

Projet ADD, t-SNE

Kevin McKenna, Ismaïl Ramdé, Fanny Rebiffé

4/14/2021

Introduction

Les techniques de réduction de dimensions occupent une grande place dans le domaine de la statistique. Elles prennent toute leur importance quand on a affaire à des données de très grande taille (dimensions). Pour cela nous nous intéressons en particulier à deux méthodes très connus notamment t-sne ((t-distributed Stochastic Neighbor Embedding) et l'ACP (Analyse en Composantes principales). Toutefois il faut noter que ces méthodes ne conservent pas la totalité des informations contenues dans les données initiales. L'enjeu est donc de trouver un juste milieu entre réduction de dimension et perte d'informations. Dans les lignes qui suivent nous étudierons ces deux méthodes tout en les mettant en opposition afin d'en déduire les forces et faiblesses de chacune d'elle.

Description du tSNE

Tsne (t-distributed Stochastic Neighbor Embedding), dont le t désigne une loi t de student, est une technique de réduction de dimensions non linéaire par apprentissage automatique. Elle permet de visualiser des données de très grandes dimensions à travers un prolongement dans une variété de plus petite dimension afin de mettre en lumière des caractéristiques intéressantes.

Contrairement à certaines méthodes comme l'ACP qui préserve une grande distance par paire pour maximiser la variance et MDS qui vise à préserver le classement des distances entre les espaces d'entrée et de sortie, t-SNE met l'accent sur les petites distances. En d'autres termes il ne préserve que les similitudes locales. Sa force réside dans la création de clusters compacts pour la visualisation. En effet cette méthode prend en compte chaque point du jeu de données séparément et affecte une probabilité P_{ij} conditionnelle (ou poids) gaussienne à chacun des autres points en fonction de leur distance par rapport à ce point. À l'issu de cela une nouvelle dimension inférieure au jeu de données est calculée à travers une nouvelle distribution Q_{ij} de t-student, qui minimise la divergence de Kullback-Leibler entre P et Q. Cette méthode permet d'obtenir un espace de dimension inférieur tout en respectant une distribution proche en Kullback-Leibler divergence d'origine.

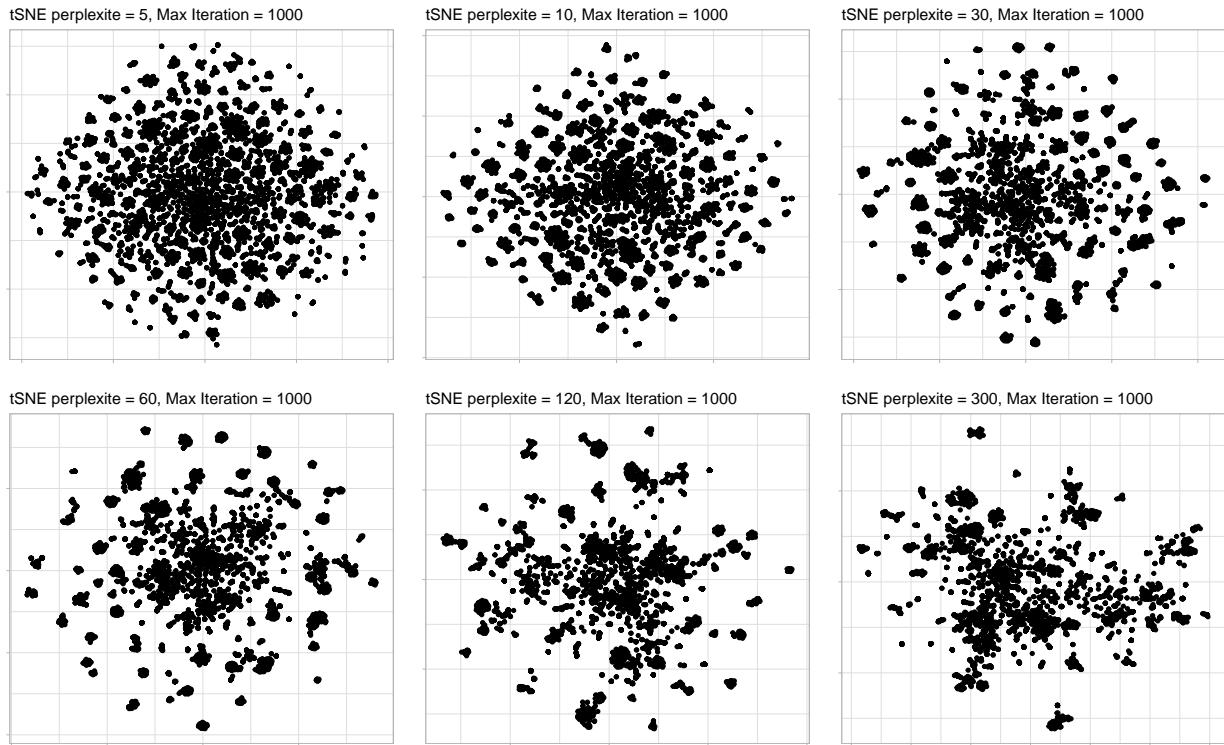
Les paramètres:

