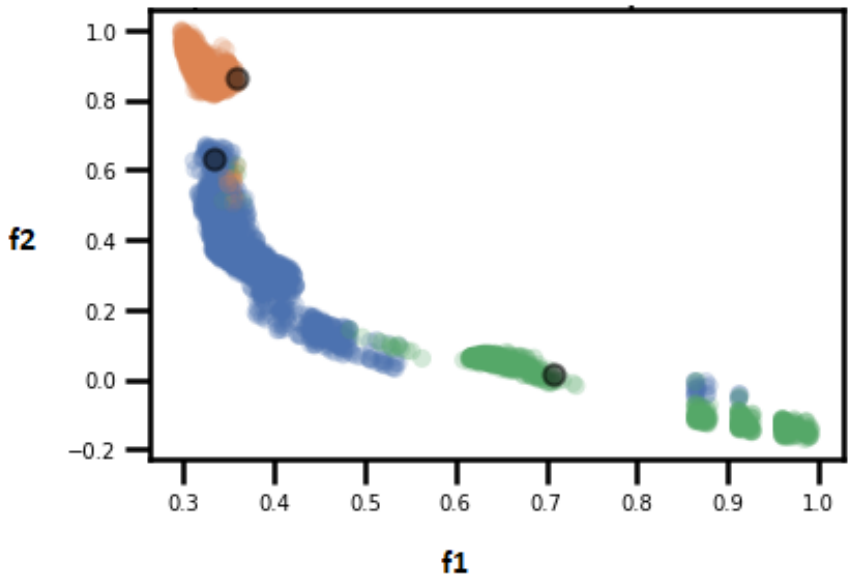

Figure 14 – Représentants par groupement par l'heuristique 2

Conclusion

Merci pour votre attention, des questions ?

References I

C.Audet and W. Hare (2017)
Biobjective Optimization
Derivative-free and blackbox optimization
Springer, pp247–262.

Stanford university
https ://thedishonscience.stanford.edu/posts/clustering-breast-cancer-data/images/unsupervised-learning-example.png

MacKay David (2003)
Chapter 20. An Example Inference Task : Clustering
Information theory, inference and learning algorithms
Cambridge university press, pp284–292.

K. Fukunaga, L. Hostetler (1975)
The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition
IEEE Transactions on Information Theory vol 21 no. 1 pp32–40

Campello R.J.G.B., Moulavi D., Sander J. (2013)
Density-Based Clustering Based on Hierarchical Density Estimates.
Advances in Knowledge Discovery and Data Mining.

References I

Ng, Andrew Y and Jordan, Michael I and Weiss, Yair (2002).
On spectral clustering : analysis and an algorithm
Advances in Neural Information Processing Systems.

Ulrike von Luxburg (2007).
A Tutorial on Spectral Clustering
Springler, Statistics and Computing, 17 (4), 2007

Sklearn library
https ://scikit-learn.org/stable/modules/clustering.html