

Nonlinear Dimensionality Reduction (a.k.a. Manifold Learning)

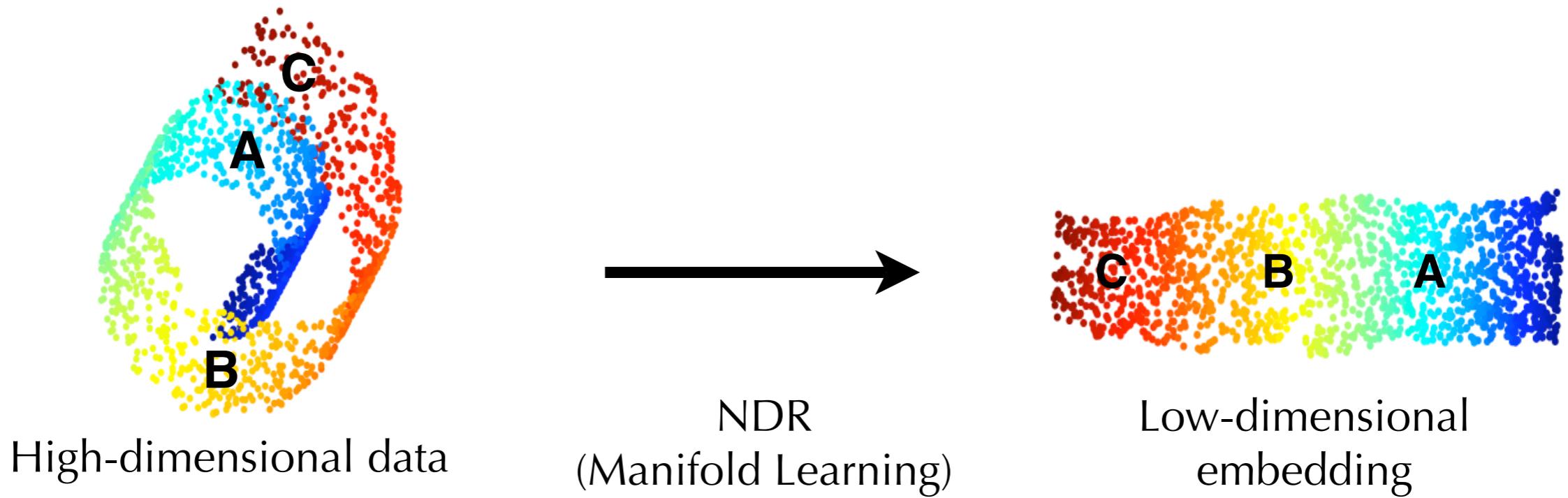
David Capel

346B IST Bldg

capel@cse.psu.edu



What is “nonlinear dimensionality reduction?”



- We often suspect that high-dim may actually lie on or near a low-dim manifold (often much lower!)
- It would be useful if we could reparametrize the data in terms of this manifold, yielding a low-dim *embedding*
- **BUT** - we typically don't know the form of this manifold

Why might this be useful?

- The variation observed in high-dimensional signals often has much lower-dimensional explanation



64x64 pixel images parametrized by just 3 variables (pose and lighting direction)

- Discovering these modes of variation helps us understand the underlying structure of the data and the process that generated it
 - Visualization of high-dimensional data
 - Machine learning and pattern recognition

Okay, so how do we learn the embedding?

- Given high-dim data sampled from an unknown low-dim manifold, how can we automatically recover a good embedding?



A Global Geometric Framework for Nonlinear Dimensionality Reduction

Tenenbaum, de Silva and Langford
Science (Vol. 290, Dec 2000, 2319-2323)

Nonlinear Dimensionality Reduction by Locally Linear Embedding

Roweis and Saul
Science (Vol. 290, Dec 2000, 2323-2327)

Outline

- Linear subspace embedding
 - Principal Components Analysis (PCA)
 - Metric Multidimensional Scaling (MDS)
- Non-linear manifold learning
 - Isomap (Tenenbaum et al.)
 - Locally Linear Embedding (Roweis et al.)
- Some examples

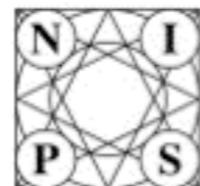
An excellent tutorial ...

Spectral Methods for Dimensionality Reduction

Prof. Lawrence Saul

**Dept of Computer & Information Science
University of Pennsylvania**

NIPS*05 Tutorial, December 5, 2005



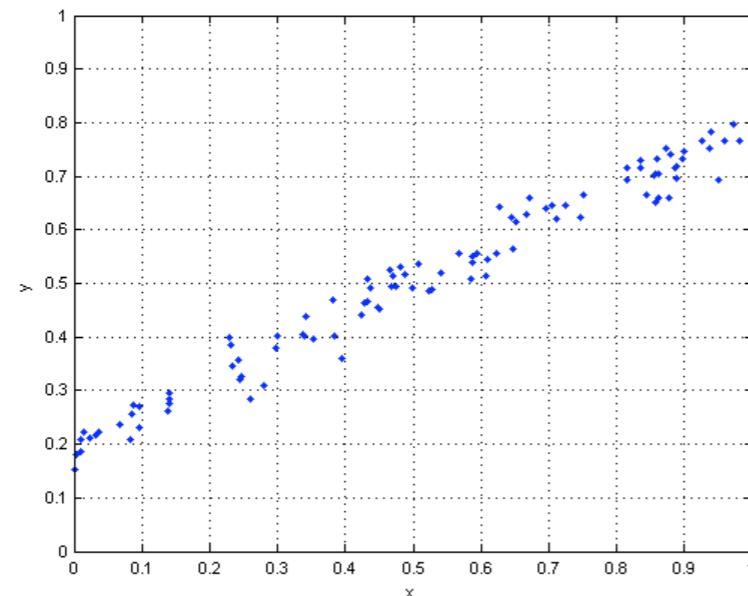
**Neural Information
Processing Systems
Conference**

... from which I have borrowed liberally! Thanks Lawrence!

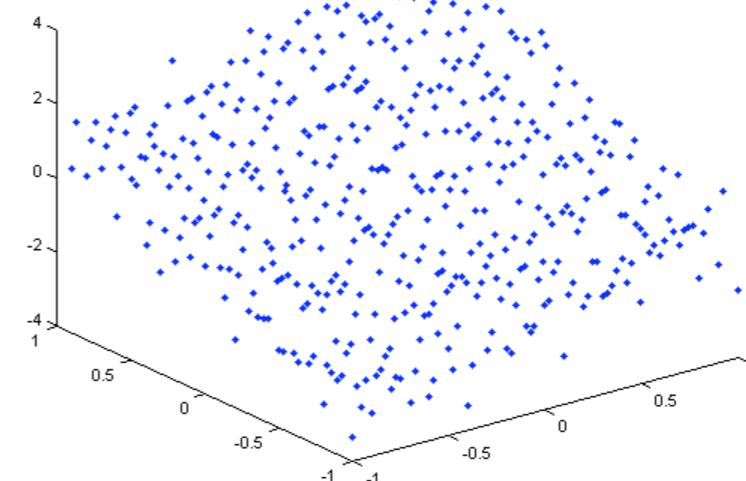
Background - Linear Subspace Embedding

Linear subspaces

- We may often assume that our high-dim data lies on/near a linear subspace



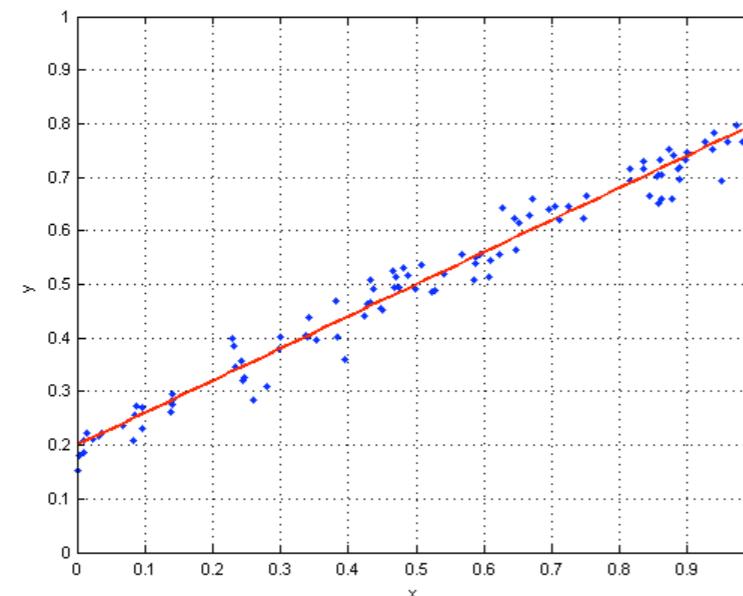
$D_{\text{high}}=2$
 $D_{\text{low}}=1$



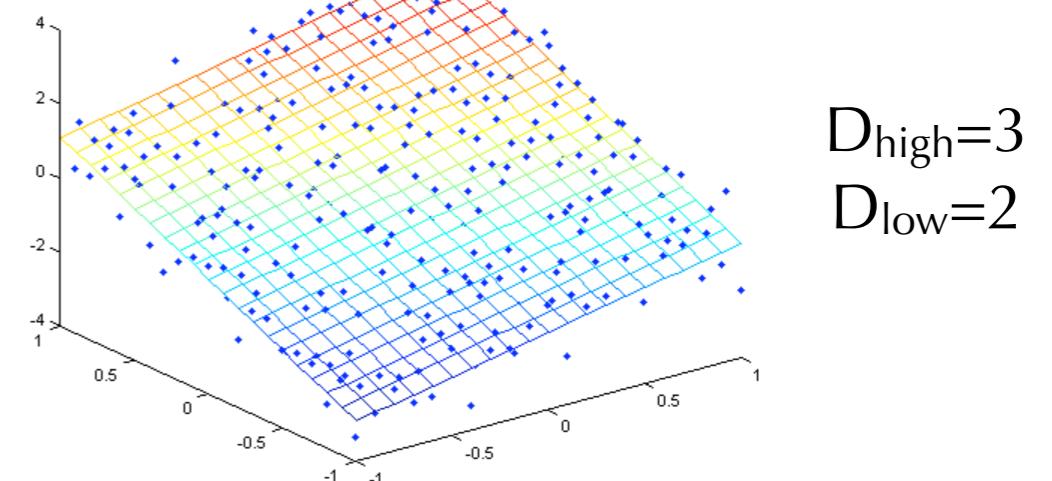
$D_{\text{high}}=3$
 $D_{\text{low}}=2$

Linear subspaces

- We may often assume that our high-dim data lies on/near a linear subspace



$$\begin{aligned} D_{\text{high}} &= 2 \\ D_{\text{low}} &= 1 \end{aligned}$$



$$\begin{aligned} D_{\text{high}} &= 3 \\ D_{\text{low}} &= 2 \end{aligned}$$

- In this case, well-known, stable tools exist for determining the parameters of this subspace
 - Principal Components Analysis
 - Metric Multidimensional Scaling
- Among the most widely-used algorithms in engineering!

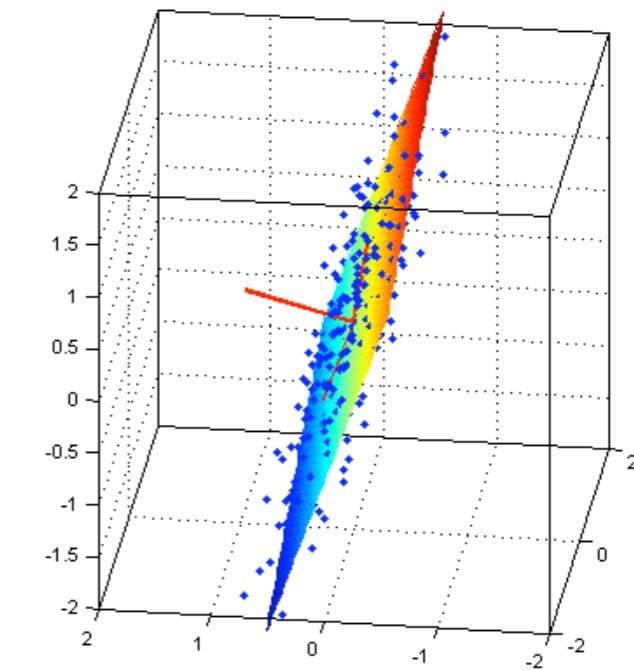
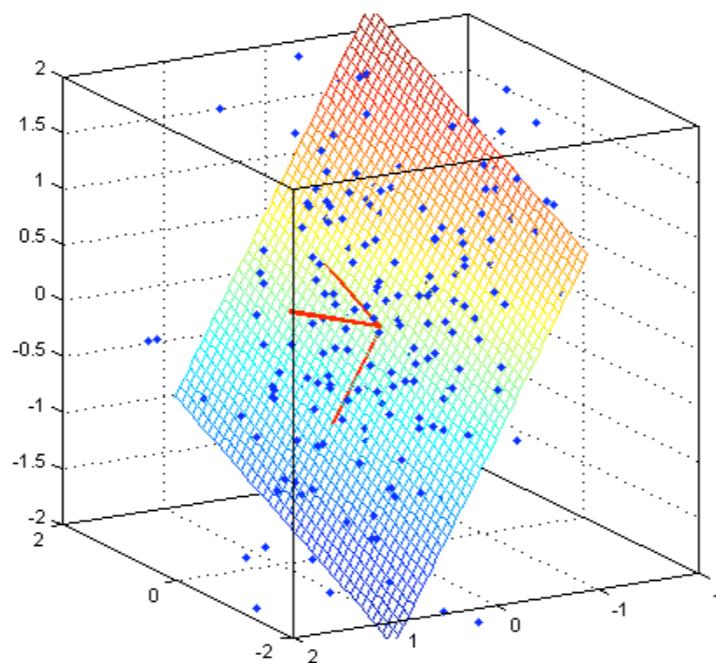
Notation

- We have a quantity \mathbf{N} of \mathbf{D} -dimensional data points \mathbf{x}
- We seek to map \mathbf{x} to a set of \mathbf{d} -dimensional points \mathbf{y}
- \mathbf{N} is large and $\mathbf{d} \ll \mathbf{D}$

