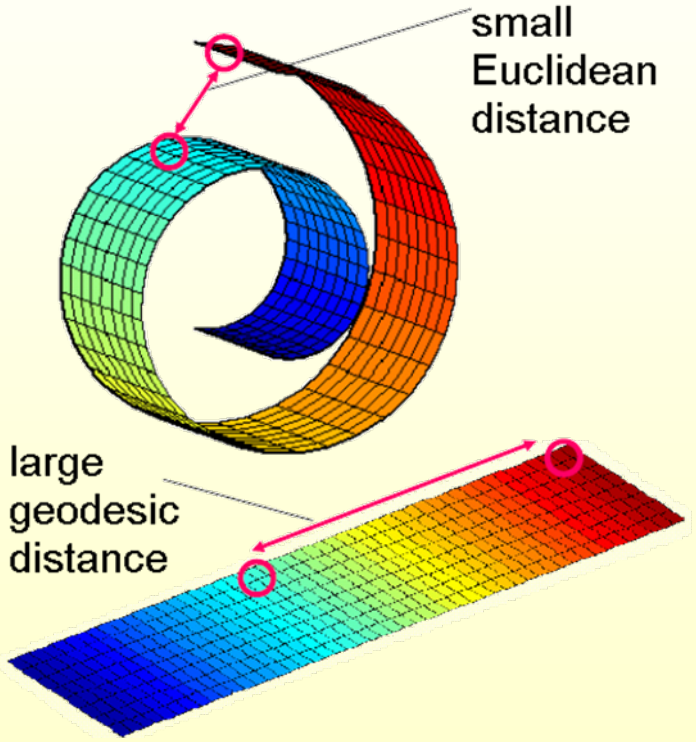
CS233: Geometric and Topological Data Analysis

Multi-Dimensional Scaling, Non-Linear Dimensionality Reduction

18 April 2018

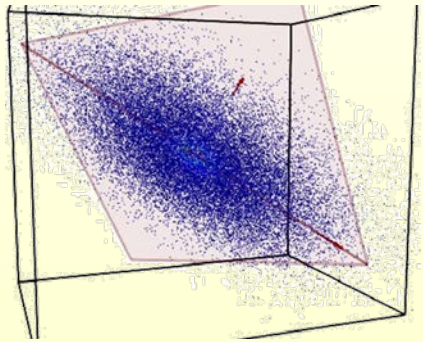


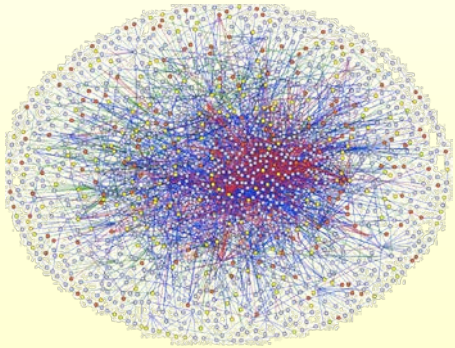
Slides ack: Michael Friendly, Olga Sorkine, Sungkyu Jung, Young Baik, Kevin Zhao

Last Time: Spectral Graph Theory

Connect Linear Space and Graph Views of Data

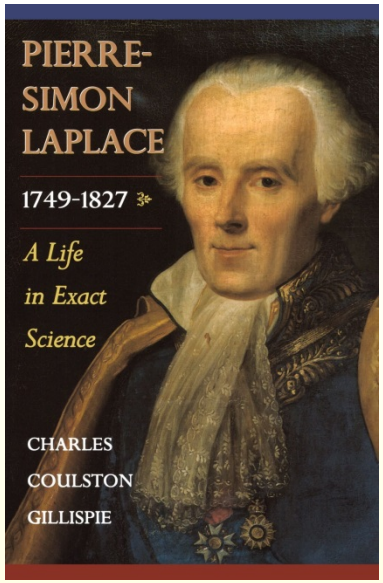
- Points in R^d
  - via near-neighbor graphs
- Graph
  - via matrix representations of graphs





What to do if the dimension is too high?

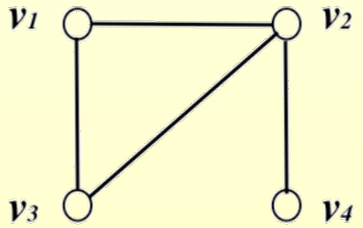
- Embed the data into a lower-d space
- But how to deal with non-linear low-d structure?
- “Unfold” non-linear structure in the intrinsic dimension



The Laplacian

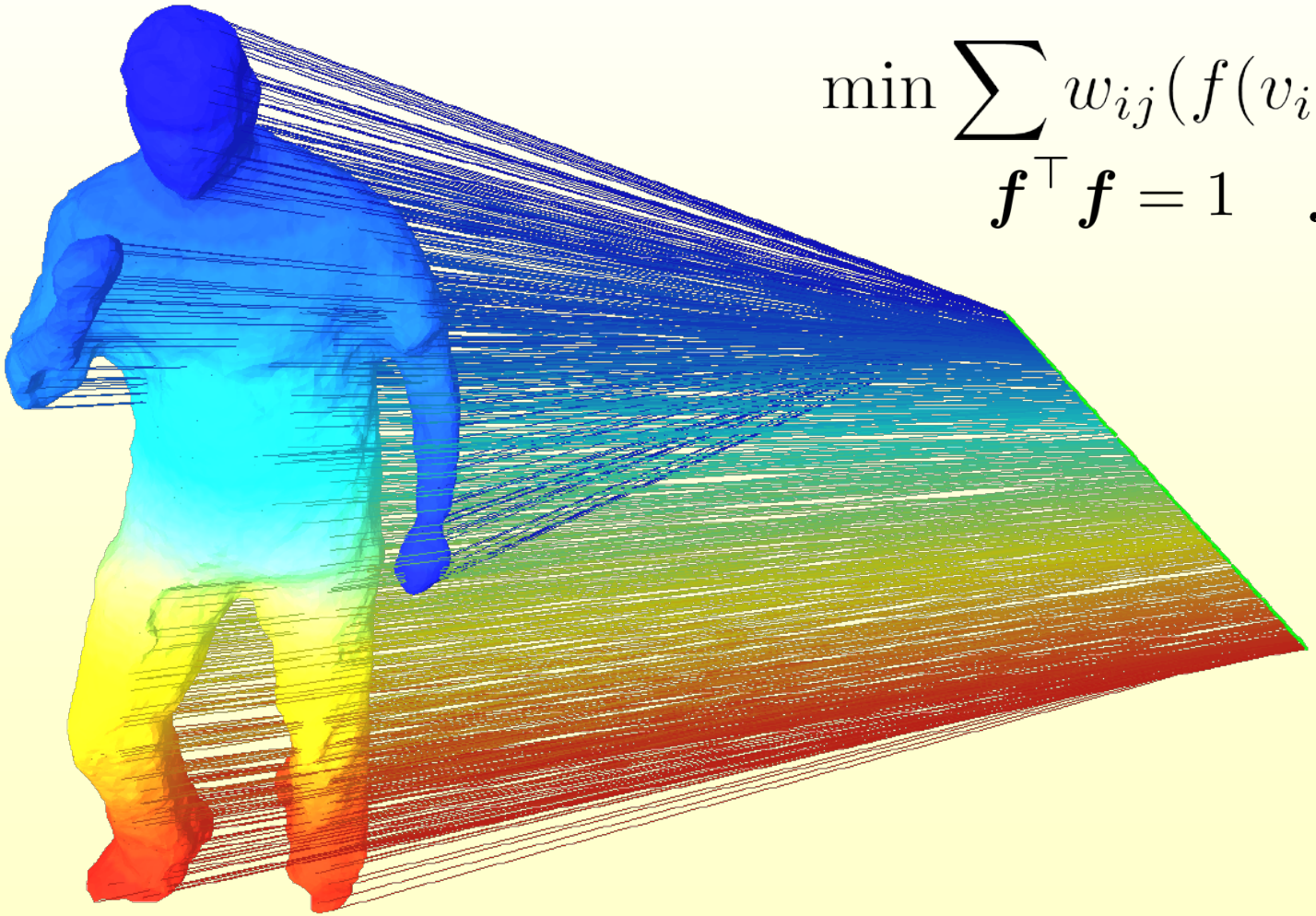
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2}$$

$$\mathbf{L} = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$



$$\boldsymbol{L} = \nabla^\top \nabla$$

1-d Embedding via the Fiedler Vector



$$\min \sum w_{ij}(f(v_i) - f(v_j))^2$$
$$\boldsymbol{f}^\top \boldsymbol{f} = 1 \quad \boldsymbol{f} \perp \mathbf{1}$$

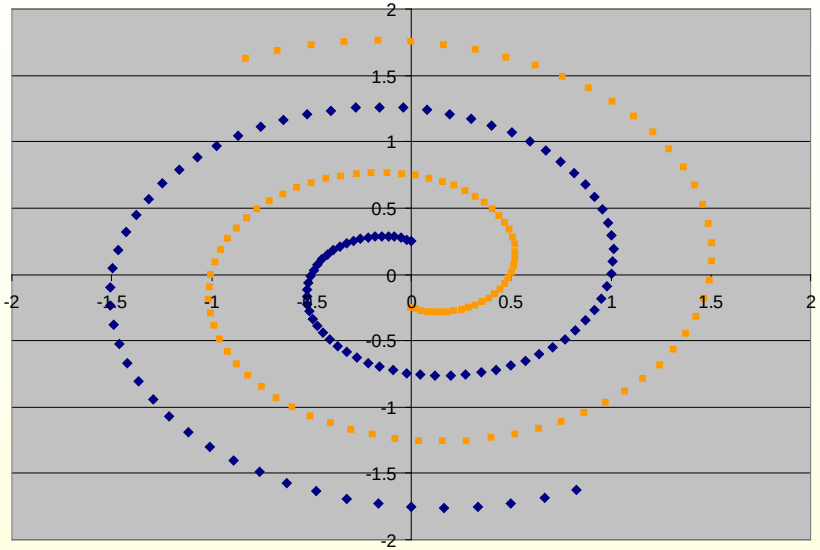
Higher-d Embeddings

- Embed the graph in a k-dimensional Euclidean space. The embedding is given by the $n \times k$ matrix $\mathbf{F} = [\boldsymbol{f}_1 \boldsymbol{f}_2 \ldots \boldsymbol{f}_k]$ where the i-th row of this matrix – $\boldsymbol{f}^{(i)}$ – corresponds to the Euclidean coordinates of the i-th graph node v_i.
- We need to minimize (Belkin & Niyogi '03):

$$\arg \min_{\boldsymbol{f}_1 \cdots \boldsymbol{f}_k} \sum_{i,j=1}^{n} w_{ij} \|\boldsymbol{f}^{(i)} - \boldsymbol{f}^{(j)}\|^2 \text{ with: } \mathbf{F}^\top \mathbf{F} = \mathbf{I}.$$

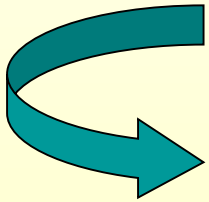
- The solution is provided by the matrix of eigenvectors corresponding to the k lowest nonzero eigenvalues of the eigenvalue problem $\mathbf{L}\boldsymbol{f} = \lambda \boldsymbol{f}$.

Spirals Again

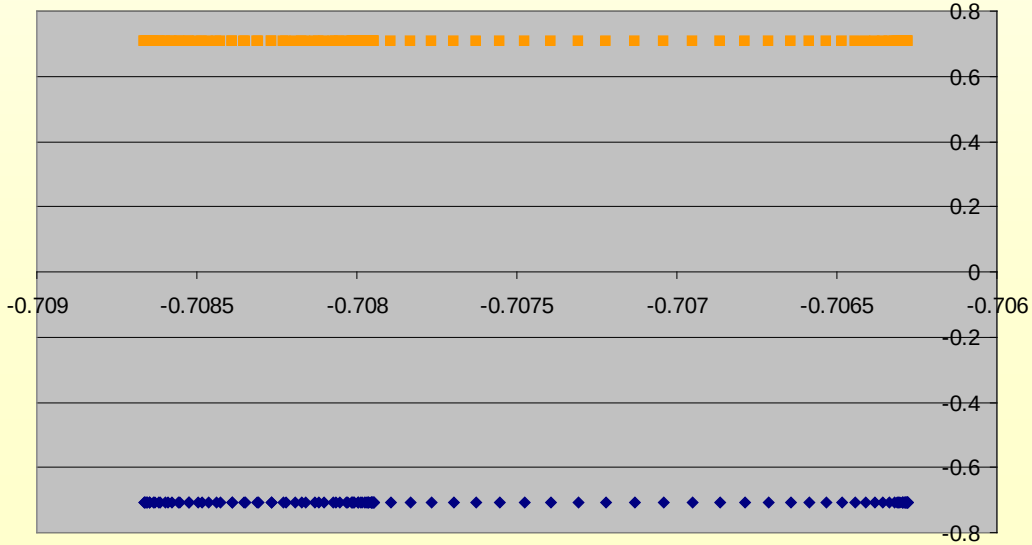


Dataset exhibits complex cluster shapes

$\Rightarrow$ Direct *k*-means performs very poorly in this space due to bias toward dense spherical clusters.



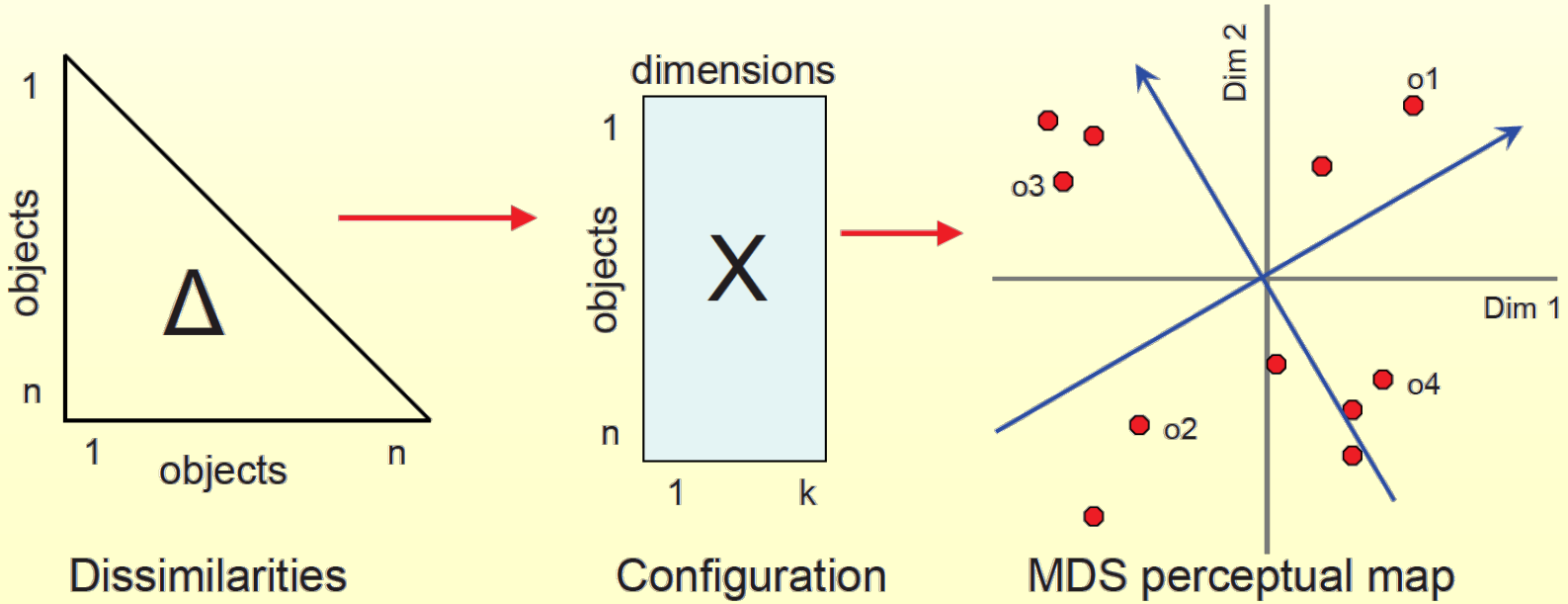
In the embedded space given by two leading eigenvectors, clusters are trivial to separate.



