Lecture 13: Even more dimension reduction techniques

Felix Held, Mathematical Sciences

MSA220/MVE440 Statistical Learning for Big Data

10th May 2019

CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Recap: kernel PCA

Given a set of m-dimensional feature vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and a kernel $k(\mathbf{x}, \mathbf{y})$, form the Gram matrix $\mathbf{K} = (k(\mathbf{x}_i, \mathbf{x}_j))_{ij}$ and perform

- Solve the eigenvalue problem $\mathbf{K}\mathbf{a}_i = \lambda_i n \mathbf{a}_i$ for λ_i and $\mathbf{a}_i$
- Scale $\mathbf{a}_i$ such that

$$\mathbf{a}_i^T \mathbf{K} \mathbf{a}_i = 1$$

The projection of a feature vector $\mathbf{x}$ onto the i-th principal component in the implicit space of the $\boldsymbol{\phi}(\mathbf{x})$ is

$$\eta_i(x) = \sum_{l=1}^{n} a_{il} k(\mathbf{x}, \mathbf{x}_l)$$

Centring and kernel PCA

- The derivation assumed that the implicitly defined feature vectors $\boldsymbol{\phi}(\mathbf{x}_l)$ were centred. **What if they are not?**
- In the derivation we look at scalar products $\boldsymbol{\phi}(\mathbf{x}_i)^T \boldsymbol{\phi}(\mathbf{x}_l)$. Centring in the implicit space leads to

$$\left(\boldsymbol{\phi}(\mathbf{x}_i) - \frac{1}{n}\sum_{j=1}^{n}\boldsymbol{\phi}(\mathbf{x}_j)\right)^T \left(\boldsymbol{\phi}(\mathbf{x}_l) - \frac{1}{n}\sum_{j=1}^{n}\boldsymbol{\phi}(\mathbf{x}_j)\right) =$$

$$K_{il} - \frac{1}{n}\sum_{j=1}^{n} K_{ji} - \frac{1}{n}\sum_{j=1}^{n} K_{jl} + \frac{1}{n^2}\sum_{j=1}^{n}\sum_{m=1}^{n} K_{jm}$$

- Using the **centring matrix** $\mathbf{J} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T$, centring in the implicit space is equivalent to transforming $\mathbf{K}$ as

$$\mathbf{K}' = \mathbf{JKJ}$$

- Algorithm is the same, apart from using $\mathbf{K}'$ instead of $\mathbf{K}$.

Dimension reduction while preserving distances

Preserving distance

Like in cartography, the goal of dimension reduction can be subject to different sub-criteria, e.g. PCA preserves the directions of largest variance.

What if we want to **preserve the distance** while reducing the dimension?

For given vectors $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^p$ we want to find $\mathbf{y}_1, \dots, \mathbf{y}_n \in \mathbb{R}^m$ where $m < p$ such that

$$\|\mathbf{x}_i - \mathbf{x}_l\|_2 \approx \|\mathbf{y}_i - \mathbf{y}_l\|_2$$

Distance matrices and the linear kernel

Given a data matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, note that

$$\mathbf{X}\mathbf{X}^T = \begin{pmatrix} \mathbf{x}_1^T\mathbf{x}_1 & \cdots & \mathbf{x}_1^T\mathbf{x}_n \\ \vdots & & \vdots \\ \mathbf{x}_n^T\mathbf{x}_1 & \cdots & \mathbf{x}_n^T\mathbf{x}_n \end{pmatrix} = \mathbf{K}$$

which is also the **Gram matrix $\mathbf{K}$ of the linear kernel**.

Let $\mathbf{D} = (\|\mathbf{x}_l - \mathbf{x}_m\|_2)_{lm}$ be the distance matrix in the Euclidean norm. Note that

$$\|\mathbf{x}_l - \mathbf{x}_m\|_2^2 = \mathbf{x}_l^T\mathbf{x}_l - 2\mathbf{x}_l^T\mathbf{x}_m + \mathbf{x}_m^T\mathbf{x}_m$$

and (with element-wise exponentiation)

$$-\frac{1}{2}\mathbf{D}^2 = \mathbf{X}\mathbf{X}^T - \frac{1}{2}\mathbf{1}\operatorname{diag}(\mathbf{X}\mathbf{X}^T) - \frac{1}{2}\operatorname{diag}(\mathbf{X}\mathbf{X}^T)\mathbf{1}^T.$$