Il possède plusieurs paramètres dont les principaux auxquels nous allons nous intéresser sont la **perplexité**, le **nombre de composants principaux** à conserver et le **nombre d'itérations** pour exécuter tSNE.

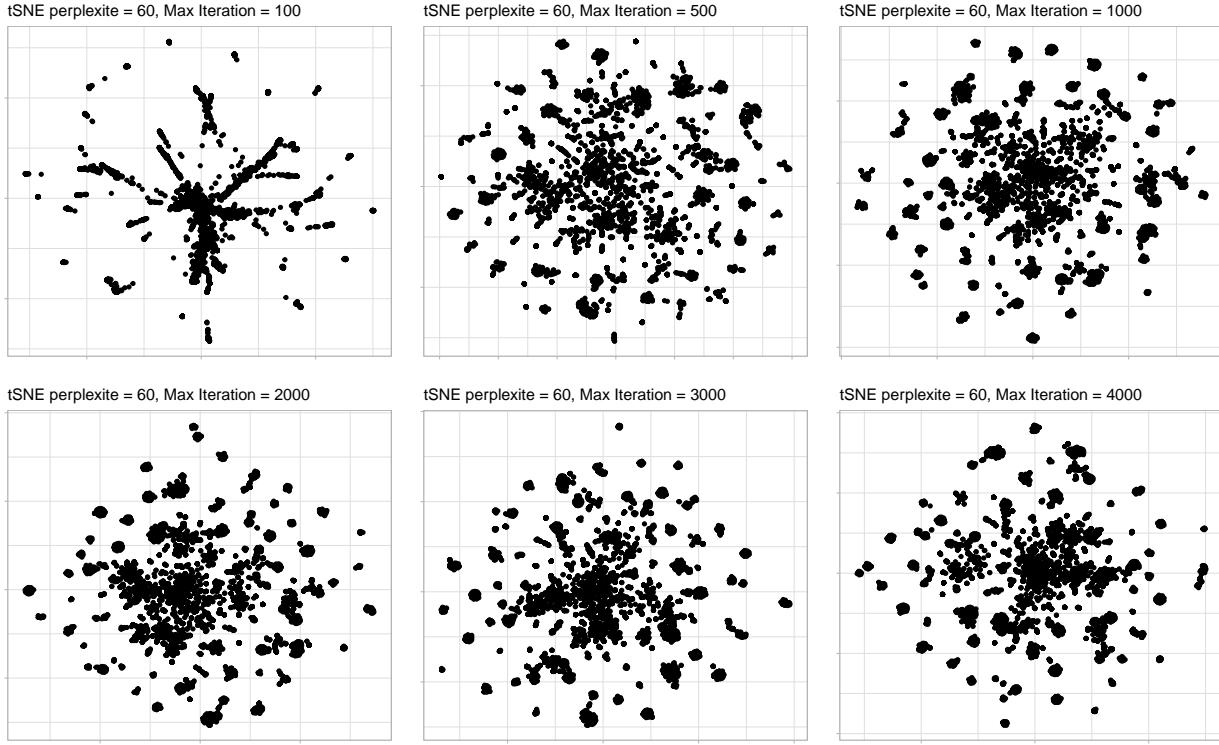
- **initial_dims** (par défaut 50) si pca = TRUE alors la variable initial_dims conditionne le nombre de dimensions ou de composants principaux.
- **perplexité** (par défaut 30) : elle représente la taille de la variance de p_{ij} c'est à dire la balance entre prise en compte de la structure globale et locale. En d'autre termes c'est une estimation du nombre de proches voisins par rapport à l'ensemble du jeu de données que possède chaque points.
- **max_iter** (1000 par défaut) : C'est le nombre d'itération qu'il faut pour obtenir un résultat stable.

