MACHINE LEARNING

Spectral Clustering & Nonlinear Embeddings

Finding Non-Linear Manifolds Through Spectral Decomposition

Methods that depend on spectral decomposition of matrices are often referred to as *spectral methods.*

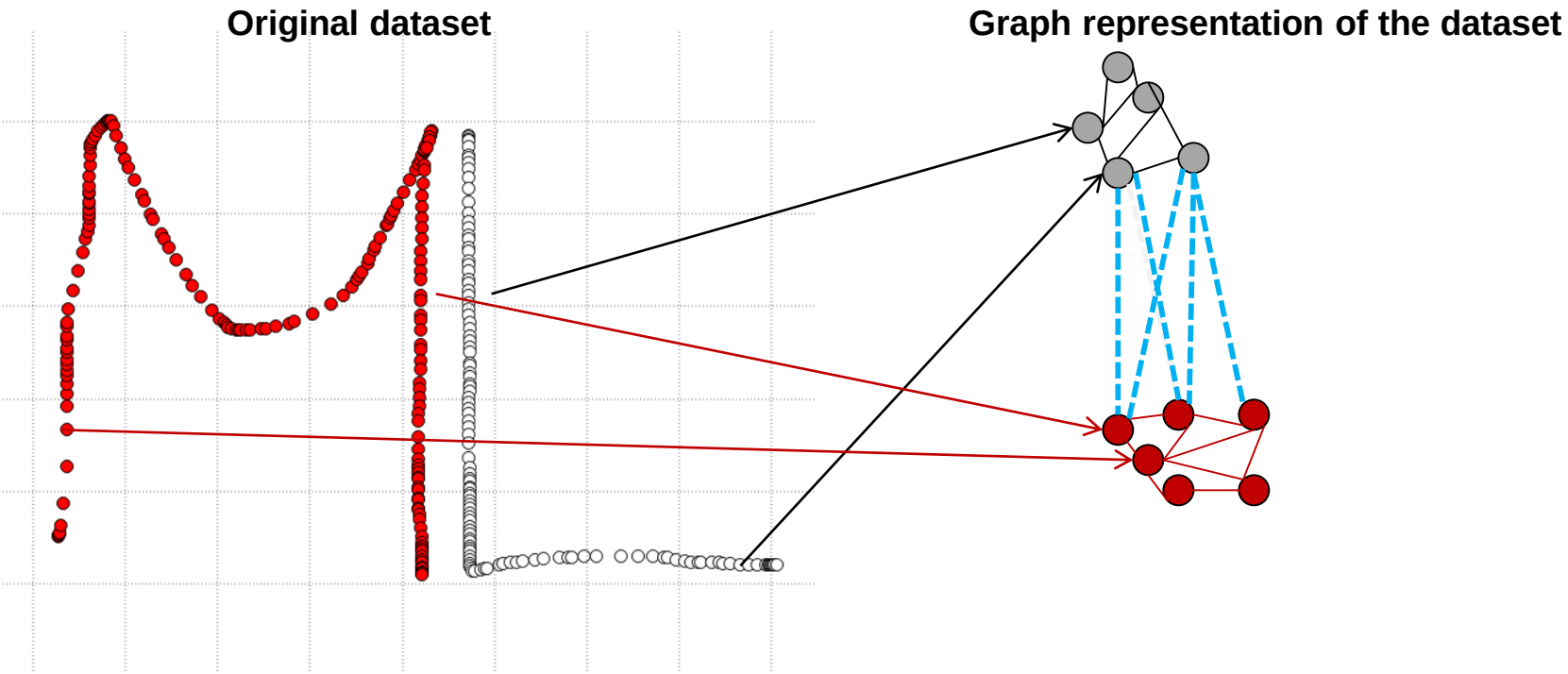
PCA/kPCA and CCA/kCCA are spectral methods as they depend on performing an eigenvalue and eigenvector decomposition of a matrix.

Depending on which matrix we decompose, we get a different set of projections.

Non-Linear Manifolds from Spectral Clustering

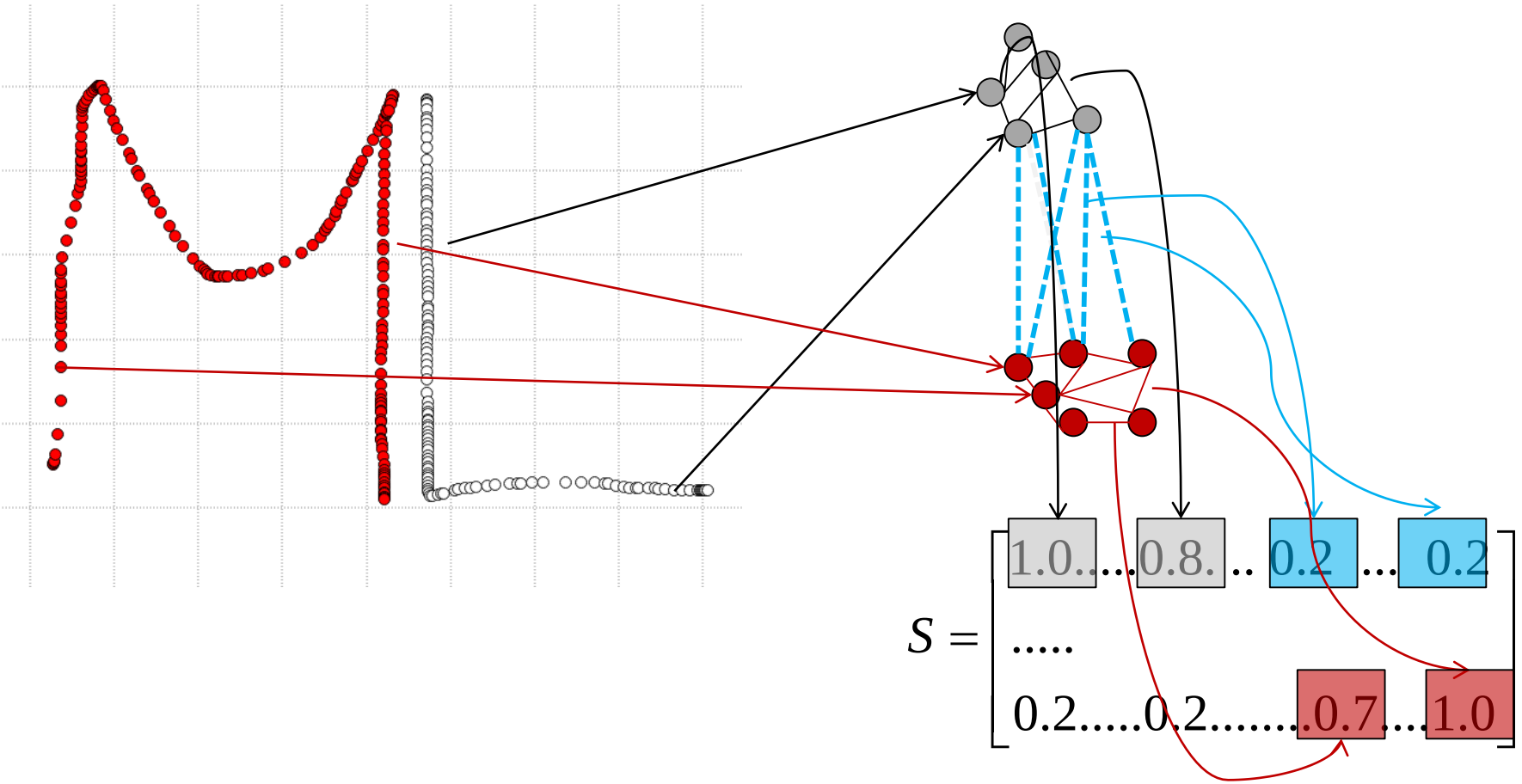
- Spectral clustering proceeds also through spectral decomposition.
- It decomposes the *Graph Laplacian matrix*.
  - The Graph Laplacian is a matrix representation of a graph.
  - Its decomposition entails a notion of connectivity across data-points and, hence, leads naturally to a decomposition by group (clustering).