Through calculation it can be shown that with $\mathbf{J} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T$

$$\mathbf{K} = \mathbf{J}\left(-\frac{1}{2}\mathbf{D}^2\right)\mathbf{J}$$

Finding an exact embedding

- Can be shown that if $\mathbf{K}$ is positive semi-definite then there exists an **exact embedding** in $m = \text{rank}(\mathbf{K}) \leq \text{rank}(\mathbf{X}) \leq \min(n, p)$ dimensions.
  1. Perform PCA on $\mathbf{K} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$
  2. If $m = \text{rank}(\mathbf{K})$, set
  $$\mathbf{Y} = (\sqrt{\lambda_1}\mathbf{u}_1, \dots, \sqrt{\lambda_p}\mathbf{u}_m) \in \mathbb{R}^{n \times m}$$
  3. The rows of $\mathbf{Y}$ are the sought-after embedding, i.e. for $\mathbf{y}_l = \mathbf{Y}_{l\cdot}$ it holds that
  $$\|\mathbf{x}_i - \mathbf{x}_l\|_2 = \|\mathbf{y}_i - \mathbf{y}_l\|_2$$
- **Note:** This is not guaranteed to lead to dimension reduction, i.e. $m = p$ possible. However, usually the internal structure of the data is lower-dimensional and $m < p$.

Multi-dimensional scaling

- Keeping only the first $q < m$ components of $\mathbf{y}_l$ is known as **classical scaling** or **multi-dimensional scaling (MDS)** and minimizes the so-called **stress** or **strain**

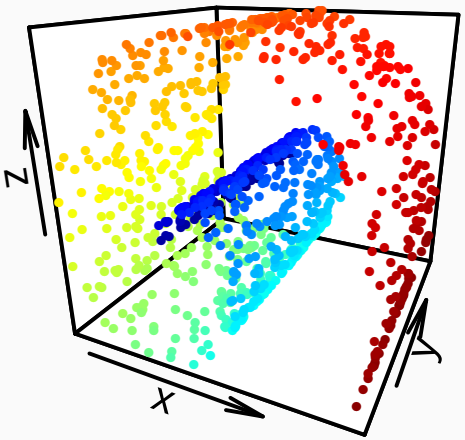
$$d(\mathbf{D}, \mathbf{Y}) = \left(\sum_{i \neq j} (D_{ij} - \|\mathbf{y}_i - \mathbf{y}_j\|_2)^2 \right)^{1/2}$$

- Results also hold for general distance matrices $\mathbf{D}$ as long as $\lambda_1, \ldots, \lambda_m > 0$ for $m = \mathrm{rank}(\mathbf{K})$. This is called **metric MDS**.

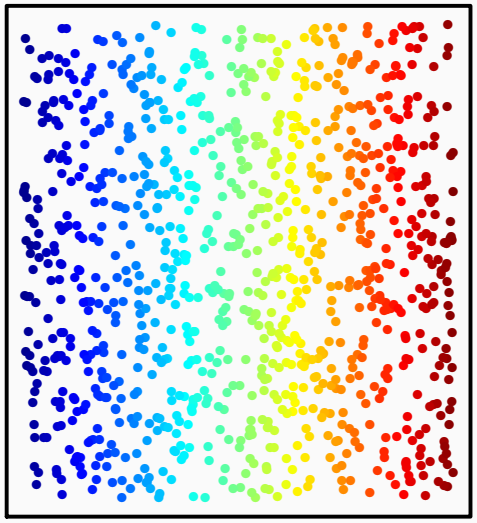
Lower-dimensional data in a high-dimensional space

A problematic geometry

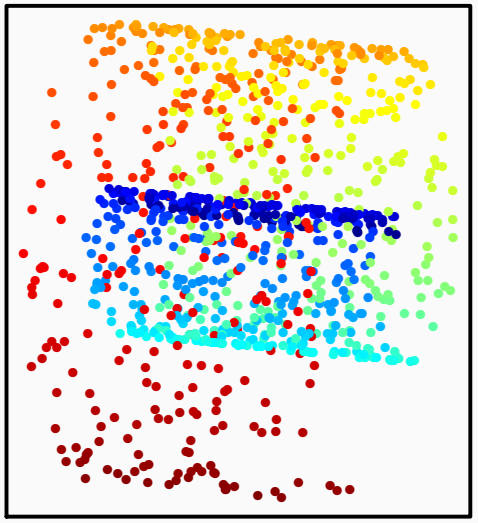
Swiss roll (n = 1000)



