

Non-Linear Dimensionality Reduction

Presentation by Daniel Steinberg

CS3750, Fall 2011

Some Slides are New but Most are From or Have been Adapted From:
Fall 2011 Slides by Md. Abedul Haque,
Fall 2011 Slides by Saeed Amizadeh,
And Slides by Iyad Batal

Outline

- Linear Dimensionality Reduction
 - PCA
 - MDS
- Non-Linear Dimensionality Reduction
 - Laplacian Eigenmaps
 - Locally Linear Embedding
 - Isomap

Outline

- Linear Dimensionality Reduction
 - PCA
 - MDS
- Non-Linear Dimensionality Reduction
 - Laplacian Eigenmaps
 - Locally Linear Embedding
 - Isomap

Dimensionality Reduction

- Linear Dimensionality Reduction Methods
 - PCA
 - Finds a low-dimensional embedding of the data points that maximizes the variance of the projected data
 - Classical MDS
 - Finds an embedding that preserves the inter-point distances.
 - Equivalent to PCA when those distances are Euclidean.

Multi-Dimensional Scaling (MDS)

- Multi-Dimensional Scaling [Cox and Cox, 1994] .
- MDS give points in a low dimensional space such that the **Euclidean distances** between them best approximate the original distance matrix.
Given distance matrix
(I is number of observations)

$$\Delta := \begin{pmatrix} \delta_{1,1} & \delta_{1,2} & \cdots & \delta_{1,I} \\ \delta_{2,1} & \delta_{2,2} & \cdots & \delta_{2,I} \\ \vdots & \vdots & & \vdots \\ \delta_{I,1} & \delta_{I,2} & \cdots & \delta_{I,I} \end{pmatrix}.$$

Map input point pairs (x_i, x_j) to (z_i, z_j) such that $\|z_i - z_j\| \approx \delta_{i,j}$

- Classical MDS: the norm $\| \cdot \|$ is the **Euclidean distance**.
- $\min_{z_1, \dots, z_I} \sum_{i < j} (\|z_i - z_j\| - \delta_{i,j})^2$
 - From Wikipedia: A solution may then be found by numerical optimization techniques. For some particularly chosen cost functions, minimizers can be stated analytically in terms of matrix eigendecompositions.
 - “Citation Needed” on Wikipedia
 - This is the same approach of how Laplacian Eigenmap is solved

MDS example

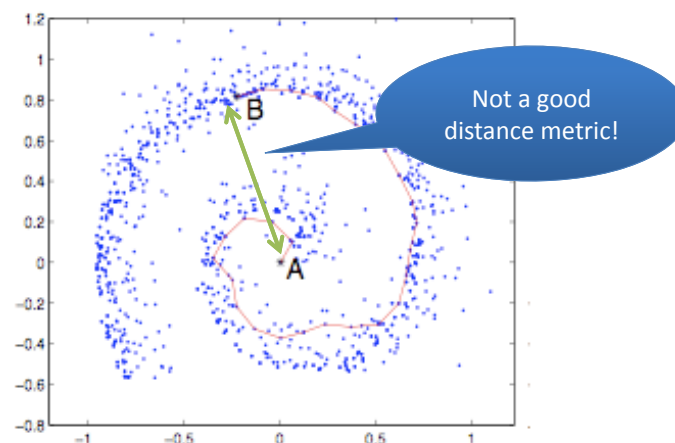
- Given pairwise distances between different cities in 3D (Δ matrix), plot the cities on a 2D plane (recover location)



PCA and MDS relation

- Preserve Euclidean distances = retaining the maximum variance.
- *Classical MDS is equivalent to PCA when the distances in the input space are the **Euclidean distance**.*
- If we have only a distance matrix (we don't know the points in the original space), we cannot perform PCA
- Both PCA and MDS are invariant to space rotation

Underestimation of distance



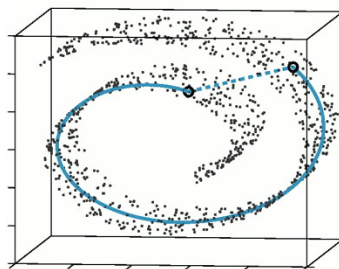
- For local distances, Euclidean distance may be sufficient
- For global distances, Euclidean distance may underestimate
 - In example above, desired distance between A and B is along the red line. Euclidean distance is an underestimate, as can be seen by the green arrow.

Outline

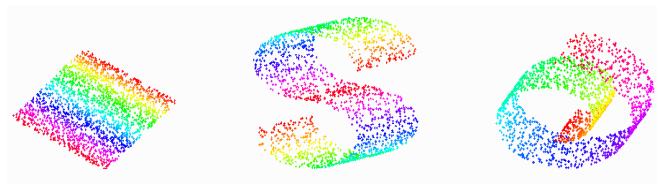
- Linear Dimensionality Reduction
 - PCA
 - MDS
- Non-Linear Dimensionality Reduction
 - Laplacian Eigenmaps
 - Locally Linear Embedding
 - Isomap

Non-Linear Dimensionality Reduction

- A special class of problem
 - Low dimensional data lying on a manifold in a higher dimensional space

