UCLouvain - EPL - LDATA2010 - Exercise session

Dimensionality reduction

1 Introduction

The general question that we will study in this statement is how can we detect structure in unsupervised, high-dimensional (HD) data. By structure we mean potential relationships existing between the samples or the features. Are the examples organized in a bunch of clusters or rather distributed on some smooth low-dimensional (LD) manifold[(1)]?

One approach to tackle this question is through visualization. Indeed if your data set is 2-dimensional, simply plotting it (one dot per example) in a 2-dimensional space (one axis per feature) helps you identify clusters, manifolds and relationships between the features. However as soon as you have more than 4 features, it is impossible to plot all features at once. A possible workaround is then to produce one 2-dimensional plot for each pair of features. If you have M features, you will have $M \cdot (M-1)/2$ plots. If $M = 100$, this means that you have to produce 4950 plots... Moreover these representations hardly allow you to get insights about the relationships linking more than two features.

The idea of having a 2-dimensional representation is nevertheless appealing. While the aforementioned trick using the feature pairs is unsuitable for HD data sets, it assumes that the data can not be modified. But what if we relax this assumption and modify the data to be representable using only a few (e.g. 2 or 3) dimensions? This idea gave birth to dimensionality reduction (DR). Typical domains of application are visualization, compression, preprocessing step for other algorithms to help alleviating the curse of dimensionality, etc [1, 2].

The goal will then be to reduce the data dimensionality to provide a faithful and meaningful representation of HD data in a lower-dimensional space. For sure reducing the dimension comes up with a loss of information. To minimize it, we will have to define more precisely what is interesting to preserve in the LD representation of the data, which will be termed as the LD embedding or space (LDS).

The general intuition that drives DR is that close or similar data items should be represented near each other, whereas dissimilar ones should be represented far from each other [1]. This means that in the LD embedding, we would like the neighborhoods around each point to be as similar as possible to the ones in the HD space (HDS). Through the DR history, authors have formalized this idea of neighborhood preservation in various ways, using several linear and nonlinear models for the mapping of data from the HDS to the LDS.

This document briefly covers:

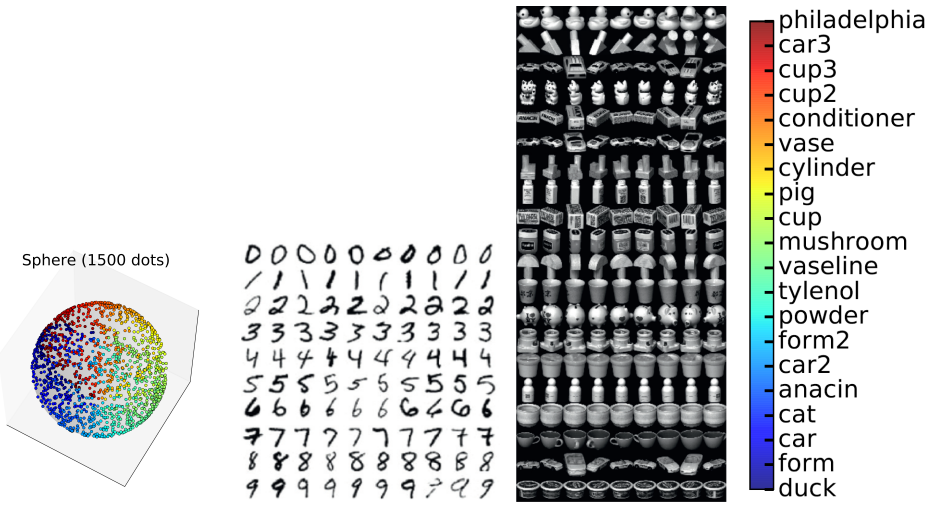
1. Intrinsic dimensionality estimation, which aims to discover the minimum number of dimension which are necessary to describe a data set;
2. A DR algorithm from the distance preservation paradigm, in which the LD embedding aims to preserve the pairwise distances between the samples in the HDS;
3. A DR algorithm from the similarity-based paradigm, which defines the LD embedding by matching similarities between the samples in both spaces;
4. DR quality assessment, which aims at evaluating the quality of a LD embedding with respect to the HDS.

We will denote by $\boldsymbol{\Xi} = [\boldsymbol{\xi}_i]_{i=1}^N$ a set of N points in some M-dimensional metric space, termed as the HDS. Let $\mathbf{X} = [\mathbf{x}_i]_{i=1}^N$ be its representation in a P_{dr}-dimensional metric space (LDS), with $P_{\text{dr}} \leq M$. In the following, we will search for 2-dimensional embeddings: $P_{\text{dr}} = 2$. The distance between the i^{th} and j^{th} points in the HDS (resp. LDS) is denoted by δ_{ij} (d_{ij}). We will assume that both the HD and LD distances are Euclidean. For convenience, we denote by $\mathcal{I} = \{1, \ldots, N\}$ the set of the indexes of the samples in the data set.

