Reduction of dimension

PCA, LLE, tSNE, UMAP

Copernicus master

2025–2026

Thomas Corpetti

Outline

Outline

Principal Component Analysis

- **Problem** : In statistical learning, we accumulate P measurements of N variables (color, contrast, ...). We obtain a data matrix X of size $P \times N$.

  $\Longrightarrow$ How to **visualize** this data when $N > 2$?

  $\Longrightarrow$ How to retain only the **useful** information ?

  ...

- **Principle of PCA** : Represent the information along the axes with the highest variance.

Principal Component Analysis

- **Problem** : In statistical learning, we accumulate P measurements of N variables (color, contrast, ...). We obtain a data matrix X of size $P \times N$.

  $\Longrightarrow$ How to **visualize** this data when $N > 2$?

  $\Longrightarrow$ How to retain only the **useful** information ?

  ...

- **Principle of PCA** : Represent the information along the axes with the highest variance.

Principal Component Analysis

- **Problem** : In statistical learning, we accumulate P measurements of N variables (color, contrast, ...). We obtain a data matrix X of size $P \times N$.

  $\Longrightarrow$ How to **visualize** this data when $N > 2$?

  $\Longrightarrow$ How to retain only the **useful** information ?

  ...

- **Principle of PCA** : Represent the information along the axes with the highest variance.

Principal Component Analysis

- **Problem** : In statistical learning, we accumulate P measurements of N variables (color, contrast, ...). We obtain a data matrix X of size $P \times N$.

  $\Longrightarrow$ How to **visualize** this data when $N > 2$?

  $\Longrightarrow$ How to retain only the **useful** information ?

  ...

- **Principle of PCA** : Represent the information along the axes with the highest variance.

Principal Component Analysis

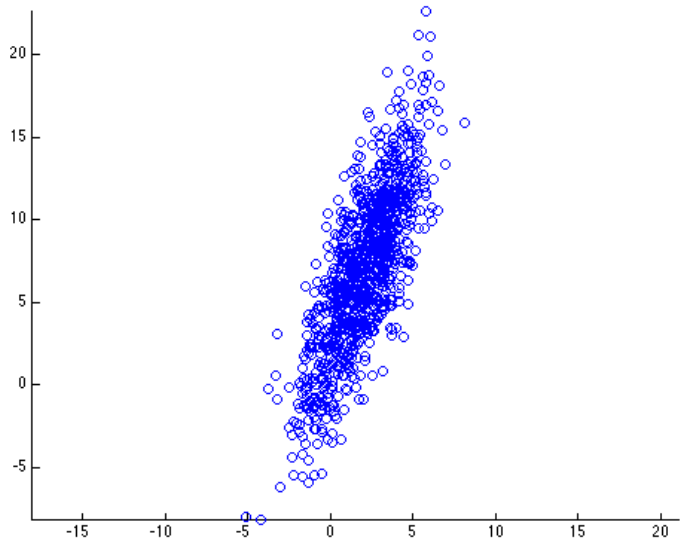
- **Problem** : In statistical learning, we accumulate P measurements of N variables (color, contrast, ...). We obtain a data matrix X of size $P \times N$.

  $\Longrightarrow$ How to **visualize** this data when $N > 2$?

  $\Longrightarrow$ How to retain only the **useful** information ?

  ...

- **Principle of PCA** : Represent the information along the axes with the highest variance.



Principal Component Analysis : How Does It Work ?

- **Objective** : Find the axes that explain the maximum variance of all the data. Input data :

$$X = \begin{bmatrix} X_{1,1} & ... & X_{1,N} \\ ... & ... & ... \\ X_{P,1} & ... & X_{P,N} \end{bmatrix}$$

- **Center of Mass** of matrix X :

$$g = [\overline{X_1}, ..., \overline{X_N}]^T$$

- **Centered Matrix** (with $\mathbf{1} = [1, 1, ..., 1]^T$) :

$$\overline{X} = \begin{bmatrix} X_{1,1} - \overline{X_1} & ... & X_{1,N} - \overline{X_N} \\ ... & ... & ... \\ X_{P,1} - \overline{X_1} & ... & X_{P,N} - \overline{X_N} \end{bmatrix} = X - \mathbf{1}g^T$$

Principal Component Analysis : How Does It Work ?

- **Objective** : Find the axes that explain the maximum variance of all the data. Input data :

$$X = \begin{bmatrix} X_{1,1} & ... & X_{1,N} \\ ... & ... & ... \\ X_{P,1} & ... & X_{P,N} \end{bmatrix}$$

- **Center of Mass** of matrix X :

$$g = [\overline{X_1}, ..., \overline{X_N}]^T$$

- **Centered Matrix** (with $\mathbf{1} = [1, 1, ..., 1]^T$) :

$$\overline{X} = \begin{bmatrix} X_{1,1} - \overline{X_1} & ... & X_{1,N} - \overline{X_N} \\ ... & ... & ... \\ X_{P,1} - \overline{X_1} & ... & X_{P,N} - \overline{X_N} \end{bmatrix} = X - \mathbf{1}g^T$$

Principal Component Analysis : How Does It Work ?

- **Objective** : Find the axes that explain the maximum variance of all the data. Input data :

$$X = \begin{bmatrix} X_{1,1} & ... & X_{1,N} \\ ... & ... & ... \\ X_{P,1} & ... & X_{P,N} \end{bmatrix}$$

- **Center of Mass** of matrix X :

$$g = [\overline{X_1}, ..., \overline{X_N}]^T$$

- **Centered Matrix** (with $\mathbf{1} = [1, 1, ..., 1]^T$) :

$$\overline{X} = \begin{bmatrix} X_{1,1} - \overline{X_1} & ... & X_{1,N} - \overline{X_N} \\ ... & ... & ... \\ X_{P,1} - \overline{X_1} & ... & X_{P,N} - \overline{X_N} \end{bmatrix} = X - \mathbf{1}g^T$$

Principal Component Analysis : How Does It Work ?

- **Standardized Centered Matrix** :

$$\widetilde{X} = \begin{bmatrix} \frac{X_{1,1}-\overline{X_1}}{\sigma(X_1)} & \cdots & \frac{X_{1,N}-\overline{X_N}}{\sigma(X_1)} \\ \cdots & \cdots & \cdots \\ \frac{X_{P,1}-\overline{X_1}}{\sigma(X_1)} & \cdots & \frac{X_{P,N}-\overline{X_N}}{\sigma(X_N)} \end{bmatrix}$$

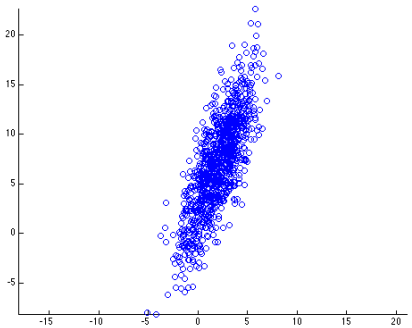
- To Standardize or Not to Standardize the Matrix ?
  - If not : a variable with high variance will dominate everything.
  - If yes : we may give importance to noisy data.

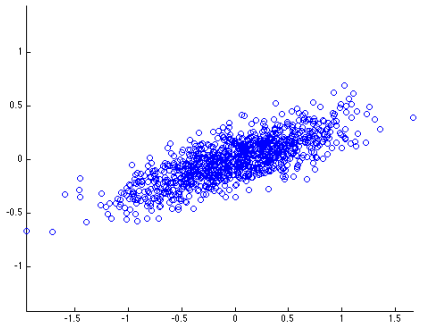
Principal Component Analysis : How Does It Work ?

- **Standardized Centered Matrix** :

$$\widetilde{X} = \begin{bmatrix} \frac{X_{1,1}-\overline{X_1}}{\sigma(X_1)} & \cdots & \frac{X_{1,N}-\overline{X_N}}{\sigma(X_1)} \\ \cdots & \cdots & \cdots \\ \frac{X_{P,1}-\overline{X_1}}{\sigma(X_1)} & \cdots & \frac{X_{P,N}-\overline{X_N}}{\sigma(X_N)} \end{bmatrix}$$

- **To Standardize or Not to Standardize the Matrix ?**
  - If not : a variable with high variance will dominate everything.
  - If yes : we may give importance to noisy data.





Not Standardized Standardized Centered

Principal Component Analysis : How Does It Work ?

- **Matrix of Correlations or Covariances** M (with $K = P - 1$) :

$$M = \frac{1}{K} W^T W$$

- Matrix is **square, symmetric, and real**.
- Let Λ be the diagonal matrix of eigenvalues of M arranged in descending order, and Q the matrix of associated eigenvectors.
- The **vector that explains the most inertia** in the cloud is the first eigenvector (same for the others).
- The variance explained by the k-th eigenvector is λ_k.

Principal Component Analysis : How Does It Work ?

- **Matrix of Correlations or Covariances** M (with $K = P - 1$) :

$$M = \frac{1}{K} W^T W$$

- Matrix is **square, symmetric, and real**.
- Let Λ be the diagonal matrix of eigenvalues of M arranged in descending order, and Q the matrix of associated eigenvectors.
- The **vector that explains the most inertia** in the cloud is the first eigenvector (same for the others).
- The variance explained by the k-th eigenvector is λ_k.

Principal Component Analysis : How Does It Work ?

- **Matrix of Correlations or Covariances** M (with $K = P - 1$) :

$$M = \frac{1}{K} W^T W$$

- Matrix is **square, symmetric, and real**.
- Let Λ be the diagonal matrix of eigenvalues of M arranged in descending order, and Q the matrix of associated eigenvectors.
- The **vector that explains the most inertia** in the cloud is the first eigenvector (same for the others).
- The variance explained by the k-th eigenvector is λ_k.

Principal Component Analysis : Key Takeaways

We have a cloud of points (centered/reduced, as needed) represented by W.

- The **Matrix of Correlations or Covariances** M is defined as (with $K = P - 1$) :

$$M = \frac{1}{K} W^T W$$

- The orthonormal basis Q that explains the most variance is represented by the matrix of **unit eigenvectors associated with eigenvalues, arranged in descending order**.
- The variance explained by each component corresponds to its associated eigenvalue.
- Coordinates of W in the new basis : $W' = Q^T W$

Principal Component Analysis : Key Takeaways

We have a cloud of points (centered/reduced, as needed) represented by W.

- The **Matrix of Correlations or Covariances** M is defined as (with $K = P - 1$) :

$$M = \frac{1}{K} W^T W$$

- The orthonormal basis Q that explains the most variance is represented by the matrix of **unit eigenvectors associated with eigenvalues, arranged in descending order**.
- The variance explained by each component corresponds to its associated eigenvalue.
- Coordinates of W in the new basis : $W' = Q^T W$

Principal Component Analysis : Key Takeaways

We have a cloud of points (centered/reduced, as needed) represented by W.

- The **Matrix of Correlations or Covariances** M is defined as (with $K = P - 1$) :

$$M = \frac{1}{K} W^T W$$

- The orthonormal basis Q that explains the most variance is represented by the matrix of **unit eigenvectors associated with eigenvalues, arranged in descending order**.
- The variance explained by each component corresponds to its associated eigenvalue.
- Coordinates of W in the new basis : $W' = Q^T W$

Principal Component Analysis : Key Takeaways

We have a cloud of points (centered/reduced, as needed) represented by W.

- The **Matrix of Correlations or Covariances** M is defined as (with $K = P - 1$) :

$$M = \frac{1}{K} W^T W$$

- The orthonormal basis Q that explains the most variance is represented by the matrix of **unit eigenvectors associated with eigenvalues, arranged in descending order**.
- The variance explained by each component corresponds to its associated eigenvalue.
- Coordinates of W in the new basis : $W' = Q^T W$

Principal Component Analysis : Key Takeaways

We have a cloud of points (centered/reduced, as needed) represented by W.

- The **Matrix of Correlations or Covariances** M is defined as (with $K = P - 1$) :

$$M = \frac{1}{K} W^T W$$

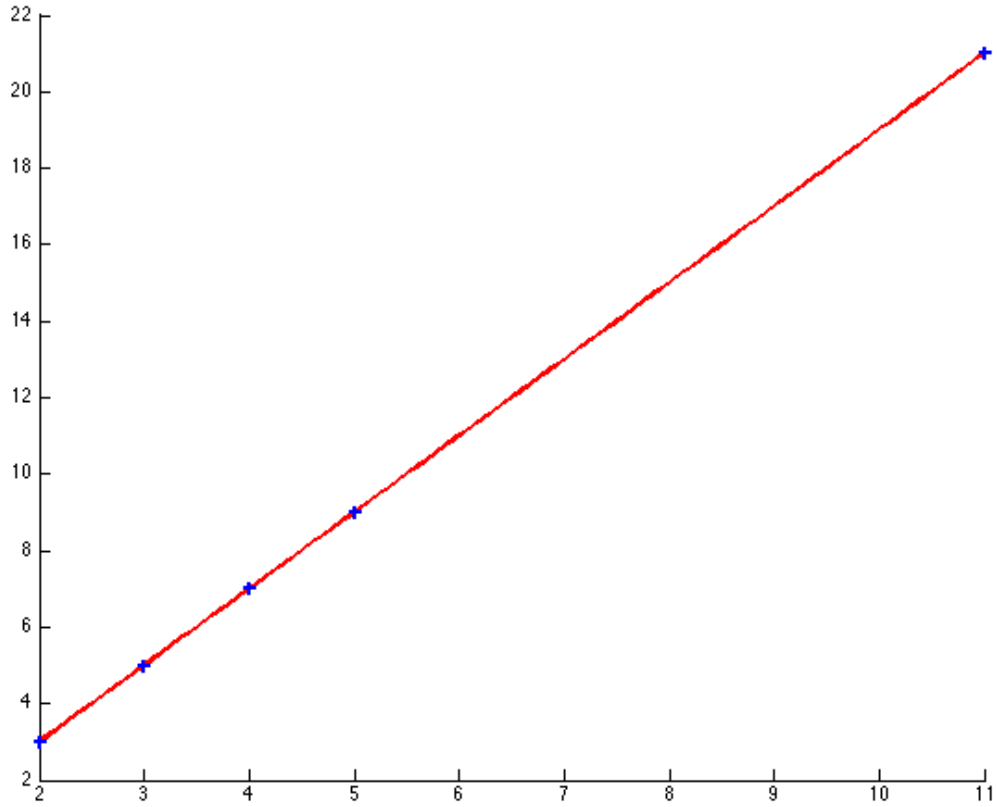
- The orthonormal basis Q that explains the most variance is represented by the matrix of **unit eigenvectors associated with eigenvalues, arranged in descending order**.
- The variance explained by each component corresponds to its associated eigenvalue.
- Coordinates of W in the new basis : $W' = Q^T W$

Principal Component Analysis : Exercises

We have the following dataset of points :

$$X = \begin{bmatrix} 2 & 3 \\ 5 & 9 \\ 4 & 7 \\ 3 & 5 \\ 11 & 21 \end{bmatrix}$$

1. What are the principal axes (we decide to center but not reduce) ?
2. What are the coordinates of these points in the basis composed of the principal components ?
3. What is the coefficient of correlation ?
4. What conclusions can we draw ?



