Nonlinear dimensionality reduction

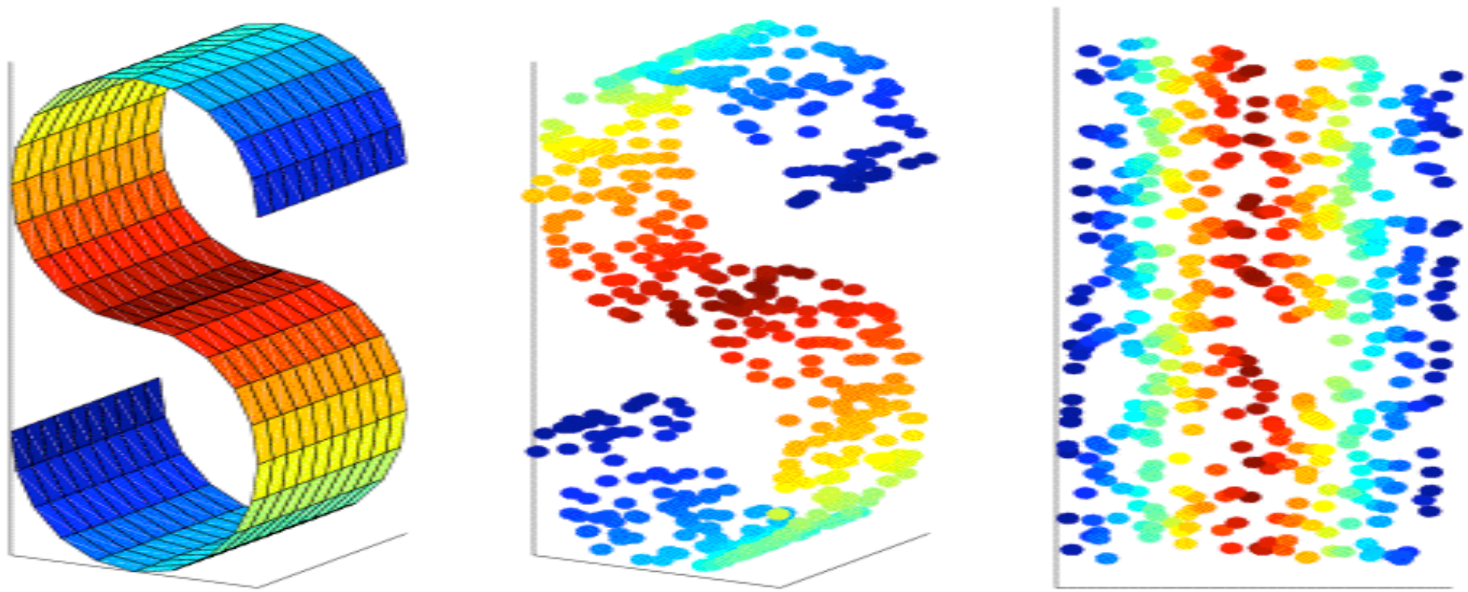
Prof. Lawrence Saul
Computer and Information Science
University of Pennsylvania

Outline

- Motivation
- Algorithm #1
- Algorithm #2
- Related work

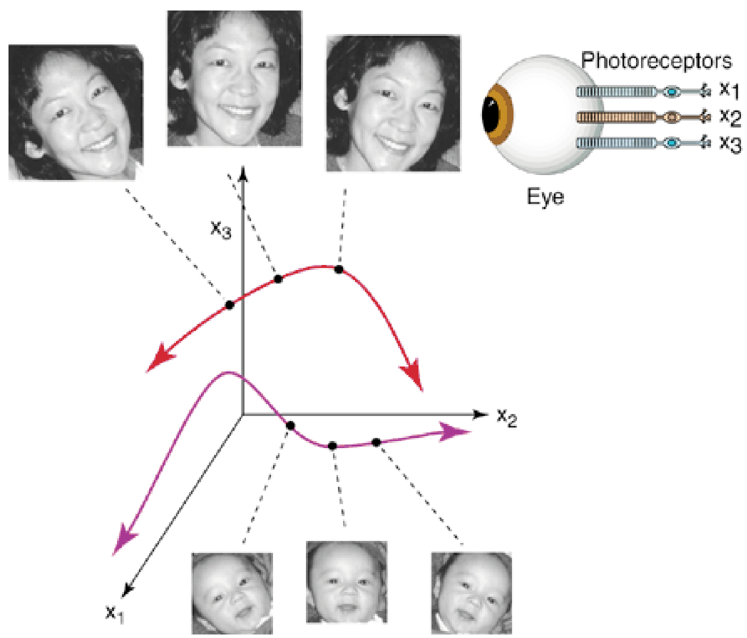
Statistics, Geometry, Computation!

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

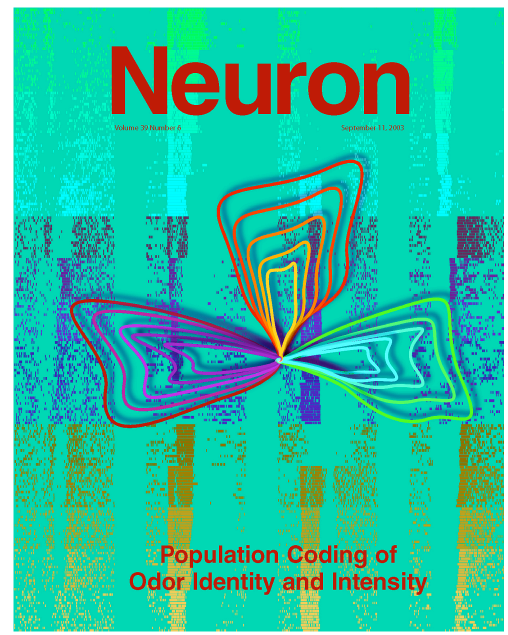


Applications

Low dimensional manifolds arise in many areas of information processing.



(Seung & Lee, 2000)



(Stopfer et al, 2003)

Unsupervised learning

- **Inputs** (high dimensional)

  $$\vec{X}_i \in \mathbb{R}^D \text{ with } i = 1,2,\ldots,N$$

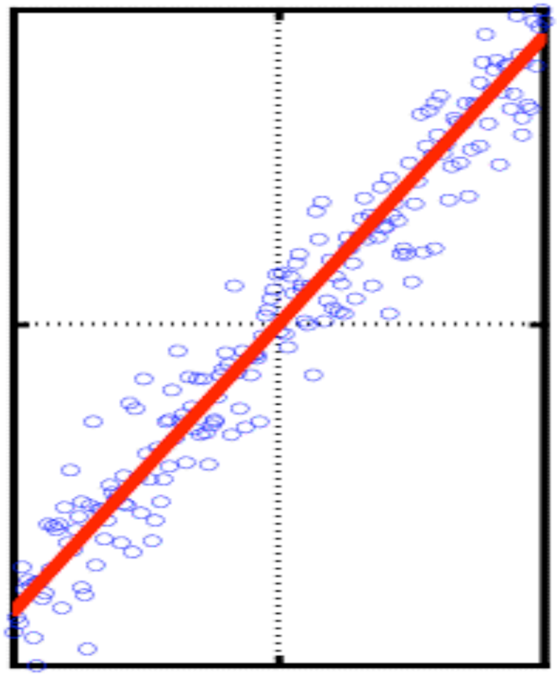
- **Outputs** (low dimensional)

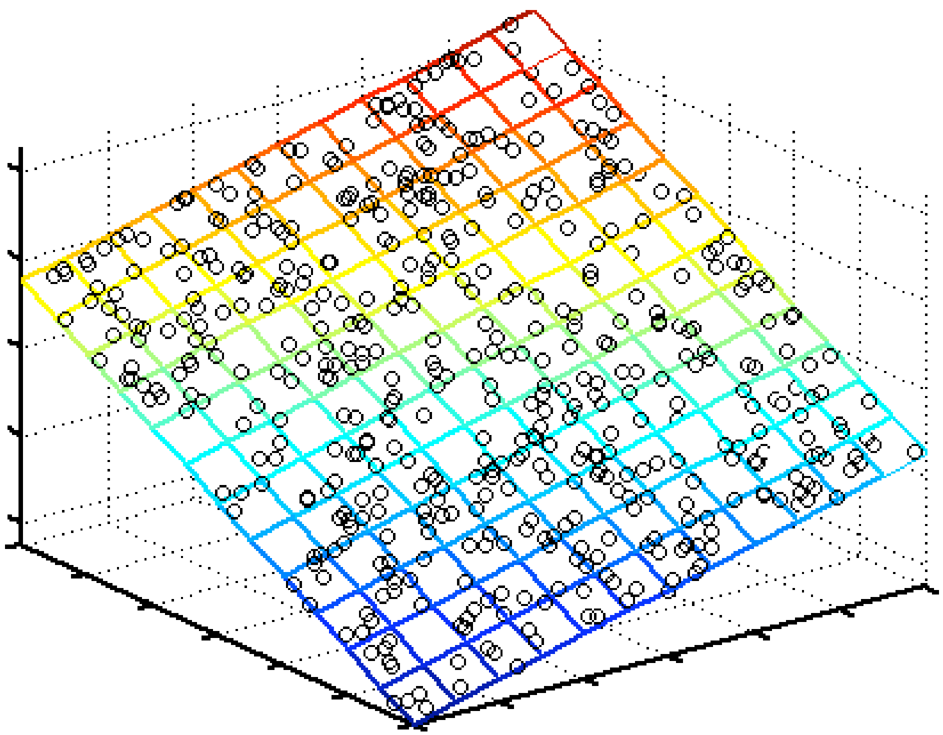
  $$\vec{Y}_i \in \mathbb{R}^d \text{ where } d < D$$

- **Embedding**

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

Subspaces





$D = 2$
$d = 1$

$D = 3$
$d = 2$

Linear methods

- **Principal component analysis**

  Project inputs into subspace of maximal variance:

$$\max\left(\mathrm{tr}\left[Y^T Y\right]\right) \text{ with } Y = PX$$

- **Multidimensional scaling**

  Project inputs into subspace that preserves pairwise distances:

$$\left|\vec{Y}_i \square \vec{Y}_j\right|^2 \square \left|\vec{X}_i \square \vec{X}_j\right|^2$$

Matrices of PCA and MDS

Algorithm	Matrix	Size
PCA	$C = XX^T$	$D \times D$
MDS	$G = X^TX$	$N \times N$

Correlation matrix: $C_{\alpha\beta} \sim E[X_\alpha X_\beta]$

Gram matrix: $G_{ij} = \vec{X}_i \cdot \vec{X}_j$

These matrices have the same rank and eigenvalues.

Spectral embeddings

- **Eigenvectors**

  eigs(C) = linear projections of PCA

  eigs(G) = projected outputs of MDS

- **Eigenvalues**

  Always nonnegative.

  Gaps indicate latent dimensionality.