The data set that we will use are:

- A 2-dimensional artificial sphere data set embedded in a 3-dimensional space ($N = 1500$, $M = 3$).
- A random subset of the MNIST test set ($N = 1500$, $M = 28^2$), which contains 1500 gray-level images of scanned handwritten digits [3]. There are 150 images per digit. The images are square, with height and width of 28 pixels.

[(1)] A manifold is a subspace embedded in another space. For instance, a 2-dimensional sphere embedded in a 3-dimensional space is a manifold. A complete definition can be found in [1].



(a) **(b)** **(c)** **(d)**

Figure 1: (a) Sphere data set. (b) Some of the MNIST images. (c) Some of the COIL-20 images. (d) Colorbar which will be used to represent the objects of the COIL-20 data set in the LD embeddings depicted in this statement.

- The COIL-20 data set ($N = 1440$, $M = 128^2$), which contains 72 gray-level images of 20 different objects [4]. The 72 images correspond to 4-degree rotations around each object. All images are square, with height and width of 128 pixels.

For the last two data sets, the images are converted in M-dimensional vectors by concatenating their pixel rows one after the other, without any preprocessing. Fig. 1 illustrates the three data sets.

Exercise 1. What relevant low-dimensional representations of these three data sets should look like?

The data sets are contained in three .mat files named *sphere*, *MNIST* and *COIL_20*. They all contain:

$X_{\mathbf{hds}}$: a matrix containing the HD data set. It has one row per sample and one column per feature.

t : a vector with as many samples as elements. It provides a label for each sample. It should **NOT** be used to perform the dimensionality reduction, but only to represent the samples in the LDS (e.g. by using colormaps). For the MNIST data set, the i$^{\text{th}}$ element gives the digit of the image stored in the i$^{\text{th}}$ row of X_{hds}. For the COIL data set, it gives the index of the object of the image stored in the i$^{\text{th}}$ row of X_{hds}. For the sphere data set, it gives the longitude computed from the coordinates stored in the i$^{\text{th}}$ row of X_{hds}. It determines the colors of the points in Fig 1a. The LD embeddings of the Sphere data presented in this statement depict each sample using the same color as in Fig 1a.

In addition, *COIL_20.mat* contains a variable *obj_name*. Its gives the names of the 20 objects. The name of the object associated with the i$^{\text{th}}$ row of X_{hds} is then the t_i^{th} name stored in *obj_name*, where t_i is the i$^{\text{th}}$ element of t.

2 Intrinsic dimensionality estimation

The intrinsic dimensionality of a data set is, intuitively, the minimum number of variable needed to describe it.

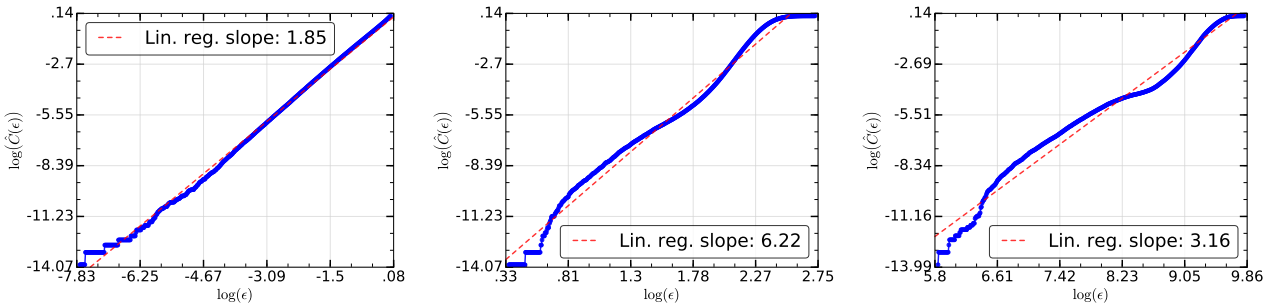
Exercise 2. What is the intrinsic dimensionality of the sphere data set?

Despite its simplicity, this definition is not rigorous enough to deal with any geometrical objects. More formal definitions can be found in [1]. We will in particular focus on one of them: the correlation dimension d_{cor}. It is derived by introducing the quantity $C(\epsilon)$. When looking at the data set on the scale of a single point, it measures the number of neighboring points lying closer than a certain threshold ϵ. This number grows as a length for a 1D manifold (embedded in some M-dimensional space), as a surface for a 2D manifold, as a volume for a 3D manifold, and so forth. Generalizing for a P-dimensional manifold gives

$$C(\epsilon) \propto \epsilon^P \Rightarrow P \propto \frac{\log C(\epsilon)}{\log \epsilon}.$$

The correlation dimension is then:

$$d_{\text{cor}} = \lim_{\epsilon \to 0} \frac{\log C(\epsilon)}{\log \epsilon}.$$



(a) Sphere. **(b)** MNIST. **(c)** COIL.