Embed Data in a Graph



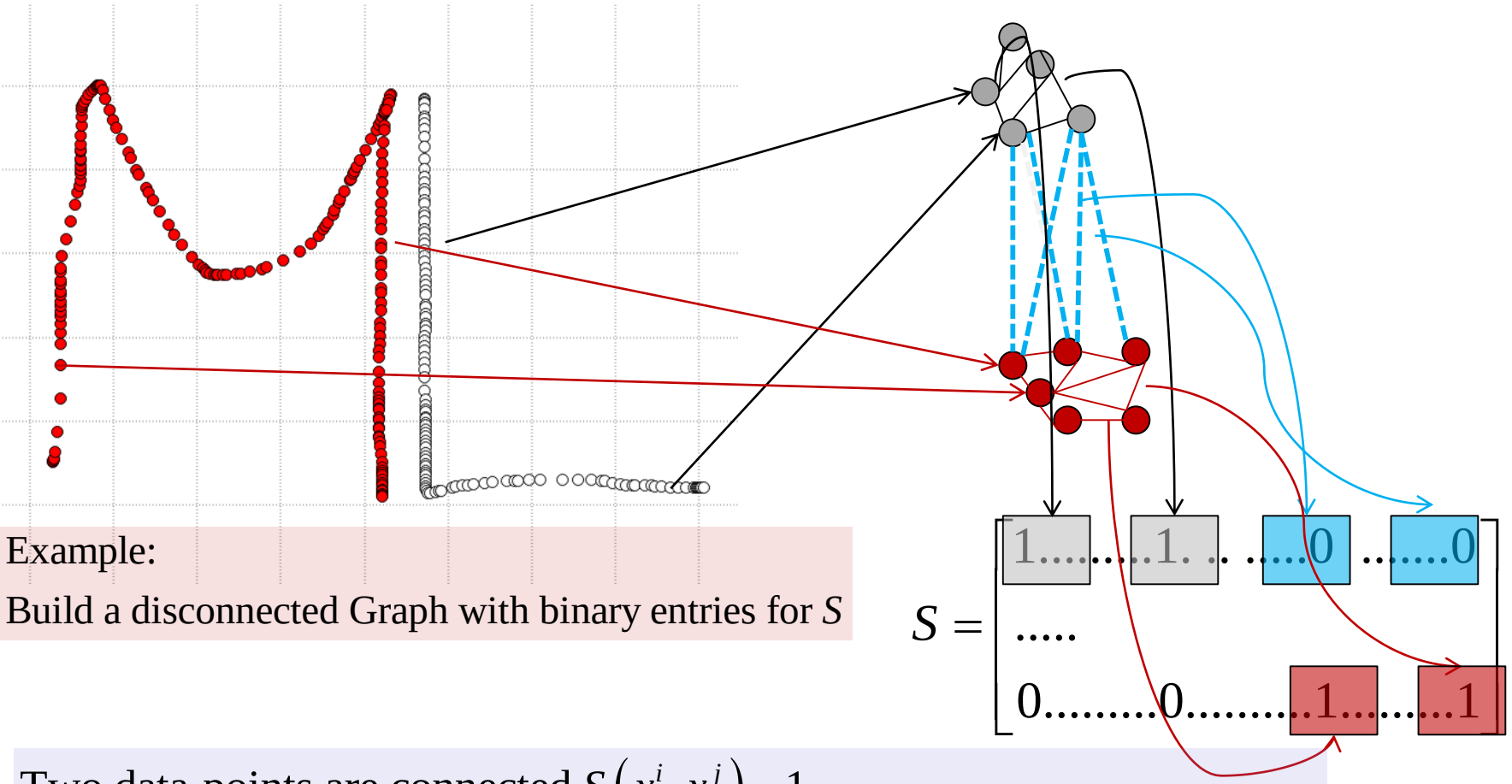
- Build a *similarity graph*
- Each vertex on the graph is a datapoint

Measure Distances in Graph



Construct the similarity matrix (adjacency matrix) *S* to denote whether points are close or far away to weight the edges of the graph.

Disconnected Graphs



Example:

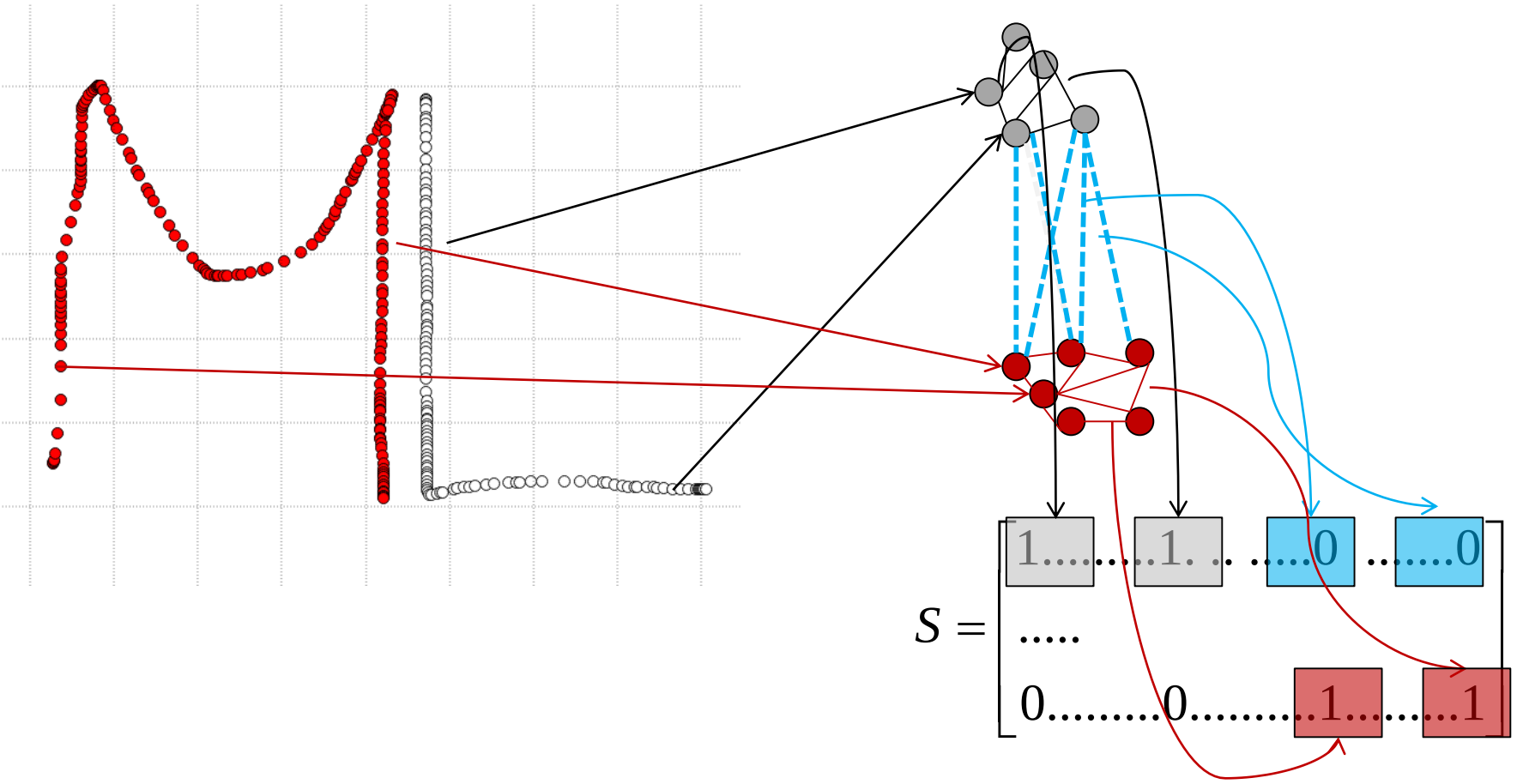
Build a disconnected Graph with binary entries for S