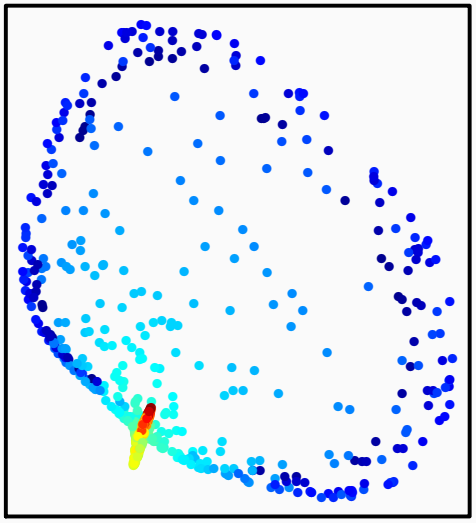
Ideal unrolled graph



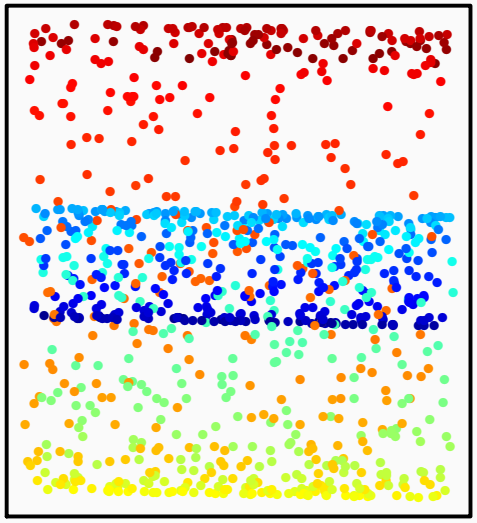
PCA



kPCA (RBF kernel, sigma = 0.13)



Classical scaling

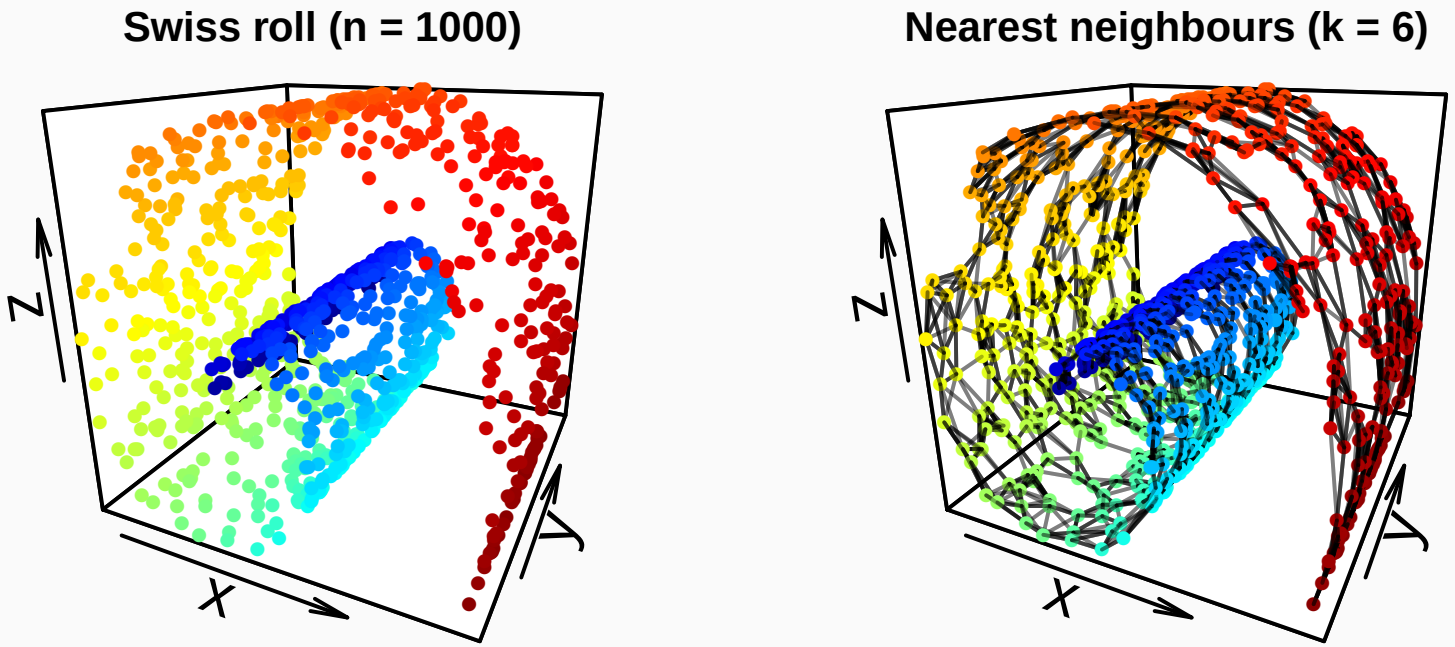


What is the problem here?

- The data has an intrinsic structure that is quite simple (2D) in itself, but much more complex in the three-dimensional space
- To understand this data set properly we need to learn about the local structure of the data
- PCA is a **global method** and will always look at all data
- kernel PCA is a **local method** but the chosen Gaussian kernel does not represent the structure of the data well
- Classical scaling performs roughly like PCA
- **What is the issue?** All approaches measure distances in the Euclidean norm in three dimensions.