Principal Components Analysis (PCA)

- Project data onto an **orthonormal basis**, chosen so as to **maximize the variance** of the projected data

$$\vec{y}_i = P\vec{x}_i$$



- Choose subspace as the d-dimensional hyper-plane spanned by directions of maximum variance

Principal Components Analysis (PCA)

- First, we center the data to have zero empirical mean

$$\sum_i \vec{x}_i = \vec{0}$$

- Then we determine an orthonormal linear projection

$$\vec{y}_i = P\vec{x}_i$$

- ... so as to maximize the projected variance

$$\text{var}(\vec{y}) = \frac{1}{n} \sum_i \|P\vec{x}_i\|^2$$

Principal Components Analysis (PCA)

- Projected variance is given by

$$\text{var}(\vec{y}) = \text{Tr}(P C P^T) \text{ with } C = n^{-1} \sum \vec{x}_i \vec{x}_i^T$$

- where C is the $D \times D$ data covariance matrix, with eigen-value decomposition

$$C = \sum_{\alpha=1}^D \lambda_\alpha \vec{e}_\alpha \vec{e}_\alpha^T \text{ with } \lambda_1 \geq \dots \geq \lambda_D \geq 0$$

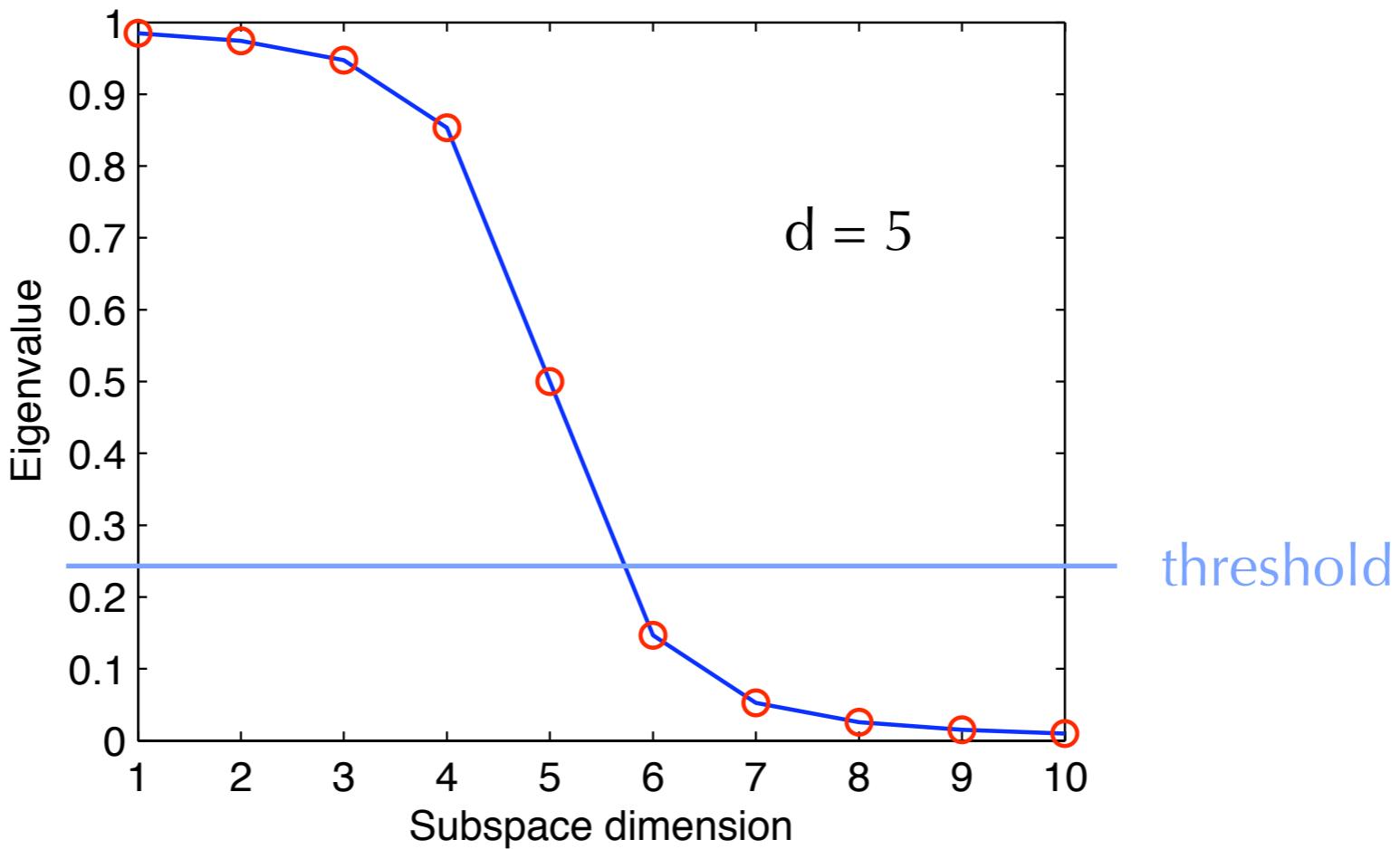
- The projected variance is maximized when

$$P = \sum_{\alpha=1}^d \vec{e}_\alpha \vec{e}_\alpha^T$$

- i.e. projecting into the sub-space spanned by the eigenvectors corresponding to the largest eigenvalues

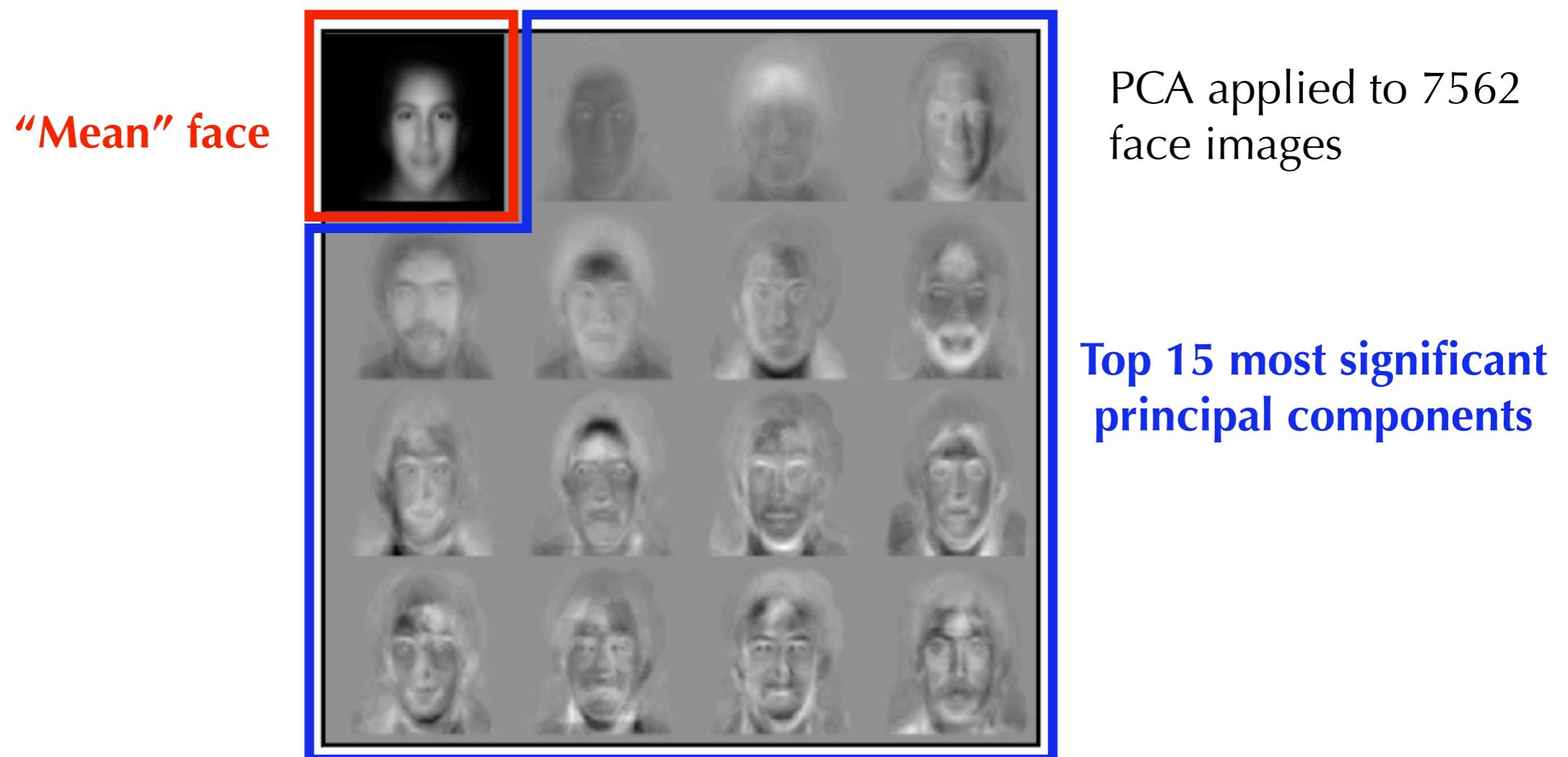
Principal Components Analysis (PCA)

- The intrinsic dimensionality of the subspace may be estimated as the number of significantly large eigenvalues



PCA Example : Eigenfaces

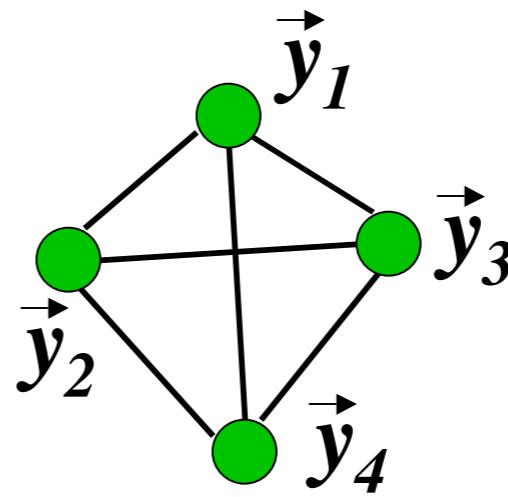
- Sirovich and Kirby (JOSA '87) pioneered application of PCA to model the variation observed in face images
- High-dim (e.g. 128x128 pixel) face images may be modeled by just 50-100 principal components



Multidimensional Scaling (MDS)

- An alternative approach to PCA based on preserving pairwise distances

$$\begin{bmatrix} 0 & \Delta_{12} & \Delta_{13} & \Delta_{14} \\ \Delta_{12} & 0 & \Delta_{23} & \Delta_{24} \\ \Delta_{13} & \Delta_{23} & 0 & \Delta_{34} \\ \Delta_{14} & \Delta_{24} & \Delta_{34} & 0 \end{bmatrix}$$



Given $n(n - 1)/2$ pairwise distances $d_{ij} = \|X_i - X_j\|$, find a low-dimensional embedding $X \rightarrow y$ such that $\|y_i - y_j\| \approx d_{ij}$.

Multidimensional Scaling (MDS)

- Given centered mean-zero data X , we can express the dot products $G_{ij} = \langle X_i, X_j \rangle$ in terms of pairwise distances d_{ij}

$$G_{ij} = \frac{1}{2} \left[\frac{1}{n} \sum_k (d_{ik}^2 + d_{kj}^2) - d_{ij}^2 - \frac{1}{n^2} \sum_{kl} d_{kl}^2 \right] \quad (\text{n.b. useful lemma!})$$

- We then seek new vectors y_i so as to minimize the error function

$$err(y) = \sum_{ij} (G_{ij} - y_i^\top y_j)^2$$

- Matrix \mathbf{G} , consisting of all possible dot products $\langle i, j \rangle$ is known as a *Gram matrix*

Multidimensional Scaling (MDS)

- We aim to approximate \mathbf{G}

$$err(y) = \sum_{ij} (G_{ij} - y_i^\top y_j)^2$$

- Again using the eigen-decomposition of the Gram matrix

$$\mathbf{G} = \sum_{\alpha=1}^n \lambda_\alpha \vec{v}_\alpha \vec{v}_\alpha^\top \quad \text{with } \lambda_1 \geq \dots \geq \lambda_n \geq 0$$

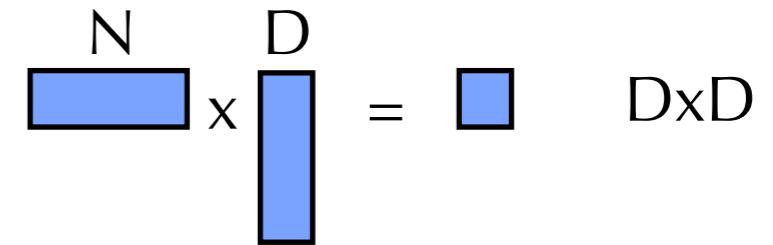
- We immediately see that the optimal approximation of \mathbf{G} is given by an outer-product of the most significant eigenvectors

$$y_{\alpha i} = \sqrt{\lambda_\alpha} v_{\alpha i} \quad \text{for } \alpha = 1, 2, \dots, d$$

