Dimensionality Reduction Part 3: The Local Manifold

Ng Yen Kaow

Dimensionality Reduction

- Linear methods
  - PCA (Principal Component Analysis)
  - cMDS (Classical Multidimensional Scaling)
- Non-linear methods
  - KPCA (Kernel PCA)
  - mMDS (Metric MDS)
  - **Isomap**
  - **LLE** (Locally Linear Embedding)
  - **Laplacian Eigenmap**
  - t-SNE (t-distributed Stochastic Neighbor Embedding)
  - UMAP (Uniform Manifold Approximation and Projection)

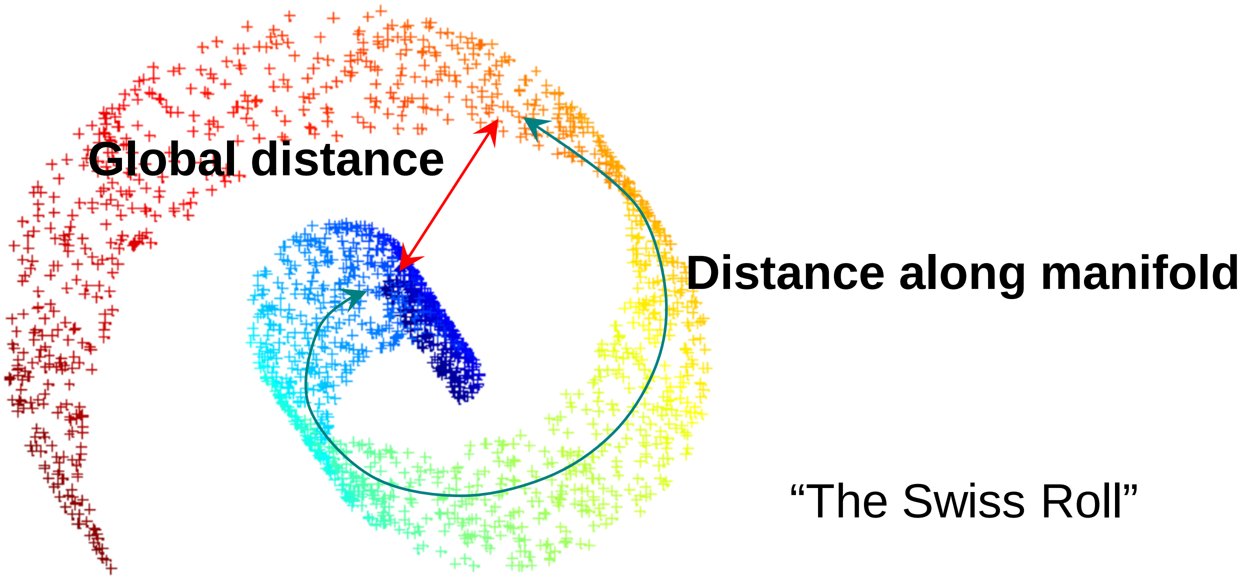
Keys principles for PCA/MDS

	Property in matrix	Linearity of mapped space	Principle	Dimensionality reduction
PCA	Pairwise (**global**) covariance	**Linearly** mapped space (or no mapping)	Maximizes covariance in mapped space	Principal eigenvectors
cMDS	Pairwise (**global**) inner product	**Linearly** mapped space (or no mapping)	Recovers original structure	Principal eigenvectors
mMDS	Pairwise (**global**) metric distance	**Non-linearly** mapped space	Find approximation in low dimension	
Kernel PCA	Pairwise (**global**) covariance	**Non-linearly** mapped space	Maximizes covariance in mapped space	Principal eigenvectors

