MO433 - Unsupervised Learning
Dimensionality Reduction and Data Visualization

Alexandre Xavier Falcão

Institute of Computing - UNICAMP

afalcao@ic.unicamp.br

Introduction

Given an unlabeled dataset with N samples $\mathrm{x}_i \in \mathbb{R}^n$, $i = 1, 2, \ldots, N$, where $n \gg 3$.

Direct visualization is impossible, so dimensionality reduction from $\mathrm{x} \in \mathbb{R}^n$ to $\mathrm{z} \in \mathbb{R}^d$, $d \ll n$, is needed to:

- visualize the structure of the data and its PDF, when $d \in \{2, 3\}$, for better understanding and user interaction, and
- uncover latent structure of the data for more effective processing and analysis.

Introduction

Given an unlabeled dataset with N samples $\mathrm{x}_i \in \mathbb{R}^n$, $i = 1, 2, \ldots, N$, where $n \gg 3$.

Direct visualization is impossible, so dimensionality reduction from $\mathrm{x} \in \mathbb{R}^n$ to $\mathrm{z} \in \mathbb{R}^d$, $d \ll n$, is needed to:

- visualize the structure of the data and its PDF, when $d \in \{2, 3\}$, for better understanding and user interaction, and
- uncover latent structure of the data for more effective processing and analysis.

However, how does dimensionality reduction impact the PDF transformation from $p(\mathrm{x})$ to $p(\mathrm{z})$?

Agenda

- Linear methods:
  - PCA: Maximizes variance preservation.
  - MDS: Preserves pairwise distances.
- Non-linear methods:
  - t-SNE: Preserves local neighborhoods.
  - UMAP: Preserves topological structure.

Agenda

- Linear methods:
  - PCA: Maximizes variance preservation.
  - MDS: Preserves pairwise distances.
- Non-linear methods:
  - t-SNE: Preserves local neighborhoods.
  - UMAP: Preserves topological structure.

Methods based on neural networks are left to other lectures.

PCA: Principal Component Analysis

Let $\mathrm{X} \in \mathbb{R}^{N \times n}$ be the data matrix, with each row containing a sample $x_i \in \mathbb{R}^n$. Its sample mean vector $\boldsymbol{\mu} \in \mathbb{R}^n$, centered data matrix $\mathrm{X}_c \in \mathbb{R}^{N \times n}$, and sample covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$ are defined as

$$\boldsymbol{\mu} = \frac{1}{N} \sum_{i=1}^{N} \mathrm{x}_i = \frac{1}{N} \mathrm{X}^T 1_N,$$

$$\mathrm{X}_c = \mathrm{X} - 1_N \boldsymbol{\mu}^T, \text{and}$$

$$\boldsymbol{\Sigma} = \frac{1}{N-1} \mathrm{X}_c^T \mathrm{X}_c,$$

where $1_N = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{R}^{N \times 1}$.

PCA: Principal Component Analysis

PCA finds the optimal subspace $\mathbb{R}^d$ that maximizes the preserved variance through data centralization, rotation, and projection.
Objective:

$$\text{maximize } \frac{\sum_{i=1}^{d} \lambda_i}{\sum_{i=1}^{n} \lambda_i} = \text{fraction of variance explained.}$$

Method: Eigenvalue decomposition of covariance matrix $\mathbf{\Sigma} = \mathrm{V}\mathbf{\Lambda}\mathrm{V}^T$ and projection

$$\mathrm{Z} = \mathrm{X}_c\mathrm{V}_d \in \mathbb{R}^{N\times d}$$

where

- $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$,
- $\mathrm{V} = [\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_n]$ contains the eigenvectors on each column,
- $\mathrm{V}_d = [\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_d]$ contains the first d principal components.

PCA: Distribution of Projected Data

Given $\mathsf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, **after centering:**

$$\mathsf{X}_c = \mathsf{X} - 1_N \boldsymbol{\mu}^T \sim \mathcal{N}(0, \boldsymbol{\Sigma}).$$

Distribution of projected data:

$$\mathsf{Z} = \mathsf{X}_c \mathsf{V}_d \sim \mathcal{N}(0, \boldsymbol{\Lambda}_d),$$

where $\boldsymbol{\Lambda}_d = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_d)$.