Data-driven distance measure (I)

We can create a local, data-driven distance measure by looking at the k nearest neighbours of a data point.



Data-driven distance measure (II)

Computation

1. For a data point $\mathbf{x}_l$ find the k nearest neighbours
2. Construct a graph between data points and their k nearest neighbours, weighting each edge by the Euclidean distance
3. To measure distance between data points measure their **geodesic distance**, i.e. find the shortest path in the weighted graph and sum up the weights

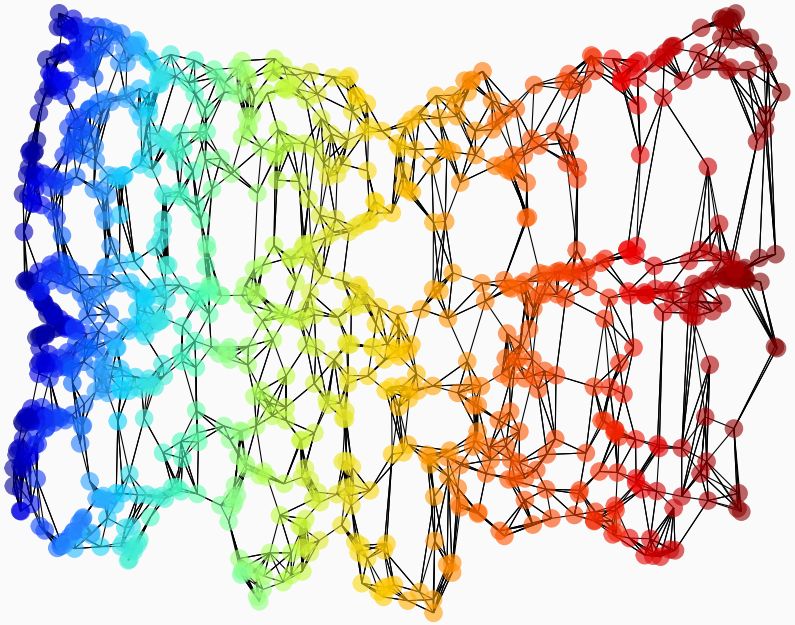
This creates a distance matrix $\mathbf{D}_G$ between data points that is more adapted to the actual geometry.

To **embed the geometry** in a lower-dimensional space, MDS can be applied to $\mathbf{D}_G$.

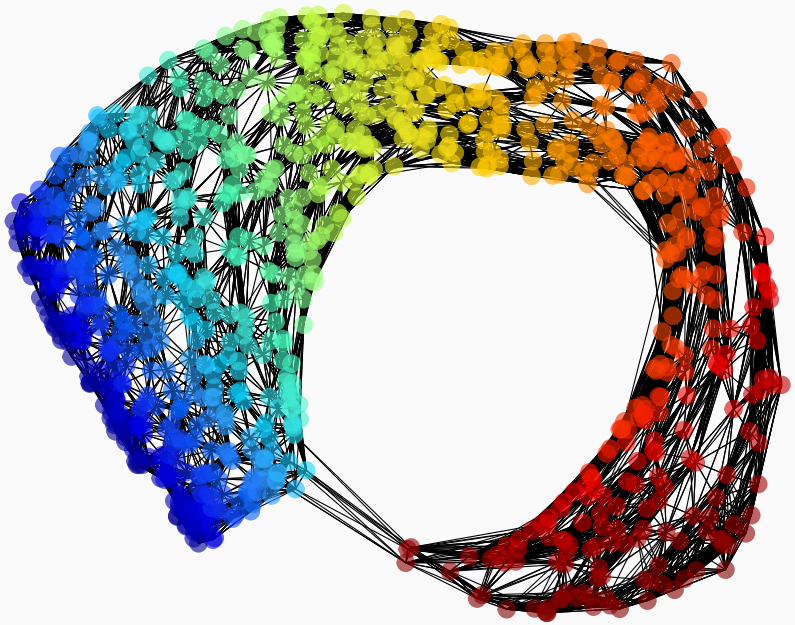
Isomap

The combination of geodesic, local distance measure and classical scaling is called **Isomap**.

Isomap (knn = 6)



Isomap (knn = 20)



Caveats of Isomap

- The distance between two vectors in Euclidean space can always be measure, but it can happen that there is no connection in the graph between two data points.
- **When?** If the graph of the data falls into two (or more) components. Distance is considered infinite in these cases.
- Implementations typically return a different embedding for each component
- Isomap has problems with datasets that have **varying density**
- Number of nearest neighbours has to be carefully tuned

t-distributed Stochastic Neighbour Embedding (tSNE)

t-distributed stochastic neighbour embedding (tSNE) follows a similar strategy as Isomap, in the sense that it **measures distances locally**.