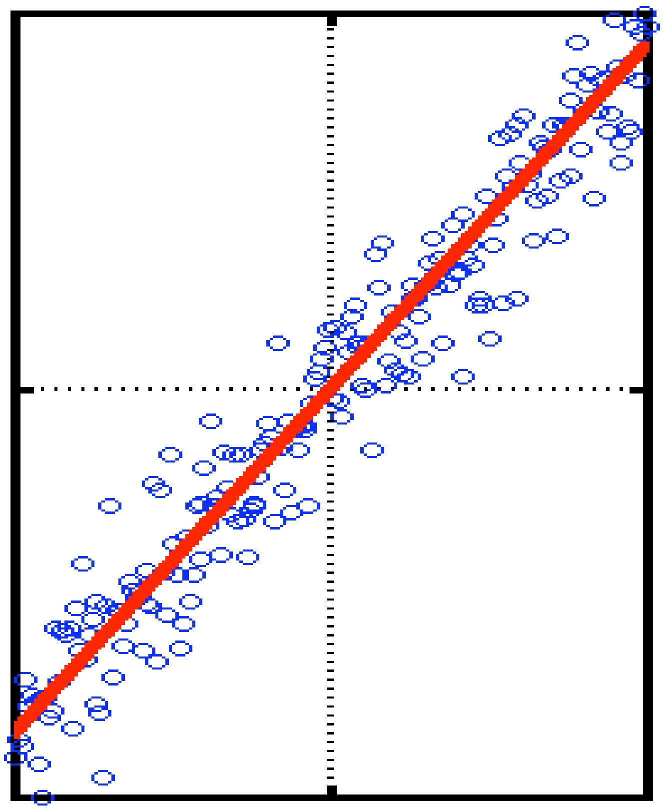
Different intuitions, but equivalent results.

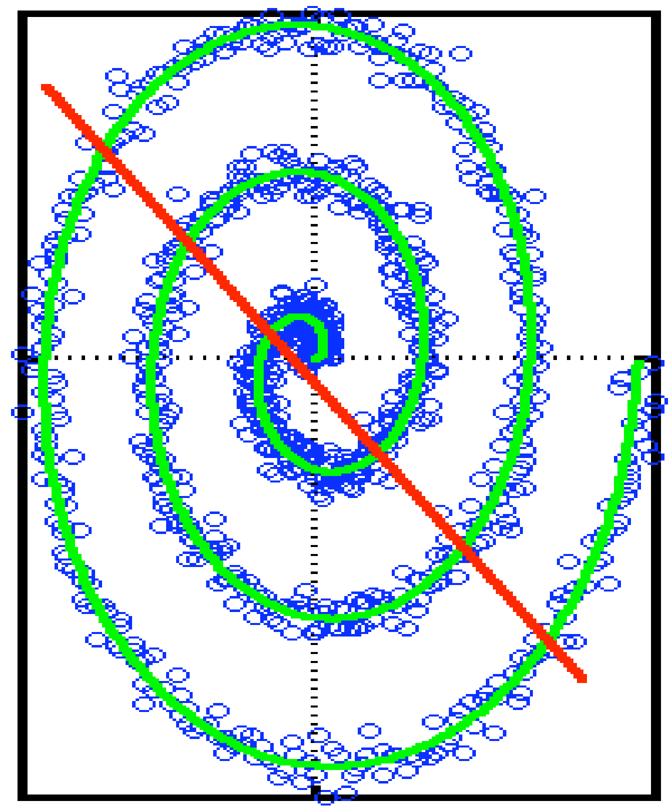
Properties of PCA and MDS

- Strengths
  - Eigenvector methods
  - Non-iterative
  - No local optima
  - No “free” parameters
- Weakness

  PCA and MDS are linear methods.

Subspaces vs Manifolds





Linear methods are limited.

Non-eigenvector methods

- Examples
  - Autoencoder neural networks
  - Self-organizing maps
  - Latent variable models
- Issues
  - Local optima
  - Weaker guarantees
  - Harder implementations

Questions

- **Are there eigenvector methods for nonlinear dimensionality reduction?**

$(\text{Yes})^n$ with $n \geq 8$

- **Equally simple as PCA and MDS?**

Almost!

Eigenvector methods

- **Today**

  Locally linear embedding (LLE)

  Semidefinite embedding (SDE)

- **Others**

  Kernel PCA

  Isomap

  Laplacian eigenmaps

  Local tangent space alignment

  Hessian LLE

  Charting

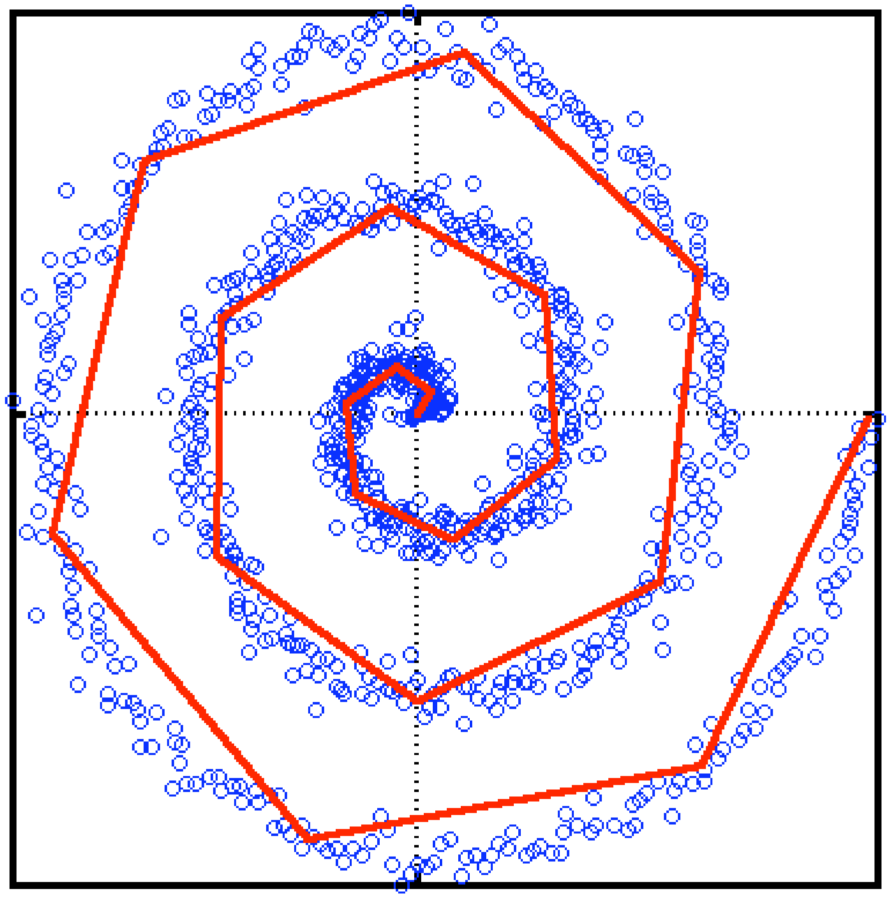
Outline

- Motivation
- Algorithm #1: LLE
  ``Think globally, fit locally.”
- Algorithm #2
- Related work

Local linearity

A manifold is locally linear, even if globally nonlinear.

How can we use this?



Previous work

- Cluster inputs, then perform PCA:

  k-lines,
  k-planes,
  local PCA,
  mixture models,…

- Problem solved? No!
  - No global coordinates.
  - Prone to local optima.
  - Iterative optimizations.

Locally Linear Embedding (LLE)

- Steps
  1. Nearest neighbor search.
  2. Least squares fits.
  3. Sparse eigenvalue problem.
- Properties
  - Obtains highly nonlinear embeddings.
  - Non-iterative, not prone to local minima.

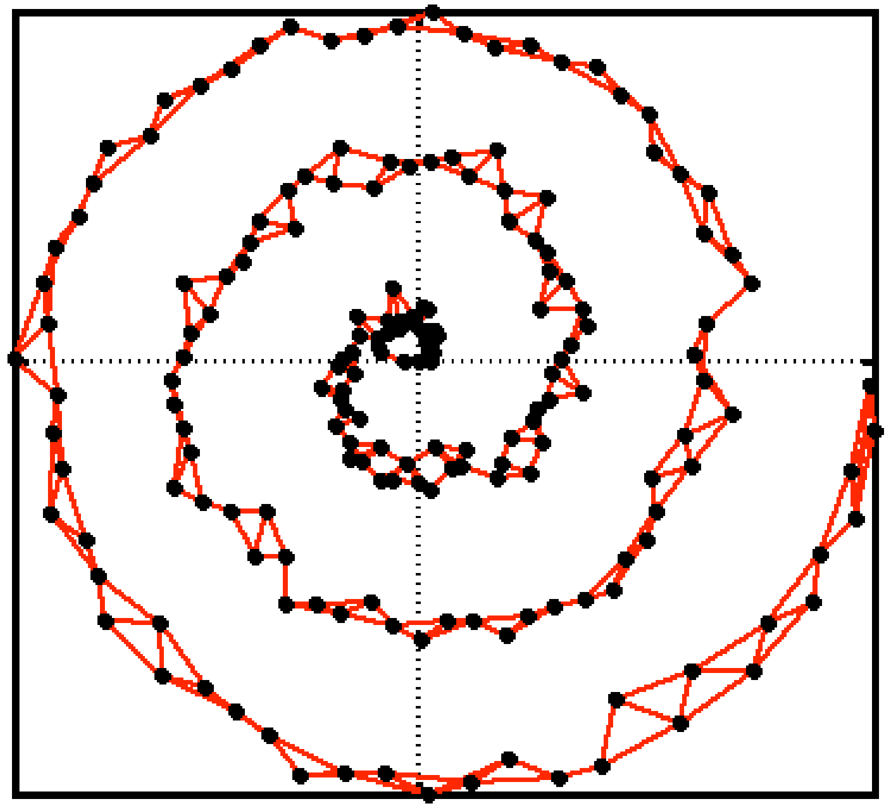
Step 1. Identify neighbors.