- PCA readily allows embedding of **out-of-sample examples**

Drawback with global properties

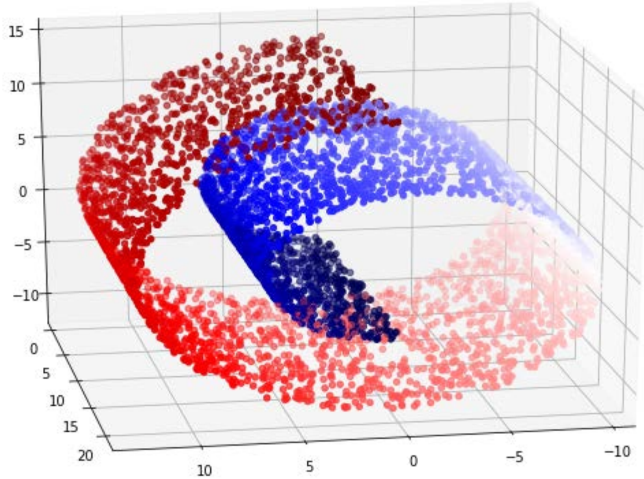
- Global properties on some manifolds cannot characterize the manifold well



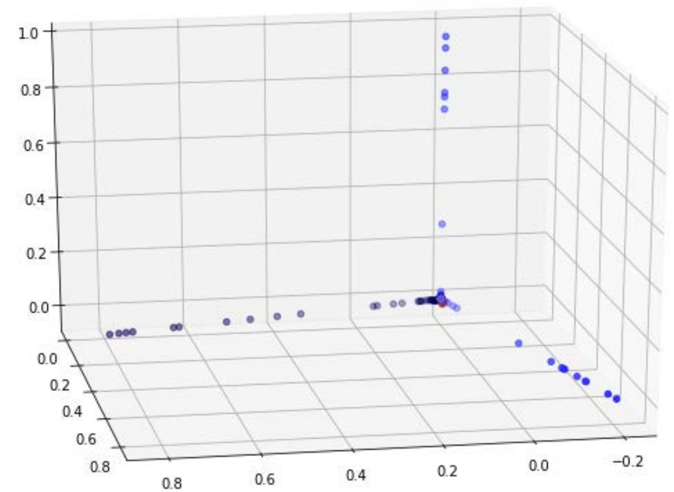
“The Swiss Roll”

- Techniques based on preserving global properties does poorly on the Swiss roll

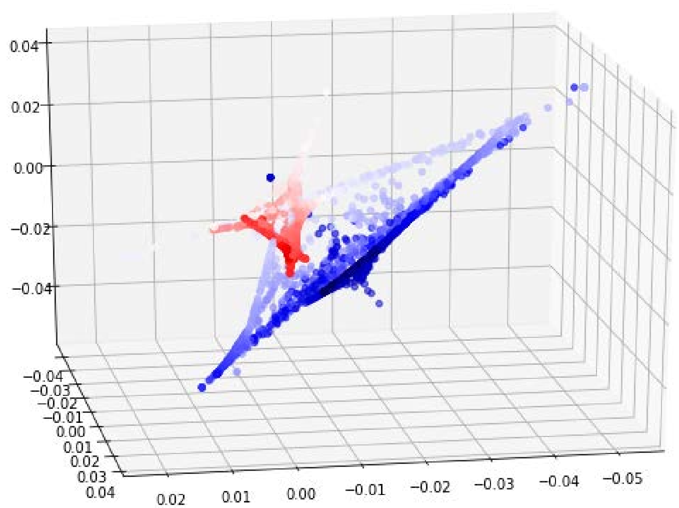
Drawback with global properties



Original Data



Data projected to main three eigenvectors in **kernel PCA** (rbf kernel)



On the other hand, methods such as **LLE** (in later slides) can handle the Swiss roll