Multidimensional Scaling

Multidimensional Scaling (MDS)

- A “distance preserving” embedding of the data into a Euclidean space
  - Sometimes distances are observed directly (e.g., similarity ratings)
  - Sometimes they can be calculated from a data table (e.g., Euclidean distances, correlations)



Formally (Metric MDS) …

- Given a (symmetric) matrix of pairwise “dis-similarities” between n objects / data sets

$$M = \left(\delta_{ij} \right)_{n \times n}$$

No need to satisfy the triangle inequality

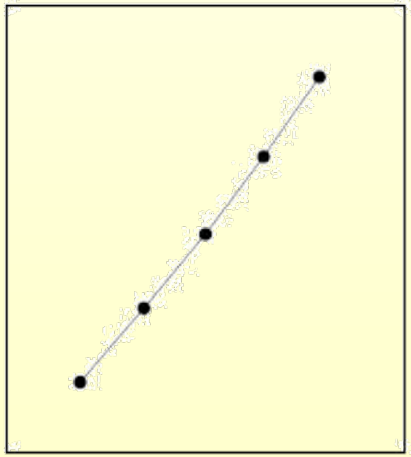
- Find n points in low-dimensional space R^d, so that their distance matrix is as close as possible to M

- Low d (=2,3) allows us to visualize the data directly

Distances and Dimensionality

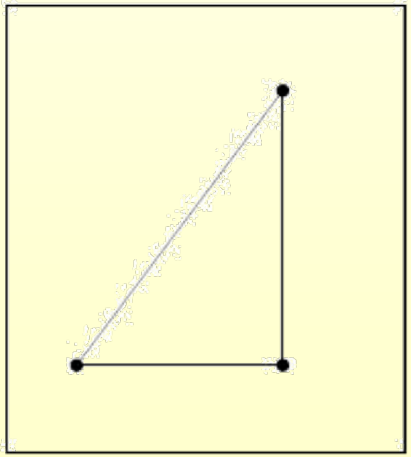
- How do distances/dissimilarities determine dimensionality?

$$D = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 2 & 3 \\ 2 & 1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 1 \\ 4 & 3 & 2 & 1 & 0 \end{bmatrix}$$



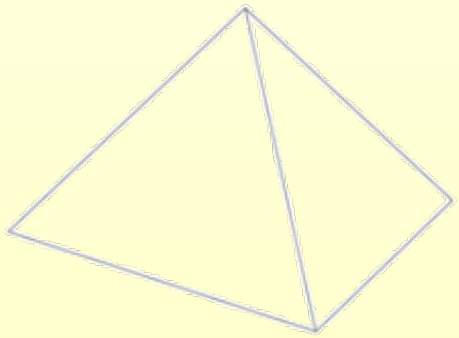
k=1

$$D = \begin{bmatrix} 0 & 3 & 4 \\ 3 & 0 & 5 \\ 4 & 5 & 0 \end{bmatrix}$$



k=2

$$D = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$



k=3

MDS: Motivating Example

- Travel times by train between French cities

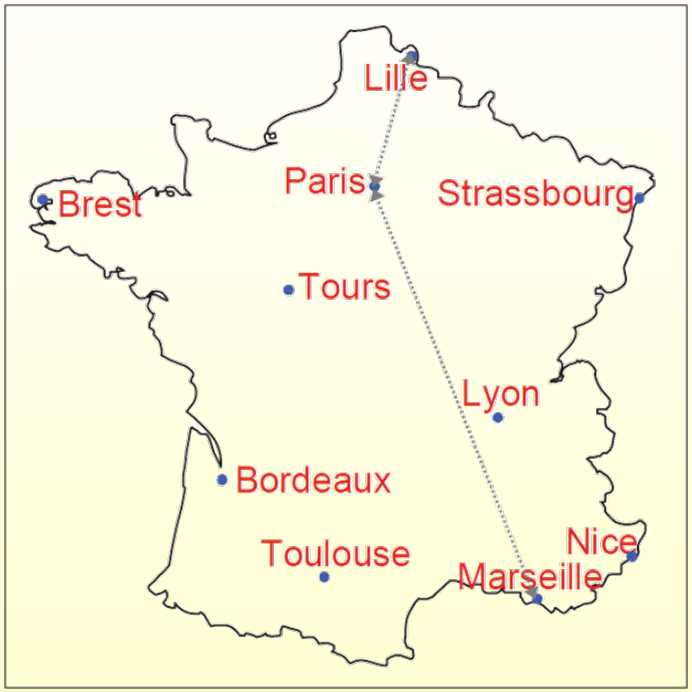
	Bor-deaux	Brest	Lille	Lyon	Mar-seille	Nice	Paris	Strassb ourg	Tou-louse	Tours
Bordeaux	0									
Brest	9:58	0								
Lille	6:39	7:11	0							
Lyon	8:05	7:11	4:52	0						
Marseille	5:47	8:49	6:12	1:35	0					
Nice	8:30	13:36	8:20	4:33	2:26	0				
Paris	2:59	4:17	1:04	2:01	3:00	5:52	0			
Strassbourg	8:08	10:16	6:54	4:36	7:04	11:15	4:01	0		
Toulouse	2:02	13:52	9:42	4:25	3:26	6:29	5:14	10:56	0	
Tours	2:36	5:38	4:17	4:21	5:13	9:04	1:13	6:03	6:06	0

- Considerations:
  - The recovered configuration (**X** = map) should be 2D
  - NSEW not relevant to distances– may have to rotate
  - Travel time not necessarily ~ map distance (TGV)
  - May need to consider other relations between dissimilarity and distance in MD space

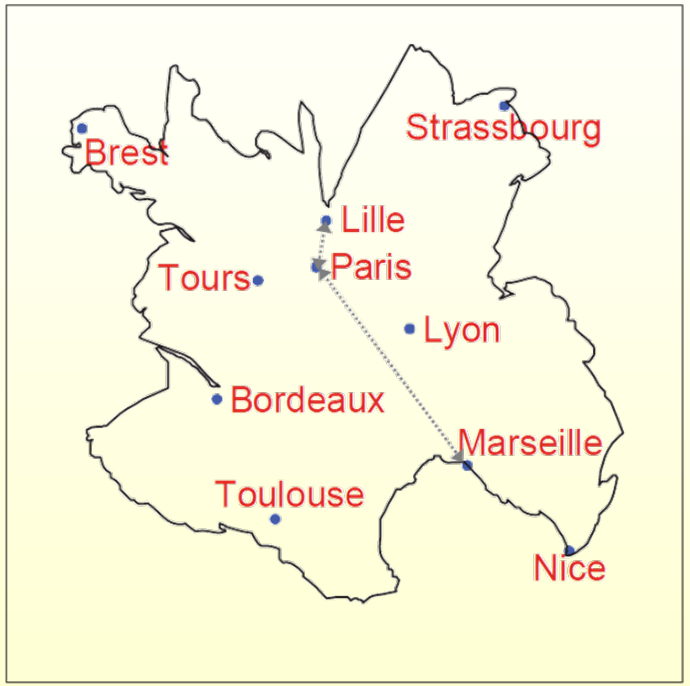
Source: Patrick Groenen, "Past, Present, and Future of Multidimensional Scaling", CARME, 2011

MDS: Motivating Example

- Travel times by train between French cities



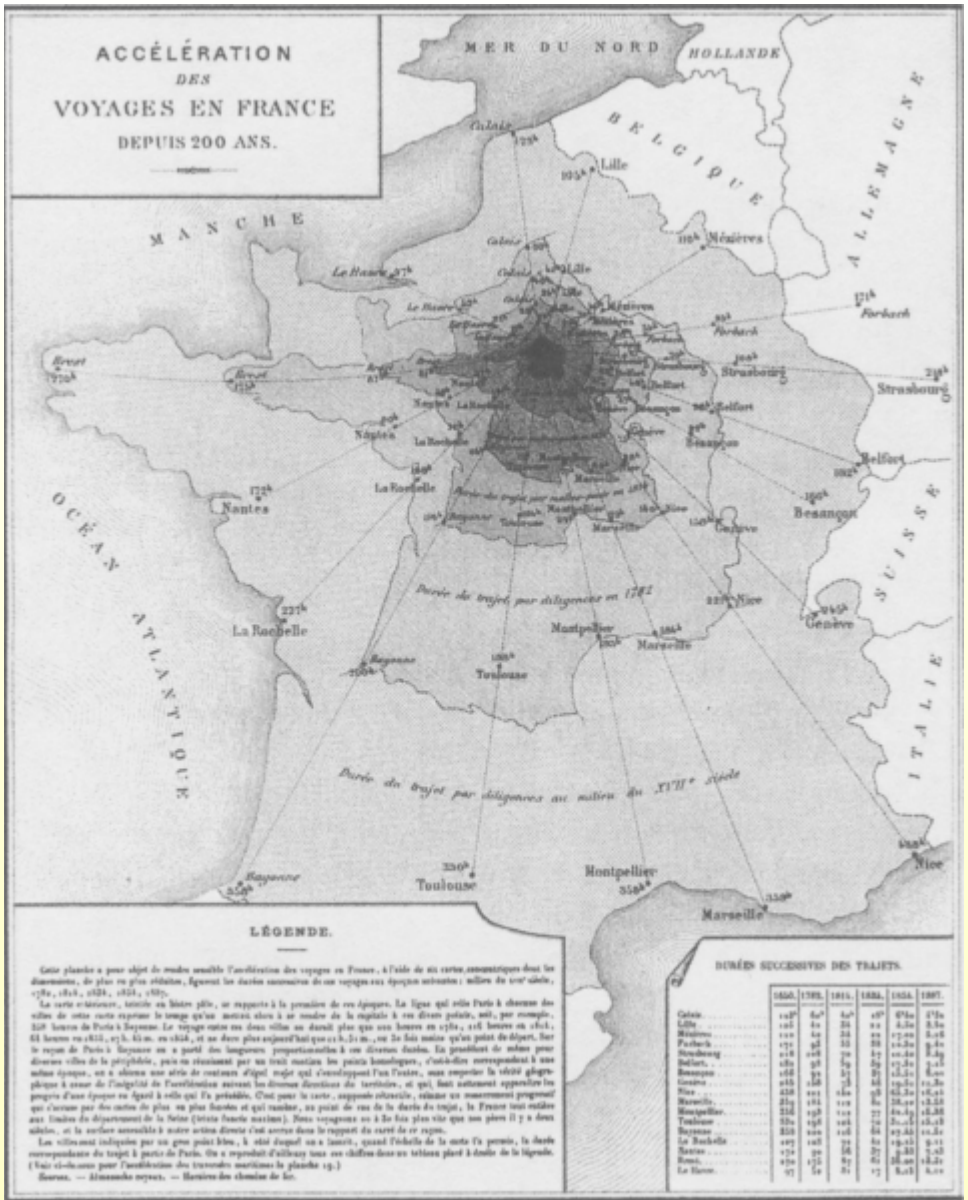
Actual map of France



MDS recovered map

Marseille, Lille, Lyon, ... closer to Paris in travel time (TGV)

MDS: Motivating Example



- Travel time only partially related to map distance

Emile Cheysson (1888) – anaphoric map, showing decrease in time to travel from Paris over 200 years

- Other considerations relating distance and dissimilarity

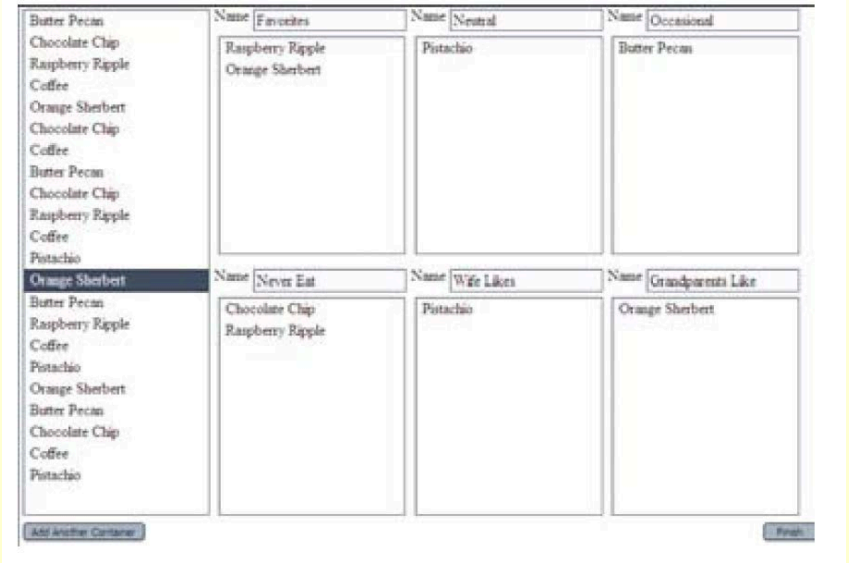
MDS Has Many Uses

- Psychology (perception, cognition)
- Political science (voting behavior, court decisions)
- Sociology (social network analysis)
- Archeology (artifact similarity)
- Biology/Chemistry (molecular structure, species analysis)
- Document retrieval & classification
- Graph layout
- Pattern recognition
- Dimension reduction
- …

Obtaining Dissimilarity Data

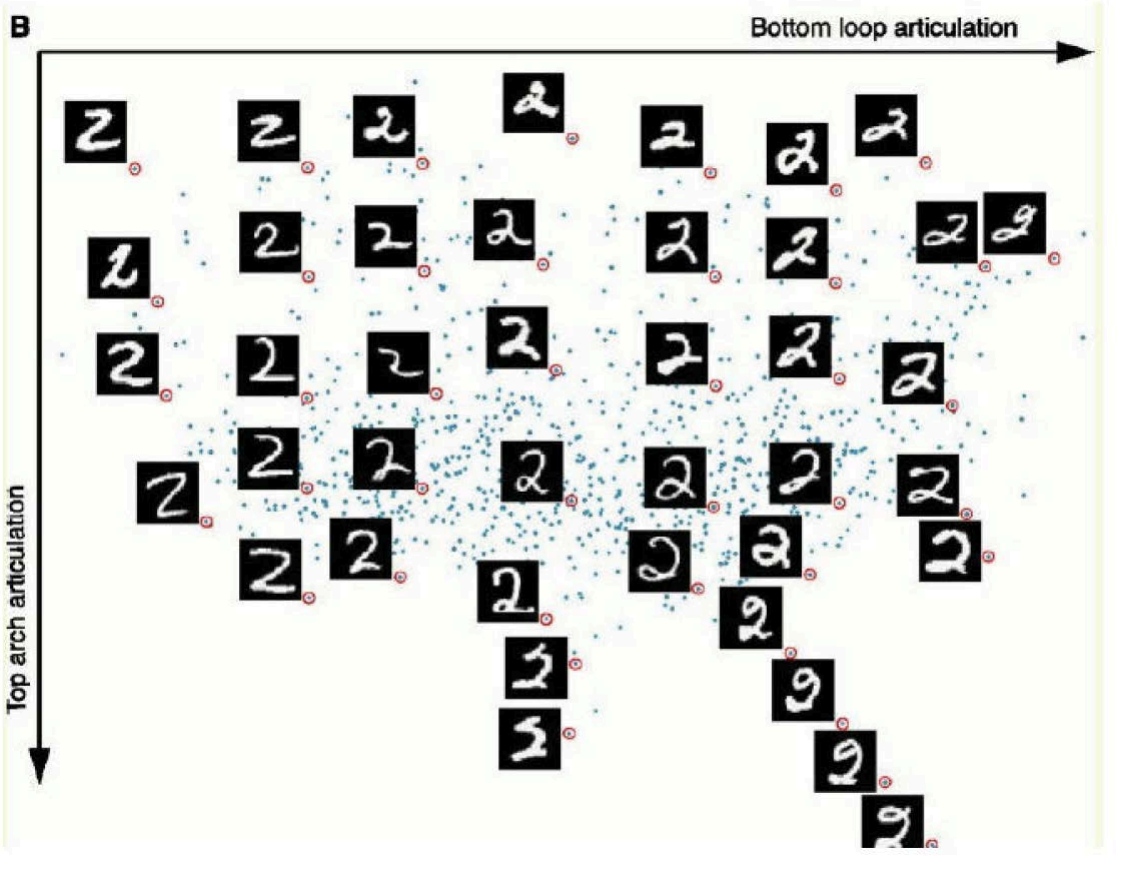
- Direct ratings (flavor comparisons)
- Confusion data (Morse dots and dashes)
- Co-occurrence data (Amazon recommendations)
- Sorting into groups
- Memory recall

Sometimes distances are naturally defined – but sometimes we seek subjective dissimilarities



Can every distance matrix be realized in a Euclidean space?