- 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

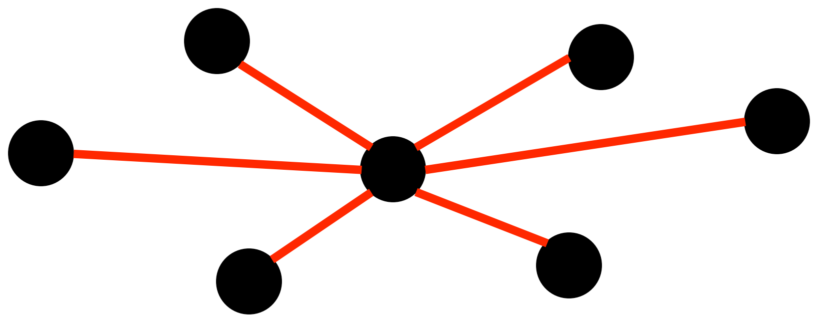
Assumptions:

- 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:

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

Least squares fits

- **Choose weights to minimize errors:**

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

- **Constraints:**

Nonzero W_{ij} only for neighbors.
Weights must sum to one: $\sum_j W_{ij} = 1$

Symmetry

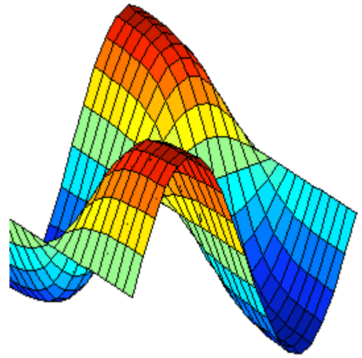
- **Cost per input**

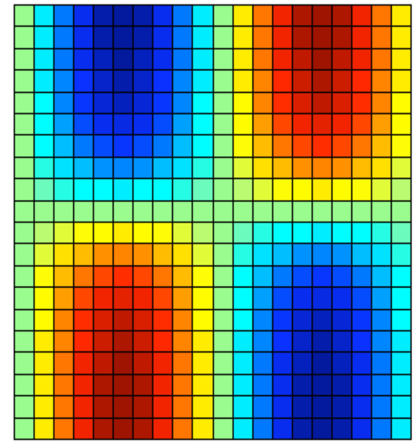
$$\varepsilon_i(W) = \left| \vec{X}_i - \sum_j W_{ij} \vec{X}_j \right|^2$$

- **Local invariance**

Optimal weights W_{ij} are invariant to rotations, translations, and rescalings.

Manifolds





- **Local linearity**

  Each neighborhood map looks like a translation, rotation, and rescaling.

- **Local geometry**

  These transformations do not affect the weights W_{ij}: they remain valid.

Step 3. Compute the embedding.

- **Embedding**
  Map inputs to outputs: $\vec{X}_i \in \mathbb{R}^D$ to $\vec{Y}_i \in \mathbb{R}^d$

- **Minimize reconstruction errors.**
  Optimize outputs Y_i for fixed weights W_{ij}:

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

- **Constraints**
  Center outputs on origin: $\sum_i \vec{Y}_i = \vec{0}.$
  Impose unit covariance matrix: $\frac{1}{N} \sum_i \vec{Y}_i \vec{Y}_i^T = I_d.$

Sparse eigenvalue problem

- **Quadratic form**

$$\Phi(Y) = \sum_{ij} A_{ij}\left(\vec{Y}_i \bullet \vec{Y}_j\right) \text{ with } A = (I - W^T)(I - W)$$

- **Rayleigh-Ritz theorem**

Optimal embedding given by bottom d+1 eigenvectors.

- **Solution**

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

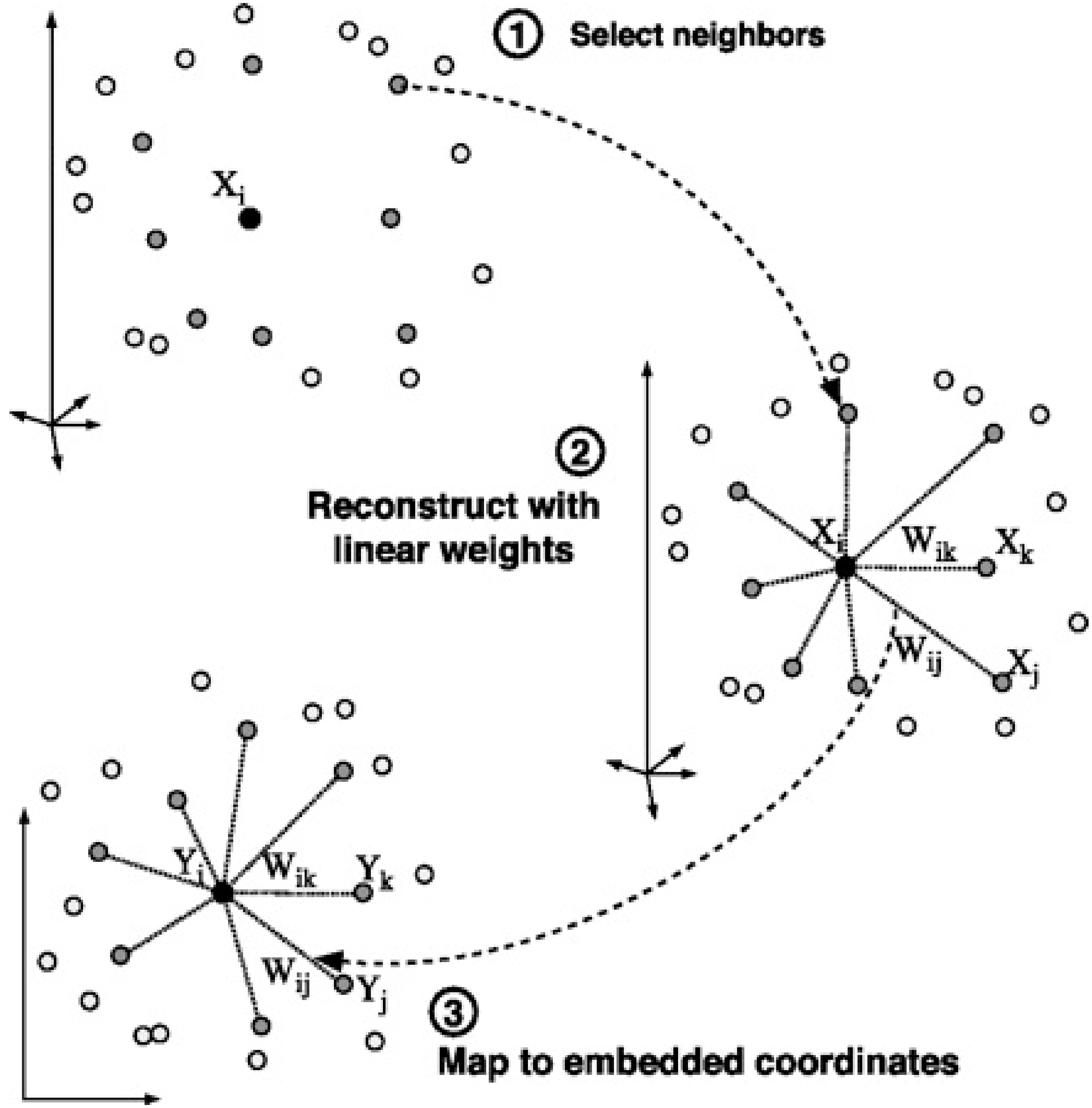
Summary of LLE

- **Three steps**
  1. **Compute K nearest neighbors.**
  2. **Compute weights W_{ij}.**
  3. **Compute outputs Y_i.**
- **Optimizations**