Key properties:

- **Independent components:** $Z_i \sim \mathcal{N}(0, \lambda_i)$, $\mathrm{Cov}(Z_i, Z_j) = 0$ for $i \neq j$.
- **Exact Gaussian preservation:**

$$p_{\mathsf{Z}}(\mathsf{z}) = \frac{1}{(2\pi)^{d/2}\sqrt{\det(\boldsymbol{\Lambda}_d)}} \exp\left(-\frac{1}{2}\mathsf{z}^T \boldsymbol{\Lambda}_d^{-1} \mathsf{z}\right).$$

Result: For Gaussian data, PCA achieves perfect distributional preservation!

PCA: Non-Gaussian Distributions

Linear transformation preserves

- **first two moments:** $\mathbb{E}[\mathrm{Z}] = 0$, $\mathrm{Cov}(\mathrm{Z}) = \mathbf{\Lambda}_d$.
- **orthogonality:** $\mathrm{Cov}(Z_i, Z_j) = 0$.

What is NOT preserved.

- **Higher-order moments:** Skewness, kurtosis may change.
- **Multimodal structure:** Modes may be merged.
- **Non-linear dependencies:** Complex relationships are lost.
- $p_{\mathrm{Z}}(\mathrm{z}) \neq$ simple transformation of $p_{\mathrm{X}}(\mathrm{x})$.

Central Limit Effect:

$$Z_i = \mathrm{v}_i^T \mathrm{X}_c = \sum_{j=1}^{n} v_{ij} X_{cj} \quad \text{(linear combination)}$$

Projected components may become more Gaussian-like, but original distributional structure can be significantly distorted.

MDS: Multidimensional Scaling

MDS finds coordinates in $\mathbb{R}^d$ that preserve pairwise distances through distance matrix analysis and coordinate reconstruction.

Objective:

$$\text{minimize} \sum_{i<j} (d_{ij} - \|z_i - z_j\|)^2$$

where $d_{ij} = \|x_i - x_j\|$ are original distances.

Classical MDS: Eigenvalue decomposition of Gram matrix $G = V\Lambda V^T$ and embedding

$$Z = V_d \Lambda_d^{1/2} \in \mathbb{R}^{N \times d}$$

where Z contains the reconstructed coordinates of samples in d-dimensional space.

MDS: Multidimensional Scaling

MDS finds coordinates in $\mathbb{R}^d$ that preserve pairwise distances through distance matrix analysis and coordinate reconstruction.

Objective:

$$\text{minimize} \sum_{i<j} (d_{ij} - \|z_i - z_j\|)^2$$

where $d_{ij} = \|x_i - x_j\|$ are original distances.

Classical MDS: Eigenvalue decomposition of Gram matrix $G = V\Lambda V^T$ and embedding

$$Z = V_d \Lambda_d^{1/2} \in \mathbb{R}^{N \times d}$$

where Z contains the reconstructed coordinates of samples in d-dimensional space.In MDS, the dimension d may be $<$, $>$, or $=$ to the original dimension n.

MDS: Multidimensional Scaling

- $G = -\frac{1}{2}HD^2H$ is the double-centered Gram matrix.
- $H = I_N - \frac{1}{N}1_N 1_N^T$ is the centering matrix, centering D^2 by subtracting row means and column means, and I_N is the $N \times N$ identity matrix.
- D^2 contains squared distances: $D_{ij}^2 = d_{ij}^2$.
- $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N)$ with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$.
- **Stress** measures embedding quality:

$$\text{Stress} = \sqrt{\frac{\sum_{i<j}(d_{ij} - \hat{d}_{ij})^2}{\sum_{i<j} d_{ij}^2}}$$

where d_{ij} are original distances and $\hat{d}_{ij}$ are embedding distances. Lower stress indicates better distance preservation.

