

Reconnaissance des formes et méthodes neuronales (RCP208)

Réduction de dimension – t-SNE

Nicolas Audebert

(prenom.nom@cnam.fr)

<http://cedric.cnam.fr/vertigo/Cours/ml/>

Département Informatique
Conservatoire National des Arts & Métiers, Paris, France

26 octobre 2023

Plan du cours

- 1 Motivation
- 2 Réduction linéaire de dimension
- 3 L'algorithme LLE
- 4 L'algorithme t-SNE
- 5 L'algorithme UMAP
- 6 Justification mathématique
- 7 Plongement spectral

Problèmes en grande dimension

Les jeux de données réels sont souvent non-structurées et de grande dimension (images, sons, séries temporelles...) :

- peu adaptés à la visualisation,
- souffrent du *fléau de la dimensionalité*
 - le nombre d'échantillons nécessaires pour représenter fidèlement l'espace de dimension n est impraticable,
 - Exemple : 100 points sur le segment $[0, 1]$ \implies 10^4 points sur le carré, 10^6 points sur le cube, 10^{2n} points sur l'hypercube !
 - les dimensions bruitées réduisent le pouvoir discriminant de la distance euclidienne.
 - La distance minimale et la distance maximale entre un point de référence q et les points d'un jeu de données $\mathcal{D} = \{x_1, x_2, \dots, x_n\}$ devient la même quand $d \nearrow$:

$$\lim_{d \rightarrow +\infty} \mathbb{E} \left(\frac{\max_{x_i} \|q - x_i\| - \min_{x_i} \|q - x_i\|}{\min_{x_i} \|q - x_i\|} \right) \rightarrow 0$$

La *réduction de dimension* a deux intérêts principaux :

- *visualiser* la structure du nuage des observations,
- *simplifier* les données pour des traitements ultérieurs (par ex., classification).

Exemple pratique



- Une image est représentée par une matrice de 8×8 pixels
 - vecteur de dimension $d = 64$
 - peu visualisable

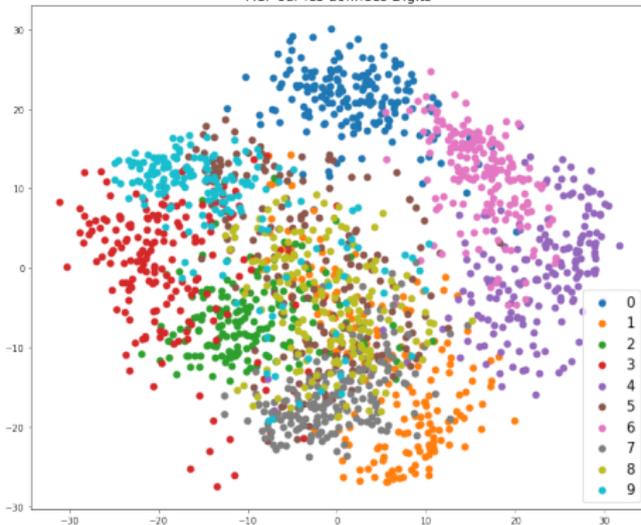
$$\begin{bmatrix} 0. & 0. & 11. & 12. & 0. & 0. & 0. & 0. \\ 0. & 2. & 16. & 16. & 16. & 13. & 0. & 0. \\ 0. & 3. & 16. & 12. & 10. & 14. & 0. & 0. \\ 0. & 1. & 16. & 1. & 12. & 15. & 0. & 0. \\ 0. & 0. & 13. & 16. & 9. & 15. & 2. & 0. \\ 0. & 0. & 0. & 3. & 0. & 9. & 11. & 0. \\ 0. & 0. & 0. & 0. & 9. & 15. & 4. & 0. \\ 0. & 0. & 9. & 12. & 13. & 3. & 0. & 0. \end{bmatrix}$$

Exemple pratique

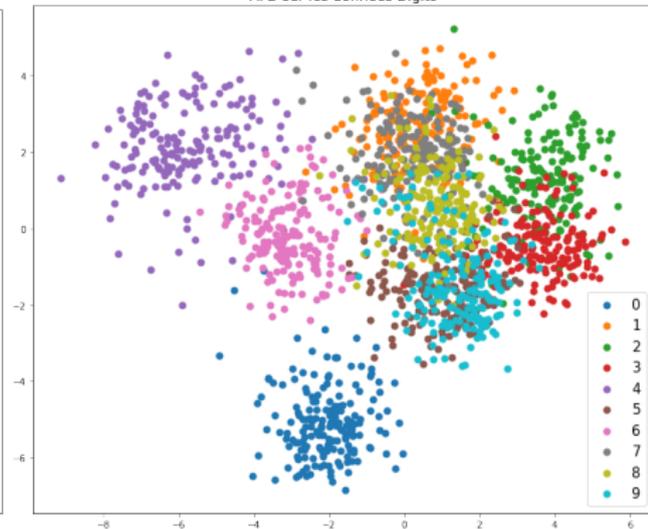


- Une image est représentée par une matrice de 8×8 pixels
 - vecteur de dimension $d = 64$
 - peu visualisable

ACP sur les données Digits



AFD sur les données Digits



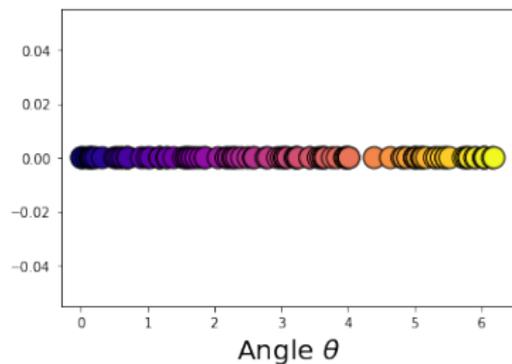
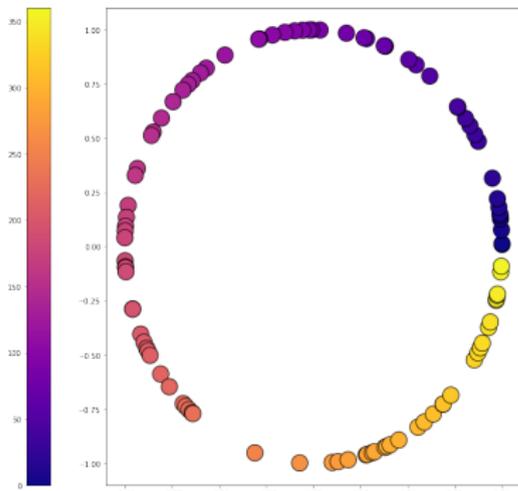
Apprentissage de variété

- Les jeux de données en grande dimension sont difficiles à visualiser
- Hypothèse : la dimension observée des données est artificiellement grande
 - Les données sont engendrées par un ensemble de variables intrinsèques (des “degrés de liberté”)
 - On suppose qu’il y a moins de degrés de liberté que de variables dans les observations
 - Autrement dit, les variables observées sont partiellement redondantes ou superflues

Apprentissage de variété : exemple

- Cercle : nuage d'observations bidimensionnel (x, y)
 - Les points vérifient la contrainte : $x^2 + y^2 = 1$
- En reparamétrisant les observations :
 - $x = \cos \theta, y = \sin \theta$

⇒ on constate que les observations sont engendrées par une variété en une dimension (le segment $[-\pi, +\pi]$)

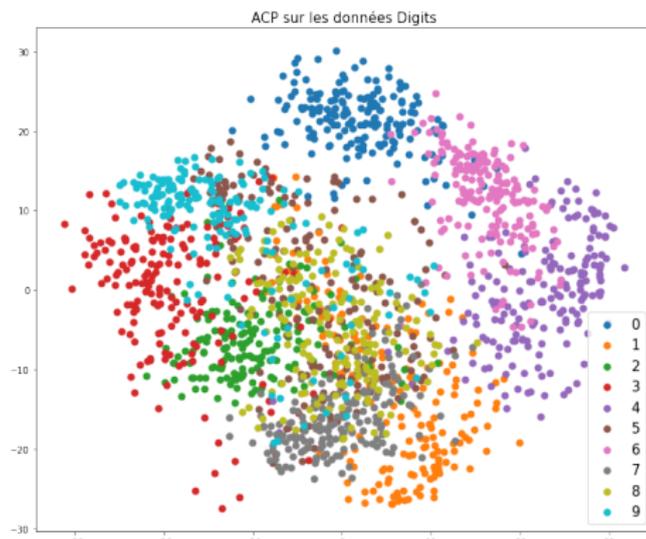


Plan du cours

- 1 Motivation
- 2 Réduction linéaire de dimension**
- 3 L'algorithme LLE
- 4 L'algorithme t-SNE
- 5 L'algorithme UMAP
- 6 Justification mathématique
- 7 Plongement spectral

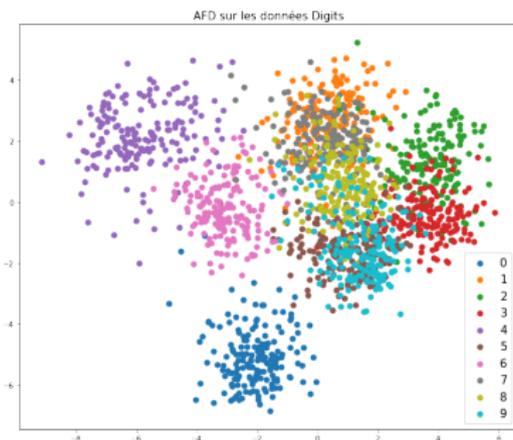
Analyse en composantes principales

- Soit R une matrice de n lignes (une par observation) et d colonnes (une par variable).
- L'analyse en composantes principales du nuage d'observations consiste à rechercher le sous-espace de dimension k engendré par les k vecteurs propres u_α associés aux k plus grandes valeurs propres λ_α de la matrice $X^T X$, avec :
 - $X = R$ pour l'ACP générale,
 - $X^T X$ la matrice des covariances empiriques pour l'ACP centrée,
 - $X^T X$ la matrice des corrélations empiriques pour l'ACP normée.



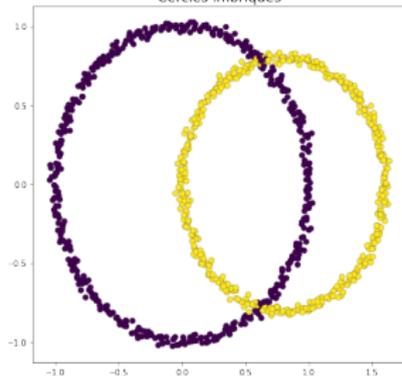
Analyse factorielle discriminante

- On suppose avoir pour chaque observation une variable "de classe" $Y \in \{1, \dots, q\}$.
- Les covariances entre variables se décomposent en deux termes : $S = E + D$
 - E est la covariance inter-classes, D est la covariance intra-classe.
- L'analyse factorielle discriminante consiste à chercher les k premiers axes discriminants : le sous-espace de dimension k engendré par les k premiers vecteurs propres généralisés u_α qui vérifient $Eu_\alpha = \lambda Su_\alpha$
- La matrice $S^{-1}E$ n'étant pas symétrique, les axes factoriels discriminants ne sont généralement pas orthogonaux.