Idea: Measure distance of feature $\mathbf{x}_l$ to another feature $\mathbf{x}_i$ proportional to the likelihood of $\mathbf{x}_i$ under a Gaussian distribution centred at $\mathbf{x}_l$ with an isotropic covariance matrix.

Computation of tSNE

For feature vectors $\mathbf{x}_1, \dots, \mathbf{x}_n$, set

$$p_{i|l} = \frac{\exp(-\|\mathbf{x}_l - \mathbf{x}_i\|_2^2/(2\sigma_l^2))}{\sum_{k \neq l} \exp(-\|\mathbf{x}_l - \mathbf{x}_k\|_2^2/(2\sigma_l^2))} \quad \text{and} \quad p_{il} = \frac{p_{i|l} + p_{l|i}}{2n},\ p_{ll} = 0$$

The variances σ_i^2 are chosen such that the **perplexity** (here: **approximate number of close neighbours**) of each marginal distribution (the $p_{i|l}$ for fixed l) is constant.

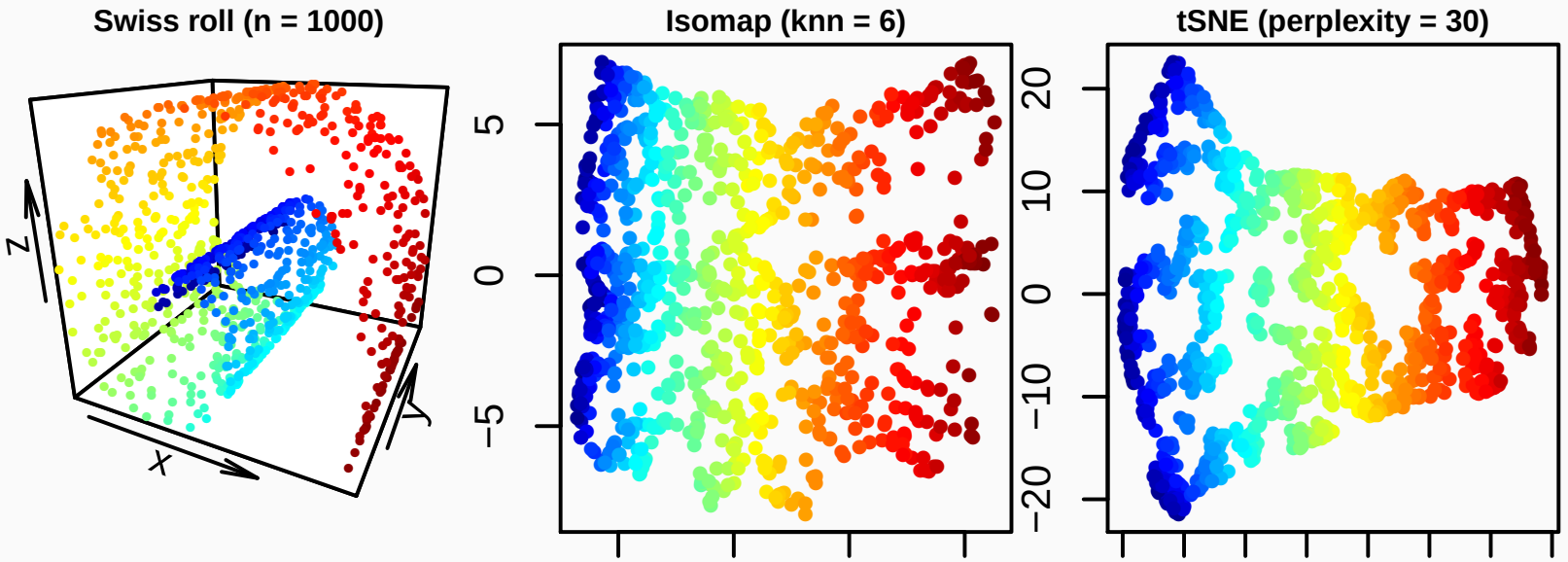
In the lower-dimensional embedding distance between $\mathbf{y}_1, \dots, \mathbf{y}_n$ is measured with a **t-distribution with one degree of freedom** or **Cauchy distribution**

$$q_{il} = \frac{\left(1 + \|\mathbf{y}_i - \mathbf{y}_l\|_2^2\right)^{-1}}{\sum_{k \neq j} \left(1 + \|\mathbf{y}_k - \mathbf{y}_j\|_2^2\right)^{-1}} \quad \text{and} \quad q_{ll} = 0$$