Figure 2: Correlation dimension estimation.

The above limit can not be computed on discrete data sets. However by applying l'Hospital's rule, one can show that

$$d_{\text{cor}} = \lim_{\epsilon_1,\epsilon_2 \to 0} \frac{\log C(\epsilon_2) - \log C(\epsilon_1)}{\log \epsilon_2 - \log \epsilon_1}.$$

Approximating $C(\epsilon)$ by the proportion $\hat{C}(\epsilon)$ of the pairwise distances in the data set which are $\leq \epsilon$, the scale-dependent correlation dimension is

$$\hat{d}_{\text{cor}}(\epsilon_1, \epsilon_2) = \frac{\log \hat{C}(\epsilon_2) - \log \hat{C}(\epsilon_1)}{\log \epsilon_2 - \log \epsilon_1}.$$

It is practically computed as the average slope of the curve in a log-log plot of $\hat{C}(\epsilon)$ versus ϵ [1]. The latter ranges between the smallest and largest pairwise distances in the data set. The whole procedure to estimate the correlation dimension is then:

1. Compute all the pairwise Euclidean distances between the samples in $\boldsymbol{\Xi}$ and remove the zero ones.
2. For a number (e.g. $\lfloor N/3 \rceil$) of ϵ between the smallest and largest distances, evaluate the proportion $\hat{C}(\epsilon)$ of the pairwise distances which are $\leq \epsilon$. The ϵ values will be such $\log \epsilon$ is uniformly spaced between $\log\left(\min_{i\in\mathcal{I}, j\in\mathcal{I}\setminus\{j\}} \delta_{ij}\right)$ and $\log\left(\max_{i\in\mathcal{I}, j\in\mathcal{I}\setminus\{j\}} \delta_{ij}\right)$.
3. Plot $\log\left(\hat{C}(\epsilon)\right)$ against $\log \epsilon$ and compute a linear regression between them. Its slope is the estimated correlation dimension.

The results on our three data sets is depicted by Fig. 2.

Exercise 3. Estimate the correlation dimension of the three data sets and produce three figures such as in Fig. 2.

Exercise 4. Discuss the correlation dimension estimates obtained in Fig. 2. Why the estimation for the COIL data set is much smaller than for the MNIST one?

3 Distance preservation paradigm

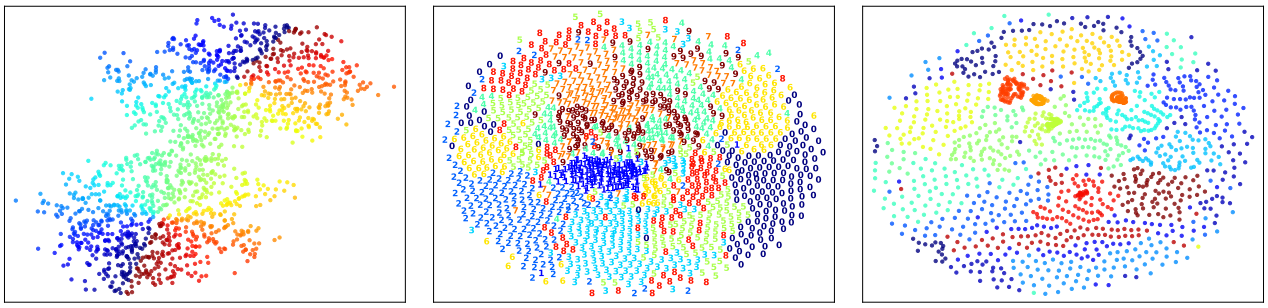
Distance preservation-based DR algorithms derive the LD embedding by trying to reproduce in the LDS the pairwise distances observed in the HDS. A very large number of DR methods were designed based on this idea. We will specifically focus on Curvilinear Component Analysis (CCA) [5]. It computes the LD coordinates $\mathbf{x}_i$, $i \in \mathcal{I}$, by minimizing the cost function

$$C_{\text{CCA}}(\mathbf{X}) = \sum_{i\in\mathcal{I}, j\in\mathcal{I}\setminus\{i\}} (\delta_{ij} - d_{ij})^2 H(\lambda - d_{ij}),$$