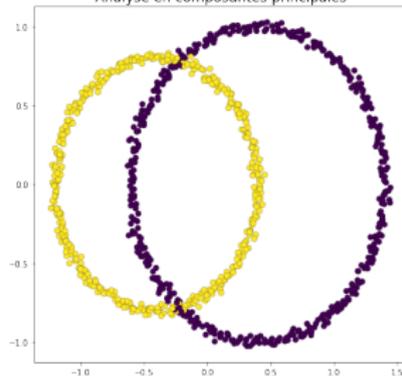


Limites de la réduction linéaire de dimension

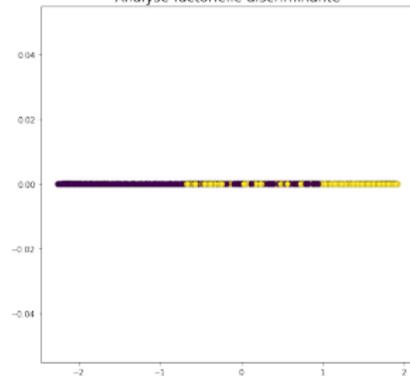
Cercles imbriqués



Analyse en composantes principales



Analyse factorielle discriminante



- Il n'existe pas de transformation linéaire permettant de séparer les deux cercles.
- Quels algorithmes pour la réduction de dimension **non-linéaire** ?

Plan du cours

- 1 Motivation
- 2 Réduction linéaire de dimension
- 3 L'algorithme LLE**
- 4 L'algorithme t-SNE
- 5 L'algorithme UMAP
- 6 Justification mathématique
- 7 Plongement spectral

Cadre formel

Considérons une matrice d'observations X de n échantillons de dimension p :

- on cherche X' de dimension $q < p$ qui représente bien la structure de X ;
- les voisins x'_j de x'_i dans l'espace réduit doivent être les mêmes que les voisins x_j de x_i dans l'espace de départ.

Locally Linear Embedding [6]

Principe : localement, le nuage d'observations X est engendré par un sous-espace affine.

1 Pour chaque point $x_i \in X \subset \mathbb{R}^p$:

- calculer les k plus proches voisins $V_i = \{x'_1, x'_2, \dots, x'_k\}$ de x_i

2 résoudre

$$\arg \min_W \sum_i \left\| x_i - \sum_{j \in V_i} w_{i,j} x_j \right\|^2 \quad \text{t.q.} \quad \sum_j w_{i,j} = 1$$

chercher les paramètres W tels que x_i peut se reconstruire par la combinaison linéaire de ses voisins

3 résoudre

$$\arg \min_Y \sum_i \left\| y_i - \sum_{j \in V_i} w_{i,j} y_j \right\|^2 \quad \text{t.q.} \quad y_i \in \mathbb{R}^q$$

chercher la matrice réduite Y tel que y_i se reconstruit par la même combinaison linéaire

Illustration

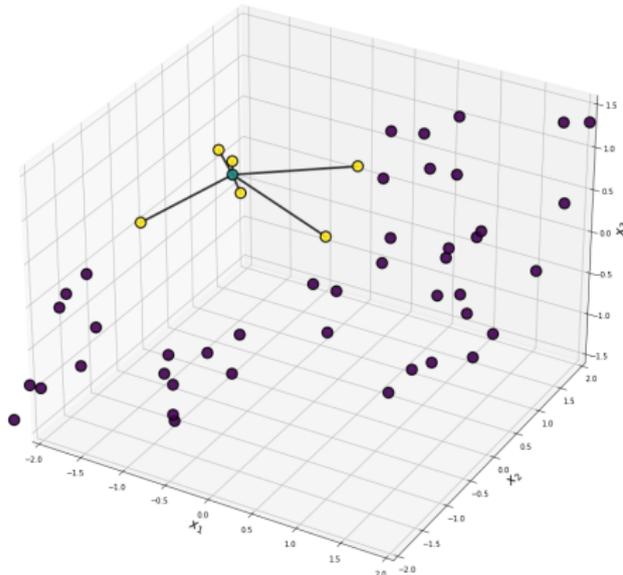
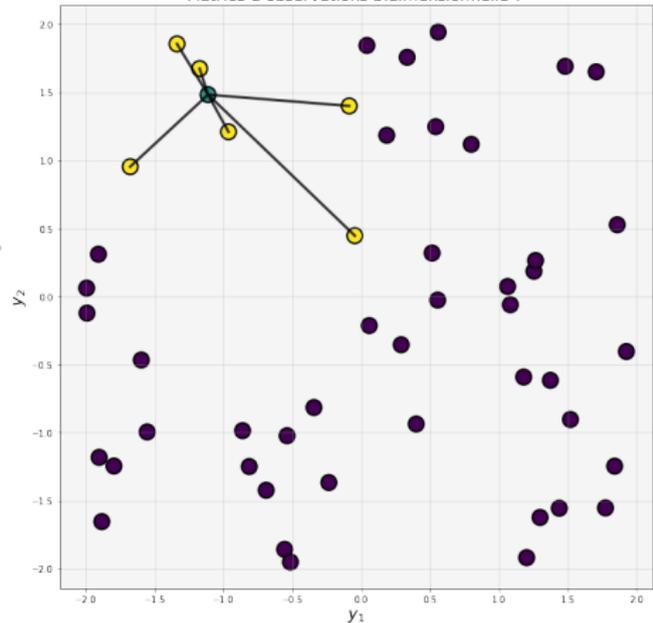
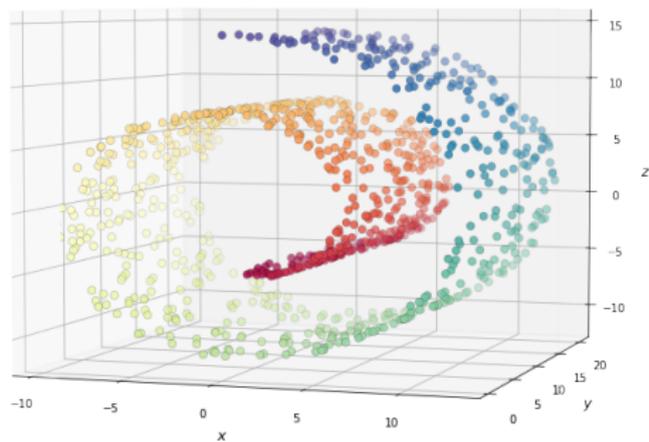
Matrice d'observations tridimensionnelle X Matrice d'observations bidimensionnelle Y 

FIG. – LLE détermine les coefficients W tels que le point bleu peut se reconstruire par combinaison linéaire de ses 6 plus proches voisins (en jaune). LLE détermine ensuite les coordonnées des points équivalents dans l'espace réduit de sorte que le point réduit correspondant se reconstruise par la même combinaison linéaire de ses voisins.

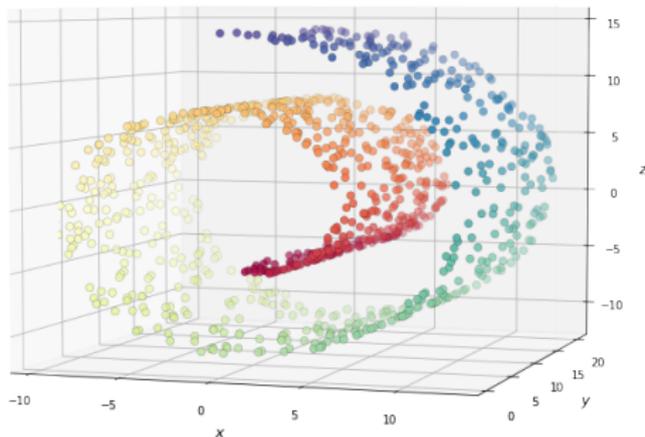
Exemple

Jeu de données "Gâteau roulé"

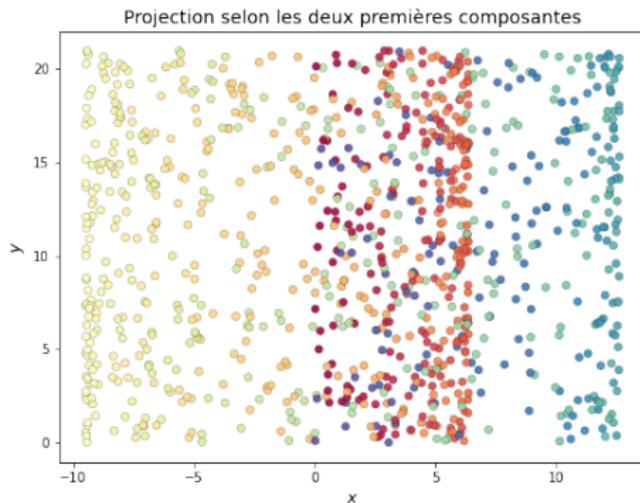


Exemple

Jeu de données "Gâteau roulé"



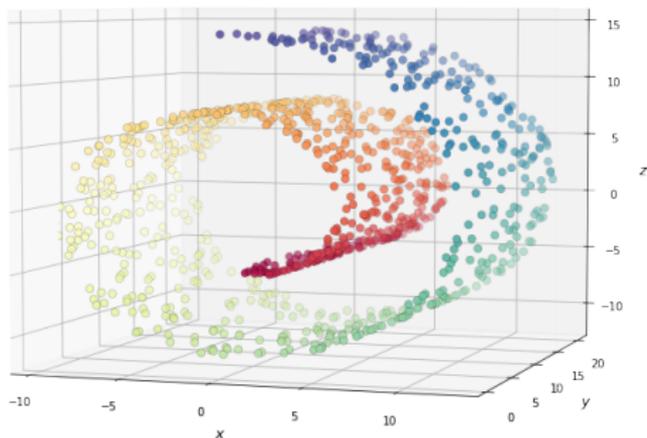
Projection aléatoire



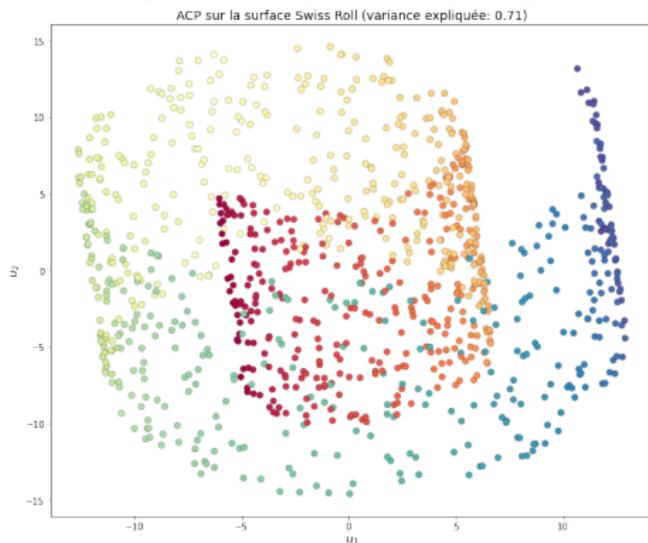
La projection selon les deux premières composantes ne rend pas compte de la structure du jeu de données : les composantes choisies sont arbitraires.