Choix des paramètres

Nous allons décrire la façon dont nous avons choisi la valeur de nos paramètres perplexité et maximum d'itérations. Tout d'abord, avec un nombre de données entre 50.000 et 100.000, une perplexité entre 50-100 suffit. Il existe également un principe selon lequel la valeur de perplexité optimale est la racine du nombre de données, ici 100. Pour le maximum d'itérations, il s'avère qu'il n'y a pas techniquement un maximum tant que cette valeur est inférieure au nombre de données. Cependant, à partir d'un certains nombre d'itérations, le résultat varie peu mais le calcul prend beaucoup de temps. Par contre, si il y a pas assez d'itérations, l'algorithme n'aura pas assez de boucles pour converger vers la solution et on obtiendra alors un mauvais modèle. Nous avons donc testé plusieurs valeurs pour trouver un modèle correct, c'est à dire le meilleur compromis temps de calcul, qualité du résultat.



Dans le graphique au dessus, on fixe le maximum d'iterations et on fait varier la perplexité. On peut voir qu'avec une perplexité petite, notre modèle n'est pas bien défini : beaucoup de données ne sont pas regroupées, et les clusters sont mal définis. En revanche pour les perplexités élevées, on n'obtient pas toujours de bons modèles, certains clusters qui sont un peu alongés, parce que le tSNE n'a pas le temps de finir de s'exécuter. Finalement, une bonne perplexité ici est environ 60.