Two data-points are connected $S\left(x^i, x^j\right)=1$

if a) the similarity between them is higher than a threshold;

or b) if they are k-nearest neighbors $\left(\text{according to the similarity metric}\right)$.

Connected Components in a Graph



If all blue connections have value zero in the similarity matrix, then the matrix is composed of blocks with each block corresponding to one **connected component** (all datapoints within a block are connected).

Spectral Clustering Properties

- Spectral clustering can discover the number of connected components, i.e. *number of clusters in the dataset*.
- Knowing this number, it can then identify to which cluster each points belong.
- Proper clustering would however depend on choosing well the similarity matrix.

Constructing the Graph Laplacian Matrix

Given a similarity matrix S.

Construct the diagonal matrix D composed of the sum on each line of S:

$$D = \begin{bmatrix} \sum_i S_{1i} & \dots & \dots & 0 \\ 0 & \sum_i S_{2i} & \dots & 0 \\ \dots & \dots & \sum_i S_{3i} & \\ 0 & \dots & \dots & \sum_i S_{4i} \end{bmatrix}$$

Build the Laplacian matrix: $L = D - S$

Constructing the Graph Laplacian Matrix

$$S = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$ (4x4 example)

Construct the diagonal matrix D composed of the sum on each line of S:

$$D = \begin{bmatrix} 2 & \cdots & 0 \\ 0 & 2 & \cdots 0 \\ \cdots & & 2 \\ 0 & \cdots & 2 \end{bmatrix}$$

$$L = D - S = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ 0 & 2 & -1 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ -1 & 0 & 0 & 2 & -1 \end{bmatrix}$$

and then, build the Graph Laplacian matrix : $L = D - S$

L is positive semi-definite $\rightarrow$ spectral decomposition possible

Graph Laplacian

Eigenvalue decomposition of the Graph Laplacian matrix:

$L = U \Lambda U^T$

All eigenvalues of L are positive and the smallest eigenvalue of L is zero:

$\Rightarrow$ If we order the eigenvalues by increasing order:

$\lambda_1 = 0 \leq \lambda_2 \leq .... \leq \lambda_M$.

Theorem (see annexes):
If the graph has k connected components, then the eigenvalue λ=0 has multiplicity k.

Role of the eigenvalues in spectral clustering

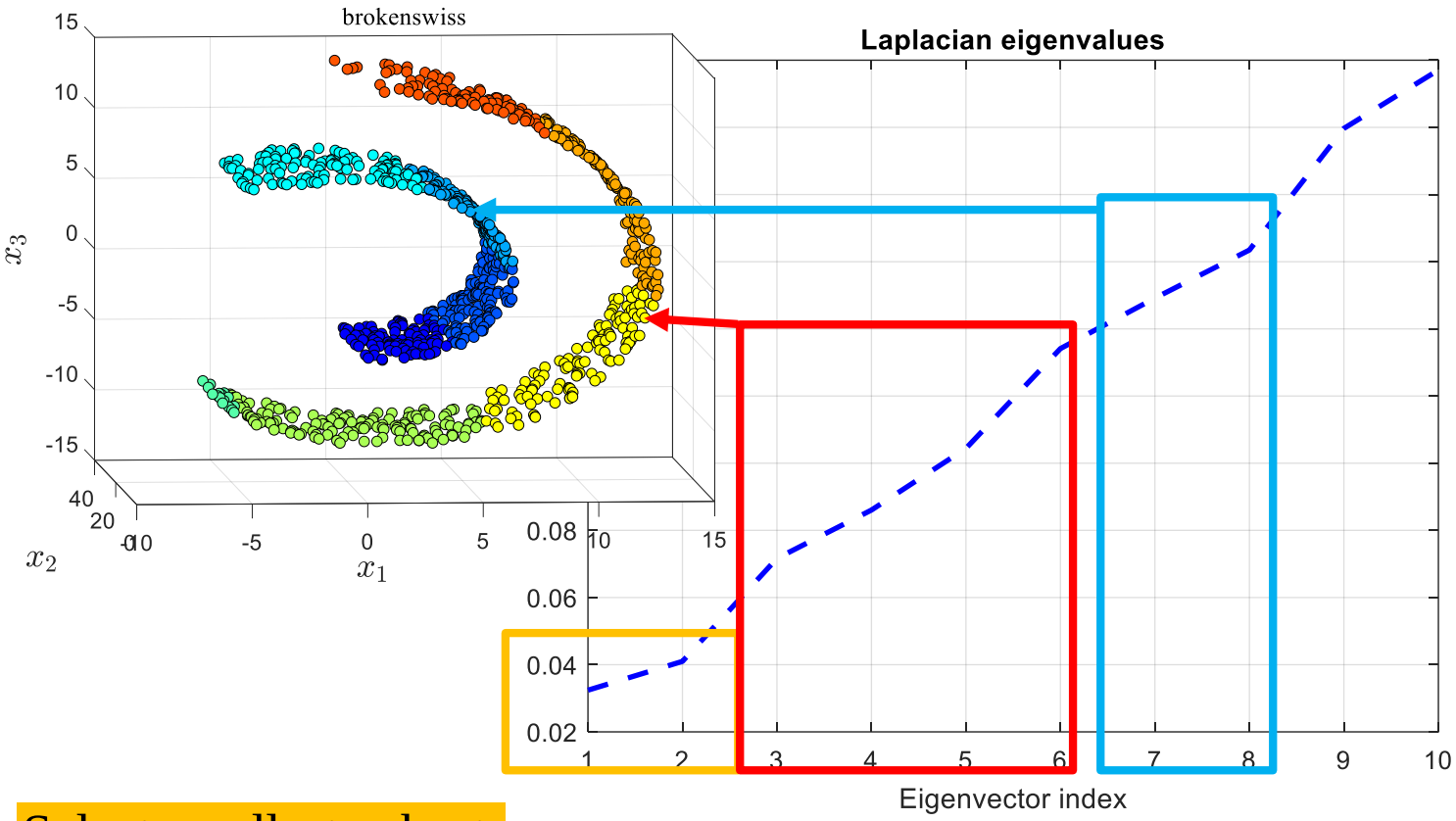
$\Rightarrow$ The multiplicity of the eigenvalue 0 determines the number of connected components in a graph.