Example: Pattern Recognition



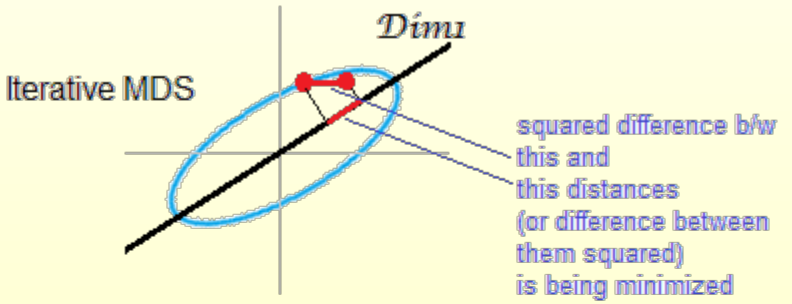
MDS of judged similarity of handwritten “2”s

Goal: determine features important in pattern recognition

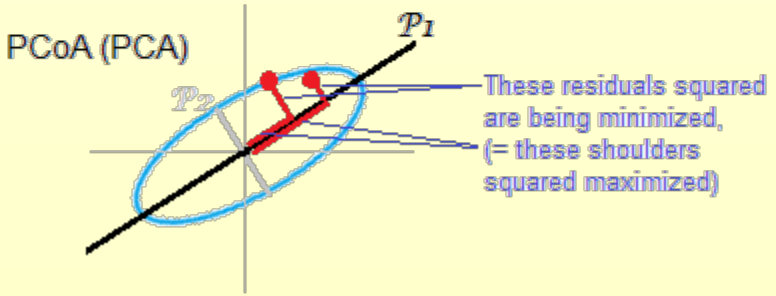
Classic Metric MDS

- Sometimes we can model our data as points in a high-dimensional Euclidean space – and we are looking for an embedding to a lower-dimensional space that preserves (absolute or relative) distances (in the high-d space) as much as possible.

- In this case the problem has a clean geometric solution.



Is this the same as PCA?



Classic Metric MDS

- To go from dimension D down to dimension d
- Given data $X \in R^{D \times n}$

$$X = \begin{pmatrix} | & | & & | \\ \mathbf{x_1} & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ | & | & & | \end{pmatrix} \quad \text{and} \quad M = \left(\text{dist}^2(\mathbf{x_i}, \mathbf{x_j}) \right)_{n \times n}$$

- We look for X',

$$X' = \begin{pmatrix} | & & | \\ \mathbf{x_1}' & \cdots & \mathbf{x_n}' \\ | & & | \end{pmatrix} \in R^{d \times n}$$

- We can assume the $\mathbf{x}_i$' are centered

Classic Metric MDS

- So that we minimize $\| M' - M \|$ (related to the *stress* of the system)

- where $M' = \left(\text{dist}^2(\mathbf{x_i}', \mathbf{x_j}') \right) = \left(\left\| \mathbf{x_i}' - \mathbf{x_j}' \right\|^2 \right) \in R^{n \times n}$

- M' is the Euclidean distances matrix for points $\mathbf{x_i}'$.

$$\min \| M' - M \|$$

The Math Details

- Ideally we want $M' = \left(\left\| \mathbf{x}_i' - \mathbf{x}_j' \right\|^2 \right) = M$

$$\left(< \mathbf{x}_i' - \mathbf{x}_j', \ \mathbf{x}_i' - \mathbf{x}_j' > \right) = M$$

$$\left(\| \mathbf{x}_i' \|^2 + \| \mathbf{x}_j' \|^2 - 2 < \mathbf{x}_i', \mathbf{x}_j' > \right) = M$$

$$\begin{pmatrix} \| \mathbf{x}_1' \| & \| \mathbf{x}_1' \| & \cdots & \| \mathbf{x}_1' \| \\ \| \mathbf{x}_2' \| & \| \mathbf{x}_2' \| & \cdots & \| \mathbf{x}_2' \| \\ & & \vdots & \\ \| \mathbf{x}_n' \| & \| \mathbf{x}_n' \| & \cdots & \| \mathbf{x}_n' \| \end{pmatrix} \quad \begin{pmatrix} \| \mathbf{x}_1' \| & \| \mathbf{x}_2' \| & & \| \mathbf{x}_n' \| \\ \| \mathbf{x}_1' \| & \| \mathbf{x}_2' \| & & \| \mathbf{x}_n' \| \\ \vdots & \vdots & \cdots & \vdots \\ \| \mathbf{x}_1' \| & \| \mathbf{x}_2' \| & & \| \mathbf{x}_n' \| \end{pmatrix}$$

want to get rid of these

$$\begin{pmatrix} — & \mathbf{x}_1' & — \\ & \vdots & \\ — & \mathbf{x}_n' & — \end{pmatrix} \begin{pmatrix} | & & | \\ \mathbf{x}_1' & \cdots & \mathbf{x}_n' \\ | & & | \end{pmatrix}$$

$X'^T \qquad X'$

The Magic Matrix *J*

$$J = \begin{pmatrix} \frac{n-1}{n} & -\frac{1}{n} & \cdots & -\frac{1}{n} \\ -\frac{1}{n} & \frac{n-1}{n} & -\frac{1}{n} & -\frac{1}{n} \\ \vdots & & \ddots & \vdots \\ -\frac{1}{n} & \cdots & -\frac{1}{n} & \frac{n-1}{n} \end{pmatrix}_{n\times n} = I - \frac{1}{n}\begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & 1 & 1 \\ \vdots & & \ddots & \vdots \\ 1 & \cdots & 1 & 1 \end{pmatrix} = I - \frac{1}{n}K$$

$$\begin{pmatrix} a & a & \cdots & a \end{pmatrix} \cdot J = 0$$

$$J \cdot \begin{pmatrix} b \\ b \\ \vdots \\ b \end{pmatrix} = 0$$

So We Get to The Gram Matrix

Cleaning the system:

$$\times J \Big/ \begin{pmatrix} \|\mathbf{x}_1'\| & \|\mathbf{x}_1'\| & \dots & \|\mathbf{x}_1'\| \\ \|\mathbf{x}_2'\| & \|\mathbf{x}_2'\| & \dots & \|\mathbf{x}_2'\| \\ & & \vdots & \\ \|\mathbf{x}_n'\| & \|\mathbf{x}_n'\| & \dots & \|\mathbf{x}_n'\| \end{pmatrix} + \begin{pmatrix} \|\mathbf{x}_1'\| & \|\mathbf{x}_2'\| & & \|\mathbf{x}_n'\| \\ \|\mathbf{x}_1'\| & \|\mathbf{x}_2'\| & & \|\mathbf{x}_n'\| \\ \vdots & \vdots & \dots & \vdots \\ \|\mathbf{x}_1'\| & \|\mathbf{x}_2'\| & & \|\mathbf{x}_n'\| \end{pmatrix} - 2X'^T X' = M \Big/ \times J$$

$$J = I - \frac{1}{n} K$$

$$-2X'^T X' = JMJ$$

$$X'^T X' = -\tfrac{1}{2} JMJ =: B$$

Note that $X'K = KX'^T = 0$, as X' is centered

$$X'^T X' = B$$

So from the distance matrix we can get the Gram (inner product) matrix.

And Finally Use the Spectral Hammer

We will use the spectral decomposition of B:

$$X'^T X' = B = \begin{pmatrix} | & & | \\ \mathbf{v_1} & \cdots & \mathbf{v_n} \\ | & & | \end{pmatrix} \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} \begin{pmatrix} | & & | \\ \mathbf{v_1} & \cdots & \mathbf{v_n} \\ | & & | \end{pmatrix}^T$$

$$X'^T X' = \begin{pmatrix} | & & | & \vdots & | \\ | & & | & & | \\ \mathbf{v_1} & \cdots & \mathbf{v_d} & \vdots & \mathbf{v_n} \\ | & & | & & | \\ | & & | & \vdots & | \end{pmatrix} \begin{pmatrix} \sqrt{\lambda_1} & & & & \\ & \ddots & & & \\ & & \sqrt{\lambda_d} & & \\ & & & \ddots & \\ & & & & \sqrt{\lambda_n} \end{pmatrix} \begin{pmatrix} \sqrt{\lambda_1} & & & & \\ & \ddots & & & \\ & & \sqrt{\lambda_d} & & \\ & & & \ddots & \\ & & & & \sqrt{\lambda_n} \end{pmatrix}^T \begin{pmatrix} | & & | & \vdots & | \\ | & & | & & | \\ \mathbf{v_1} & \cdots & \mathbf{v_d} & \vdots & \mathbf{v_n} \\ | & & | & & | \\ | & & | & \vdots & | \end{pmatrix}^T$$

$d \times d$

$n \times d$

X'^T $\qquad$ X'

So We Get the *X'*

So we find X' by throwing away the last $n-d$ eigenvalues

$$X' = \begin{pmatrix} \sqrt{\lambda_1}\,\mathbf{v_1} \\ \cdots \quad \cdots \quad \cdots \\ \sqrt{\lambda_d}\,\mathbf{v_d} \end{pmatrix}_{d \times n}$$

For this X' : $$X' = \arg\min_{X'} \left\| X'^T X' - B \right\|_{L^2}$$

$$\|A\|_{L^2} = \sqrt{\sum_{i,j} A_{ij}^2}$$

This choice minimizes the inner product (and distance) loss

More General Metric MDS

- In general, we minimize directly the square loss on distances

$$\text{stress} = \mathcal{L}(\hat{d}_{ij}) = \left(\sum_{i<j} (\hat{d}_{ij} - f(d_{ij}))^2 / \sum d_{ij}^2 \right)^{\frac{1}{2}}$$

f monotonic

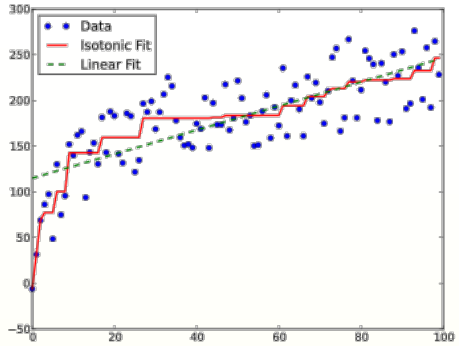
- Sammon mapping

$$\text{Sammon's stress}(\hat{d}_{ij}) = \frac{1}{\sum_{\ell<k} d_{\ell k}} \sum_{i<j} \frac{(\hat{d}_{ij} - d_{ij})^2}{d_{ij}}$$

  - This weighting system normalizes the squared-errors in pairwise distances by using the distance in the original space. As a result, Sammon mapping preserves the small d_{ij}, giving them a greater degree of importance in the fitting procedure than for larger values of d_{ij}

Generally solved by gradient descent

Non-Metric MDS



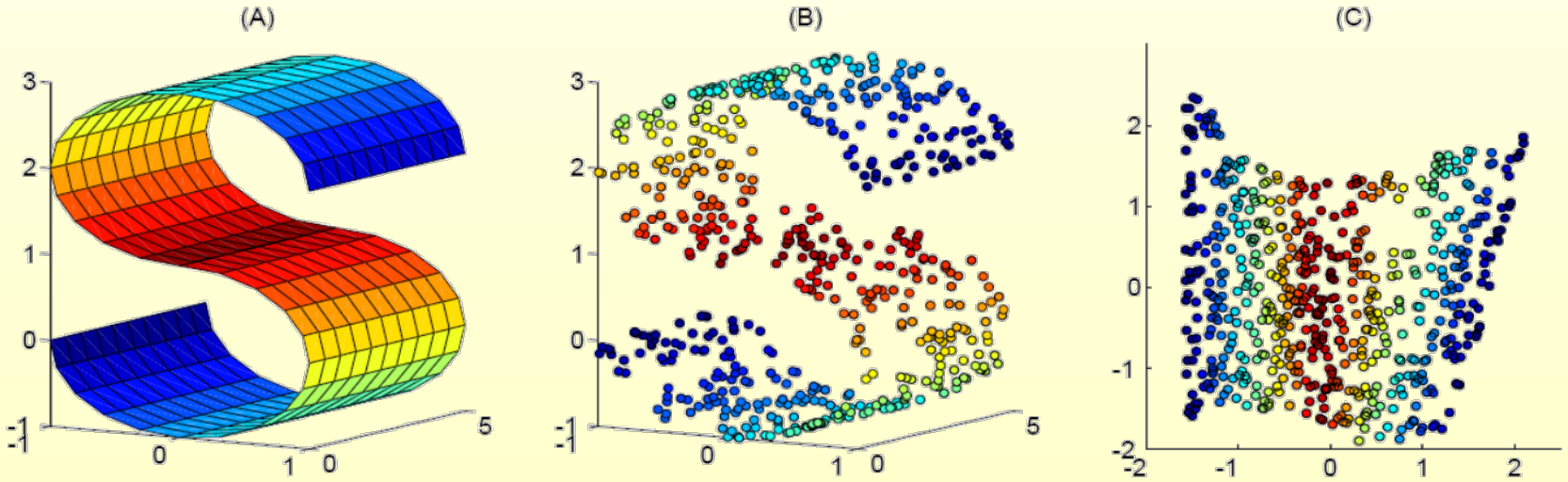
- Sometimes all we can say is that (dis)similarity is *ordinally* related to distance in MD space (only ordering of distances matters, not the actual values)
- If we only have ordering information, we can use *monotone (isotonic) regression* to find "disparities" that are compatible with the ordering constraints of the dissimilarities (not unique)
- We can then use alternating least squares (ALS) by repeating
  - a monotone regression step, followed by
  - a metric stress reduction step

$$Stress(\hat{\mathbf{D}}, \mathbf{X}) = \left[\frac{\sum_{i<j} \left(\hat{d}_{ij} - d_{ij}(\mathbf{X}) \right)^2}{\sum_{i<j} d_{ij}^2(\mathbf{X})} \right]^{1/2}$$

Non-Linear Dimensionality Reduction

Non-Linear Dimensionality Reduction

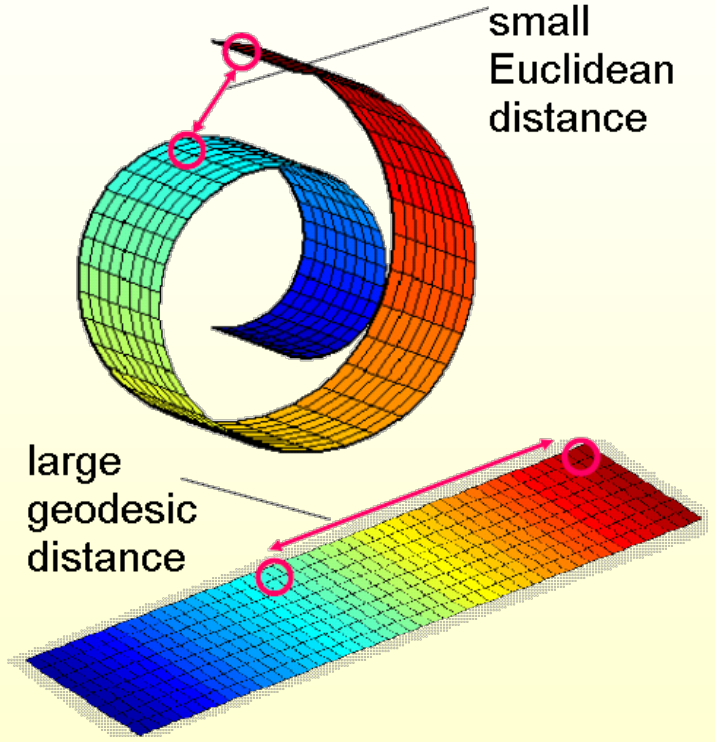
- Many data sets contain essential nonlinear structures that “invisible” to PCA and MDS
- Must resort to some non-linear dimensionality reduction approaches



