Spectral Methods for Dimensionality Reduction

Prof. Lawrence Saul

Dept of Computer & Information Science

University of Pennsylvania

UCLA IPAM Tutorial, July 11-14, 2005

Penn
UNIVERSITY of PENNSYLVANIA

Institute for Pure & Applied Mathematics
IPAM
University of California, Los Angeles

Nonlinear dimensionality reduction

Given high dimensional data sampled from a low dimensional submanifold, how to compute a faithful embedding?

Notation

- **Inputs (high dimensional)**

  $$\vec{x}_i \qquad {}^{D} \text{ with } i = 1, 2, \ldots, n$$

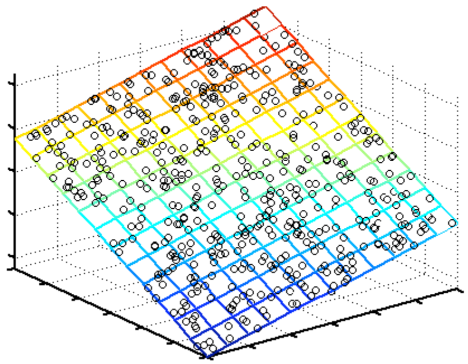
- **Outputs (low dimensional)**

  $$\vec{y}_i \qquad {}^{d} \text{ where } d \ll D$$

- **Goals**

  Nearby points remain nearby.
  Distant points remain distant.
  (Estimate *d*.)

Linear vs nonlinear



What computational price must we pay for nonlinear dimensionality reduction?

Yesterday

- **Linear methods**
  - **principal components analysis (PCA)**
  - **metric multidimensional scaling (MDS)**
- **Isomap**
  1. **k-nearest neighbors**
  2. **Shortest paths through graph**
  3. **MDS on geodesic distances**

A nonlinear method with most of the advantages of linear ones…

Yesterday vs today

- **MDS and Isomap**
  - preserve global pairwise distances
  - construct large, dense matrices
  - compute top eigenvectors
- **“Local” methods**
  - preserve local geometric relationships
  - construct large, sparse matrices
  - compute bottom eigenvectors

Algorithms

2000
Isomap
(Tenenbaum, de Silva, & Langford)

Locally Linear Embedding
(Roweis & Saul)

2002
Laplacian eigenmaps
(Belkin & Niyogi)

2003
Hessian LLE
(Donoho & Grimes)

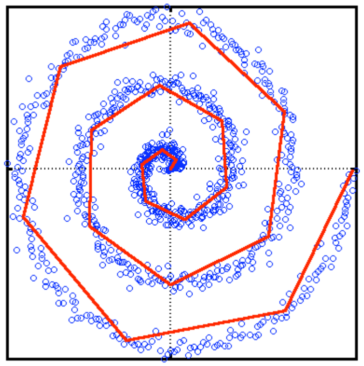
2004
Maximum variance unfolding
(Weinberger & Saul)
(Sun, Boyd, Xiao, & Diaconis)

2005
Conformal eigenmaps
(Sha & Saul)

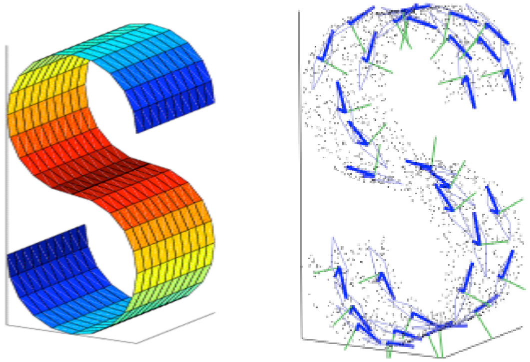
Questions for today

- **How to exploit local linearity?**

Manifolds are globally nonlinear, but locally linear.



- **Isn't this an old idea?**
  - k-means, k-subspaces
  - mixture models
  - self-organizing maps



How (not) to use local linearity

iterative clustering,
subspace quantization

k-means clustering

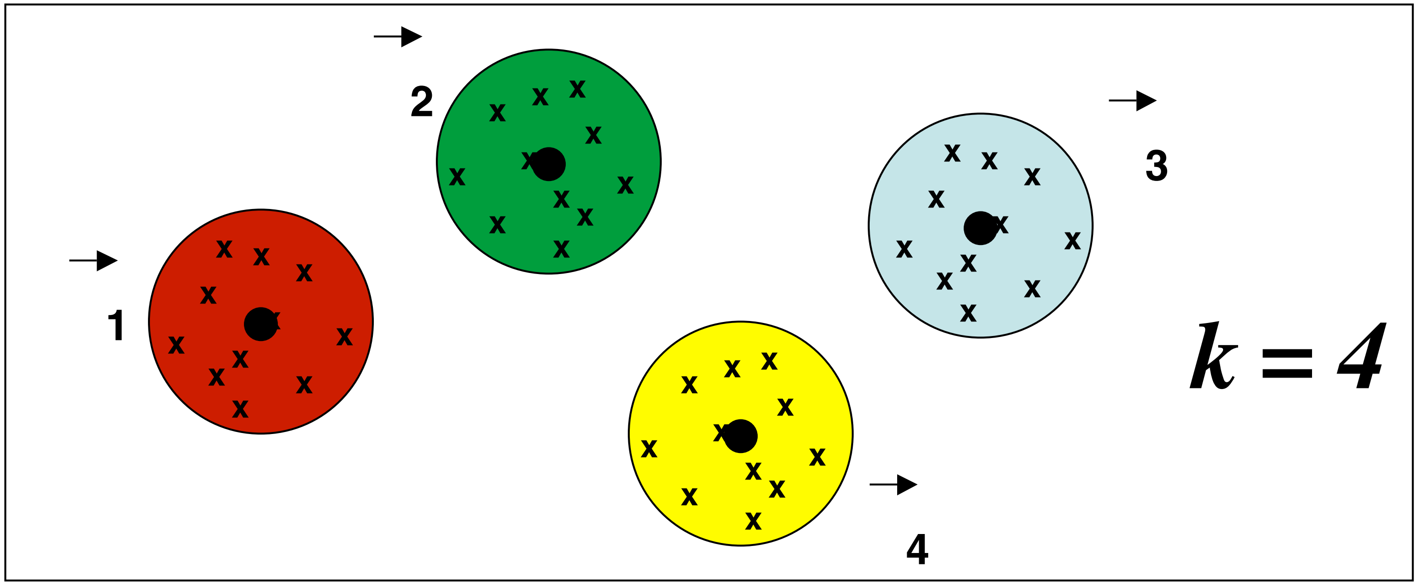
- **Goal**

  Map each continuous input $\vec{x}_i$ to a discrete label y_i $\{1, 2, ..., k\}$.

- **Algorithm**

  1. **Randomly choose k “centroids” $\vec{\ }$.**
  2. **Set $y_i = \text{argmin} \; ||\vec{x}_i - \vec{\ } \; ||$.**
  3. **Set $\vec{\ }$ to mean of inputs with $y_i =$.**
  4. **Iterate steps 2-3 until convergence.**

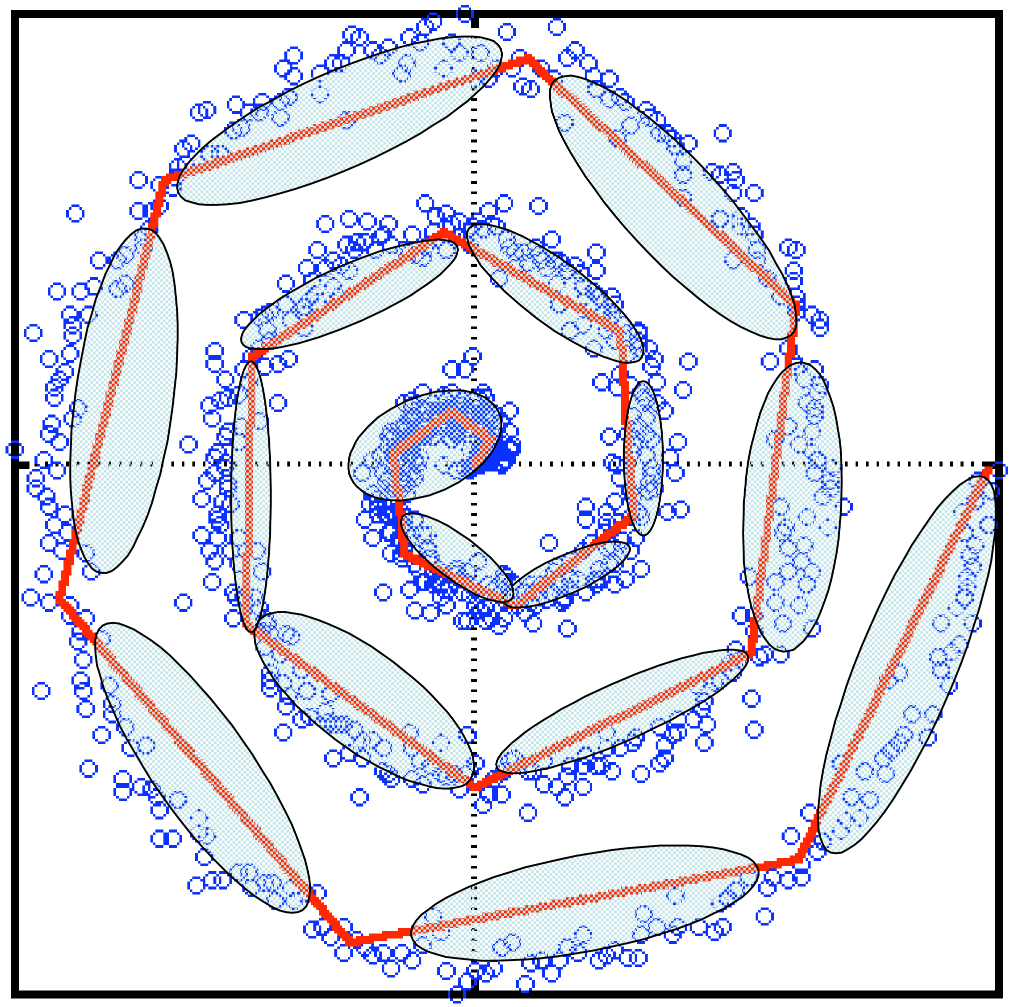
k-means clustering



- **Generalizations**
  - ellipsoidal vs spherical clusters
  - unbalanced vs balanced clusters
  - soft (probabilistic) assignment
  - k-lines, k-planes, k-subspaces

