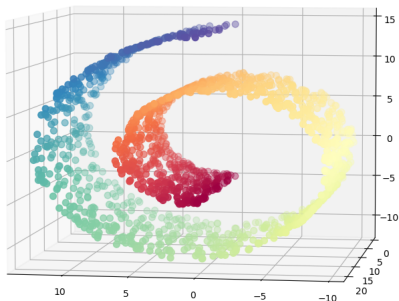


# UMAP : Uniform Manifold Approximation and Projection

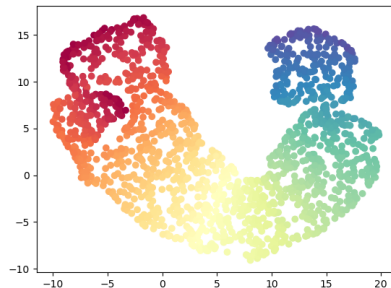
Youness et Ivanhoé

28 octobre 2023

# Introduction



**Figure** – Représentation 3D du swiss roll dataset.



**Figure** – UMAP projection ( $n\_neighbors=15$ ,  $min\_dist = 0.1$ ).

**Problématique :** Comment fonctionne l'algorithme UMAP et pourquoi utiliser cette méthode sur un jeu de données ?

1 Les intérêts de la méthode UMAP sur un jeu de données

2 Présentation théorique de la méthode UMAP

- Matrice de probabilités jointes sur l'espace de départ
- Similarité entre les individus de l'espace d'arrivé
- Fonction d'entropie croisée et minimisation

3 Utiliser UMAP au quotidien et ses résultats

4 Conclusion et faiblesses de UMAP

# Les intérêts de la méthode UMAP sur un jeu de données

- Méthode de *réduction de dimension* utile pour la visualisation des données.
- Projection dans un espace de plus petite dimension pour des données non-linéaires donc non-séparables par un sous-espace vectoriel.
- Retransmettre dans un espace de plus petite dimension la structure topologique des données initiales.
- Utile pour faire du clustering et peut servir pour entraîner des modèles de machine learning.

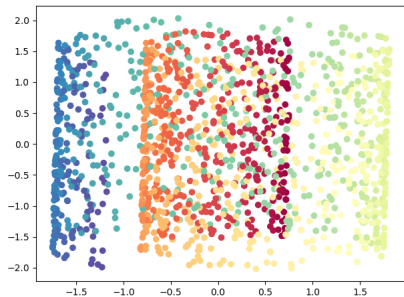


Figure – Scatter plot de l'ACP du dataset swiss roll avec les deux premières composantes principales

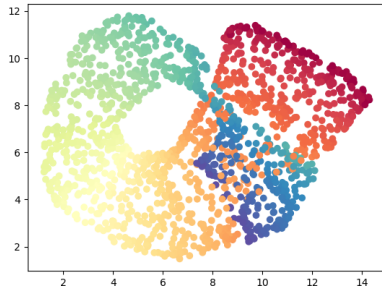


Figure – UMAP projection du nuage de points dans un plan. ( $n\_neighbors=500$ ,  $min\_dist = 0.5$ )

## Remarque

*L'algorithme de UMAP ressemble à celui de T-SNE. Il est présenté comme plus rapide et pouvant s'adapter à une réduction de la dimension  $d \geq 2$ . UMAP préserve aussi davantage la structure globale du jeu de données par rapport à T-SNE.*

## Idée :

Pour un jeu de données  $X \in \mathcal{M}_{n,p}(\mathbb{R})$  nous cherchons à préserver les relations locales entre les individus dans un espace de dimension plus petit ( $d \lll p$ ). Cela revient à trouver  $Y \in \mathcal{M}_{n,d}(\mathbb{R})$  sous certaines contraintes.

## Les étapes importantes :

- 1 Pour un nombre de voisins  $k$  fixé, donner pour tout couple d'individus  $(x_i, x_j)$  une probabilité  $p_{i,j}$  d'appartenir à un même voisinage.
- 2 Initialiser la matrice  $Y$  et en déduire la similarité entre les couples d'individus  $(y_i, y_j)$  que l'on note  $q_{i,j}$ .
- 3 Minimiser sur l'espace d'arrivée une fonction objectif d'entropie croisée qui mesure la dissimilarité entre les deux distributions  $p$  et  $q$ . Ici  $q$  est une fonction de  $Y$ .

# Matrice de probabilités jointes sur l'espace de départ

- 1 Fixons  $k \geq 2$  le nombre de voisins à considérer. Soit  $x_i, x_j$  deux individus, la similarité  $p_{i|j}$  entre deux observations est définie par :

$$p_{i|j} = \exp \left( - \frac{\|x_i - x_j\|_2 - \rho_i}{\sigma_i} \right)$$

- On définit  $\rho_i = \min_{i \neq j} \|x_i - x_j\|$ .
- L'individu  $i$  est l'un des  $k$  voisins de l'individu  $i$  et on définit  $\sigma_i$  de

telle sorte que 
$$\sum_{j=1}^k \exp \left( - \frac{\max(0, \|x_i - x_j\|_2 - \rho_i)}{\sigma_i} \right) = \log_2(k)$$

- 2 Symétrisation pour obtenir la probabilité d'appartenir à un même voisinage :

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



# Similarité entre les individus de l'espace d'arrivé

Fixons une distance minimale "*min\_dist*".