Trends in Dimensionality Reduction

Global property preserving
Local manifold preserving
Graph property preserving

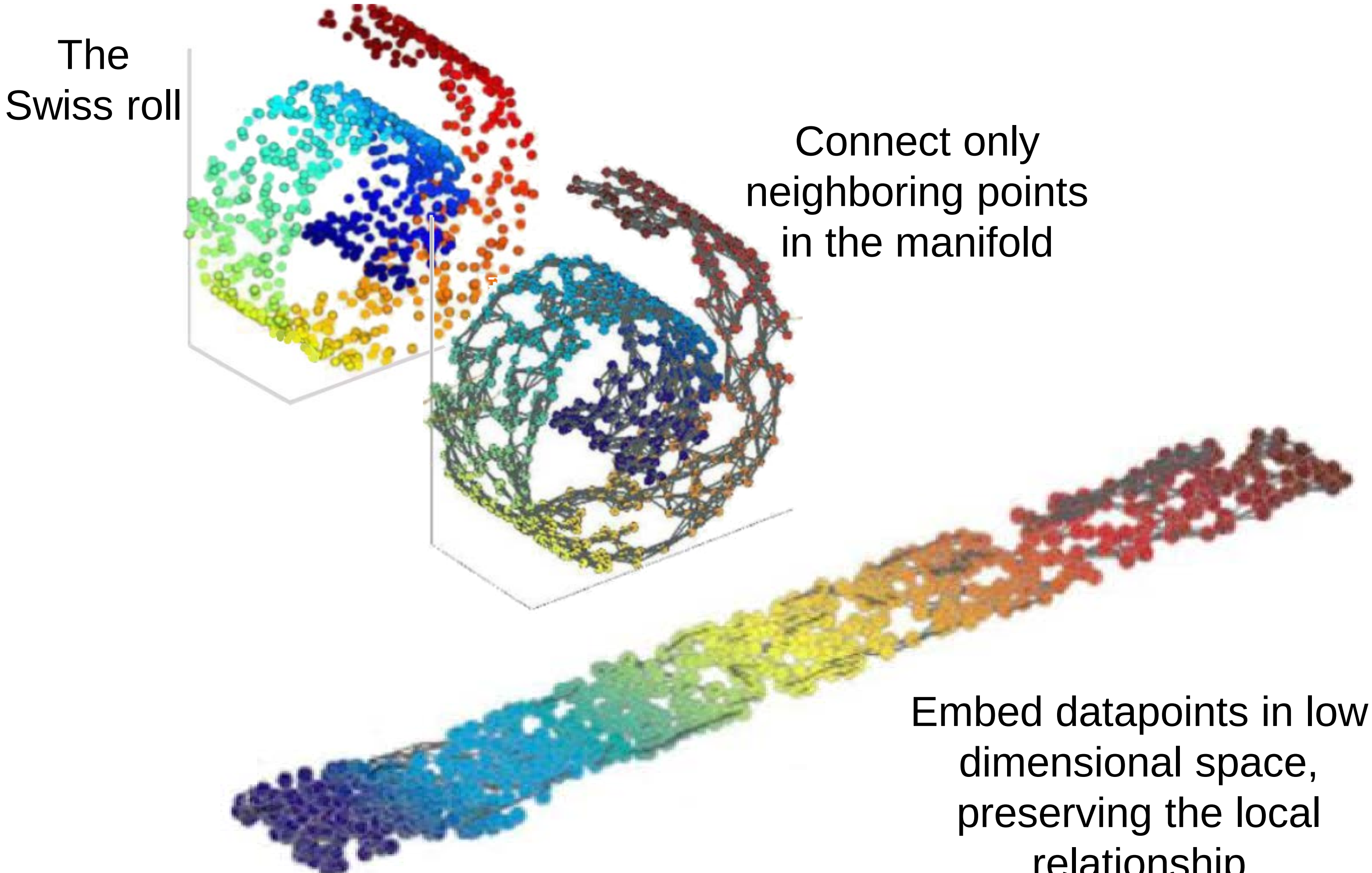
1901	PCA
1958	MDS
1963	SVM
1964	Kernel Perceptron
1969	Sammon's Mapping
1992	Kernel SVM
1997	Metric MDS
1998	Kernel PCA
2000	Isomap, LLE
2001	Laplacian Eigenmap
2008	t-SNE
2018	UMAP