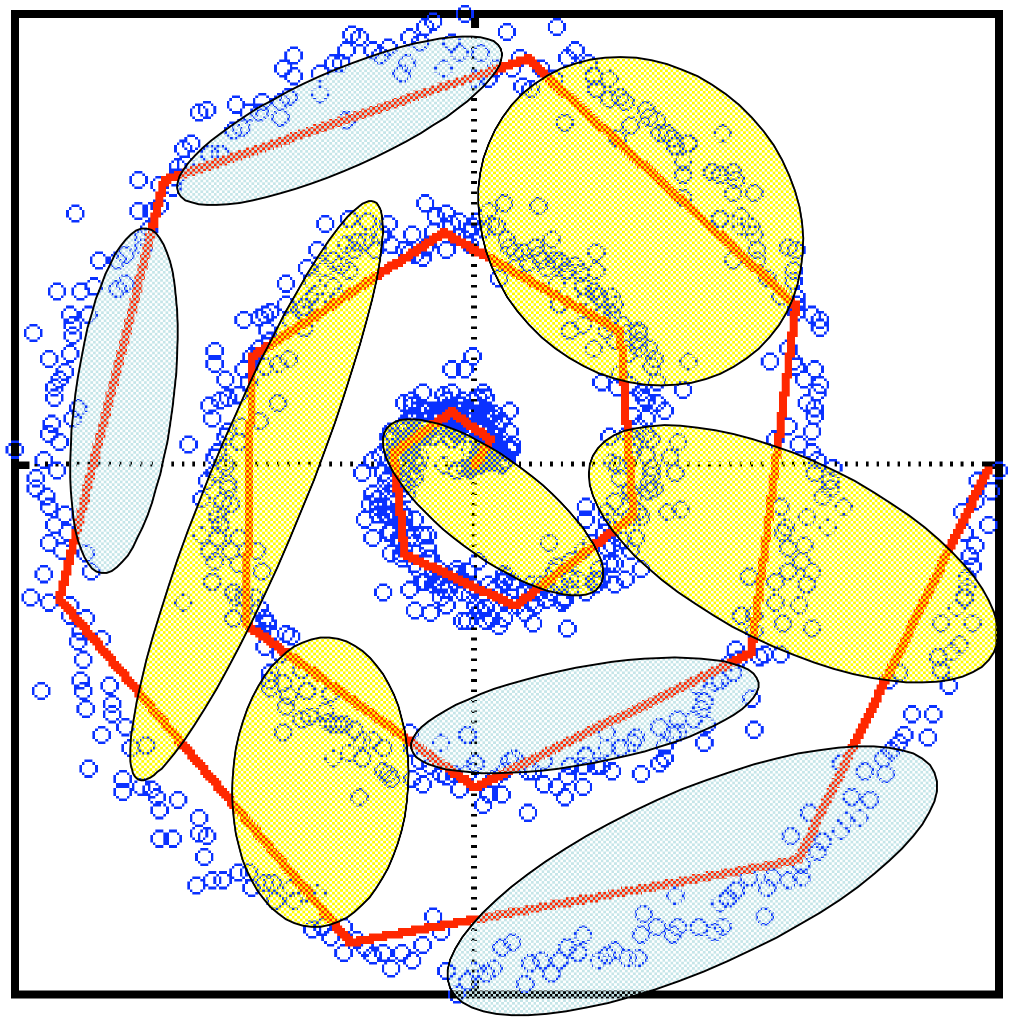
Problem solved?

Can simple iterative clustering algorithms, properly generalized, solve the problem of manifold learning?



Problem solved? No.

Iterative clustering algorithms are sensitive to initial conditions, with many spurious local minima.



Also, remember the goal…

- **Inputs** **(high dimensional)**

  $$\vec{x}_i \qquad {}^{D} \text{ with } i = 1, 2, \ldots, n$$

- **Outputs** **(low dimensional)**

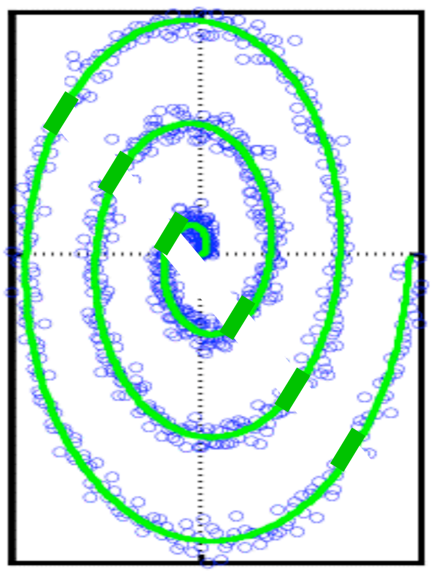
  $$\vec{y}_i \qquad {}^{d} \text{ where } d \ll D$$

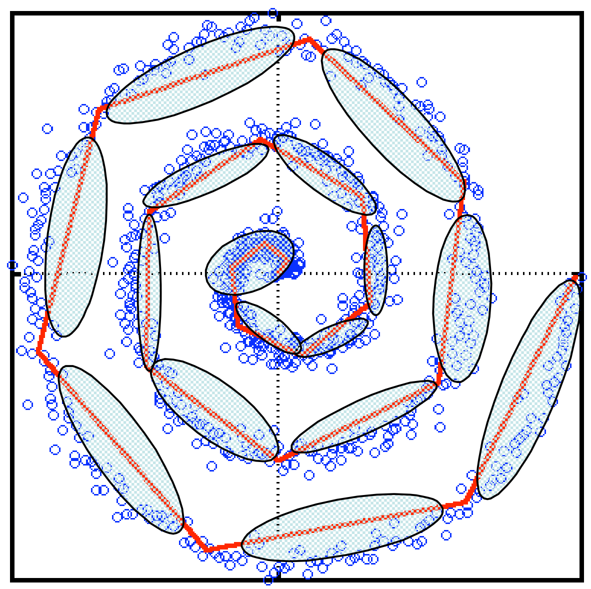
- **Goals**

  Nearby points remain nearby.
  Distant points remain distant.
  (Estimate *d*.)

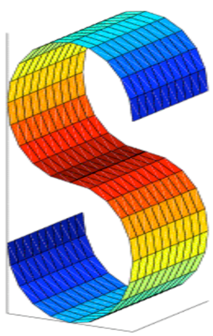
Local vs global

Clustering algorithms do not map their inputs into a single continuous global coordinate system of lower dimensionality.

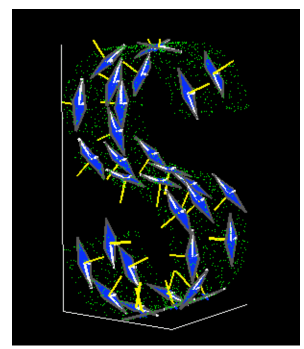




Locally Linear Embedding



“Think globally, fit locally.”



Algorithm

- **Steps**
  1. **Nearest neighbor search.**
  2. **Least squares fits.**
  3. **Sparse eigenvalue problem.**
- **Properties**
  - **Obtains highly nonlinear embeddings.**
  - **Not prone to local minima.**
  - **Sparse graphs yield sparse problems.**

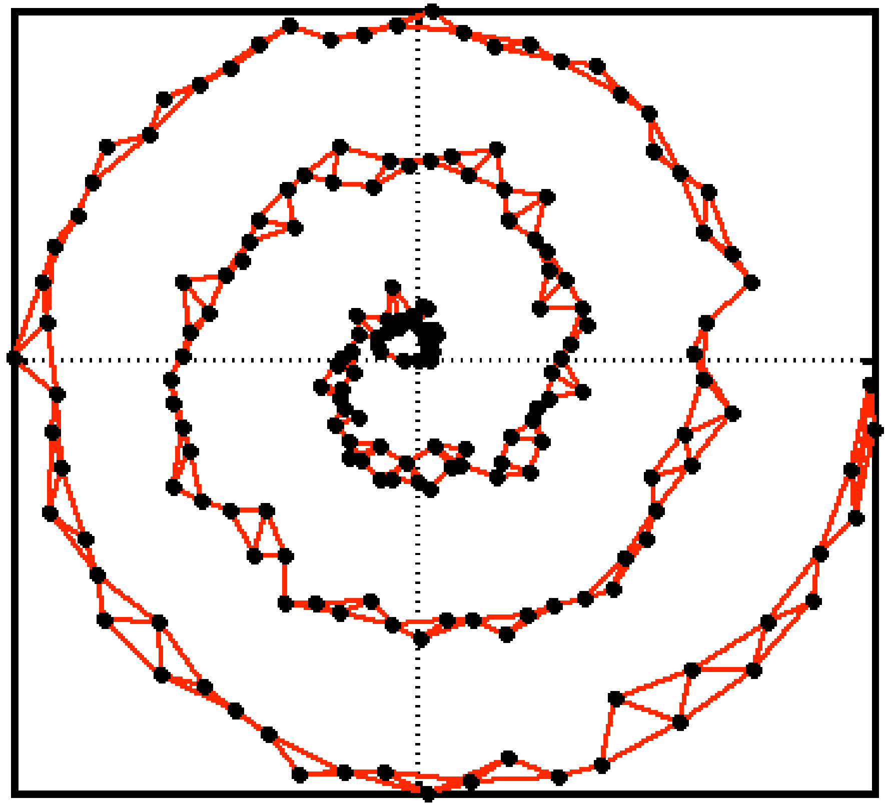
Step 1. Identify neighbors.