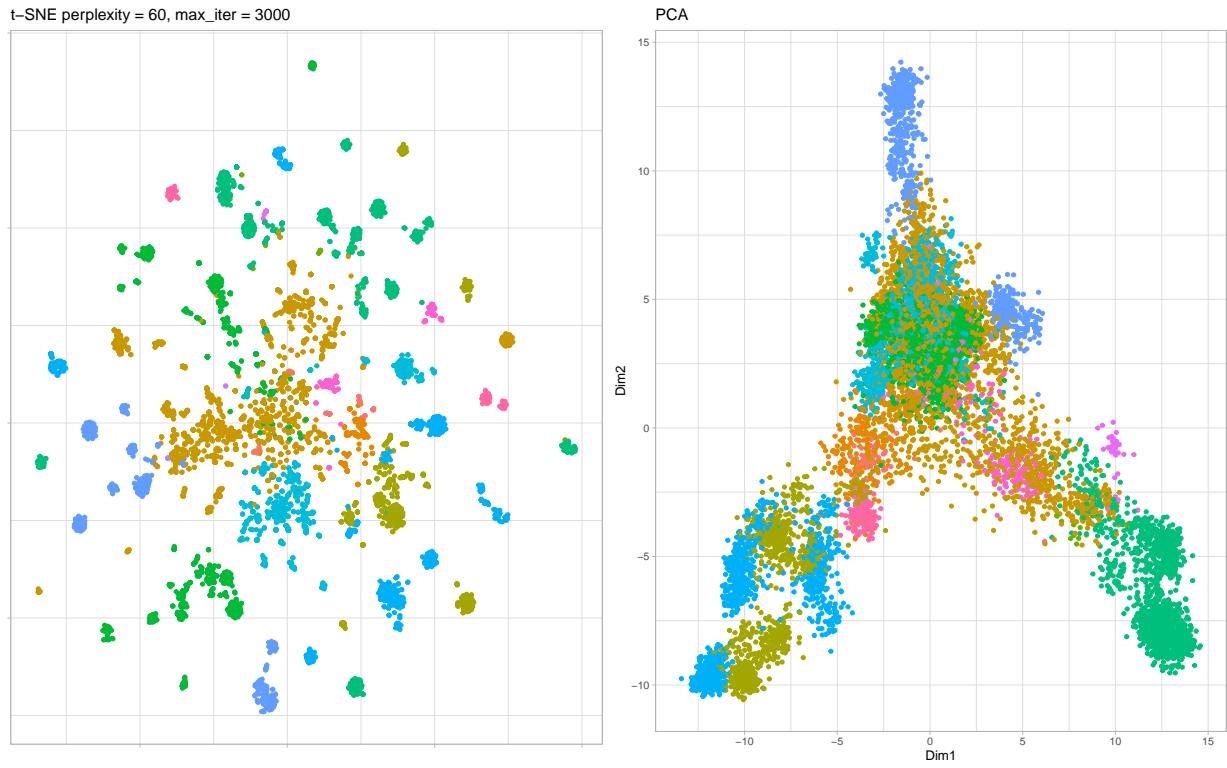
Si on fixe le perplexité et qu'on regarde l'impact du nombre maximum d'iterations, on peut voir qu'avec un nombre faible, l'algorithme n'a pas fini à les regrouper en clusters. Mais qu'après un certain seuil, ici ~ 2000 iterations, les graphiques ne changent plus beaucoup. Les graphiques s'améliorent toujours mais le temps de calcul est beaucoup plus long comme on peut voir dans le tableau ci-dessous. Finalement, les bonnes paramètres à garder sont une perplexité de 60 et un max.iteration de 3000.

Table 1: Temps de calcul en fonction de la perplexité et du nombre d'itérations

perplexity	time.in.seconds	iterations	time.in.seconds
5	83.79442	100	74.68954
10	85.10769	500	133.24716
30	98.45598	1000	149.77069
60	125.92110	2000	164.68776
120	159.99884	3000	290.15608
300	233.67175	4000	419.98032

Comparaison des projections PCA et t-SNE avec les clusters identifiés par k-means dans l'espace original

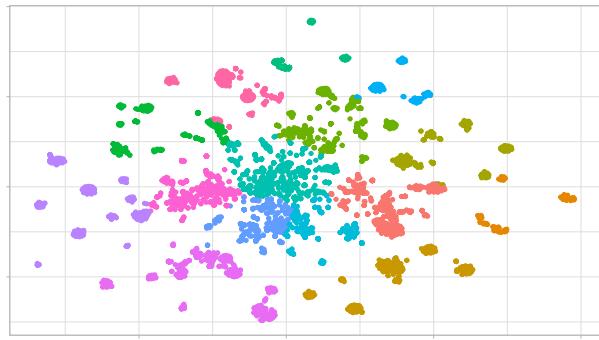
Comparaison de la répartition des clusters K-means basés sur l'espace initial dans l'espace ACP et l'espace t-SNE : Dans le graphique ci-dessous, les groupes de l'espace initial sont bien distincts dans l'espace t-SNE (peu de points rouge au sein d'une grappe de points bleus par exemple). Au contraire, dans l'espace PCA, bien qu'on puisse identifier des zones ayant des colorations dominantes, toutes les zones ne sont pas d'une seule couleur notamment à l'embranchement (le centre sur Y dessiné par l'espace PCA) où au moins trois couleurs se mêlent. On peut donc dire que, sur notre jeu de données, l'espace t-SNE offre un espace plus favorable au clustering que l'espace PCA.