MDS: Distribution Effects

What is preserved:

- **Pairwise distances:** $\|z_i - z_j\| \approx d_{ij}$.
- **Relative positions:** Neighborhood structure maintained.
- **Global geometry:** Overall shape preserved when possible.

What changes in the PDF:

- **Local density distortion:** Volume elements stretched/compressed non-uniformly.
- **Boundary effects:** Edge regions may show artificial density patterns.
- **Dimensionality effects:**
  - If $d < n$: Information loss, potential mode merging.
  - If $d > n$: Volume expansion, density spreading.

Unlike PCA, the relationship between $p_X(x)$ and $p_Z(z)$ cannot be expressed analytically - it must be studied empirically.

t-SNE: t-Distributed Stochastic Neighbor Embedding

t-SNE finds coordinates in $\mathbb{R}^d$, $d < n$, that preserve local neighborhoods through probabilistic similarity matching.
Objective: Minimize KL divergence between p_{ij} and q_{ij}.

$$C = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

High-dimensional similarities (Gaussian):

$$p_{ij} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq l} \exp(-\|x_k - x_l\|^2 / 2\sigma_l^2)}$$

Low-dimensional similarities (t-distribution):

$$q_{ij} = \frac{(1 + \|z_i - z_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|z_k - z_l\|^2)^{-1}}$$

where σ_i is determined by a given perplexity parameter.

t-SNE: Finding σ_i for given perplexity

Goal: For each x_i, find σ_i by binary search such that the effective number of neighbors equals the target perplexity.

1. **Input:** Target perplexity Perp, tolerance tol $\leftarrow 10^{-5}$.
2. **Initialize:** $\sigma_i^{\min} \leftarrow 0$, $\sigma_i^{\max} \leftarrow +\infty$, $\sigma_i \leftarrow 1$.
3. **Repeat until convergence:**
   - Compute conditional probabilities:

$$p_{j|i} \leftarrow \frac{\exp(-\|x_i - x_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|x_i - x_k\|^2/2\sigma_i^2)}.$$

   - Compute entropy: $H_i \leftarrow -\sum_{j\neq i} p_{j|i}\log_2 p_{j|i}$.
   - Compute current perplexity: $\text{Perp}_i \leftarrow 2^{H_i}$.
   - **If** $|\text{Perp}_i - \text{Perp}| < \text{tol}$: **stop**.
   - **Else if** $\text{Perp}_i > \text{Perp}$: $\sigma_i^{\max} \leftarrow \sigma_i$, $\sigma_i \leftarrow (\sigma_i + \sigma_i^{\min})/2$.
   - **Else**: $\sigma_i^{\min} \leftarrow \sigma_i$, $\sigma_i \leftarrow (\sigma_i + \sigma_i^{\max})/2$.

t-SNE: Complete Algorithm

Input: Data $\mathsf{X} \in \mathbb{R}^{N \times n}$, perplexity, learning rate η, iterations T.

Step 1: Compute high-dimensional similarities

- For each i: find σ_i using binary search (previous slide).
- Compute conditional probabilities: $p_{j|i} = \frac{\exp(-\|\mathsf{x}_i - \mathsf{x}_j\|^2 / 2\sigma_i^2)}{\sum_k \exp(-\|\mathsf{x}_i - \mathsf{x}_k\|^2 / 2\sigma_i^2)}$.
- Symmetrize: $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$.

Step 2: Initialize low-dimensional embedding

- Random initialization: $\mathsf{z}_i \sim \mathcal{N}(0, 10^{-4}\mathsf{I})$ for $i = 1, \ldots, N$.

Step 3: Gradient descent optimization

- For $t = 1, \ldots, T$:
  1. Compute low-dim similarities: $q_{ij} = \frac{(1 + \|\mathsf{z}_i - \mathsf{z}_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|\mathsf{z}_k - \mathsf{z}_l\|^2)^{-1}}$.
  2. Compute gradient:
     $\frac{\partial C}{\partial \mathsf{z}_i} = 4 \sum_j (p_{ij} - q_{ij})(\mathsf{z}_i - \mathsf{z}_j)(1 + \|\mathsf{z}_i - \mathsf{z}_j\|^2)^{-1}$.
  3. Update: $\mathsf{z}_i \leftarrow \mathsf{z}_i - \eta \frac{\partial C}{\partial \mathsf{z}_i}$.