- Examples of neighborhoods
  - *k* nearest neighbors
  - Neighbors within radius *r*
  - Metric based on prior knowledge
- Assumptions
  - Data is sampled from a manifold.
  - Manifold is well sampled.

Nearest neighbor graph

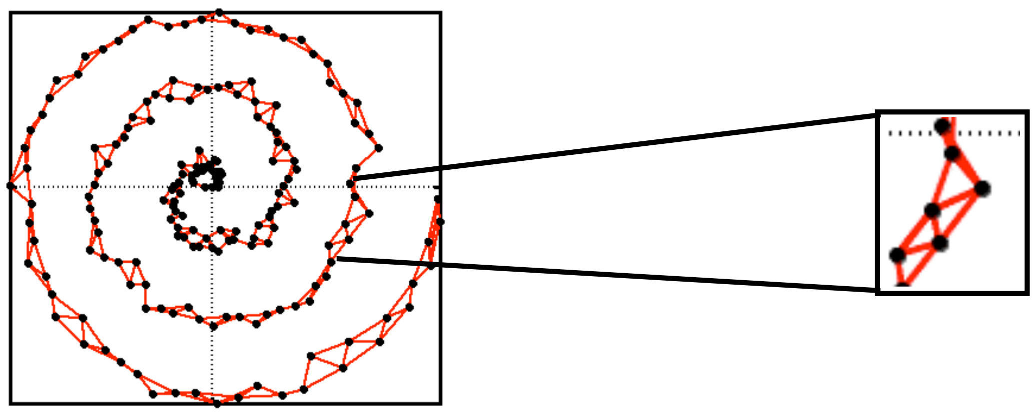
Assumptions:

- Graph is connected.
- Neighborhoods on the graph correspond to neighborhoods on the manifold.



Step 2. Compute weights.

- Characterize local geometry of each neighborhood by weights W_{ij}.



- Compute weights by reconstructing each input (linearly) from neighbors.

Linear reconstructions

- **Local linearity**

  Neighbors lie on locally linear patches of a low dimensional manifold.

- **Reconstruction errors**

  Least squared errors should be small:

$$(W) = \quad {}_{i} \left| \vec{x}_i \qquad {}_{j} W_{ij}\vec{x}_j \right|^2$$

Least squares fits

- **Local reconstructions**

  Choose weights to minimize:

  $$(W) = \;_{i} \left| \vec{x}_i \qquad _{j}\, W_{ij}\vec{x}_j \right|^2$$

- **Constraints**

  Nonzero W_{ij} only for neighbors.

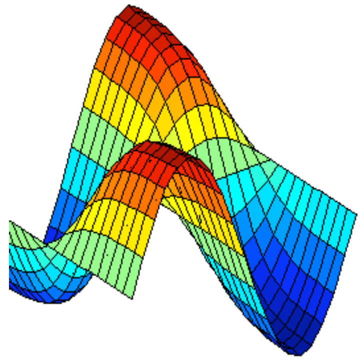
  Weights must sum to one:

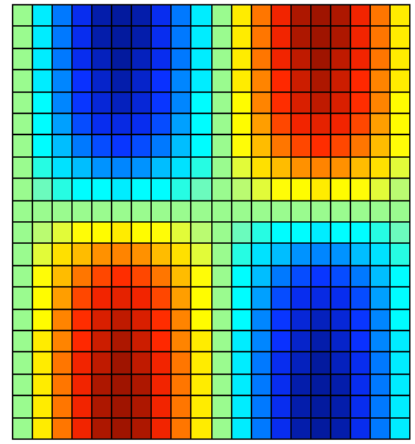
  $$_{j}\; W_{ij} = 1$$

- **Local invariance**

  Optimal weights W_{ij} are invariant to rotation, translation, and scaling.

Symmetries





- **Local linearity**

  If each neighborhood map looks like a translation, rotation, and rescaling...

- **Local geometry**

  …then these transformations do not affect the weights W_{ij}: they remain valid.

Thought experiment

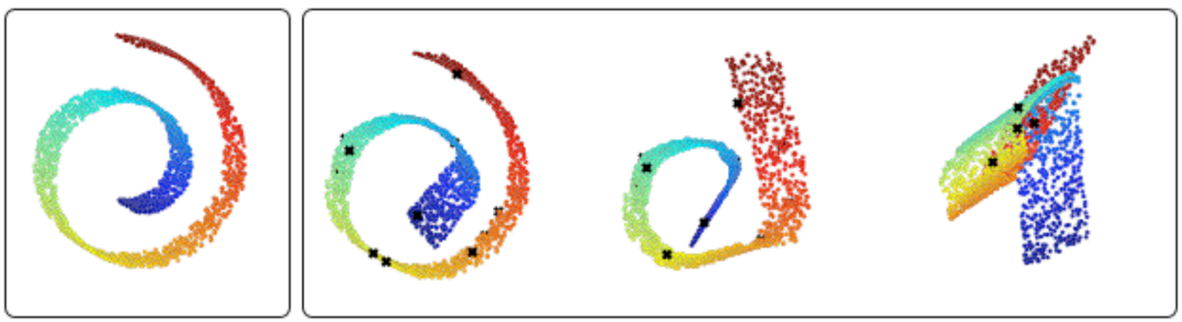
- **Reconstruction from landmarks**

Clamp subset of inputs (“landmarks”), then reconstruct others by minimizing:

$$\boxed{(W) = {}_{i} \left| \vec{x}_i \qquad {}_{j} W_{ij}\vec{x}_j \right|^2}$$

with respect to $\vec{x}_i$!

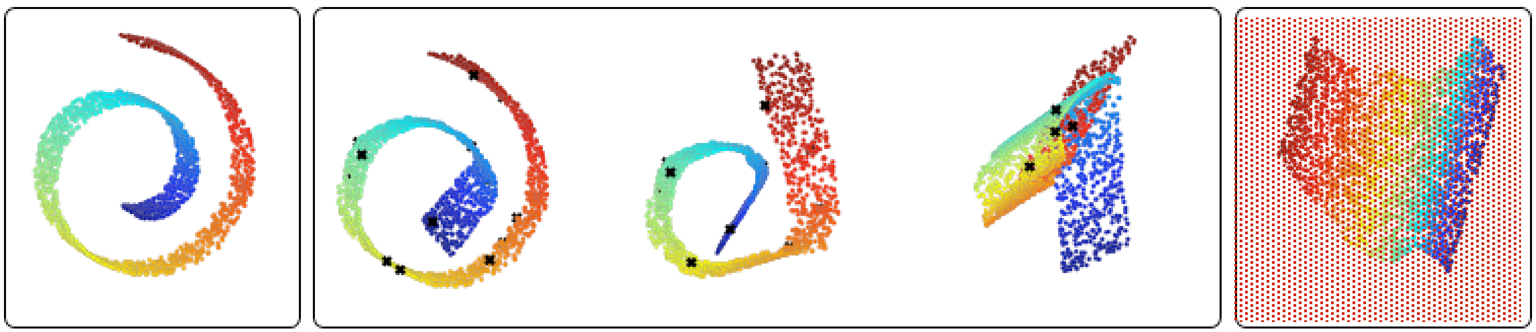
n=2000 inputs



Number of landmarks: L = 15, L = 10, L = 5

Thought experiment (con't)

- Locally linear reconstruction
  - Very accurate for sufficiently large number of landmarks.
  - Increasingly linearized with decreasing number of landmarks.



Number of landmarks: L = 15, L = 10, L = 5 , L = 0 ?

Step 3. “Linearization”

- **Low dimensional representation**

  Map inputs to outputs: $\vec{x}_i \quad {}^{D}$ to $\vec{y}_i \quad {}^{d}$

- **Minimize reconstruction errors.**

  Optimize outputs for fixed weights:

$$\boxed{(y) = {}_{i} \left| \vec{y}_i \qquad {}_{j} W_{ij} \vec{y}_j \right|^2}$$

- **Constraints**

  Center outputs on origin: ${}_{i} \vec{y}_i = \vec{0}$

  Impose unit covariance matrix: $\frac{1}{N} \; {}_{i} \; \vec{y}_i \vec{y}_i^{T} = I_d.$

Sparse eigenvalue problem

- **Quadratic form**

$$(y) = \quad {}_{ij} \quad {}_{ij}\left(\vec{y}_i \quad \vec{y}_j\right) \text{ with } \quad = (I \quad W)^T (I \quad W)$$

- **Rayleigh-Ritz quotient**

Optimal embedding given by bottom d+1 eigenvectors.

- **Solution**

Discard bottom eigenvector [1 1 … 1]. Other eigenvectors satisfy constraints.