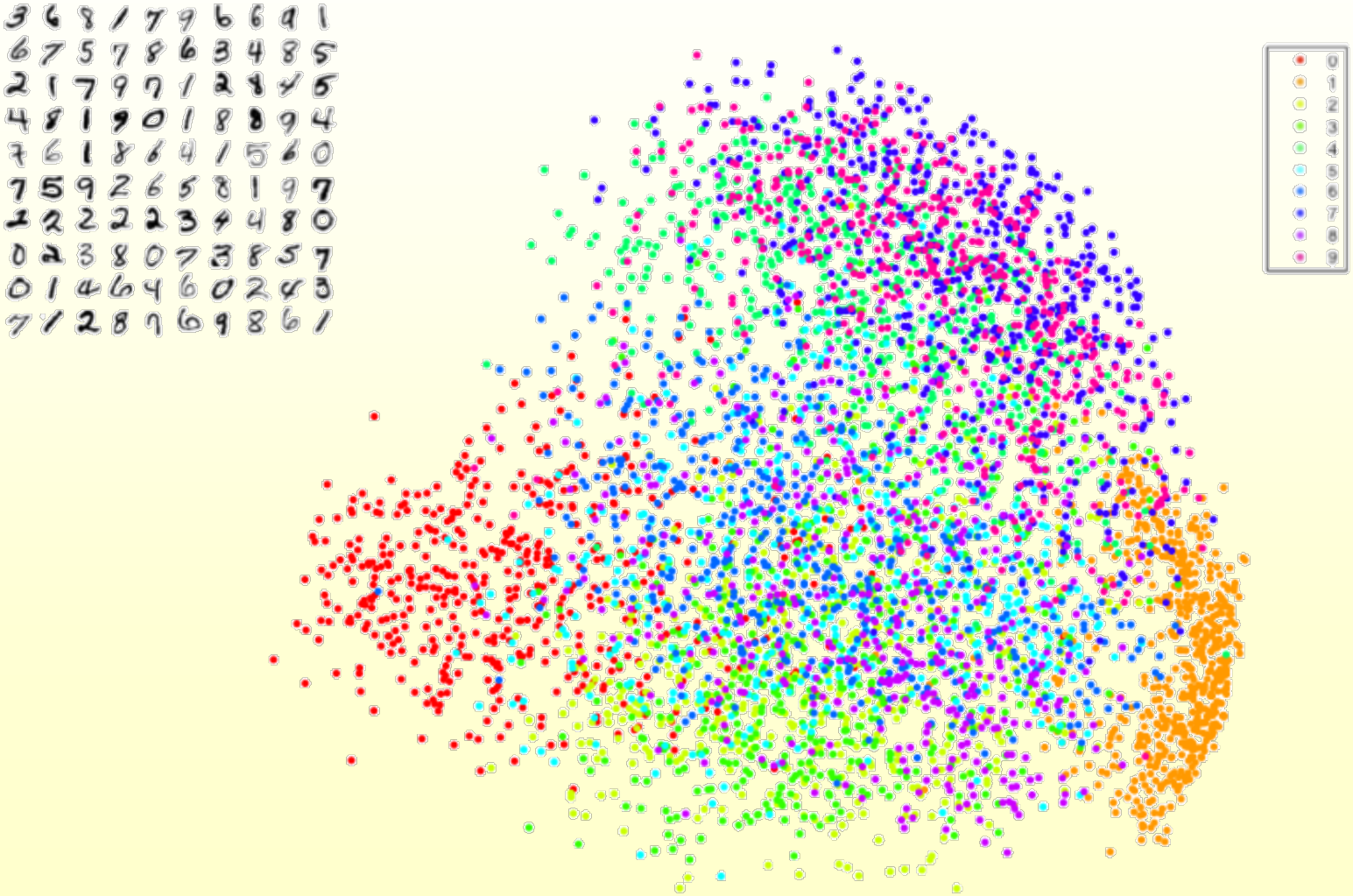
Data sampled from a non-linear manifold

The Choice of Distance

- We can try to capture the manifold structure through the right notion of distance directly on the manifold (geodesic distance)

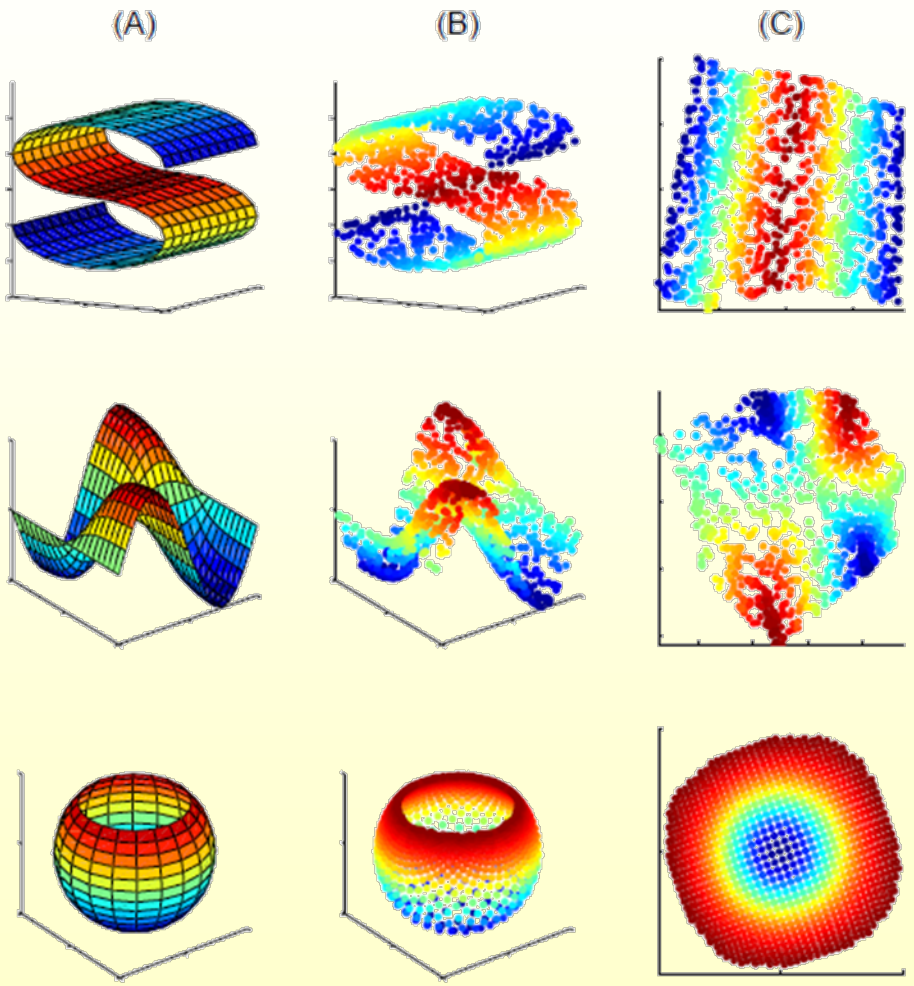


PCA Cares About Large Distances



The Challenge of NLDR

- An unsupervised learning algorithm – it must discover the global internal coordinates of the manifold, without external signals that suggest how the data should be embedded in low dimensions

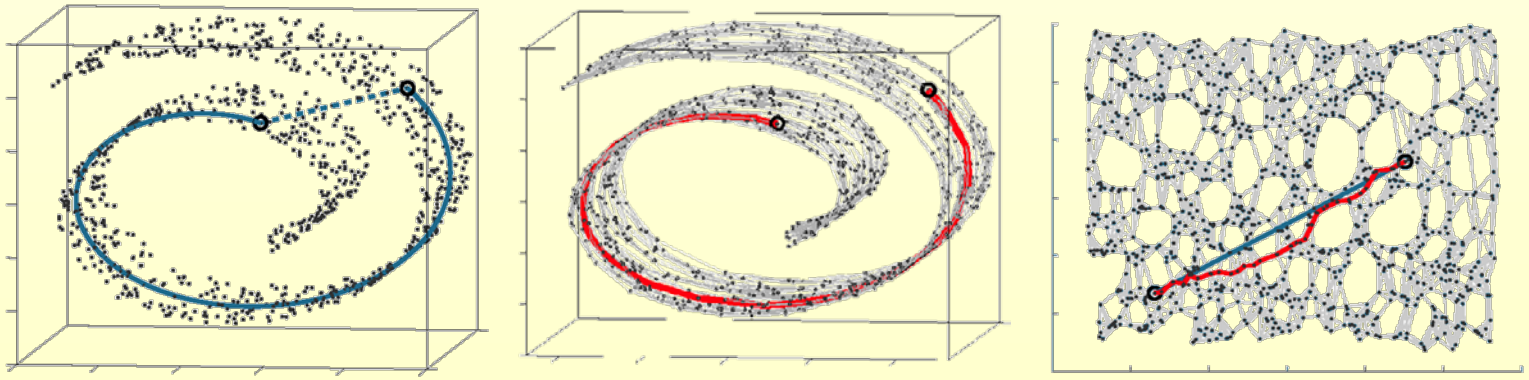


Isomap

ISOMAP

(J. B. Tenenbaum, V. de Silva and J. C. Langford)

- Example of non-linear structure (Swiss roll)
  - Only the geodesic distances reflect the true low-dimensional geometry of the manifold
- ISOMAP (Isometric Feature Mapping)
  - Preserves the intrinsic geometry of the data
  - Uses the geodesic manifold distances between all pairs

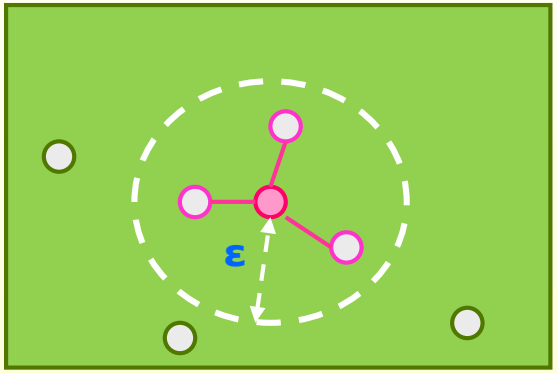


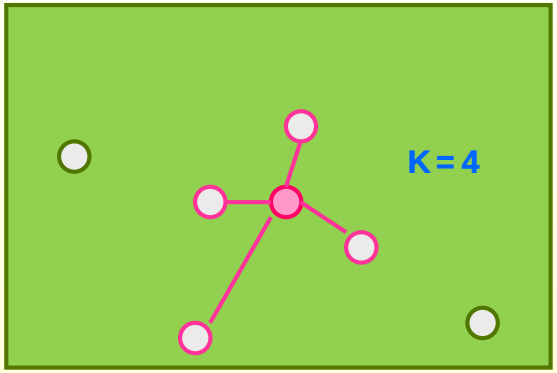
ISOMAP (Algorithm Description)

- **Step 1**
  - Form a near-neighbor graph G on the original data points, weighing the edges based on their original distances $d_X(i, j)$
- **Step 2**
  - Estimate the geodesic distances $d_G(i, j)$ between all pairs of points on the sampled manifold by computing their shortest path distances in the graph G.
- **Step 3**
  - Construct an embedding of the data in *d*-dimensional Euclidean space *Y* that best preserves the distances (MDS).

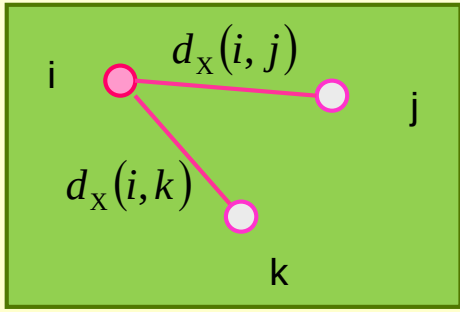
Near Neighbor Graph

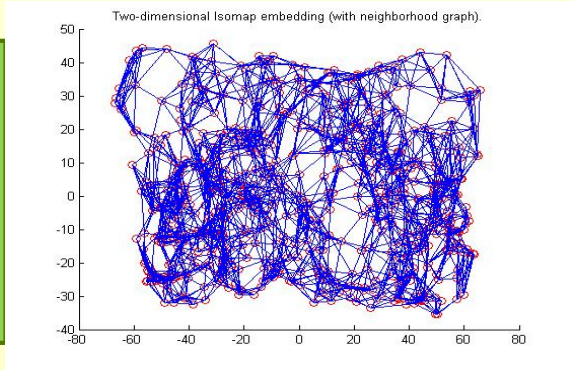
- **Step 1**
  - Determining neighboring points within a fixed radius based on the input space distance $d_X(i, j)$, or use a fixed # of neighbors





  - These neighborhood relations are represented as a weighted graph G over the data points.



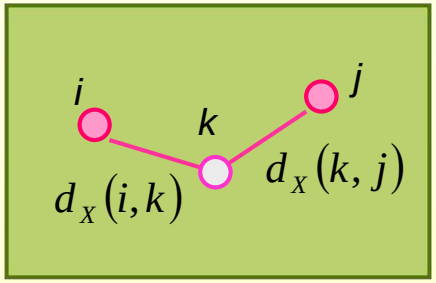


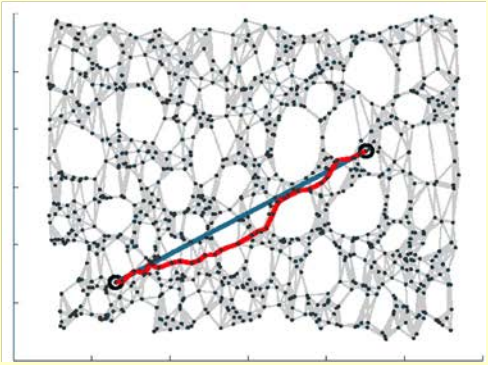
Shortest Path Computation

- **Step 2**
  - Estimating the geodesic distances $d_G(i,j)$ between all pairs of points on the manifold by computing their shortest path distances in the graph G.
  - Can be done using classic graph algorithms for APSPs: Floyd/Warshall's algorithm or Dijkstra's algorithm

$$d_G(i,j) = d_X(i,j) \quad \text{neighborin g i, j}$$
$$d_G(i,j) = \infty \qquad \text{othewise}$$

$$for\ \mathrm{k} = 1,2,\ldots,\mathrm{N}$$
$$d_G(i,j) = \min\{\, d_X(i,j),\ d_X(i,k) + d_X(k,j)\}$$