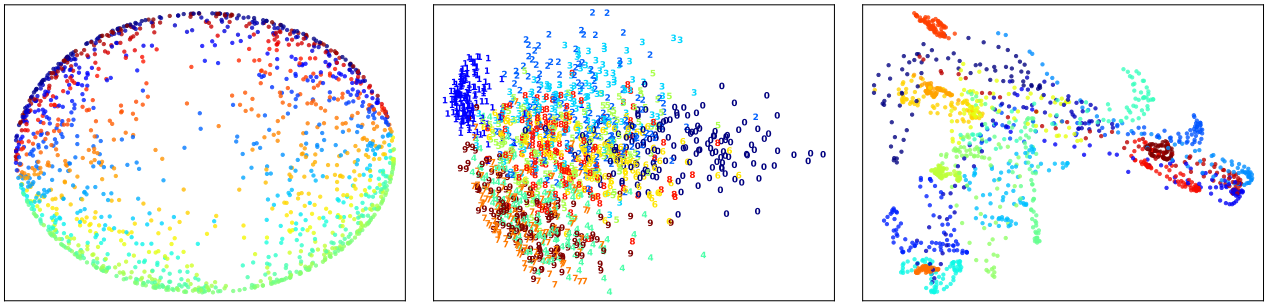
where H is a step function:

$$H(\lambda - d_{ij}) = \begin{cases} 1 & \text{if } d_{ij} \leq \lambda, \\ 0 & \text{otherwise.} \end{cases}$$

The hyper-parameter λ can be tuned according to several heuristics [5]. Starting from some LD coordinates (e.g. randomly distributed), the cost function C_{CCA} is minimized by gradient descent variants. Fig. 3 presents the results of its application on the three data sets, as well as their PCA projection on their first two principal components.



(a) Sphere - CCA. **(b)** MNIST - CCA. **(c)** COIL - CCA.



(d) Sphere - PCA. **(e)** MNIST - PCA. **(f)** COIL - PCA.

Figure 3: Two-dimensional embeddings resulting from applying CCA and PCA on the data sets.

Exercise 5. Discuss the results depicted in Fig. 3. In particular,

1. How is the PCA projection of the sphere data set different from the CCA one? Why does the PCA crush the sphere while CCA tend to unfold it?
2. Is CCA able to tear smooth manifolds such as the sphere data set? Motivate your answer by analyzing C_{CCA}.
3. Can you visualize the presumed clusters of digits in the MNIST data set? How can you explain the result of Fig. 3e?
4. Can you identify the presumed smooth, one-dimensional manifolds in the COIL data set?

4 Similarity-based dimensionality reduction

Although being popularly used for many years, distance-preserving DR methods tend to be sensitive to the curse of dimensionality, and in particular to the norm concentration phenomenon, when applied on very high-dimensional spaces. On the other side, more recent similarity-based DR schemes circumvent the curse of dimensionality thanks to their shift-invariance property, which will be detailed during the lecture. Stochastic Neighbor Embedding (SNE) [2] first introduced this new paradigm. Starting from pairwise distances, similarities between pairs of data points can be defined in the high- and low-dimensional spaces for all $i \in \mathcal{I}$, $j \in \mathcal{I}\backslash\{i\}$:

$$\sigma_{ij} = \frac{\exp\left(-\pi_i \delta_{ij}^2/2\right)}{\sum_{k\in\mathcal{I}\backslash\{i\}} \exp\left(-\pi_i \delta_{ik}^2/2\right)}, \qquad \sigma_{ii} = 0, \qquad s_{ij} = \frac{\exp\left(-p_i d_{ij}^2/2\right)}{\sum_{k\in\mathcal{I}\backslash\{i\}} \exp\left(-p_i d_{ik}^2/2\right)}, \qquad s_{ii} = 0.$$

The normalized Gaussian functions centered on $\boldsymbol{\xi}_i$ and $\mathbf{x}_i$ have π_i and p_i as precisions. π_i is determined by binary search in such a way that the perplexity of the distribution $\boldsymbol{\sigma}_i = [\sigma_{ij},\ j \in \mathcal{I}\backslash\{i\}]$ equals a unique user-defined value $K_\star$:

$$\pi_i \text{ s.t. } \log K_\star = -\sum_{j\in\mathcal{I}\backslash\{i\}} \sigma_{ij} \log \sigma_{ij}.$$

The $K_\star$ can be interpreted as a measure of soft neighborhood size (i.e. a typical number of neighbors around each point). On the other hand, p_i is set to 1.

Exercise 6. How will π_i adapt in sparser and denser regions of the feature space?

Exercise 7. Assuming that $p_i = \beta$ for all $i \in \mathcal{I}$, why can we set $\beta = 1$ without loss of generality (at least when d_{ij} is Euclidean)?