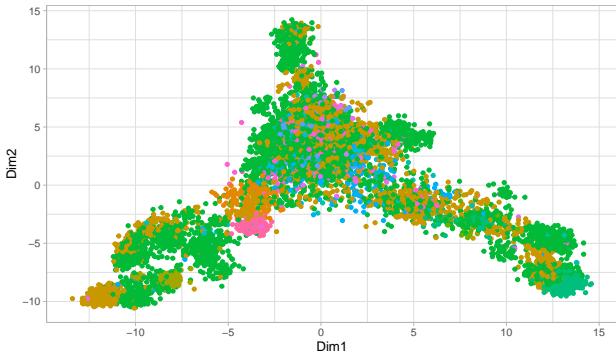
Clustering dans les espaces PCA et t-SNE

Interessons nous aux clusters déterminés par la méthode K-means sur les espaces PCA et t-SNE. Il faut noter que l'espace PCA est difficile à représenter car il est en grande dimension. Sur un plan 2D, t-SNE offre une visualisation plus clair de ses clusters.

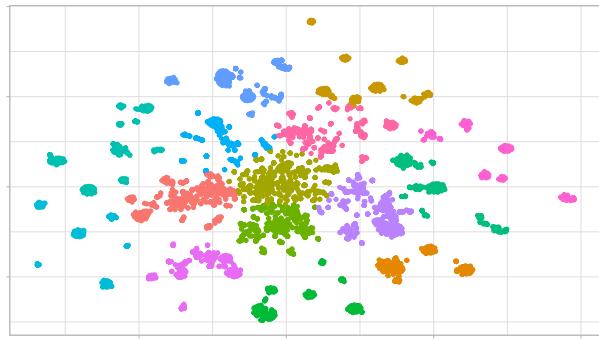
t-SNE perplexity = 60, max_iter = 3000, seed=1



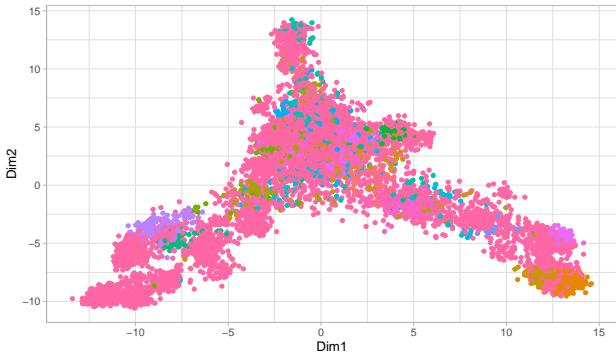
PCA, toutes dimensions, seed=1



t-SNE perplexity = 60, max_iter = 3000, seed=3

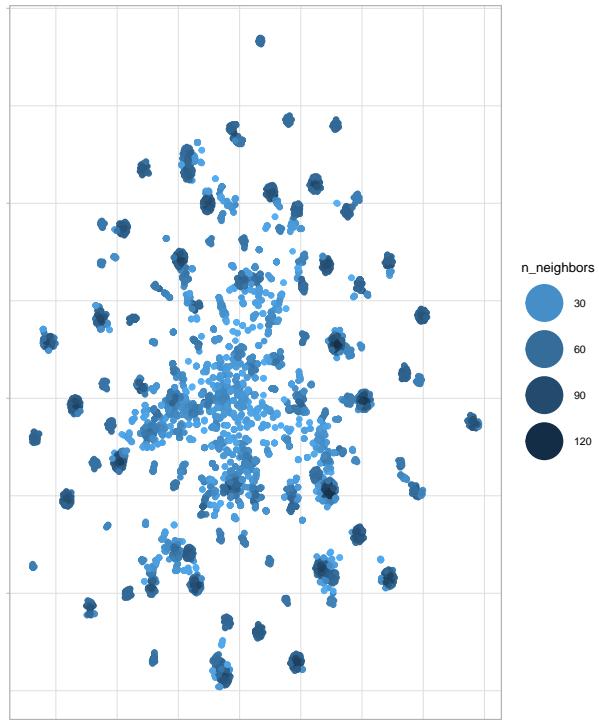


PCA, toutes dimensions, seed=3



De plus, on constate qu'en répétant l'algorithme K-means avec des initialisations différentes (set.seed), les résultats sont plus variables dans l'espace PCA que t-SNE. Il apparaît que dans l'espace PCA, dont la densité est plus homogène, les clusters sont plus aléatoires que dans l'espace t-SNE. Ce principe de densité est représenté dans la figure suivante.