To determine the $\mathbf{y}_l$ the KL divergence between the distributions $P = (p_{il})_{il}$ and $Q = (q_{il})_{il}$ is minimized with gradient descent

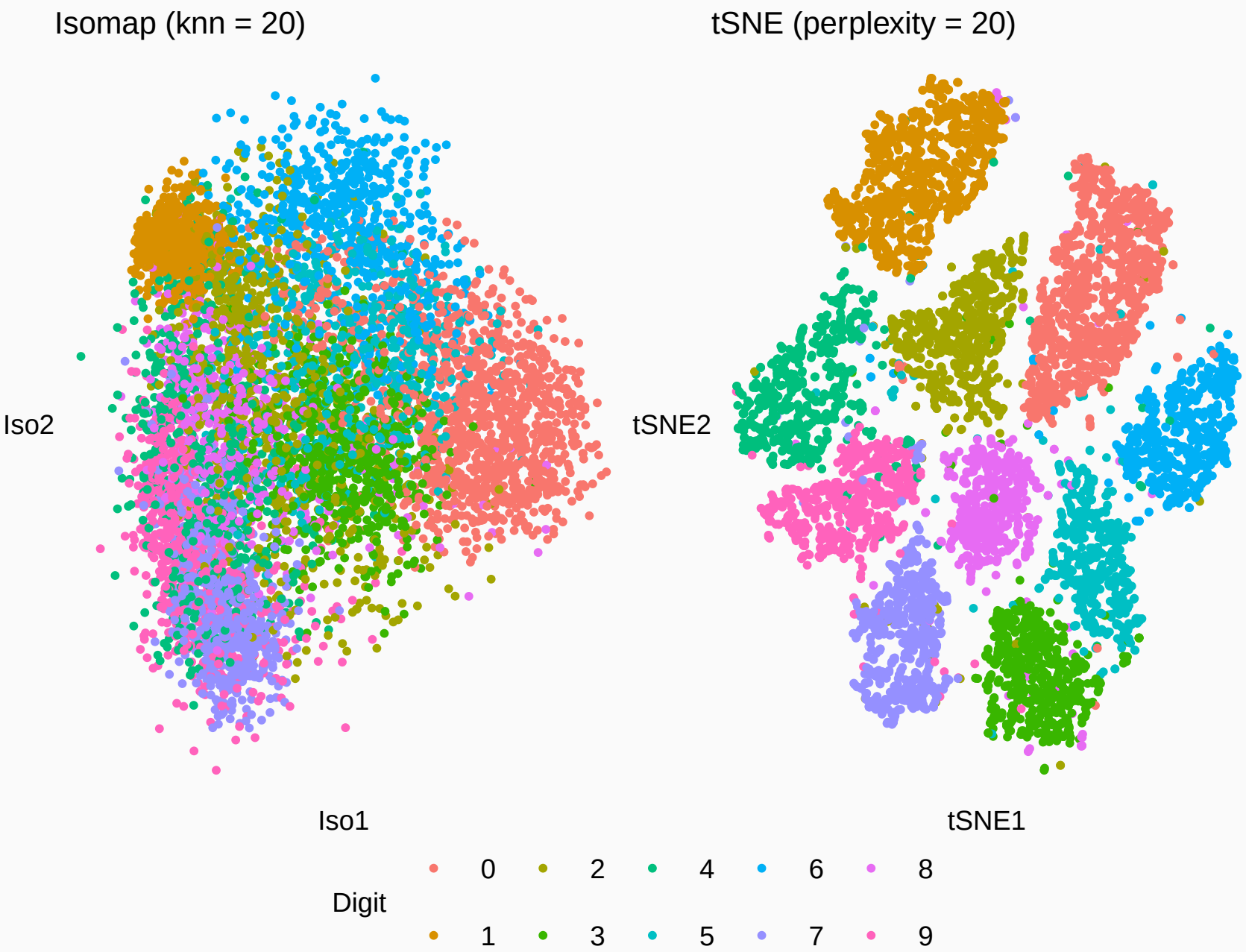
$$\mathrm{KL}(P||Q) = \sum_{i \neq l} p_{il} \log \frac{p_{il}}{q_{il}}$$

Revisiting the Swiss roll with tSNE



- Results are similar to Isomap
- Slightly more condensed, but manages the main goal to unroll data