Exemple

Jeu de données "Gâteau roulé"



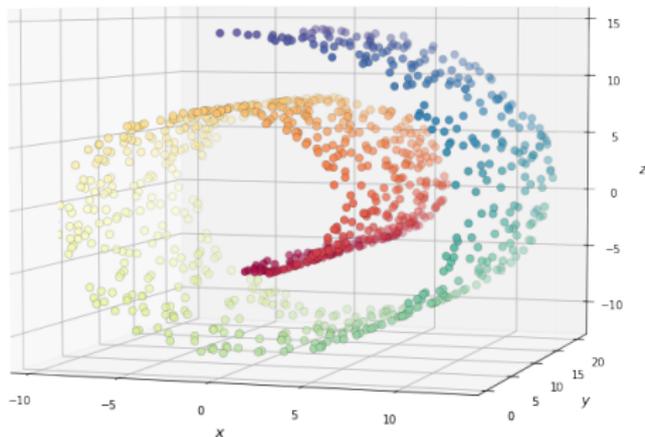
Analyse en composantes principales



L'analyse en composantes principales extrait les directions qui expliquent le plus la variance mais masque la troisième dimension.

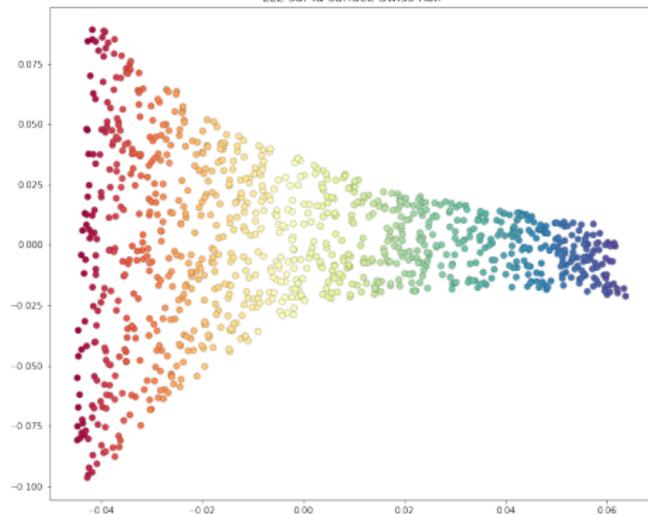
Exemple

Jeu de données "Gâteau roulé"



Locally Linear Embedding

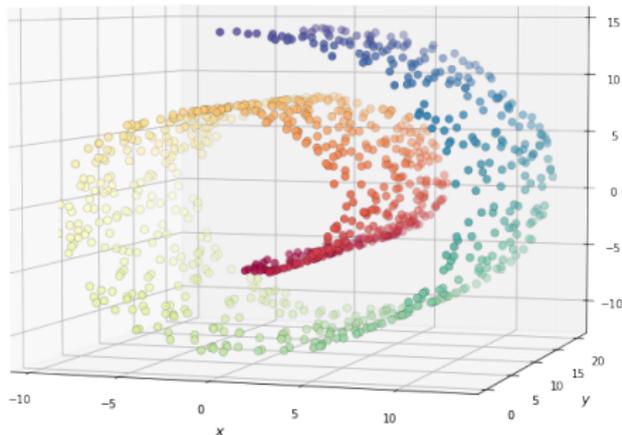
LLE sur la surface Swiss Roll



L'approche par voisinage de LLE permet de mieux respecter la structure locale et de "dérouler" la surface du jeu de données.

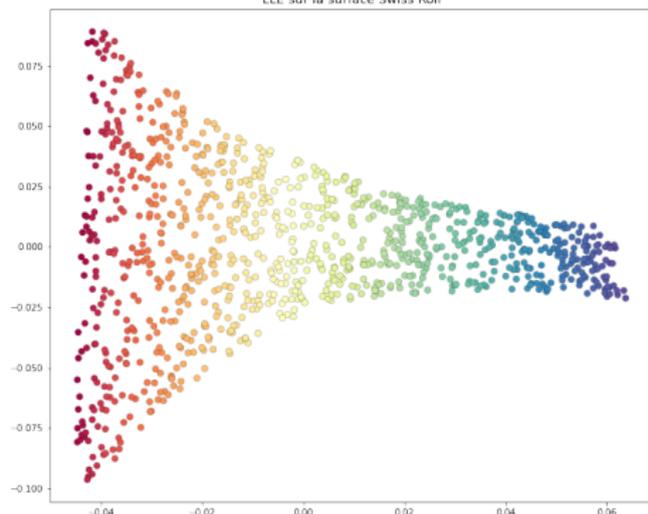
Exemple

Jeu de données "Gâteau roulé"



Locally Linear Embedding

LLE sur la surface Swiss Roll



L'approche par voisinage de LLE permet de mieux respecter la structure locale et de "dérouler" la surface du jeu de données.

Voisinages \neq distances

LLE préserve les voisinages, pas les distances. Les reconstructions sont *locales* \implies les distances entre de points éloignés n'ont pas de signification !

Plan du cours

- 1 Motivation
- 2 Réduction linéaire de dimension
- 3 L'algorithme LLE
- 4 L'algorithme t-SNE**
- 5 L'algorithme UMAP
- 6 Justification mathématique
- 7 Plongement spectral

Principe général

- *t-distributed Stochastic Neighbor Embedding* [4]
 - Algorithme de réduction de dimension non-linéaire
 - Principe général similaire à LLE : chercher un jeu de données dans un espace de dimension réduite qui présente les mêmes structures locales au voisinage de chaque point.
- Le voisinage d'un point x_i est représenté par la probabilité conditionnelle $p_{i,j} = p(x_j|x_i)$ que x_j soit considéré comme un "voisin" de x_i .
 - On cherche alors les points x'_j dans l'espace réduit pour lesquels $p'_{i,j} \simeq p_{i,j}$,
 - les distributions des voisinages sont semblables dans l'espace de départ et dans l'espace réduit.

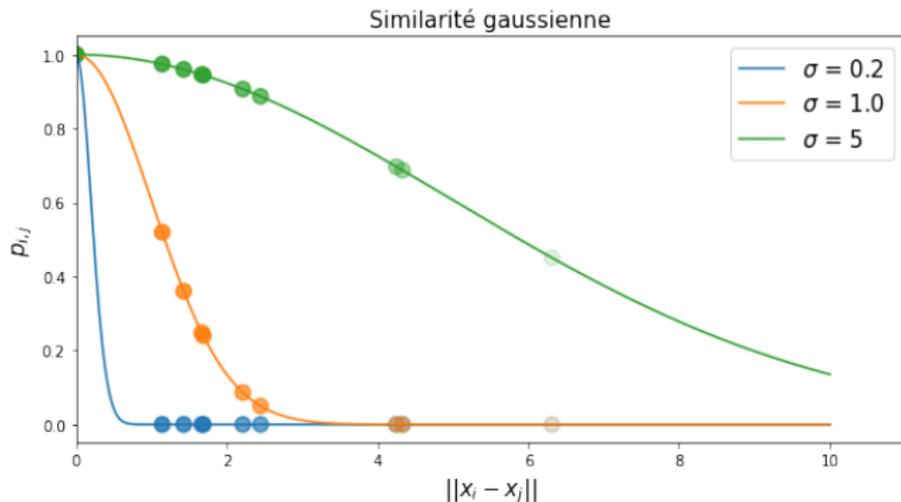
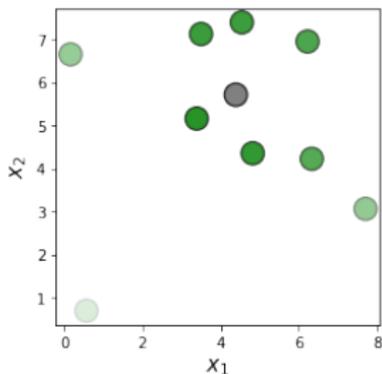
Similarité dans l'espace des observations

- Considérons une matrice d'observations X de n observations de dimension D .
- À chaque x_i , on associe la probabilité conditionnelle $p_{i,j} = p(x_j|x_i)$ définie par :

$$p_{i,j} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma^2)}{\sum_{k \neq i} \exp(-\|x_k - x_i\|^2 / 2\sigma^2)}$$

avec pour convention $p_{i,i} = 0$ de sorte à ce que $\sum_j p_{i,j} = 1$.

- L'écart-type σ est un paramètre de la méthode (la *perplexité*).



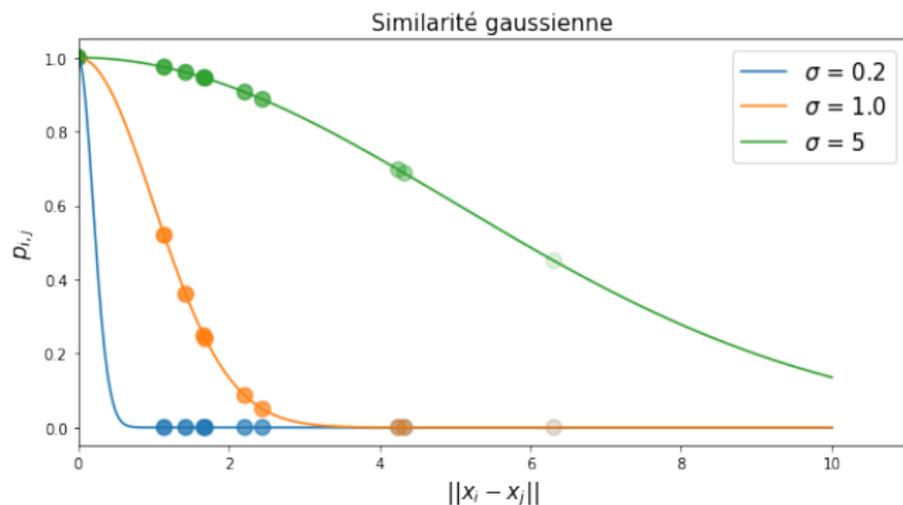
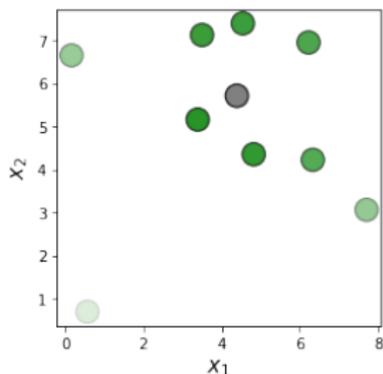
Similarité dans l'espace des observations

- Considérons une matrice d'observations X de n observations de dimension D .
- À chaque x_i , on associe la probabilité conditionnelle $p_{i,j} = p(x_j|x_i)$ définie par :

$$p_{i,j} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma^2)}{\sum_{k \neq i} \exp(-\|x_k - x_i\|^2 / 2\sigma^2)}$$

avec pour convention $p_{i,i} = 0$ de sorte à ce que $\sum_j p_{i,j} = 1$.

- L'écart-type σ est un paramètre de la méthode (la *perplexité*).