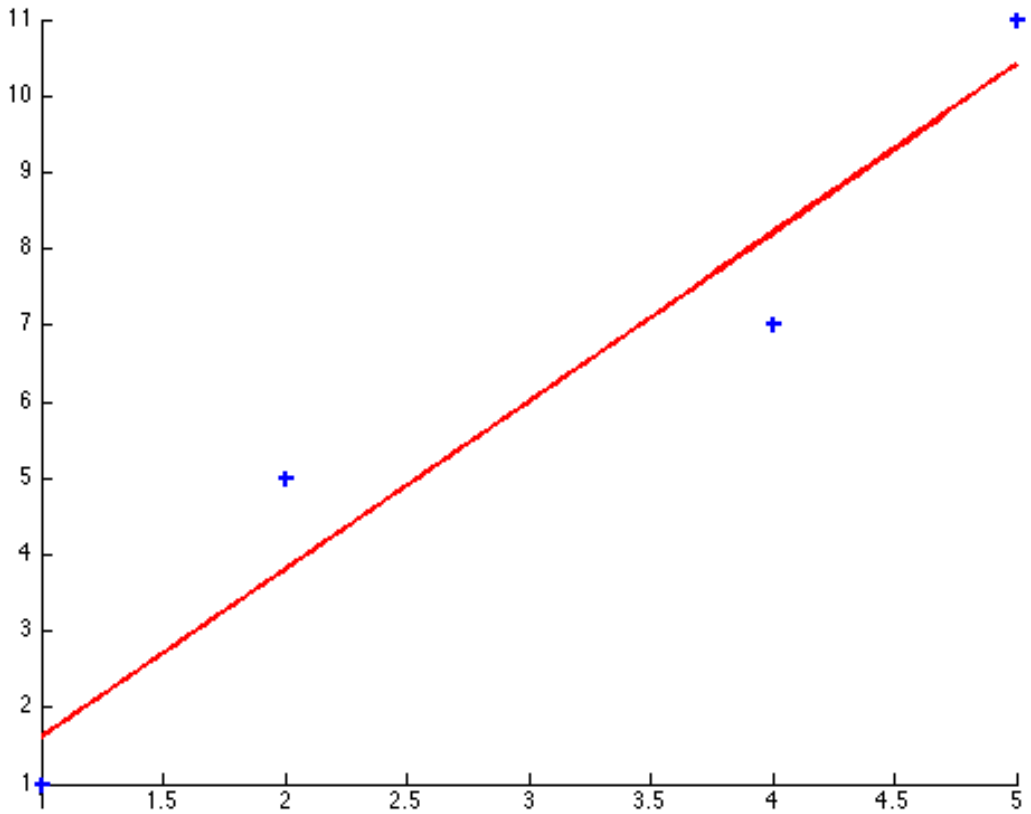
Dispersion of the data along a perfect straight line

Principal Component Analysis : Exercises

We have the following data points :

$$X = \begin{bmatrix} 1 & 1 \\ 2 & 5 \\ 4 & 7 \\ 5 & 11 \end{bmatrix}$$

1. What are the principal axes (we decide to center it but not to reduce it) ?
2. What are the coordinates of these points in the basis composed of the principal components ?
3. What is the correlation coefficient ?
4. What can we conclude ?



Dispersion of points

Principal Component Analysis : Exercises

Course Grades : We consider the following table of course grades

	Maths	Physics	French	Languages	Music
Erwan	6	6	5	5.5	8
Armelle	8	8	8	8	9
Rudy	6	7	11	9.5	11
Albert	14.5	14.5	15.5	15	8
Didier	14	14	12	12	10
Nicolas	11	10	5.5	7	13
Hélène	5.5	7	14	11.5	10
Catherine	13	12.5	8.5	9.5	12
Ronan	9	9.5	12.5	12	18

Averages

Maths	Physics	French	Languages	Music
9.67	9.83	10.22	10.00	11.00

See `https://tinyurl.com/47cvvauw`

Principal Component Analysis : Exercises

Course Grades : We consider the following table of course grades

	Maths	Physics	French	Languages	Music
Erwan	6	6	5	5.5	8
Armelle	8	8	8	8	9
Rudy	6	7	11	9.5	11
Albert	14.5	14.5	15.5	15	8
Didier	14	14	12	12	10
Nicolas	11	10	5.5	7	13
Hélène	5.5	7	14	11.5	10
Catherine	13	12.5	8.5	9.5	12
Ronan	9	9.5	12.5	12	18

Averages

	Maths	Physics	French	Languages	Music
	9.67	9.83	10.22	10.00	11.00

See `https://tinyurl.com/47cvvauw`

Principal Component Analysis : Exercises

We want to perform PCA on the grade table.

Here is the covariance matrix

```
mat_var_cov =

   12.8125   11.1562    2.9896    5.1562    0.1250
   11.1562   10.0625    4.6354    5.9062    0.0625
    2.9896    4.6354   13.5694   10.3438    0.4375
    5.1562    5.9062   10.3438    8.6250    0.8125
    0.1250    0.0625    0.4375    0.8125    9.7500
```

1. What operations should be performed ?
2. Calculate the first row.
3. The eigenvalues are as follows : $\lambda_1 = 31.41$, $\lambda_2 = 13.67$, $\lambda_3 = 9.69$, $\lambda_4 = 0.04$, $\lambda_5 = 0.005$. What are the proportions of variance explained by the first 3 axes ?
4. Is it correct to limit the representation to the first 3 axes ?

Principal Component Analysis : Exercises

We want to perform PCA on the grade table.

Here is the covariance matrix

```
mat_var_cov =

   12.8125   11.1562    2.9896    5.1562    0.1250
   11.1562   10.0625    4.6354    5.9062    0.0625
    2.9896    4.6354   13.5694   10.3438    0.4375
    5.1562    5.9062   10.3438    8.6250    0.8125
    0.1250    0.0625    0.4375    0.8125    9.7500
```

1. What operations should be performed ?
2. Calculate the first row.
3. The eigenvalues are as follows : $\lambda_1 = 31.41$, $\lambda_2 = 13.67$, $\lambda_3 = 9.69$, $\lambda_4 = 0.04$, $\lambda_5 = 0.005$. What are the proportions of variance explained by the first 3 axes ?
4. Is it correct to limit the representation to the first 3 axes ?

Principal Component Analysis : Exercises

We want to perform PCA on the grade table.

Here is the covariance matrix

```
mat_var_cov =

   12.8125   11.1562    2.9896    5.1562    0.1250
   11.1562   10.0625    4.6354    5.9062    0.0625
    2.9896    4.6354   13.5694   10.3438    0.4375
    5.1562    5.9062   10.3438    8.6250    0.8125
    0.1250    0.0625    0.4375    0.8125    9.7500
```

1. What operations should be performed ?
2. Calculate the first row.
3. The eigenvalues are as follows : $\lambda_1 = 31.41$, $\lambda_2 = 13.67$, $\lambda_3 = 9.69$, $\lambda_4 = 0.04$, $\lambda_5 = 0.005$. What are the proportions of variance explained by the first 3 axes ?
4. Is it correct to limit the representation to the first 3 axes ?

Principal Component Analysis : Exercises

Here is the matrix of unit eigenvectors

```
V =

    0.5174   -0.5654   -0.0524   -0.2882    0.5717
    0.5096   -0.3717   -0.0144    0.5378   -0.5592
    0.4955    0.6469    0.1144    0.4001    0.4035
    0.4753    0.3317    0.0207   -0.6827   -0.4444
    0.0323    0.1168   -0.9917    0.0393    0.0152
```

Reminder of the order : Math, Physics, French, Languages, Music

1. Interpret these values.
2. What are the first 3 coordinates of Erwan in this basis ?
3. Interpret the grades in this new space.

Principal Component Analysis : Exercises

Here is the matrix of unit eigenvectors

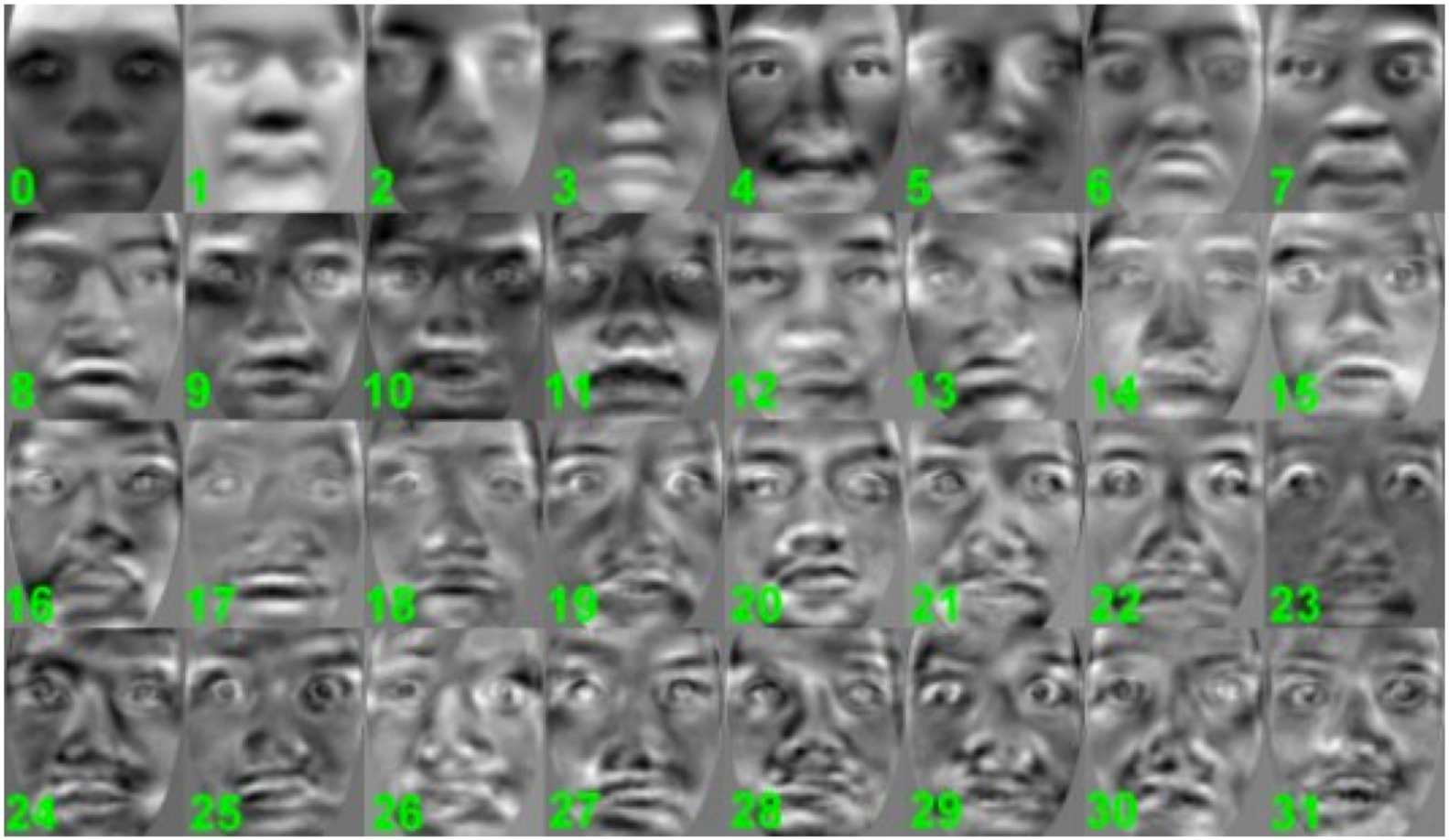
```
V =

    0.5174   -0.5654   -0.0524   -0.2882    0.5717
    0.5096   -0.3717   -0.0144    0.5378   -0.5592
    0.4955    0.6469    0.1144    0.4001    0.4035
    0.4753    0.3317    0.0207   -0.6827   -0.4444
    0.0323    0.1168   -0.9917    0.0393    0.0152
```

Reminder of the order : Math, Physics, French, Languages, Music

1. Interpret these values.
2. What are the first 3 coordinates of Erwan in this basis ?
3. Interpret the grades in this new space.