In order to find the low-dimensional coordinates of the data points, SNE minimizes the sum of the KL divergences between the distributions $\boldsymbol{\sigma}_i$ and $\boldsymbol{s}_i = [s_{ij},\ j \in \mathcal{I}\backslash\{i\}]$ for each $i \in \mathcal{I}$:

$$C_{\text{SNE}} = \sum_{\substack{i\in\mathcal{I}\\ j\in\mathcal{I}\backslash\{i\}}} \sigma_{ij} \log \frac{\sigma_{ij}}{s_{ij}}, \tag{1}$$

where C_{SNE} is SNE cost function. Gradient-based optimization methods are then typically used.

Exercise 8. Does C_{SNE} equally weight the different types of DR errors (i.e. close HD data points represented far in the LDS, and far HD data points represented close in the LDS)?

Many recent DR algorithms were designed on the top of SNE. In particular, t-SNE modifies SNE by symmetrizing the HD similarities and by defining the LD ones using a Student t-distribution with one degree of freedom [6]:

$$\sigma_{ij,t} = \frac{\sigma_{ij} + \sigma_{ji}}{2N}, \qquad s_{ij,t} = \frac{\left(1 + d_{ij}^2\right)^{-1}}{\sum_{\substack{k,l\in\mathcal{I}\\ k\neq l}} \left(1 + d_{kl}^2\right)^{-1}}. \tag{2}$$

The t-SNE cost function is the same as SNE one: $C_{t\text{-SNE}} = C_{\text{SNE}}$.

Exercise 9. The Student t-distribution has a heavier tail than the Gaussian distribution. What will be the consequences of using a heavy-tailed distribution in the LDS?

The two-dimensional embeddings resulting from applying t-SNE on the data sets for various perplexities $K_\star$ are depicted in Fig. 4 and 5.

Exercise 10. Describe the observed effect of the perplexity $K_\star$. Is there some value which can suit all data sets?

Exercise 11. Describe the embeddings produced by t-SNE on the sphere data set for small perplexities. Why does t-SNE introduce some artificial clusters? Why is the problem solved using larger perplexities?

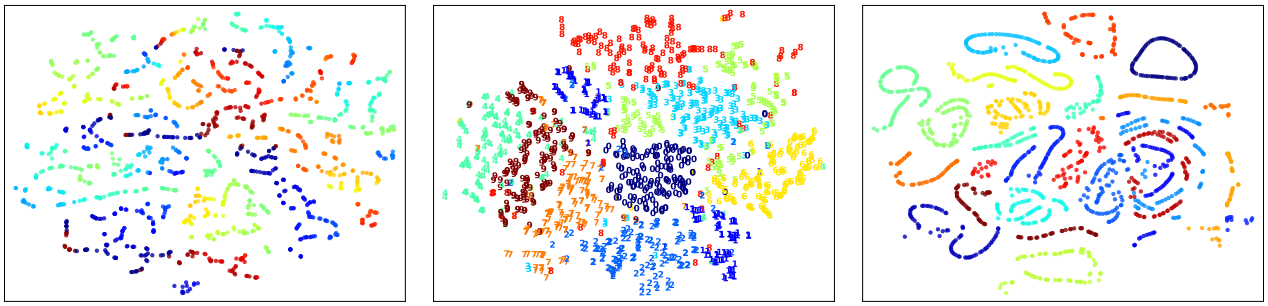
Exercise 12. Can you identify clusters of digits in the MNIST data sets and one-dimensional manifolds for the COIL-20 one? For which perplexities?

Many variants of SNE and t-SNE have been proposed during the last few years. Some of them will be described during the lecture too.

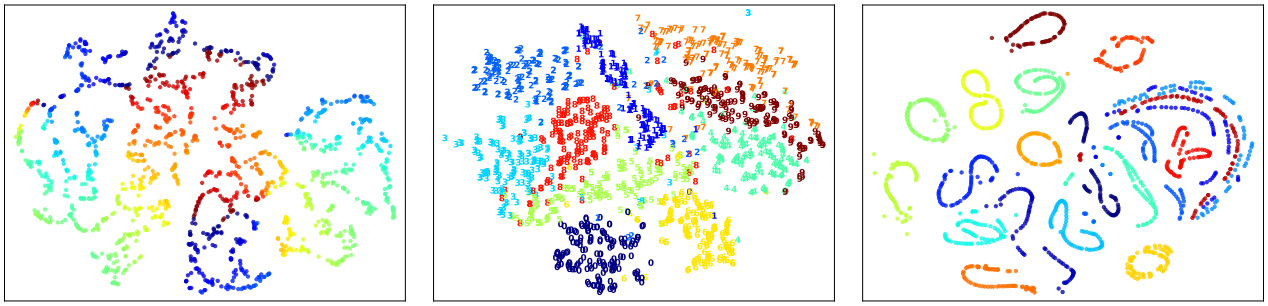
Thanks to its popularity, t-SNE have been implemented in different programming languages. In particular,

- In **Matlab**, you can use the `tsne.m` function in the folder `tSNE_matlab`. Use X_{hds} for the `X` argument, 2 for `no_dims` and some perplexity for `perplexity`.
- In **Python**, you can use the `TSNE` function of the module `sklearn.manifold`. Set the `perplexity` argument to the perplexity you want, `n_components` to 2 and `method` to 'exact'. You can let the other argument to their default values, or explore their effects.
- In **R**, you can use the `tsne` function of the package `tsne`. Set `k` to 2, *initial_dims* to the number of features of your data set, `perplexity` to your perplexity and `whiten` to FALSE. You can let the other argument to their default values, or explore their effects.