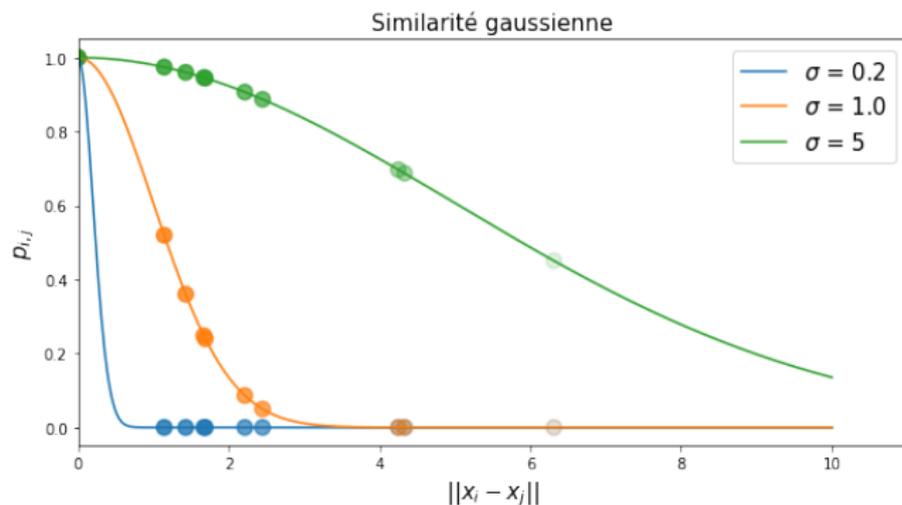
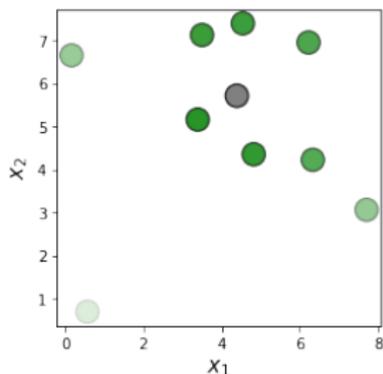
Similarité dans l'espace des observations

- Considérons une matrice d'observations X de n observations de dimension D .
- À chaque x_i , on associe la probabilité conditionnelle $p_{i,j} = p(x_j|x_i)$ définie par :

$$p_{i,j} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma^2)}{\sum_{k \neq i} \exp(-\|x_k - x_i\|^2 / 2\sigma^2)}$$

avec pour convention $p_{i,i} = 0$ de sorte à ce que $\sum_j p_{i,j} = 1$.

- L'écart-type σ est un paramètre de la méthode (la *perplexité*).



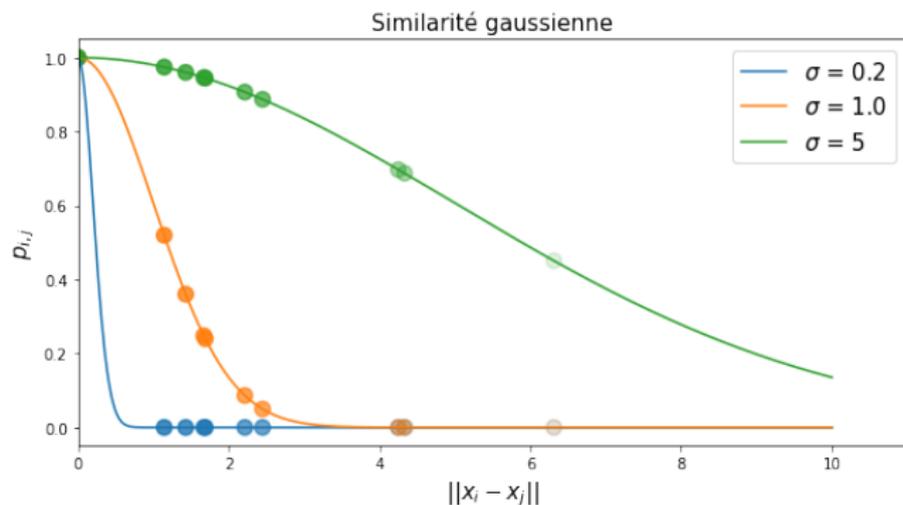
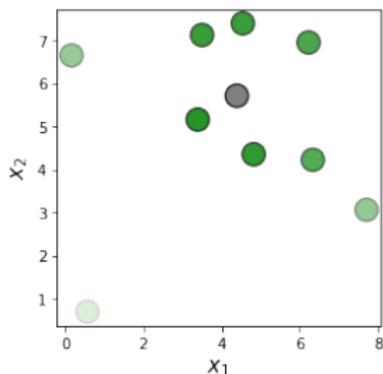
Similarité dans l'espace des observations

- Considérons une matrice d'observations X de n observations de dimension D .
- À chaque x_i , on associe la probabilité conditionnelle $p_{i,j} = p(x_j|x_i)$ définie par :

$$p_{i,j} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma^2)}{\sum_{k \neq i} \exp(-\|x_k - x_i\|^2 / 2\sigma^2)}$$

avec pour convention $p_{i,i} = 0$ de sorte à ce que $\sum_j p_{i,j} = 1$.

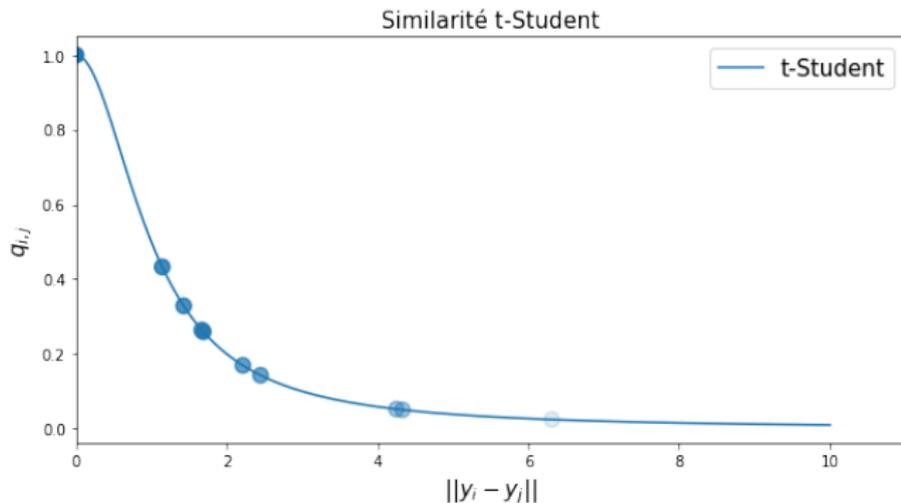
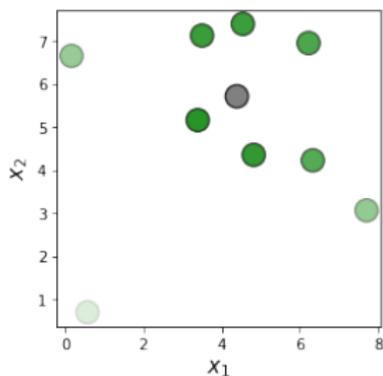
- L'écart-type σ est un paramètre de la méthode (la *perplexité*).



Similarité dans l'espace réduit

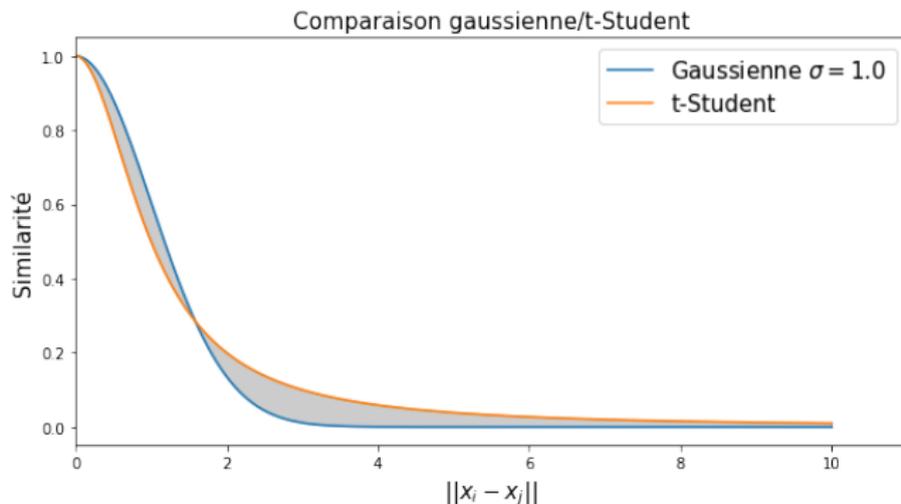
- L'objectif de t-SNE est d'obtenir une matrice réduite Y de dimension $d < D$ dont les probabilités conditionnelles $q_{i,j}$ s'approchent de $p_{i,j}$.
- Pour Y , on définit la similarité en utilisant une distribution t – Student à 1 degré de liberté :

$$q_{i,j} = \frac{\sum_{k \neq i} 1 + \|y_i - y_k\|^2}{1 + \|y_i - y_j\|^2}$$



Choix des distributions

- Similarité gaussienne : $p_{i,j} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2 / 2\sigma^2)}$
 - Décroissance rapide : les voisins et les non-voisins sont facilement distinguables
 - Queue courte : les valeurs éloignées sont toutes $\simeq 0 \rightarrow$ tous les non-voisins sont à 0
- Similarité Student : $q_{i,j} = \frac{\sum_{k \neq i} 1 + \|y_k - y_i\|^2}{1 + \|y_i - y_j\|^2}$
 - Décroissance rapide : les voisins et les non-voisins sont facilement distinguables
 - Queue longue : meilleure répartition dans l'espace, pas "d'agglutinement"



Optimisation

- Une mesure de dissimilarité entre distributions est la *divergence de Kullback-Leibler* :

$$L = KL(p|q) = \sum_{i,j} p_{i,j} \log \left(\frac{p_{i,j}}{q_{i,j}} \right)$$

- Si $L \rightarrow 0$ alors $q_{i,j} \rightarrow p_{i,j}$ (si la divergence est nulle, les distributions sont égales).

Preuve

$$KL(p|q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} = - \sum_i P(i) \log \frac{Q(i)}{P(i)}$$

- \log est une fonction concave donc, par l'inégalité de Jensen :

$$\sum_i P(i) \log \frac{Q(i)}{P(i)} \leq \log \left(\sum_i P(i) \frac{Q(i)}{P(i)} \right) = \log \sum_i Q(i) = \log(1) = 0$$

- L'égalité est vraie ssi $\frac{Q(i)}{P(i)}$ est constante, c'est-à-dire $\forall i Q(i) = \lambda P(i)$.
- Comme P et Q sont des probabilités, $\lambda = 1$ donc $KL(p|q) = 0$ ssi $\forall i P(i) = Q(i)$

Algorithme

t-SNE modifie les points y_i de sorte à minimiser L en utilisant un algorithme de descente de gradient :

1 Pour chaque paire de points (x_i, x_j) :

- Calculer la similarité $p_{i,j} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma^2)}{\sum_{k \neq l} \exp(-\|x_k - x_l\|^2 / 2\sigma^2)}$

2 Distribuer aléatoirement les points y_1, y_2, \dots, y_n dans l'espace de dimension d

3 Tant que la divergence de Kullback-Leibler $\geq \varepsilon$ ou que l'algorithme n'a pas convergé ou que le nombre maximal d'itérations n'est pas atteint :

- Déplacer chaque point y_i par :

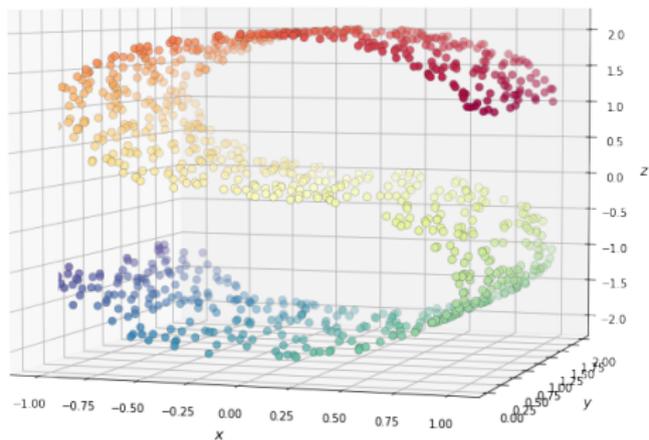
$$y_i := y_i - \alpha \frac{\partial L(p, q)}{\partial y_i}$$