Linear

Non-linear

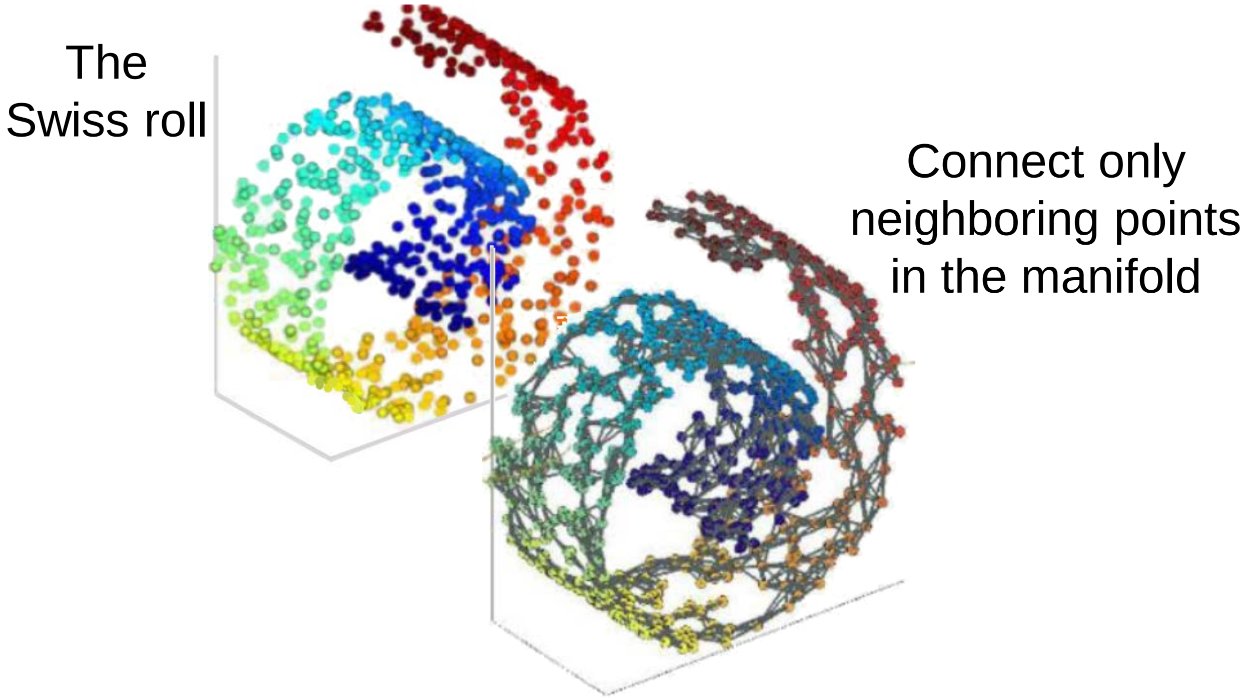
Non-global

Local structure preserving mapping



Weinberger and Saul. “Unsupervised Learning of Image Manifolds by Semi-definite Programming”, CVPR 2004

Isomap idea



- Isomap performs only the first step to find (Euclidean) distances of neighboring points
- Pairwise (geodesic) distances are estimated using the neighboring distances
- Then, MDS is used on the estimated geodesic distances

Isomap algorithm

1. Construct neighborhood graph
   - Find nearest k neighbors $N(x_i)$ of each point x_i
   - Construct a neighborhood graph by connecting x_i to the points in $N(x_i)$ with **Euclidean distance** set as edge weight
2. Compute (shortest) distance matrix M
   - Find shortest distance between pairwise points on the graph
3. Find eigenvectors of M using MDS (or PCA)

Isomap

- At first look, appear to be very different from kernel PCA (or PCA)
- However, from a kernel perspective, Isomap is similarly a kernel method
  - Discussed in Ham *et al*. "A kernel view of the dimensionality reduction of manifolds", 2003
  - Such a framework allows **mapping out-of-sample examples to the embedded space**
    - Discussed in Bengio *et al*. "Out-of-Sample extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering", 2003