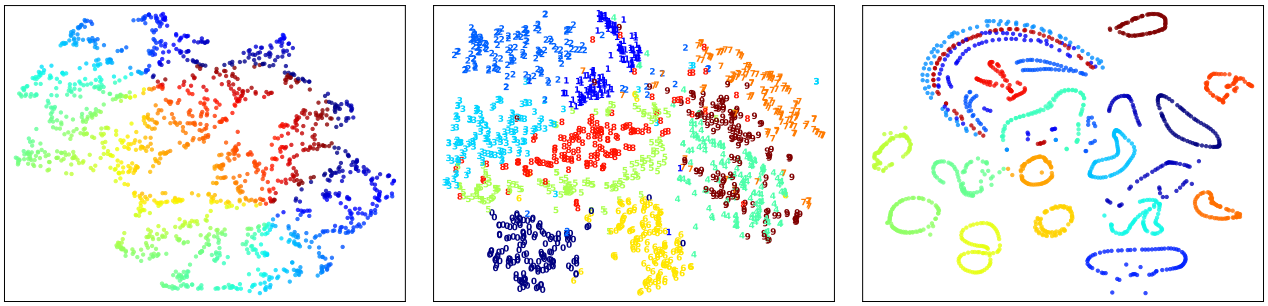
Exercise 13. Using either Matlab, Python or R, apply t-SNE on the data sets, using different perplexities. In particular, try other perplexities than the ones depicted in Fig. 4 and 5. $K_\star$ should always remain between 2 and $N - 1$. You can use the t variable provided with each data set to color the points in your LD embeddings.



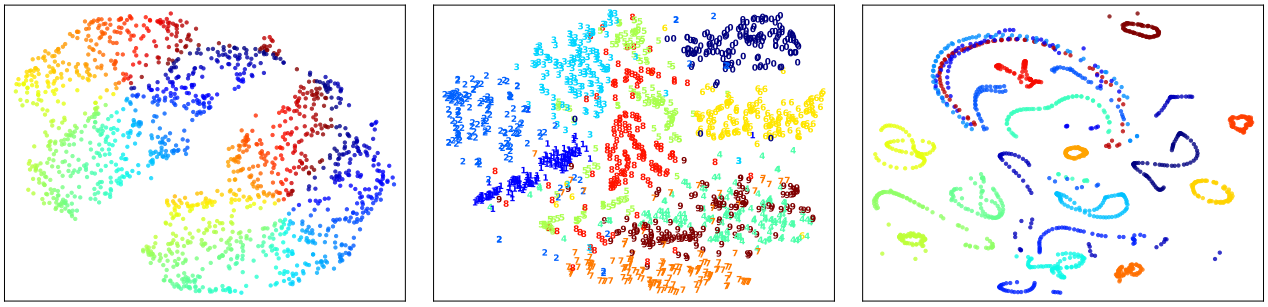
(a) Sphere - $K_\star = 5$. **(b)** MNIST - $K_\star = 5$. **(c)** COIL - $K_\star = 5$.



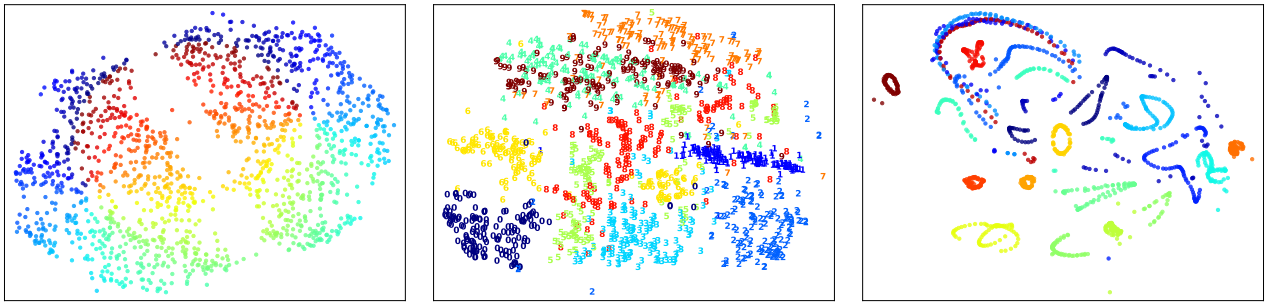
(d) Sphere - $K_\star = 15$. **(e)** MNIST - $K_\star = 15$. **(f)** COIL - $K_\star = 15$.