- Expression analytique :

$$\frac{\partial L(p, q)}{\partial y_i} = 4 \sum_j (p_{i,j} - q_{i,j})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$$

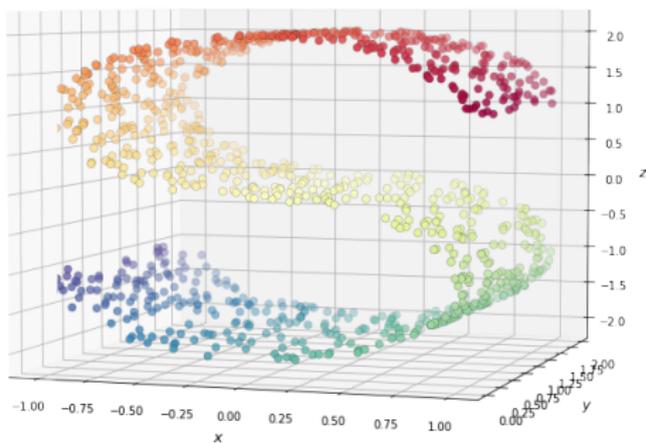
Exemple

Jeu de données "en S"



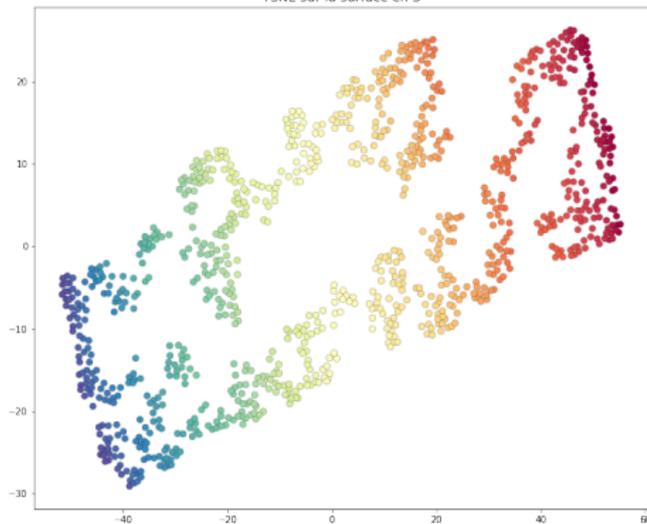
Exemple

Jeu de données “en S”



t-SNE

TSNE sur la surface en S

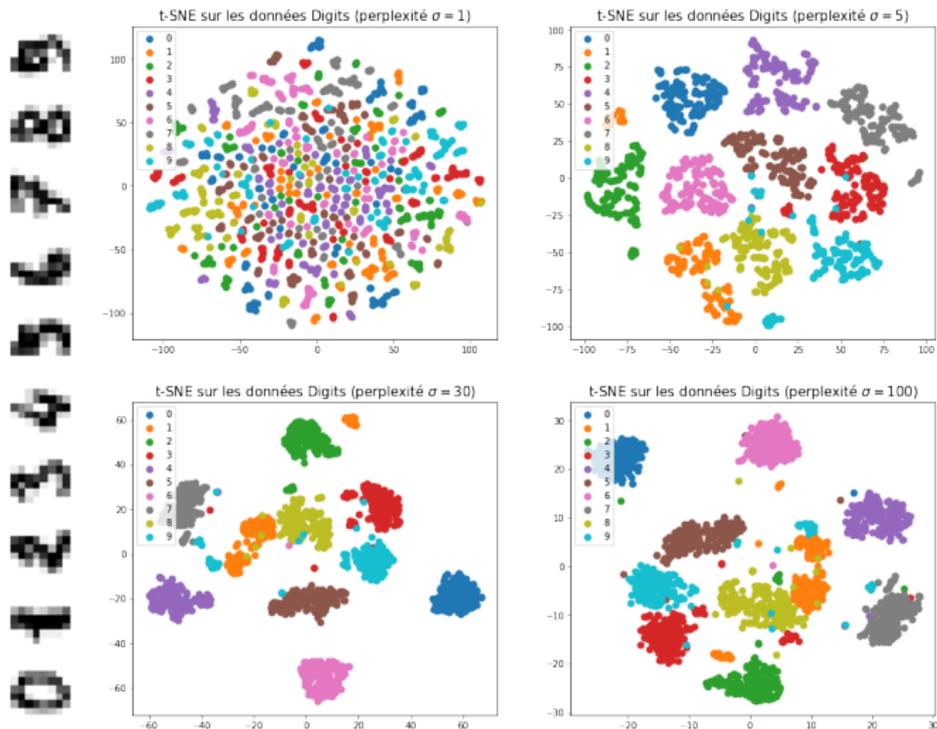


La projection par t-SNE permet de “dérouler” la surface en S.

Initialisation

- L'algorithme classique de t-SNE initialise les points y_i de façon aléatoire.
 - Stochasticité élevée
 - Faible préservation de la structure globale
 - Dépendance forte des résultats à l'initialisation
- **Solution** : appliquer une analyse en composantes principales pour le calcul des y_i^0 initiaux.
 - Préserve la structure globale
 - Meilleure reproductibilité
 - Plus robuste en pratique

Perplexité



- Perplexité faible : structure locale, seuls les voisins immédiats sont pris en compte.
- Perplexité grande : structure globale, tous les points sont tous les voisinages.

Optimisation

- La descente de gradient dans t-SNE fait intervenir deux phases :
 - **Phase 1** : *early exaggeration*
 - tous les probabilités conditionnelles dans l'espace initial sont multipliées par un facteur γ pour augmenter artificiellement la séparation des groupes naturels dans les données
 - **Phase 2** : optimisation finale
 - valeurs réelles de probabilités conditionnelles pour affiner le placement des observations localement dans les groupes
- Il existe des heuristiques pour choisir le facteur d'exagération et le pas de la descente de gradient [1] :
 - $\alpha = \max(\frac{1}{4} \frac{N}{\text{exagération}}, 50)$
 - paramètre `learning_rate='auto'` dans scikit-learn

Approximation

- t-SNE est une méthode coûteuse due à sa complexité élevée
 - Calcul du gradient : $O(dN^2)$
 - Gradient pour un seul y_i : $O(dN)$:

$$\frac{\partial L}{\partial y_i} = 4 \sum_j (p_{i,j} - q_{i,j})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$$

- Approximation de Barnes-Hut [3]
 - Calcul du gradient : $O(dN \log N)$
 - Seulement pour la visualisation (espace réduit 2D ou 3D)

Limites

- Sensibilité aux paramètres et stochasticité de la méthode :
 - t-SNE est une méthode stochastique : deux répétitions ne donnent pas nécessairement la même projection
 - L'initialisation a un impact significatif sur le résultat final
 - Les paramètres d'optimisation (exagération, pas d'apprentissage et surtout perplexité) peuvent grandement modifier la projection finale.
- Interprétabilité sujette à caution [9]
 - La taille des groupes dans t-SNE n'a pas de signification.
 - La distance entre les groupes n'a pas (toujours) de signification.
 - Avec une perplexité faible, le bruit dans les données peut relier des artificiellement groupes ou au contraire les surdiviser.
- Inconvénients pratiques
 - Méthode transductive non-paramétrique : impossible de projeter une nouvelle observation, il faut refaire l'optimisation.
 - Non-inversible : impossible de retrouver l'antécédent dans l'espace des observations d'un point arbitraire de l'espace réduit.

Plan du cours

- 1 Motivation
- 2 Réduction linéaire de dimension
- 3 L'algorithme LLE
- 4 L'algorithme t-SNE
- 5 L'algorithme UMAP**
- 6 Justification mathématique
- 7 Plongement spectral

Principe général de UMAP

- *Uniform Manifold Approximation & Projection* (UMAP) [5]
 - Algorithme de réduction non-linéaire de dimension similaire à t-SNE ;
 - Paquet `umap-learn` en Python (interface compatible avec scikit-learn).
- Idée générale : trouver une représentation des données en plus faible dimension qui a **la même topologie** que le nuage des observations dans l'espace de départ.
 - 1 Définir une matrice de similarités S appropriée ;
 - 2 À partir de S , construire un graphe de similarités ;
 - 3 Construire une représentation vectorielle des données dans un espace de dimension inférieure qui présente le même graphe similarités.
- Différences majeures avec t-SNE
 - Utiliser une notion de perplexité variable pour définir une similarité *locale* qui dépend de la densité autour de chaque point ;
 - Considérer une famille de fonctions de similarité plus large inspirée de la loi t de Student pour l'espace d'arrivée.

Similarité adaptative

- UMAP définit une similarité adaptative qui varie selon la densité locale des données :

$$p_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2 - \rho_i}{\sigma_i}\right)$$

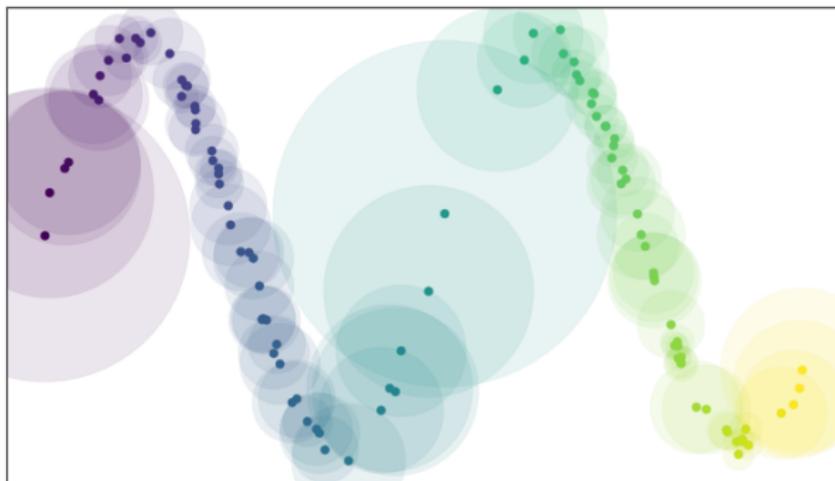


Figure de https://umap-learn.readthedocs.io/en/latest/how_umap_works.html

Similarité adaptative

- UMAP définit une similarité adaptative qui varie selon la densité locale des données :

$$p_{i|j} = \exp\left(-\frac{\|x_i - x_j\|^2 - \rho_i}{\sigma_i}\right)$$

- ρ_i est la distance entre x_i et son plus proche voisin
- noyau gaussien adaptatif avec une normalisation locale sur la perplexité
 - σ_i est fixé de sorte à ce que $0 \leq p_{i|j} \leq 1$, c'est-à-dire :

$$\sum_{j=1}^k \exp\left(\frac{-\max(0, \|x_i - x_j\| - \rho_i)}{\sigma_i}\right) = \log_2(k)$$