A more impressive example of tSNE

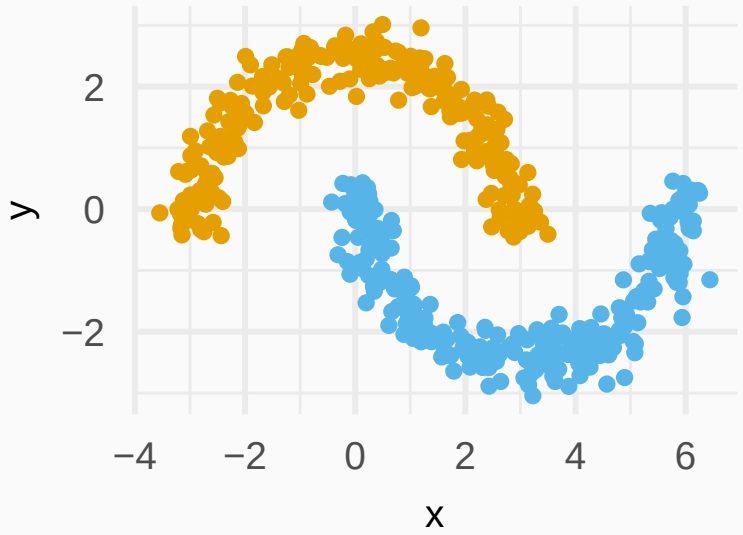


Caveats of tSNE

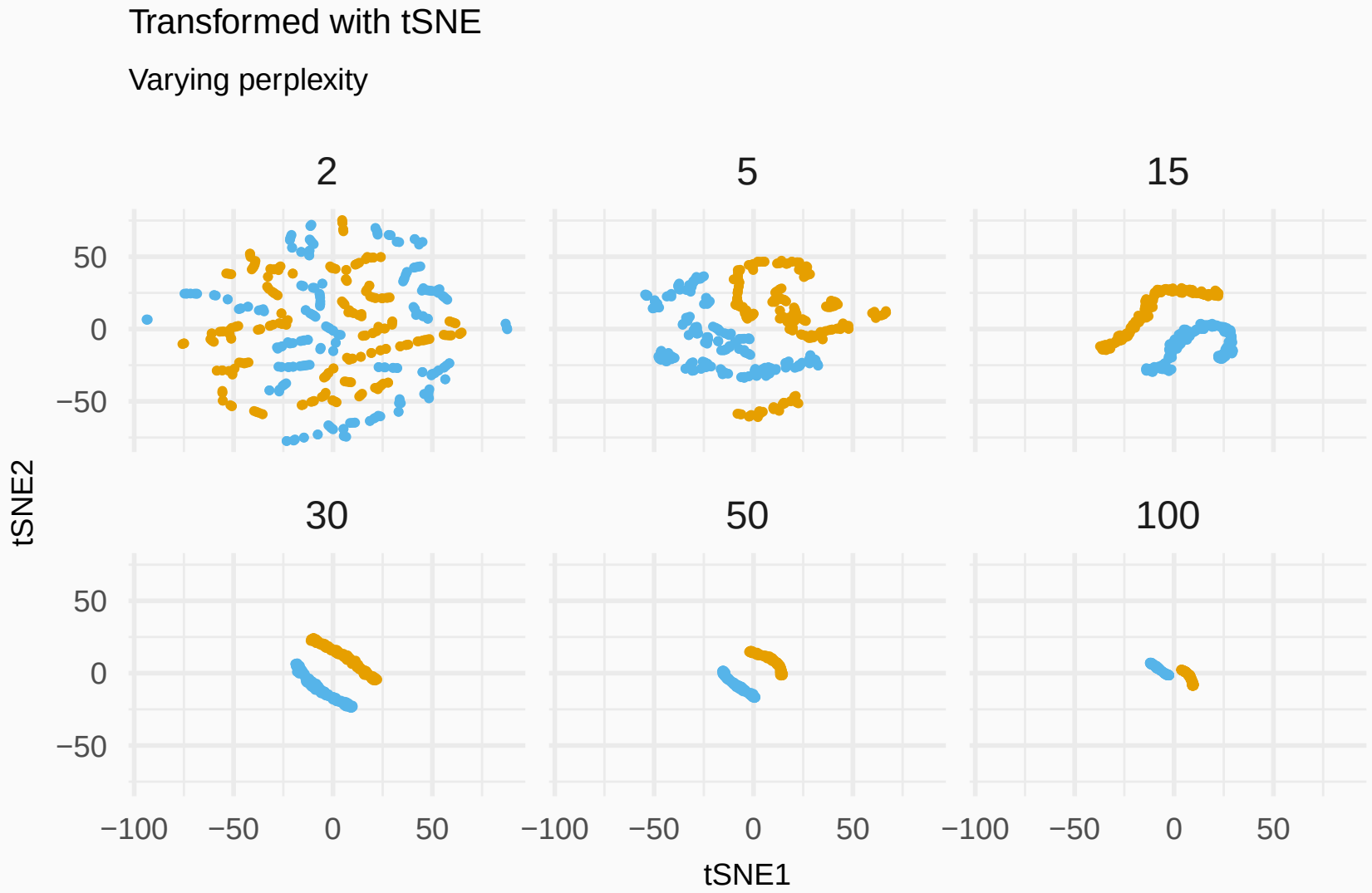
tSNE is a powerful method but comes with some difficulties as well