(g) Sphere - $K_\star = 30$. **(h)** MNIST - $K_\star = 30$. **(i)** COIL - $K_\star = 30$.

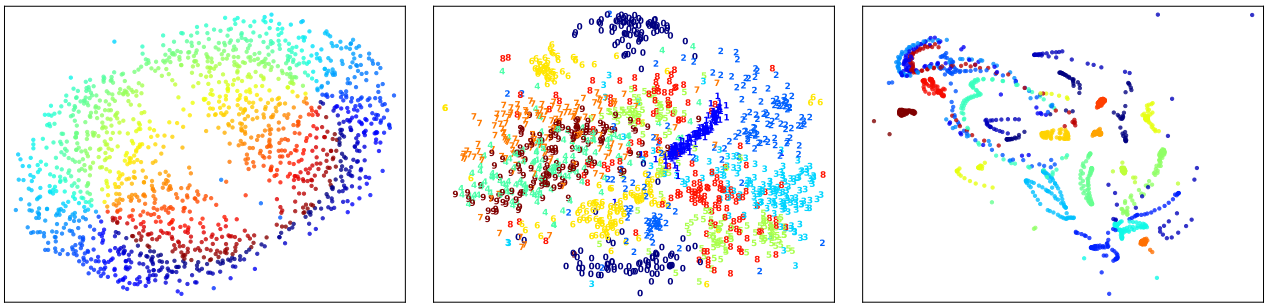


(j) Sphere - $K_\star = 75$. **(k)** MNIST - $K_\star = 75$. **(l)** COIL - $K_\star = 75$.

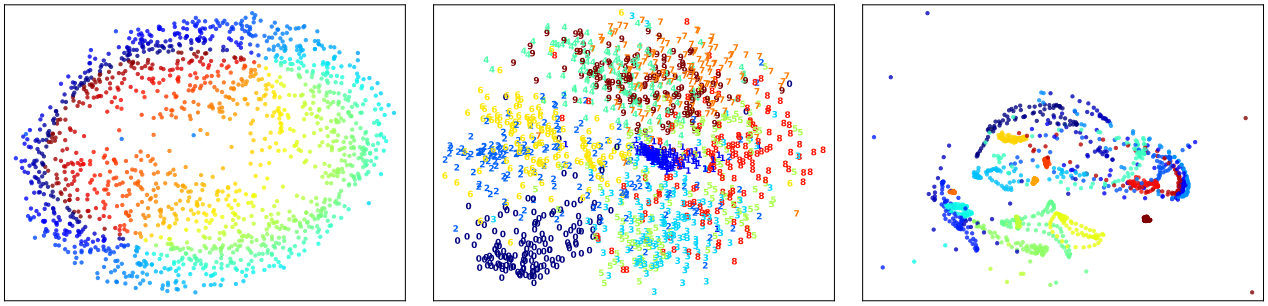


(m) Sphere - $K_\star = 150$. **(n)** MNIST - $K_\star = 150$. **(o)** COIL - $K_\star = 150$.

Figure 4: Two-dimensional embeddings resulting from applying t-SNE on the data sets with different perplexities.



(a) Sphere - $K_\star = 500$. **(b)** MNIST - $K_\star = 500$. **(c)** COIL - $K_\star = 500$.



(d) Sphere - $K_\star = 1000$. **(e)** MNIST - $K_\star = 1000$. **(f)** COIL - $K_\star = 1000$.

Figure 5: Two-dimensional embeddings resulting from applying t-SNE on the data sets with different perplexities.

5 Quality assessment

Although producing 2-D embeddings of a data set is visually pleasant, describing whether one visualization is *better* than the other is not straightforward (e.g. Fig. 4f and 4i). This motivates the development of quality scores assessing the DR performances. Since DR is mainly unsupervised, one must first define a criterion according to which LD embeddings may be compared. A first idea would be to choose the cost function of some DR algorithm and rank the different DR results according to the cost function values. However, this obviously favors the DR algorithm optimizing the considered cost function directly. For this reason, some studies developed DR quality criteria measuring the HD neighborhood preservation in the LDS [7]. Due to its discrete nature, it is hard to directly optimize the neighborhood preservation using conventional tools, but nothing prevents to employ it in order to quantify the DR performances.