- k est un paramètre réglant le nombre de plus proches voisins à considérer.
- Pour deux points x_i et x_j , la probabilité jointe d'être dans un même voisinage doit être symétrique, donc :

$$p_{i,j} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i}$$

- la symétrisation est nécessaire car $\rho_i \neq \rho_j$;
- la symétrisation de t-SNE aurait aussi été envisageable : $p_{i,j} = \frac{p_{i|j} + p_{j|i}}{2N}$

Similarité dans l'espace d'arrivée

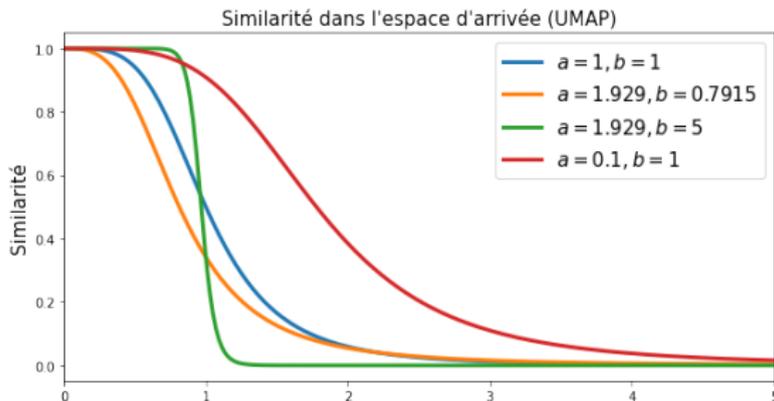
- La similarité dans l'espace d'arrivée est une variante de la t -Student :

$$q_{i,j} = \frac{1}{1 + a(y_i - y_j)^{2b}}$$

- une queue plus épaisse pour éviter le problème d'agglutinement dans l'espace cible ;
- le choix des paramètres a et b est fait automatiquement de sorte à ce que la similarité soit la plus proche possible de la fonction définie par morceaux :

$$\left(1 + a(y_i - y_j)^{2b}\right)^{-1} = \begin{cases} 1 & \text{si } y_i - y_j \leq \text{min_dist} \\ e^{-(y_i - y_j)} & \text{si } y_i - y_j > \text{min_dist} \end{cases}$$

avec min_dist la distance minimale autorisée dans l'espace d'arrivée entre deux points.



Optimisation

- Comme pour t-SNE, l'objectif est de faire tendre $q_{ij} \rightarrow p_{ij}$
- La fonction objectif à minimiser est l'entropie croisée entre les deux distributions :

$$CE(X, Y) = \sum_{i=1}^N \sum_{j=1}^N \left[p_{ij} \log \left(\frac{p_{ij}}{q_{ij}} \right) + (1 - p_{ij}) \log \left(\frac{1 - p_{ij}}{1 - q_{ij}} \right) \right]$$

- $CE(p, q) = H(p) + D_{KL}(p||q)$
 - $H(p)$ est l'entropie de Shannon de la distribution p ;
 - Comme $H(p) \geq 0$, $CE(X, Y) \geq D_{KL}(p||q)$ et minimiser l'entropie croisée revient à minimiser la divergence de Kullback-Leibler et donc à faire tendre $q_{ij} \rightarrow p_{ij}$.
- UMAP optimise la fonction objectif en réalisant une descente de gradient stochastique :
 - Mise à jour des points dans l'espace d'arrivée : $y_i := y_i - \alpha \frac{\partial CE}{\partial y_i}$
 - α correspond au pas d'apprentissage de l'algorithme de descente de gradient.
 - Pour accélérer le calcul de CE sur un grand jeu de données, on ne considère qu'un échantillon limité pour en calculer une valeur approchée.
 - L'échantillon contient un mélange de voisins ($p_{ij} \simeq 1$) et de non-voisins ($p_{i,j} \simeq 0$) de sorte à optimiser les deux termes de l'entropie croisée simultanément.

Initialisation des points dans l'espace d'arrivée

- Comme pour t-SNE, l'initialisation des points dans l'espace d'arrivée peut grandement influencer sur le résultat de UMAP.
- Dans le cas de t-SNE avec l'*early exaggeration*, il a été observé que la première phase de l'algorithme revient approximativement à réaliser un plongement spectral [2].
 - L'initialisation par défaut de UMAP consiste donc à réaliser un plongement spectral.
 - Moins d'aléatoire au moment de l'initialisation.
 - Convergence plus rapide de la descente de gradient stochastique.

Algorithme

Données : nuage de points $\mathcal{E} = \{x_i\}_{1 \leq i \leq N}$, nombre de plus proches voisins k , dimension de l'espace d'arrivée d , paramètres (a, b) de la similarité q_{ij}

Sortie : nuage de points réduit $\mathcal{F} = \{y_i\}_{1 \leq i \leq N}$

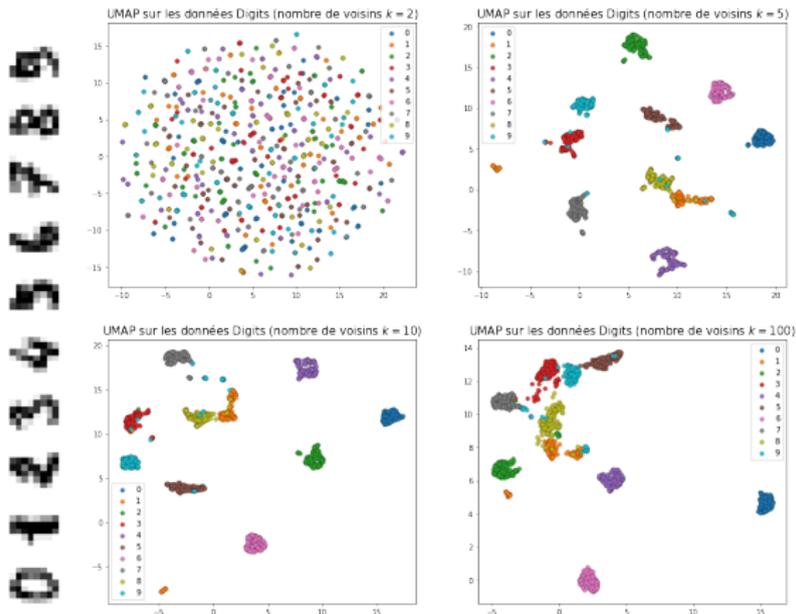
- 1 Déterminer pour chaque point x_i ses k plus proches voisins
- 2 Calculer $p_{ij}(X)$ pour toutes les paires (i, j)
- 3 Initialiser les points $\mathcal{F} = \{y_i\}$ à l'aide d'un plongement spectral dans \mathbb{R}^d
- 4 Tant que l'entropie croisée $CE \geq \varepsilon$ ou que l'algorithme n'a pas convergé :
 - Tirer aléatoirement un échantillon uniforme de m observations x_i
 - Calculer q_{ij} pour les points de l'échantillon
 - Calculer $CE(p, q)$ sur cet échantillon
 - Déplacer chaque point y_i par :

$$y_i := y_i - \alpha \frac{\partial CE}{\partial y_i}$$

- Répéter tant que le critère de convergence n'est pas atteint

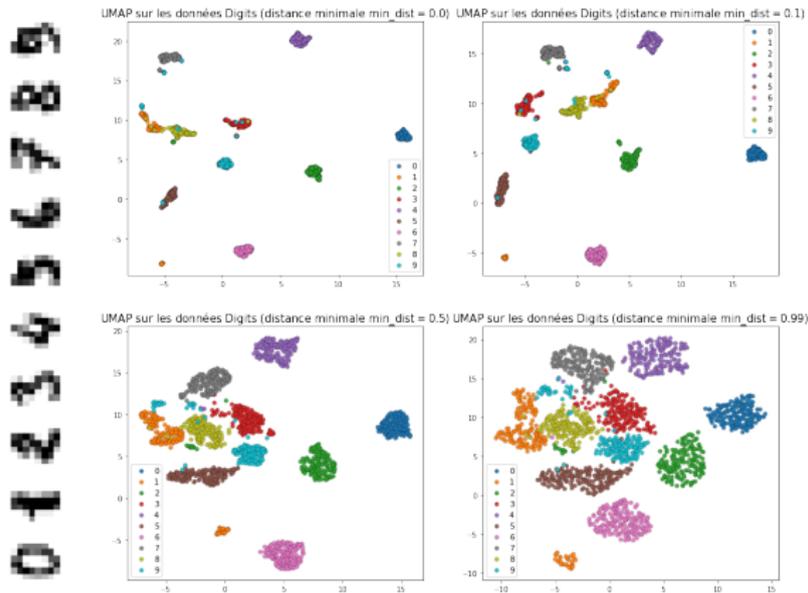
Paramétrage

- Principaux paramètres de l'algorithme UMAP
 - 'n_components' : la dimension de l'espace (euclidien) d'arrivée.
 - 'n_neighbours' : le nombre de voisins à considérer pour la définition de la similarité adaptative (définit implicitement les valeurs de a et b).
 - 'min_dist' : la distance minimale autorisée entre deux points dans l'espace d'arrivée.



Paramétrage

- Principaux paramètres de l'algorithme UMAP
 - 'n_components' : la dimension de l'espace (euclidien) d'arrivée.
 - 'n_neighbours' : le nombre de voisins à considérer pour la définition de la similarité adaptative (définit implicitement les valeurs de a et b).
 - 'min_dist' : la distance minimale autorisée entre deux points dans l'espace d'arrivée.



UMAP (semi-)supervisé

- Considérons un jeu de données contenant des **étiquettes de groupes** :
 $\mathcal{X} = \{(x_1, l_1), \dots, (x_n, l_n)\}$ avec x_i les observations et l_i les étiquettes (variables catégorielles)
 - Comment inclure la connaissance des étiquettes dans l'algorithme UMAP ?
⇒ définir une distance mixte sur le couple (x_i, l_i)
- La distance sur les étiquettes des groupes est définie comme étant :

$$d^{\text{classe}}(x_i, x_j) = \begin{cases} 0 & \text{si } l_i = l_j \\ 0.5 & \text{si } l_i \text{ ou } l_j \text{ est inconnu} \\ 1 & \text{si } l_i \neq l_j \end{cases}$$

- La distance totale impliquée dans le calcul des similarités entre points dans l'espace de départ s'obtient par une somme pondérée par un paramètre λ (`target_weight`) :

$$d(x_i, x_j) = (1 - \lambda)d^{\text{obs}}(x_i, x_j) + \lambda d^{\text{classe}}(x_i, x_j)$$

- $\lambda = 0$ → seule la distance entre les données x_i est prise en compte
- $\lambda = 0.5$ → la distance totale est une moyenne de la distance entre les observations et entre les étiquettes
- $\lambda = 1$ → seule la distance entre les étiquettes est prise en compte

Limites

- Projection paramétrique de nouvelles données
 - Comme t-SNE, UMAP est une méthode transductive et il n'y a pas d'expression analytique de la transformation permettant de projeter un point de l'espace de départ vers l'espace d'arrivée;
 - Il est possible d'approcher cette transformation à l'aide d'un modèle de régression, par exemple un réseau de neurones (perceptron multi-couche).
 - ParametricUMAP dans le paquet `umap-learn`.
- Interprétabilité sujette à caution
 - Mêmes précautions que t-SNE sur l'absence de signification des densités des groupes ou des distances inter-groupes.
- Réglage des paramètres de l'algorithme
 - Le nombre de voisin `n_neighbors` de UMAP est moins sensible que la perplexité de t-SNE aux grandes valeurs;
 - Il est important de comparer plusieurs visualisations avec des paramètres différents pour observer les structures locales et la structure globale.