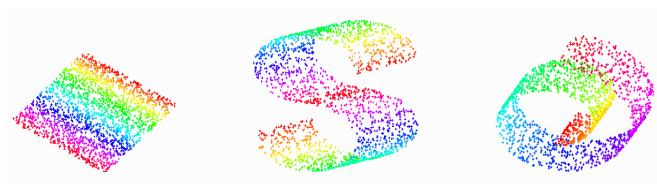


Manifolds



- Three examples of manifolds
- All three are two-dimensional manifolds embedded in 3 dimensions
 - Linear, “S”-shape, “Swiss roll”
- For all three, we would like to recover:
 - That the data is only two-dimensional
 - “Consistent” locations for the data in 2D

Manifolds



(a)

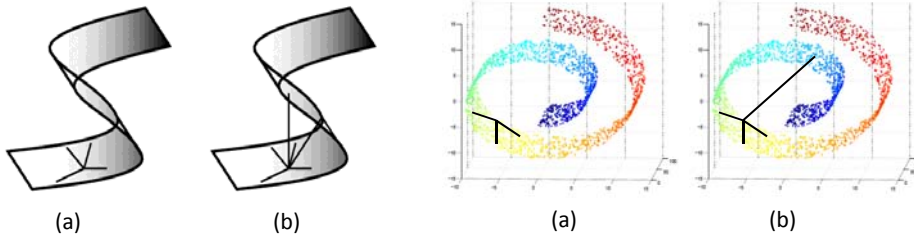
(b)

(c)

- PCA : works for (a)
- Doesn't do much good for (b) or (c)
 - Linear subspace doesn't explain it well
- What do we mean by “consistent locations”?
 - Preserve local relationships and structure
 - One possibility: preserve distances

Preserving Local/Global Relationships

- MDS – produces a linear embedding
 - Preserved *all pairwise distances*



- Nonlinear manifold:
 - local distances (a) make sense
 - but, global distances (b) don't respect the geometry

Dimensionality Reduction

- Methods that preserve local structure
 - Laplacian Eigenmaps
 - Locally Linear Embedding (LLE)
 - Isomap

Outline

- Linear Dimensionality Reduction
 - PCA
 - MDS
- Non-Linear Dimensionality Reduction
 - Laplacian Eigenmaps
 - Locally Linear Embedding
 - Isomap

Laplacian Matrix Components

- Given Graph G
- Let e_{ij} = edge weight from node i to node j
- Weight Matrix W (e.g., Heat Kernel)

$$w_{ij} = \begin{cases} e_{ij} & \text{if } i \text{ is adjacent to } j \\ 0 & \text{otherwise} \end{cases}$$

- Matrix D
 - Diagonal Matrix with diagonal elements containing column sums of W. Since W is symmetric, using row sums is equivalent.
 - Often Referred to as a Degree Matrix when all edge weights are 1 (that is, when W is the adjacency matrix)

$$d_{ij} = \begin{cases} \sum_j w_{ij} & \text{if } i = j \text{ (diagonal element)} \\ 0 & \text{otherwise} \end{cases}$$

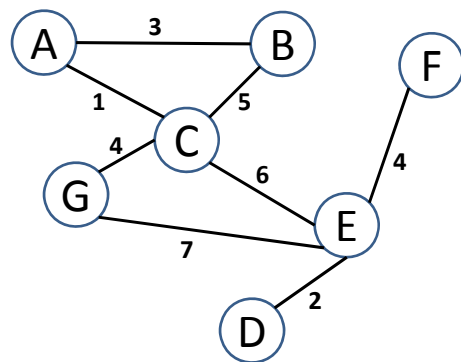
Laplacian Matrix

- Laplacian Matrix $L = D - W$
- L is positive semidefinite (PSD)
 - We will show why later
 - We will use this property for optimization

A symmetric square Matrix $A \in \mathbb{R}^{n \times n}$ is positive semidefinite if for all $x \in \mathbb{R}^n$, $x^T A x \geq 0$.

$x^T A x$ is called a *quadratic form*.

Example: Laplacian Matrix of a Graph



Note: Node A corresponds to row 1 and column 1, Node B to row 2 and column 2, ..., Node G to row 7 and column 7

$$W = \begin{bmatrix} 0 & 3 & 1 & 0 & 0 & 0 & 0 \\ 3 & 0 & 5 & 0 & 0 & 0 & 0 \\ 1 & 5 & 0 & 0 & 6 & 0 & 4 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 6 & 2 & 0 & 4 & 7 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 & 7 & 0 & 0 \end{bmatrix}$$

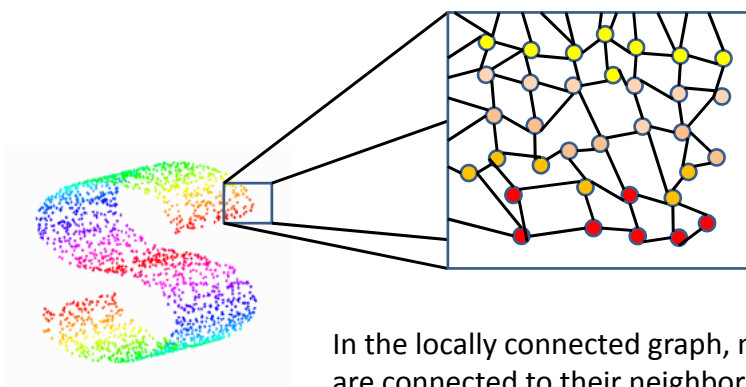
$$D = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 16 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 19 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 11 \end{bmatrix}$$

$$L = D - W = \begin{bmatrix} 4 & -3 & -1 & 0 & 0 & 0 & 0 \\ -3 & 8 & -5 & 0 & 0 & 0 & 0 \\ -1 & -5 & 16 & 0 & -6 & 0 & -4 \\ 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & -6 & -2 & 19 & -4 & -7 \\ 0 & 0 & 0 & 0 & -4 & 4 & 0 \\ 0 & 0 & -4 & 0 & -7 & 0 & 11 \end{bmatrix}$$