Summary of LLE

- **Three steps**
  1. **Compute k-nearest neighbors.**
  2. **Compute weights W_{ij}.**
  3. **Compute outputs $\vec{y}_i$.**
- **Optimizations**

$$(W) = \;{}_{i}\; \left| \vec{x}_i \qquad {}_{j} W_{ij}\vec{x}_j \right|^2$$

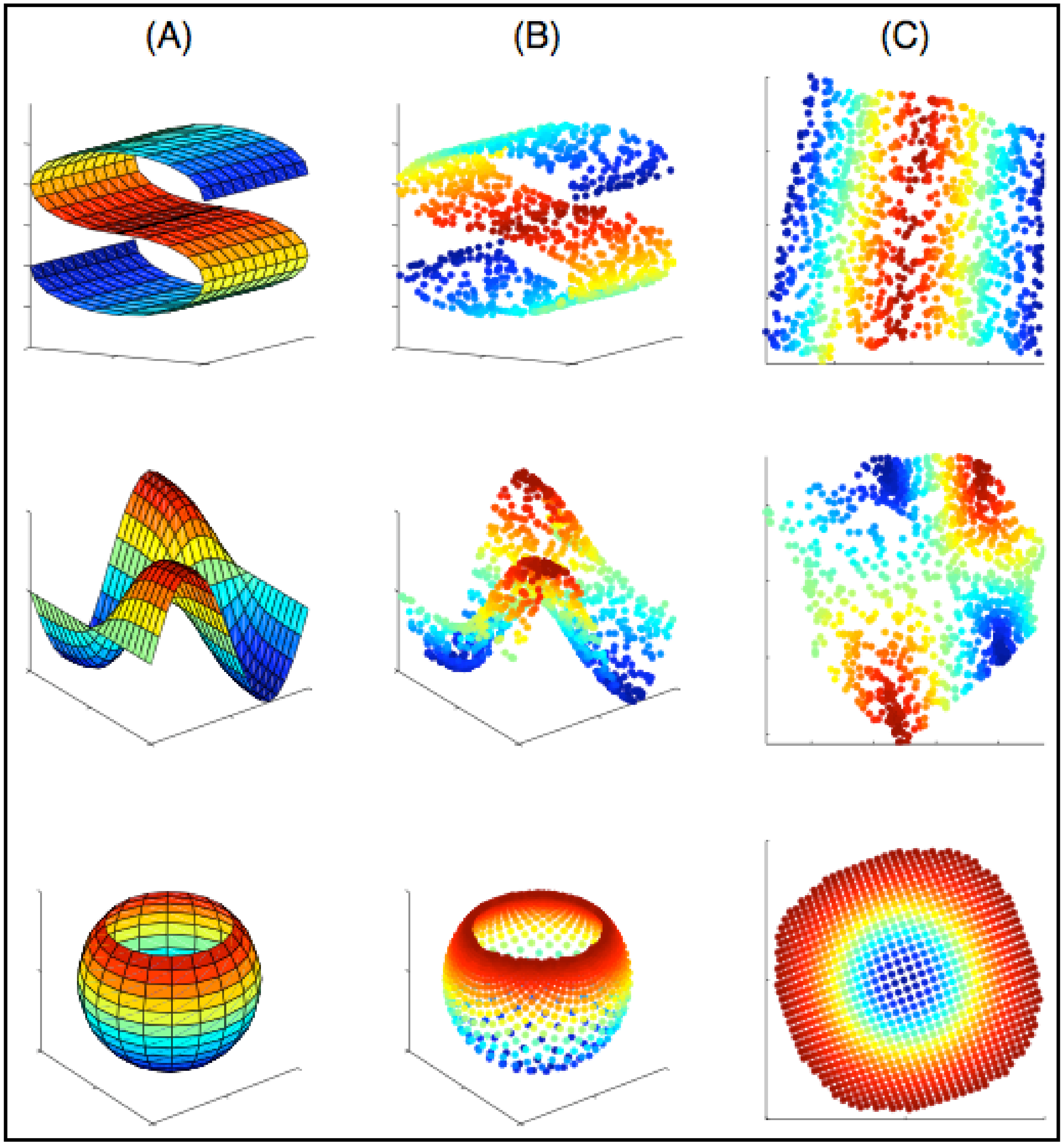
$$(y) = \;{}_{i}\; \left| \vec{y}_i \qquad {}_{j} W_{ij}\vec{y}_j \right|^2$$