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

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

Locally Linear Embedding

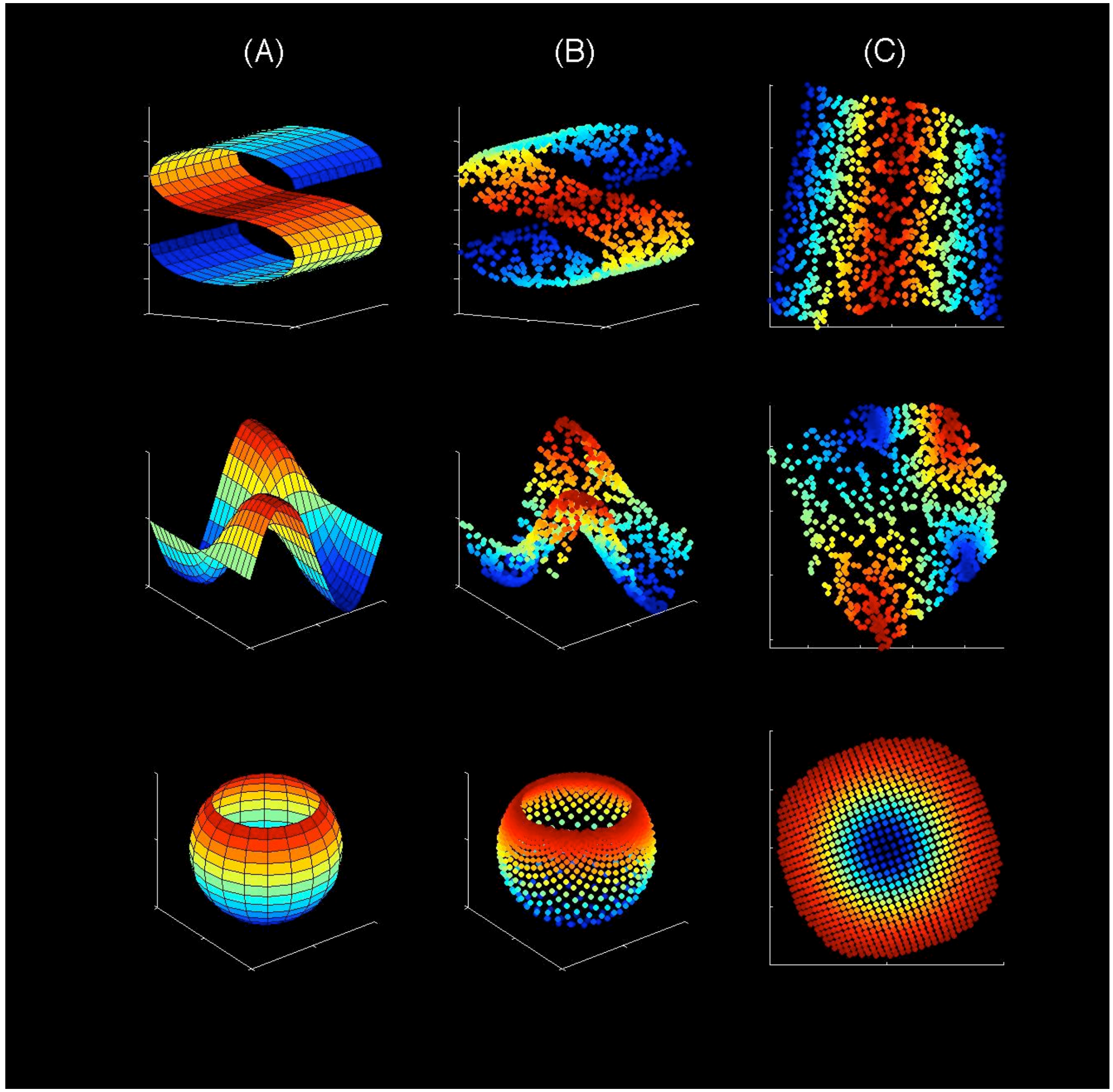


Surfaces

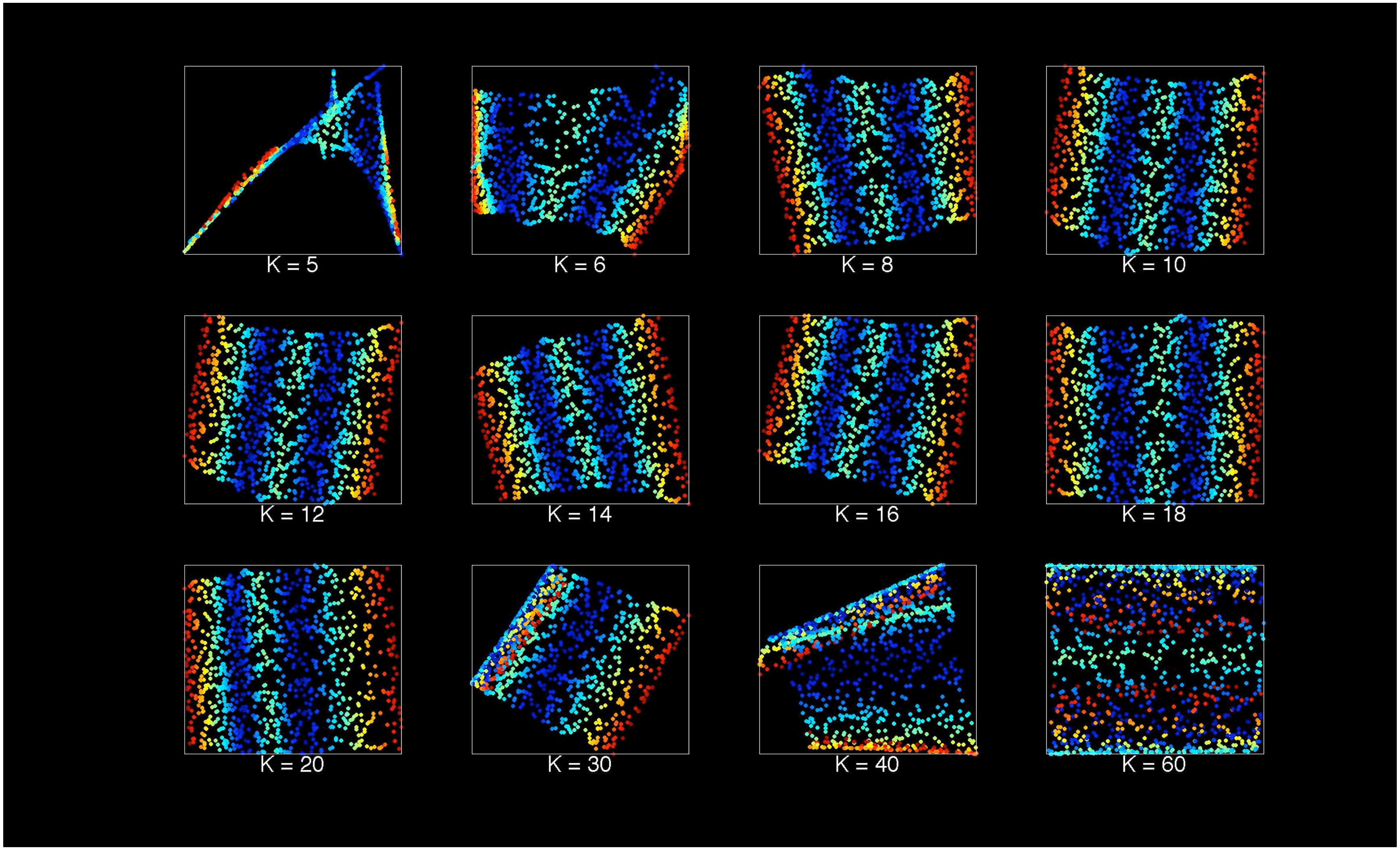
N=1000
inputs

K=8
neighbors

D=3
d=2



Effect of neighborhood size



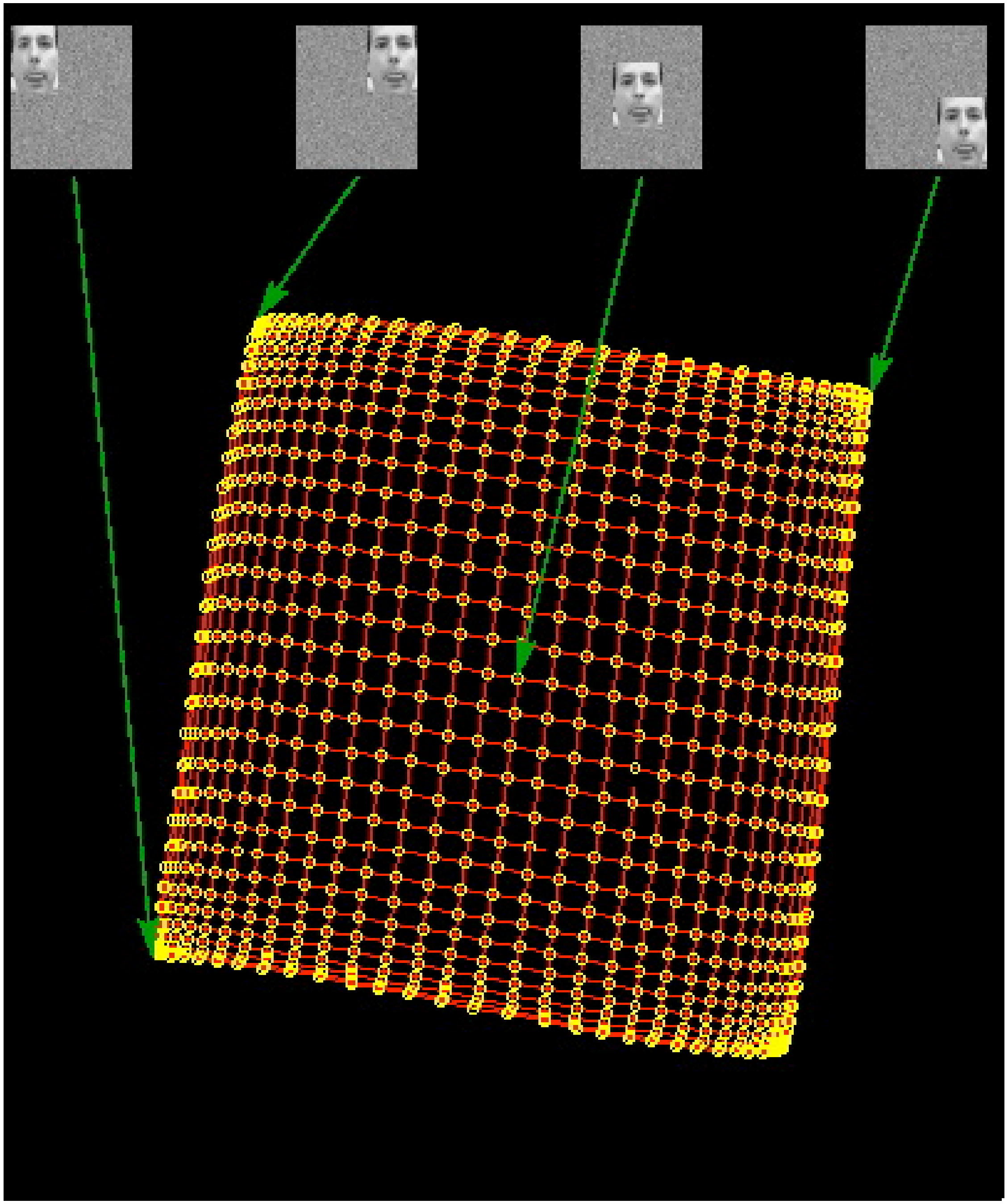
Translated faces

N=961 images

K=4 neighbors

D=3009 pixels

d=2 manifold



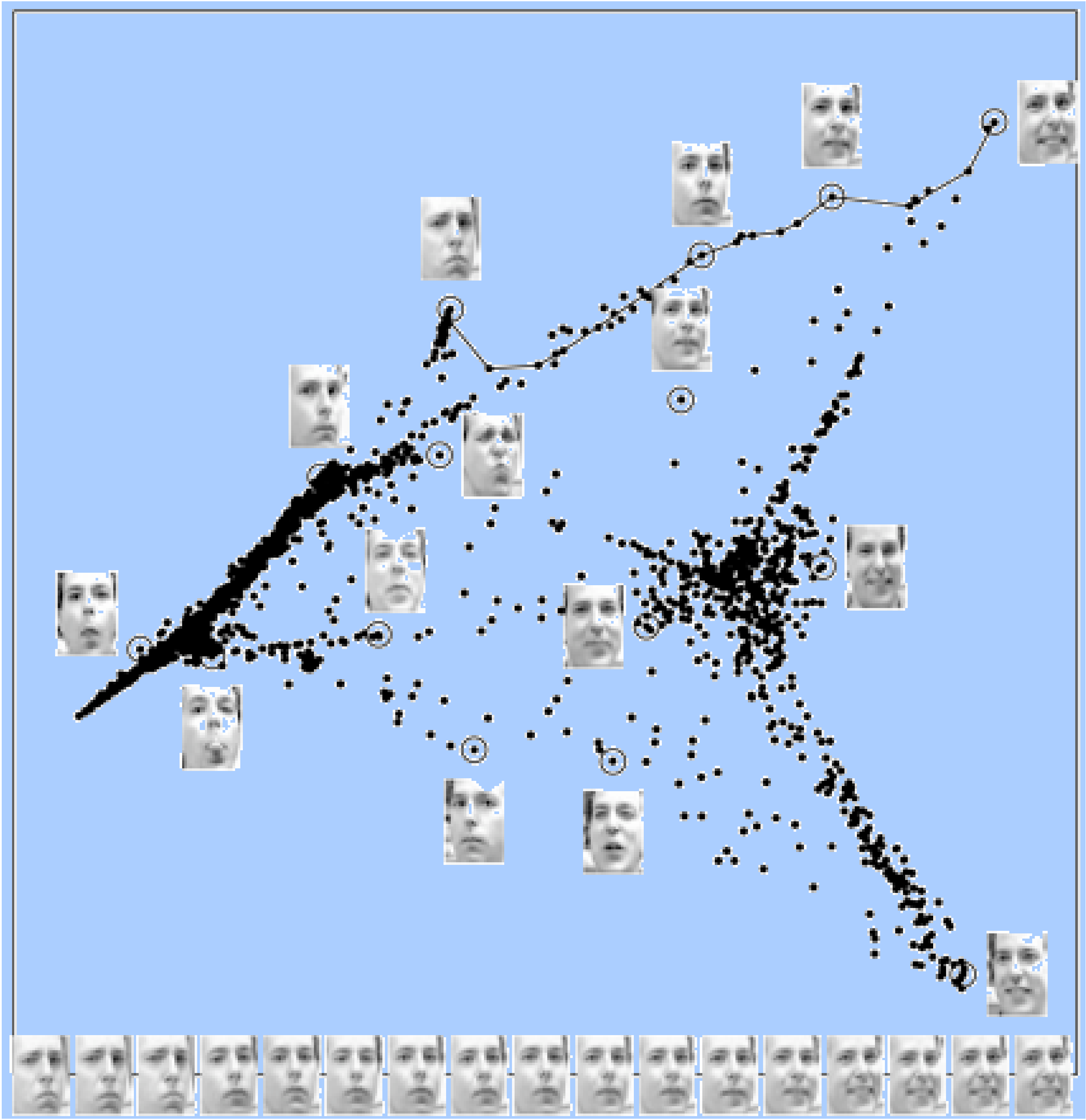
Pose and expression

N=1965 images

K=12 neighbors

D=560 pixels

d=2 (shown)