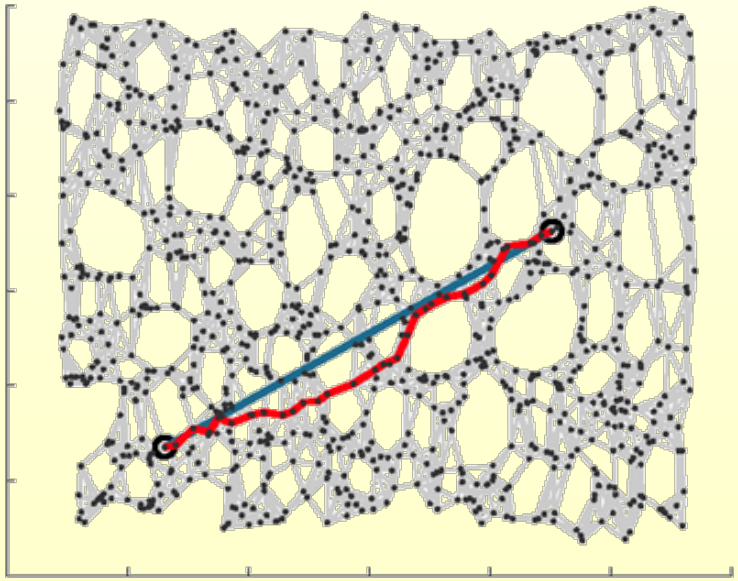




Euclidean Embedding

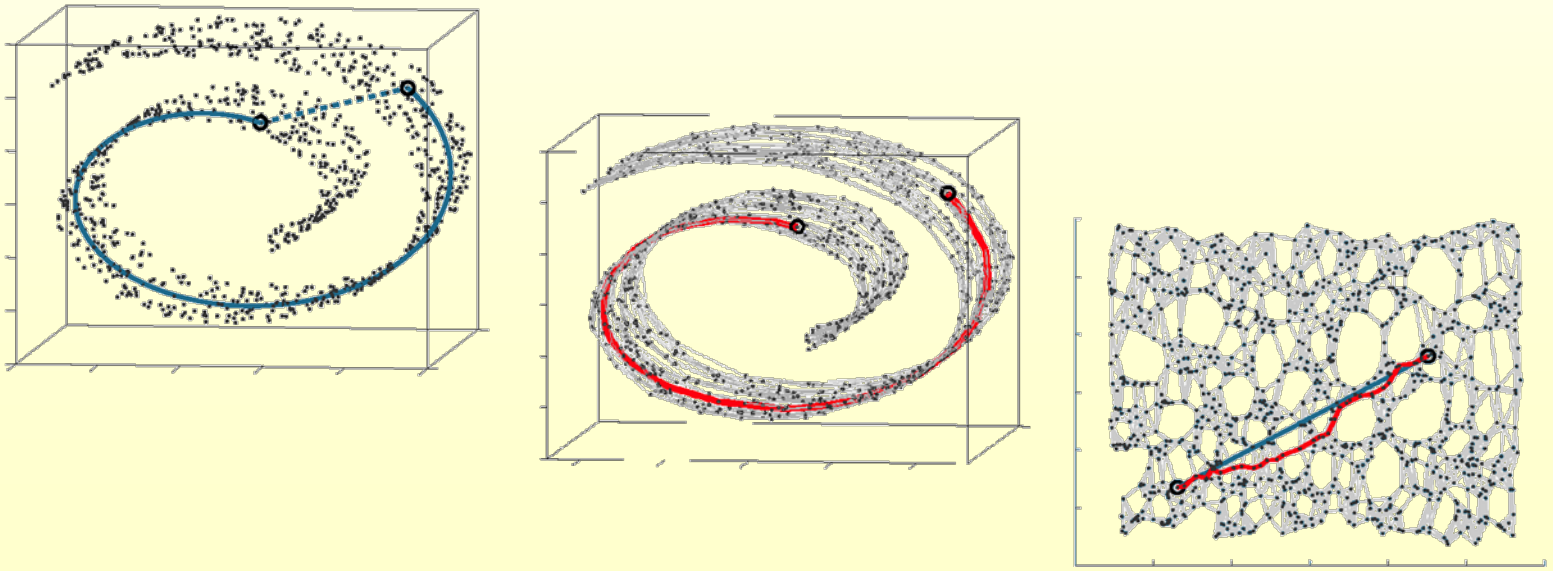
- **Step 3**
  - Constructing an embedding of the data in *d*-dimensional Euclidean space *Y* that best preserves the inter-point distances

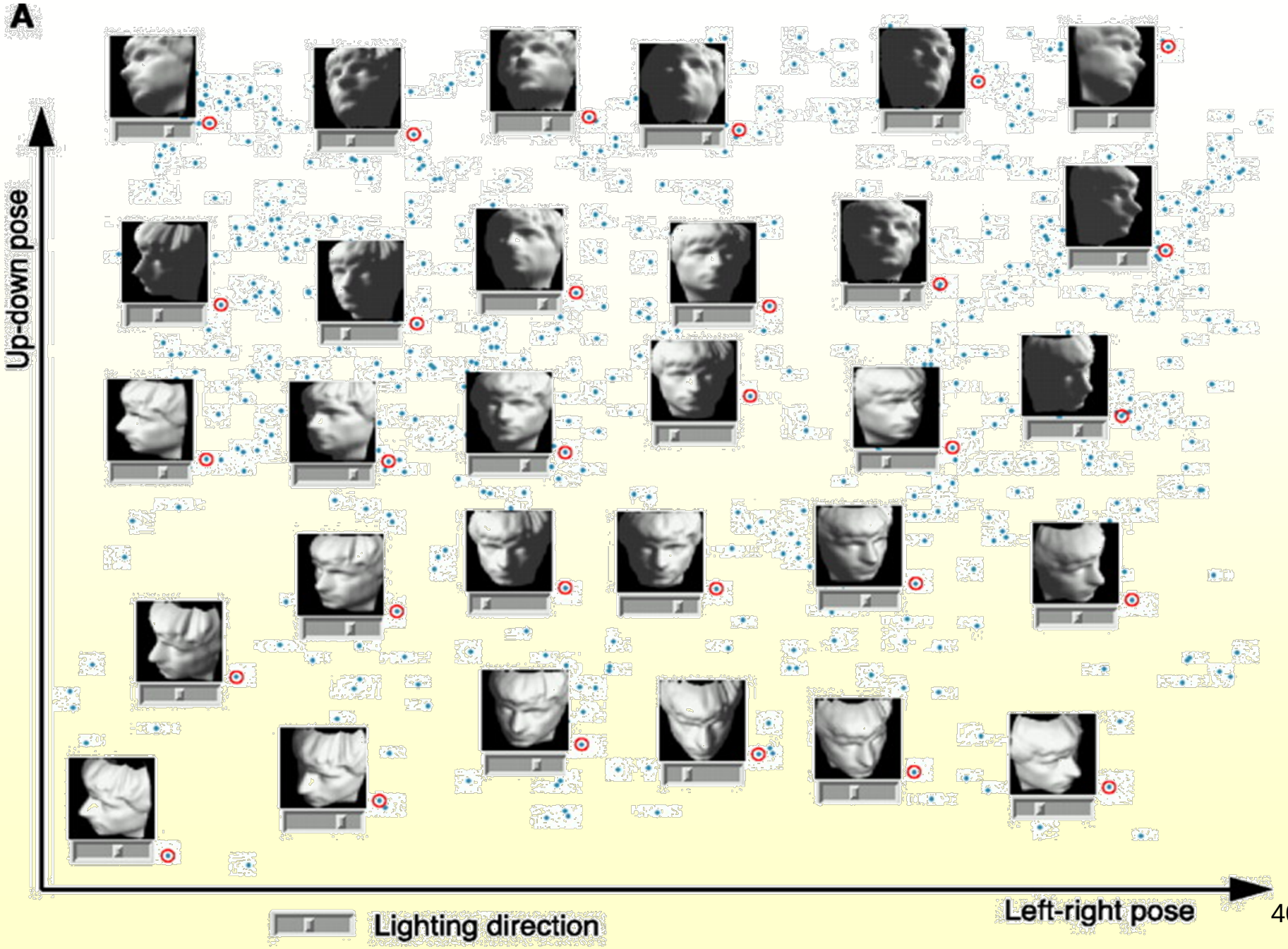
- This is of course nothing but an MDS problem



Recovery Guarantees

- Isomap is guaranteed asymptotically to recover the true dimensionality and geometric structure of non-linear manifolds.
- As the sample data point density increases, the graph distances provide increasingly better approximations to the intrinsic geodesic distances.




A
Up-down pose
Left-right pose
Lighting direction

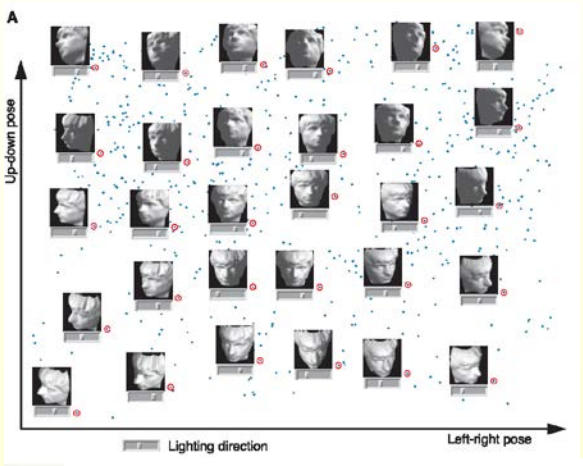
ISOMAP Examples

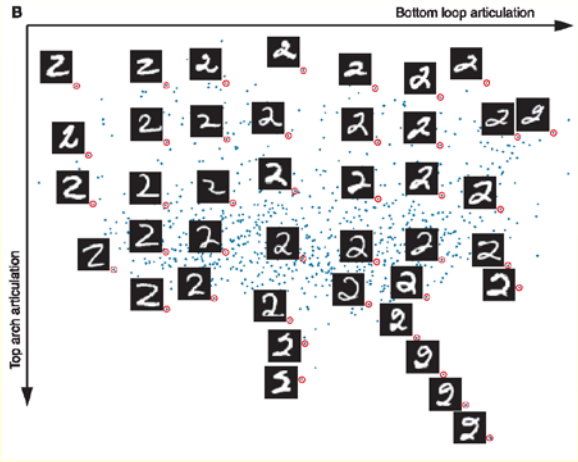
 # Face

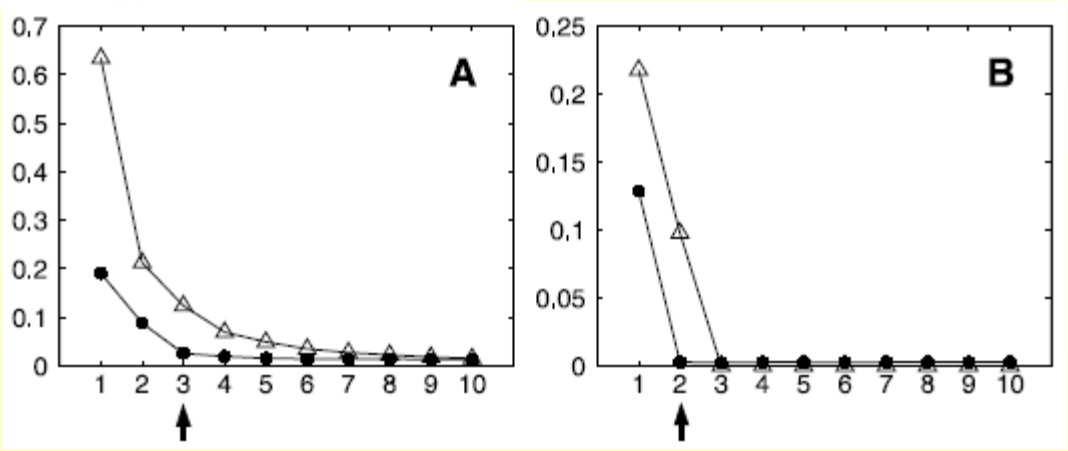
: face pose and illumination

Hand writing

: bottom loop and top arch







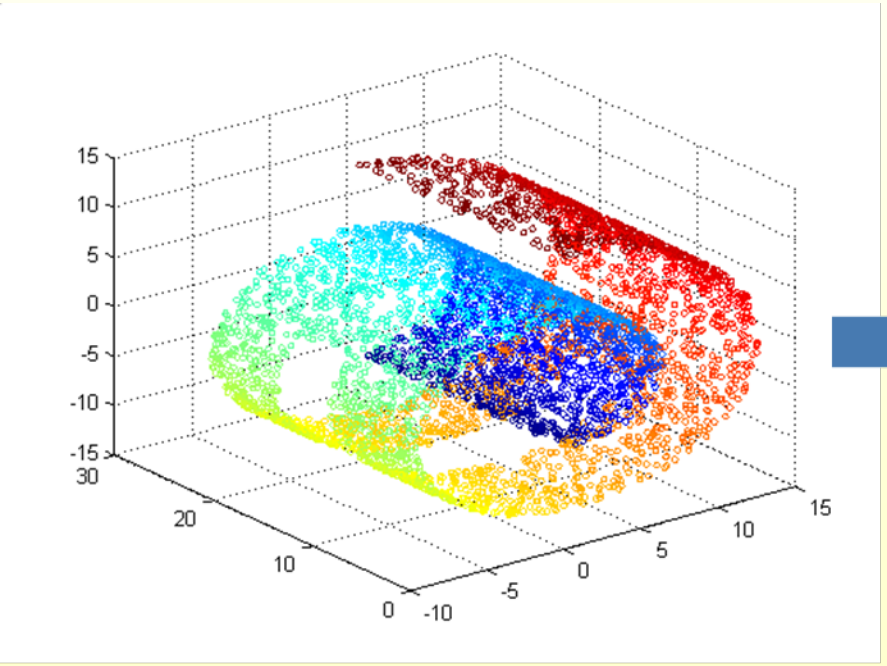
MDS : open triangles

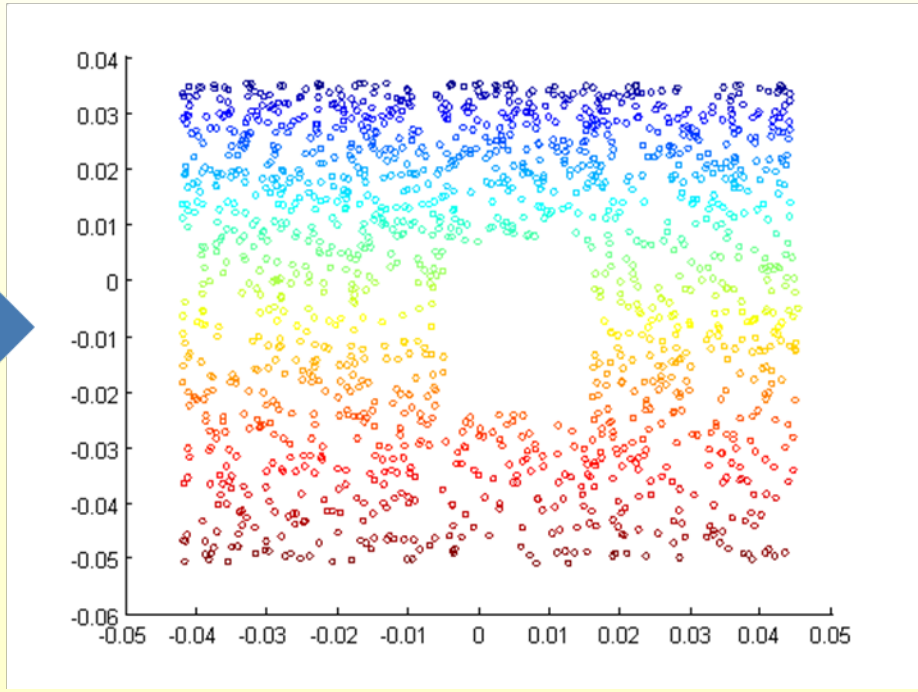
Isomap : filled circles

Laplacian Eigenmaps

Laplacian Eigenmaps
(M. Belkin, P. Niyogi)

- Start same as Isomap, but use a spectral embedding in lieu of MDS



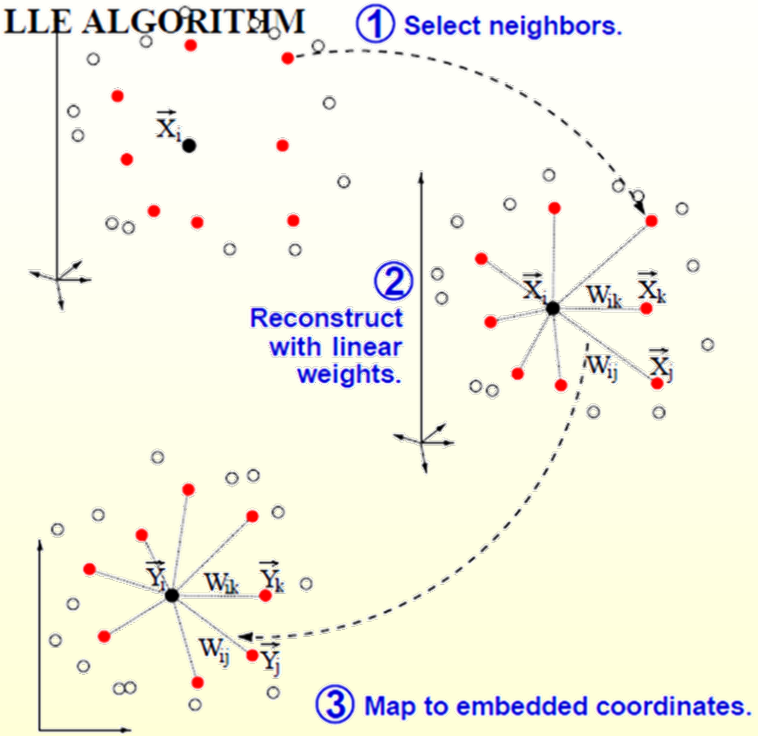


Hole distorts long geodesic distances, but affects less diffusion distances

Locally Linear Embeddings

Locally Linear Embeddings (LLE)