Laplacian Eigenmap idea

- The normalized Laplacian L encodes structure of the graph
  - The eigenvectors of L known to encode important features of the graph (see slides on Spectral Clustering)
- The Laplacian can be considered as eigenfunctions similar to kernel functions
  - Discussed in Bengio *et al*. "Learning eigenfunctions links Spectral Embedding and Kernel PCA", 2004
  - Readily gives rise to using Laplacian in similar way as Kernel PCA

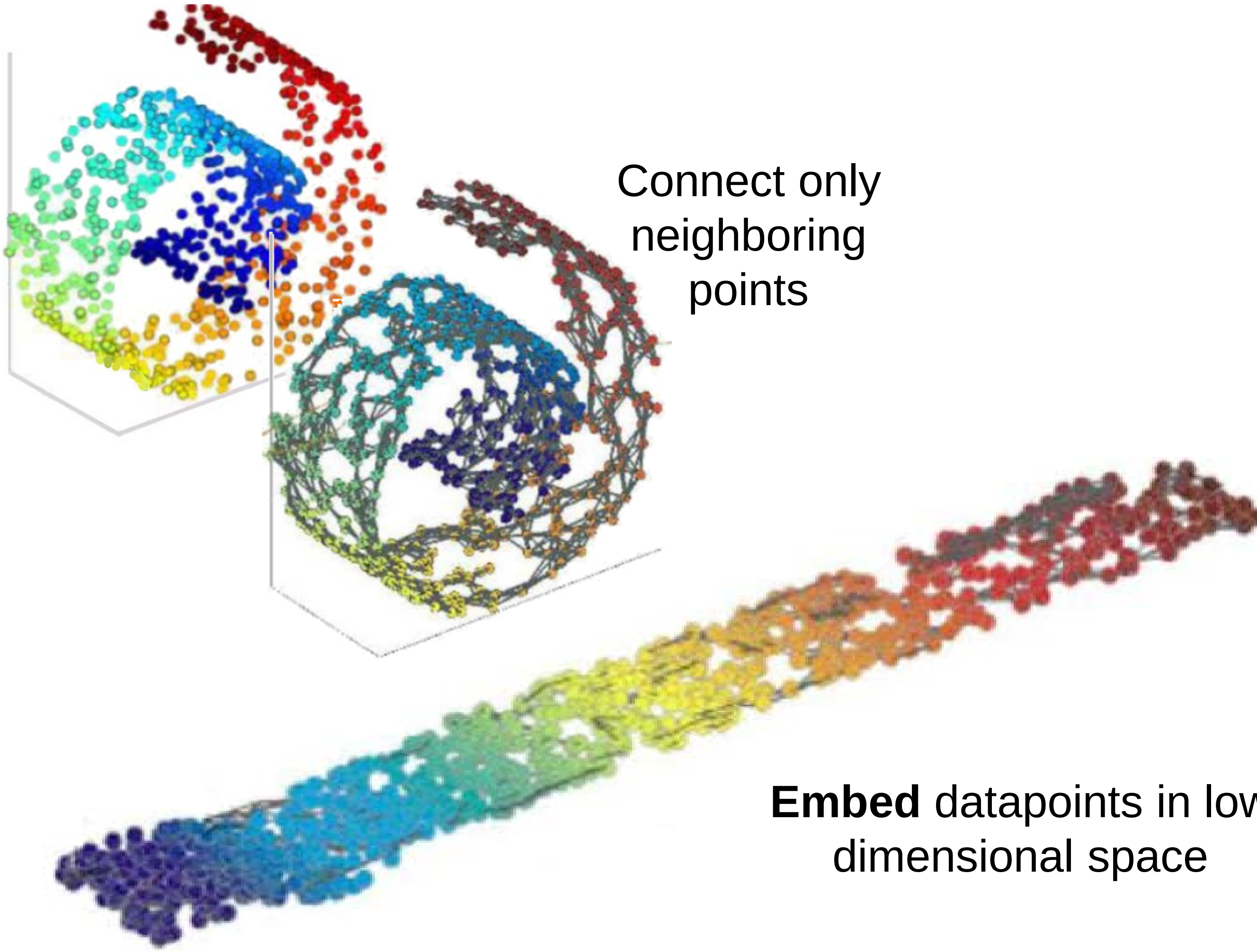
Laplacian Eigenmap algorithm

1. Construct neighborhood graph
   - Find nearest k neighbors $N(x_i)$ of each point x_i
   - Construct a neighborhood graph by connecting x_i to the points in $N(x_i)$ with **Gaussian heat function** set as edge weight
2. Construct normalized Laplacian L and degree matrix D
3. Find the eigenvectors for the generalized eigenvalue system $Lu = \lambda Du$

Laplacian Eigenmap

- Mapping of out-of-sample examples not immediately available like in Kernel PCA
  - Discussed in Bengio *et al*. "Out-of-Sample extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering", 2003

Locally Linear Embedding (LLE)



- LLE is the first algorithm that runs the full scheme

Locally Linear Embedding (LLE)

1. Construct neighborhood graph
   - Find nearest k neighbors $N(x_i)$ of each point x_i
2. Find matrix W which minimizes its sum of squares error in representing each x_i with its neighbors

   x_i as a **linear** combination of its neighbors

   - If suffices that for each i

     $$\text{error}(w_i) = \left\|x_i - \sum_{j \neq i} w_{ij} x_j\right\|^2$$