→ Identifying the number of clusters using the eigenvalue decomposition of the Laplacian matrix is then immediate (using above) *when* the similarity matrix is sparse.

What happens when the similarity matrix is full?

Idea: the smallest eigenvalues (close to zero) provide also information on the partitioning of the graph.

Role of the eigenvalues in spectral clustering



Select smallest subset.

We order the eigenvalues by increasing order, removing the first one (zero):

$$\lambda_1 \neq 0 \leq \lambda_2 \leq \lambda_2 \leq .... \leq \lambda_M .$$

Algorithm to determine the number of clusters

1) Create a similarity matrix S.

2) Build the Laplacian matrix L.

3) Decompose the Laplacian matrix: $L = U \Lambda U^T$

4) Order the eigenvalues by increasing order:

$\lambda_1 = 0 \leq \lambda_2 \leq .... \leq \lambda_M$

5) Determine the number of clusters by looking at the multiplicity of $\lambda = 0$, or apply a threshold on the eigenvalues, such that small $\lambda \rightarrow 0$.

This provides an indication of the number of clusters K.

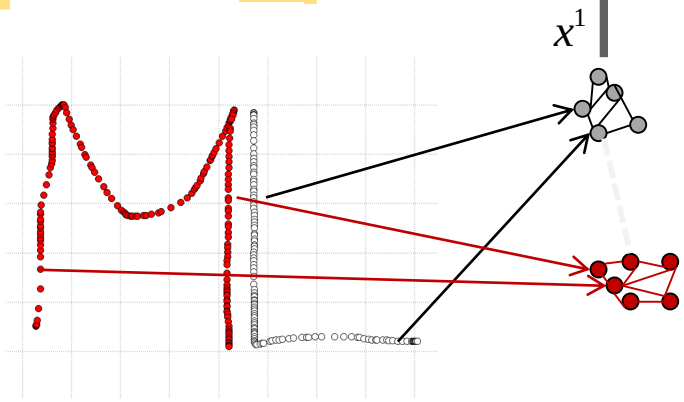
We do not yet know how the points are partitioned in the clusters!
Let us see now how we can infer the clusters from the eigenvalue decomposition.

Finding the clusters

Eigenvectors of the Laplacian matrix in U:

$$U = \begin{bmatrix} e_1^1 & e_1^2 & \ldots\ldots & e_1^M \\ e_2^1 & & & \\ \cdot & & & \\ \cdot & & & \\ \cdot & & & \\ e_M^1 & e_M^2 & \ldots\ldots & e_M^M \end{bmatrix} \qquad y^1 = \begin{bmatrix} e_1^1 \\ \cdot \\ \cdot \\ \cdot \\ e_1^M \end{bmatrix}$$

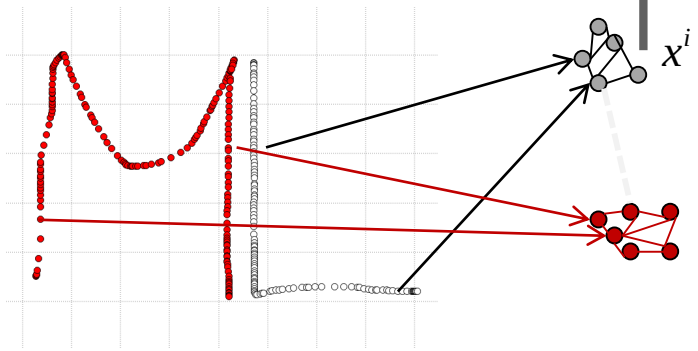
Construct an embedding for each of the M datapoints x^i through y^i.



Finding the clusters

Eigenvectors of the Laplacian matrix in U:

$$U = \begin{bmatrix} e_1^1 & e_1^2 & \ldots\ldots & e_1^M \\ e_2^1 & & & \\ \cdot & & & \\ \cdot & & & \\ e_M^1 & e_M^2 & \ldots\ldots & e_M^M \end{bmatrix} \longleftarrow y^i = \begin{bmatrix} e_i^1 \\ \cdot \\ \cdot \\ \cdot \\ e_i^M \end{bmatrix}$$



Construct an embedding for each of the M datapoints x^i through y^i.

This amounts to a non-linear mapping

$$X = \{x^i\}_{i=1}^M \xrightarrow{\phi} Y = \{y^i\}_{i=1}^M$$

Finding the clusters

Eigenvectors of the Laplacian matrix in U:

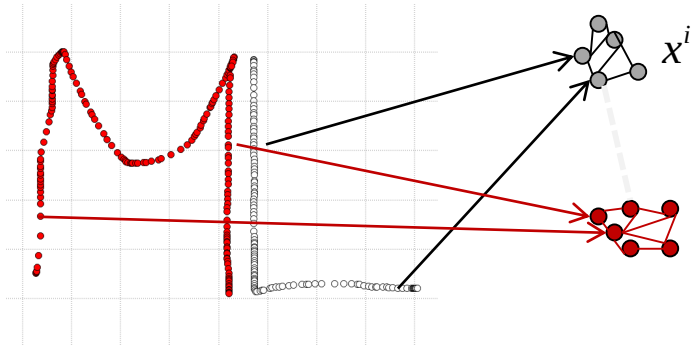
$$U = \begin{bmatrix} e_1^1 & e_1^2 & \ldots\ldots e_1^M \\ e_2^1 & & \\ \cdot & & \\ \cdot & & \\ \cdot & & \\ e_M^1 & e_M^2 & \ldots\ldots e_M^M \end{bmatrix} \quad K = 2 \qquad \longleftarrow \quad y^i = \begin{bmatrix} e_i^1 \\ \cdot \\ \cdot \\ \cdot \\ e_i^M \end{bmatrix}$$

Construct an embedding for each of the M datapoints x^i through y^i.