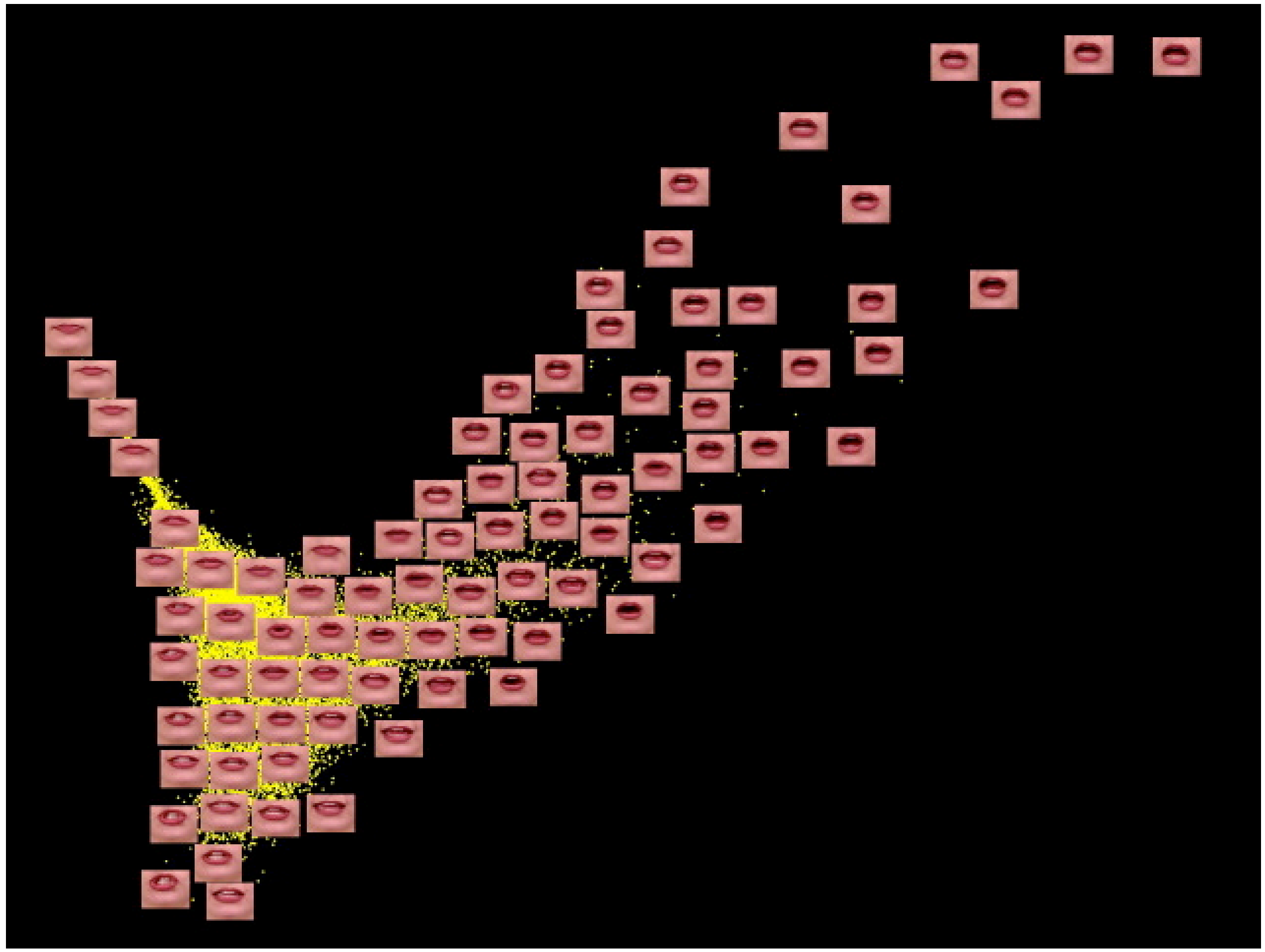
Lips

N=15960
images

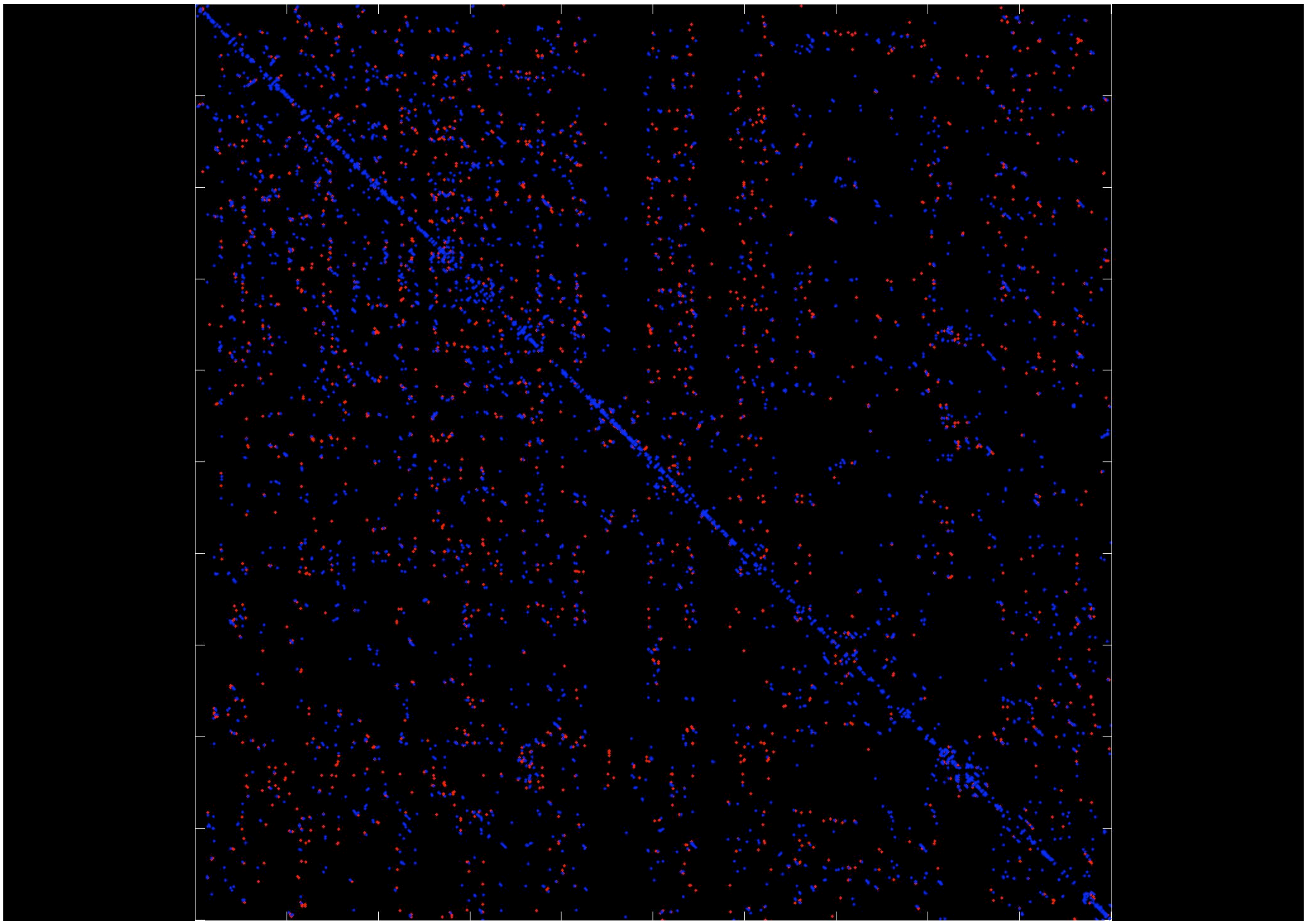
K=24
neighbors

D=65664
pixels

d=2
(shown)

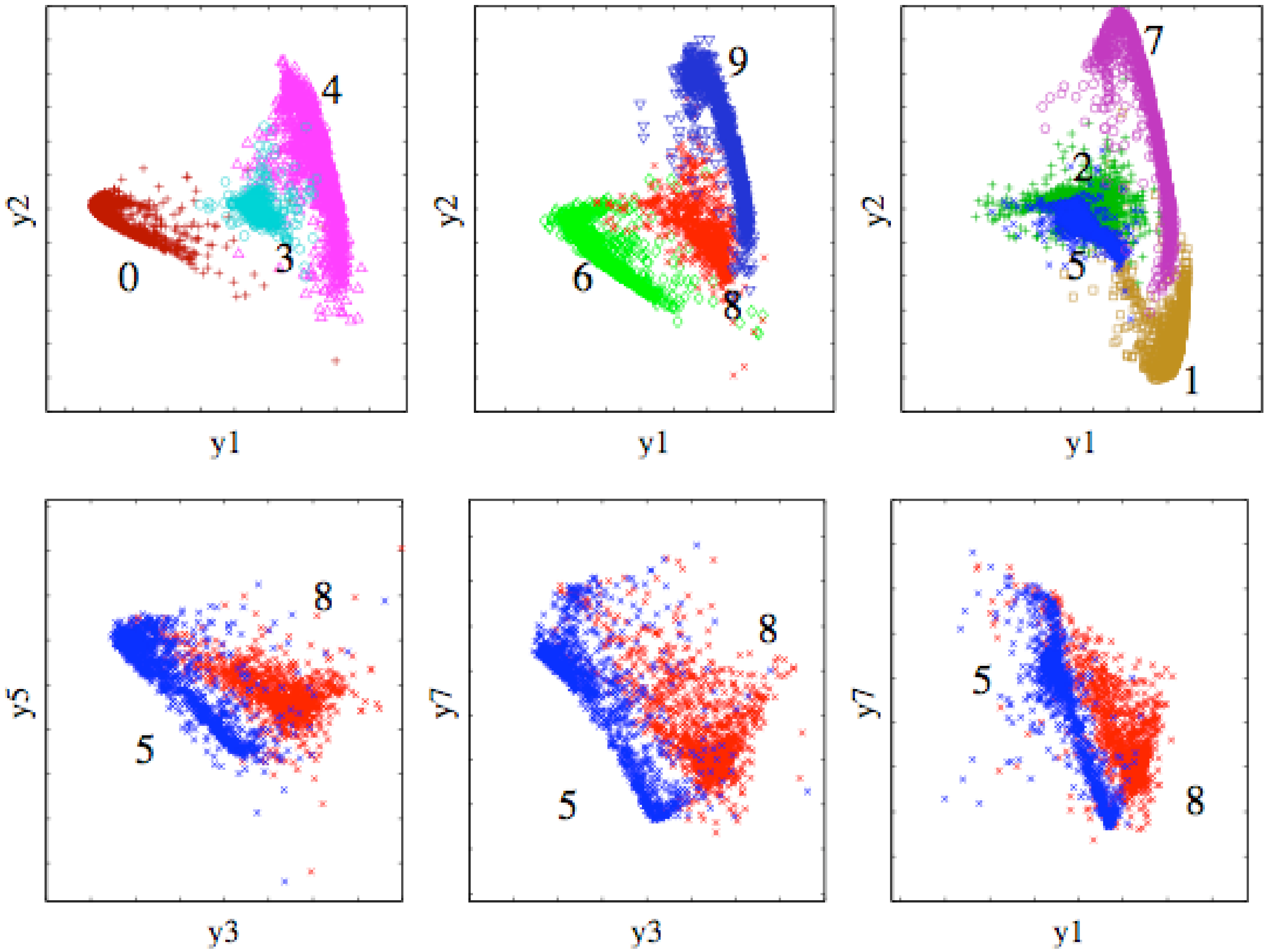


Sparseness of the weight matrix



Handwritten digits

N=11000 images
C=10 classes
D=256 pixels
d=8



Digit classes are naturally clustered.