PCA vs. MDS

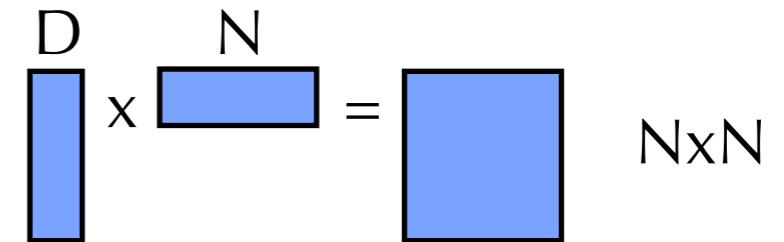
- The methods are in some sense “dual” to each other
 - In PCA, we compute the $D \times D$ covariance matrix

$$C_{ij} = \frac{1}{n} \sum_k x_{ik} x_{jk}$$



- In MDS, we compute the $N \times N$ Gram matrix

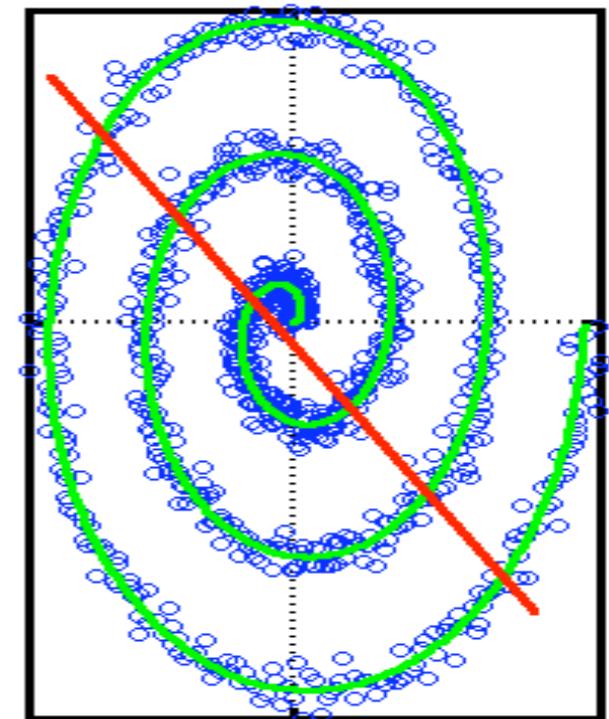
$$G_{ij} = \vec{x}_i \circ \vec{x}_j$$



- For Euclidean distances d_{ij} in MDS, the two methods yield the same embedding results (up to an arbitrary rotation)

PCA vs. MDS

- Both PCA and MDS have similar strengths
 - polynomial time algorithms (non-iterative)
 - no local optima
 - no parameters to set
 - can estimate subspace dimension
 - very well understood!
- BUT - Limited to linear projections
- **How can we generalize to arbitrary manifolds?**



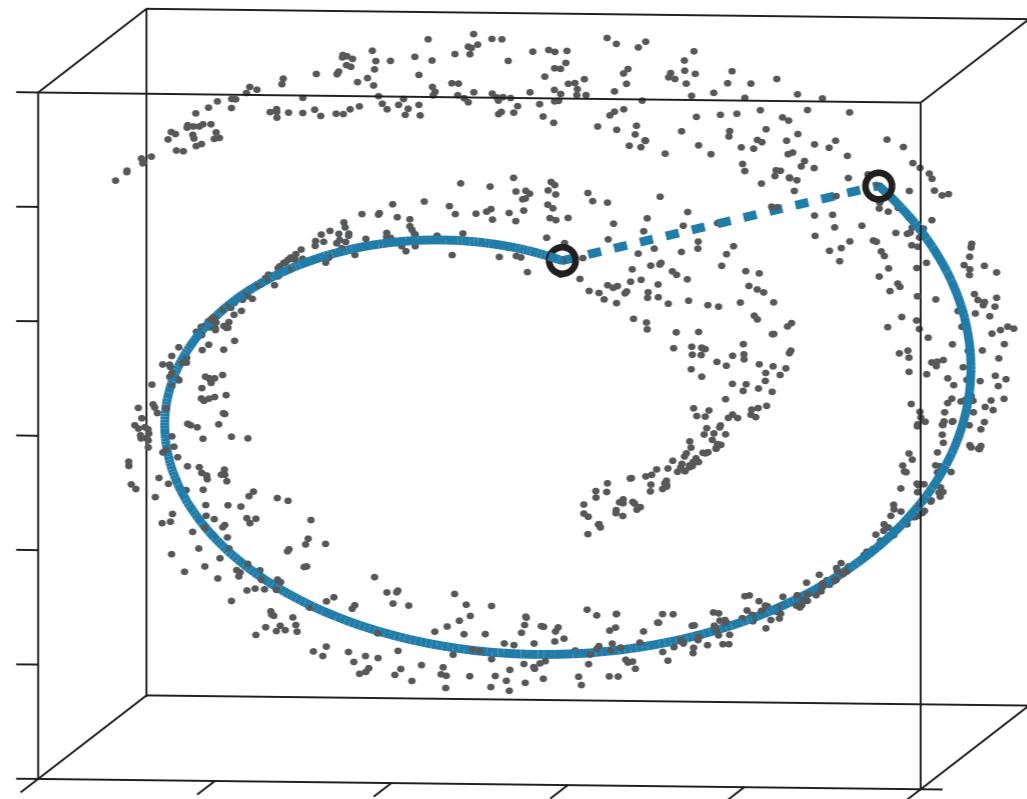
Nonlinear Dimensionality Reduction

Method 1: Isometric Feature Mapping (IsoMap)

Isometric Feature Mapping (IsoMap)

Tenenbaum et al.
(Science, Dec '00)

- Recall that MDS seeks an embedding that preserves pairwise distances between data points
- **BUT** - Geodesic distances measured on the manifold may be longer than the corresponding Euclidean straight-line distance d_{ij}

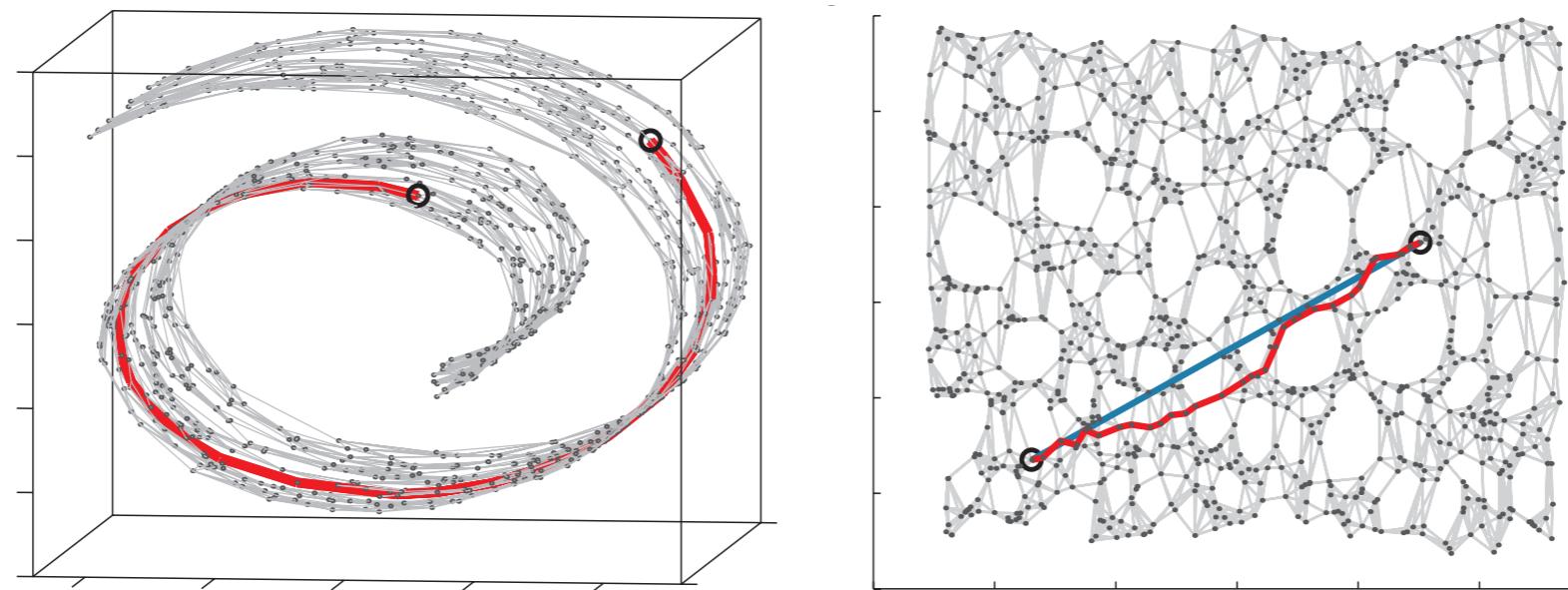


IsoMap

- Idea : Use geodesic rather than Euclidean distances in MDS
- But - How can we compute geodesics without knowing the manifold?

IsoMap

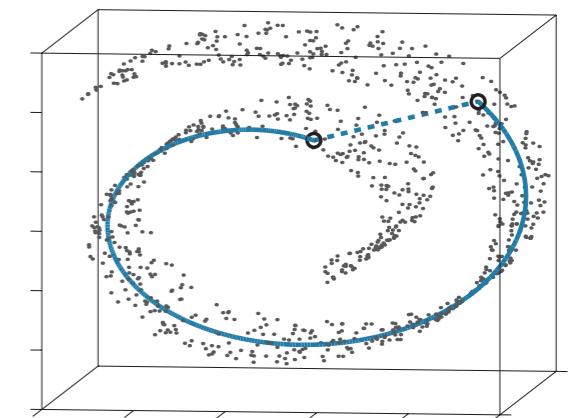
- **Idea : Use geodesic rather than Euclidean distances in MDS**
- **But - How can we compute geodesics without knowing the manifold?**



- **Answer :** Build an adjacency graph and approximate geodesic distances by shortest-paths through the graph

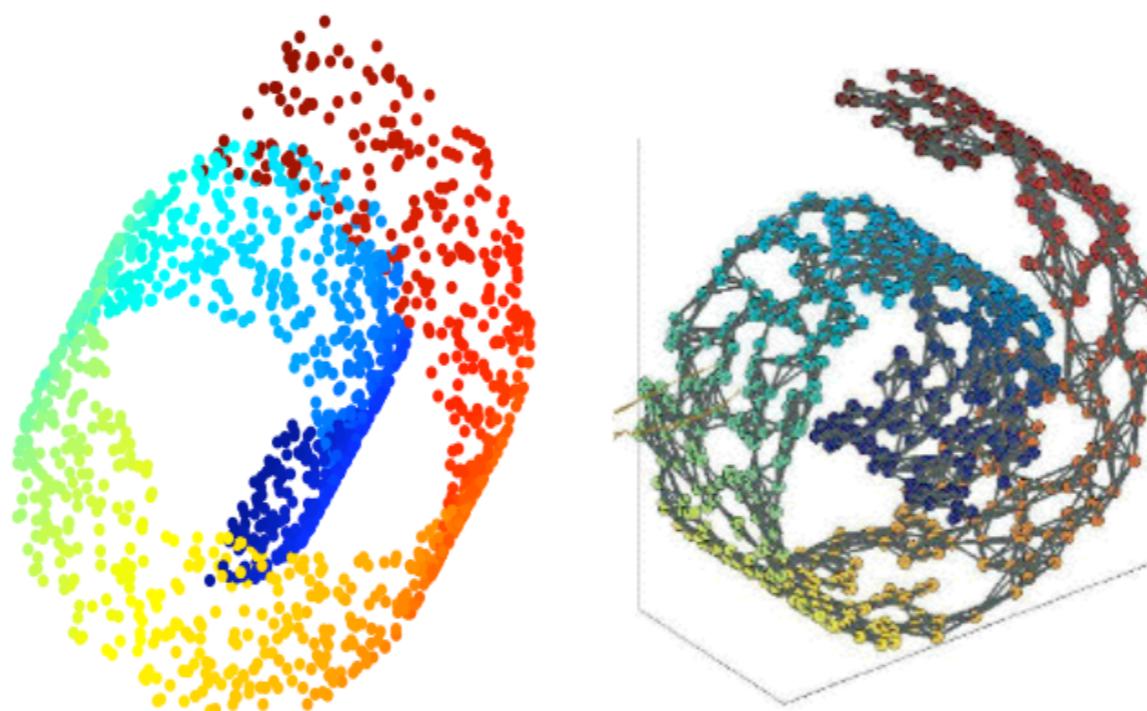
IsoMap

- **Step 1 - Build the adjacency graph over high-dim points X**
- Neighborhood selection
 - Choice 1: k-nearest neighbors
 - Choice 2: neighbors within a fixed radius (epsilon-ball)
- Assume graph is fully connected
 - no isolated islands of points
- Assume graph neighborhoods reflect manifold neighborhoods
 - no “short-cuts” between distant points on manifold
 - sensitive to choice of neighborhood size



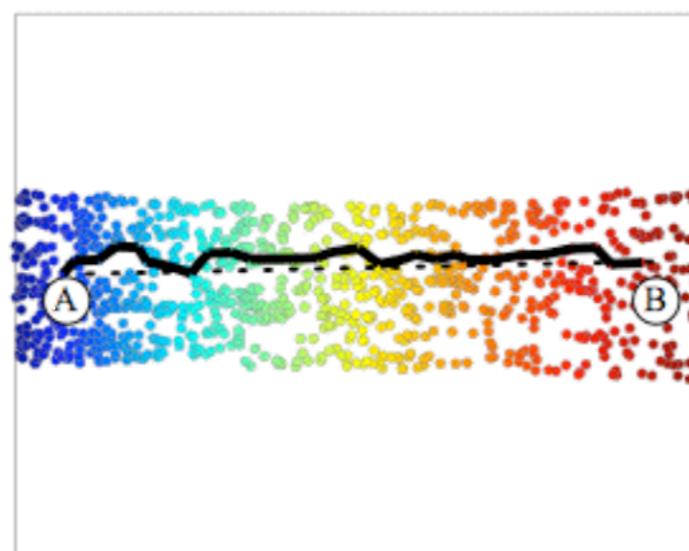
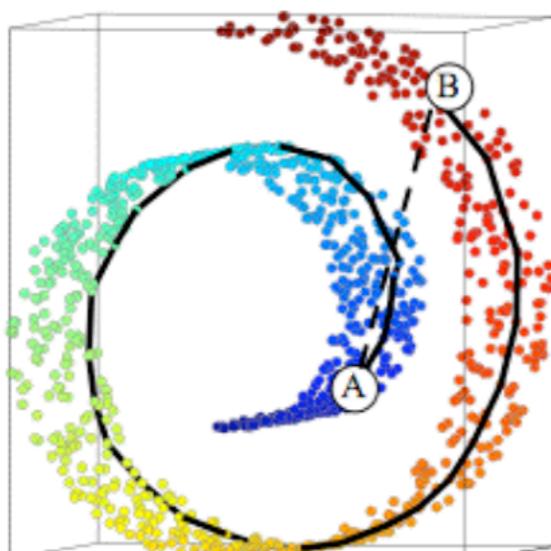
IsoMap