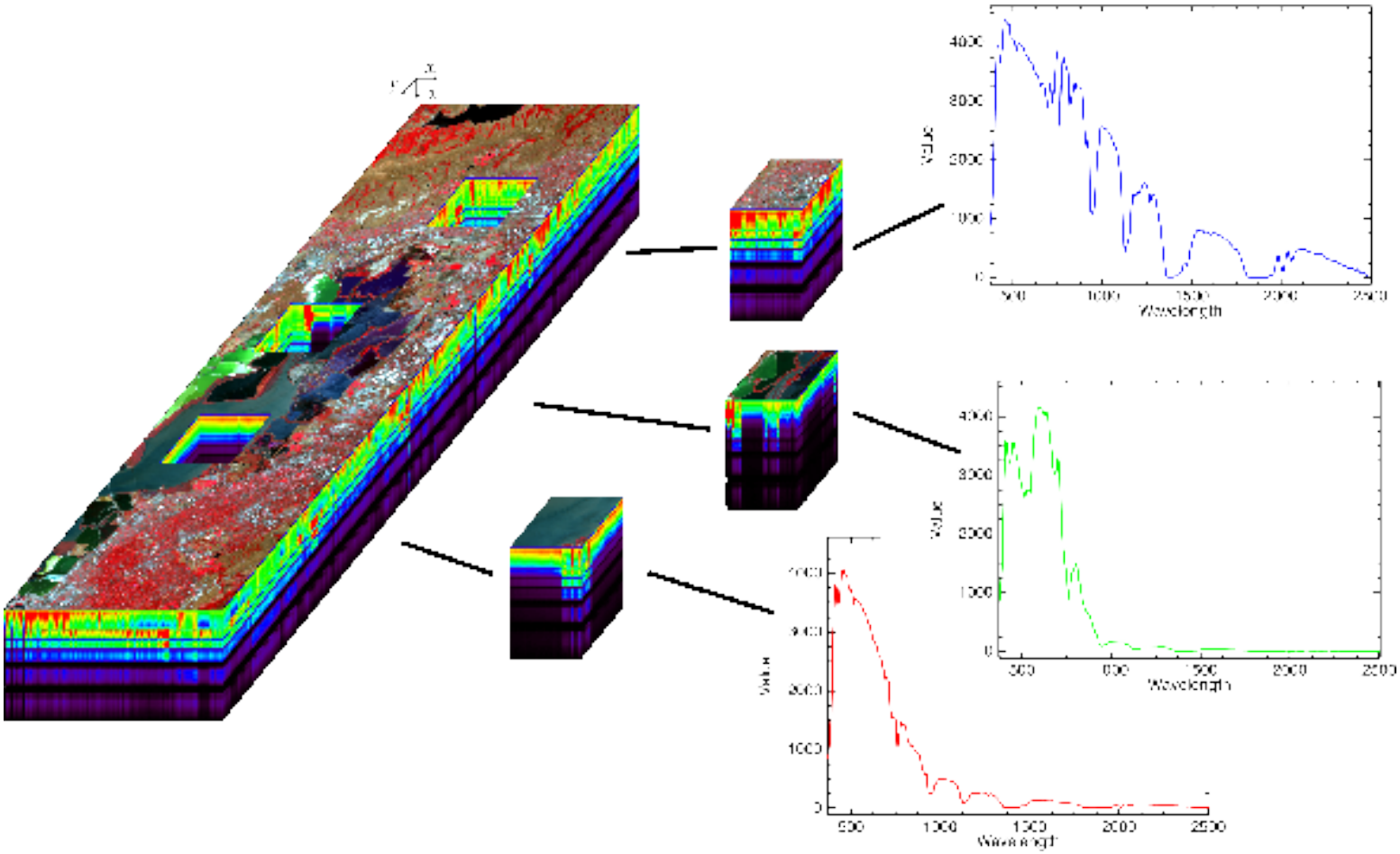
Face Recognition

Creating sets of eigenvectors ("eigenfaces")



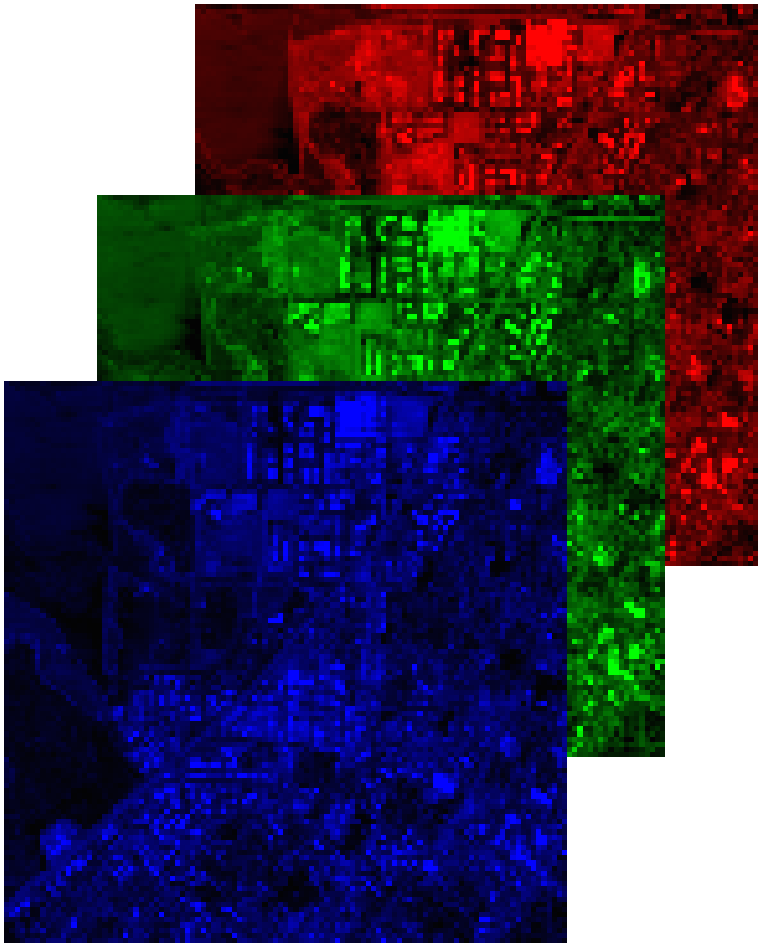
Useful for dimensionality reduction, face recognition, etc.

Hyperspectral Remote Sensing



Hyperspectral Remote Sensing

Color visualization



Interpolation in this Basis

Interpolation in this Basis

Instead of interpolating $X_{missing} = MX$, we interpolate

Outline

Locally Linear Embedding

- We have a data matrix X of size $P \times N$ (P measurements i N dimensions).
- Assumption : Locally, the observation cloud X is generated by an affine subspace.
- We seek X' of dimension $M < N$ that accurately represents the structure of X.
  - The neighbors $\mathbf{x}'_j$ of $\mathbf{x}'_i$ in the reduced space must be the same as the neighbors $\mathbf{x}_j$ of $\mathbf{x}_i$ in the original space.

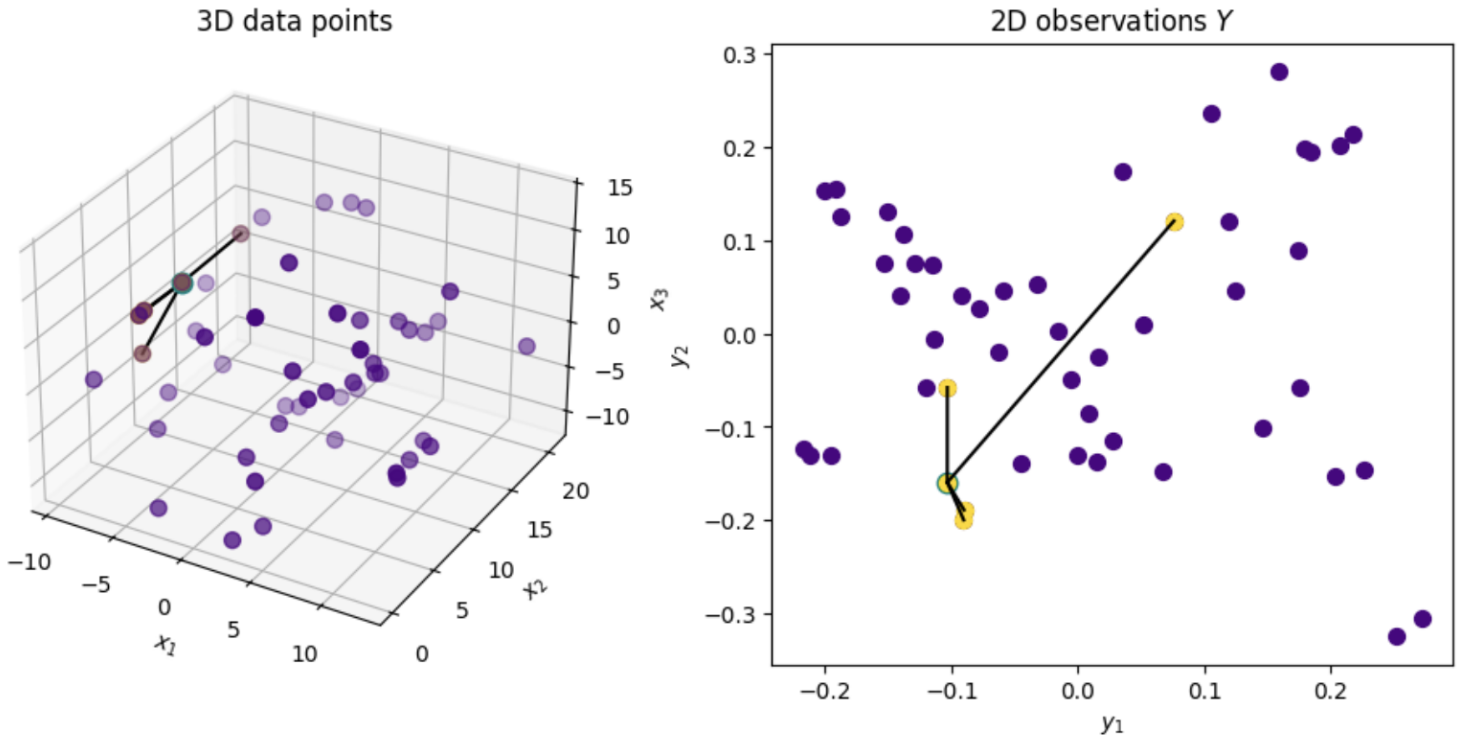
Locally Linear Embedding

- We have a data matrix X of size $P \times N$ (P measurements i N dimensions).
- Assumption : Locally, the observation cloud X is generated by an affine subspace.
- We seek X' of dimension $M < N$ that accurately represents the structure of X.
  - The neighbors $\mathbf{x}'_j$ of $\mathbf{x}'_i$ in the reduced space must be the same as the neighbors $\mathbf{x}_j$ of $\mathbf{x}_i$ in the original space.

Locally Linear Embedding

- We have a data matrix X of size $P \times N$ (P measurements i N dimensions).
- Assumption : Locally, the observation cloud X is generated by an affine subspace.
- We seek X' of dimension $M < N$ that accurately represents the structure of X.
  - The neighbors $\mathbf{x}'_j$ of $\mathbf{x}'_i$ in the reduced space must be the same as the neighbors $\mathbf{x}_j$ of $\mathbf{x}_i$ in the original space.

Illustration



Locally Linear Embedding : process

- For each point $\mathbf{x}_i \in \mathbb{R}^N$
  1. Find its neighbors $V_i = \{\mathbf{x}'_1, \mathbf{x}'_2, \dots, \mathbf{x}'_k\}$
  2. Find the best linear combination of $\mathbf{x}_i$ with these neighbors by computing W^* with

$$W^* = \arg\min_W \sum_i \left\| \mathbf{x}_i - \sum_{j \in V_i} w_{ij} \mathbf{x}_j \right\|^2 \text{ s.t. } \sum_j w_{ij} = 1$$

- Find the best reduced data that matches with this linear combination

$$Y^* = \arg\min_Y \sum_i \left\| \mathbf{y}_i - \sum_{j \in V_i} w^*_{ij} \mathbf{y}_j \right\|^2 \text{ with } \mathbf{y}_i \in \mathbb{R}^M$$

Locally Linear Embedding : process

- For each point $\mathbf{x}_i \in \mathbb{R}^N$
  1. Find its neighbors $V_i = \{\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_k\}$
  2. Find the best linear combination of $\mathbf{x}_i$ with these neighbors by computing W^* with

$$W^* = \arg\min_W \sum_i \left\| \mathbf{x}_i - \sum_{j \in V_i} w_{ij} \mathbf{x}_j \right\|^2 \text{ s.t. } \sum_j w_{ij} = 1$$

- Find the best reduced data that matches with this linear combination

$$Y^* = \arg\min_Y \sum_i \left\| \mathbf{y}_i - \sum_{j \in V_i} w^*_{ij} \mathbf{y}_j \right\|^2 \text{ with } \mathbf{y}_i \in \mathbb{R}^M$$

Locally Linear Embedding : process

- For each point $\mathbf{x}_i \in \mathbb{R}^N$
  1. Find its neighbors $V_i = \{\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_k\}$
  2. Find the best linear combination of $\mathbf{x}_i$ with these neighbors by computing W^* with

$$W^* = \arg\min_W \sum_i \left\| \mathbf{x}_i - \sum_{j \in V_i} w_{ij} \mathbf{x}_j \right\|^2 \text{ s.t. } \sum_j w_{ij} = 1$$

- Find the best reduced data that matches with this linear combination

$$Y^* = \arg\min_Y \sum_i \left\| \mathbf{y}_i - \sum_{j \in V_i} w^*_{ij} \mathbf{y}_j \right\|^2 \text{ with } \mathbf{y}_i \in \mathbb{R}^M$$

Locally Linear Embedding : process

- For each point $\mathbf{x}_i \in \mathbb{R}^N$
  1. Find its neighbors $V_i = \{\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_k\}$
  2. Find the best linear combination of $\mathbf{x}_i$ with these neighbors by computing W^* with

$$W^* = \arg\min_W \sum_i \left\| \mathbf{x}_i - \sum_{j \in V_i} w_{ij} \mathbf{x}_j \right\|^2 \quad \text{s.t.} \sum_j w_{ij} = 1$$

- Find the best reduced data that matches with this linear combination

$$Y^* = \arg\min_Y \sum_i \left\| \mathbf{y}_i - \sum_{j \in V_i} w^*_{ij} \mathbf{y}_j \right\|^2 \quad \text{with } \mathbf{y}_i \in \mathbb{R}^M$$