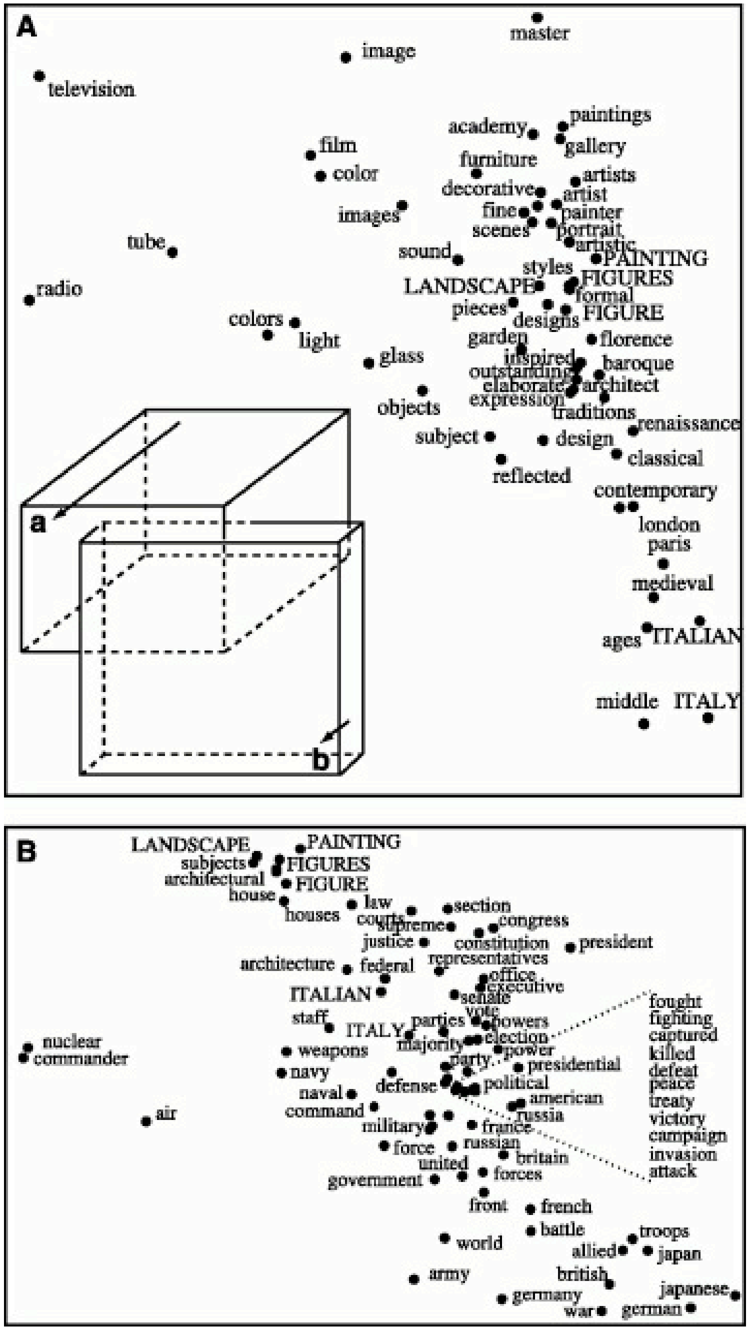
Word document counts

N=5000
words

K=20
neighbors

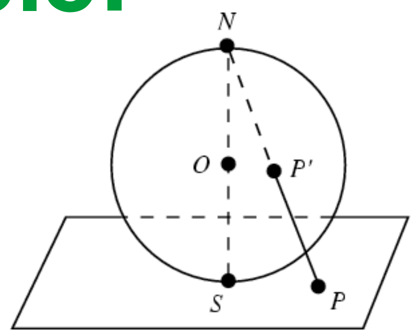
D=31000
documents

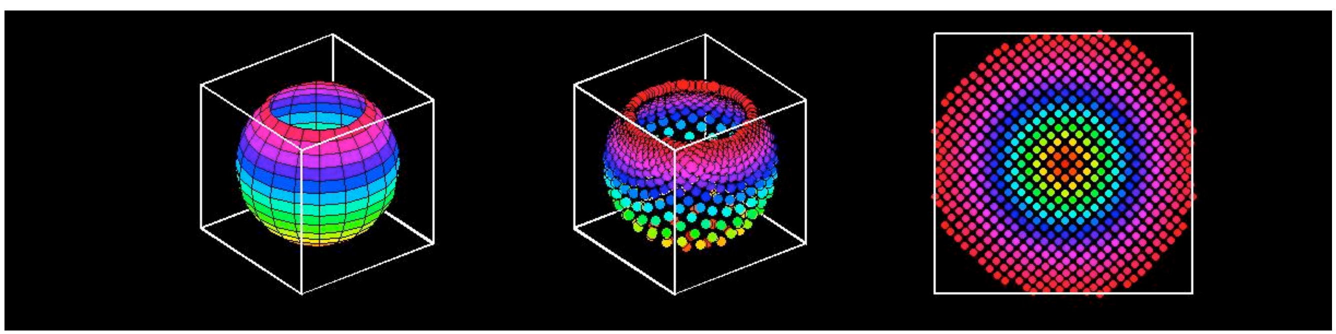
d=3,4,5
(shown)



Symmetries of LLE

- Conformal transformations
  - Angle-preserving mappings.
  - Local scaling, rotation, and translation.
- Example:





Summary of LLE

- Three steps:
  1. K nearest neighbors of inputs X_i.
  2. Least squares fits for weights W_{ij}.
  3. Sparse eigensystem for outputs Y_i.
- Local symmetries:
  - translation
  - rotation
  - rescaling

"Think globally, fit locally"

Outline

- Motivation
- Algorithm #1: LLE
- Algorithm #2: SDE
  How to unfold a manifold...
- Related work

Beyond linearity…

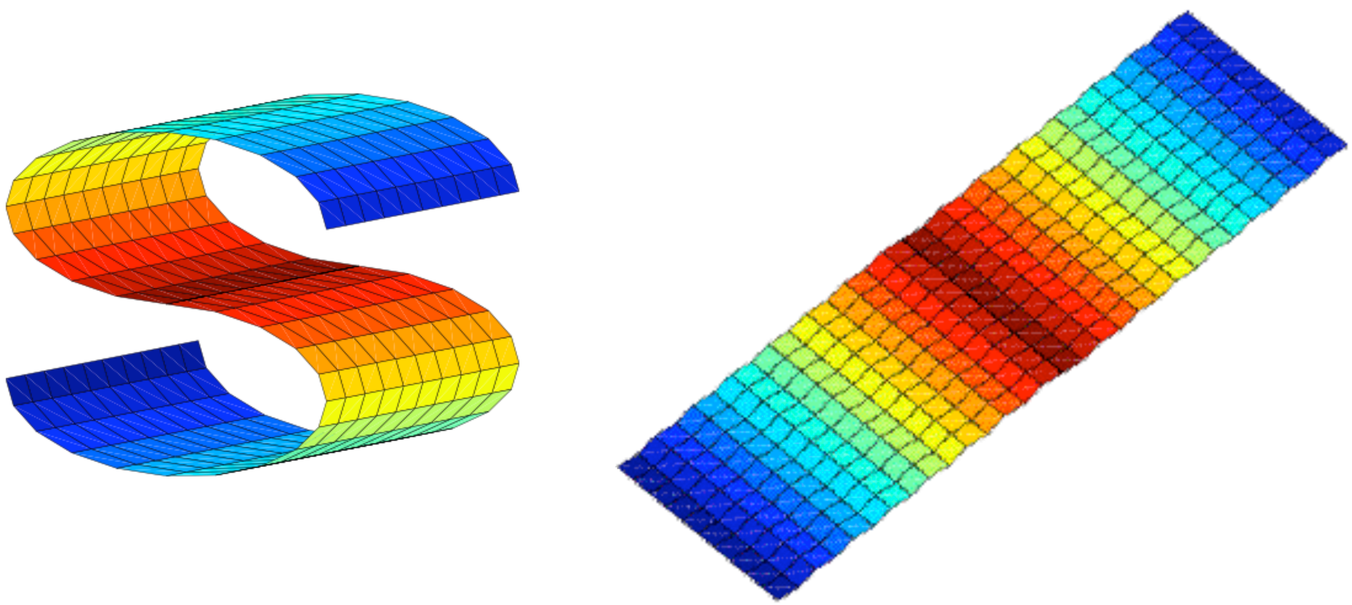
What larger class of mappings:

- Includes rotations and translations as a special case?
- Unravels manifolds into subsets of Euclidean space?

Isometry

- Intuitively

  Whatever you can do to a sheet of paper without holes, tears, or self-intersections.



Isometry (con't)

- Informally

  A smooth, invertible mapping that preserves distances and looks *locally* like a rotation plus translation.