Laplacian Eigenmaps Overview

- We have n data points in d dimensions
- We want to project the data onto a lower dimension k
- In lower dimension, relative distances between points should preserve relative distances (along the manifold) from the higher dimensional space.
- Algorithm Overview
 - Form a graph G where nodes correspond to data points and are connected to **local** nodes
 - Weight the edges of graph G according to local distances between nodes [data points]
 - Compute matrices W , D , and Laplacian matrix L of graph G
 - Compute Eigenvalues and Eigenvectors of $(D^{-1}L)$
 - Use k eigenvectors corresponding to smallest non-zero eigenvalues, to project data onto k dimensions.

Local Neighborhood Graph



In the locally connected graph, nodes are connected to their neighbors. One way to choose neighbors is k -nearest-neighbors.

Notice that nodes are **not** connected to **distant** nodes.

Laplacian Eigenmaps Algorithm

- Step 1: Construct the graph
 - Construct the adjacency graph G by connecting neighboring nodes (i,j)
- Neighbors selection
 - ϵ -neighborhoods $\|x_i - x_j\|^2 < \epsilon$
 - Advantages
 - Geometrically motivated
 - Relationships are symmetric
 - Disadvantages
 - Can lead to Disconnected graphs
 - Difficult to choose ϵ
 - n nearest neighbors
 - Advantages
 - Easier to choose
 - No disconnected graphs
 - Disadvantages
 - Less geometrically intuitive (asymmetric relationships)

Laplacian Eigenmaps Algorithm

- Step 2: Choose the weights
 - Simple-minded
 - 1 if connected
 - 0 otherwise
 - Heat Kernel
 - $W_{ij} = e^{\frac{-\|x_i - x_j\|^2}{t}}$ if connected
 - 0 otherwise
 - Heat kernel with $t = \infty$ is equivalent to the *simple-minded* approach

Laplacian Eigenmaps Algorithm

- Step 3: Eigenmaps
 - Construct Laplacian matrix
 - Construct diagonal weight matrix D from weight matrix. $D_{ii} = \sum_j W_{ij}$
 - Construct Laplacian matrix $L = D - W$
 - W, and thus L, will be sparse. Nodes [data points] are only connected to nearby [similar] nodes.
 - Laplacian is a symmetric, positive semi-definite matrix
 - Compute eigenvalues and eigenvectors of the generalized eigenvector problem:

$$Lf = \lambda Df$$
 - Alternative (but equivalent) formulation:
Find eigenvalues and eigenvectors of matrix $(D^{-1}L)$

$$(D^{-1}L)f = \lambda f$$

Laplacian Eigenmaps Algorithm

- Step 3: Eigenmaps

$$Lf = \lambda Df$$
 - Let, f_0, f_1, \dots, f_{k-1} be the solutions ordered according to increasing eigenvalues

$$Lf_0 = \lambda_0 Df_0$$


$$Lf_1 = \lambda_1 Df_1$$

$$\dots$$

$$Lf_{k-1} = \lambda_{k-1} Df_{k-1}$$

$$0 = \lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_{k-1}$$
 - We leave out eigenvector f_0 . Take the next m eigenvectors to construct m-dimensional embedding $(f_1(i), \dots, f_m(i))$

Laplacian Eigenmaps Motivation

- Consider the problem of mapping weighted graph G **into a line** so that the connected nodes stay as close as possible
- Let $\mathbf{y} = (y_1, y_2, \dots, y_n)^T$ be such a map
- A “reasonable criterion” for a good map is to minimize $\sum_{ij} (y_i - y_j)^2 W_{ij}$
- $\sum_{ij} (y_i - y_j)^2 W_{ij} = 2\mathbf{y}^T \mathbf{L} \mathbf{y}$  Next Slide Shows Derivation
- $\operatorname{argmin}_{\mathbf{y}} \mathbf{y}^T \mathbf{L} \mathbf{y}$ subject to $\mathbf{y}^T \mathbf{D} \mathbf{y} = 1$
 - The constraint removes arbitrary scaling factor
- From equation * above, L is positive semidefinite since $(y_i - y_j)^2 \geq 0$ and $W_{ij} \geq 0$
- Since L is positive semidefinite, the solution to our minimization problem is the generalized eigenvalue solution
 - $\mathbf{L} \mathbf{y} = \lambda \mathbf{D} \mathbf{y}$
- 1 is an eigenvector corresponding to eigenvalue 0.
- To eliminate this trivial solution, add constraint $\mathbf{y}^T \mathbf{D} \mathbf{1} = 0$