Output: Embedding $\mathsf{Z} \in \mathbb{R}^{N \times d}$.

t-SNE: Distribution Effects

What is preserved:

- **Local neighborhoods:** Similar samples stay close.
- **Cluster structure:** Well-separated groups enhanced.
- **Relative similarities:** p_{ij} relationships maintained locally.

What changes in the PDF:

- **Heavy-tailed distribution:** t-distribution creates more space for distant samples.
- **Enhanced cluster separation:** Between-cluster distances artificially inflated.
- **Compressed within-cluster density:** Samples within clusters pulled together.
- **Global structure lost:** Large-scale relationships distorted.
- **Non-metric embedding:** Distances in $\mathbb{R}^d$ not meaningful.

Like MDS, the relationship between $p_X(x)$ and $p_Z(z)$ cannot be expressed analytically and depends heavily on perplexity choice.

UMAP: Uniform Manifold Approximation and Projection

UMAP finds coordinates in $\mathbb{R}^d$ that preserve topological structure.

Objective: Minimize cross-entropy between fuzzy set memberships.

$$C = \sum_{ij} w_{ij} \log\left(\frac{w_{ij}}{v_{ij}}\right) + (1 - w_{ij}) \log\left(\frac{1 - w_{ij}}{1 - v_{ij}}\right).$$

High-dimensional fuzzy membership:

$$w_{ij} = \exp\left(-\frac{\max(0, d_{ij} - \rho_i)}{\sigma_i}\right).$$

Low-dimensional membership (uniform distribution):

$$v_{ij} = \frac{1}{1 + a\|z_i - z_j\|^{2b}},$$

where $d_{ij} = \|x_i - x_j\|$, ρ_i is distance to nearest neighbor, σ_i controls local connectivity, and (a, b) are fitted to uniform distribution model.

UMAP: Finding σ_i, a, and b.

For each x_i, σ_i is obtained by binary search, such that $\sum_{j \in \text{k-neighors}} w_{ij} = \log_2(k)$.

1. **Input:** n_neighbors k, tolerance tol $= 10^{-5}$.
2. **Initialize:** $\sigma_i^{\min} \leftarrow 0$, $\sigma_i^{\max} \leftarrow +\infty$, $\sigma_i \leftarrow 1$.
3. **Repeat until convergence:**
   - Compute: $S = \sum_{j \in \text{k-neighbors}} \exp\left(-\frac{\max(0, d_{ij} - \rho_i)}{\sigma_i}\right)$.
   - **If** $|S - \log_2(k)| < \text{tol}$: **stop**.
   - **Else if** $S > \log_2(k)$: $\sigma_i^{\max} \leftarrow \sigma_i$, $\sigma_i \leftarrow (\sigma_i + \sigma_i^{\min})/2$.
   - **Else**: $\sigma_i^{\min} \leftarrow \sigma_i$, $\sigma_i \leftarrow (\sigma_i + \sigma_i^{\max})/2$.

UMAP: Finding σ_i, a, and b.

For each x_i, σ_i is obtained by binary search, such that $\sum_{j \in \text{k-neighors}} w_{ij} = \log_2(k)$.

1. **Input:** n_neighbors k, tolerance tol $= 10^{-5}$.
2. **Initialize:** $\sigma_i^{\min} \leftarrow 0$, $\sigma_i^{\max} \leftarrow +\infty$, $\sigma_i \leftarrow 1$.
3. **Repeat until convergence:**
   - Compute: $S = \sum_{j \in \text{k-neighbors}} \exp\left(-\frac{\max(0, d_{ij} - \rho_i)}{\sigma_i}\right)$.
   - **If** $|S - \log_2(k)| <$ tol: **stop**.
   - **Else if** $S > \log_2(k)$: $\sigma_i^{\max} \leftarrow \sigma_i$, $\sigma_i \leftarrow (\sigma_i + \sigma_i^{\min})/2$.
   - **Else**: $\sigma_i^{\min} \leftarrow \sigma_i$, $\sigma_i \leftarrow (\sigma_i + \sigma_i^{\max})/2$.