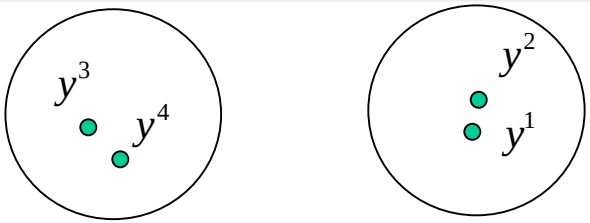
This amounts to a non-linear mapping

$$X = \{x^i\}_{i=1}^M \xrightarrow{\phi} Y = \{y^i\}_{i=1}^M$$

$\rightarrow$ Reduce dimensionality by picking $K < M$ eigenvectors e^i, $i = 1 \ldots K$, with smallest eigenvalues.



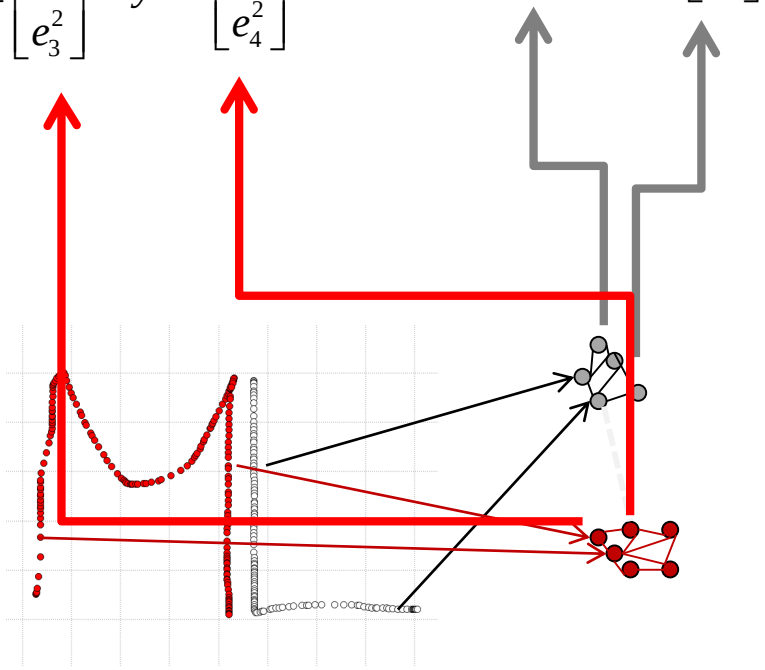
Finding the clusters



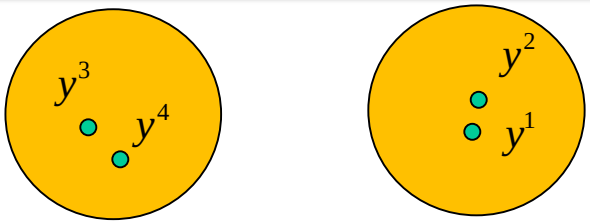
Hypothesis:
Points well grouped in original space generate grouped images y^i.

Apply any standard clustering technique on y.

$$y^3 = \begin{bmatrix} e_3^1 \\ e_3^2 \end{bmatrix} \quad y^4 = \begin{bmatrix} e_4^1 \\ e_4^2 \end{bmatrix} \quad y^1 = \begin{bmatrix} e_1^1 \\ e_1^2 \end{bmatrix} \quad y^2 = \begin{bmatrix} e_2^1 \\ e_2^2 \end{bmatrix}$$



Complete Spectral Clustering Algorithm



5: Apply K-Means on the set of $y^1,...y^M \in \mathbb{R}^K$.

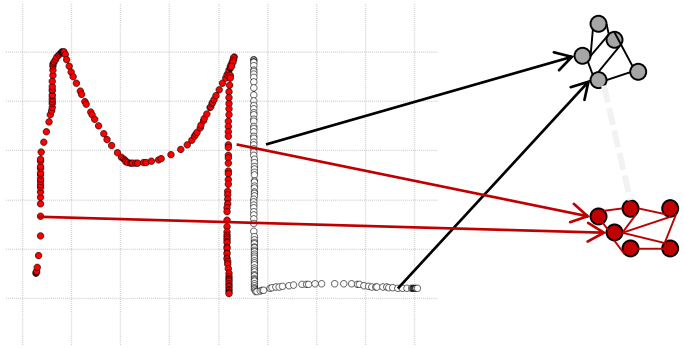
4: Construct an embedding of each of the M datapoints x^i through y^i, i-th line of the eigenvectors.

3: Pick K smallest eigenvalues and associated eigenvectors

2: Decompose the Laplacian matrix

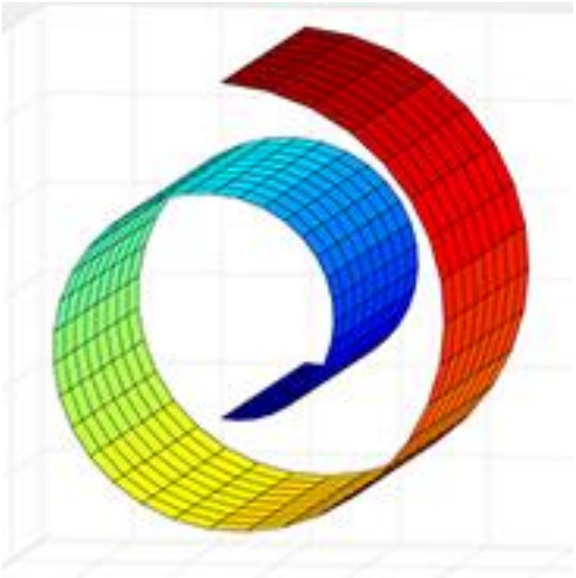
1: Build a graph representation of the data

6: Cluster datapoints in x according to their clustering in y.