Derive $\sum_{ij} (y_i - y_j)^2 W_{ij} = 2\mathbf{y}^T \mathbf{L} \mathbf{y}$ (from previous slide)

$$\begin{aligned}
 \sum_{i,j} (y_i - y_j)^2 W_{i,j} &= \sum_{i,j} (y_i - y_j)(y_i - y_j) W_{i,j} \\
 &= \sum_{i,j} (y_i^2 - 2y_i y_j + y_j^2) W_{i,j} \\
 &= \sum_{i,j} y_i^2 W_{i,j} - \sum_{i,j} 2y_i y_j W_{i,j} + \sum_{i,j} y_j^2 W_{i,j} \\
 &= \sum_i y_i^2 \sum_j W_{i,j} - 2 \sum_{i,j} y_i y_j W_{i,j} + \sum_j y_j^2 \sum_i W_{i,j} \\
 &= \sum_i y_i^2 \sum_j W_{i,j} + \sum_j y_j^2 \sum_i W_{i,j} - 2 \sum_{i,j} y_i y_j W_{i,j} \\
 &= \sum_i y_i^2 \mathbf{D}_{i,i} + \sum_j y_j^2 \mathbf{D}_{j,j} - 2 \sum_{i,j} y_i y_j W_{i,j} \\
 &= 2 \sum_i y_i^2 \mathbf{D}_{i,i} - 2 \sum_{i,j} y_i y_j W_{i,j} \\
 &= 2\mathbf{y}^T \mathbf{D} \mathbf{y} - 2\mathbf{y}^T \mathbf{W} \mathbf{y} \\
 &= 2\mathbf{y}^T (\mathbf{D} - \mathbf{W}) \mathbf{y} \\
 &= 2\mathbf{y}^T \mathbf{L} \mathbf{y}
 \end{aligned}$$

Notes:

Given a symmetric square Matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ and $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{x}^T \mathbf{A} \mathbf{x}$ is called a *quadratic form*.

$$\begin{aligned}
 \mathbf{x}^T \mathbf{A} \mathbf{x} &= \sum_i x_i (\mathbf{A} \mathbf{x})_i \\
 &= \sum_i x_i \left(\sum_j \mathbf{A}_{i,j} x_j \right) \\
 &= \sum_i \sum_j \mathbf{A}_{i,j} x_i x_j
 \end{aligned}$$

If \mathbf{A} is a diagonal matrix,

$$\begin{aligned}
 \mathbf{x}^T \mathbf{A} \mathbf{x} &= \sum_i x_i (\mathbf{A} \mathbf{x})_i \\
 &= \sum_i x_i (\mathbf{A}_{i,i} x_i) \\
 &= \sum_i x_i^2 \mathbf{A}_{i,i}
 \end{aligned}$$

Also, recall from earlier:

- $\sum_j W_{i,j} = \mathbf{D}_{i,i}$
- $\mathbf{L} = \mathbf{D} - \mathbf{W}$

Laplacian Eigenmaps Beyond the Line

- How to find the embedding into m-dimensional space?
- The embedding is $Y = [\mathbf{y}_1 \mathbf{y}_2 \dots \mathbf{y}_m]$
- Minimize Objective function:

$$\sum_{ij} ||\mathbf{y}^{(i)} - \mathbf{y}^{(j)}||^2 W_{ij} = \text{tr}(Y^T L Y) \text{ i.e.}$$

$$\underset{Y}{\text{argmin}} \text{tr}(Y^T L Y) \quad \text{subject to } Y^T L Y = I$$

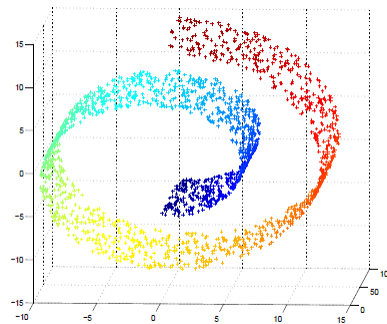
- Solution is provided by the matrix of eigenvectors corresponding to the lowest eigenvalues of the generalized eigenvalue problem
 $L\mathbf{y} = \lambda D\mathbf{y}$
- Ignore Zero Eigenvalue
 - In one-dimensional motivation, we imposed this as an additional constraint

Laplacian Eigenmaps

- So each eigenvector is a function from nodes to \mathbb{R} in a way that "close by" points are assigned "close by" values.
- The eigenvalue of each eigenfunction gives a measure of how "close by" are the values of close by points
- By using the first m eigenfunctions for determining our m-dimensions we have our solution.

