Data Visualization

**Laboratory work №8**

# Nonlinear Dimension Reduction

Dimensionality reduction techniques are employed in data science to transform high-dimensional data into a lower-dimensional space while preserving the underlying relationships between data points. This is beneficial for visualization, data analysis, and machine learning algorithms that struggle with high-dimensional data.

**8.1. Multidimensional Scaling (MDS)**

MDS aims to minimize a stress function, typically squared error, that measures the difference between the original pairwise distances (*dij*) and the distances (*d'ij*) in the lower-dimensional embedding. Here's a basic formula for the stress function:

 (1)

where *i* and *j* iterate over all data points. MDS uses optimization algorithms to iteratively adjust the low-dimensional coordinates to minimize this stress function.

**8.2. Isomap**

Isomap leverages geodesic distances, which are the shortest path distances between points along the underlying manifold. It first constructs a nearest neighbors graph to capture the local structure. Here's the formula for a typical k-nearest neighbors graph:

Connect points *pi* and *pj* if pj is among the *k* nearest neighbors of *pi* (and vice versa).

Then, Isomap computes geodesic distances using Floyd-Warshall algorithm or other shortest path algorithms. Finally, classical MDS is applied to the geodesic distance matrix to obtain the low-dimensional representation.

**8.3. t-Distributed Stochastic Neighbor Embedding (t-SNE)**

t-SNE defines a probability distribution (*pij*) in the high-dimensional space that represents the likelihood of data point *pi* being a neighbor of *pj*. A common choice is the Gaussian distribution:

 (2)

where σ is a hyperparameter controlling the neighborhood size. t-SNE then defines a similar probability distribution (*qij*) in the low-dimensional embedding and uses Kullback-Leibler divergence to measure the difference between these distributions:

 (3)

t-SNE optimizes this divergence to ensure the low-dimensional embedding retains the local structure of the high-dimensional data.

**8.4 Python implementation**

One of the implementations of manifold learning algorithm is implemented by scikit-learn library <https://scikit-learn.org/stable/modules/manifold.html>/

There are two basic examples of using such methods:

1. For S-like dataset <https://scikit-learn.org/stable/auto_examples/manifold/plot_compare_methods.html>
2. For MNIST-dataset <https://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html>

# Variants of tasks

1. Download dataset according your variant   
   <https://github.com/a-vodka/dv/tree/master/lab/dataset>
2. Perform dimension reduction using Isomap, Locally Linear Embedding, Multi-dimensional Scaling (MDS), t-distributed Stochastic Neighbor Embedding (t-SNE) onto 2D and 3D space.
3. Analyze results. Find, which methods gives best results for your dataset.