- Step 2 - Compute approximate geodesics
- Weight graph edges by inter-point distances
- Apply Dijkstra's all-pairs shortest-paths algorithm $O(N^2 \lg N + N^2 k)$



IsoMap

- Step 3 : Apply MDS to geodesic distances
- Top d eigenvectors of Gram matrix give the embedded, d -dimensional points
- Dimensionality of manifold may be estimated by number of significant eigenvalues, just as in PCA/MDS

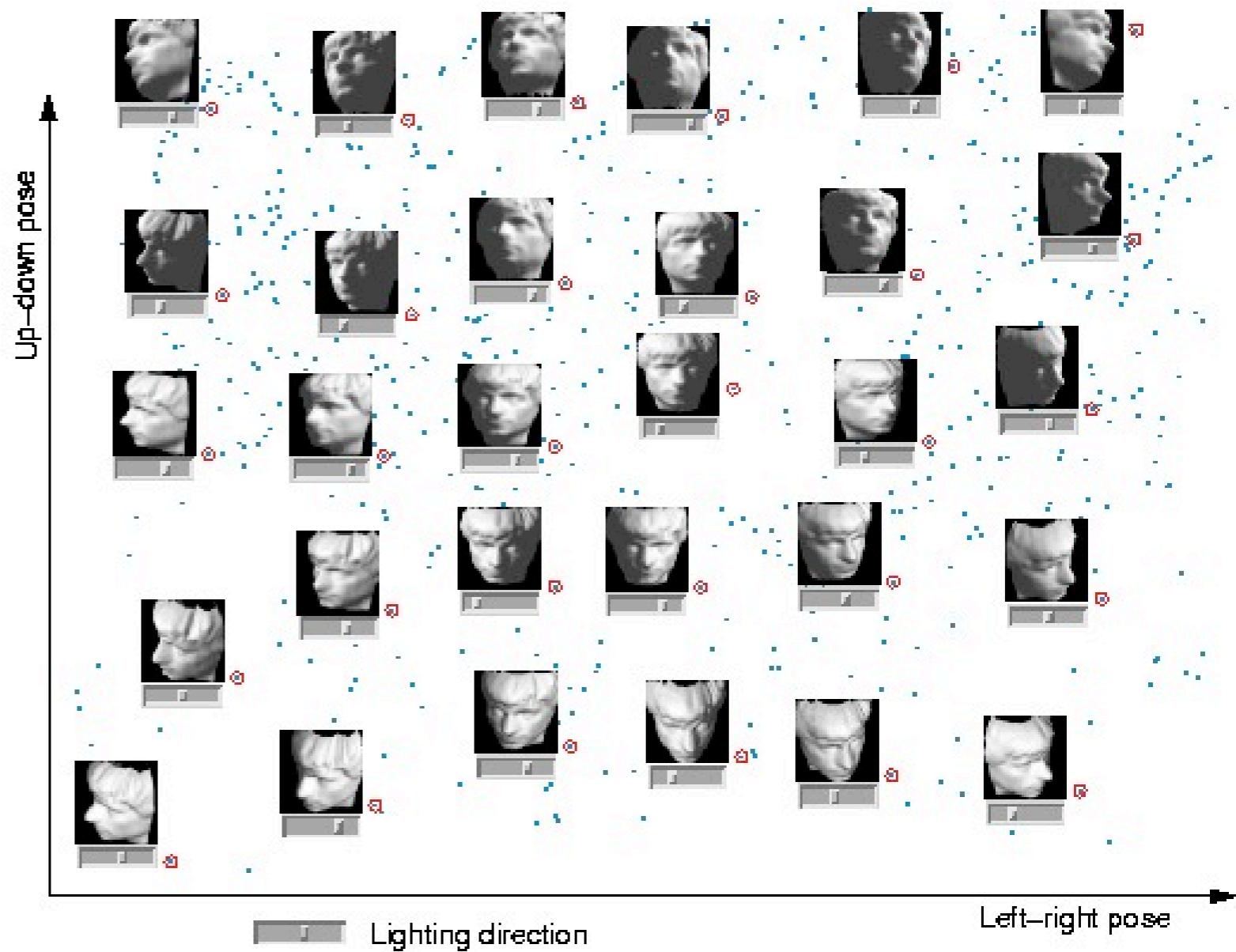
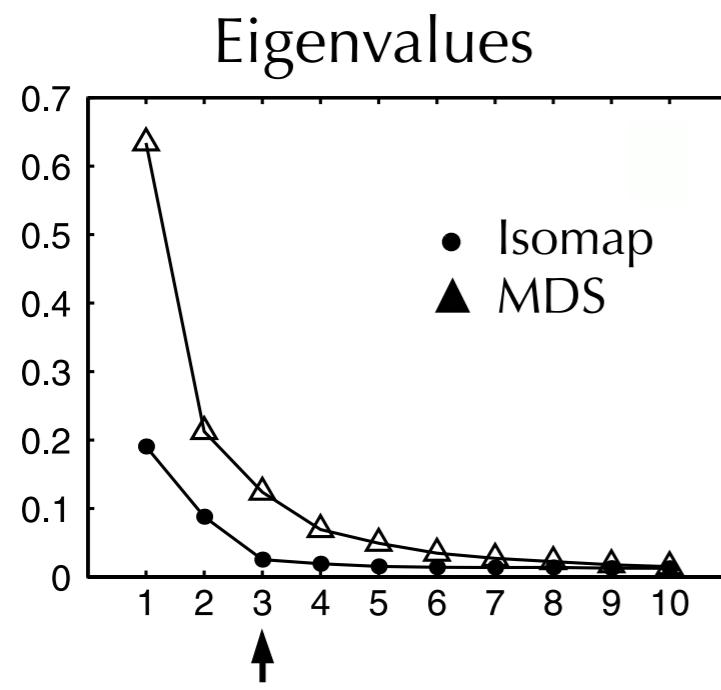


$N = 1024$ points
 $k = 12$ nearest neighbours

IsoMap examples

- Faces - varying pose and illumination
- 3 true degrees of freedom (dof) in total

- 64x64 pixel images
- $N = 698$
- $k = 6$

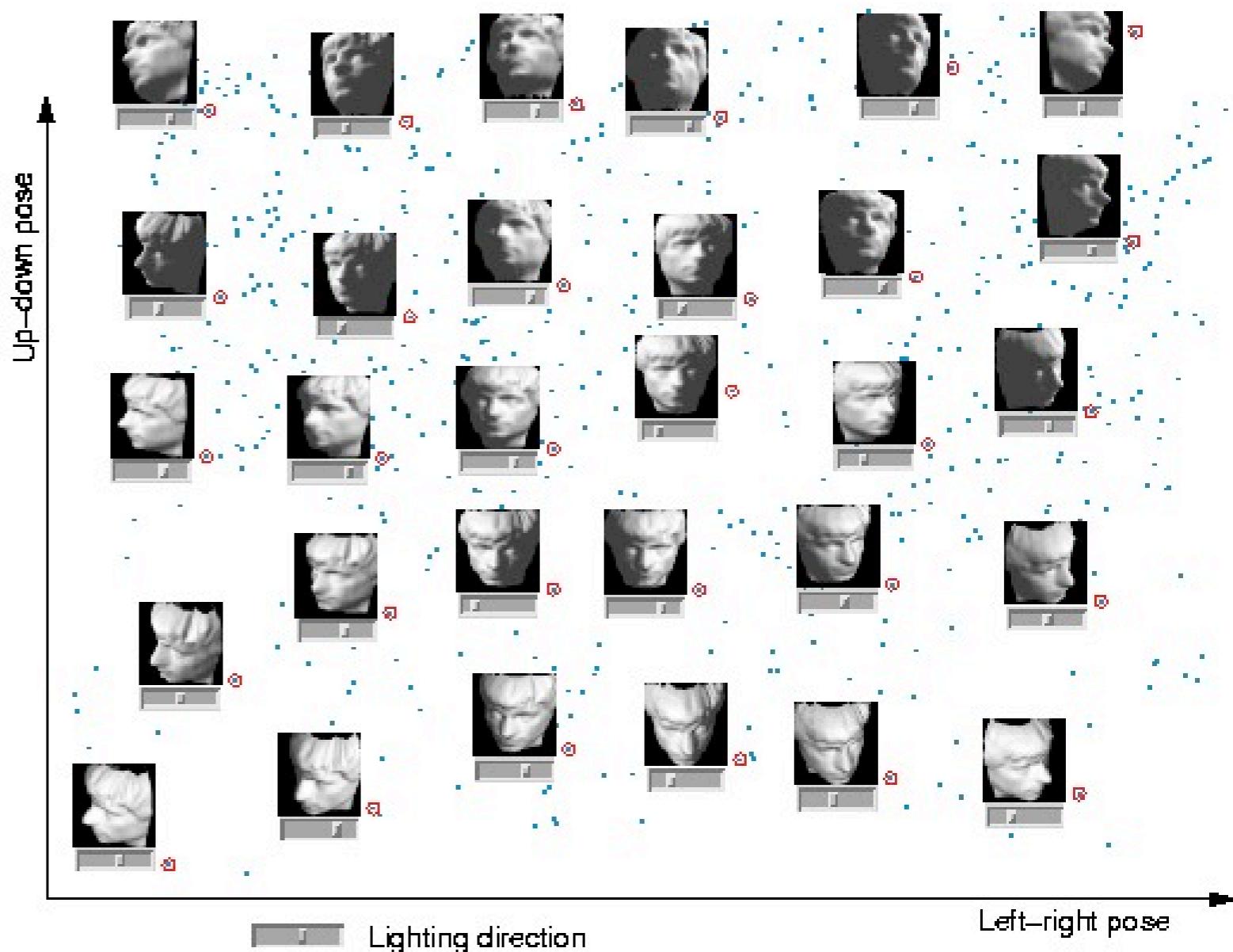


IsoMap examples

- Faces - varying pose and illumination
- 3 true degrees of freedom (dof) in total

IsoMap recovers the low-dimensional structure in the data

Coordinates in the embedding correspond to meaningful modes of variation in the image