Laplacian Eigenmap Example

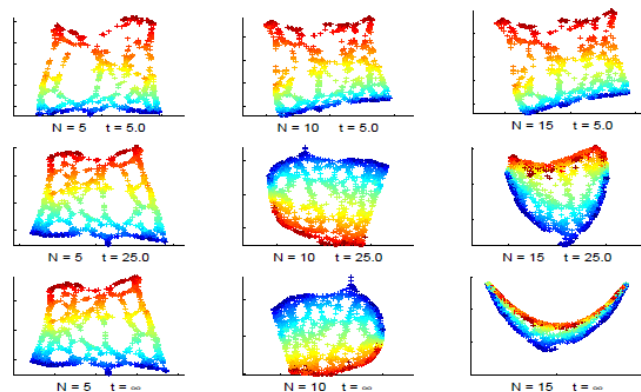
- Swiss roll



2000 random data points on the manifold

Laplacian Eigenmap Example

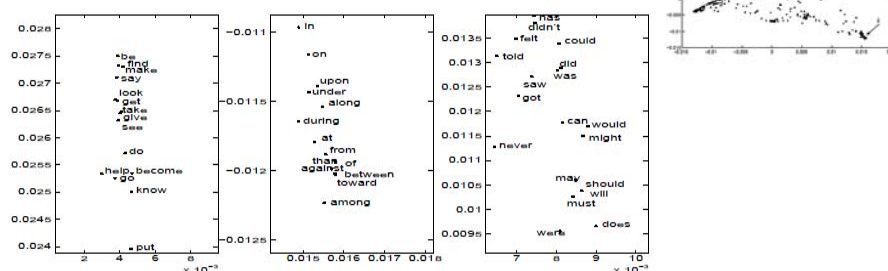
- 2D embedding of the swiss roll



Free parameters, N and t . N = Number of neighbors, t = Heat kernel parameter

Laplacian Eigenmap Example

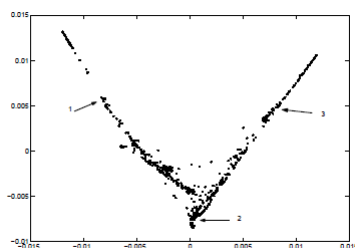
- 300 most frequent words from Brown corpus
- Each word is represented by a 600 dimensional vector
- Laplacian Eigenmap with $N = 14$, $t = \text{inf}$



Fragments labelled by arrows, from left to right. The first is exclusively infinites of verbs, the second contains prepositions and the third mostly modal and auxiliary verbs

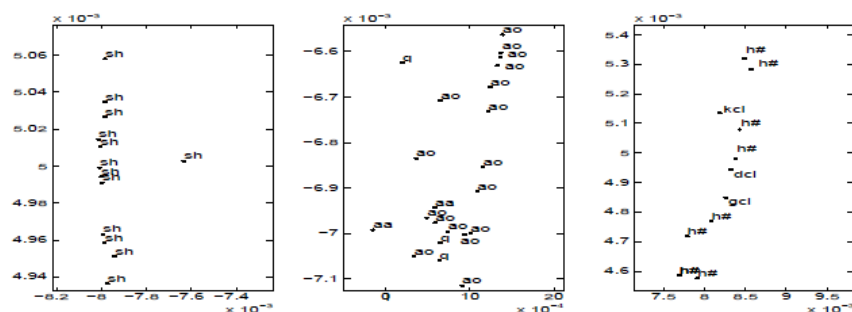
Laplacian Eigenmap Example: Speech

- Speech signal is high dimensional but distinctive phonetic dimensions are few
- 30 ms window at 5 ms interval
- 256 Fourier coefficients for each 30 ms chunk
- 685 such vectors



685 speech data points plotted in the two dimensional Laplacian spectral representation

Laplacian Eigenmap Example: Speech



A blowup of the three selected regions. The data points corresponding to the same region have similar phonetic identity

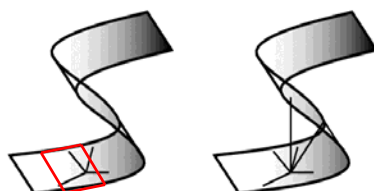
Outline

- Linear Dimensionality Reduction
 - PCA
 - MDS
- Non-Linear Dimensionality Reduction
 - Laplacian Eigenmaps
 - Locally Linear Embedding
 - Isomap

Locally Linear Embedding (LLE)

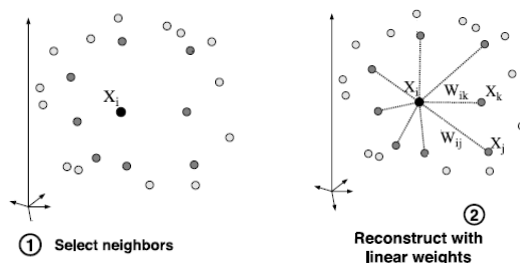
Sam T. Roweis, L. K. Saul 2000

- Manifold Characteristics/Key Assumption
 - Provided there is sufficient data, we expect each data point and its neighbors to lie on or close to a **locally linear patch**



LLE Algorithm

- Step 1:
 - Assign neighbors to each data point X_i
- Step 2
 - Characterize the local geometry of linear patches by linear coefficients that **reconstruct each point from its neighbors**

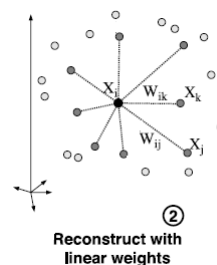


LLE Algorithm

- Step 2: How to assign weights?
 - Minimize cost function measuring reconstruction error

$$\epsilon(W) = \sum_i |X_i - \sum_j W_{ij} X_j|^2$$

- Weight W_{ij} summarizes the contribution of the j^{th} data point to the i^{th} reconstruction
- Assign weights under two constraints
- $W_{ij} = 0$ if X_j does not belong to set of neighbors of X_i
- The rows of the weight matrix sum to one i.e. $\sum_j W_{ij} = 1$
- Closed-form solution, Constrained Least Squares Problem (appendix A in paper)