Surfaces

N=1000
inputs

k=8
nearest neighbors

D=3
d=2
dimensions



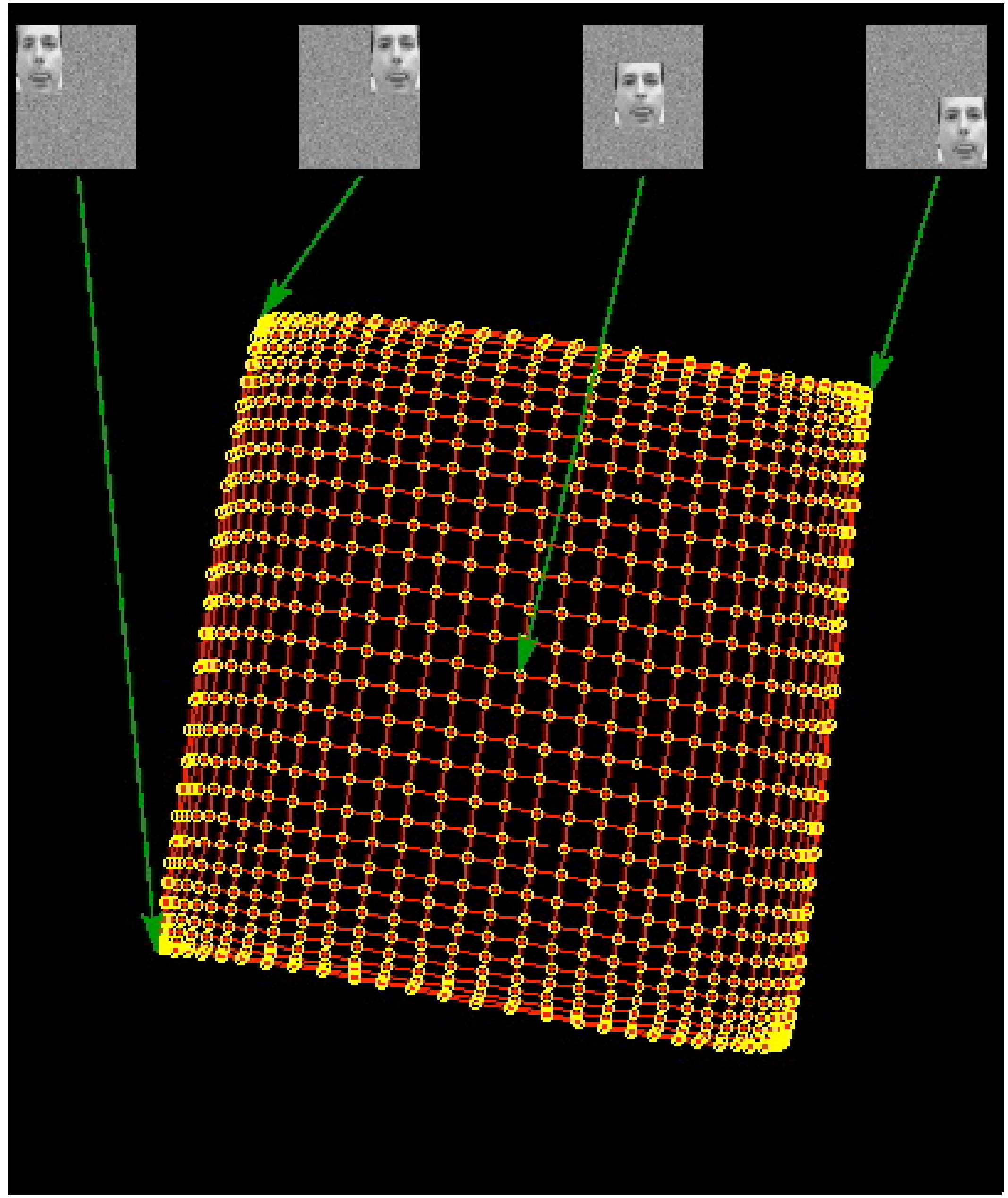
Translated faces

N=961
images

k=4
nearest neighbors

D=3009
pixels

d=2
manifold



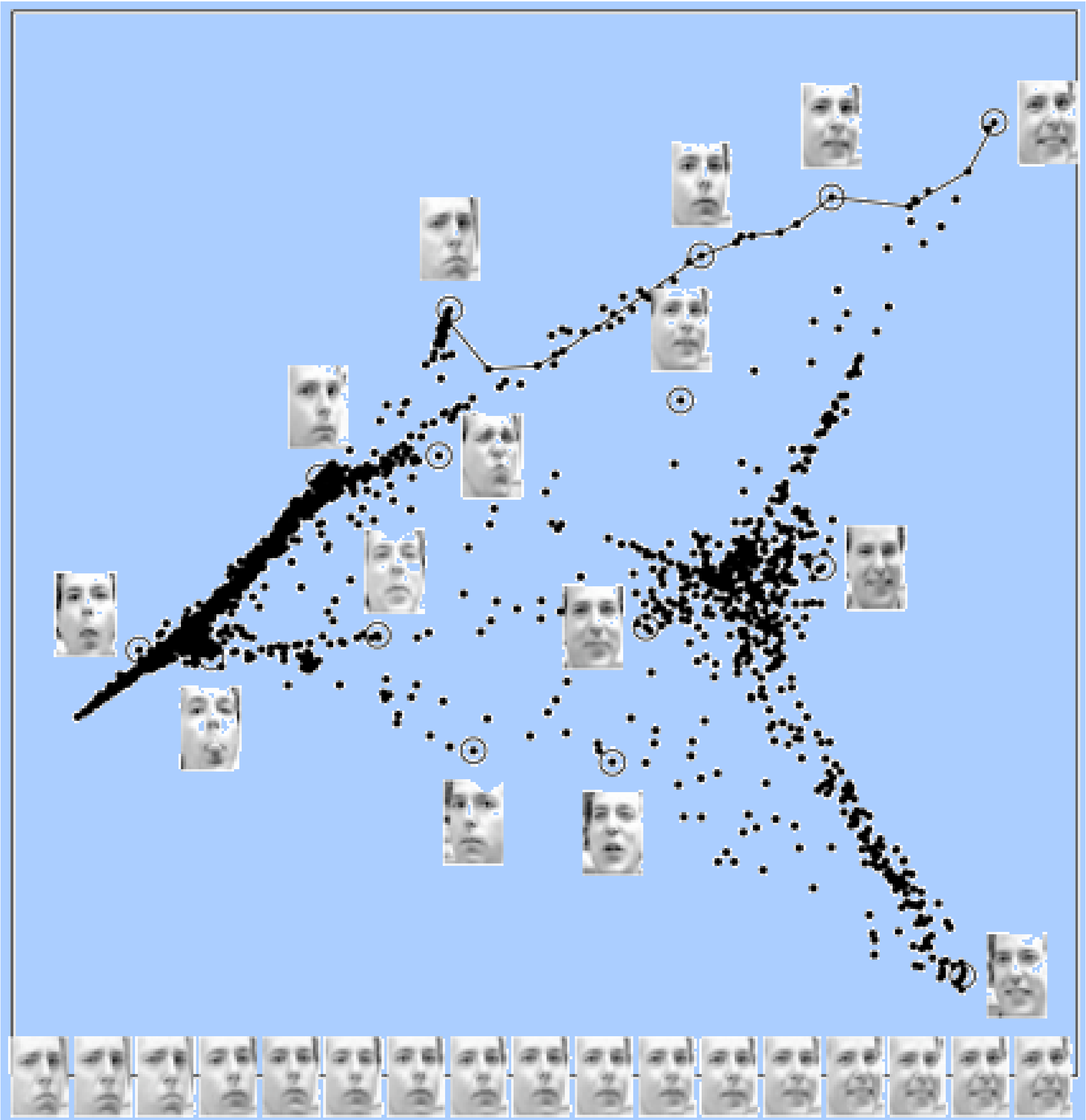
Pose and expression

N=1965 images

k=12 nearest neighbors

D=560 pixels

d=2 (shown)



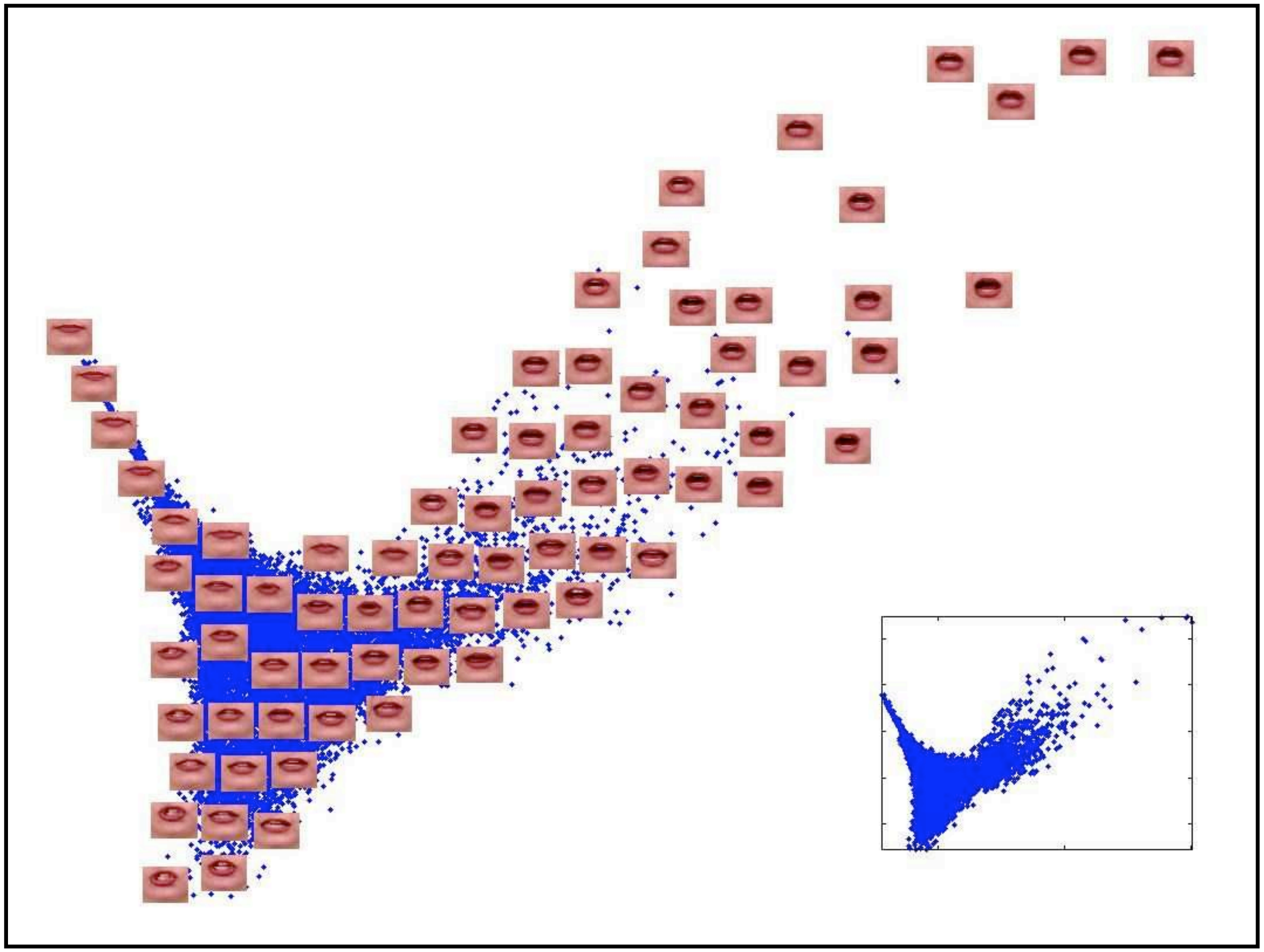
Lips

N=15960
images

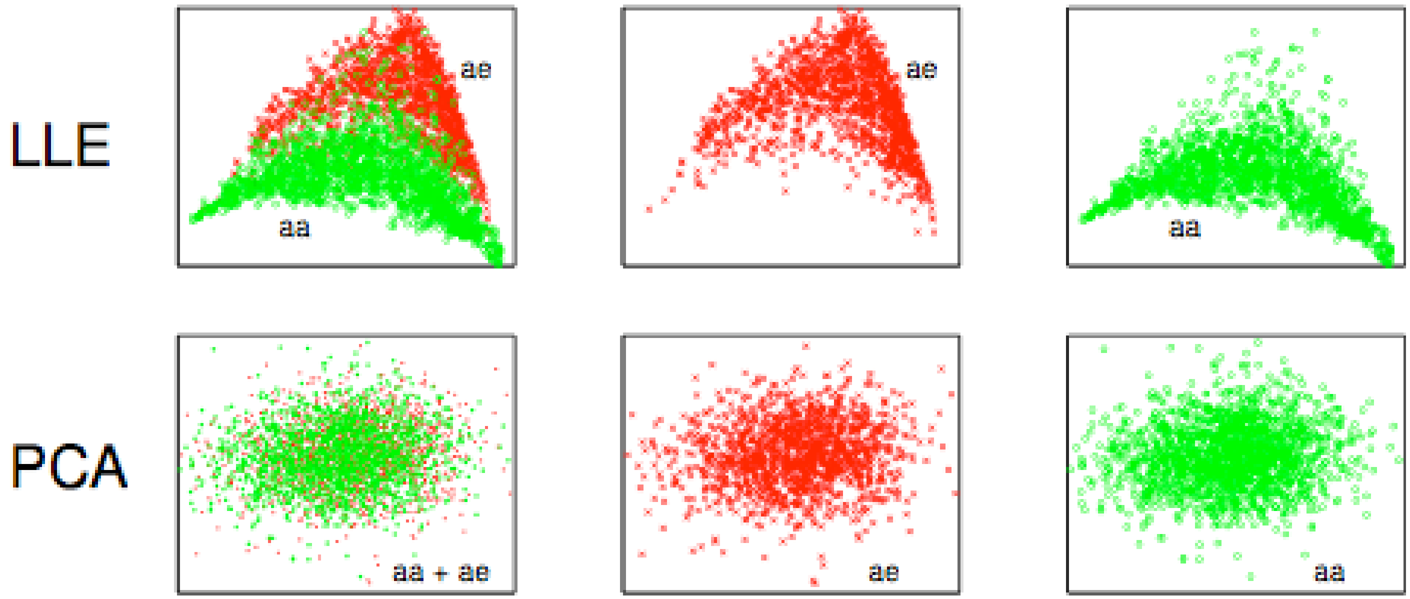
K=24
neighbors

D=65664
pixels

d=2
(shown)



Vowels: /aa/ (“hot”) vs /ae/ (“hat”)