     is minimized
3. Find low dimensional $y_1, \ldots, y_n$ that is most consistent with W
   - Minimize

     $$\text{error}(y_1, \ldots, y_n) = \sum_{i=1}^{n} \left\|y_i - \sum_{j \neq i} w_{ij} y_j\right\|^2$$

LLE Step 2: Find matrix W

- For each i, find $w_{i1}, \ldots, w_{ik}$ such that $\left\| x_i - \sum_{j \neq i} w_{ij} x_j \right\|^2$ is minimized
  - Further require that $\sum_j w_{ij} = 1$
    1. Then, solution will be invariant to translation
       Let $x_j' \to x_j + c$. Then,
       $$\begin{aligned} x_j' - \sum_{j \neq i} w_{ij} x_j' &= x_j + c - \sum_{j \neq i} w_{ij} \left(x_j + c \right) \\ &= x_j + c - \sum_{j \neq i} w_{ij} x_j - c \\ &= x_j - \sum_{j \neq i} w_{ij} x_j \end{aligned}$$
    2. Also, w_{ij} can be interpreted as transition probability

LLE Step 2: Find matrix W

- For each i, find $w_{i1}, \ldots, w_{ik}$ such that $\left\| x_i - \sum_{j \neq i} w_{ij} x_j \right\|^2$ is minimized

1. Let $x_j' \to x_j - x_i$ (Center x_j)

   Then, $\left\| x_i' - \sum_{j \neq i} w_{ij} x_j' \right\|^2 = \left\| \sum_{j \neq i} w_{ij} x_j' \right\|^2$

2. Let $C_i = [x_1', \ldots, x_k']$

   Then, $\left\| \sum_{j \neq i} w_{ij} x_j' \right\|^2 = w_i^{\mathrm{T}} C_i C_i^{\mathrm{T}} w_i$

   Or, $\left\| \sum_{j \neq i} w_{ij} x_j' \right\|^2 = w_i^{\mathrm{T}} G_i w_i$ for $G_i = C_i C_i^{\mathrm{T}}$

$\Rightarrow$ Minimize $w_i^{\mathrm{T}} G_i w_i$ subject to $\sum_j w_{ij} = 1$

Cannot be done by eigendecomposition of G_i since constraint $\sum_j w_{ij} = 1$ cannot be fulfilled