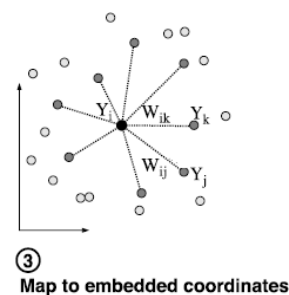


LLE Algorithm

- Step 3: Map to embedded coordinates
 - Each high-dimensional observation X_i is mapped to a low-dimensional vector Y_i
 - Choose Y_i to minimize the embedding cost function

$$\Phi(Y) = \sum_i \left| \vec{Y}_i - \sum_j W_{ij} \vec{Y}_j \right|^2$$

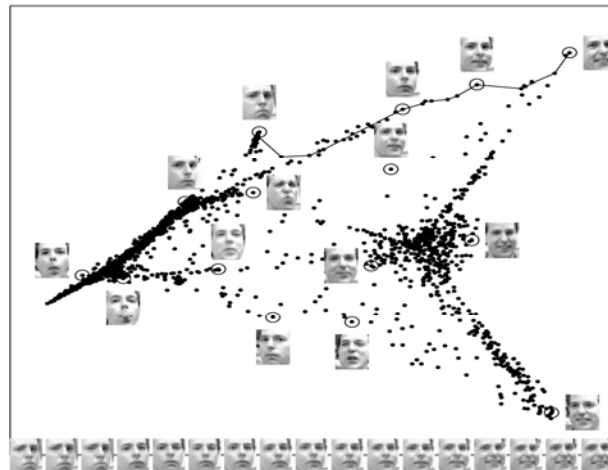
- The cost function can be minimized (subject to constraints) by solving a sparse $N \times N$ eigenvalue problem (appendix B in paper).
 - Bottom d non-zero eigenvectors



LLE Algorithm

- The constrained weights obey an important symmetry
 - For a particular data point, the weights are invariant to rotation, rescaling and translation of the data point and its neighbors.
- The same weights that reconstruct the data points in D dimensions should reconstruct it in the manifold coordinate in d dimensions.
 - The weights characterize the intrinsic geometric properties of each neighborhood.

LLE Example



Images of faces mapped into the embedding space described by the first two coordinates of LLE. Representative faces are shown next to circled points. The bottom images correspond to points along the top-right path (linked by solid line) illustrating one particular mode of variability in pose and expression.

Effect of K

- Require dense data points on the manifold for good estimation

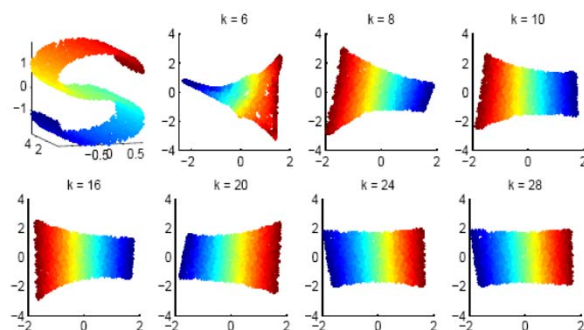


FIG. 5. S-curve (top left) and computed 2D coordinates by LLE with various neighborhood size k .

LLE and Laplacian Eigenmap

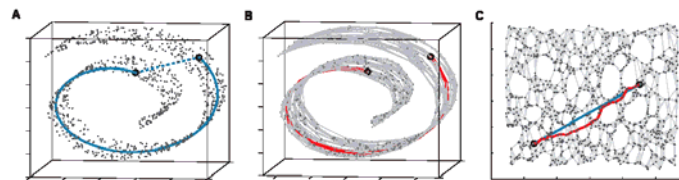
- LLE is connected with Laplacian Eigenmap
- LLE minimizes $y^T(I-W)^T(I-W)y$ which reduces to finding eigenvectors of $(I-W)^T(I-W)$
- They show that finding eigenvectors of $(I-W)^T(I-W)$ can be re-interpreted as finding eigenvectors of iterated Laplacian L^2 . Eigenvectors of L^2 coincide with those of L .

Outline

- Linear Dimensionality Reduction
 - PCA
 - MDS
- Non-Linear Dimensionality Reduction
 - Laplacian Eigenmaps
 - Locally Linear Embedding
 - Isomap

Isomap

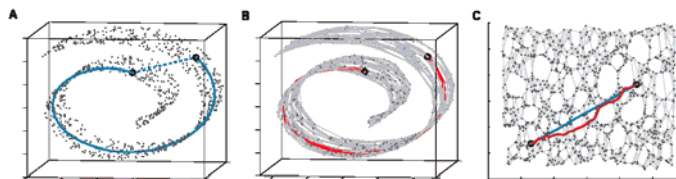
Josh. Tenenbaum, Vin de Silva, John Langford 2000



- Classical MDS – uses Euclidean distance
- What we really want:
 - Distance measurements along manifold (geodesics)
 - Find low dimensional reconstruction which also has these geodesic distances
- Isomap: Classical MDS with geodesic distances.