N=3000 log-power spectra
K=10 nearest neighbors
D=400 window size

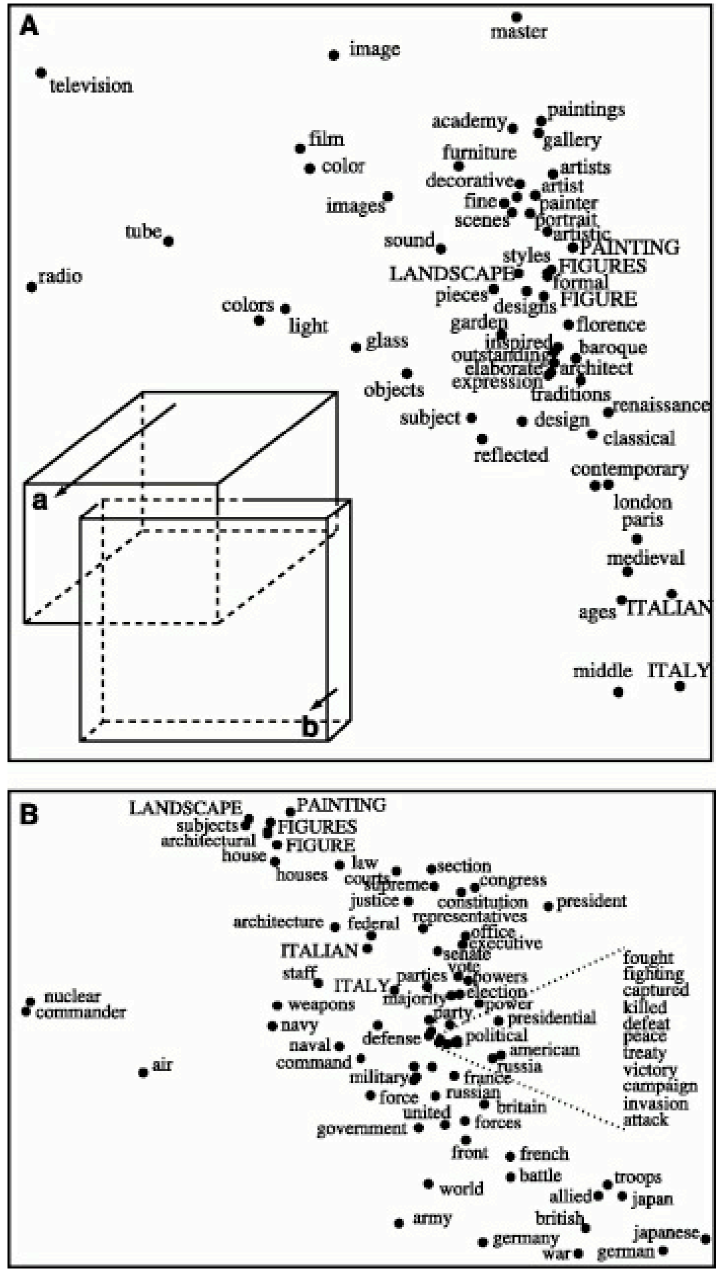
Word-document counts

n=5000
words

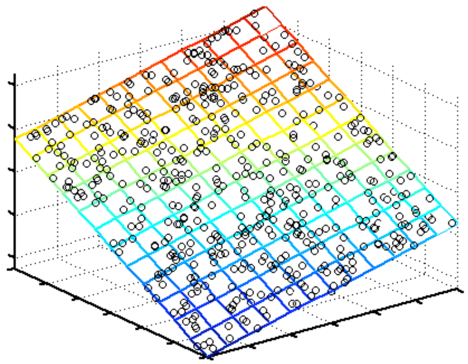
k=20
nearest neighbors

D=31000
documents

d=3,4,5
(shown)



Linear vs nonlinear



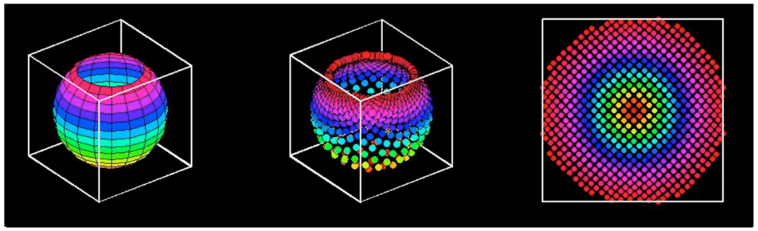
What computational price must we pay for nonlinear dimensionality reduction?

Properties of LLE

- Strengths
  - Polynomial-time optimizations
  - No local minima
  - Non-iterative (one pass thru data)
  - Non-parametric
  - Only heuristic is neighborhood size.
- Weaknesses
  - Sensitive to “shortcuts”
  - No out-of-sample extension
  - No estimate of dimensionality

LLE versus Isomap

- **Many similarities**
  - Graph-based, spectral method
  - No local minima
- **Essential differences**
  - Does not estimate dimensionality
  - No theoretical guarantees
  + Constructs sparse vs dense matrix
  ? Preserves weights vs distances



Conformal mapping

Algorithms

2000
Isomap
(Tenenbaum, de Silva, & Langford)

Locally Linear Embedding
(Roweis & Saul)

2002
Laplacian eigenmaps
(Belkin & Niyogi)

2003
Hessian LLE
(Donoho & Grimes)

2004
Maximum variance unfolding
(Weinberger & Saul)
(Sun, Boyd, Xiao, & Diaconis)

2005
Conformal eigenmaps
(Sha & Saul)

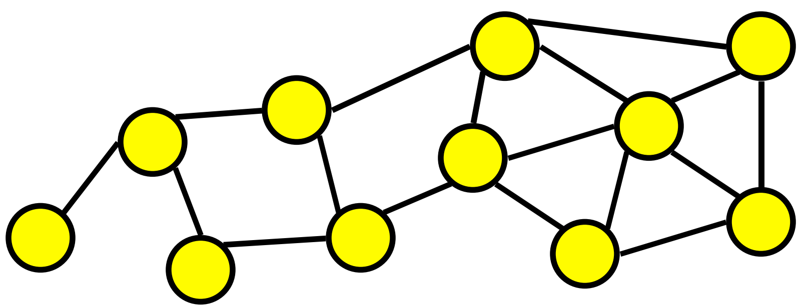
Laplacian eigenmaps

- **Key idea:**

  Map nearby inputs to nearby outputs, where nearness is encoded by graph.

- **Physical intuition:**

  Find lowest frequency vibrational modes of a mass-spring system.



Summary of algorithm

- **Three steps**

  1. **Identify k-nearest neighbors**
  2. **Assign weights to neighbors:**

$$\boxed{W_{ij} = 1 \ \text{ or } \ W_{ij} = \exp\left(\quad \| \vec{x}_i \quad \vec{x}_j \|^2 \right)}$$

  3. **Compute outputs by minimizing:**

$$\boxed{(y) = \quad {}_{ij} \frac{W_{ij} \| \vec{y}_i \quad \vec{y}_j \|^2}{\sqrt{D_{ii} D_{jj}}} \ \text{ where } \ D_{ii} = \quad {}_{j} W_{ij}}$$

(sparse eigenvalue problem as in LLE)

Laplacian vs LLE

- **More similar than different**
  - Graph-based, spectral method
  - Sparse eigenvalue problem
  - Similar results in practice
- **Essential differences**
  - Preserves locality vs local linearity
  - Uses graph Laplacian

$$L = D \quad W \qquad \text{(unnormalized)}$$

$$\mathcal{L} = I \quad D^{\ 1/2} W D^{\ 1/2} \qquad \text{(normalized)}$$

Analysis on Manifolds

- **Laplacian in $\mathcal{R}^d$**

  Function $f(x_1,x_2,\ldots,x_d)$ has Laplacian:

$$f = {}_{i} \frac{{}^{2} f}{x_i^2}$$

- **Manifold Laplacian**

  Change is measured along tangent space of manifold.

- **Stokes theorem**

$${}_{M} \| \; f \|^2 = {}_{M} f \;\; f$$

Spectral graph theory

- **Manifolds and graphs**

  Weighted graph is discretized representation of manifold.

- **Laplacian operators**

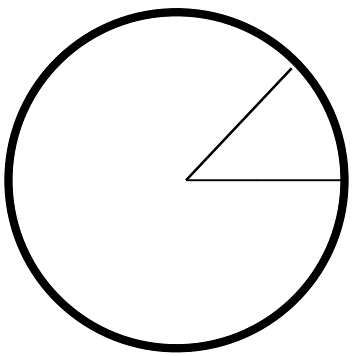
  Laplacian measures smoothness of functions over manifold (or graph).

$$
{}_{M} \| \quad f \|^2 = \quad {}_{M} f \quad f \qquad \text{(manifold)}
$$

$$
{}_{ij} W_{ij} \left(f_i \qquad f_j \right)^2 = f \quad Lf \qquad \text{(graph)}
$$

Example: S^1 (the circle)

- **Continuous**
  - Eigenfunctions of Laplacian are basis for periodic functions on circle, ordered by smoothness.
  - Eigenvalues measure smoothness.

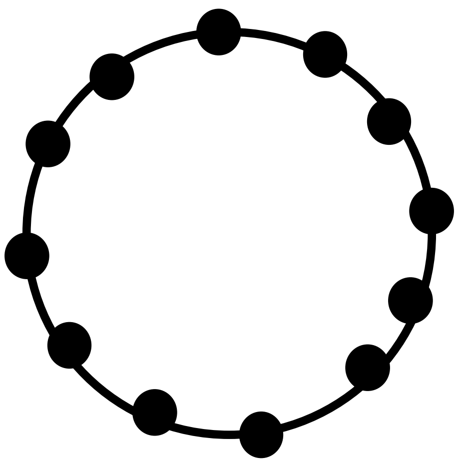


$$-\frac{{}^{2} f_m}{{}^{2}} = {}_m f_m(\quad)$$

$$f_m(\quad) = \left\{ \begin{matrix} \sin(m\quad) \\ \cos(m\quad) \end{matrix} \right\} \text{ with } {}_{\mathrm{m}} = m^2$$

Example: S^1 (the circle)

- **Discrete (n equally spaced points)**
  - **Eigenvectors of graph Laplacian are discrete sines and cosines.**
  - **Eigenvalues measure smoothness.**