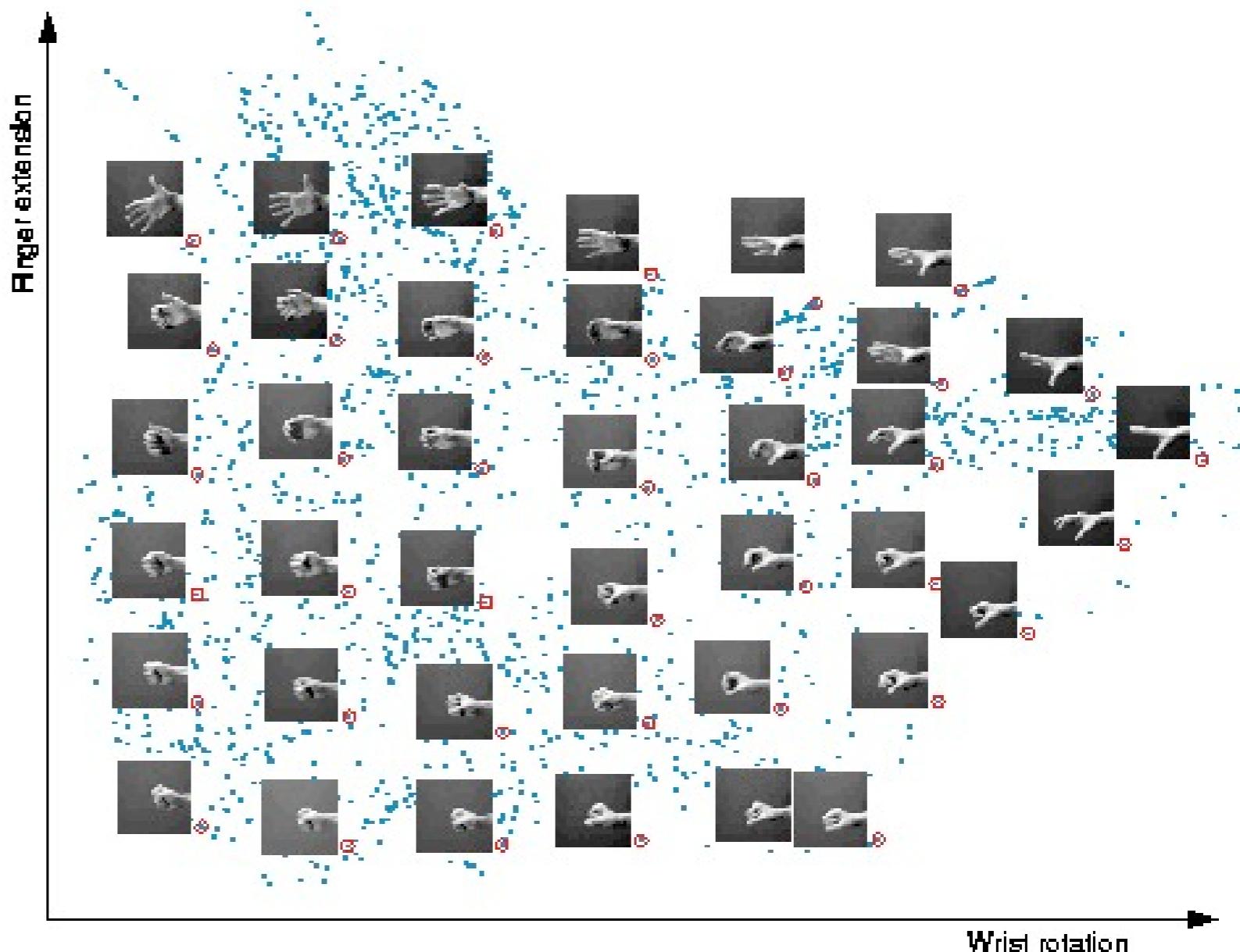


IsoMap examples

- Hand images - varying wrist rotation and finger extension

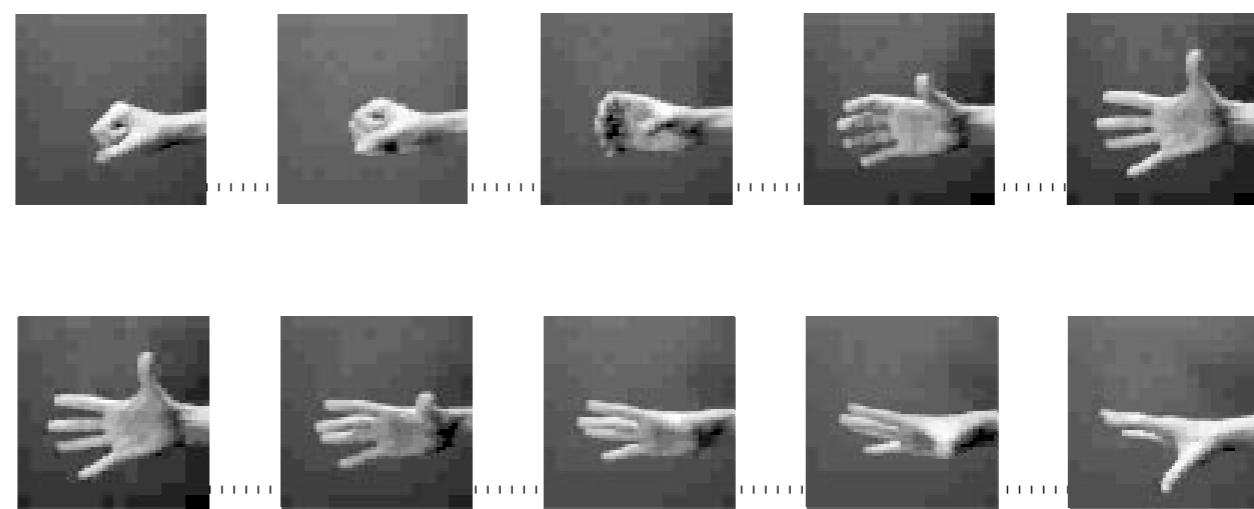
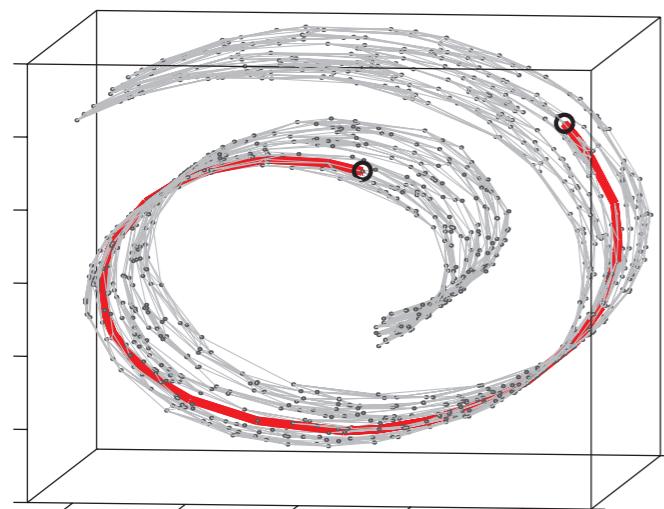
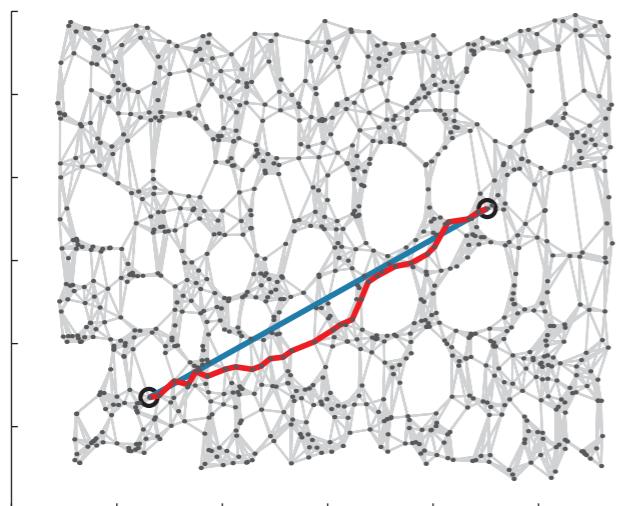
- 64x64 pixel images
- $N = 2000$
- $k = 6$

Trajectories in the embedding correspond to meaningful variations in the image



IsoMap examples

- Interpolations along “straight” lines in the embedding space yield realistic, though highly nonlinear, transitions in the image



Problem

- Isomap does not scale well
- For large N , all-pairs shortest paths computation is too expensive

Scaling-up: Landmark Isomap

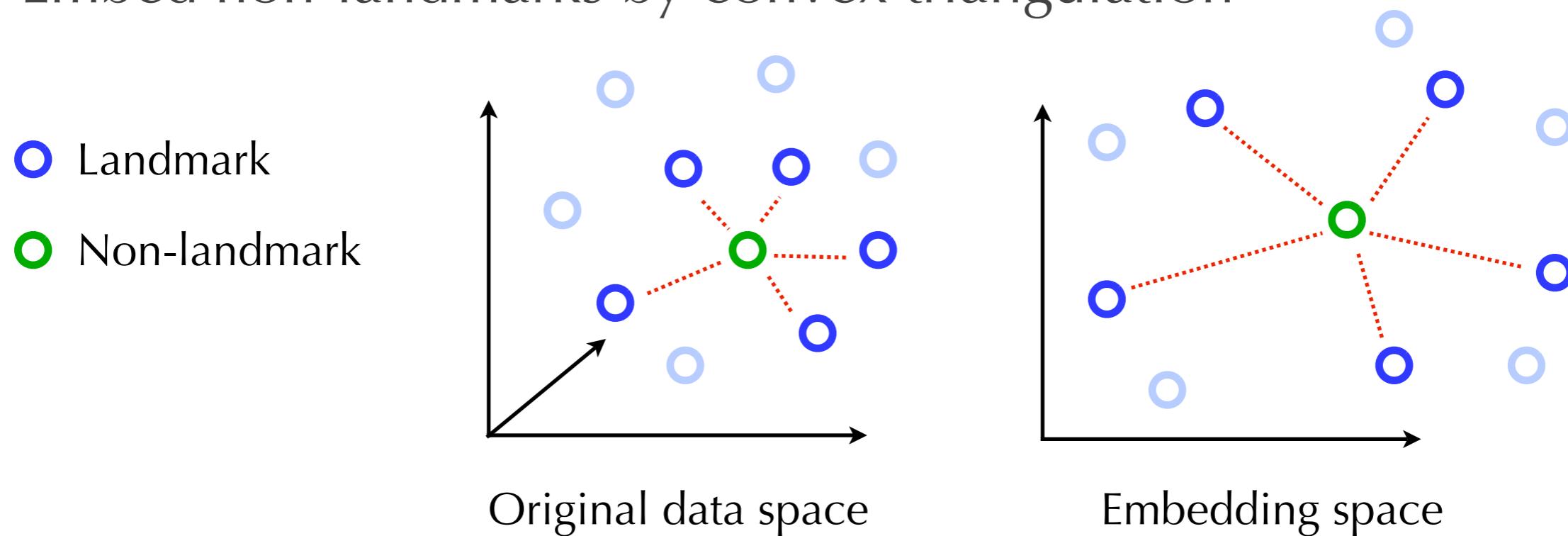
de Silva &
Tenenbaum '03

Problem

- Isomap does not scale well
- For large N , all-pairs shortest paths computation is too expensive

Solution

- Compute embedding using a subset of the data (landmarks)
- Embed non-landmarks by convex triangulation

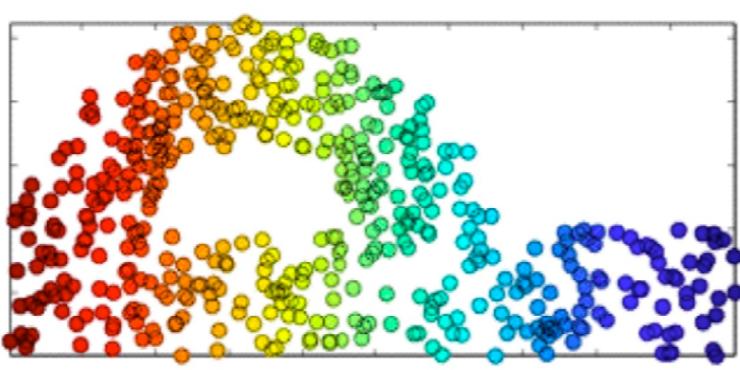
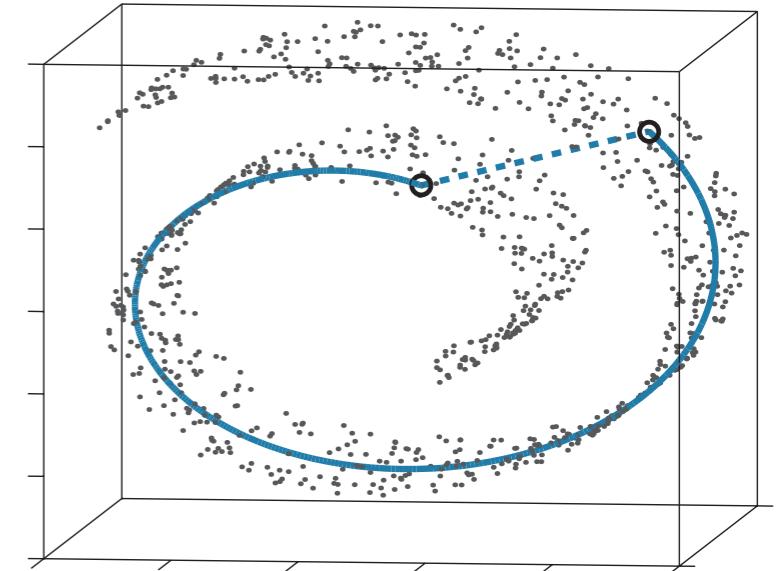


IsoMap strengths

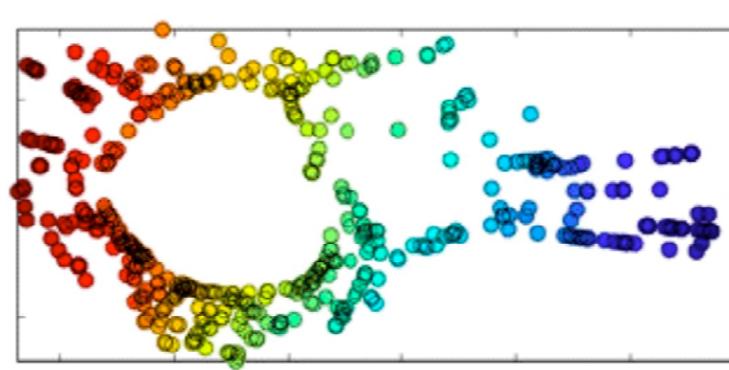
- Strengths inherited from MDS
 - Polynomial time algorithm
 - No local optima
 - Non-iterative
 - Automatic intrinsic dimensionality estimate
- Isomap adds a single heuristic parameter
 - graph neighbourhood size k
- Guaranteed asymptotic convergence
 - For data living on a convex submanifold of Euclidean space, and given large enough sample N , Isomap is guaranteed to recover the true manifold, up to a rotation and translation.

IsoMap weaknesses

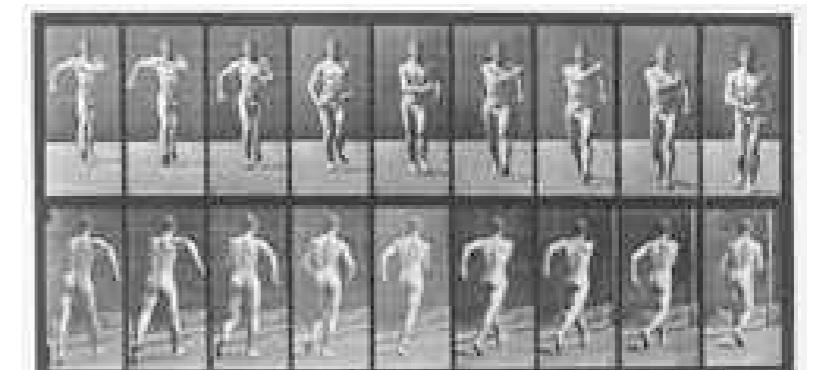
- Sensitive to “short-cuts” due to k being too large
- Does not scale well to very large N
 - $N \times N$ dense eigenvector problem is expensive
- Convexity assumption
 - Cannot handle manifolds with “holes”



Input



IsoMap embedding



e.g. periodic motion

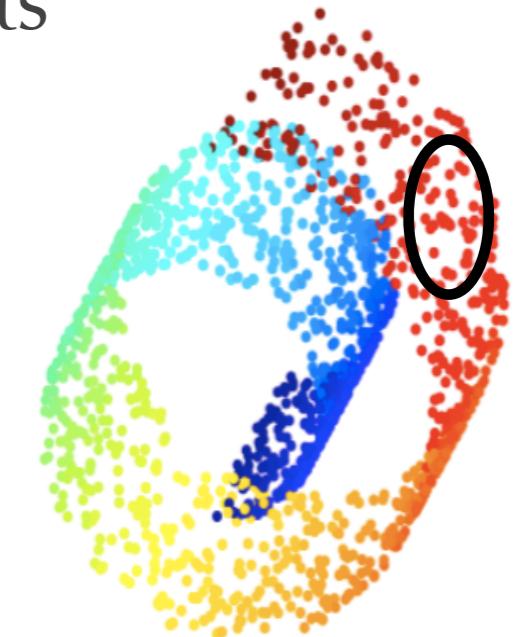
Nonlinear Dimensionality Reduction

Method 2: Locally Linear Embedding

Locally Linear Embedding (LLE)