Return to the Lagrange multiplier method

LLE Step 2: Find matrix W

- Minimize $w_i^{\mathrm{T}} G_i w_i$ subject to $\sum_j w_{ij} = 1$
  1. Use Lagrange multiplier to constrain $\sum_j w_{ij} = 1$
     That is, $\mathbf{1}^{\mathrm{T}} w_i - 1 = 0$,
     Lagrangian, $\mathcal{L}(w_i, \lambda) = w_i^{\mathrm{T}} G w_i - \lambda\left(\mathbf{1}^{\mathrm{T}} w_i - 1\right)$
     $\frac{\partial \mathcal{L}}{\partial w_i} = 2G_i w_i - \lambda \mathbf{1} = 0 \Rightarrow G_i w_i = \frac{\lambda}{2}\mathbf{1}$
     $\frac{\partial \mathcal{L}}{\partial \lambda} = \mathbf{1}^{\mathrm{T}} w_i - 1 = 0$
  2. If G is invertible
     $G_i w_i = \frac{\lambda}{2}\mathbf{1} \Rightarrow w_i = \frac{\lambda}{2} G_i^{-1}\mathbf{1}$
     Find $G_i^{-1}\mathbf{1}$ or solve linear equations $G_i w_i = \frac{\lambda}{2}\mathbf{1}$
     Then, scale λ such that $\sum_j w_{ij} = 1$

LLE Step 2: Find matrix W

3. If G is not invertible ($k \geq m$, rank deficient), use Tikhonov regularization
   Minimize $w_i^{\mathrm{T}} G_i w_i + \alpha w_i^{\mathrm{T}} w_i$ instead, subject to $||w_i|| = 1$, where α determines the degree of regularization

   $$\mathcal{L}(w_i, \lambda) = w_i^{\mathrm{T}} G w_i + \alpha w_i^{\mathrm{T}} w_i - \lambda\left(\mathbf{1}^{\mathrm{T}} w_i - 1\right)$$

   $$\frac{\partial \mathcal{L}}{\partial w_i} = 2G_i w_i + 2\alpha w_i - \lambda \mathbf{1} = 0$$

   $$(G_i + \alpha I) w_i = \frac{\lambda}{2} \mathbf{1}$$

   $$w_i = \frac{\lambda}{2} (G_i + \alpha I)^{-1} \mathbf{1}$$

   Find $w_i = (G_i + \alpha I)^{-1} \mathbf{1}$ or solve linear equations $(G_i + \alpha I) w_i = \mathbf{1}$. Scale λ such that $\sum_j w_{ij} = 1$

LLE Step 3: Find low-D $y_1, \ldots, y_n$

- Find $y_1, \ldots, y_n \in \mathbb{R}^q$ such that $\left\| y_i - \sum_{j \neq i} w_{ij} y_j \right\|^2$ is minimized
  - To restrict equivalent solutions due to translation, require that $\sum_i y_i = 0$ (centered)
  - Let Y be the matrix formed by y_i as the rows, and u_i be the columns of Y. To ensure that u_i are orthogonal, require that $Y^{\mathrm{T}} Y = nI$

i.e. $Y^{\mathrm{T}}Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} \begin{bmatrix} u_1 & \ldots & u_n \end{bmatrix} = \begin{bmatrix} u_1 u_1 & \ldots & u_1 u_n \\ \vdots & \ddots & \vdots \\ u_n u_1 & \ldots & u_n u_n \end{bmatrix}$

$\begin{bmatrix} u_1 u_1 & \ldots & u_1 u_n \\ \vdots & \ddots & \vdots \\ u_n u_1 & \ldots & u_n u_n \end{bmatrix} = nI \Rightarrow u_i u_j = 0$ for $i \neq j$