Locally Linear Embedding : process

- For each point $\mathbf{x}_i \in \mathbb{R}^N$
  1. Find its neighbors $V_i = \{\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_k\}$
  2. Find the best linear combination of $\mathbf{x}_i$ with these neighbors by computing W^* with

$$W^* = \arg\min_W \sum_i \left\| \mathbf{x}_i - \sum_{j \in V_i} w_{ij} \mathbf{x}_j \right\|^2 \text{ s.t. } \sum_j w_{ij} = 1$$

- Find the best reduced data that matches with this linear combination

$$Y^* = \arg\min_Y \sum_i \left\| \mathbf{y}_i - \sum_{j \in V_i} w^*_{ij} \mathbf{y}_j \right\|^2 \text{ with } \mathbf{y}_i \in \mathbb{R}^M$$

Locally Linear Embedding : process

- For each point $\mathbf{x}_i \in \mathbb{R}^N$
  1. Find its neighbors $V_i = \{\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_k\}$
  2. Find the best linear combination of $\mathbf{x}_i$ with these neighbors by computing W^* with

$$W^* = \arg\min_W \sum_i \left\| \mathbf{x}_i - \sum_{j \in V_i} w_{ij} \mathbf{x}_j \right\|^2 \text{ s.t. } \sum_j w_{ij} = 1$$

- Find the best reduced data that matches with this linear combination

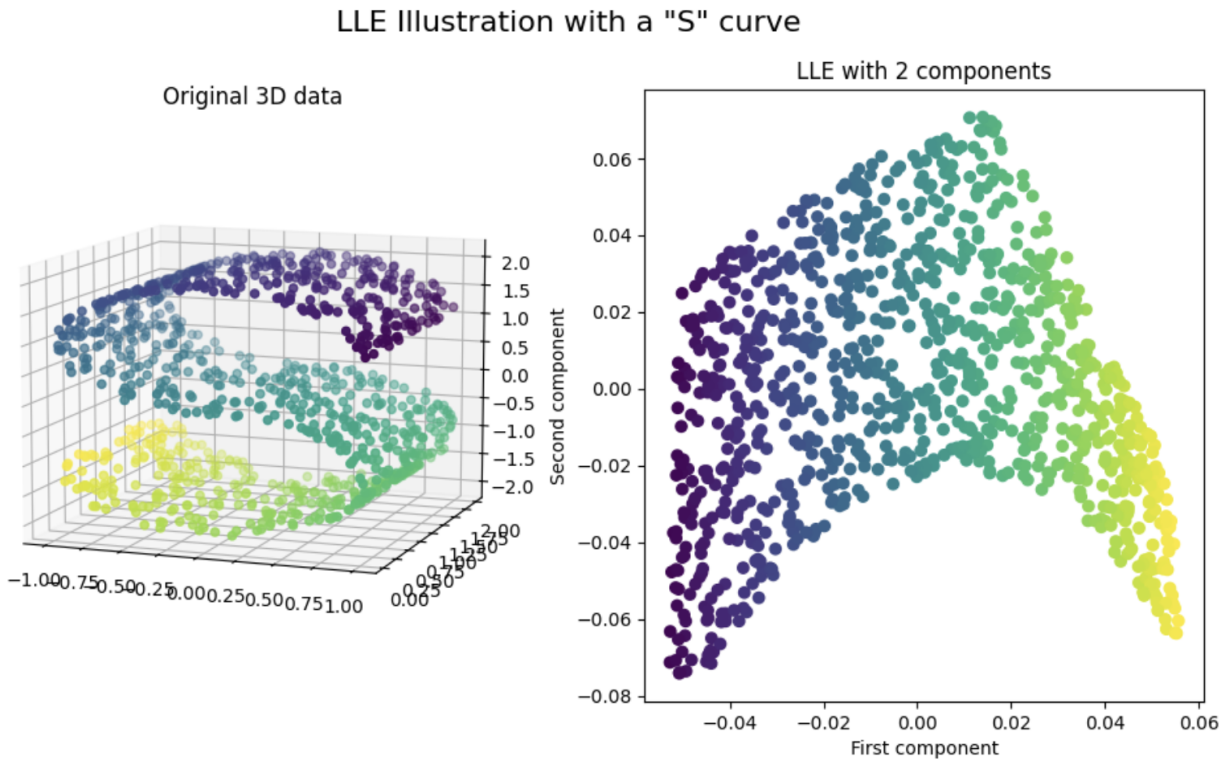
$$Y^* = \arg\min_Y \sum_i \left\| \mathbf{y}_i - \sum_{j \in V_i} w^*_{ij} \mathbf{y}_j \right\|^2 \text{ with } \mathbf{y}_i \in \mathbb{R}^M$$

Illustration

- Main parameters
  1. Number of neighbors `n_neighbors`
  2. Number of components `n_components`

Illustration

- Main parameters
  1. Number of neighbors `n_neighbors`
  2. Number of components `n_components`



Outline

t-SNE : t-distributed stochastic neighbor embedding

- LLE : Distances
- Distances $\Longrightarrow$ Distributions
- t-SNE : Distributions

t-SNE : t-distributed stochastic neighbor embedding

- LLE : Distances
- Distances $\Longrightarrow$ Distributions
- t-SNE : Distributions

t-SNE : t-distributed stochastic neighbor embedding

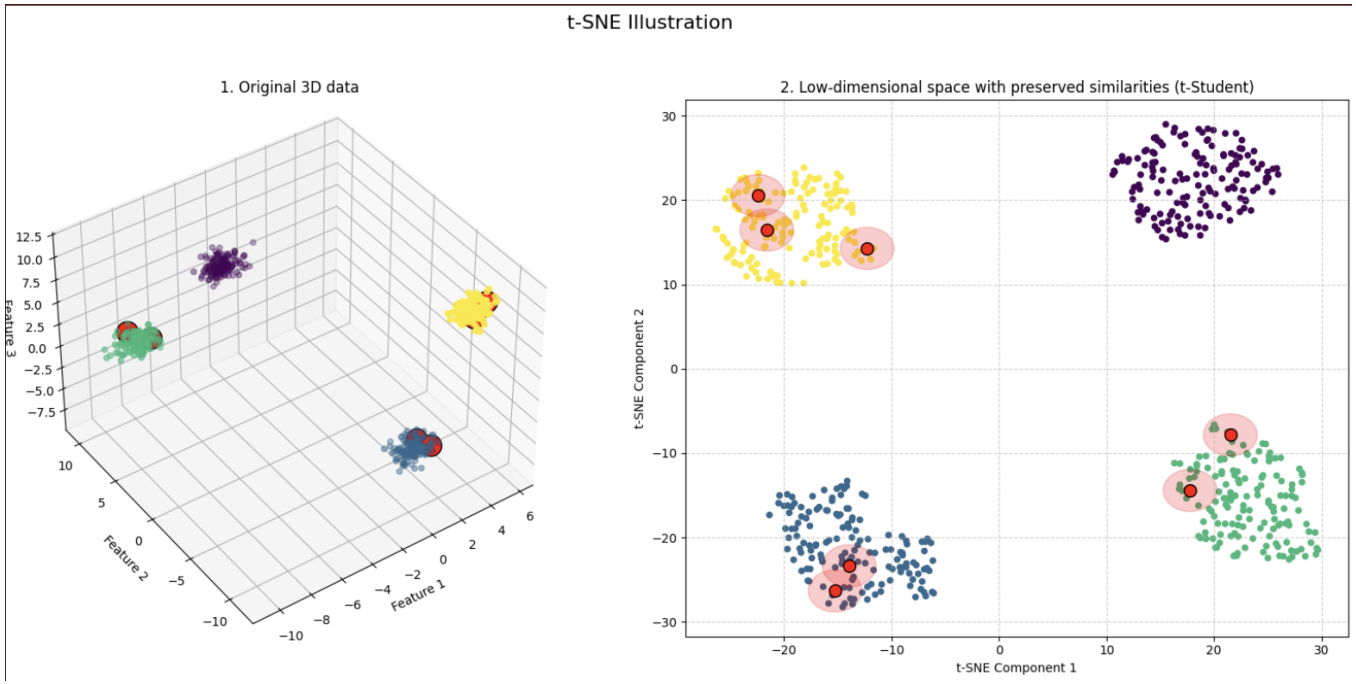
- LLE : Distances
- Distances $\Longrightarrow$ Distributions
- t-SNE : Distributions

t-SNE : t-distributed stochastic neighbor embedding

- t-distributed Stochastic Neighbor Embedding
  - A non-linear dimensionality reduction algorithm.
  - General principle similar to LLE : preserve the same local structures in the neighborhood of each point within a reduced-dimensional space.

t-SNE : t-distributed stochastic neighbor embedding

- t-distributed Stochastic Neighbor Embedding
  - A non-linear dimensionality reduction algorithm.
  - General principle similar to LLE : preserve the same local structures in the neighborhood of each point within a reduced-dimensional space.



t-SNE : t-distributed stochastic neighbor embedding

- t-distributed Stochastic Neighbor Embedding
  - A non-linear dimensionality reduction algorithm.
  - General principle similar to LLE : preserve the same local structures in the neighborhood of each point within a reduced-dimensional space.
- The neighborhood of a point $\mathbf{x}_i$ is represented here by the conditional probability $p_{j|i} = p(\mathbf{x}_j|\mathbf{x}_i)$ that $\mathbf{x}_j$ would be considered a "neighbor" of $\mathbf{x}_i$.

t-SNE : t-distributed stochastic neighbor embedding

- t-distributed Stochastic Neighbor Embedding
  - A non-linear dimensionality reduction algorithm.
  - General principle similar to LLE : preserve the same local structures in the neighborhood of each point within a reduced-dimensional space.
- The neighborhood of a point $\mathbf{x}_i$ is represented here by the conditional probability $p_{j|i} = p(\mathbf{x}_j|\mathbf{x}_i)$ that $\mathbf{x}_j$ would be considered a "neighbor" of $\mathbf{x}_i$.
  - We search for points $\mathbf{y}_i$ in the reduced space for which $p(\mathbf{y}_j|\mathbf{y}_i) \approx p(\mathbf{x}_j|\mathbf{x}_i)$.

t-SNE : t-distributed stochastic neighbor embedding, in practice (1/4)

- Consider a matrix X of size $P \times N$ (P observations $\mathbf{x}_i$, of dimension N).
- To each pair $\mathbf{x}_i, \mathbf{x}_j$, we associate the joint probability $p_{ij} = p_{ji}$ defined by :

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n} \quad \text{with} \quad p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_i^2)}$$

with the convention that $p_{ii} = 0$ so that $\sum_{i,j} p_{ij} = 1$.

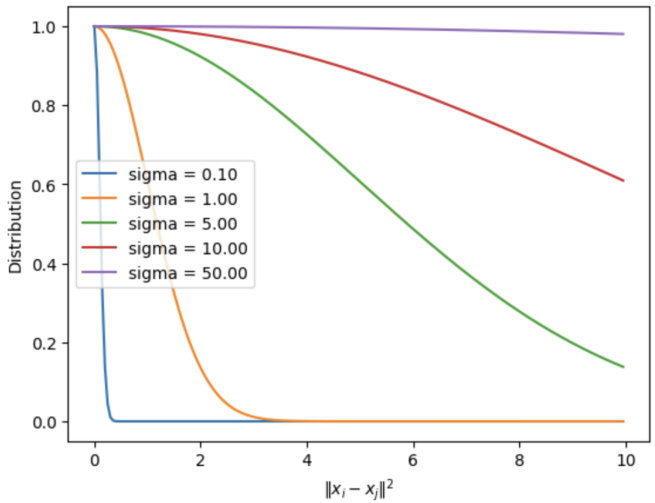
t-SNE : t-distributed stochastic neighbor embedding, in practice (1/4)

- Consider a matrix X of size $P \times N$ (P observations $\mathbf{x}_i$, of dimension N).
- To each pair $\mathbf{x}_i, \mathbf{x}_j$, we associate the joint probability $p_{ij} = p_{ji}$ defined by :

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n} \quad \text{with} \quad p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2 / 2\sigma_i^2)}$$

with the convention that $p_{ii} = 0$ so that $\sum_{i,j} p_{ij} = 1$.