(S. T. Roweis and L. K. Saul)



- Define neighborhood relations between points (build NN graph)
  - *k* nearest neighbors
  - ε-balls
- Find weights that reconstruct each data point from its neighbors:

$$\min_{\sum_j w_{ij}=1} \left\| \mathbf{x}_i - \sum_{j \in N(i)} w_{ij} \mathbf{x}_j \right\|^2$$

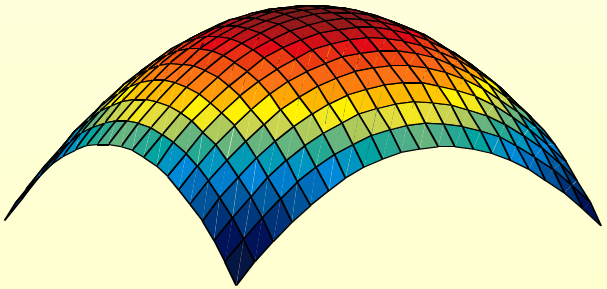
- Find low-dimensional coordinates so that the same weights hold: $\mathbf{x}_1', \ldots, \mathbf{x}_n' \in R^d$

$$x_i' = Y_i$$

$$\min_{\mathbf{x}_1', \ldots, \mathbf{x}_n'} \sum_i \left\| \mathbf{x}_i' - \sum_{j \in N(i)} w_{ij} \mathbf{x}_j' \right\|^2$$

From Local to Global

- The weights w_{ij} capture the local shape
  - Invariant to translation, rotation and scale of the neighborhood
  - If the neighborhood lies on a manifold, the *local* mapping from the global coordinates (R^D) to the surface coordinates (R^d) is almost linear
  - Thus, the weights w_{ij} should hold also for manifold (R^d) coordinate system!



$$\min_{\sum_j w_{ij}=1} \left\| \mathbf{x_i} - \sum_{j \in N(i)} w_{ij} \mathbf{x_j} \right\|^2$$

$$\min_{\mathbf{x_1}', \ldots, \mathbf{x_n}'} \sum_i \left\| \mathbf{x_i}' - \sum_{j \in N(i)} w_{ij} \mathbf{x_j}' \right\|^2$$

Solving the Minimizations

- Linear least squares (using Lagrange multipliers)

$$\min_{\sum_j w_{ij}=1} \left\| \mathbf{x_i} - \sum_{j \in N(i)} w_{ij} \mathbf{x_j} \right\|^2$$

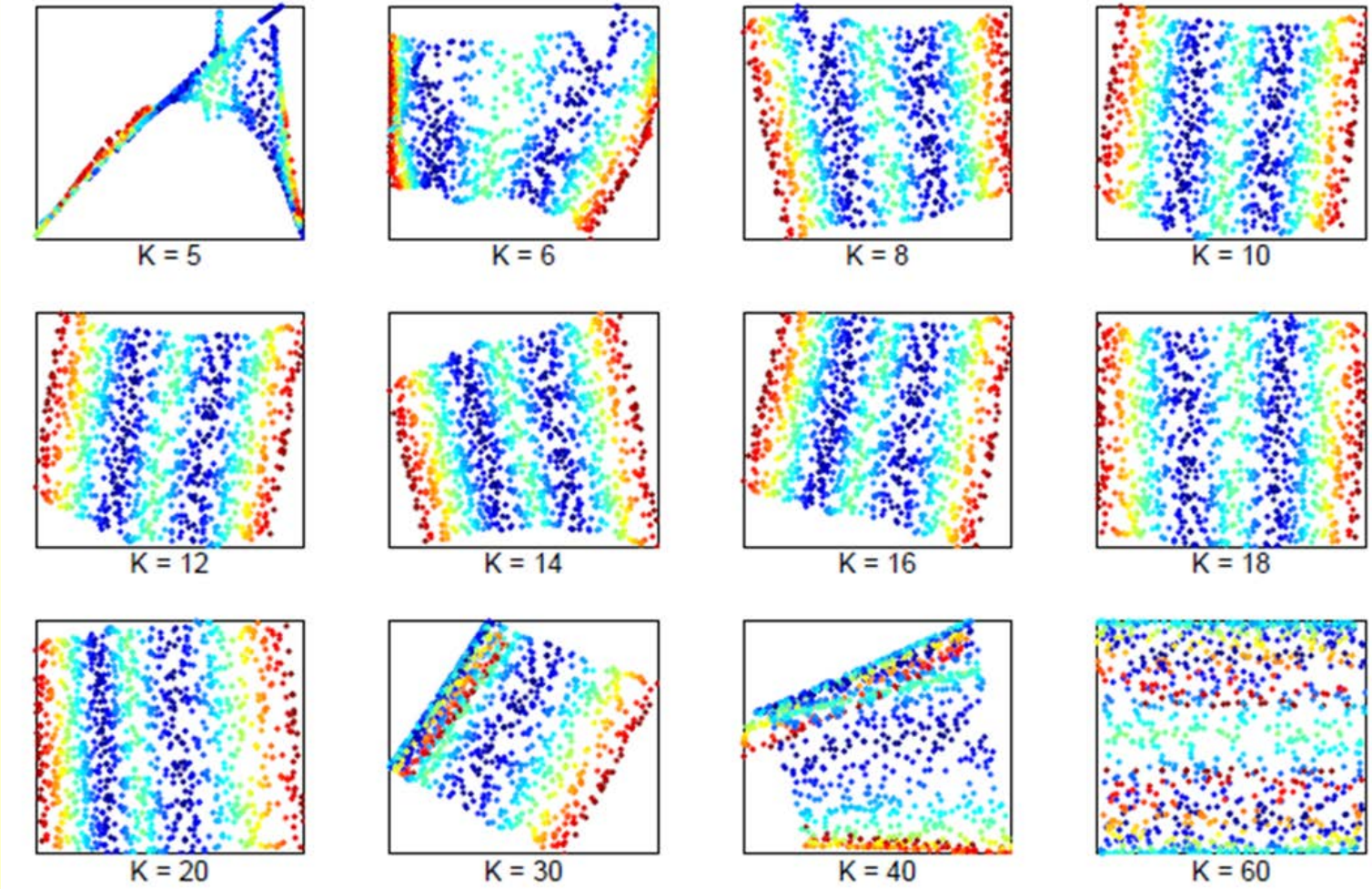
- To find $\mathbf{x_1}', \ldots, \mathbf{x_n}' \in R^d$ that minimize,

$$\min_{\mathbf{x_1}', \ldots, \mathbf{x_n}'} \sum_i \left\| \mathbf{x_i}' - \sum_{j \in N(i)} w_{ij} \mathbf{x_j}' \right\|^2$$

a sparse eigenvalue problem is solved. Additional constraints are added for conditioning:

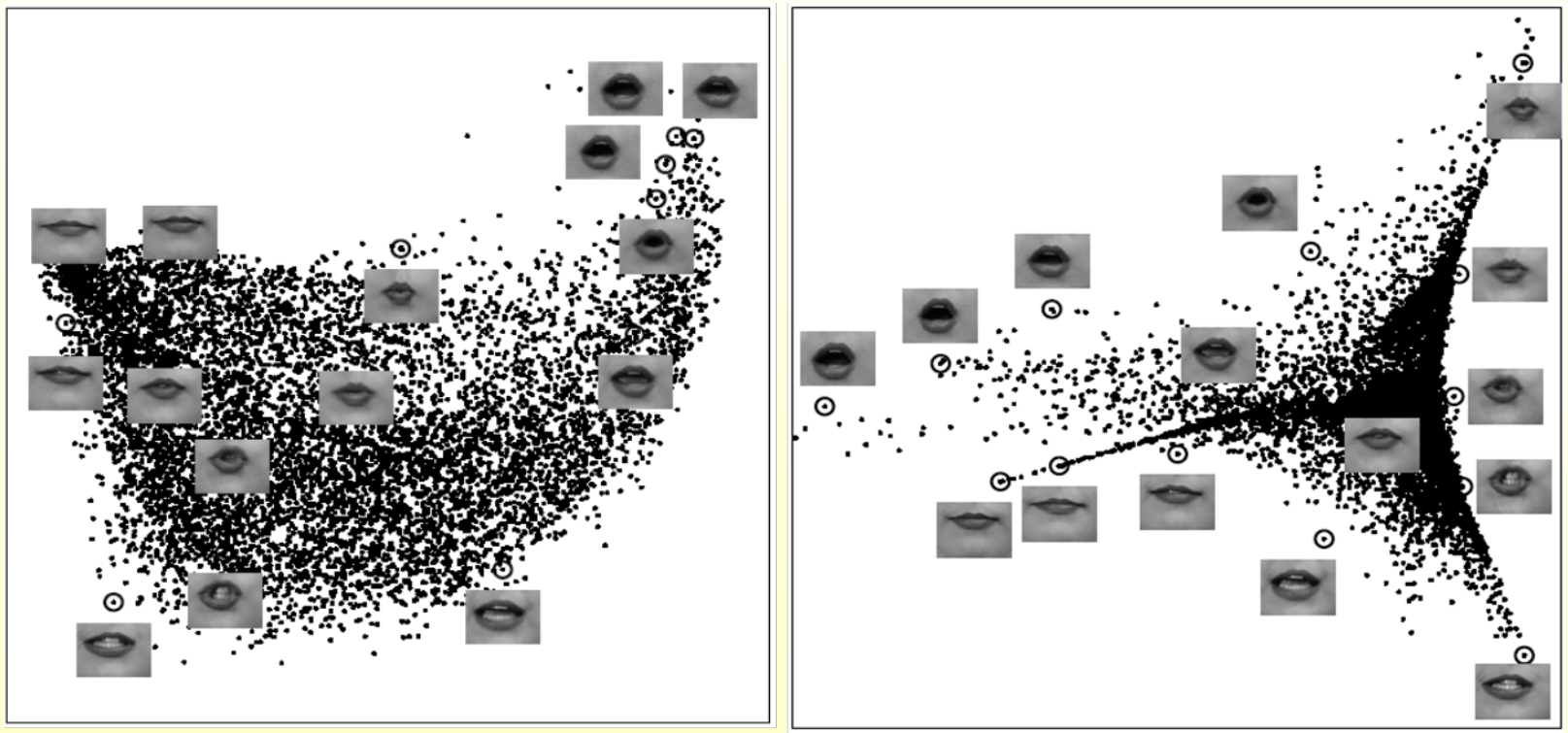
$$\sum_i \mathbf{x_i}' = 0, \qquad \frac{1}{n} \sum_i \mathbf{x_i}' \mathbf{x_i}'^T = I$$