- Convergence to local minimum (i.e. repeated runs can give different results)
- Perplexity is hard to tune (as with any tuning parameter)

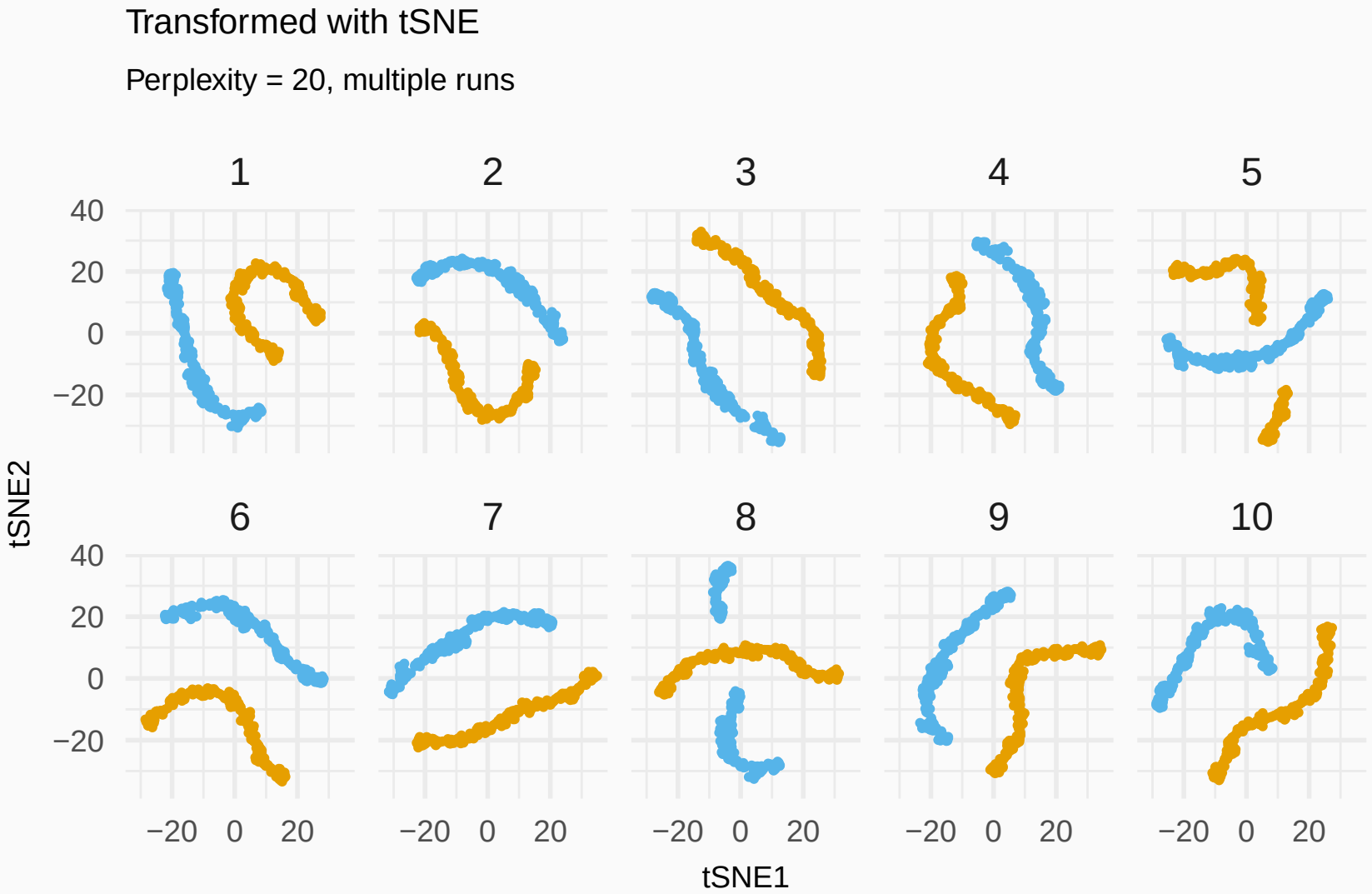
Let's see what tSNE does to our old friend, the moons dataset.



Influence of perplexity on tSNE



tSNE multiple runs



Take-home message

- Dimension reduction has multiple sub-goals, like preserving structure
- Data that has a lower-dimensional structure in a high-dimensional room can be tricky to uncover
- Isomap and tSNE are powerful dimension reduction techniques that also uncover structure, but be careful about applying them blindly