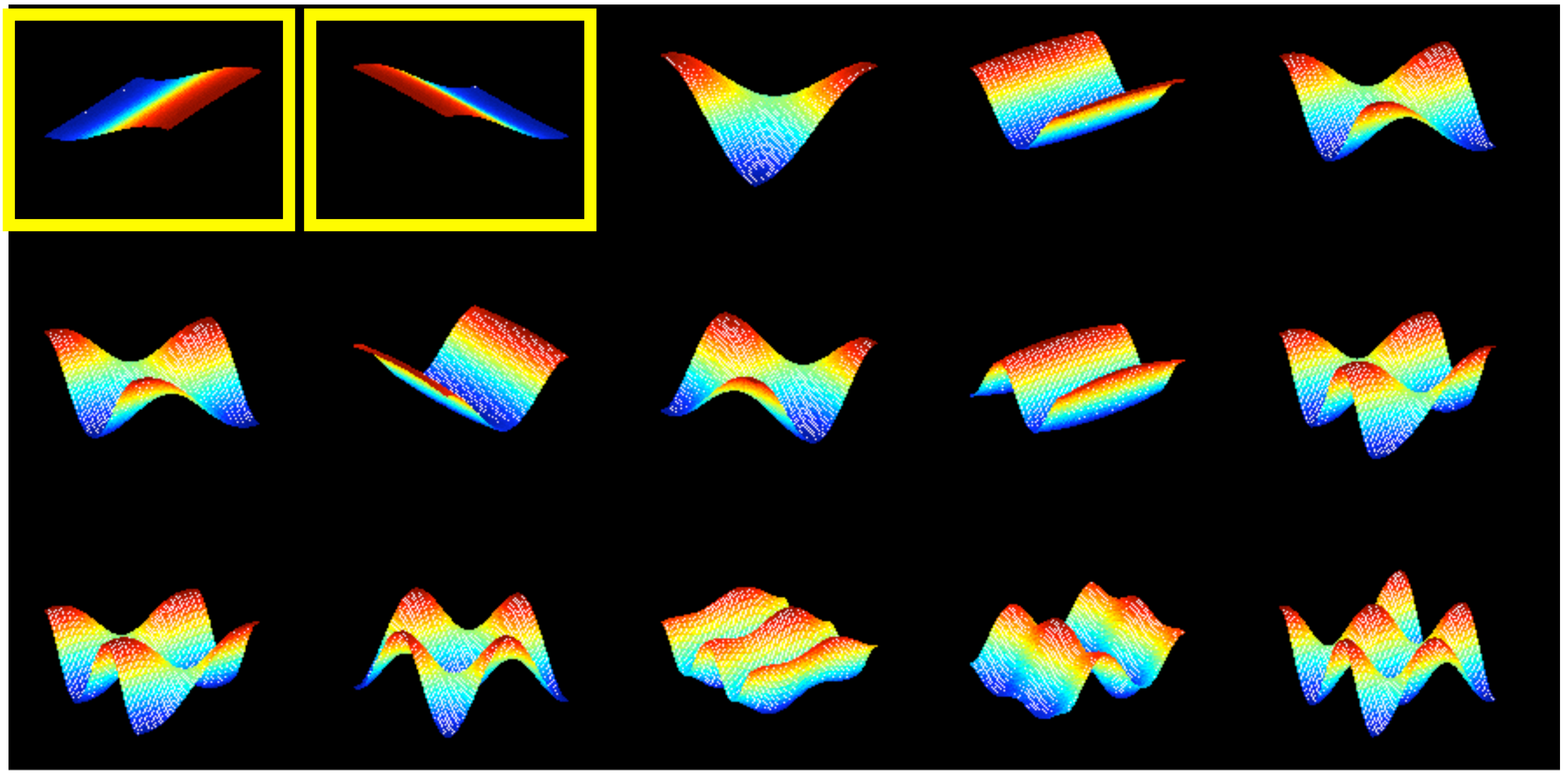


Graph embedding from Laplacian eigenmaps:

$$\boxed{\vec{y}_k = \left(\cos\left(2\quad k/n\right), \sin\left(2\quad k/n\right)\right)}$$

Example: Swiss roll

eigenvectors of graph Laplacian



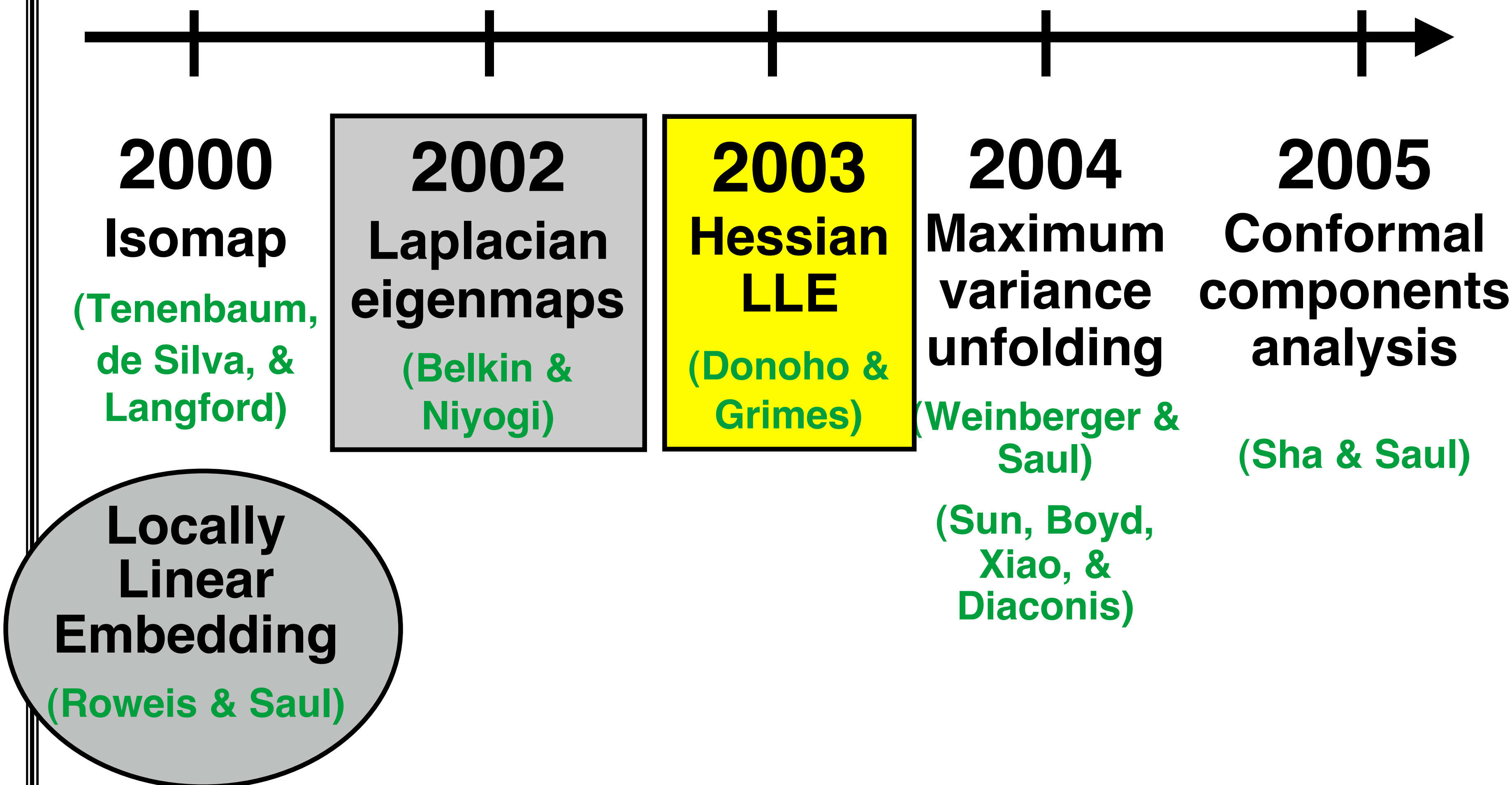
A critical view…

- LLE and Laplacian eigenmaps
  - Construct quadratic form over functions on graph.
  - Take d lowest cost (but non-constant) functions as manifold coordinates.
- Theoretical guarantees?
  - When do bottom eigenvectors give the “right answer”?
  - Depends on the definition of the “right answer”…

A critical view (con't)

- **Assumption**
  - Sample inputs from manifold that is isometrically embedded in $\mathcal{R}^D$.
  - Assume manifold is locally isometric to an open subset of $\mathcal{R}^d$, where $d < D$.
- **Hypothesis**
  - Isomap's top d eigenvectors recover parameterization for convex subsets.
  - Can bottom d (nonzero) eigenvectors of sparse matrix method do better?

Algorithms


2000
Isomap
(Tenenbaum, de Silva, & Langford)
2002
Laplacian eigenmaps
(Belkin & Niyogi)
2003
Hessian LLE
(Donoho & Grimes)
2004
Maximum variance unfolding
(Weinberger & Saul)
(Sun, Boyd, Xiao, & Diaconis)
2005
Conformal components analysis
(Sha & Saul)
Locally Linear Embedding
(Roweis & Saul)

Hessian LLE

- **Assumption**

  Data manifold M is locally isometric to open, connected subset of $\mathcal{R}^d$.

- **Key ideas**
  - **Define Hessian via orthogonal coordinates on tangent planes of M.**
  - **Quadratic form (f) averages Frobenius norm of Hessian over M.**

$$(f) = {}_{M} \left\| H_f(m) \right\|^2 dm$$

$$\cancel{{}_{M} \| \; f \|^2 = {}_{M} \; f \; f}$$

Hessian LLE

$$(f) = \quad_{M} \left\| H_f(m) \right\|^2 dm$$

$$\cancel{{}_{M} \| \quad f \|^2 = \quad_{M} \quad f \quad f}$$

- **Key ideas (con't)**
  - **Every function with vanishing Hessian is linear. (Not so for Laplacian.)**
  - **Bottom eigenfunctions in null space of $H(f)$ yield isometric coordinates.**
  - **Graph-based discretization yields algorithm.**

Hessian LLE

- **Three steps**
  1. **Construct graph from kNN.**
  2. **Estimate Hessian operator at each data point.**
  3. **Compute bottom eigenvectors of sparse quadratic form.**

$$(f) = {}_{M} \left\| H_f(m) \right\|^2 dm$$

- **What's new?**

  (1) and (3) are same as before.