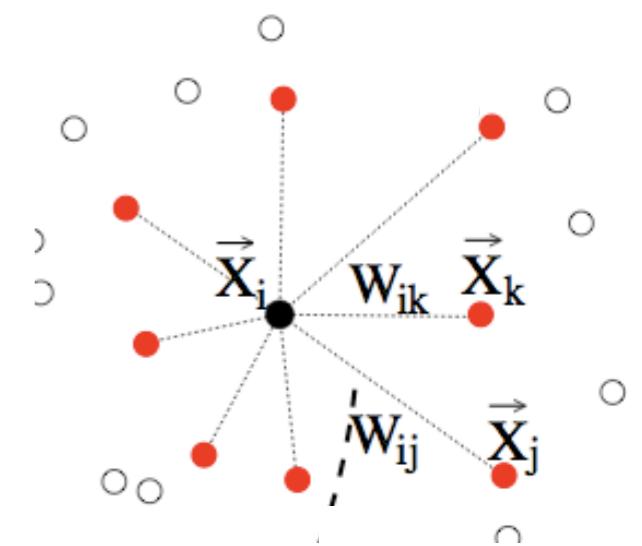
Roweis & Saul,
Science, Dec '00

- “Think locally, fit globally!” - an alternative to Isomap
- LLE aims to preserve local manifold geometry in its embedding



Idea

- Assume manifold is locally linear
 - We expect each D-dim data point to lie on or near a locally linear patch of the manifold
- Characterize each point x_i as a convex linear combination of its k-nearest neighbors x_j
- Seek an embedding that preserves these weights

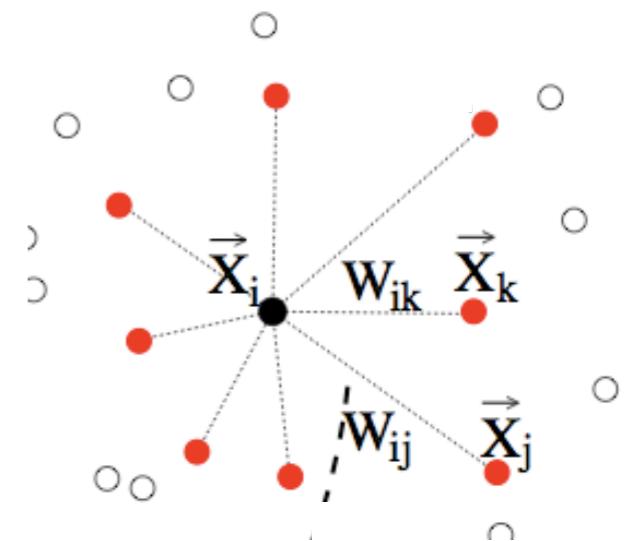


Locally Linear Embedding

- **Step 1:** Compute k-nearest neighbors for each point x_i
 - Same as in Isomap
- **Step 2:** Compute weights W_{ij} that best reconstruct x_i as a convex sum of its neighbors x_j

$$\arg \min_W \Phi(W) = \sum_i \left\| \vec{x}_i - \sum_{j \in \mathcal{N}_i} W_{ij} \vec{x}_j \right\|^2$$

subject to $\sum_j W_{ij} = 1$



- This is easily solved using a Lagrange multiplier
- Note that local weights are invariant to *translation, rotation and scale*
- Hence weights should be preserved under a well-behaved embedding

Locally Linear Embedding

- **Step 3:** Choose embedded coordinates y_i that minimize reconstruction error using previously computed weights W_{ij}

$$\arg \min_{\vec{y}} \Theta(\vec{y}) = \sum_i \left\| \vec{y}_i - \sum_{j \in \mathcal{N}_i} W_{ij} \vec{y}_j \right\|^2$$

subject to $\sum_i y_i = 0$ (zero mean)

$$\frac{1}{N} \sum_i y_i y_i^\top = I_d \quad (\text{unit covariance})$$

- Since the embedding is only defined up to an arbitrary translation and scale, the constraints serve to make the problem well-posed

Locally Linear Embedding

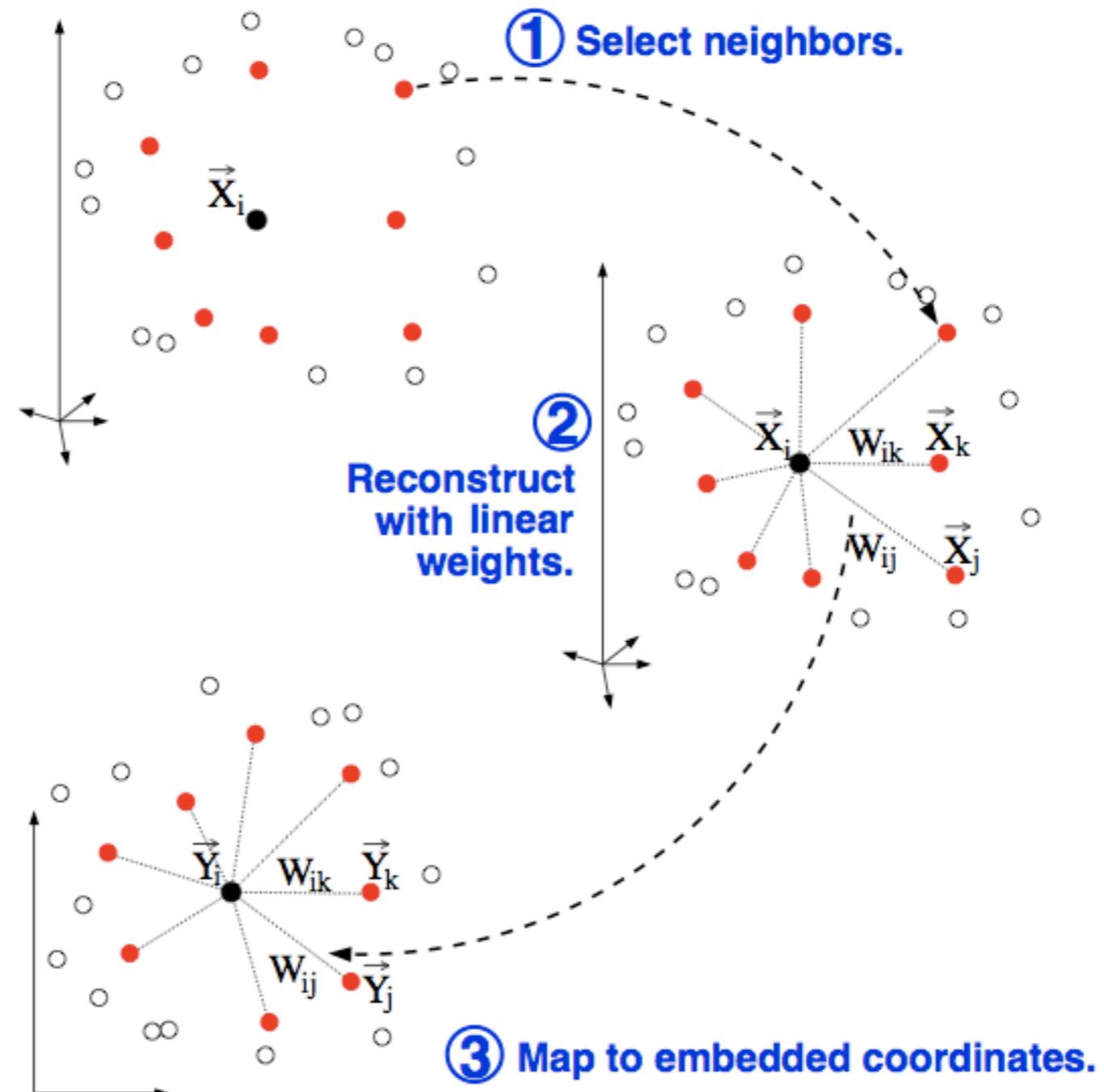
- The result is given by the eigenvectors of the matrix Q corresponding to the $d+1$ smallest eigenvalues, where

$$Q = (I - W)^\top (I - W)$$

- The bottom eigenvector is the vector $[1 \ 1 \ 1 \ 1]^\top$, an exact null-vector corresponding to a free translation mode.
- Discarding it imposes the zero-mean constraint.
- The remaining d eigenvectors give the embedding
- **Note :** W and hence Q is very sparse (compare to IsoMap G)
- Efficient algorithms exist for large, sparse eigenvector problems

LLE summary

1. Compute the neighbors of each data point, \vec{X}_i .
2. Compute the weights W_{ij} that best reconstruct each data point \vec{X}_i from its neighbors, minimizing the cost in eq. (1) by constrained linear fits.
3. Compute the vectors \vec{Y}_i best reconstructed by the weights W_{ij} , minimizing the quadratic form in eq. (2) by its bottom nonzero eigenvectors.