The parameters a and b require to solve

$$\int_0^{\text{min\_dist}} \frac{1}{1 + ax^{2b}} dx = \int_{\text{min\_dist}}^{+\infty} \frac{1}{1 + ax^{2b}} dx$$

by Levenberg-Marquardt curve fitting, where $v(x) = \frac{1}{1+ax^{2b}} = 0.5$ for distance $x = \textit{min\_dist}$ between z_i and z_j.

UMAP: Parameter Estimation for a and b

Problem: Find (a, b) such that $v(d) = \frac{1}{1+ad^{2b}}$ matches uniform distribution behavior.

Curve Fitting Approach:

1. **Construct target curve** $\phi(x)$ based on min_dist and spread
2. **Generate sample points** $(x_i, \phi(x_i))$
3. **Minimize nonlinear least squares:**

$$\min_{a,b} \sum_i \left[\frac{1}{1 + ax_i^{2b}} - \phi(x_i) \right]^2$$

Typical values: For min_dist $= 0.1$, spread $= 1.0$:

$$a \approx 1.576, \quad b \approx 0.895$$

Physical interpretation:

- a: controls attraction/repulsion balance
- b: controls decay rate (transition sharpness)

UMAP: Complete Algorithm

Input: Data $X \in \mathbb{R}^{N \times n}$, n_neighbors k, min_dist, learning rate α.

Step 1: Construct high-dimensional fuzzy simplicial set.

- For each x_i: find k-nearest neighbors and their distanceρ_i.
- Find σ_i such that $\sum_{j \in \text{neighbors}} \exp\left(-\frac{\max(0, d_{ij} - \rho_i)}{\sigma_i}\right) = \log_2(k)$.
- Compute: $w_{ij} = \exp\left(-\frac{\max(0, d_{ij} - \rho_i)}{\sigma_i}\right)$.
- Symmetrize: $w_{ij} \leftarrow w_{ij} + w_{ji} - w_{ij} \cdot w_{ji}$ (fuzzy set union).

Step 2: Optimize low-dimensional representation

- Find (a, b) through Levenberg-Marquardt curve fitting.
- Initialize: z_i using spectral embedding (eigenvectors of the fuzzy simplicial set).
- For each epoch: sample edges (i, j) and optimize

$$v_{ij} = \frac{1}{1 + a\|z_i - z_j\|^{2b}}$$

Update z_i using gradient descent optimization.

Output: Embedding $Z \in \mathbb{R}^{N \times d}$

UMAP: Distribution Effects

What is preserved:

- **Local neighborhoods:** Similar samples stay close (like t-SNE).
- **Global structure:** Better than t-SNE due to topological approach - connected components, holes preserved.
- **Relative distances:** More meaningful than in t-SNE.

What changes in the PDF:

- **Uniform density assumption:** UMAP constructs the low-dimensional similarities based on a uniform distribution model in dimension n.
- **Better distance preservation and smoother density transitions:** Less distortion than t-SNE.
- **Parameter-dependent structure:** n_neighbors and min_dist affect density patterns.

Like t-SNE and MDS, the relationship between $p_X(x)$ and $p_Z(z)$ cannot be expressed analytically.

Complementary literature

From `webspace.science.uu.nl/~telea001/uploads/PAPERS`, read papers:

- VAST16/paper.pdf (Visualizing the Hidden Activity of Artificial Neural Networks).
- CCIS21/paper.pdf (Improving Deep Learning Projections by Neighborhood Analysis).
- Inf23/paper.pdf (Quantitative and Qualitative Comparison of 2D and 3D Projection Techniques for High-Dimensional Data).
- CAG23/paper.pdf (Measuring the Quality of Projections of High-dimensional Labeled Data).
- SN23/paper4.pdf (Stabilizing and Simplifying Sharpened Dimensionality Reduction Using Deep Learning).