- [σ] is chosen based on the desired **perplexity** ($\sim$ average number of neighbors).



t-SNE : t-distributed stochastic neighbor embedding, in practice (2/4)

Similarity in the low-dimensional space (or embedding space)

- The objective of t-SNE is to obtain a low-dimensional matrix $\mathbf{Y}$ of dimension $P \times M$, where $M < N$, such that the similarities q_{ij} approximate the p_{ij}.
- For $\mathbf{Y}$, the similarity is defined using a Student's t-distribution with 1 degree of freedom (also known as the Cauchy distribution) :

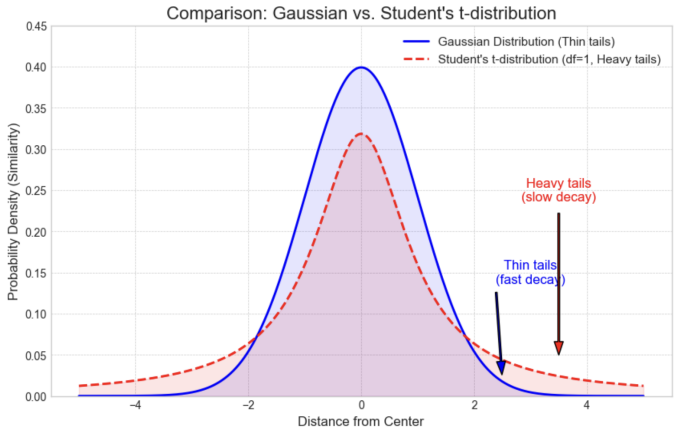
$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}}$$

with the convention, here again, that $q_{ii} = 0$.

t-SNE : t-distributed stochastic neighbor embedding, in practice (3/4)

Gaussian vs. Student's t-distribution

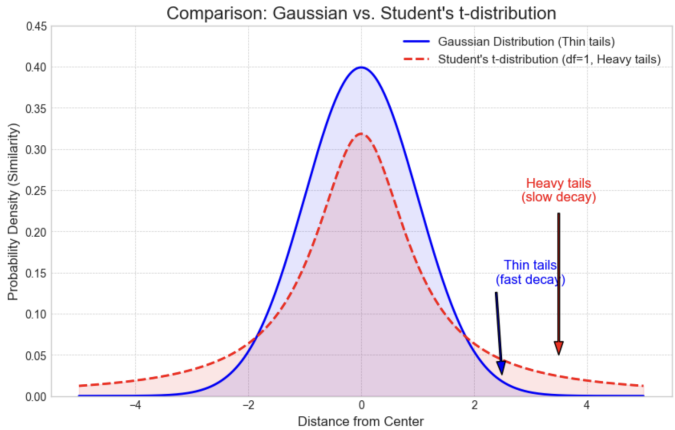
- **High-Dimensional Space : Gaussian Distribution**
  - Thin tails : probability drops off very quickly with distance.
  - $\Longrightarrow$ Only immediate neighbors are considered important.
  - **Goal :** To accurately capture the *local structure*.
- **Low-Dimensional Space : Student's t-distribution**
  - Heavy tails : probability decreases much more slowly.
  - $\Longrightarrow$ Allows non-neighbors to be placed **farther apart**.
  - Solves the **crowding problem** for clear, separated clusters.



t-SNE : t-distributed stochastic neighbor embedding, in practice (3/4)

Gaussian vs. Student's t-distribution

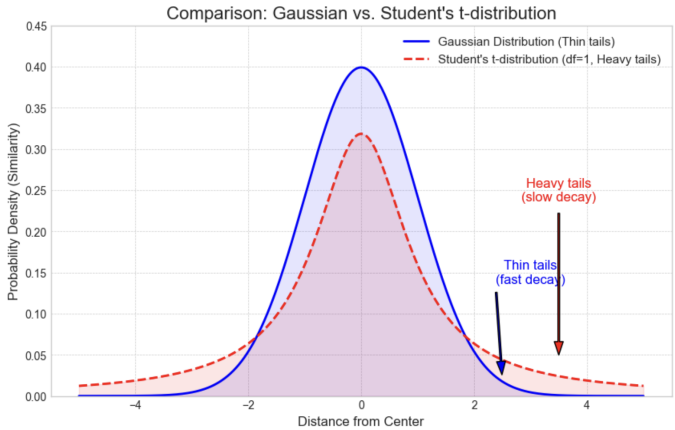
- **High-Dimensional Space : Gaussian Distribution**
  - Thin tails : probability drops off very quickly with distance.
  - $\Longrightarrow$ Only immediate neighbors are considered important.
  - **Goal :** To accurately capture the *local structure*.
- **Low-Dimensional Space : Student's t-distribution**
  - Heavy tails : probability decreases much more slowly.
  - $\Longrightarrow$ Allows non-neighbors to be placed **farther apart**.
  - Solves the **crowding problem** for clear, separated clusters.



t-SNE : t-distributed stochastic neighbor embedding, in practice (3/4)

Gaussian vs. Student's t-distribution

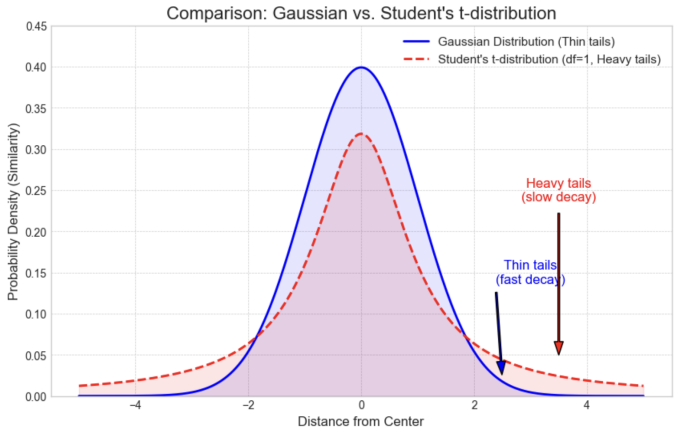
- **High-Dimensional Space : Gaussian Distribution**
  - Thin tails : probability drops off very quickly with distance.
  - $\Longrightarrow$ Only immediate neighbors are considered important.
  - **Goal :** To accurately capture the *local structure*.
- **Low-Dimensional Space : Student's t-distribution**
  - Heavy tails : probability decreases much more slowly.
  - $\Longrightarrow$ Allows non-neighbors to be placed **farther apart**.
  - Solves the **crowding problem** for clear, separated clusters.



t-SNE : t-distributed stochastic neighbor embedding, in practice (3/4)

Gaussian vs. Student's t-distribution

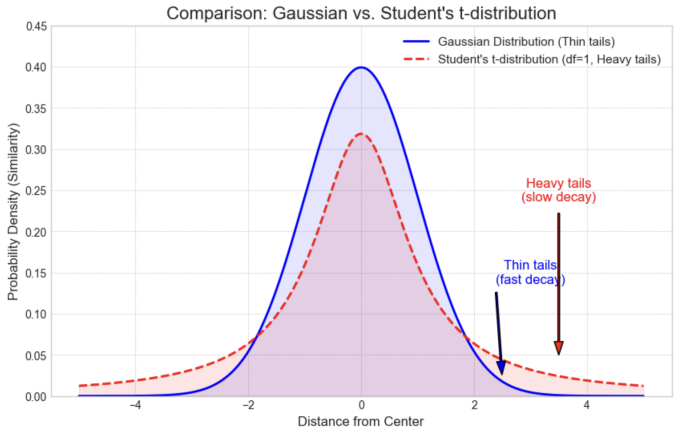
- **High-Dimensional Space : Gaussian Distribution**
  - Thin tails : probability drops off very quickly with distance.
  - $\Longrightarrow$ Only immediate neighbors are considered important.
  - **Goal :** To accurately capture the *local structure*.
- **Low-Dimensional Space : Student's t-distribution**
  - Heavy tails : probability decreases much more slowly.
  - $\Longrightarrow$ Allows non-neighbors to be placed farther apart.
  - Solves the crowding problem for clear, separated clusters.



t-SNE : t-distributed stochastic neighbor embedding, in practice (3/4)

Gaussian vs. Student's t-distribution

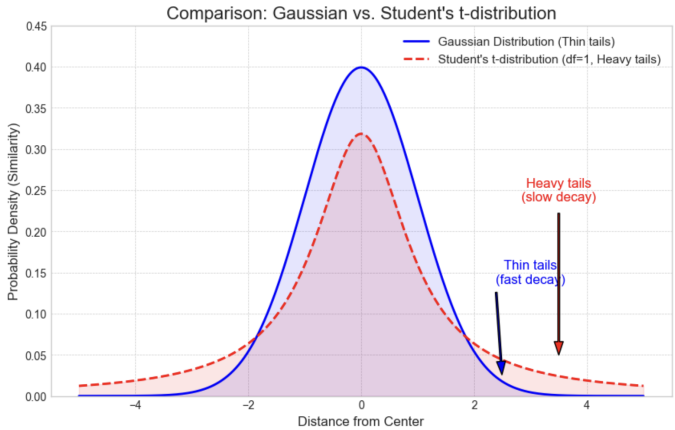
- **High-Dimensional Space : Gaussian Distribution**
  - Thin tails : probability drops off very quickly with distance.
  - $\Longrightarrow$ Only immediate neighbors are considered important.
  - **Goal :** To accurately capture the *local structure*.
- **Low-Dimensional Space : Student's t-distribution**
  - Heavy tails : probability decreases much more slowly.
  - $\Longrightarrow$ Allows non-neighbors to be placed **farther apart**.
  - Solves the crowding problem for clear, separated clusters.



t-SNE : t-distributed stochastic neighbor embedding, in practice (3/4)

Gaussian vs. Student's t-distribution

- **High-Dimensional Space : Gaussian Distribution**
  - Thin tails : probability drops off very quickly with distance.
  - $\Longrightarrow$ Only immediate neighbors are considered important.
  - **Goal :** To accurately capture the *local structure*.
- **Low-Dimensional Space : Student's t-distribution**
  - Heavy tails : probability decreases much more slowly.
  - $\Longrightarrow$ Allows non-neighbors to be placed **farther apart**.
  - Solves the **crowding problem** for clear, separated clusters.



t-SNE : t-distributed stochastic neighbor embedding, in practice (3/3)

- A measure of dissimilarity between distributions is the **Kullback-Leibler divergence** :

$$L_{t-\text{SNE}} = \sum_i D_{\text{KL}}(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

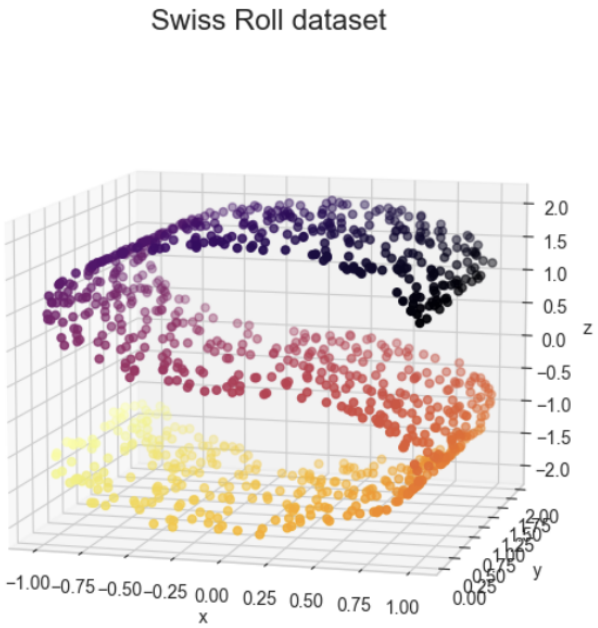
- To minimize the differences between the q_{ij} and the p_{ij}, it is sufficient to minimize $L_{t-\text{SNE}}$.

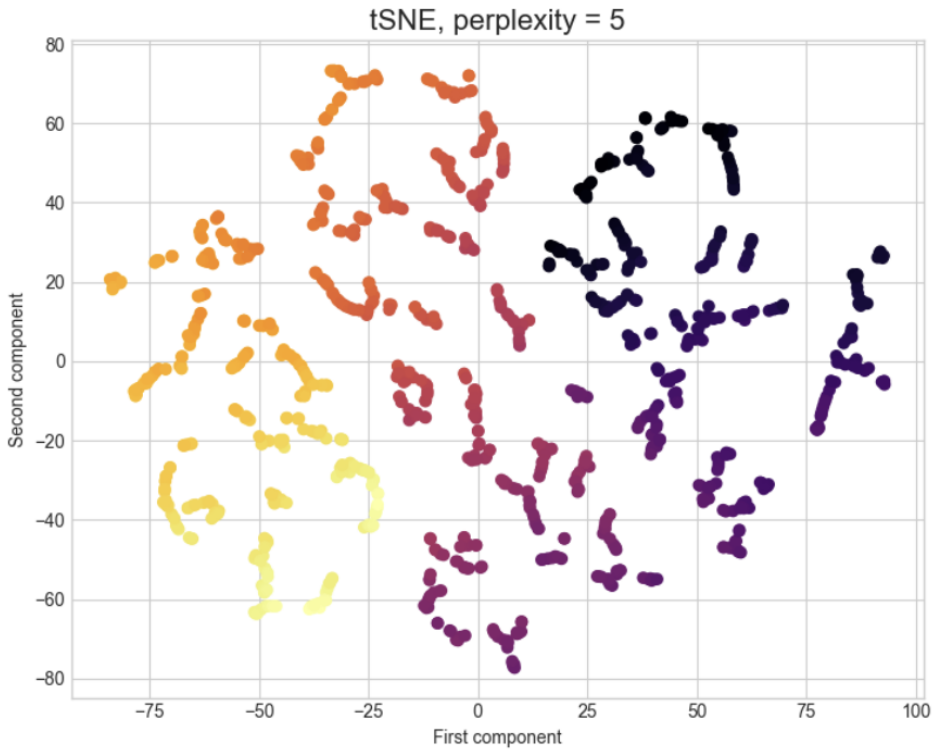
Illustration

- Main parameters
  1. Perplexity `perplexity`
  2. Number of components `n_components`

Illustration

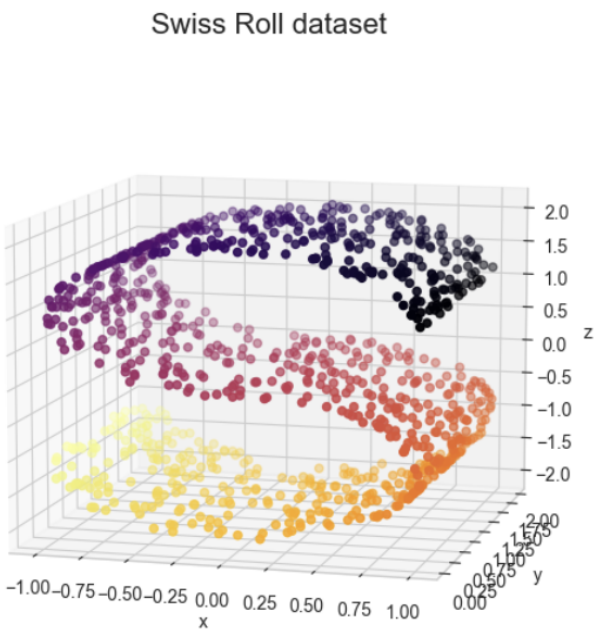
- Main parameters
  1. Perplexity `perplexity`
  2. Number of components `n_components`