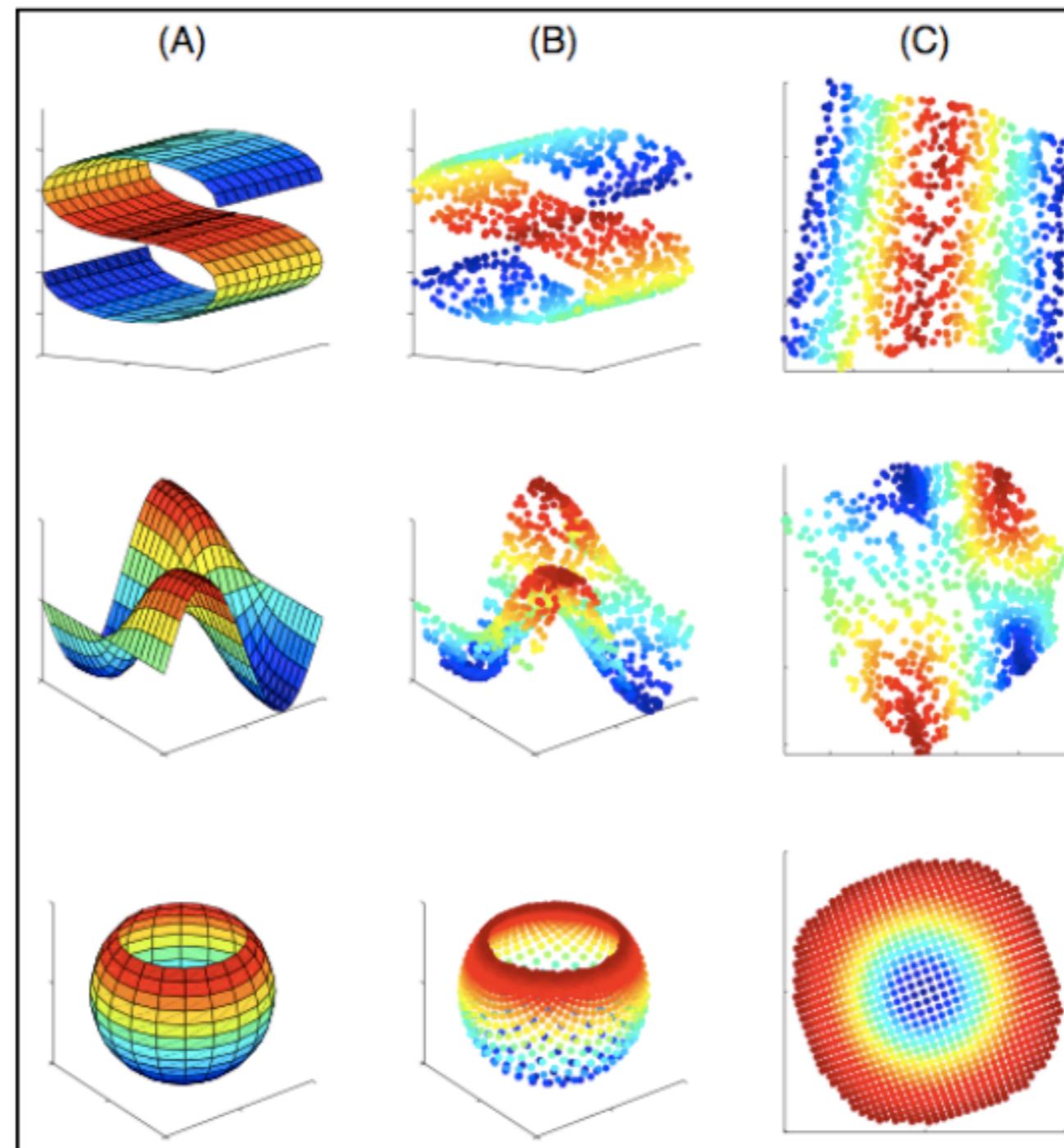


LLE examples

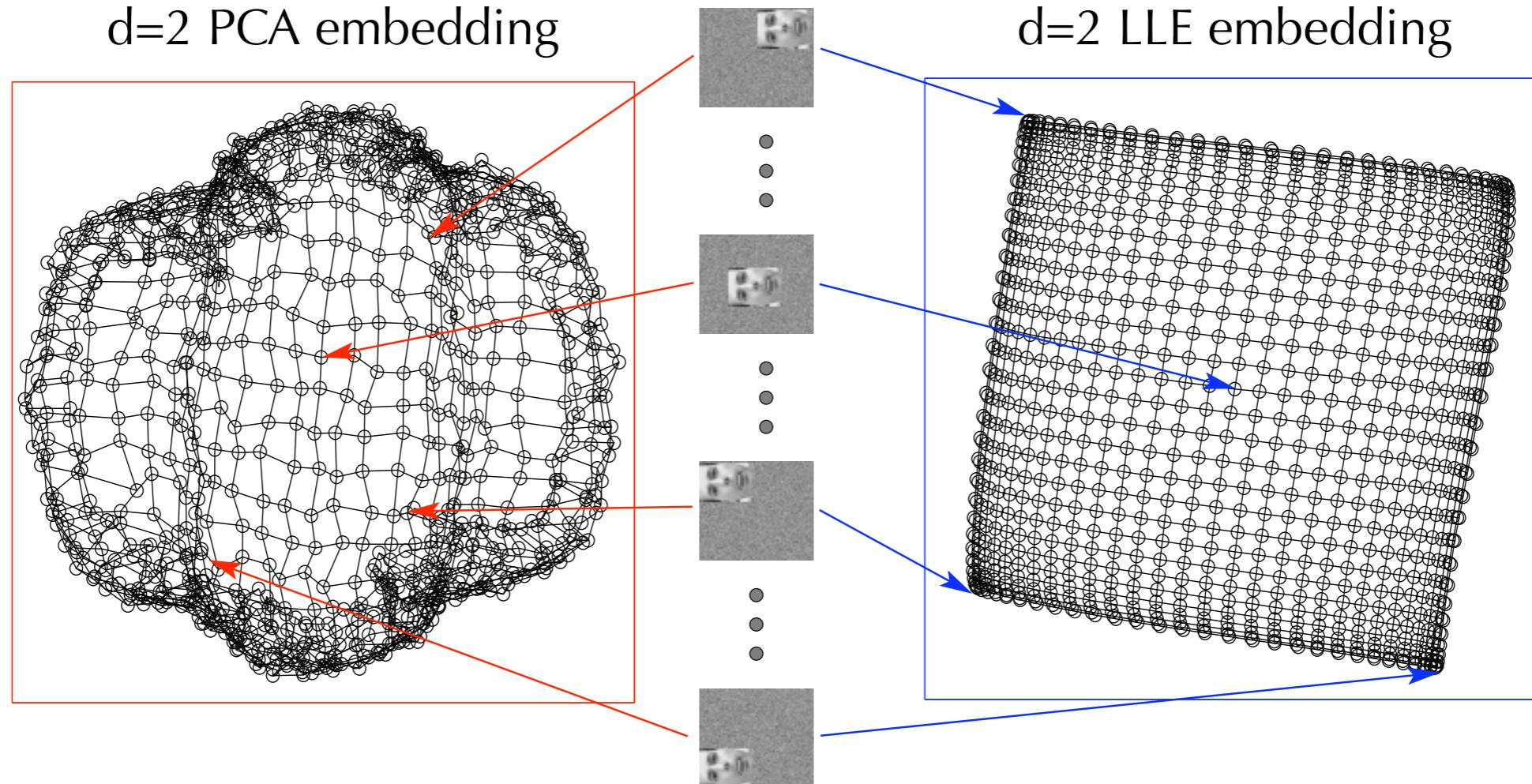
**N=1000
inputs**

**k=8
nearest
neighbors**

**D=3
d=2
dimensions**



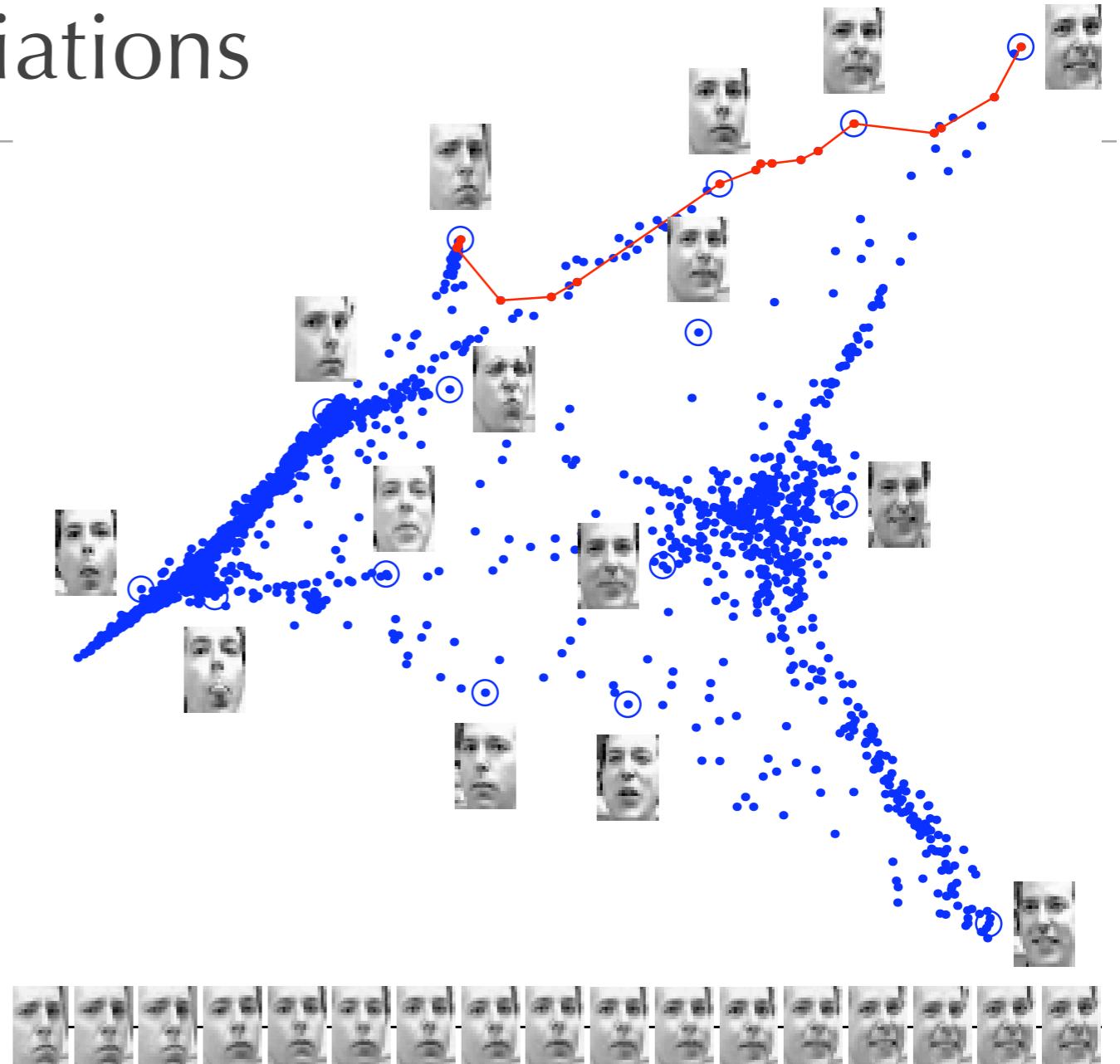
PCA vs LLE example



- Input: 30x30 images of a translating face ($N=961$)
- PCA fails to recover a meaningful 2-d embedding
- LLE discovers the 2 translational degrees of freedom in the input

LLE example - Face variations

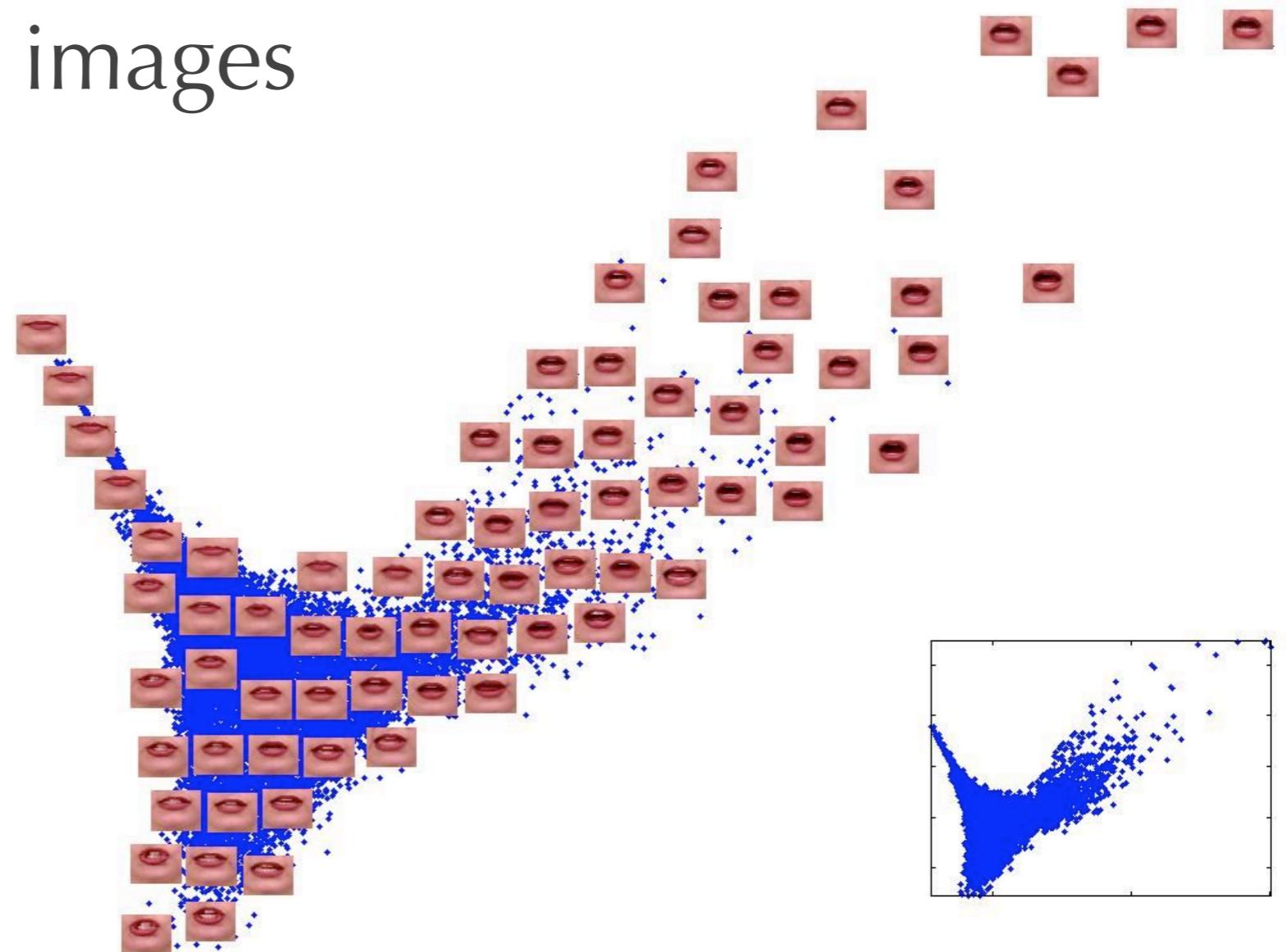
- 20x28 pixel images
- N=1965
- k=12
- d=2



- The 2-d LLE embedding coordinates correspond roughly to variations in pose and expression
- The trajectory (red) corresponds to a realistic facial transition (bottom row)

LLE example - Lips images

- 256x256 pixel images
- $N=15960$
- $k=24$
- $d=2$

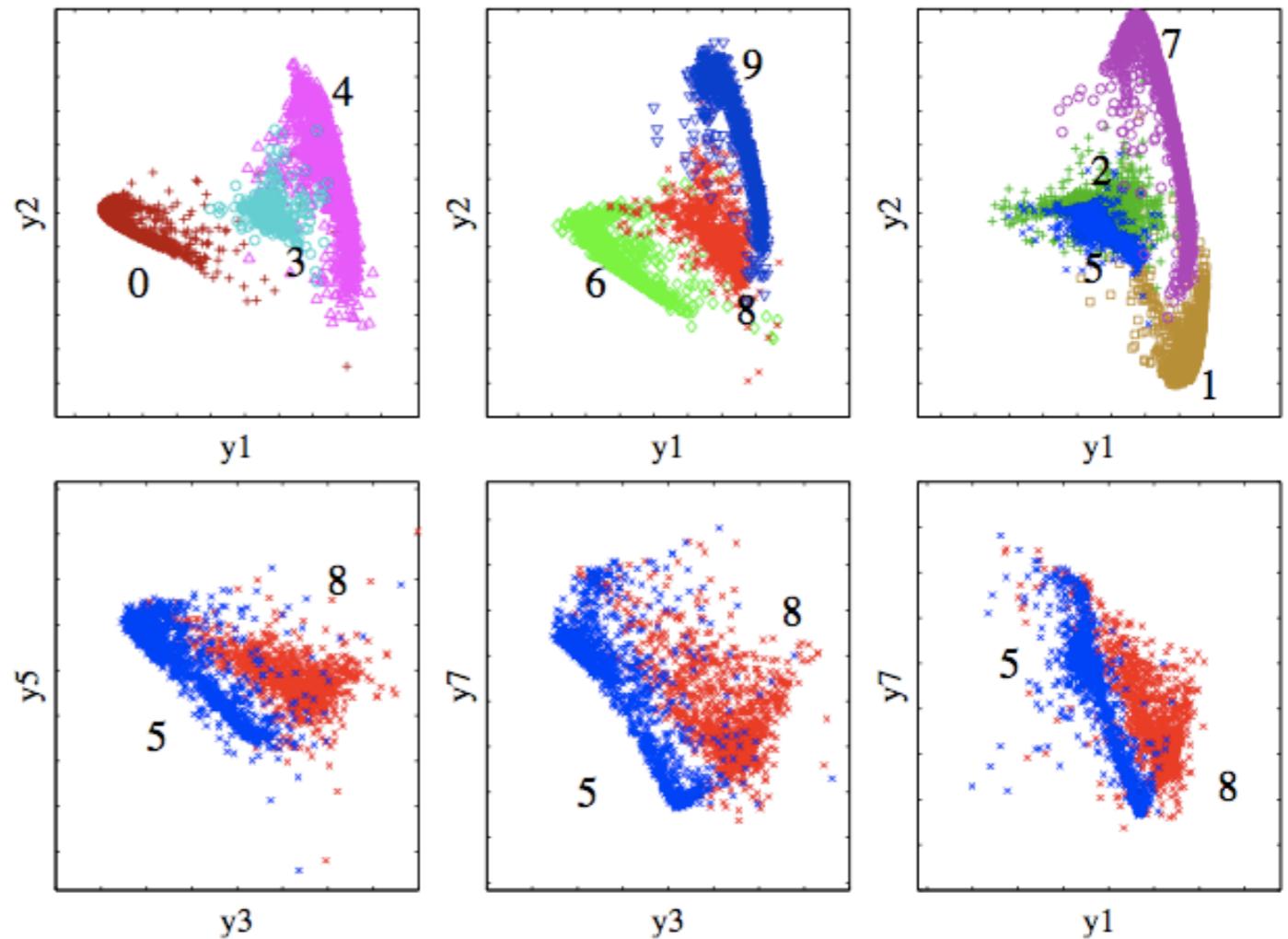


- Trajectories in the 2-d embedding correspond to smooth variations in the mouth configuration
- Note: LLE easily handles the large problem size ($N=15960$) thanks to sparse weights matrix