Non-linear embeddings
Laplacian Eigenmaps
Isomaps

Non-linear embeddings

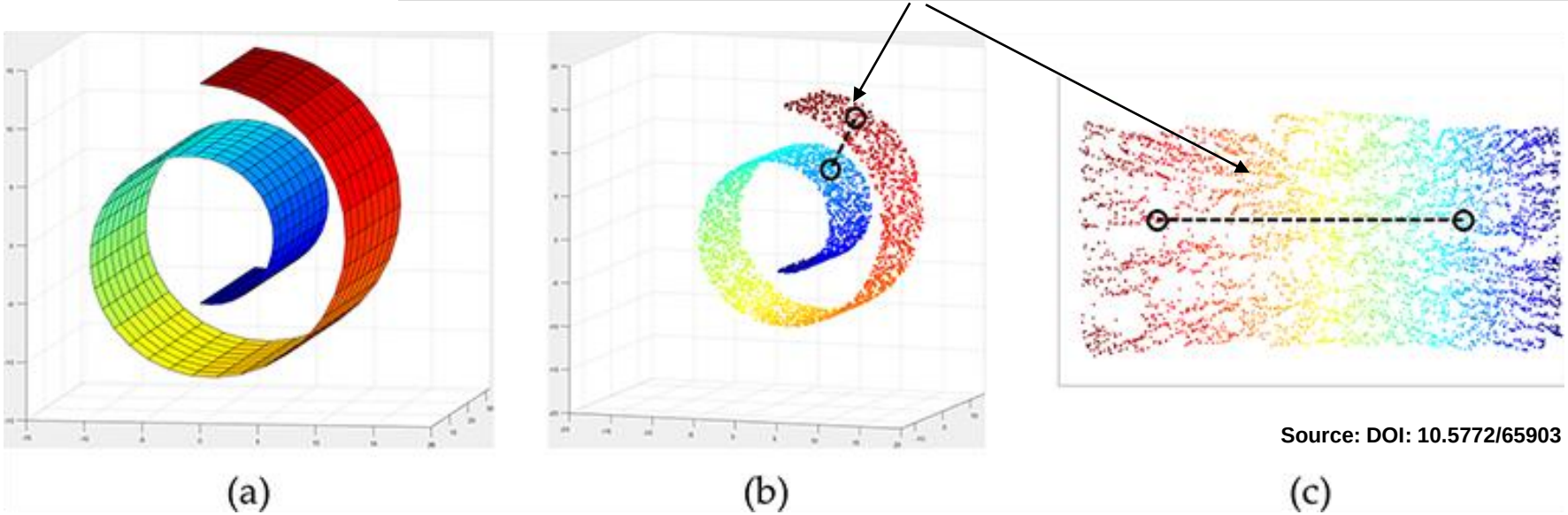


Data lives on a 2D manifold folded onto itself in 3D.

→ The data lives in a latent space which is lower-dimensional.

Non-linear embeddings

Points close to one another are actually far apart on the manifold

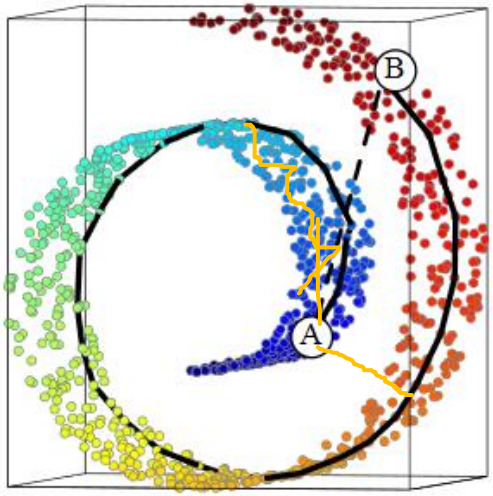


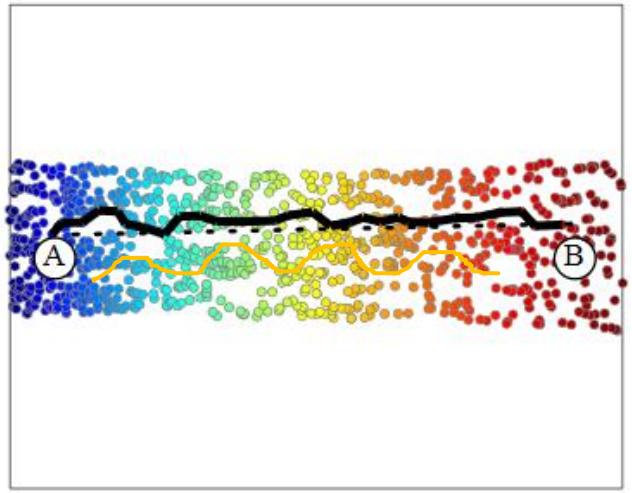
Source: DOI: 10.5772/65903

If one could flatten the manifold, this would be advantageous for:

- Visualizing the data
- Reducing the dimensionality
- Linearly separating the group of data

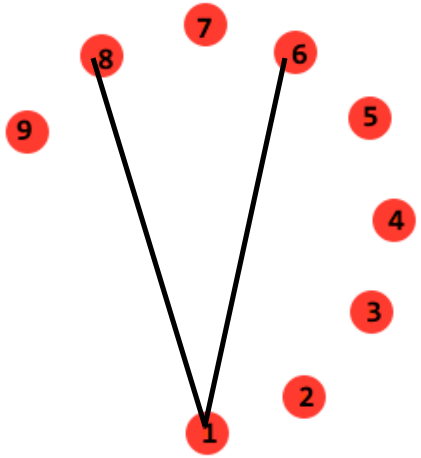
Geodesic distance





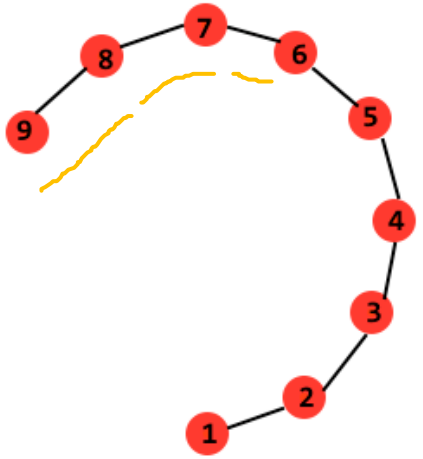
Source: DOI: 10.5772/65903

Euclidean distance



d(1,6) = d(1,8)

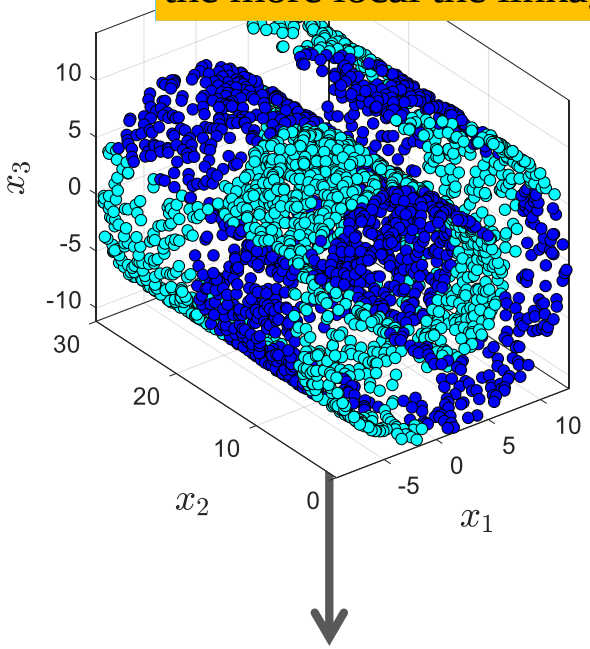
Geodesic Distance (1-neighbors)



d(1,6) < d(1,8)

Isomap embedding

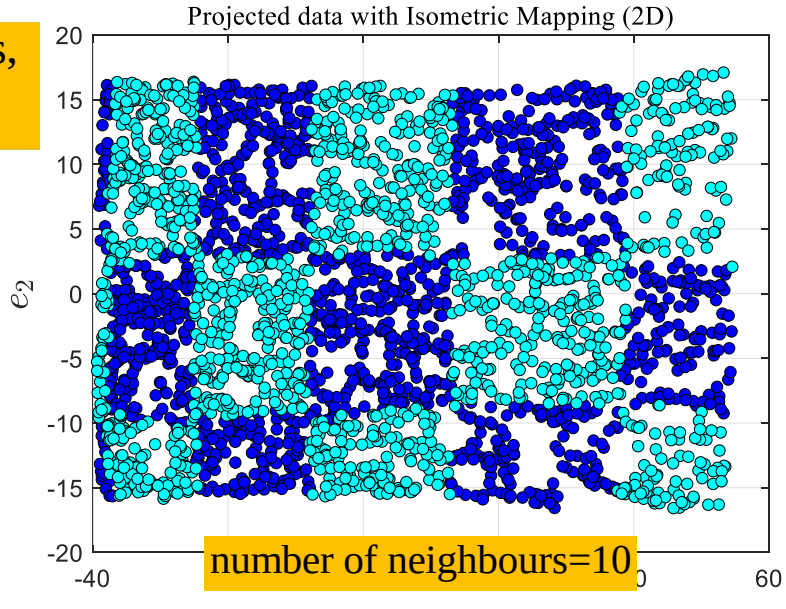
The smaller the number of neighbours, the more local the linkage.