Let ν_i^K and n_i^K index the K nearest neighbors of $\boldsymbol{\xi}_i$ and $\mathbf{x}_i$ in the HDS and LDS. Their average normalized agreement can be expressed as

$$Q_{\mathrm{NX}}(K) = \frac{1}{N} \sum_{i \in \mathcal{I}} \frac{|\nu_i^K \cap n_i^K|}{K} \in [0, 1] .$$

Exercise 14. What is the value of $Q_{\mathrm{NX}}(N-1)$?

Exercise 15. Show that, for random LD points with uniform distribution, the expectation of $Q_{\mathrm{NX}}(K)$ equals $\frac{K}{N-1}$.

This indicates that $Q_{\mathrm{NX}}(K)$ naturally tends to grow with K. In order to be able to compare different neighborhood sizes, one can define

$$R_{\mathrm{NX}}(K) = \frac{(N-1)\, Q_{\mathrm{NX}}(K) - K}{N-1-K} .$$

Exercise 16. What does a negative $R_{\mathrm{NX}}(K)$ mean?

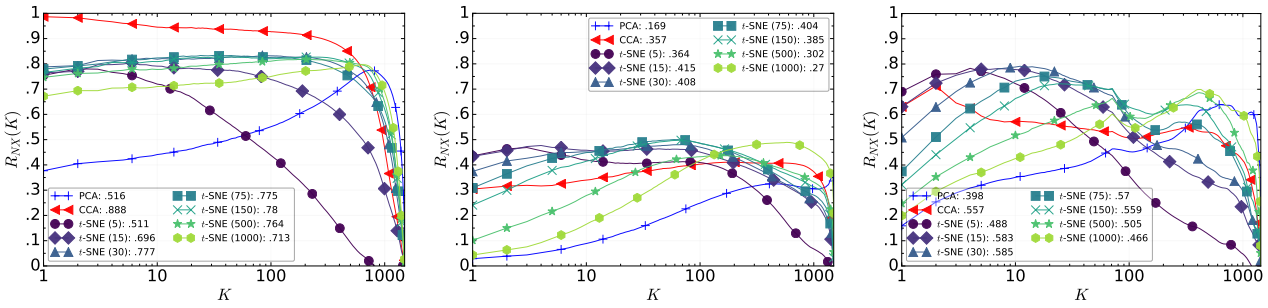
As closer neighbors typically prevail, $R_{\mathrm{NX}}(K)$ is often displayed with a log-scale for K. The area under the resulting curve

$$\mathrm{AUC} = \frac{\sum_{K=1}^{N-2} \frac{R_{\mathrm{NX}}(K)}{K}}{\sum_{K=1}^{N-2} \frac{1}{K}} \in [-1, 1]$$

grows with the DR quality, quantified at all scales with an emphasis on small ones. It can be used as a scalar score to quantify the quality of an LD embedding.

Exercise 17. Fig. 6 presents the quality assessment of the LD embeddings previously obtained with PCA, CCA and t-SNE on the three data sets.

1. Can you identify a best performing method for each data set?
2. Describe the effect of the perplexity $K_\star$ on the neighborhood preservation.
3. How does PCA and CCA behave in terms of neighborhood preservation? Which neighborhood sizes do they best preserve? Are the results of PCA and CCA similar to the ones of t-SNE for some perplexity?
4. Do the methods achieve similar performances on the three data sets? I.e., are the neighborhoods equally easy or difficult to reproduce in the LD space for each data set? Is there a link with the (intrinsic) dimensionality of the corresponding HD spaces?
5. Which method would you recommend to respectively preserve small, medium and large neighborhoods?



(a) Sphere. **(b)** MNIST. **(c)** COIL.

Figure 6: Quality assessment of the LD embeddings of Fig. 3, 4 and 5. The $R_{\mathrm{NX}}(K)$ curves are displayed and the corresponding AUC values are indicated in the legend of the figures, right next to the method names. The perplexity $K_\star$ for t-SNE is indicated in parentheses in the legend.

Many other DR algorithms have been proposed in order to improve the neighborhood preservation. Some of them will be described during the lectures.

References

[1] J. A. Lee and M. Verleysen, *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2007.

[2] G. Hinton and S. Roweis, "Stochastic neighbor embedding," in *NIPS*, vol. 15, pp. 833–840, 2002.

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[4] S. A. Nene, S. K. Nayar, H. Murase, *et al.*, "Columbia object image library (coil-20)," 1996. Technical report CUCS-005-96.

[5] P. Demartines and J. Hérault, "Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets," *IEEE Transactions on neural networks*, vol. 8, no. 1, pp. 148–154, 1997.

[6] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[7] J. A. Lee and M. Verleysen, "Quality assessment of dimensionality reduction: Rank-based criteria," *Neurocomputing*, vol. 72, no. 7, pp. 1431–1443, 2009.