tsne



pca

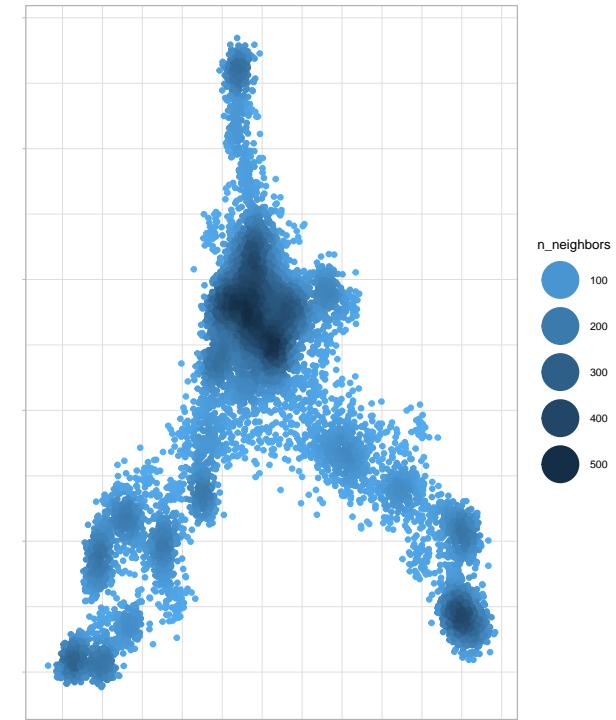
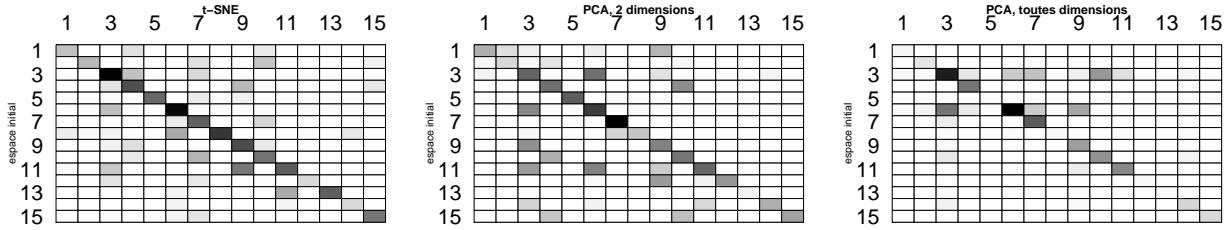


Tableau de contingence

Comparons deux à deux les clusters trouvés sur l'espace initial, l'espace réduit (PCA) et avec la méthode t-SNE. Les graphiques ci-dessous sont des illustrations du tableau de contingence. Dans le graphique PCA Vs espace initial, il y a moins de cases sombres sur la diagonale, indiquant un désaccord de classification, que dans le graphique t-SNE Vs espace initial.



Conclusion

Tout au long de notre étude, nous avons abordé non seulement la méthode t-sne qui est une réduction de dimension non-linéaire mais aussi l'ACP afin de les comparer. Ces deux méthodes ont des avantages et des inconvénients selon les paramètres (nombre de composantes, le nombre d'itération et la perplexité), et les données. Même le choix de la méthode dépend de l'application visée selon le compromis entre qualité, performance, simplicité, et taille de données à étudier, il apparaît dans notre cas que t-sne est meilleur par rapport à l'ACP.