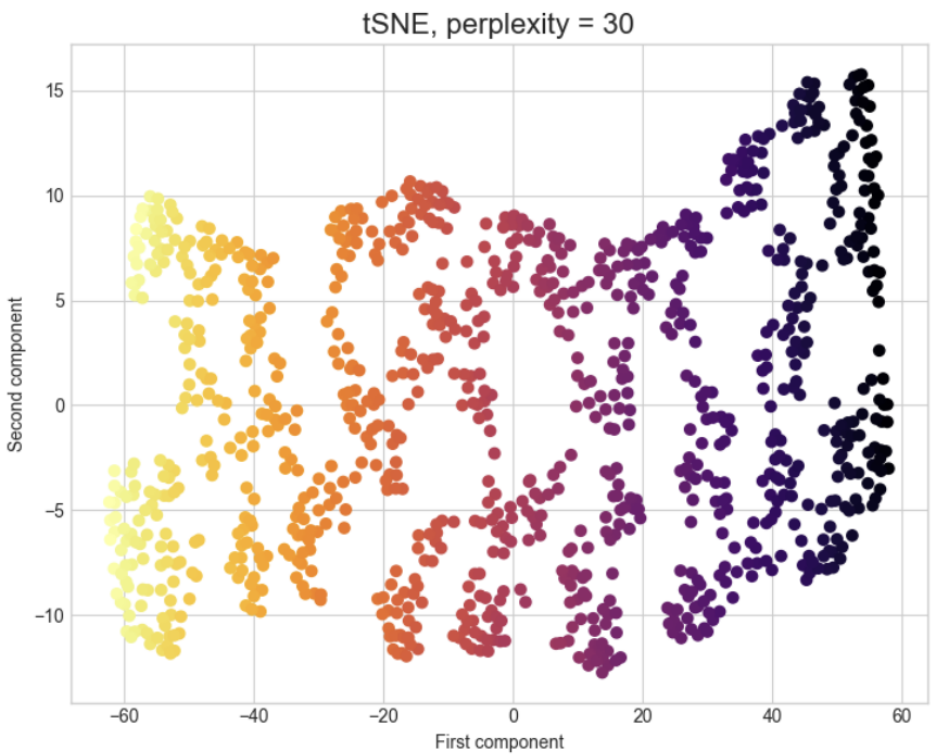




Illustration

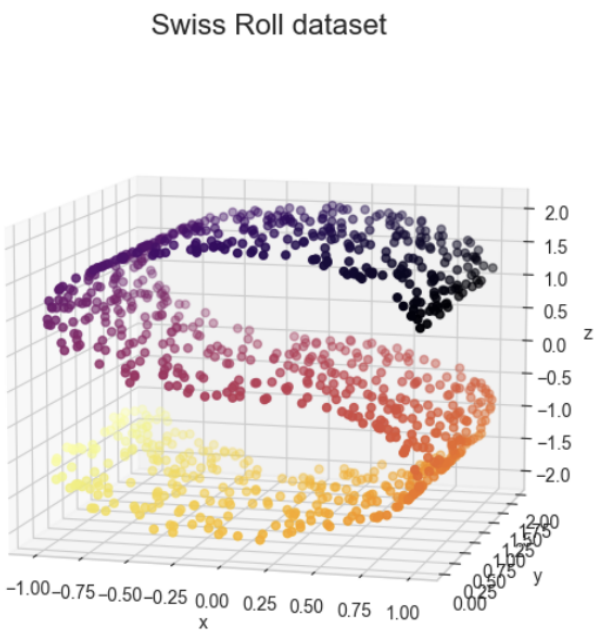
- Main parameters
  1. Perplexity `perplexity`
  2. Number of components `n_components`

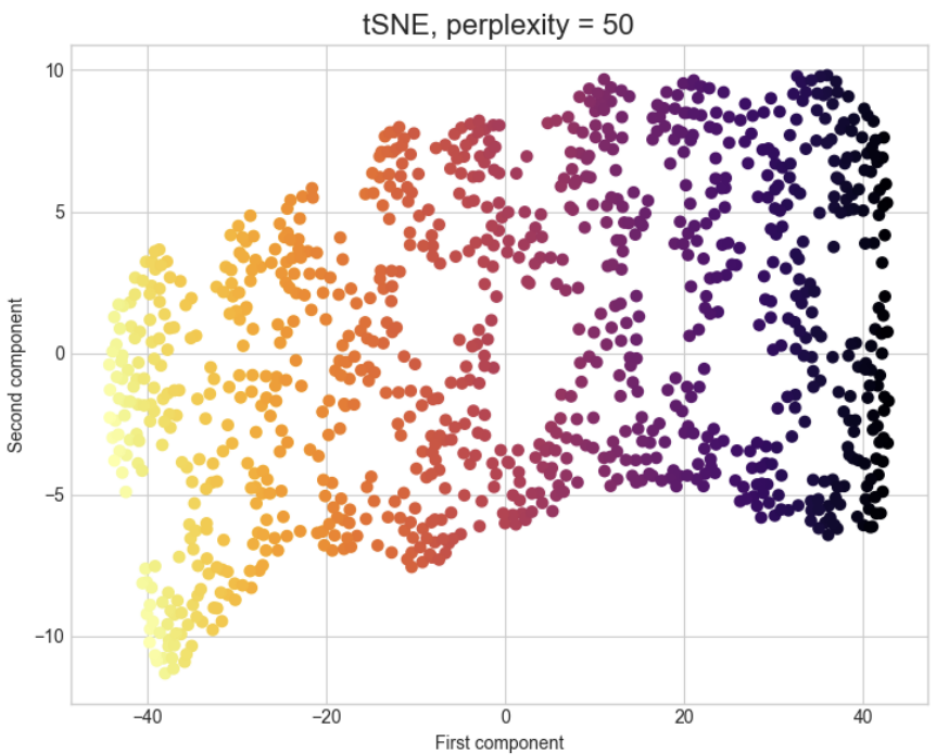




Illustration

- Main parameters
  1. Perplexity `perplexity`
  2. Number of components `n_components`





LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

LLE vs. t-SNE : Key Differences

Locally Linear Embedding (LLE)

- **Principle :** Geometric & Deterministic.
- **Assumption :** Assumes the data lies on a manifold that is *locally linear*.
- **Mechanism :** Preserves the **linear reconstruction weights** of each point from its nearest neighbors.
- **Best Use Case :** "Unrolling" single, continuous, and smooth manifolds (e.g., a Swiss Roll).

t-SNE

- **Principle :** Probabilistic & Stochastic.
- **Assumption :** Focuses on matching *similarity distributions* between points.
- **Mechanism :** Matches a Gaussian (high-D) with a heavy-tailed **t-distribution** (low-D) to preserve local similarities.
- **Best Use Case :** Visualizing distinct **clusters** and natural groupings in complex data.

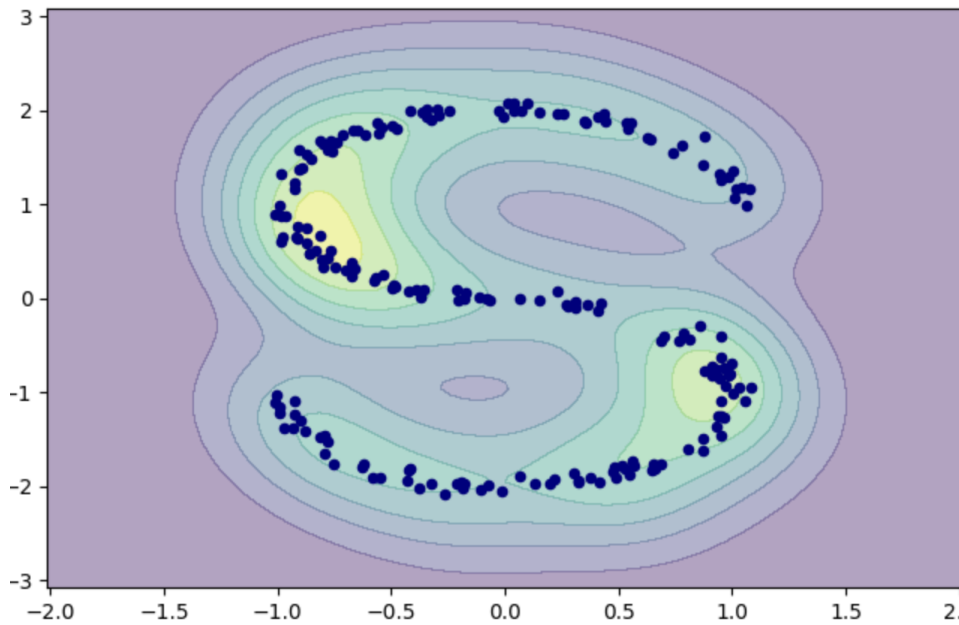
Outline

Outline

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.



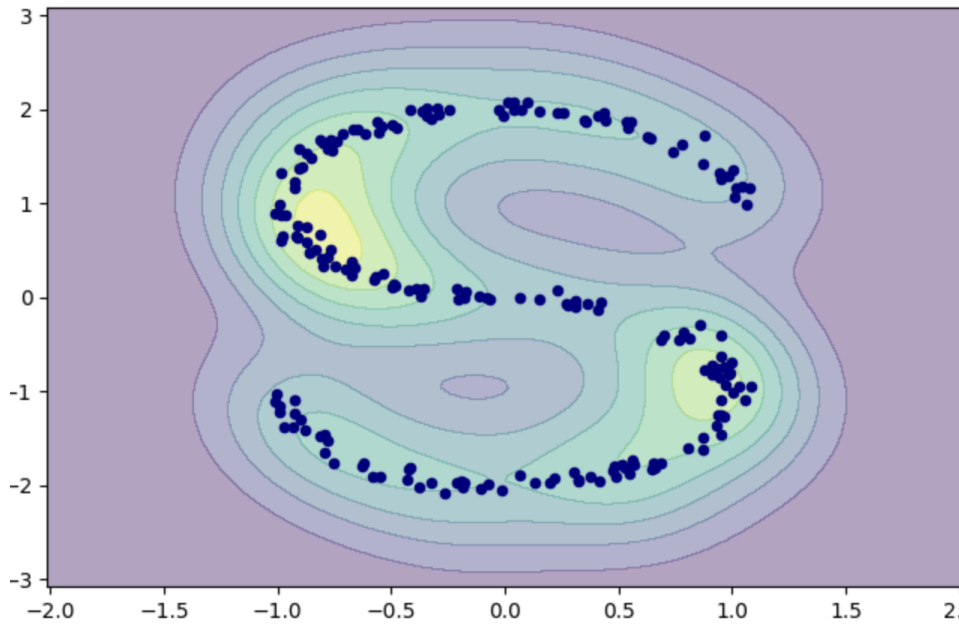
Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.



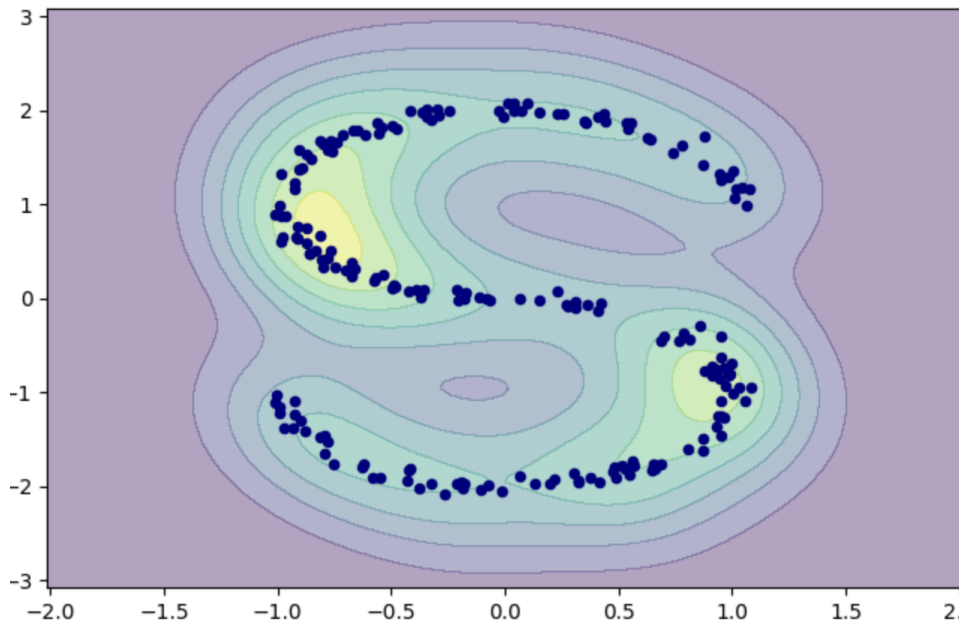
Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.



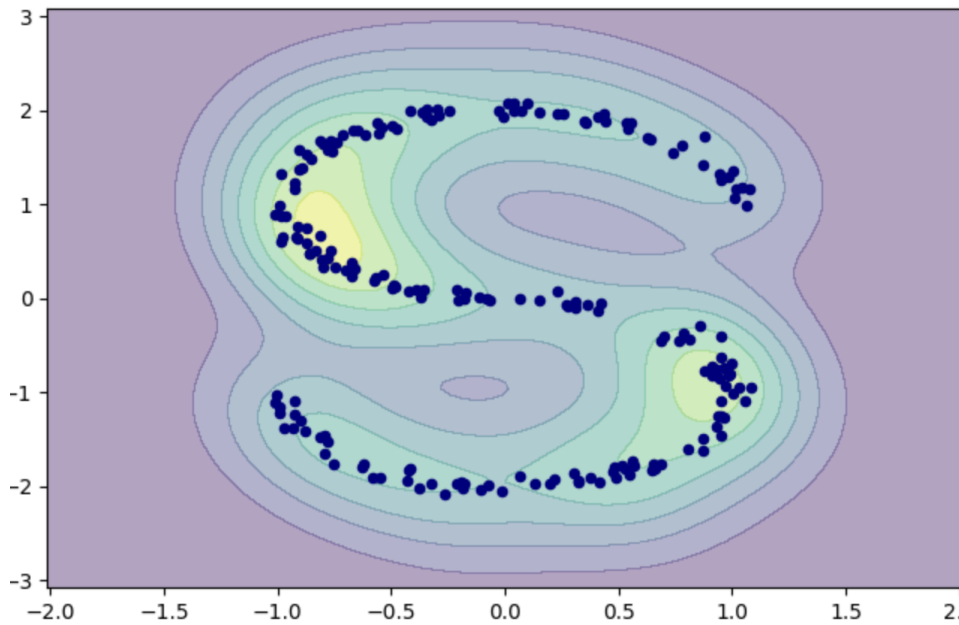
Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.