  (2) estimates Hessian. (Details omitted.)

Relation to previous work

- **Algorithm variant of LLE**

  Replaces least squares fits in LLE by estimation of Hessian.

- **Conceptual variant of Laplacian**

  Substitutes Frobenius norm of Hessian for norm of gradient vector.

- **Sparse matrix variant of Isomap**

  Also looks for isometric coordinates on data manifold.

Theoretical guarantees

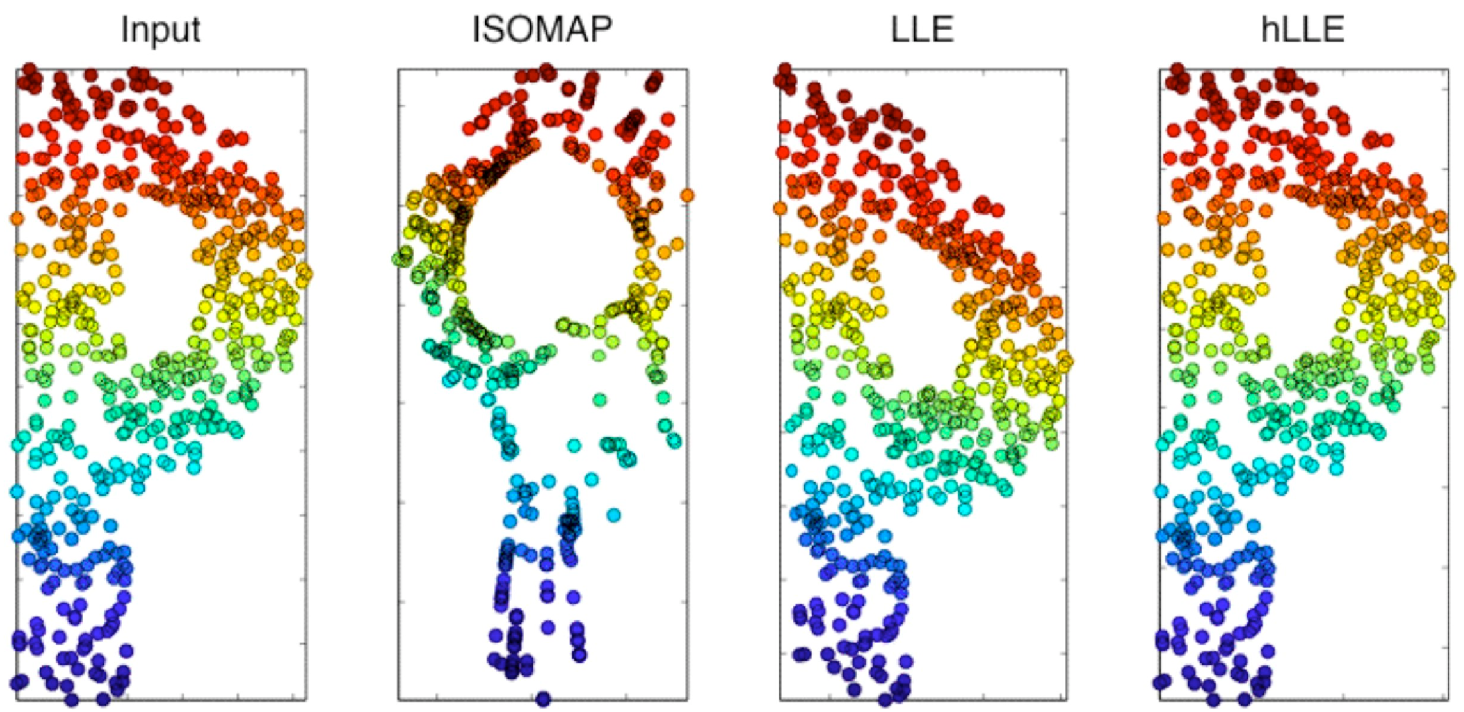
- **Asymptotic convergence**

  For data sampled from a submanifold that is isometric to an open, connected subset of Euclidean space, hLLE will recover the subset up to rigid motion.

- **No convexity assumption**

  Convergence is obtained for a larger class of manifolds than Isomap.

Connected but not convex

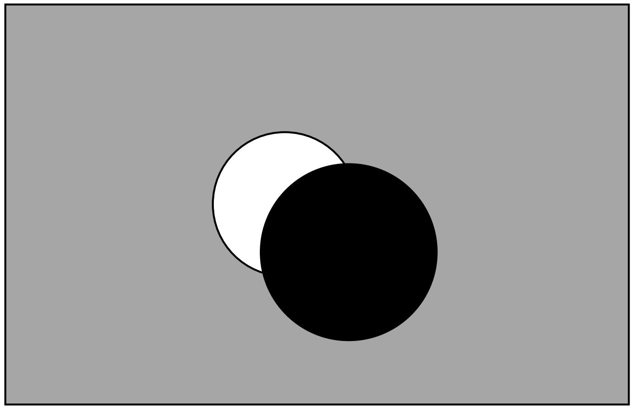


Hessian LLE yields an isometric embedding, but not Isomap or LLE.

Connected but not convex

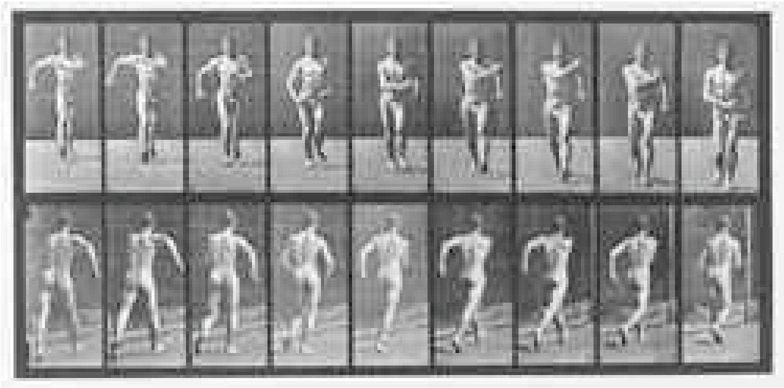
- **Occlusion**

  Images of two disks, one occluding the other.



- **Locomotion**

  Images of periodic gait.



Algorithms

2000

Isomap

(Tenenbaum, de Silva, & Langford)

Locally Linear Embedding

(Roweis & Saul)

2002

Laplacian eigenmaps

(Belkin & Niyogi)

2003

Hessian LLE

(Donoho & Grimes)

What is left to do?

Problem solved?

- For manifolds without “holes”:
  - Isomap with asymptotic guarantees
  - landmark Isomap for large data sets
- More generally:
  - hLLE with asymptotic guarantees?
  - sparse matrix method should scale well to large data sets?

(If it seems too good to be true, it usually is…)

Flies in the ointment

- **How to estimate dimensionality?**

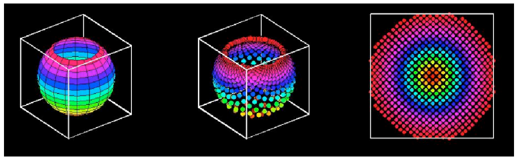
  Revealed by eigenvalue gap of Isomap, but specified in advance for (h)LLE.

- **How to compute eigenvectors?**

  Bottom eigenvalues are very closely spaced for large data sets.

- **Must we preserve distances?**

  Preserving distances may hamper dimensionality reduction.



Computing eigenvectors

- Numerical difficulty

  Inversely proportional to spacing between adjacent eigenvalues.

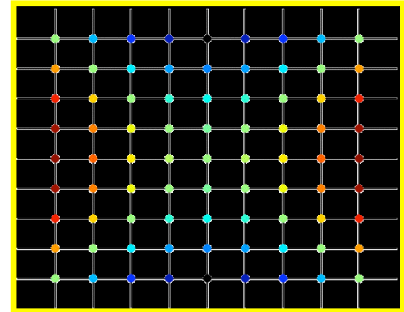
- Scaling to large data sets

  Bottom eigenvalue spacing shrinks with increased sampling of manifold.

- Conundrum

  Finer discretization of manifold trades off with ability to resolve eigenvectors.

Example



- **Lattice model**

  Inputs are n sites of hypercubic lattice. Edges connect $2d$ nearest neighbors.

- **Fourier diagonalization**

  Graph Laplacian has translational symmetry. Eigenvectors: exp($i\vec{q}\bullet\vec{x}$).

- **Eigenvalues**

  For $n=\infty$, eigenvalues are indexed continuously by $\vec{q}$ in $[-\pi,\pi]^d$; no gaps!

Can we combine strengths of:

- **Isomap**

  Eigenvalues reveal dimensionality.
  Landmark version scales well.
  Numerically stable.

- **hLLE**

  Solves sparse eigenvalue problem.
  Handles manifolds with “holes”.

- **LLE and Laplacian eigenmaps**

  Aggressive dimensionality reduction
  Locality vs distance-preserving maps

See you tomorrow...

2000
Isomap
(Tenenbaum, de Silva, & Langford)

Locally Linear Embedding
(Roweis & Saul)

2002
Laplacian eigenmaps
(Belkin & Niyogi)

2003
Hessian LLE
(Donoho & Grimes)

2004
Maximum variance unfolding
(Weinberger & Saul)
(Sun, Boyd, Xiao, & Diaconis)

2005
Conformal eigenmaps
(Sha & Saul)