- 1 Nous aimerions définir la similarité entre deux individus  $y_i, y_j$  de la manière suivante :

$$q_{i,j} = \begin{cases} 1 & \text{si } \|y_i - y_j\| \leq \text{min\_dist} \\ \exp(-\|y_i - y_j\|) & \text{si } \|y_i - y_j\| > \text{min\_dist} \end{cases}$$

- 2 Pour cela on s'inspire d'une loi de Student et on pose :

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

L'algorithme UMAP optimise les deux paramètres ( $a, b$ ) pour coller au mieux à la fonction précédente qui est définie par morceaux.

# Fonction d'entropie croisée et minimisation

**But :** trouver la distribution des points  $y_i$  dans l'espace d'arrivée telle que les similarités  $q_{i,j}$  s'approchent le plus possible des similarités  $p_{i,j}$ .

- Une fonction d'entropie croisée liée à la divergence de Kullback-Leiber :

$$\begin{aligned} CE(y_1, \dots, y_n) &= CE(p, q) = \sum_{i=1}^n \sum_{j=1}^n p_{i,j} \ln \left( \frac{p_{i,j}}{q_{i,j}} \right) + (1 - p_{i,j}) \ln \left( \frac{(1 - p_{i,j})}{(1 - q_{i,j})} \right) \\ &= KL(p, q) + KL(1 - p, 1 - q) \end{aligned}$$

- Minimiser la fonction d'entropie croisée à l'aide d'une descente de gradient stochastique :

$$y_i := y_i - \alpha \nabla_i CE(y_1, \dots, y_i, \dots, y_n)$$

Pour la descente de Gradient Stochastique une seule observation est utilisée pour calculer le gradient à chaque itération et celle-ci est choisie de manière aléatoire.

# Utiliser UMAP au quotidien et ses résultats

- Installer umap-learn et le charger.

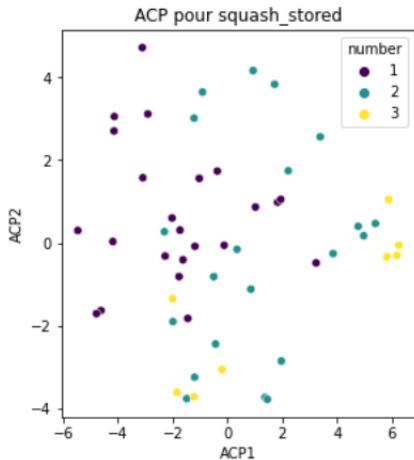
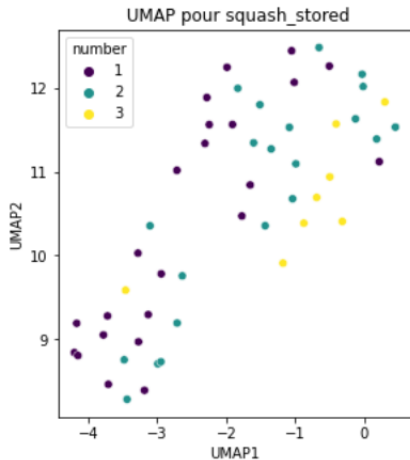
```
%pip install umap-learn  
import umap
```

- Charger un jeu de données type Swiss\_Roll.

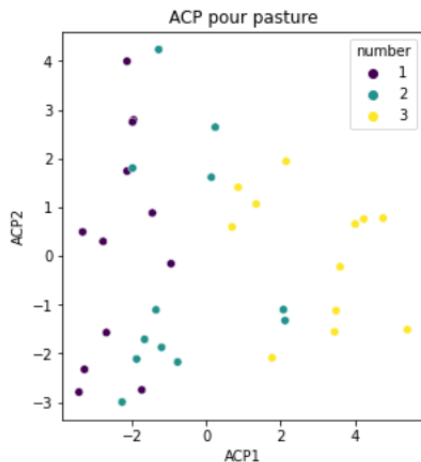
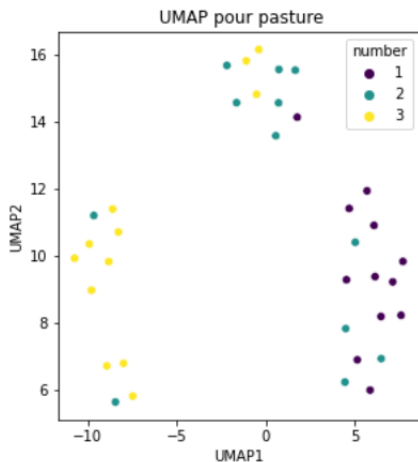
```
from sklearn.datasets import make_swiss_roll  
from mpl_toolkits.mplot3d import Axes3D  
%matplotlib  
  
X, color = make_swiss_roll(n_samples=1500)
```

- Entraînement et projection dans le plan.