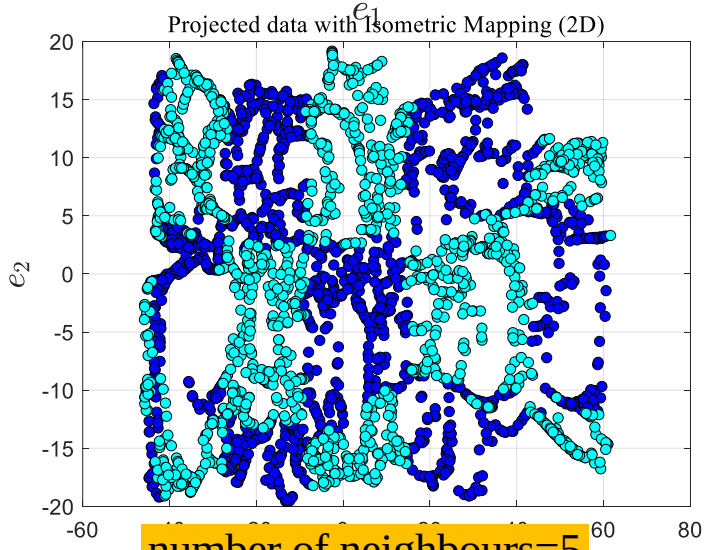
Compute pairwise distances:

$$S_{ij} = \left(\min_{k-nearest\ \text{neighbours}} d\left(x^i, x^j\right) \right)^2$$

Do an eigendecomposition of the centered similarity matrix.

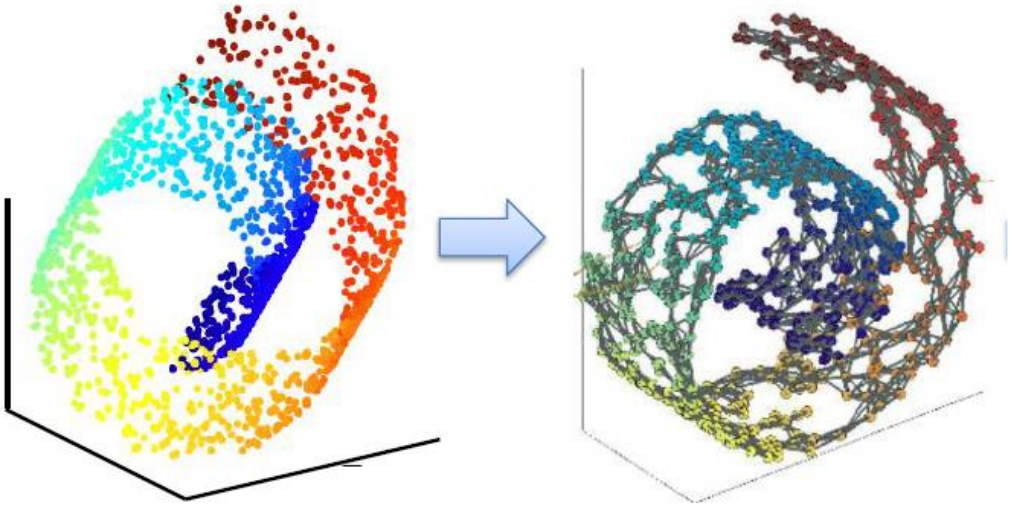


number of neighbours=10



number of neighbours=5

Laplacian Eigenmaps



Sample datapoints

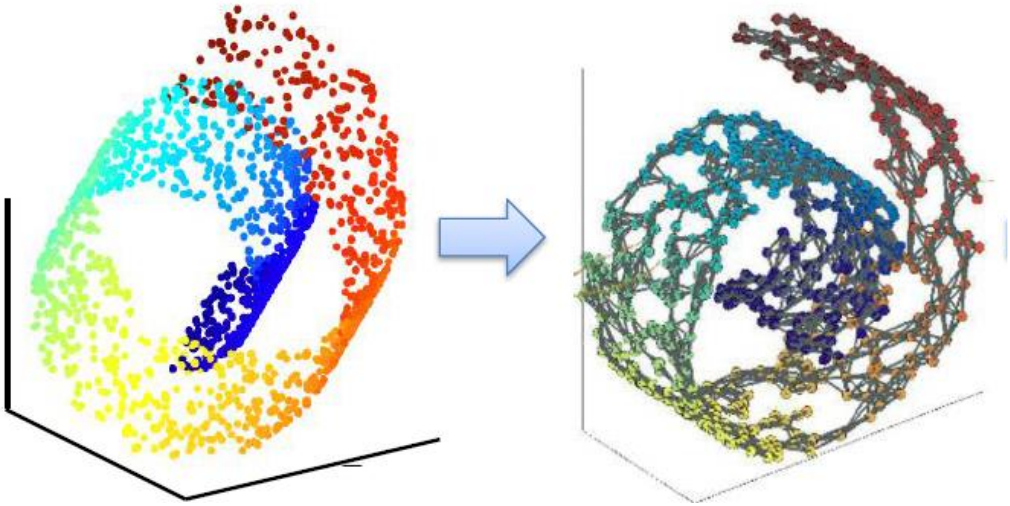
Construct graph from datapoints

Build adjacency or similarity matrix:
We put an edge (rbf kernel) between nodes *i* and *j* if:

- x^i is among k nearest neighbors of x^j or
- if x^j is among k nearest neighbors of x^i

(the relation is symmetric)