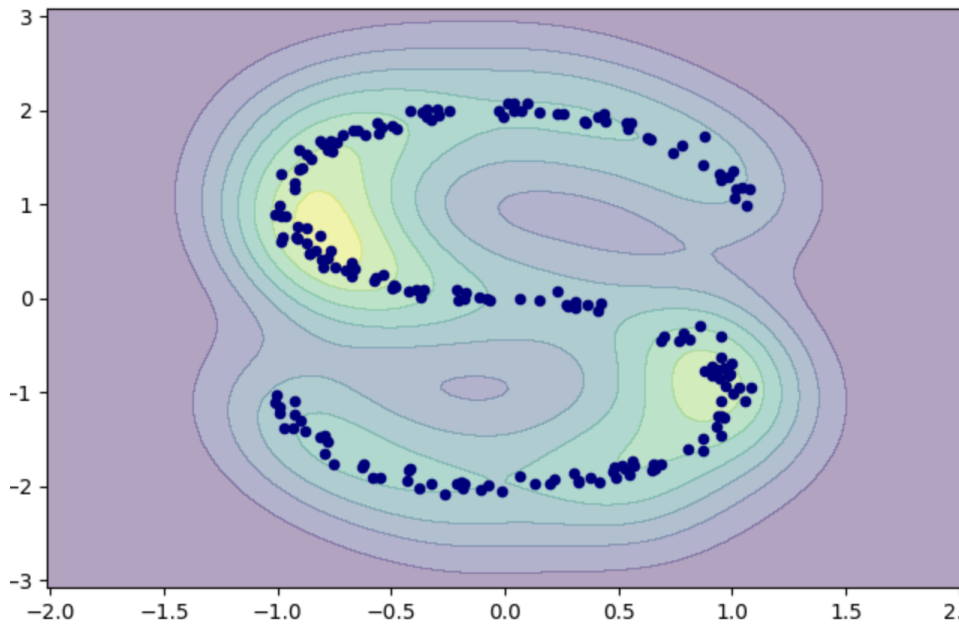
Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.



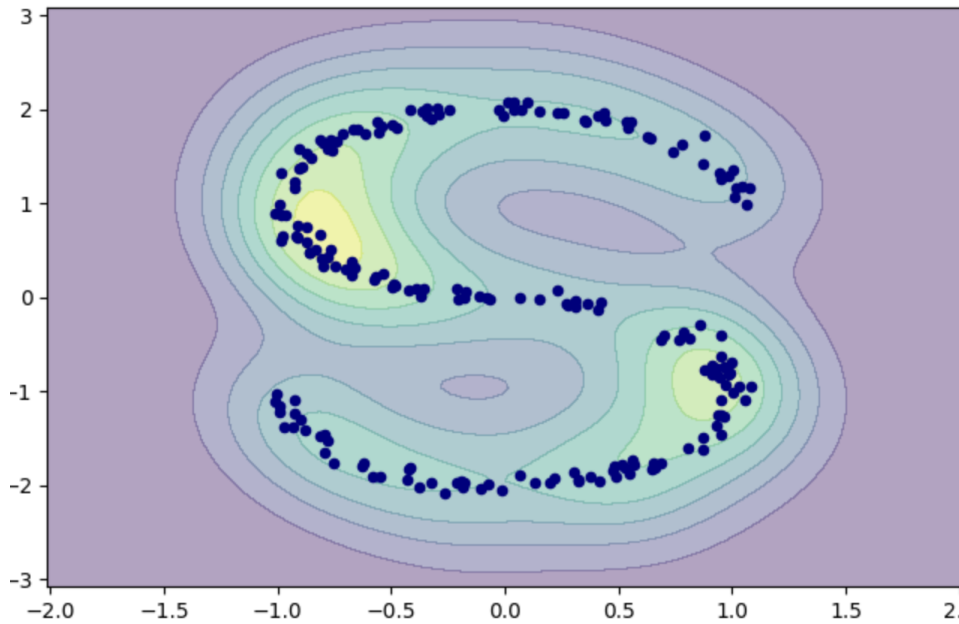
Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.



Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.

Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

Major Differences from t-SNE.

- **Adapts to Local Density :**
  - Uses a variable notion of similarity depending on the data density.
  - This allows it to better handle clusters with **varying densities** (**varying perplexity**).
- **More Flexible Similarities :**
  - Uses a broader family of similarity functions in the low-dimensional space.
- **Better Global Structure :**
  - Preserve the local/global data structure better than t-SNE

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.

Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

Major Differences from t-SNE.

- **Adapts to Local Density :**
  - Uses a variable notion of similarity depending on the data density.
  - This allows it to better handle clusters with **varying densities** (**varying perplexity**).
- **More Flexible Similarities :**
  - Uses a broader family of similarity functions in the low-dimensional space.
- **Better Global Structure :**
  - Preserve the local/global data structure better than t-SNE

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.

Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

Major Differences from t-SNE.

- **Adapts to Local Density :**
  - Uses a variable notion of similarity depending on the data density.
  - This allows it to better handle clusters with **varying densities** (**varying perplexity**).
- **More Flexible Similarities :**
  - Uses a broader family of similarity functions in the low-dimensional space.
- **Better Global Structure :**
  - Preserve the local/global data structure better than t-SNE

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.

Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

Major Differences from t-SNE.

- **Adapts to Local Density :**
  - Uses a variable notion of similarity depending on the data density.
  - This allows it to better handle clusters with **varying densities** (**varying perplexity**).
- **More Flexible Similarities :**
  - Uses a broader family of similarity functions in the low-dimensional space.
- **Better Global Structure :**
  - Preserve the local/global data structure better than t-SNE

UMAP : Uniform Manifold Approximation & Projection

General Idea

- Find a low-dimensional representation that has the same **topological structure** as the original data.

Main Steps

1. Define an appropriate / adaptive similarity metric.
2. Construct a **similarity graph** based on these similarities in the high-D space.
3. Find a low-dimensional embedding that has the most similar graph structure possible.

Major Differences from t-SNE.

- **Adapts to Local Density :**
  - Uses a variable notion of similarity depending on the data density.
  - This allows it to better handle clusters with **varying densities** (**varying perplexity**).
- **More Flexible Similarities :**
  - Uses a broader family of similarity functions in the low-dimensional space.
- **Better Global Structure :**
  - Preserve the local/global data structure better than t-SNE

UMAP : Adaptive Similarity in the Original space

- UMAP defines an adaptive similarity that varies with the local data density :

$$p_{i|j} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2 - \rho_i}{\sigma_i}\right)$$

where ρ_i is the distance from $\mathbf{x}_i$ to its nearest neighbor.

- The bandwidth σ_i is set such that :

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

where k is the parameter for the number of nearest neighbors to consider.

$\Longrightarrow$ we measure only distances higher than nearest neighbor

- $\log_2(k)$ term : every point must distribute among its k nearest neighbors

$\Longrightarrow$ Forcing the algorithm to find an adaptive local scale σ_i to meet this target.

UMAP : Adaptive Similarity in the Original space

- UMAP defines an adaptive similarity that varies with the local data density :

$$p_{i|j} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2 - \rho_i}{\sigma_i}\right)$$

where ρ_i is the distance from $\mathbf{x}_i$ to its nearest neighbor.

- The bandwidth σ_i is set such that :

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

where k is the parameter for the number of nearest neighbors to consider.

$\Longrightarrow$ we measure only distances higher than nearest neighbor

- $\log_2(k)$ term : every point must distribute among its k nearest neighbors

$\Longrightarrow$ Forcing the algorithm to find an adaptive local scale σ_i to meet this target.

UMAP : Adaptive Similarity in the Original space

- UMAP defines an adaptive similarity that varies with the local data density :

$$p_{i|j} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2 - \rho_i}{\sigma_i}\right)$$

where ρ_i is the distance from $\mathbf{x}_i$ to its nearest neighbor.

- The bandwidth σ_i is set such that :

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

where k is the parameter for the number of nearest neighbors to consider.

$\Longrightarrow$ we measure only distances higher than nearest neighbor

- $\log_2(k)$ term : every point must distribute among its k nearest neighbors

$\Longrightarrow$ Forcing the algorithm to find an adaptive local scale σ_i to meet this target.

UMAP : Adaptive Similarity in the Original space

- UMAP defines an adaptive similarity that varies with the local data density :

$$p_{i|j} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2 - \rho_i}{\sigma_i}\right)$$

where ρ_i is the distance from $\mathbf{x}_i$ to its nearest neighbor.

- The bandwidth σ_i is set such that :

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

where k is the parameter for the number of nearest neighbors to consider.

$\Longrightarrow$ we measure only distances higher than nearest neighbor

- $\log_2(k)$ term : every point must distribute among its k nearest neighbors

$\Longrightarrow$ Forcing the algorithm to find an adaptive local scale σ_i to meet this target.

UMAP : Adaptive Similarity in the Original space

- UMAP defines an adaptive similarity that varies with the local data density :

$$p_{i|j} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2 - \rho_i}{\sigma_i}\right)$$

where ρ_i is the distance from $\mathbf{x}_i$ to its nearest neighbor.

- The bandwidth σ_i is set such that :

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

where k is the parameter for the number of nearest neighbors to consider.

$\Longrightarrow$ we measure only distances higher than nearest neighbor

- $\log_2(k)$ term : every point must distribute among its k nearest neighbors

$\Longrightarrow$ Forcing the algorithm to find an adaptive local scale σ_i to meet this target.

UMAP : Adaptive Similarity in the Original space

Technical issues :

- For two points $\mathbf{x}_i$ and $\mathbf{x}_j$, the joint similarity p_{ij} must be symmetric ($p_{ij} = p_{ji}$). In UMAP, this is achieved by :

$$p_{ij} = p_{j|i} + p_{i|j} - p_{j|i} \cdot p_{i|j}$$

  - Symmetrization is necessary because $\rho_i \neq \rho_j$ (and thus $\sigma_i \neq \sigma_j$).
  - As a reminder, t-SNE uses a different symmetrization : $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$.

Similarity in the Embedding Space

- Best curve

$$f(d) = \begin{cases} 1 & \text{if } d \leq \text{min\_dist} \\ e^{-(d - \text{min\_dist})} & \text{if } d > \text{min\_dist} \end{cases}$$

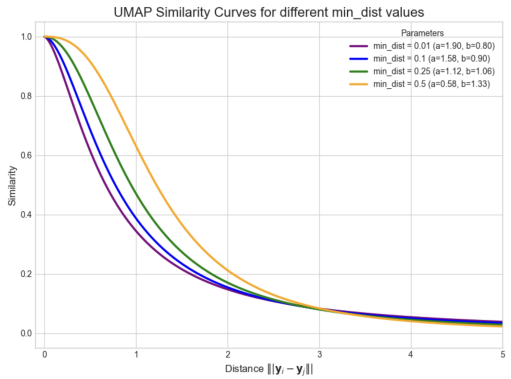
where $d = \|\mathbf{y}_i - \mathbf{y}_j\|$ and **min_dist** is the minimum allowed distance between two points in the embedding space.

- Similarity in the embedding space : inspired by the Student's t-distribution :

$$q_{ij} = \frac{1}{1 + a\|\mathbf{y}_i - \mathbf{y}_j\|^{2b}}$$

$\Longrightarrow$ **Heavy tail** to avoid the crowding problem.

- Parameters a and b are chosen automatically such that this curve, q, provides the **best possible fit** to the target curve, $f(d)$.



Similarity in the Embedding Space

- Best curve

$$f(d) = \begin{cases} 1 & \text{if } d \leq \text{min\_dist} \\ e^{-(d - \text{min\_dist})} & \text{if } d > \text{min\_dist} \end{cases}$$

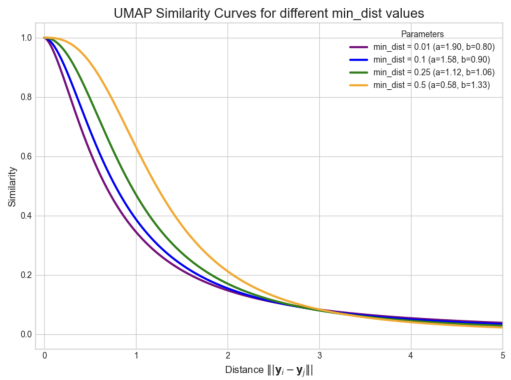
where $d = \|\mathbf{y}_i - \mathbf{y}_j\|$ and **min_dist** is the minimum allowed distance between two points in the embedding space.

- Similarity in the embedding space : inspired by the Student's t-distribution :

$$q_{ij} = \frac{1}{1 + a\|\mathbf{y}_i - \mathbf{y}_j\|^{2b}}$$

$\Longrightarrow$ **Heavy tail** to avoid the crowding problem.

- Parameters a and b are chosen automatically such that this curve, q, provides the **best possible fit** to the target curve, $f(d)$.