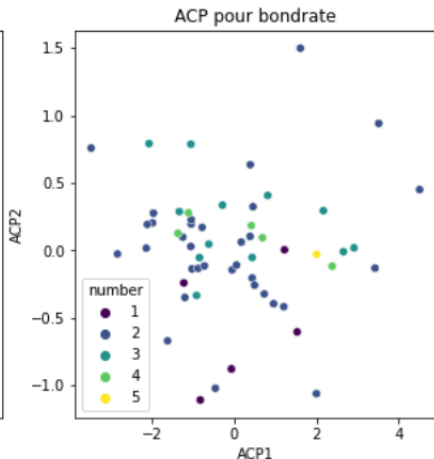
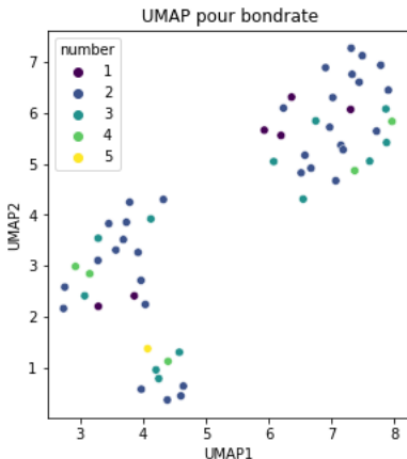
```
um = UMAP(n_components=2, n_neighbors=500, min_dist=0.5)  
embeddings = um.fit_transform(X)  
  
print(embeddings.shape)  
  
plt.scatter(embeddings[:,0], embeddings[:,1], c=color, cmap=plt.cm.Spectral)  
plt.title("Affichage du nuage de points dans l'espace réduit")  
plt.show()
```



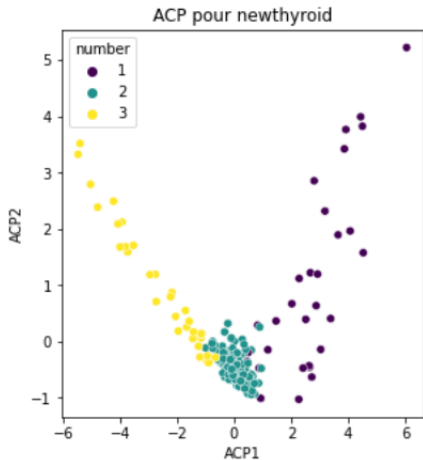
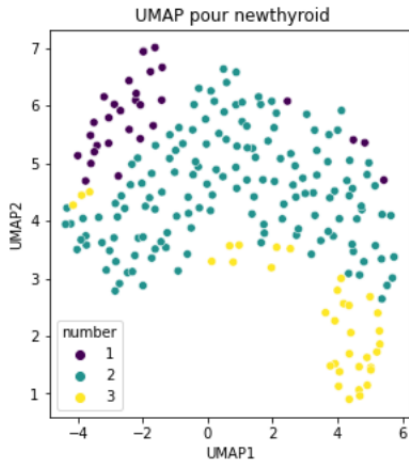
`umap.UMAP(init=X_acp,n_neighbors=25, min_dist=0.02,  
n_components=2)`



*umap.UMAP(init=X\_acp,n\_neighbors=5, min\_dist=1, n\_components=2)*



`umap.UMAP(init=X_acp,n_neighbors=25,min_dist=0.1,  
n_components=2)`



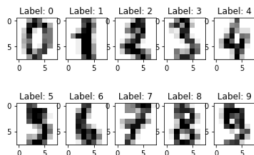
`umap.UMAP(init=X_acp,n_neighbors=100, min_dist=0.5,  
n_components=2)`

- UMAP est une méthode non-paramétrique donc il est impossible de projeter une nouvelle observation, il faut refaire l'optimisation.
- À l'inverse d'une méthode de réduction de dimension par projection linéaire type ACP, avec UMAP les coordonnées des observations dans l'espace d'arrivée sont moins interprétables physiquement.
- La recherche des bons paramètres **n\_neighbors** et **min\_dist** n'est pas évidente et ne permet pas toujours de rendre une vision globale des données initiales.

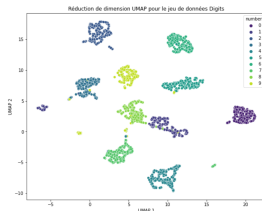


# Conclusion

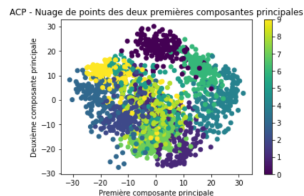
- Pour des datasets qui présentent une structure topologique marquée dans chaque sous groupe, les rendus de l'algorithme UMAP sont visibles et permettent bien de visualiser les clusters.
- Ainsi avec UMAP sur Digits, votre modèle de machine learning type KNN ou Random Forest obtient des meilleurs scores et devient plus rapide à l'exécution.



Digits Dataset.



UMAP projection  
Digits.



ACP Digits