Isomap Algorithm

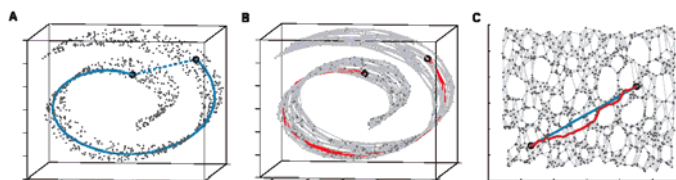
Josh. Tenenbaum, Vin de Silva, John Langford 2000



- Step 1: Construct Neighborhood graph (G)
 - Define the graph G over all data points by connecting points i and j if they are
 - Closer than ϵ – (ϵ -Isomap)
 - If i is one of the k nearest neighbors of j (k -isomap)
 - Set edge lengths equal to $d_x(i, j)$

Isomap Algorithm

Josh. Tenenbaum, Vin de Silva, John Langford 2000



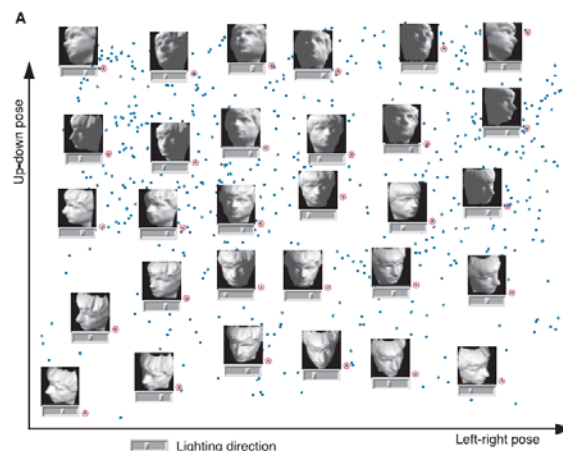
- Step 2: Compute Shortest Paths in G
 - Floyd's algorithm ($O(n^3)$) or Dijkstra's algorithm to find D_G
- Step 3: Construct d -dimensional embedding
 - Apply classical MDS on D_G to find d -dimensional embedding

Isomap

- Advantages
 - Non-linear. Preserves intrinsic geometry of the data.
 - Non-iterative polynomial time algorithm
 - Guarantee of global optimality
 - For intrinsically Euclidean manifolds, a guarantee of asymptotic convergence to the true structure
 - the ability to discover manifolds of arbitrary dimensionality
- Disadvantages
 - From Wikipedia
 - "short-circuit errors" from noise or if k is too large.
 - "Even a single short-circuit error can alter many entries in the geodesic distance matrix, which in turn can lead to a drastically different (and incorrect) low-dimensional embedding"

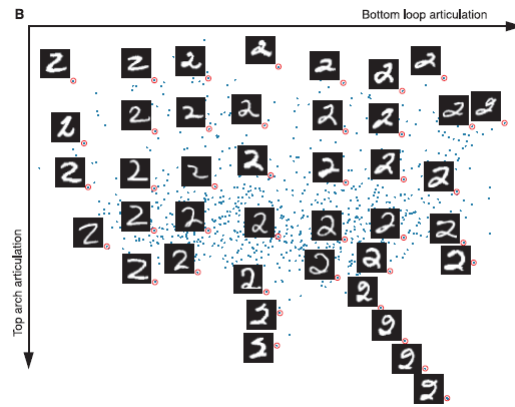
Isomap: Examples

- Dimensionality reduction for visual perception
 - 64x64 image
 - 698 raw images
 - Isomap ($k=6$)



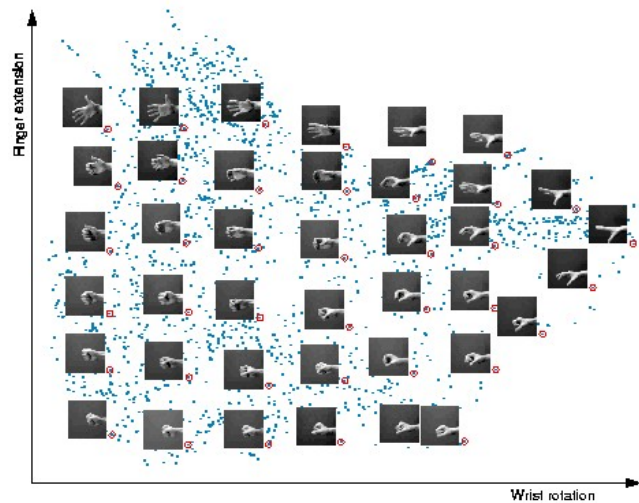
Isomap: Examples

- Handwritten '2'
 - 1000 handwritten 2s
 - Isomap ($\epsilon=4.2$)



Isomap: Examples

- Hand images
 - 64x64 image
 - 2000 images
 - Isomap ($k=6$)



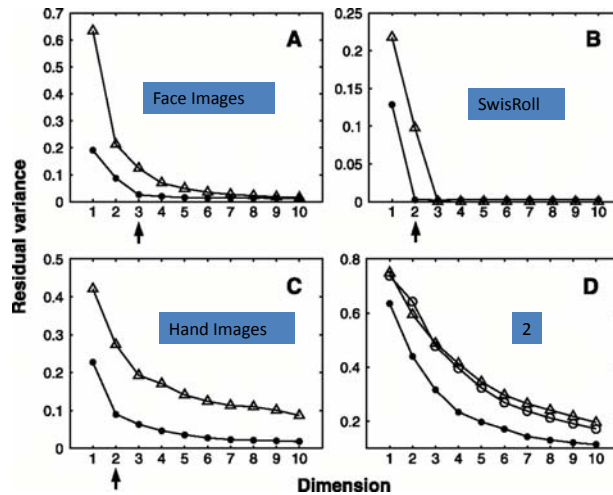
Residual Variance

Residual Variance

$$= 1 - \rho(D_m, D_y)^2$$

- where ρ is the correlation coefficient operator
- D_m contains **estimated** distances between data points in the original space
- and D_y contains distances between data points in projected space

Residual Variance is zero if estimated distances between points in high dimensional space is perfectly correlated with distances between corresponding points in projected space.