LLE example - a pattern classifier

- Recognition of hand-written digits

- 16x16 pixel images (USPS dataset)
- $N=11000$
- $k=??$ (author doesn't say)
- $d=8$



- Most digit classes are easily separable in just the first two embedding dimensions
- A classifier would be easy to construct and visualize

LLE with pairwise distances

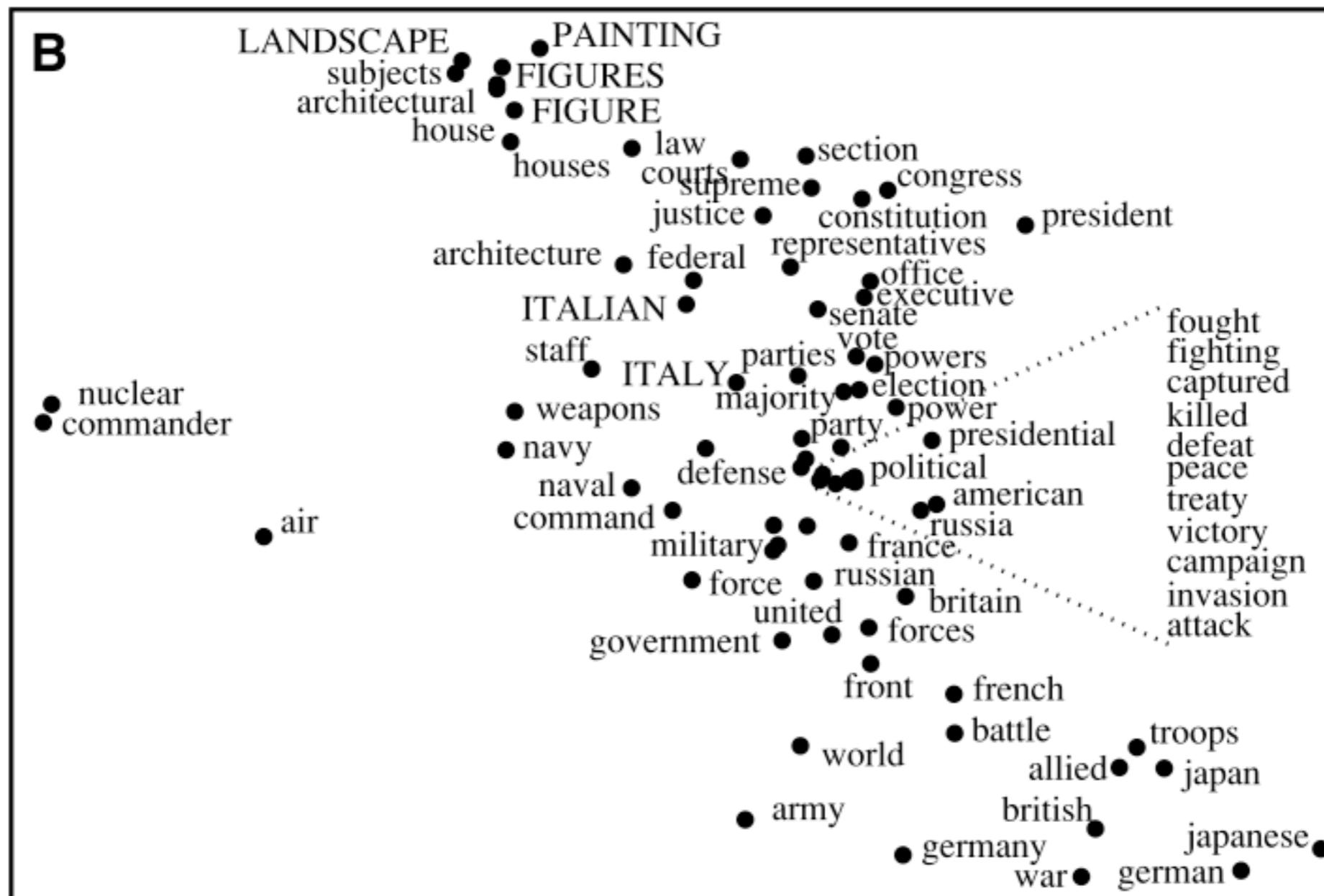
- What if we only have pairwise distances $d(X_i, X_j)$ between data points, as was the case with MDS and IsoMap?
- We can use the same trick for expressing dot products in terms of distances when computing the LLE weights W_{ij}
- The neighborhood covariance may be written as

$$C_{jk} = \frac{1}{2} (D_j + D_k - D_{jk} - D_0)$$

where $D_\ell = \sum_z \tilde{D}_{\ell z}$

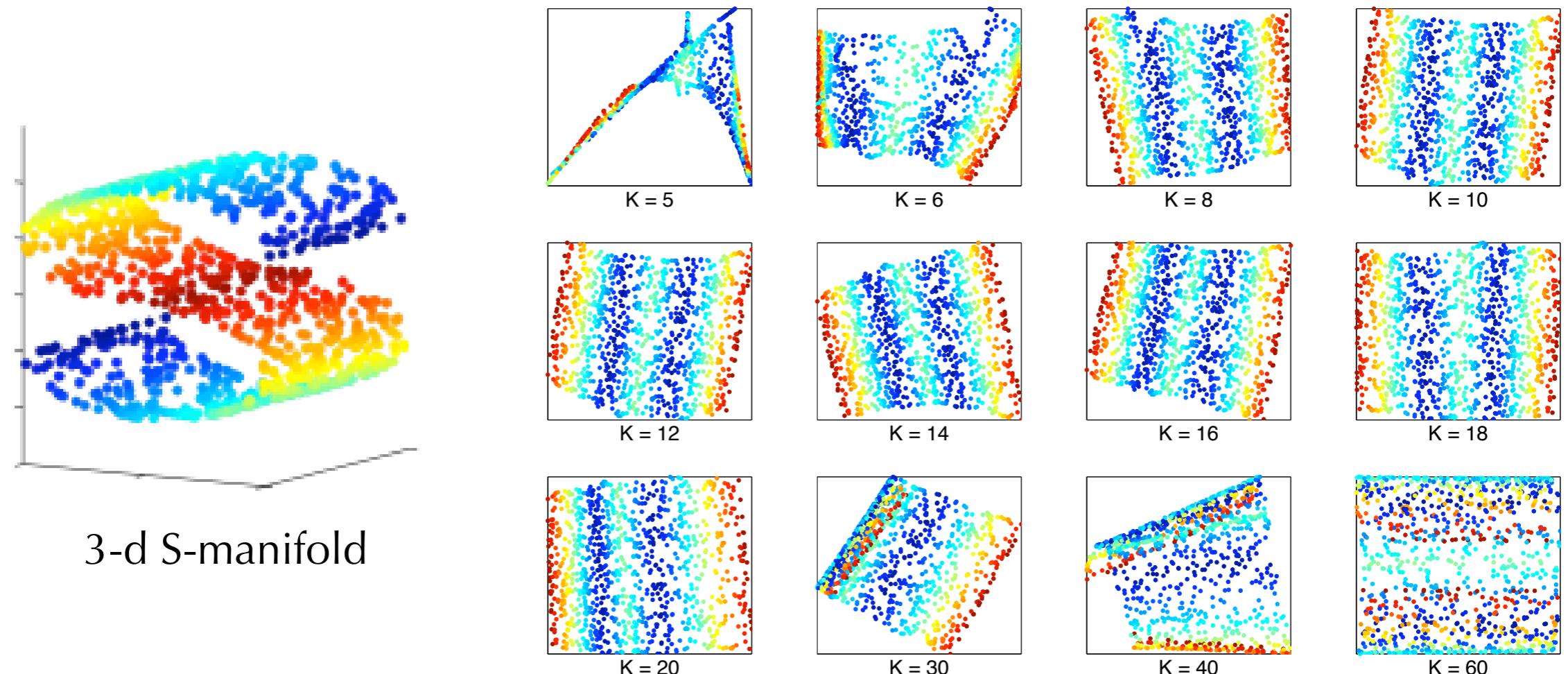
$$D_0 = \sum_{jk} D_{jk}$$

LLE with pairwise distances



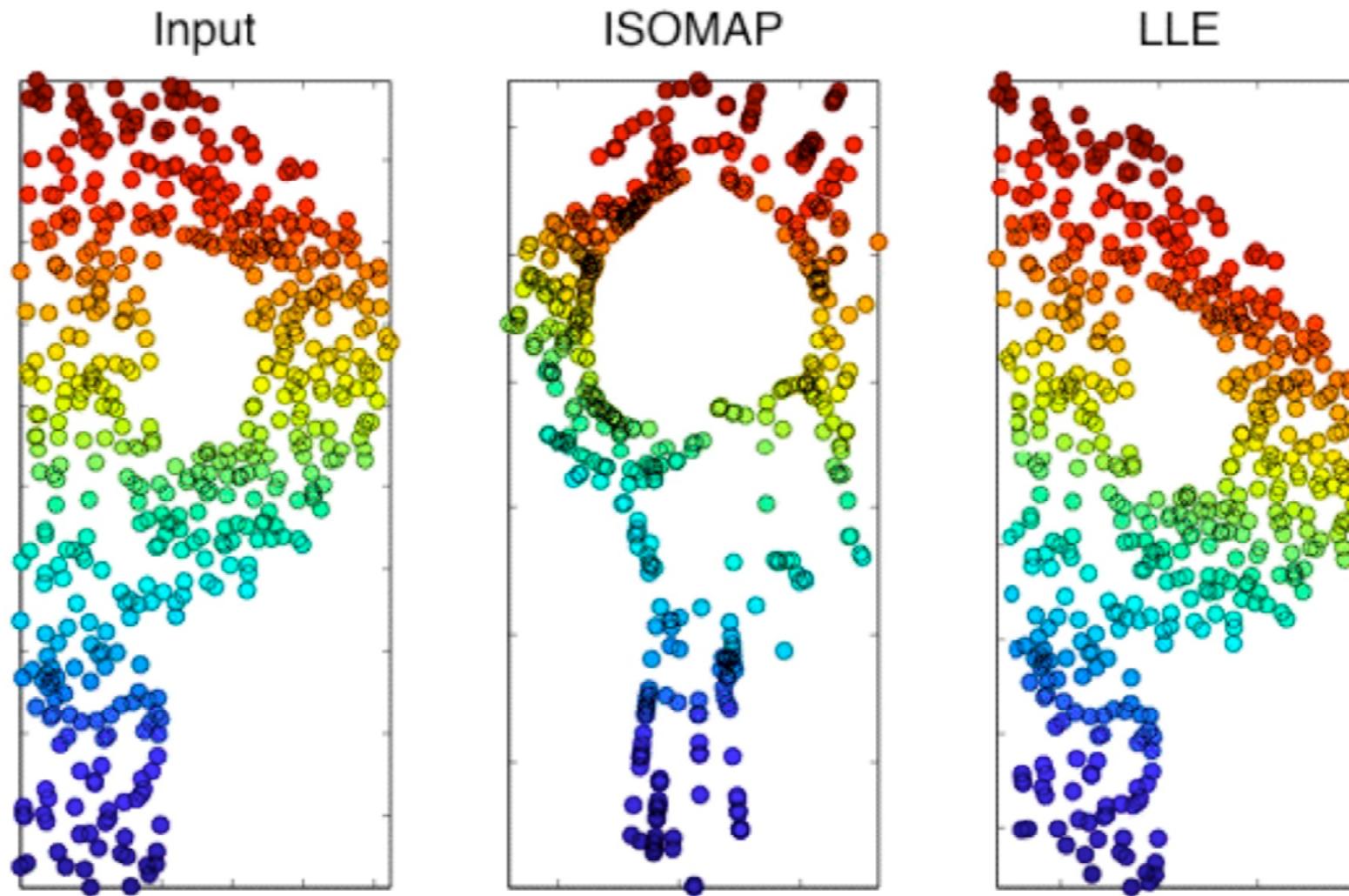
- Input: Histograms of occurrence of 5000 words in 31000 encyclopedia articles
- Distance metric: dot-products between unit-normalized histograms
- k=20
- LLE recovers a continuous semantic embedding

LLE: choosing neighborhood size k



- Neighborhood size k is varied in 2-d embedding of S-manifold
- k too low - no meaningful structure is recovered
- k too high - S is squashed onto a plane, ordering not preserved

LLE: Non-convex manifolds



- LLE handles non-convex manifolds (those with holes) a little better than IsoMap
- Not perfect - we'd prefer this particular 2d-2d embedding to be a simple isometry!

LLE strengths/weaknesses

- Similar strengths to IsoMap
 - Graph-base, eigenvector method
 - Polynomial time algorithm
 - No local optima
 - Non-iterative
 - Single heuristic parameter (neighbourhood size k)
- PLUS - Better handling of non-convex manifolds
- BUT - some additional weaknesses
 - Also sensitive to “short-cuts”
 - No asymptotic guarantees
 - No way to estimate intrinsic manifold dimension

IsoMap vs. LLE

IsoMap

- Computes top **d** eigenvectors of a **dense** NxN matrix
- Preserves distances
- Asymptotic guarantee of finding true manifold

LLE

- Computes bottom **d+1** eigenvectors of a **sparse** NxN matrix
- Preserves local linear geometry
- Copes with “holes” rather better

Major “selling point” for LLE :

- LLE avoids the need to compute a dense, all-pair shortest distance matrix
- The LLE eigenvector problem is extremely sparse
- Far more efficient in terms of both time and storage requirements

Application : Clever graphics stuff!

Unwrap Mosaics: A New Representation for Video Editing

Rav-Acha, Kohli, Rother, Fitzgibbon
SIGGRAPH 2008

<http://research.microsoft.com/unwrap/>

Recent advances and further reading

- **Linear methods**
 - Principal components analysis (PCA) finds maximum variance subspace.
 - Metric multidimensional scaling (MDS) finds distance-preserving subspace.

- **Graph-based methods**