Validation des méthodes de réduction de dimension

- Comment vérifier la pertinence d'une projection obtenue ?
 - Pas de critère simple d'interprétation comme pour les méthodes factorielles.
- Solution 1 : performances des algorithmes de k plus proches voisins
 - A minima, les algorithmes de réduction non-linéaire de dimension doivent conserver les **relations de voisinage**.
 - Calculer la précision d'un algorithme de kNN en classification sur $\mathcal{E} = \{x_i, 1 \leq i \leq N\}$ dans l'espace de départ.
 - Calculer la précision d'un kNN en classification sur $\mathcal{F} = \{y_i, 1 \leq i \leq N\}$ dans l'espace d'arrivée.
 - Les performances dans l'espace réduit devraient être équivalents ou supérieures à celles dans l'espace de départ.
 - Inconvénient : nécessite d'avoir des étiquettes de groupes sur les observations.
- Solution 2 : critères spécifiques [7]
 - La réduction de dimension doit conserver l'ordonnancement des voisins :
 - *Trustworthiness* :

$$T(k) = 1 - \frac{2}{nk(2n - 3k - 1)} \sum_{i=1}^n \sum_{j \in U_i^{(k)}} (r(i, j) - k)$$

- avec $U_i^{(k)}$ les k plus proches voisins de i ,
- et $r(i, j)$ le rang de j parmi les voisins de i .

Plan du cours

- 1 Motivation
- 2 Réduction linéaire de dimension
- 3 L'algorithme LLE
- 4 L'algorithme t-SNE
- 5 L'algorithme UMAP
- 6 Justification mathématique**
- 7 Plongement spectral

Éléments de topologie : le simplexe

- Soit $E = \mathbb{R}^k$ un espace vectoriel de dimension k .
- Un **simplexe** de dimension k est défini comme étant l'enveloppe convexe de $k + 1$ points $\{v_1, v_2, \dots, v_{k+1}\} \subset E$.
 - Le simplexe est un objet multi-dimensionnel à $k + 1$ sommets et $k + 1$ côtés (ou faces).
- Exemples :
 - $k = 0$: un point ;
 - $k = 1$: un segment (deux points) ;
 - $k = 2$: un triangle (trois points) ;
 - $k = 3$: un tétraèdre (quatre points) ;
 - etc.



0-simplex



1-simplex



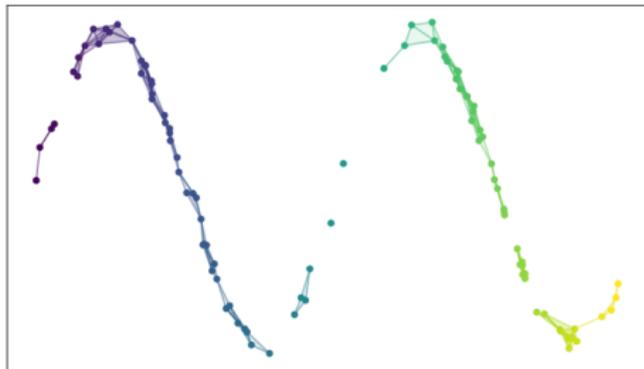
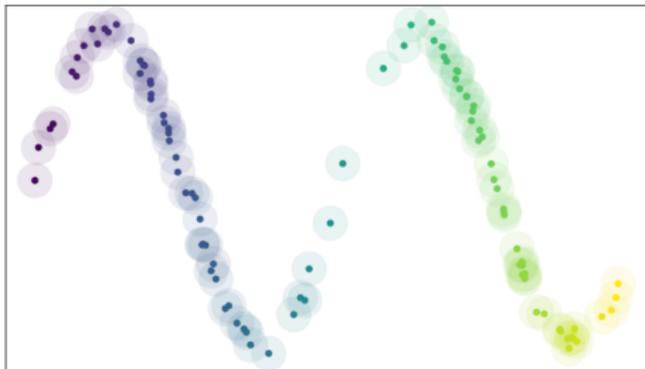
2-simplex



3-simplex

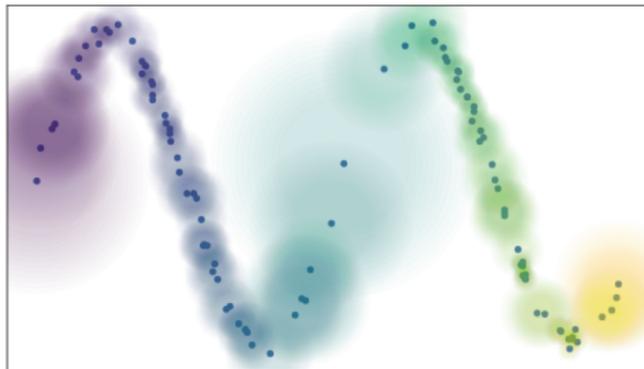
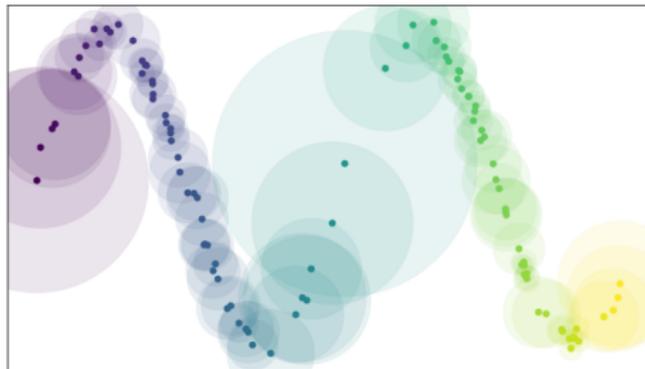
Éléments de topologie : le complexe simplicial

- Un complexe simplicial \mathcal{K} de E est défini comme un ensemble de simplexes reliés par au moins une face, c'est-à-dire :
 - $\mathcal{K} = \{S_1, S_2, \dots, S_m\}$
 - $\forall 1 \leq i \leq m$, S_i est un k -simplexe de E
 - $\forall 1 \leq i \leq m, \exists 1 \leq j \leq m, j \neq i$ tel que S_i et S_j partagent une face en commun.
- Pour estimer la topologie d'un nuage de points $\mathcal{E} = \{x_i, 1 \leq i \leq n\} \subset \mathbb{R}^k$, il est intéressant d'examiner le complexe simplicial qui "couvre" tous les points :
 - Chaque point x_i est relié à ses voisins dans une boule ouverte de rayon r ;
 - Si deux boules ouvertes s'intersectent (= si x_i et x_j sont voisins), on place un 1-simplexe reliant x_i et x_j ;
 - Si k boules ouvertes s'intersectent (= si $x_{i_1}, x_{i_2}, \dots, x_{i_k}$ sont voisins), on place un k -simplexe reliant tous ces points.



Estimation de la topologie d'un nuage de points

- Le choix du rayon d de la boule ouverte est difficile.
 - Si d est fixe, alors les boules sont uniformes ce qui suppose que les données sont uniformément réparties (ce qui est faux en général sur des données réelles).
 - Si d est variable, comment savoir quelle est la valeur appropriée pour un x_i donné ?
- Une solution partielle : construire un complexe simplicial "flou", c'est-à-dire un graphe pondéré.
 - Chaque arête dispose d'un poids $w_{ij} \leftarrow$ peut s'obtenir à partir de la notion de similarité
 - Mais les similarités peuvent-elles être uniformes sur les données ?
 - t-SNE : oui, la perplexité définit la forme de la fonction de similarité.
 - UMAP : non, la métrique locale normalisée par la distance au k^e voisin \rightarrow boules de taille variable.

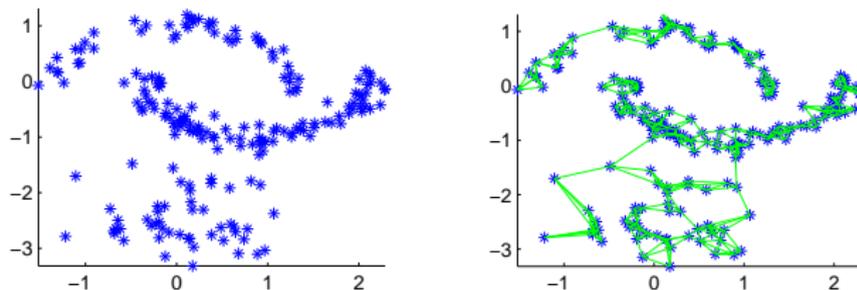


Plan du cours

- 1 Motivation
- 2 Réduction linéaire de dimension
- 3 L'algorithme LLE
- 4 L'algorithme t-SNE
- 5 L'algorithme UMAP
- 6 Justification mathématique
- 7 Plongement spectral**

Principe du plongement spectral

- Ensemble $\mathcal{E} = \{\mathbf{x}_i, 1 \leq i \leq N\}$ de N données de dimension D décrites par une matrice de **similarités** \mathbf{S} telle que l'élément $s_{ij} \geq 0$ soit la similarité entre \mathbf{x}_i et \mathbf{x}_j
- Objectif : répartir des données $\mathcal{F} = \{\mathbf{y}_i, 1 \leq i \leq N\}$ dans un espace de dimension $d < D$ tels que les distances $\|\mathbf{y}_i - \mathbf{y}_j\|$ soient faibles si les similarités s_{ij} sont fortes et inversement.
- Approche :
 - 1 À partir de \mathbf{S} construire un graphe de similarités
 - 2 À l'aide du graphe, construire une représentation vectorielle « convenable » des données



(figure issue de [8])

Graphes : quelques définitions

- Graphe non orienté $G = (V, E)$, $V = \{v_1, \dots, v_N\}$ étant l'ensemble des N sommets et E l'ensemble des arêtes
- Arêtes pondérées : soit $w_{ij} \geq 0$ le poids de l'arête qui lie les sommets v_i et v_j
 - $w_{ij} = 0 \Leftrightarrow$ absence d'arête entre v_i et v_j
 - Matrice des poids : \mathbf{W} d'élément générique w_{ij}
- Degré du sommet v_i : $d_i = \sum_{j=1}^N w_{ij}$
 - Matrice diagonale des degrés : \mathbf{D} avec $\{d_1, \dots, d_N\}$ sur la diagonale
- Chaîne :
 - Un chemin (une chaîne) $\mu(v_i, v_j)$ entre les sommets v_i et v_j est une suite d'arêtes de E permettant de relier les deux sommets
- Vecteur indicateur d'un sous-ensemble $C \subset V$: vecteur \mathbf{c} à N composantes binaires tel que $c_i = 1$ si le sommet $v_i \in C$ et $c_i = 0$ sinon
- Composante connexe :
 - $C \subset V$ forme une composante connexe si $\forall v_i, v_j \in C$ il existe une chaîne $\mu(v_i, v_j)$ les reliant et $\forall v_i \in C, \forall \mu(v_i, v_p), v_p \in C$