Isomap Vs LLE

- Isomap
 1. MDS on the geodesic distance matrix
 2. Global approach
- LLE
 1. Model local neighborhoods as linear patches and then embed in a lower dimensional manifold.
 2. Local approach
 3. Computationally efficient. Eigenvectors from sparse matrices

Graph Kernels

- We Considered Two Variations for Weighting Edges for the Laplacian Eigenmap Algorithm

- Heat Kernel

$$\mathbf{W}_{ij} = e^{\frac{-\|x_i - x_j\|^2}{\tau}} \text{ if } i \text{ and } j \text{ connected, } 0 \text{ otherwise}$$

- Simple-Minded

$$\mathbf{W}_{ij} = 1 \text{ if } i \text{ and } j \text{ connected, } 0 \text{ otherwise}$$

- Overview of Additional Graph Kernels

- Diffusion Kernel

$$\mathbf{K}_{ij} = \left[\sum_k \frac{\lambda^k}{k!} \mathbf{A}^k \right]_{ij} = [\min(\lambda \mathbf{A})]_{ij} \text{ if } i \text{ and } j \text{ connected, } 0 \text{ otherwise}$$

\mathbf{A} is the adjacency matrix. Discounts a k -length walk by $\frac{\lambda^k}{k!}$ for $0 \leq \lambda \leq 1$

- Resistance Kernel

(also called the commute-time kernel)

$$\mathbf{K} = (\mathbf{D} - \mathbf{A})^+$$

\mathbf{A} is the adjacency matrix, and \mathbf{D} is the degree matrix,

and $+$ is the Moore-Penrose pseudoinverse operator.

Interprets a graph as a network of resistance, where edge weights are interpreted relative to resistance values in an electrical network.

Summary

- Isomap, LLE and Laplacian Eigenmap: Non-linear dimensionality reduction technique
- Useful for learning manifolds, understanding low dimensional data embedded in high dimensional space.
- PCA and MDS fails for this type of data.
- All three use some technique to preserve local geometry i.e. inter-point relationships

References and Acknowledgments

- Belkin, Mikhail, and Partha Niyogi. 2003. "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation." *Neural Computation* 15 (6): 1373–96. doi:10.1162/089976603321780317.
- Roweis, S. T. & Saul, L. K. (2000), 'Locally linear embedding', *Science* 290, 2323–2326.
- Tenenbaum, J. B., de Silva, V. & Langford, J. C. (2000), 'A global geometric framework for nonlinear dimensionality reduction', *Science* 290, 2319–2323.
- <http://www.cs.nyu.edu/~roweis/lle/>
- <http://isomap.stanford.edu/>
- http://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction
- http://web.mit.edu/6.454/www/www_fall_2003/ihler/slides.pdf
- <http://cseweb.ucsd.edu/~saul/teaching/cse291s07/laplacian.pdf>
- www.public.asu.edu/~jye02/CLASSES/Spring.../Lec19-Isomap.ppt
- www.public.asu.edu/~jye02/CLASSES/Spring.../Lec20-LLE.ppt
- www.public.asu.edu/~jye02/CLASSES/.../Lec21-LaplacianEig.ppt
- <http://www.stat.purdue.edu/~vishy/talks/Graphs.pdf>
- Kunegis, J., Lommatzsch, A., Albayrak, S., & Bauckhage, C. (2008). On the scalability of graph kernels applied to collaborative recommenders. In *Workshop on Recommender Systems* (p. 35).

Additional Laplacian Eigenmaps Details

Continuous Manifold

- Laplacian of a graph is analogous to the Laplace Beltrami operator on manifolds.
- Mapping to 1-D. Find a map f such that points close together on the manifold get mapped close together on the line.
- Two points \mathbf{z} and \mathbf{x} mapped to $f(\mathbf{z})$ and $f(\mathbf{x})$. It is shown that

$$|f(\mathbf{z}) - f(\mathbf{x})| \leq \|\nabla f(\mathbf{x})\| \|\mathbf{z} - \mathbf{x}\| + o(\|\mathbf{z} - \mathbf{x}\|)$$

Additional Laplacian Eigenmaps Details

Continuous Manifold

- Gradient of f provides us with an estimate of how far apart f maps nearby points.
- Minimizing the gradient minimizes the values assigned to close by points.

$$\operatorname{argmin}_{\|f\|_{L^2(\mathcal{M})}=1} \int_{\mathcal{M}} \|\nabla f(x)\|^2$$

- Minimizing the objective function reduces to finding eigenfunctions of the Laplace Beltrami Operator