Effect of Neighborhood Size on LLE



LLE Experiments

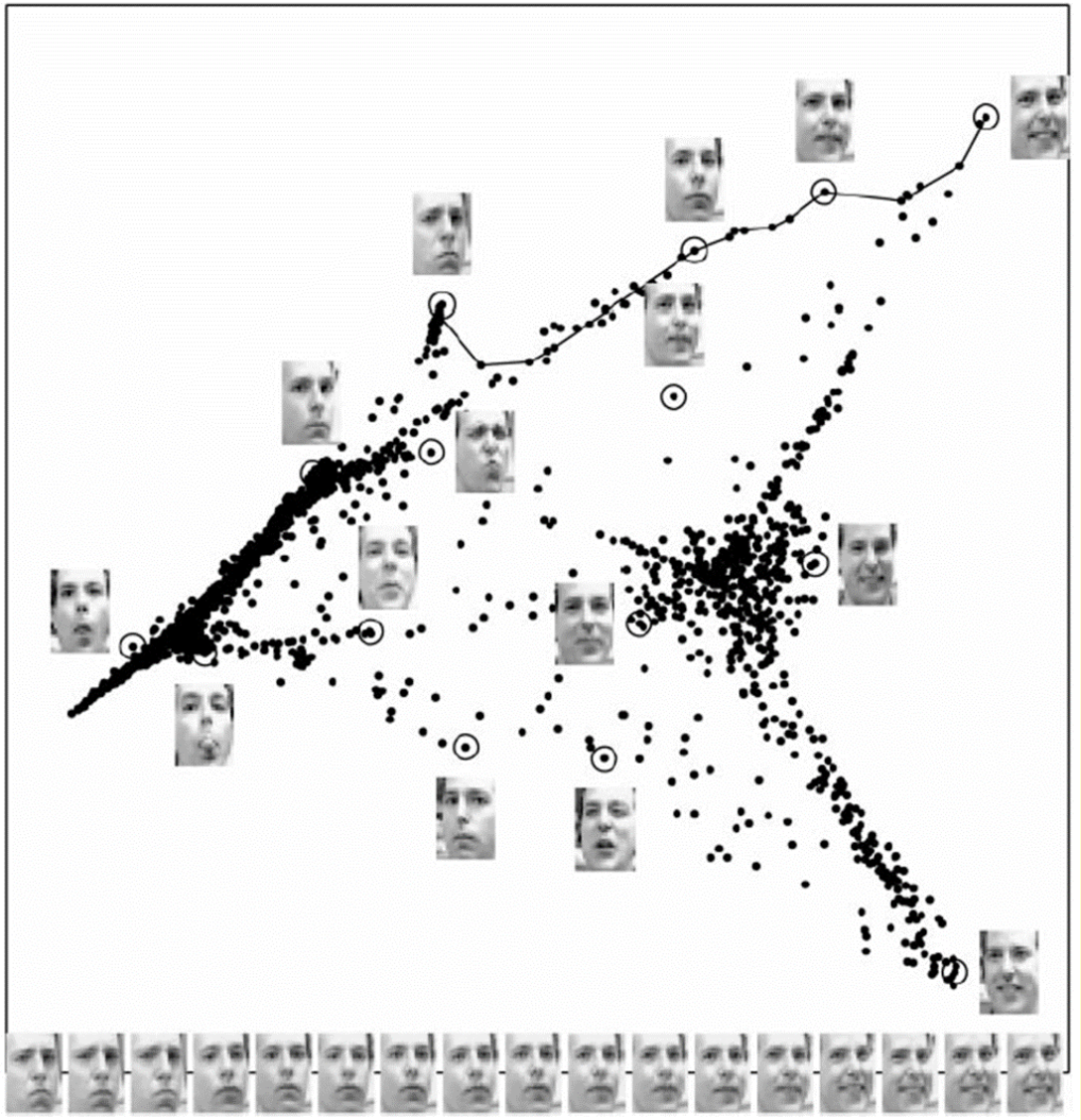
- Lips



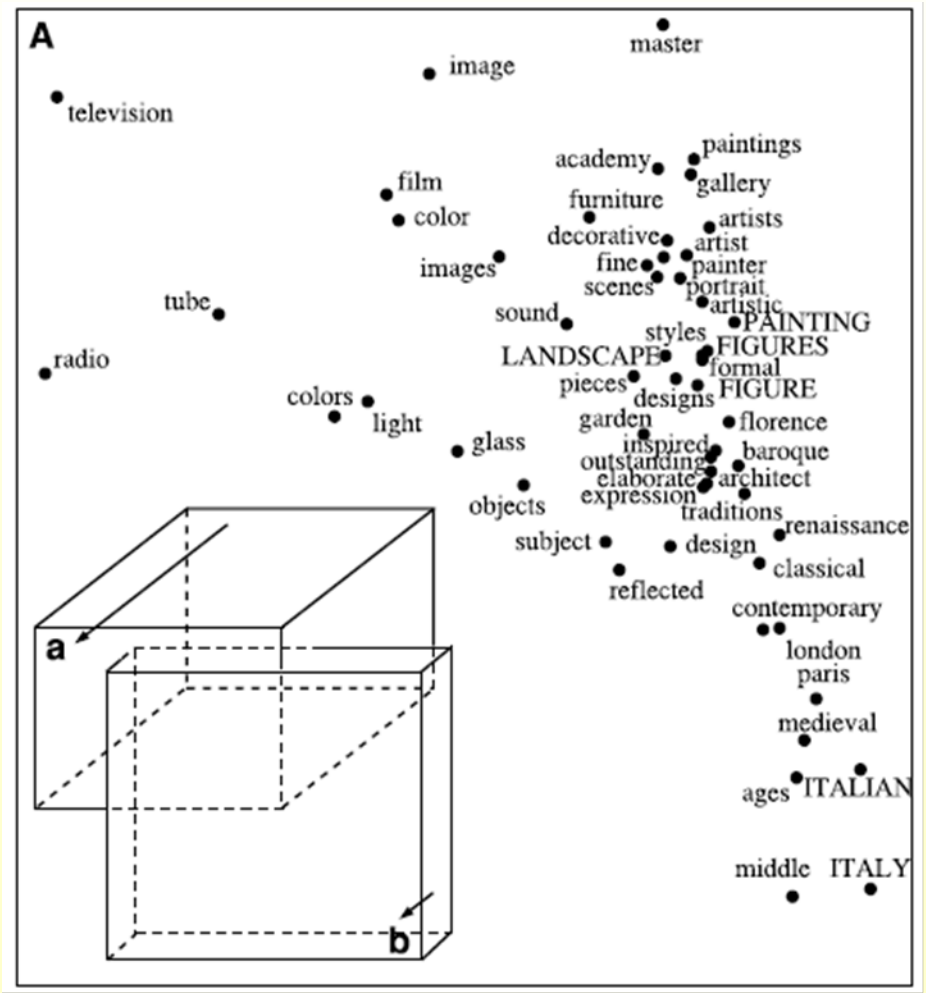
PCA

LLE

LLE Experiments: Faces

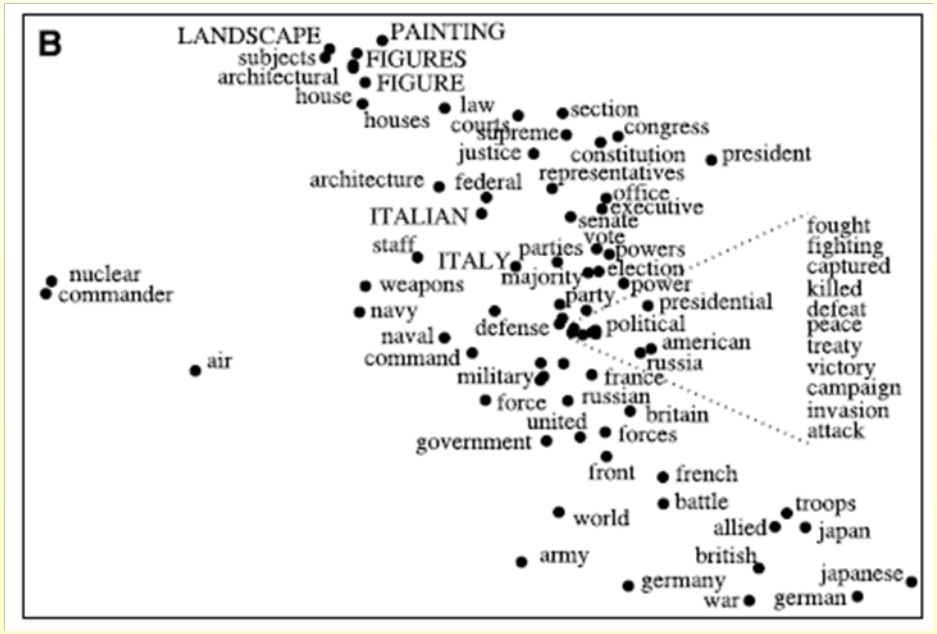


LLE Experiments: Words



Based on word counts in encyclopedia articles.

Note how LLE collocates words with similar semantic contexts in this continuous space.

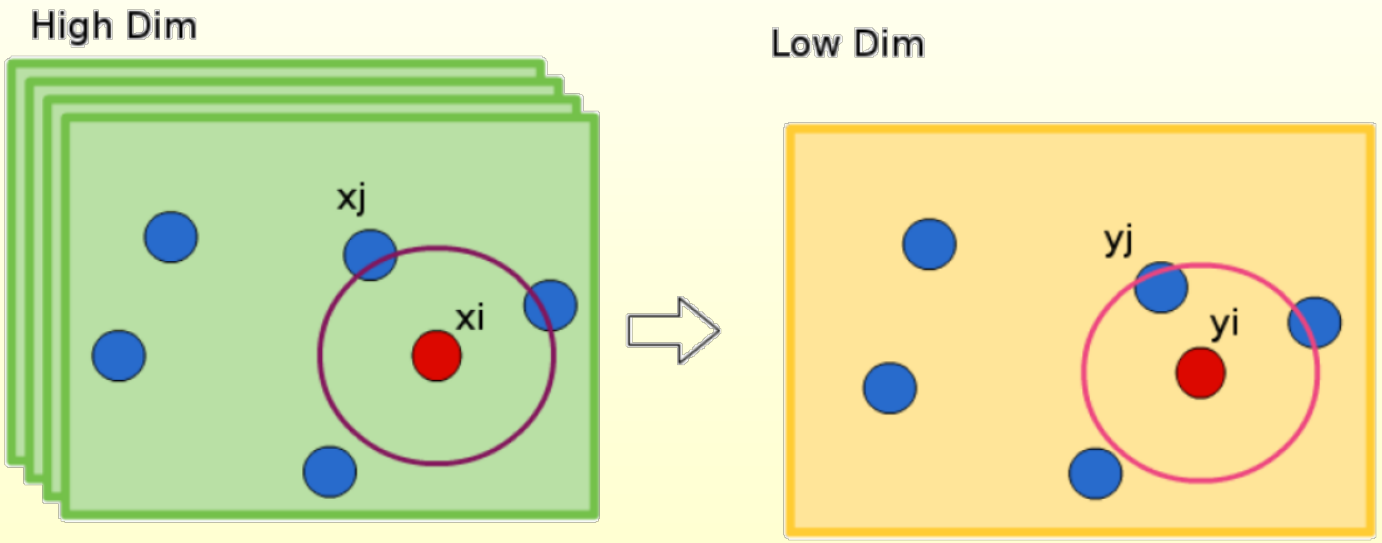


t-SNE

Laurens van der Maaten and Geoffrey Hinton, JMLR 2008

t-Distributed Stochastic Neighbor Embedding

Measure pairwise similarities between high-dimensional and low-dimensonal objects



$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)}$$

Stochastic Neighbor Embedding

Converting the high-dimensional Euclidean distances into conditional probabilities that represent similarities

- Similarity of datapoints in High Dimension

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)}$$

- Similarity of datapoints in Low Dimension

$$q_{j|i} = \frac{exp(-||y_i - y_j||^2)}{\sum_{k \neq i} exp(-||y_i - y_k||^2)}$$

- Cost function

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}}$$

Minimize the cost function using gradient descent

Symmetric SNE

- Minimize a single KL divergence between a joint probability distribution

$$C = KL(P||Q) = \sum_i \sum_{j \neq i} p_{ij} log \frac{p_{ij}}{q_{ij}}$$

- The obvious way to redefine the pairwise similarities is

$$p_{ij} = \frac{exp(-||x_i - x_j||^2/2\sigma^2)}{\sum_{k \neq l} exp(-||x_l - x_k||^2/2\sigma^2)}$$

$$q_{ij} = \frac{exp(-||y_i - y_j||^2)}{\sum_{k \neq l} exp(-||y_l - y_k||^2)}$$

However, in practice we symmetrize (or average) the conditionals

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

Set the bandwidth σ_i such that the conditional has a fixed perplexity (effective number of neighbors) $Perp(P_i) = 2^{H(P_i)}$, typical value is about 5 to 50

t-Distribution

Use heavier tail distribution than Gaussian in low-dim space, we choose

$$q_{ij} \propto (1 + ||y_i - y_j||^2)^{-1}$$

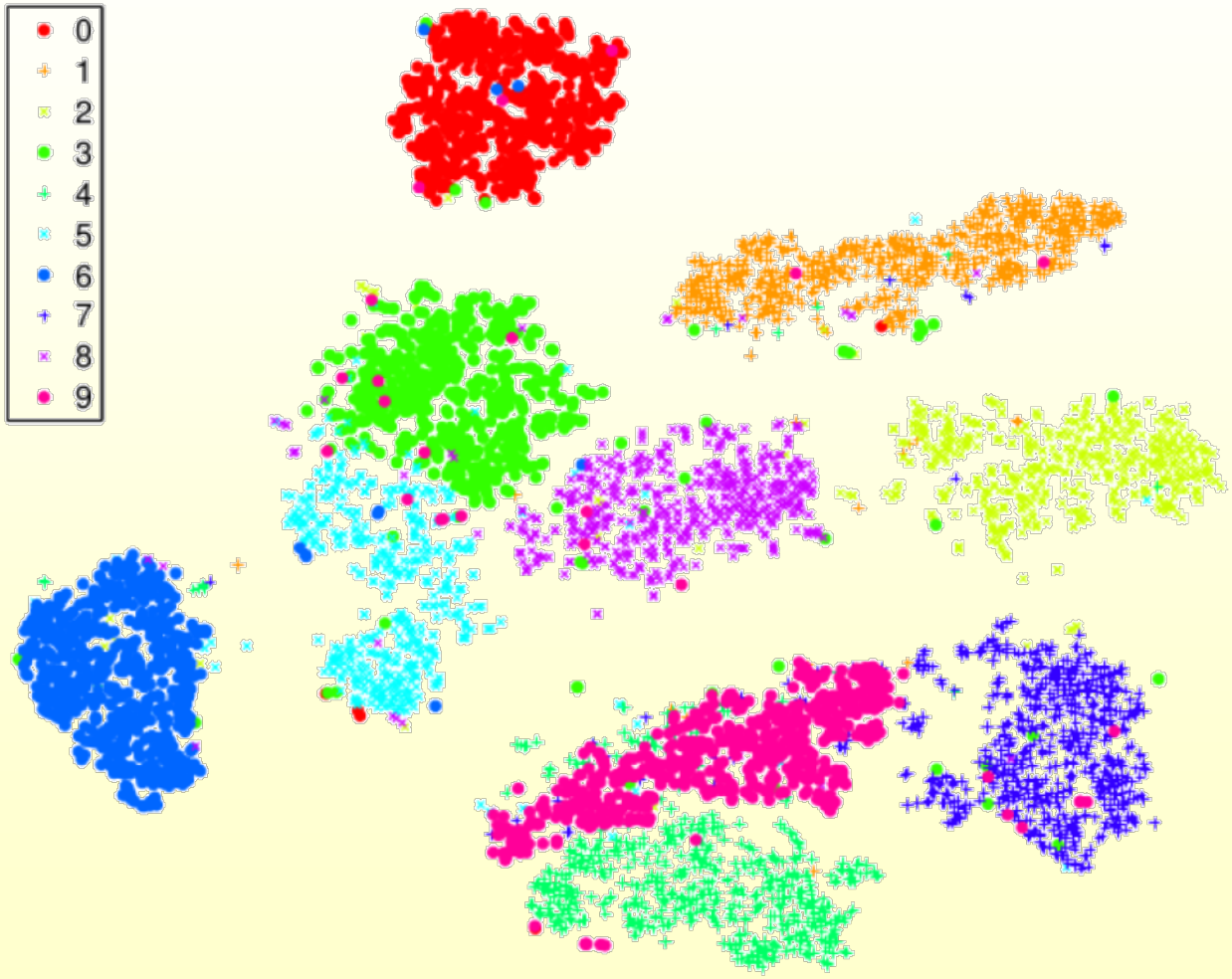
- Similarity of datapoints in High Dimension

$$p_{ij} = \frac{exp(-||x_i - x_j||^2/2\sigma^2)}{\sum_{k \neq l} exp(-||x_l - x_k||^2/2\sigma^2)}$$

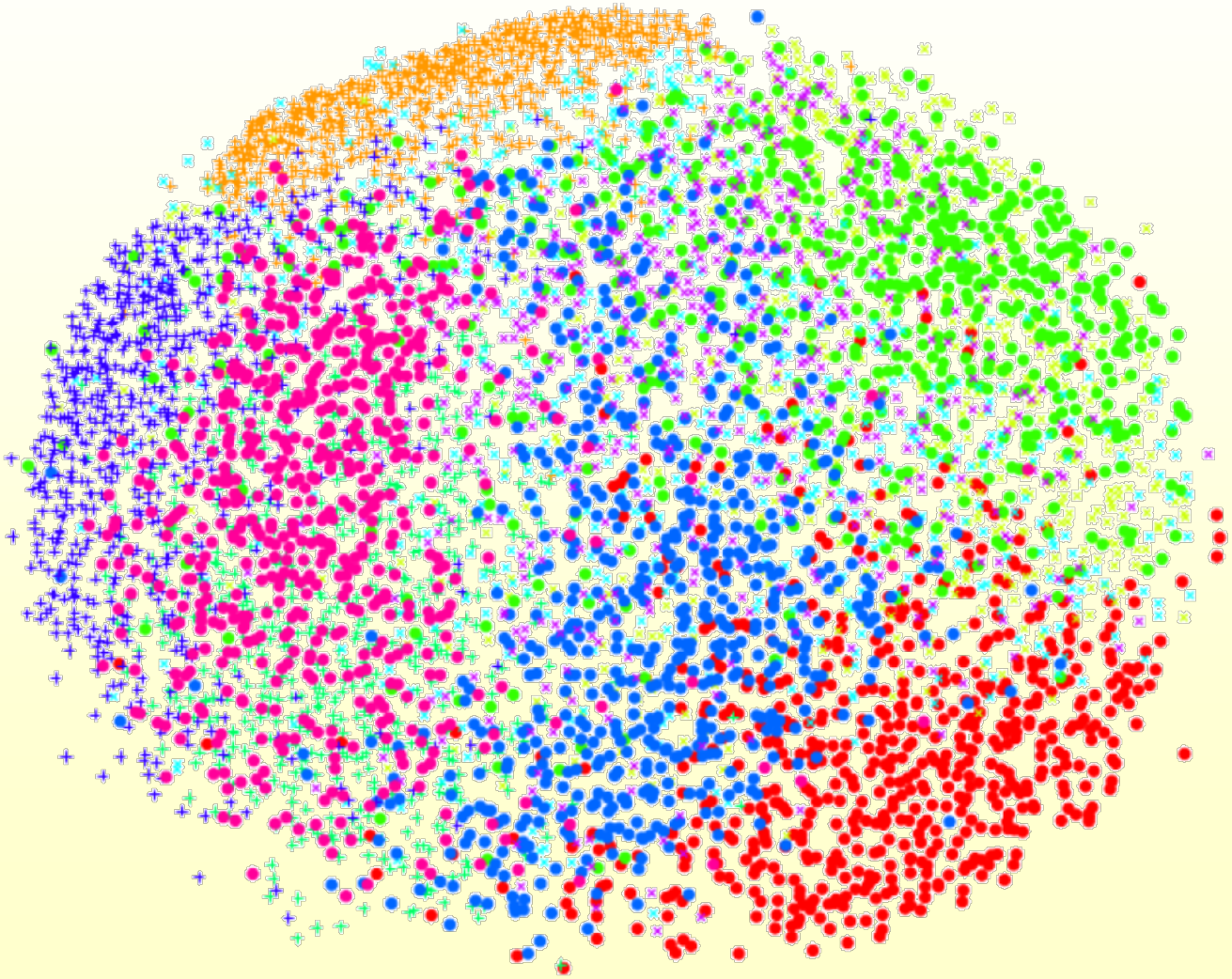
- Similarity of datapoints in Low Dimension