LLE Step 3: Find low-D $y_1, \ldots, y_n$

- Find $y_1, \ldots, y_n \in \mathbb{R}^q$ such that $\left\| y_i - \sum_{j \neq i} w_{ij} y_j \right\|^2$ is minimized subject to $Y^{\mathrm{T}} Y = nI$ and $\sum_i y_i = 0$

$$\sum_{i=1}^{n} \left(y_i - \sum_j w_{ij} y_j \right)^2$$

$$= \sum_i^n y_i^2 - y_i \left(\sum_j w_{ij} y_j \right) - \left(\sum_j w_{ij} y_j \right) y_i + \left(\sum_j w_{ij} y_j \right)^2$$

$$= Y^{\mathrm{T}} Y - Y^{\mathrm{T}} (WY) - (WY)^{\mathrm{T}} Y + (WY)^{\mathrm{T}} (WY)$$

$$= \left((I - W) Y \right)^{\mathrm{T}} \left((I - W) Y \right)$$

$$= Y^{\mathrm{T}} (I - W)^{\mathrm{T}} (I - W) Y$$

$= Y^{\mathrm{T}} M Y$ where $M = (I - W)^{\mathrm{T}} (I - W)$

LLE Step 3: Find low-D $y_1, \ldots, y_n$

- Minimize $Y^T MY$ where $M = (I - W)^{\mathrm{T}}(I - W)$ subject to $Y^{\mathrm{T}}Y = nI$ and $\sum_i y_i = 0$

  Consider first case $q = 1$ (that is, Y is column vector and $I = 1$)

  $\mathcal{L}(Y, \mu) = Y^T MY - \mu\left(\frac{Y^{\mathrm{T}}Y}{n} - 1\right) - \nu Y$

  $\frac{\partial \mathcal{L}}{\partial Y} = 2MY - 2\frac{\mu}{n}Y - \nu = 0 \Rightarrow MY = \frac{\mu}{n}Y$ (Set $\nu = 0$)

  Hence Y is a eigenvector of M

  For $q \geq 2$, simply observe that by the min-max theorem the eigenvectors for M minimizes $Y^T MY$

  Finally, since $W\mathbf{1} = \mathbf{1}$, $(I - W)\mathbf{1} = 0$
  $\Rightarrow (I - W)^{\mathrm{T}}(I - W)\mathbf{1} = 0 \Rightarrow M\mathbf{1} = 0$
  $\Rightarrow Y = \mathbf{1}$ is a eigenvector of zero eigenvalue (excluded)

LLE algorithm

1. Construct neighborhood graph

   Find nearest k neighbors $N(x_i)$ of each point x_i

2. Find matrix W which minimizes its sum of squares error in representing each x_i with its neighbors

   For each i

   Let $x_j' \rightarrow x_j - x_i$ and Let $C_i = [x_1', \ldots, x_k']$

   Solve $G_i w_i = \mathbf{1}$ where $G_i = C_i C_i^{\mathrm{T}}$

   Scale w_i such that $w_i \mathbf{1} = 1$

   Collect w_i into W

3. Find low dimensional $y_1, \ldots, y_n$ that is most consistent with W

   Find eigenvectors for $M = (I - W)^{\mathrm{T}}(I - W)$ with smallest eigenvalues

LLE out-of-sample examples

- Mapping of out-of-sample examples not immediately available like in Kernel PCA
  - Discussed in Bengio *et al*. "Out-of-Sample extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering", 2003

Comparison

	Isomap	Laplacian Eigenmap	LLE
Edge weight	Approximated geodesic distance	Gaussian $e^{-d^2/\sigma}$ (transition probability)	Coefficients w_{ij} in reconstructing x_i (transition probability)
Pairwise edge or neighborhood only	Pairwise Distant pairs use shortest path distance	Pairwise Distant pairs near zero	Neighborhood only Matrix contains mostly zeros
Matrix to decompose	Edge weight	Normalized Laplacian	Edge weight
Embedding into lower dimensional space	Use principal eigenvectors from MDS	Use principal eigenvectors that retain graph structure	Find low dimensional points that give the same w_{ij} (shown to be principal eigenvectors)
Edge weight preservation	Preserves Euclidean distance	Preserves $e^{-d^2/\sigma}$	Normalized, scale-free