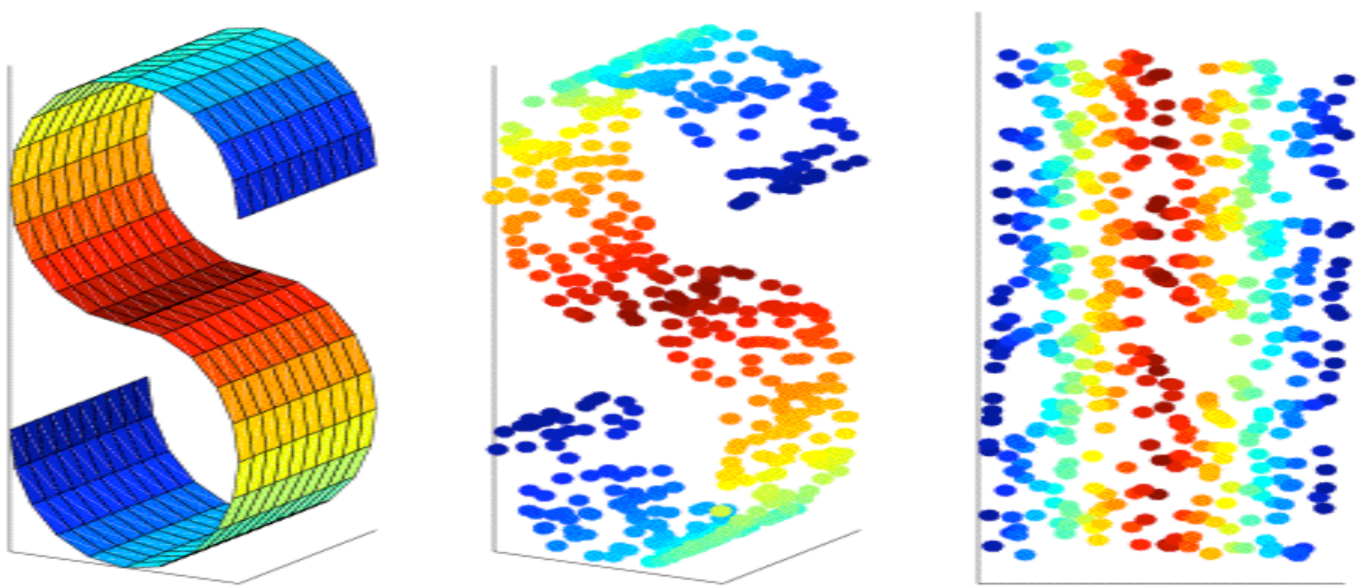
- Formally

  Two Riemannian manifolds are isometric if there is a diffeomorphism that pulls back the metric on one to the other.

Data on manifolds

From the continuous to the discrete:

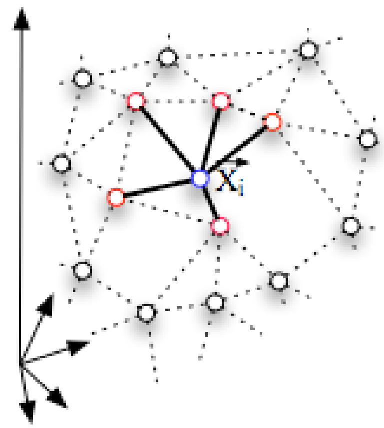
Isometry is defined between manifolds.
Can we extend the relation to data sets?



Discretely sampled manifolds

- **Neighborhood graph**

  Connect each point to its k nearest neighbors.



- **Locally isometric**

  Consider an embedding *Y* of *X* locally isometric if:

$$\left(\vec{Y}_i - \vec{Y}_j\right) \bullet \left(\vec{Y}_i - \vec{Y}_k\right) = \left(\vec{X}_i - \vec{X}_j\right) \bullet \left(\vec{X}_i - \vec{X}_k\right)$$

for all $\vec{X}_i$ with neighbors $\vec{X}_j$ and $\vec{X}_k$.

Dot product constraints

- **Gram matrices**

$$G_{ij} = \vec{X}_i \bullet \vec{X}_j \quad \text{(inputs)}$$

$$K_{ij} = \vec{Y}_i \bullet \vec{Y}_j \quad \text{(outputs)}$$

- **Locally isometric**

Consider an embedding *Y* of *X* locally isometric if:

$$K_{ii} - K_{ij} - K_{ik} + K_{jk} = G_{ii} - G_{ij} - G_{ik} + G_{jk}$$

for all $\vec{X}_i$ with neighbors $\vec{X}_j$ and $\vec{X}_k$.

Manifold learning

- **Input**

  Vectors $\vec{X}_i$ and Gram matrix $G_{ij} = \vec{X}_i \bullet \vec{X}_j$; latter determines former up to rotation.

- **Problem**

  Given $G_{ij} = \vec{X}_i \bullet \vec{X}_j$, how to construct $K_{ij} = \vec{Y}_i \bullet \vec{Y}_j$ such that Y "unfolds" the manifold of X?

- **Algorithm**

  What to optimize?
  What to constrain?

Constraints on K_{ij}

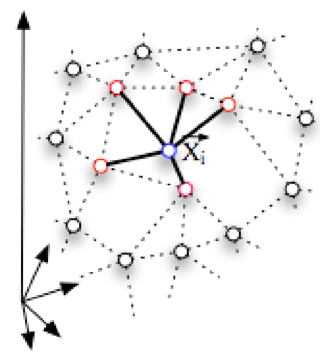
- **Centered**

  Constrain outputs to have zero mean:

$$\sum_i \vec{Y}_i = \vec{0} \quad \text{implies} \quad \left|\sum_i \vec{Y}_i\right|^2 = \sum_{ij} \vec{Y}_i \bullet \vec{Y}_j = \boxed{\sum_{ij} K_{ij} = 0}$$

- **Locally isometric**

  Preserve local angles and distances:

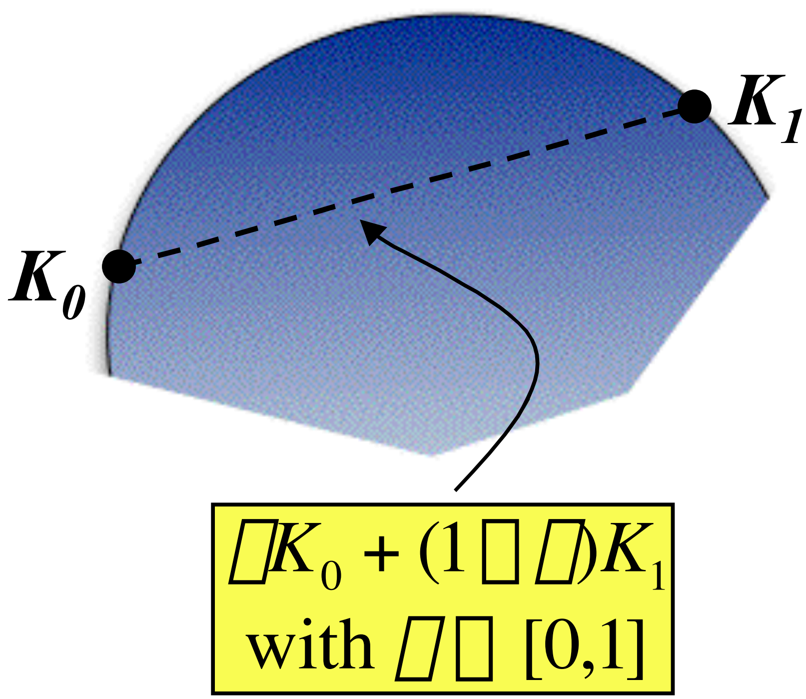


$$\boxed{K_{ii} - K_{ij} - K_{ik} + K_{jk} = G_{ii} - G_{ij} - G_{ik} + G_{jk}}$$

Constraints (con’t)

- Semidefinite

Eigenvalues of K must be nonnegative.



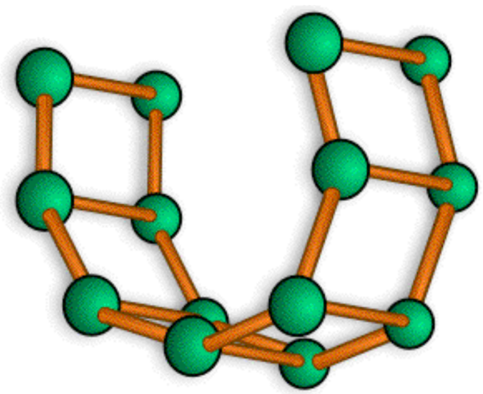
Semidefinite and linear constraints are convex.

***O(Nk²)* constraints**

***O(N²)* variables**

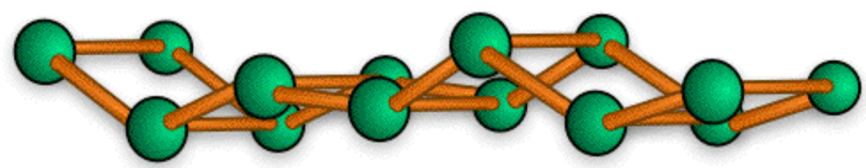
Unfolding a manifold

What function of the Gram matrix is being optimized below?



Before

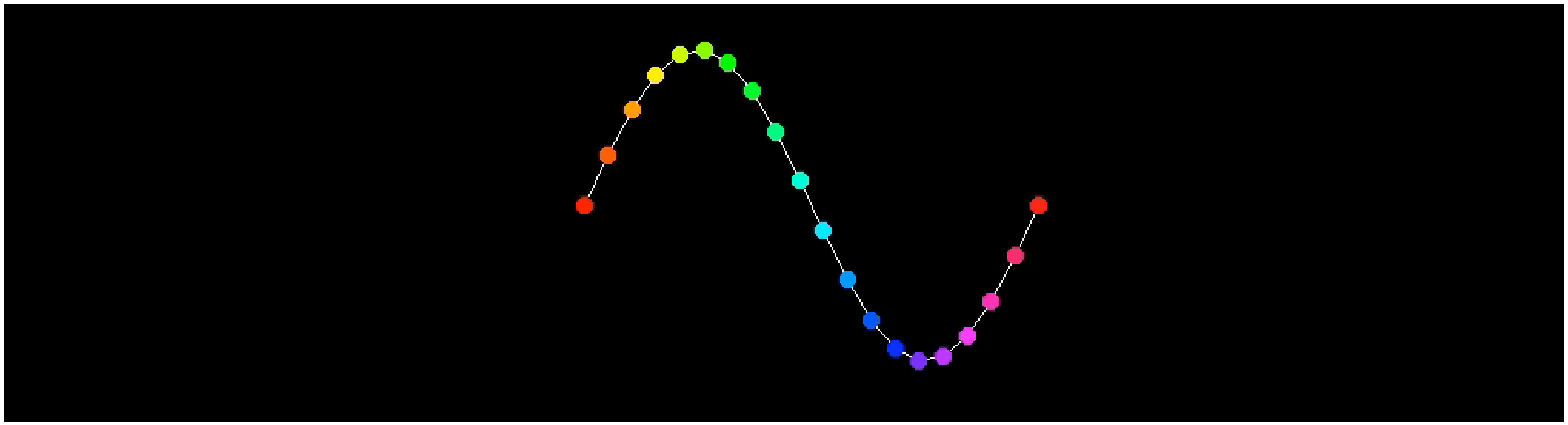
$$G_{ij} = \vec{X}_i \cdot \vec{X}_j$$



After

$$K_{ij} = \vec{Y}_i \cdot \vec{Y}_j$$

What is increasing?



Multiple choice:

(a) Pairwise distances
(b) Number of zero eigenvalues
(c) Trace of Gram matrix
(d) All of the above

Optimization

- Pull points apart

  Maximize sum of pairwise distances, same as var(*Y*) or trace(*K*):

$$\frac{1}{2N}\sum_{ij}\left|\vec{Y}_i - \vec{Y}_j\right|^2 = \sum_i \left|\vec{Y}_i\right|^2 = \sum_i K_{ii}$$

  (Similar intuition as PCA.)