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||y_k - y_l||^2)^{-1}}$$

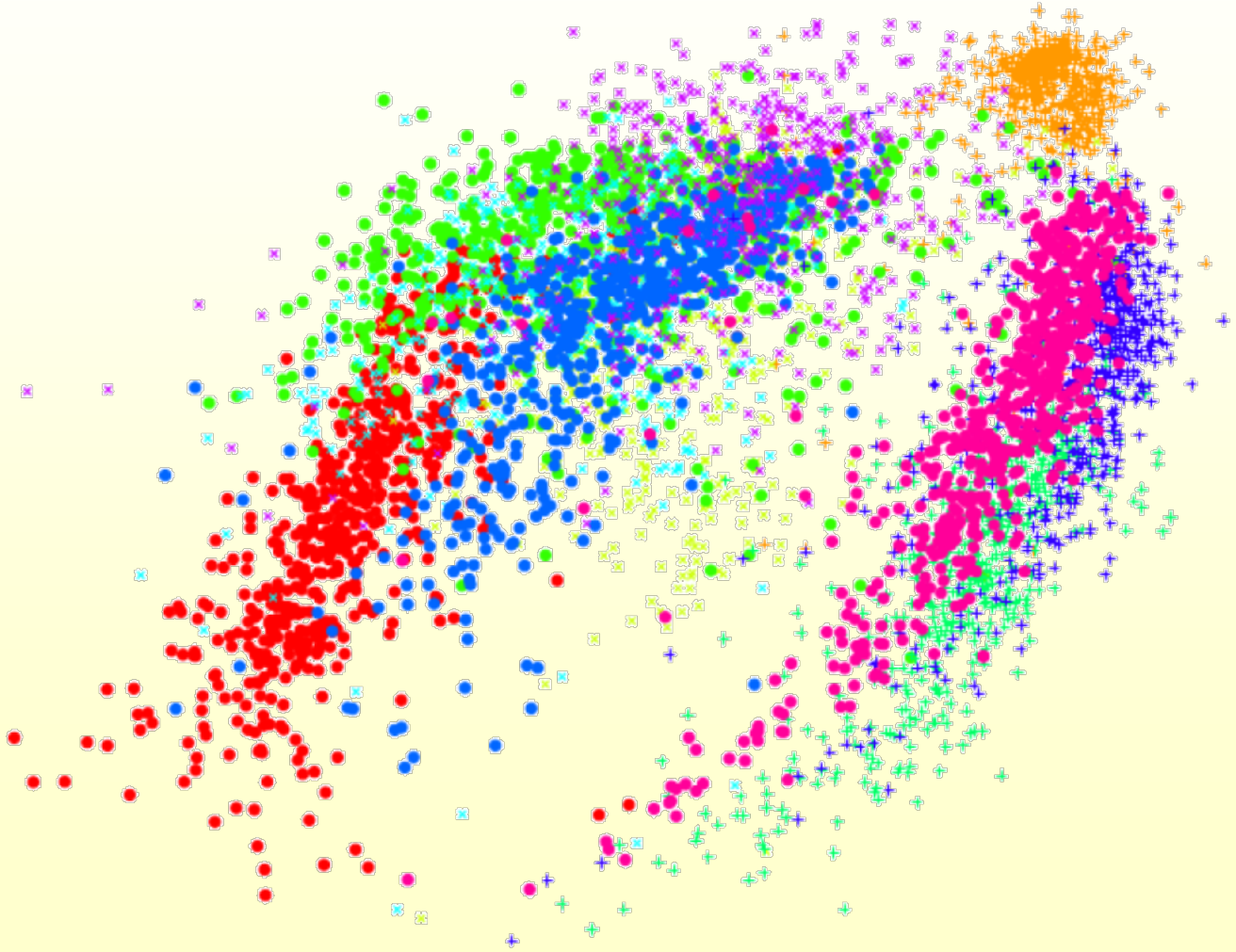
MNIST t-SNE



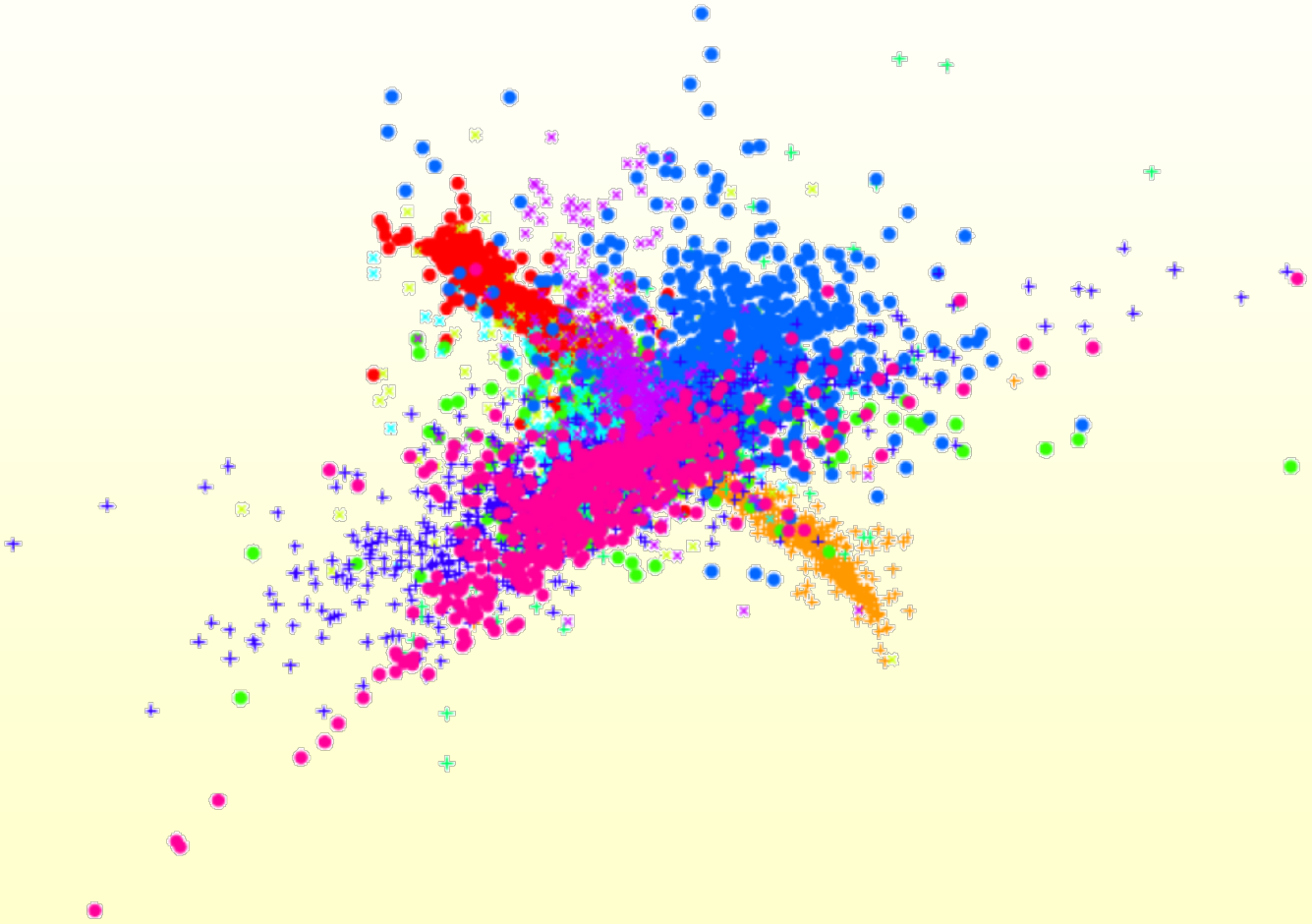
MNIST MDS w. Sammon



MNIST Isomap



MNIST LLE



Source Codes

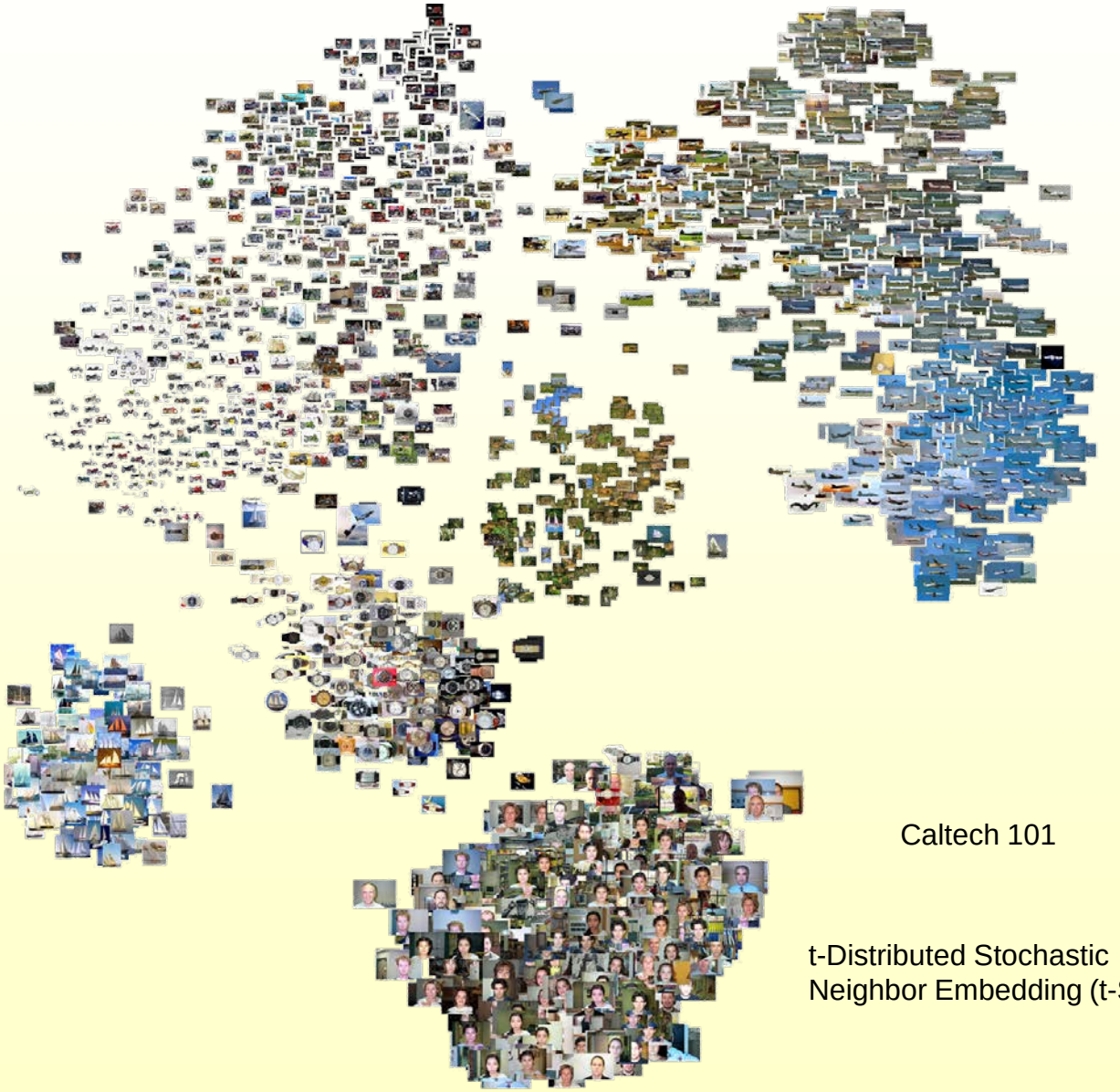
- t-SNE (Matlab, CUDA, Binary, Python, Torch, Julia, R and JavaScript)
- Parametric t-SNE (Matlab)
- Barnes-Hut-SNE (with C++, Matlab, Python, Torch, and R wrappers)

Many More Methods

Many NLDR Methods

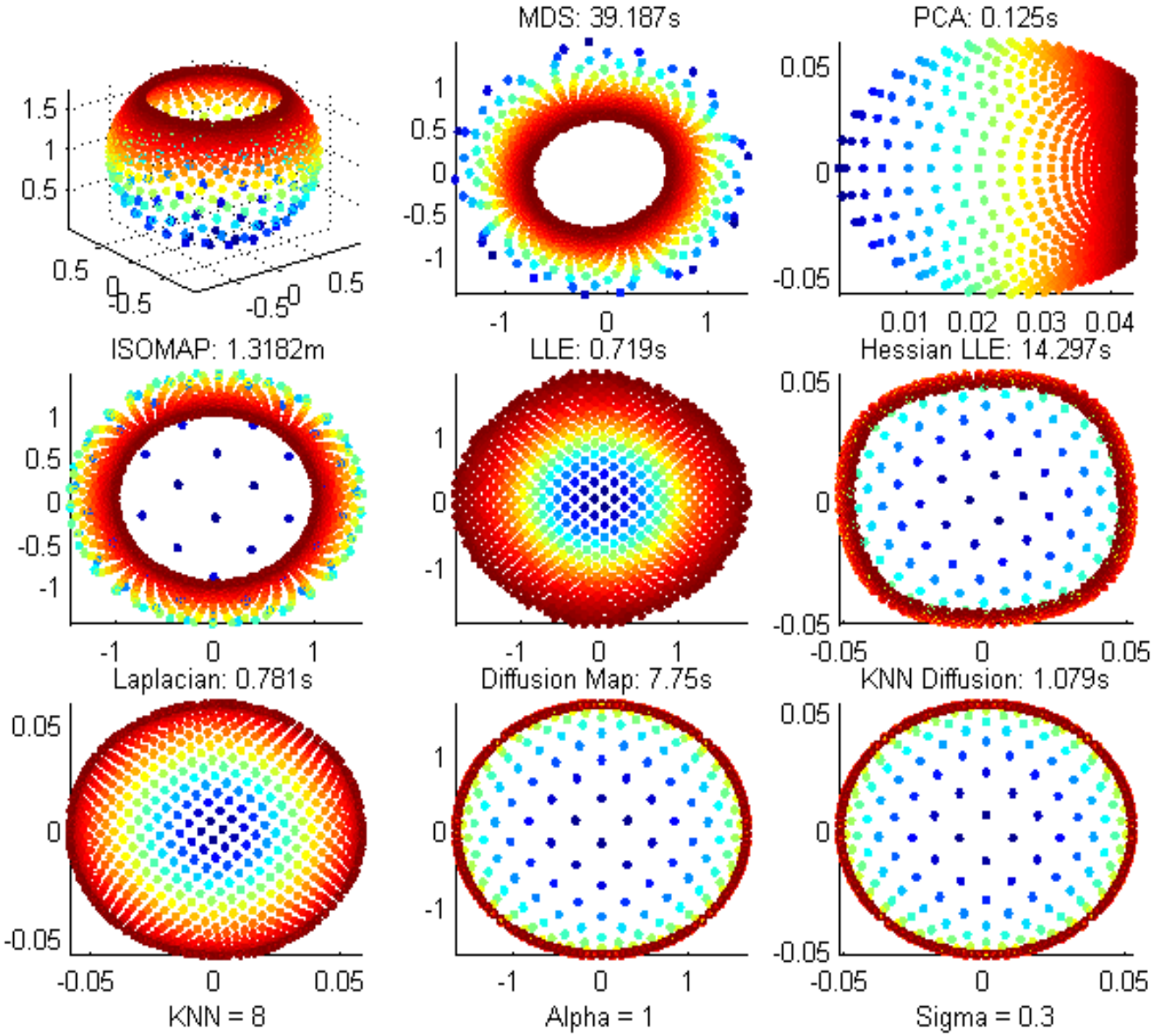
Contents [hide]

From Wikipedia



Caltech 101

t-Distributed Stochastic Neighbor Embedding (t-SNE)


MDS: 39.187s
PCA: 0.125s
ISOMAP: 1.3182m
LLE: 0.719s
Hessian LLE: 14.297s
Laplacian: 0.781s
Diffusion Map: 7.75s
KNN Diffusion: 1.079s
KNN = 8
Alpha = 1
Sigma = 0.3

	MDS	PCA	ISOMAP	LLE	Laplacian	Diffusion Map	KNN Diffusion	Hessian
Speed	Very slow	Extremely fast	Extremely slow	Fast	Fast	Fast	Fast	Slow
Infers geometry?	NO	NO	YES	YES	YES	MAYBE	MAYBE	YES
Handles non-convex?	NO	NO	NO	MAYBE	MAYBE	MAYBE	MAYBE	YES
Handles non-uniform sampling?	YES	YES	YES	YES	NO	YES	YES	MAYBE
Handles curvature?	NO	NO	YES	MAYBE	YES	YES	YES	YES
Handles corners?	NO	NO	YES	YES	YES	YES	YES	NO
Clusters?	YES	YES	YES	YES	NO	YES	YES	NO
Handles noise?	YES	YES	MAYBE	NO	YES	YES	YES	YES
Handles sparsity?	YES	YES	YES	YES	YES	NO	NO	NO may crash
Sensitive to parameters?	NO	NO	YES	YES	YES	VERY	VERY	YES

The End