Matrice laplacienne d'un graphe

- Pour un graphe non orienté à arêtes pondérées, la matrice laplacienne (non normalisée) est $\mathbf{L} = \mathbf{D} - \mathbf{W}$
- Propriétés :
 - $\forall \mathbf{x} \in \mathbb{R}^N, \mathbf{x}^T \mathbf{L} \mathbf{x} = \frac{1}{2} \sum_{i,j=1}^N w_{ij} (x_i - x_j)^2$
 - \mathbf{L} est symétrique et semi-définie positive $\Rightarrow \mathbf{L}$ a N valeurs propres réelles $0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_N$ (valeurs propres λ et vecteurs propres $\mathbf{u} : \mathbf{L} \mathbf{u} = \lambda \mathbf{u}$)
 - La valeur propre la plus faible de \mathbf{L} est 0, le vecteur propre correspondant étant le vecteur constant $\mathbf{1}$; justification : $d_i = \sum_{j=1}^N w_{ij}$, donc $d_i + \sum_{j=1}^N (-w_{ij}) = 0$
- \mathbf{L} et les composantes connexes de G :
 - La multiplicité de la valeur propre 0 de \mathbf{L} est égale au nombre de composantes connexes de G
 - Les vecteurs indicateurs de ces composantes connexes sont des vecteurs propres de \mathbf{L} pour la valeur propre 0
- Matrices laplaciennes normalisées :
 - $\mathbf{L}_{rw} = \mathbf{D}^{-1} \mathbf{L} (= \mathbf{I}_N - \mathbf{D}^{-1} \mathbf{W})$
 - $\mathbf{L}_{sym} = \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2} (= \mathbf{I}_N - \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2})$

Plongement spectral : un algorithme

Données : matrice de similarités \mathbf{S} , $N \times N$

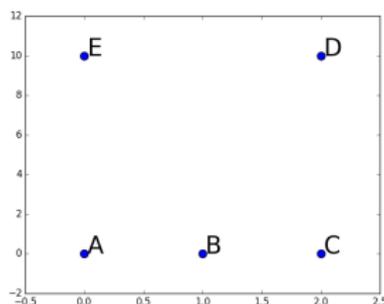
Résultat : observations réduites $\mathcal{F} = \{y_i, 1 \leq i \leq N\}$

- 1 Construire le graphe de similarités G à partir de \mathbf{S} ($w_{ij} = s_{ij}$)
- 2 Calculer la matrice laplacienne \mathbf{L}
- 3 Calculer les k vecteurs propres $\mathbf{u}_1, \dots, \mathbf{u}_k$ associés aux k plus petites valeurs propres de $\mathbf{L}\mathbf{u} = \lambda\mathbf{D}\mathbf{u}$
- 4 Soit \mathbf{U}_k la matrice $N \times k$ dont les colonnes sont les vecteurs $\mathbf{u}_1, \dots, \mathbf{u}_k$
- 5 $\mathbf{y}_i \in \mathbb{R}^k, 1 \leq i \leq N$ sont les N lignes de la matrice \mathbf{U}_k .

Exemple simple : données bidimensionnelles

■ Exemple

$$\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 2 & 0 \\ 2 & 10 \\ 0 & 10 \end{bmatrix}$$



- Matrices des similarités obtenues à partir du noyau gaussien de variance 1 appliqué à la distance euclidienne (valeurs arrondies à 6 décimales)

$$\begin{bmatrix} 1.000000 & 0.367879 & 0.018316 & 0.000000 & 0.000000 \\ 0.367879 & 1.000000 & 0.367879 & 0.000000 & 0.000000 \\ 0.018316 & 0.367879 & 1.000000 & 0.000000 & 0.000000 \\ 0.000000 & 0.000000 & 0.000000 & 1.000000 & 0.018316 \\ 0.000000 & 0.000000 & 0.000000 & 0.018316 & 1.000000 \end{bmatrix}$$

Exemple simple : matrice laplacienne

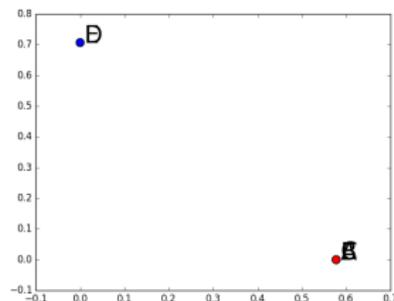
- Matrices laplaciennes normalisées L_{rw} (valeurs arrondies à 6 décimales)

$$\begin{bmatrix} 1.000000 & -0.952574 & -0.047426 & 0.000000 & 0.000000 \\ -0.500000 & 1.000000 & -0.500000 & 0.000000 & 0.000000 \\ -0.047426 & -0.952574 & 1.000000 & 0.000000 & 0.000000 \\ 0.000000 & 0.000000 & 0.000000 & 1.000000 & -1.000000 \\ 0.000000 & 0.000000 & 0.000000 & -1.000000 & 1.000000 \end{bmatrix}$$

- 2 composantes connexes, valeur propre 0.0 de L_{rw} de multiplicité 2
 - Valeurs propres de L_{rw} en ordre croissant : **0.0**, **0.0**, 1.0474, 1.9525, 2.0000

- Matrice U :

$$\begin{bmatrix} 0.577350 & 0.000000 \\ 0.577350 & 0.000000 \\ 0.577350 & 0.000000 \\ 0.000000 & 0.707107 \\ 0.000000 & 0.707107 \end{bmatrix}$$



De distance à similarité

- Comment définir une mesure de **similarité** adaptée aux données de $\mathcal{E} = \{\mathbf{x}_i, 1 \leq i \leq N\}$?
 - Une compréhension des données est très utile pour choisir une bonne mesure de similarité
 - Possible de définir des mesures de similarité à partir de fonctions-noyaux (par ex. loi normale multidimensionnelle si les données sont vectorielles)
 - Ce qui compte surtout c'est le comportement de la mesure de similarité pour des données très similaires (les données très dissimilaires ne se retrouveront pas dans le même groupe)
- Souvent on dispose des **distances** entre les données
 - Utilisation de la loi normale pour définir les similarités : $s_{ij} = \exp\left(-\frac{d(\mathbf{x}_i, \mathbf{x}_j)^2}{2\sigma^2}\right)$
 - difficulté à choisir σ lorsque des groupes différents ont des « densités » très différentes, surtout vu que la loi normale est à décroissance rapide
 - $\sigma \sim$ arête la plus longue dans l'arbre couvrant minimal des données de \mathcal{E}

Choix d'un type de graphe de similarités

■ Passage d'une matrice de similarités S à un graphe G :

1 ϵ -voisinage : $v_i, v_j \in V$ connectés si $s_{ij} \geq \epsilon$

- Difficulté à choisir ϵ lorsque des groupes différents ont des « densités » très différentes (distances très différentes aux plus proches voisins dans le groupe)

2 k plus proches voisins (*k-nearest neighbors*, *kNN*, *kppv*) : $v_i, v_j \in V$ connectés si v_i fait partie des k ppv de v_j **ou** v_j fait partie des k ppv de v_i

- Solution adaptée à la présence de groupes de « densités » différentes mais tendance à créer des liens entre groupes
- Robustesse constatée au choix de la valeur du paramètre (k)

→ solution conseillée, du moins en première approche

3 k plus proches voisins mutuels (*mutual k-nearest neighbors*) : $v_i, v_j \in V$ connectés si v_i fait partie des k ppv de v_j **et** v_j fait partie des k ppv de v_i

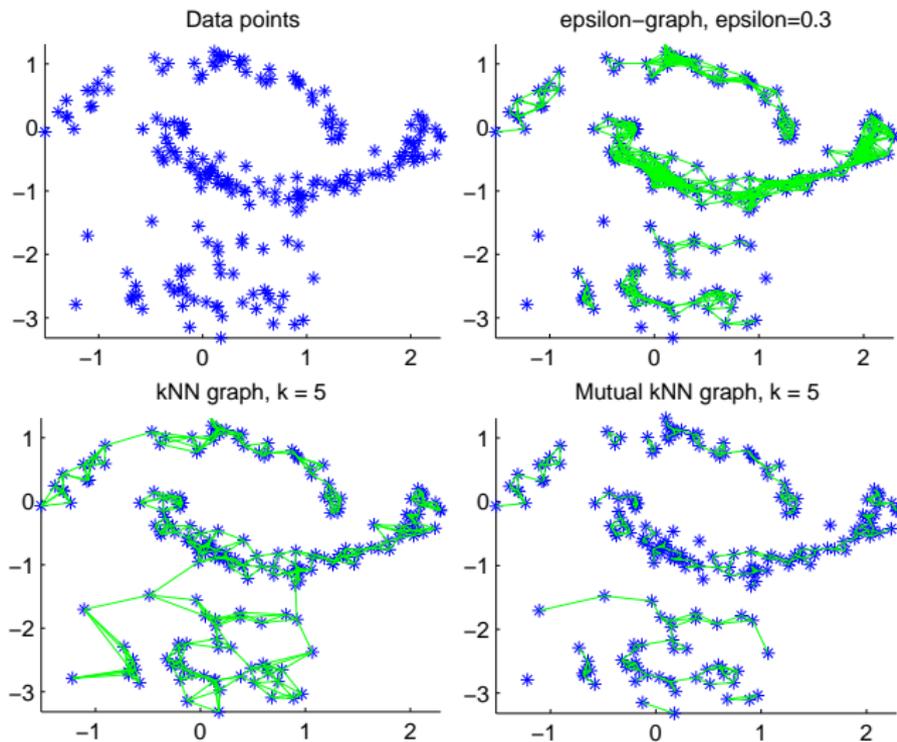
- Solution adaptée à la présence de groupes de « densités » différentes, absence de liens entre groupes de densités différentes (grâce au « **et** »)

4 Graphe complet : tous les sommets sont connectés

- Dépendance totale de la mesure de similarité pour partitionner

■ Plus le graphe est connecté, plus le coût des étapes ultérieures est élevé

Choix d'un type de graphe de similarités (2)



(figure issue de [8])

Références I



A. C. Belkina, C. O. Ciccolella, R. Anno, R. Halpert, J. Spidlen, and J. E. Snyder-Cappione.

Automated optimized parameters for T-distributed stochastic neighbor embedding improve visualization and analysis of large datasets.

Nature Communications, 10(1) :5415, Nov. 2019.



G. C. Linderman and S. Steinerberger.

Clustering with t-sne, provably.

SIAM Journal on Mathematics of Data Science, 1(2) :313–332, Jan 2019.



L. v. d. Maaten.

Accelerating t-SNE using Tree-Based Algorithms.

Journal of Machine Learning Research, 15(93) :3221–3245, 2014.



L. v. d. Maaten and G. Hinton.

Visualizing Data using t-SNE.

Journal of Machine Learning Research, 9(86) :2579–2605, 2008.



L. McInnes, J. Healy, N. Saul, and L. Großberger.

Umap : Uniform manifold approximation and projection.

Journal of Open Source Software, 3(29) :861, 2018.

Références II



S. T. Roweis and L. K. Saul.

Nonlinear Dimensionality Reduction by Locally Linear Embedding.

Science, 290(5500) :2323–2326, Dec. 2000.



S. Stasis, R. Stables, and J. Hockman.

Semantically controlled adaptive equalisation in reduced dimensionality parameter space.

Applied Sciences, 6(44) :116, Apr 2016.



U. von Luxburg.

A tutorial on spectral clustering.

CoRR, abs/0711.0189, 2007.



M. Wattenberg, F. Viégas, and I. Johnson.

How to Use t-SNE Effectively.

Distill, 1(10), Oct. 2016.