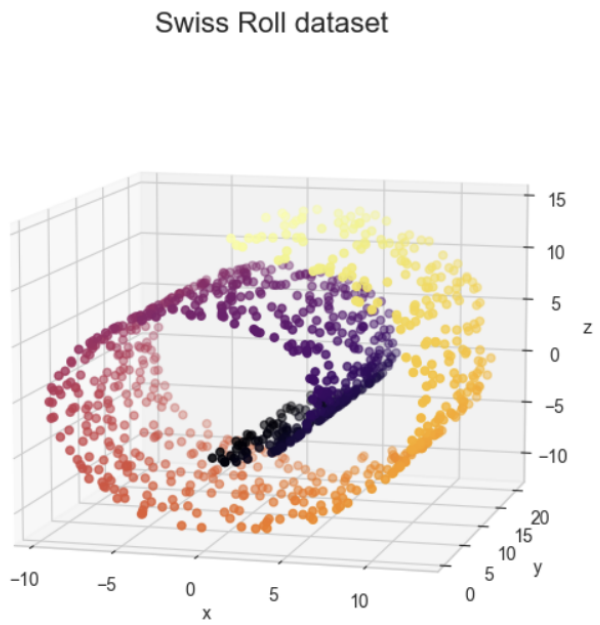


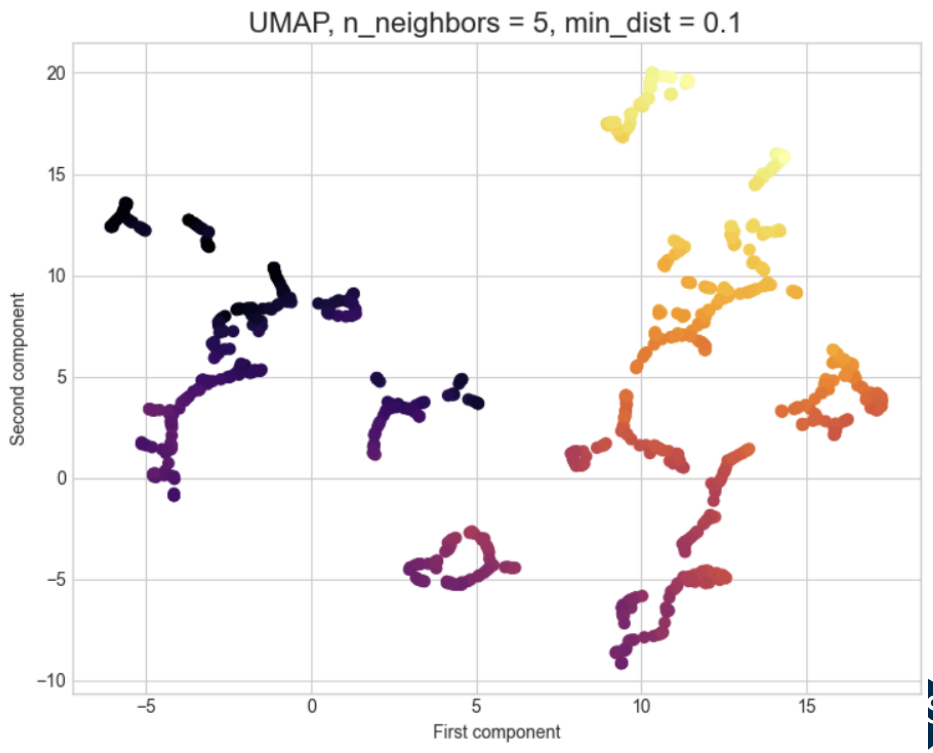
Illustration

- Main parameters
  1. Number of components `n_components`
  2. Number of neighbors `n_neighbors` (The most similar parameter to t-SNE's perplexity)
  3. Minimal distance `min_dist`

Illustration

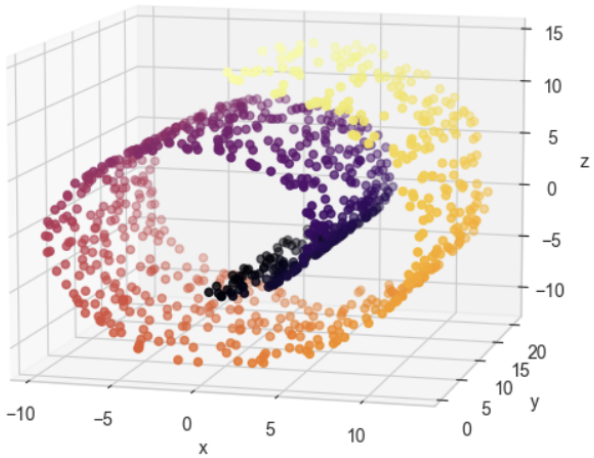
- Main parameters
  1. Number of components `n_components`
  2. Number of neighbors `n_neighbors` (The most similar parameter to t-SNE's perplexity)
  3. Minimal distance `min_dist`

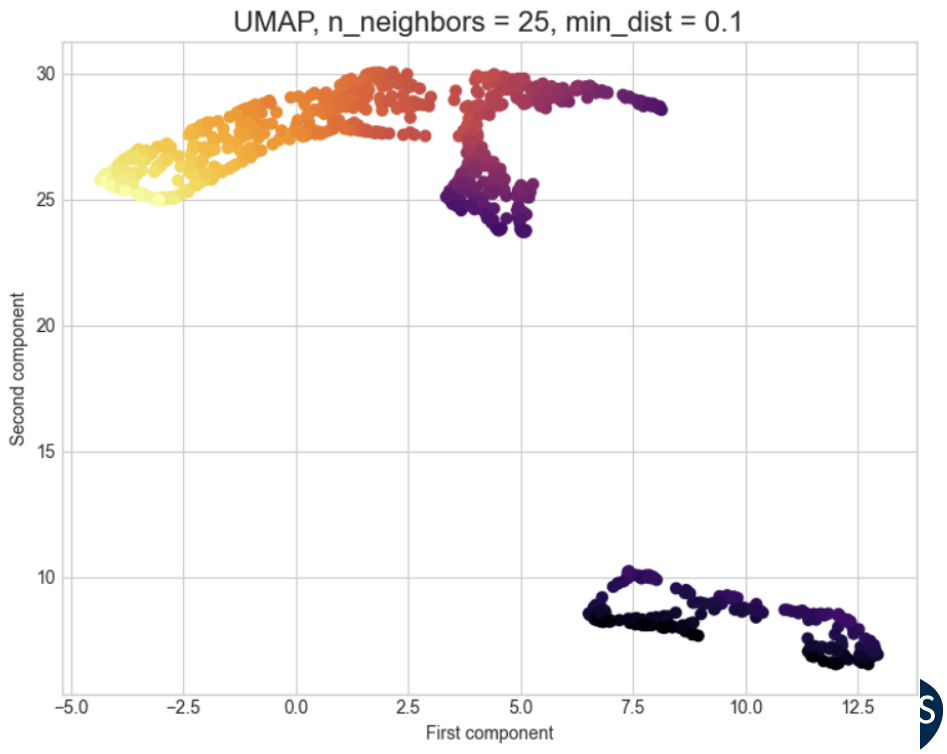




Illustration

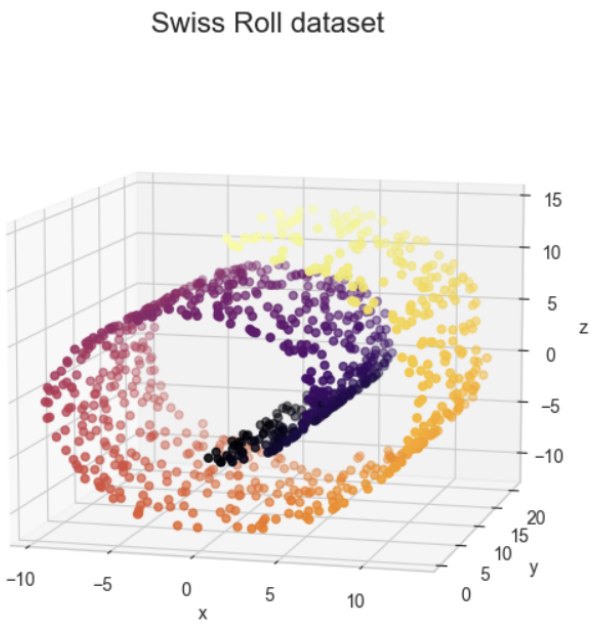
- Main parameters
  1. Number of components `n_components`
  2. Number of neighbors `n_neighbors` (The most similar parameter to t-SNE's perplexity)
  3. Minimal distance `min_dist`

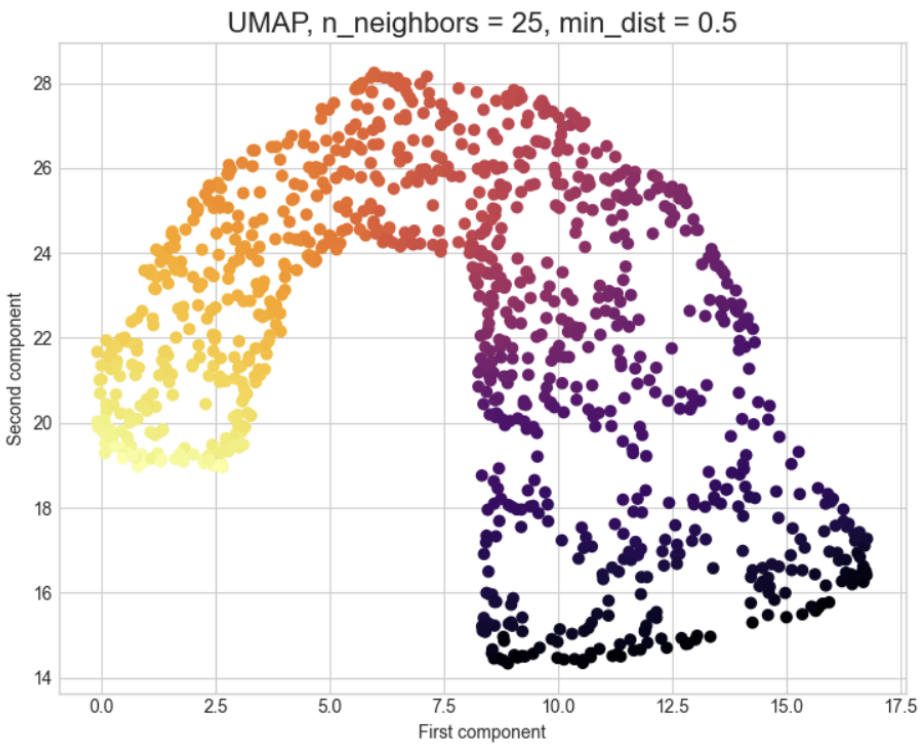




Illustration

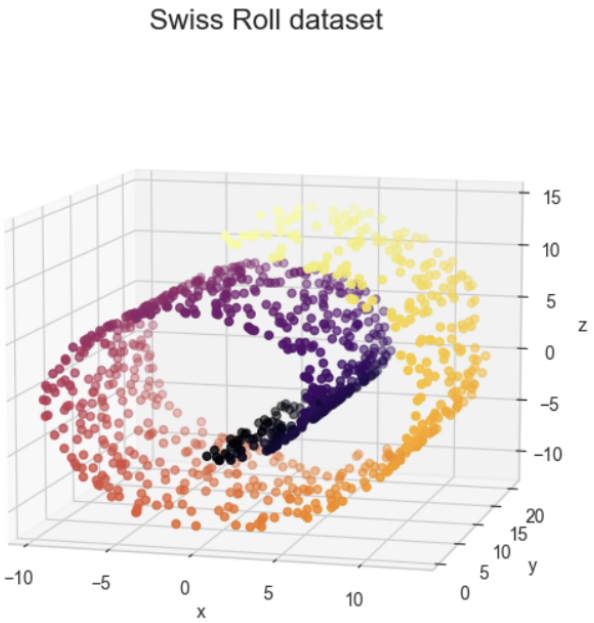
- Main parameters
  1. Number of components `n_components`
  2. Number of neighbors `n_neighbors` (The most similar parameter to t-SNE's perplexity)
  3. Minimal distance `min_dist`

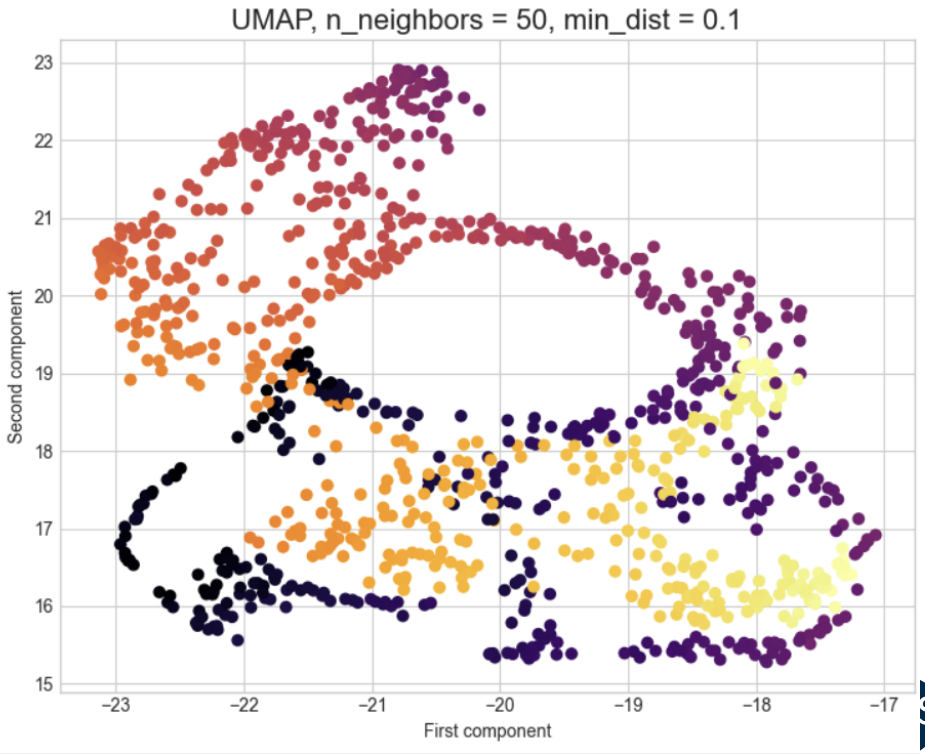




Illustration

- Main parameters
  1. Number of components `n_components`
  2. Number of neighbors `n_neighbors` (The most similar parameter to t-SNE's perplexity)
  3. Minimal distance `min_dist`





Illustration

- Main parameters
  1. Number of components `n_components`
  2. Number of neighbors `n_neighbors` (The most similar parameter to t-SNE's perplexity)
  3. Minimal distance `min_dist`