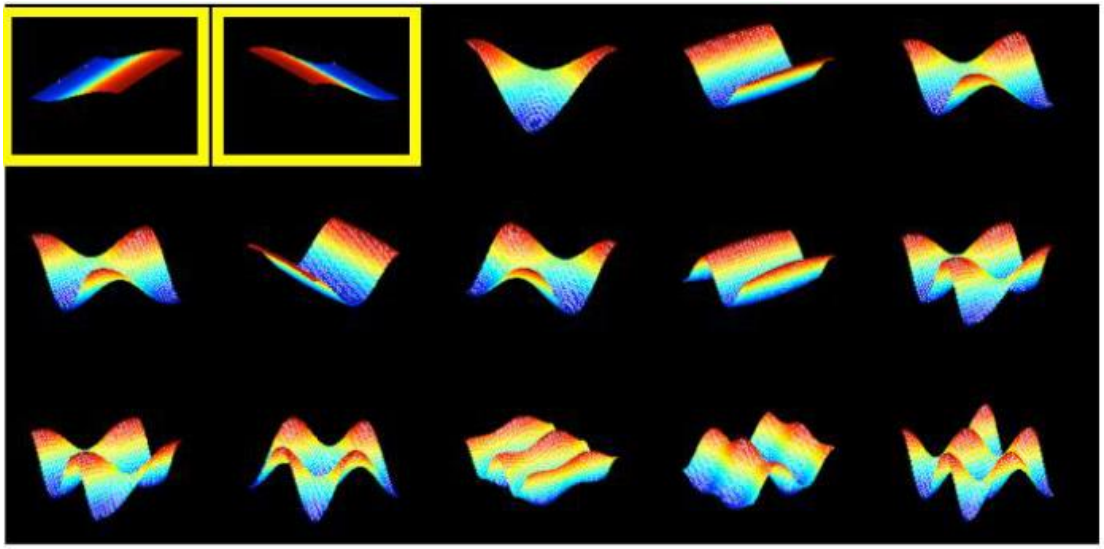
Laplacian Eigenmaps



Solve the generalized eigenvalue problem:

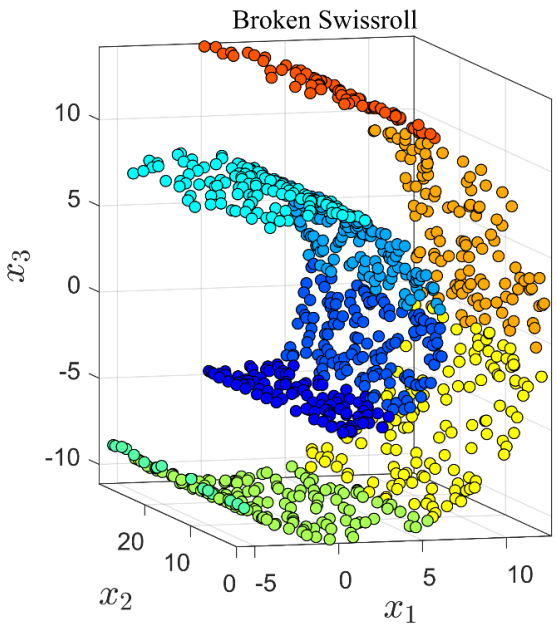
$Le^i = \lambda De^i$

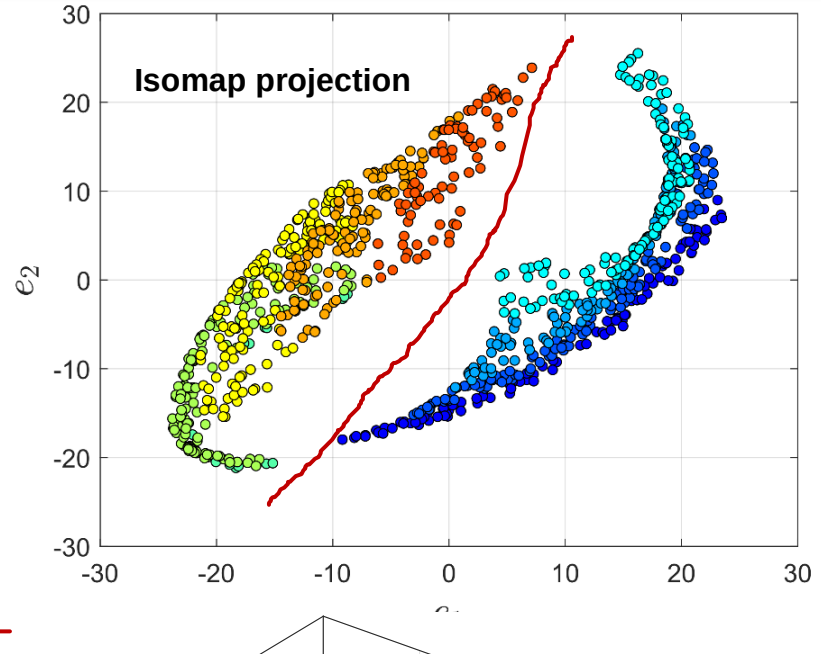
e^i : M eigenvectors

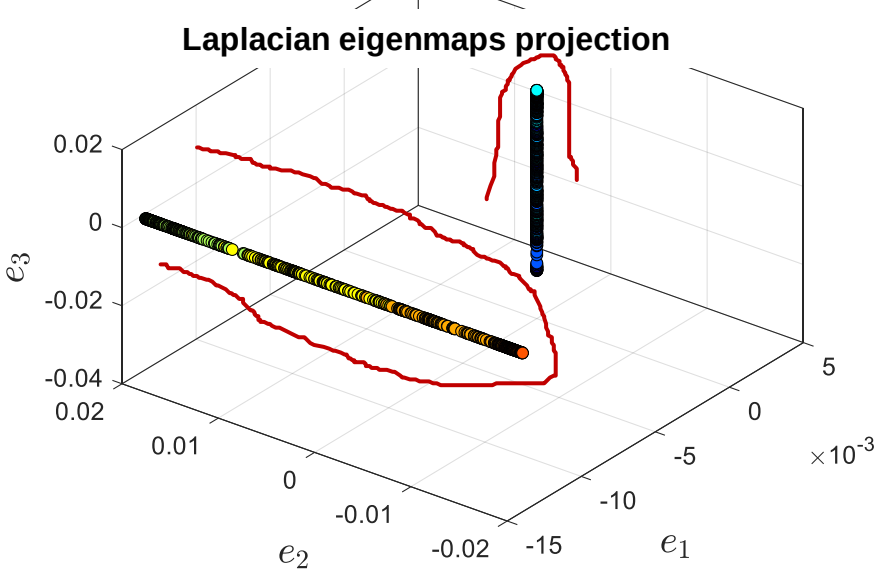


Projections on each vector e^i generate different embeddings.

Isomap versus Laplacian Eigenmaps







The geodesic distances encapsulate the fact that datapoints live on separate manifolds & allow to extract well the 2 main groups.

Summary of this lecture

- Spectral clustering decomposes the *Graph Laplacian matrix:* The Graph Laplacian is a matrix representation of a graph.
- Eigenvalue decomposition of this matrix determines ***relationships*** across datapoints induced by the similarity across datapoints embedded in the graph.
- The spectral decomposition of the *Graph Laplacian matrix* can be used to generate various projections, including scaling of the space, flattening and clustering.

Summary of spectral methods

We have seen four spectral methods to determine nonlinear features in the data:

- **Kernel PCA** - appropriate for data that live in a **single modality**.
- **Kernel CCA** – appropriate to find embeddings **across different modalities**.
- **Kernel K-means**: finds at once clustering & non-linear embedding.
- **Spectral clustering & Laplacian embeddings**: decomposition of Laplacian can be used for either finding embeddings or clustering.

Spectral methods are very powerful, if used well!

Their power relies on you **choosing well the kernel**.

→ Practice session to understand the role of the kernel and its parameters.