- Boundedness

  Follows from triangle inequality and connectedness of neighborhood graph.

Semidefinite programming

Maximize trace(*K*) subject to:

(i) $K \geq 0,$

(ii) $\sum_{ij} K_{ij} = 0,$

(iii) for all neighborhoods (ijk),

$$K_{ii} - K_{ij} - K_{ik} + K_{jk}$$
$$= G_{ii} - G_{ij} - G_{ik} + G_{jk}$$

Convex optimization

- **Solution**

  Feasible region is convex.

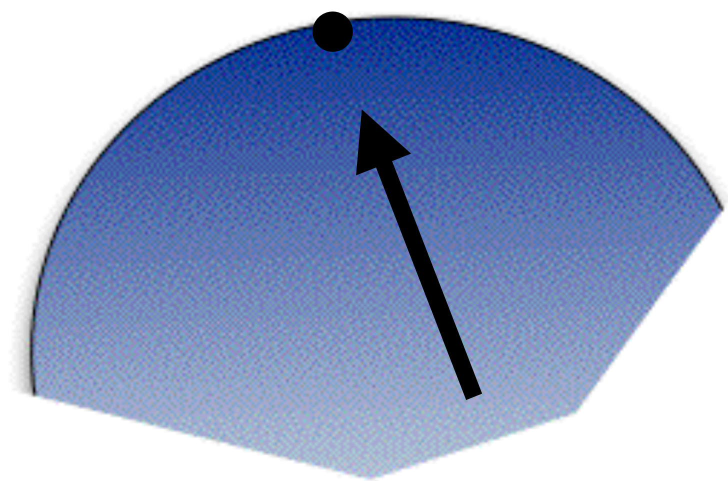
  Never empty (includes *G*).

  Objective is linear and bounded.

  Efficient algorithms exist.

- **Caveat**

  Generic solvers scale poorly.



Algorithm

1) **K nearest neighbors**

Compute nearest neighbors, distances and angles.

2) **Semidefinite programming**

Maximize trace of centered, locally isometric Gram matrices.

3) **Matrix diagonalization**

Estimate *d* from eigenvalues.
Top eigenvectors give embedding.

Name of algorithm?

Locally Isometric Kernel Matrix Embedding

Name of algorithm?

Locally Isometric Kernel Matrix Embedding

L I C K M E

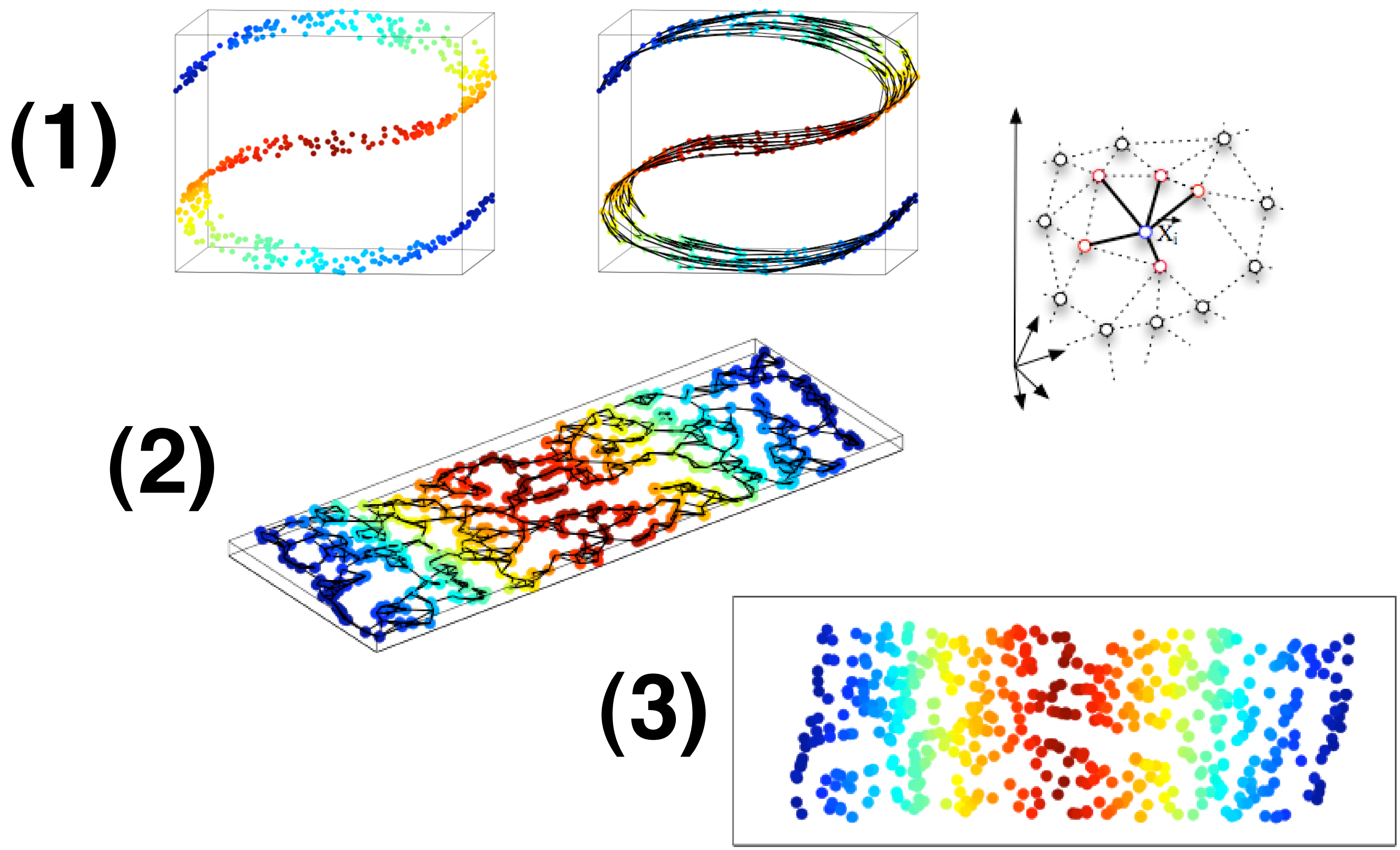
Technically accurate, but…

Name of algorithm?

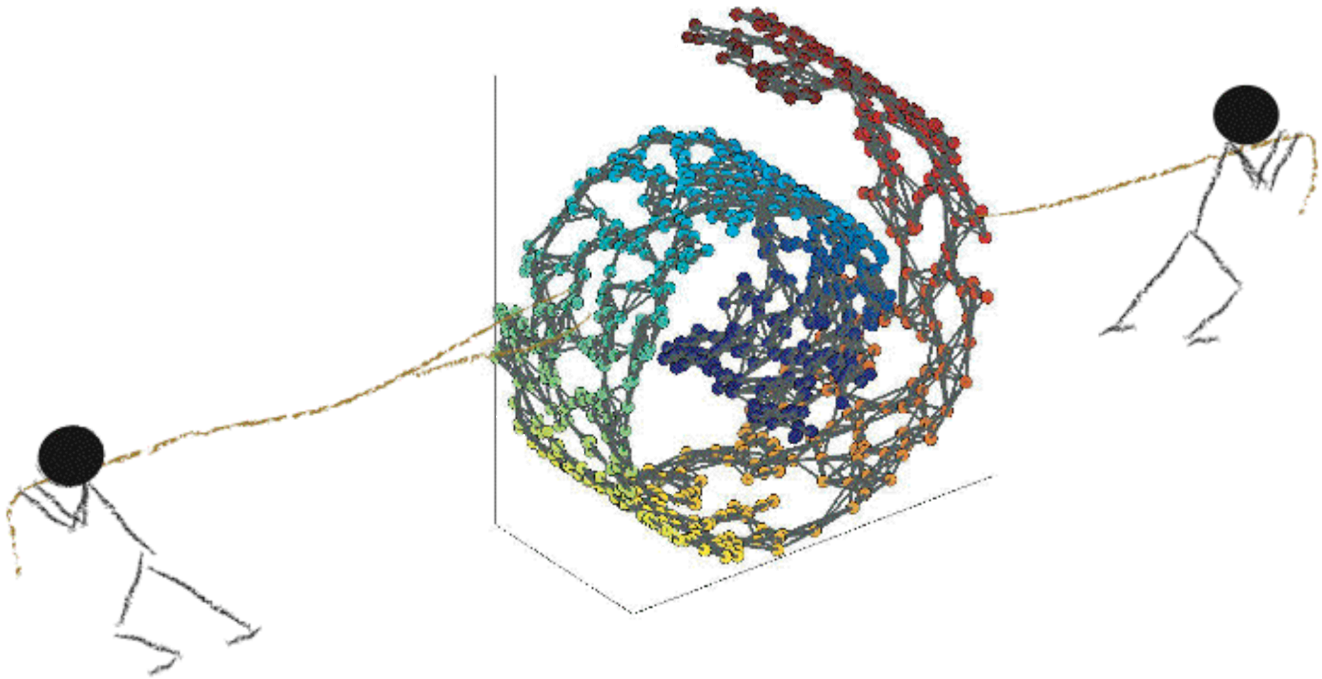
Semidefinite Embedding

SDE

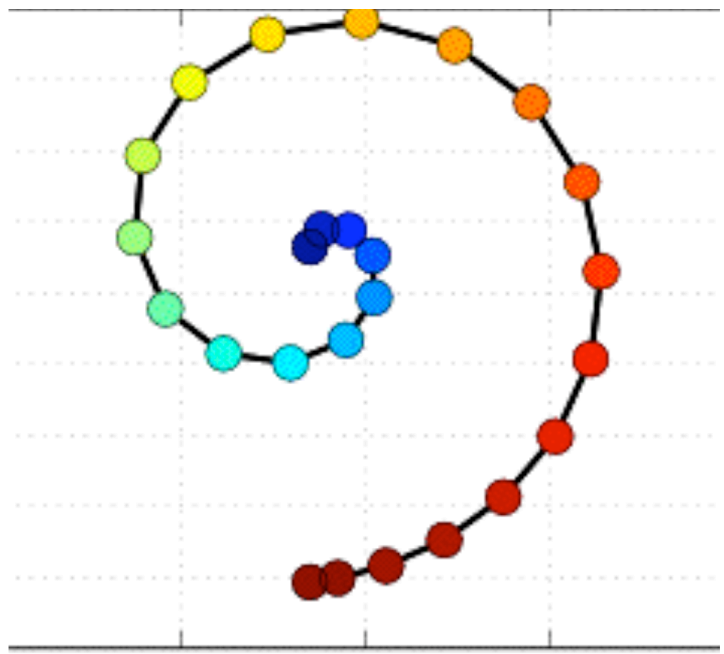
Semidefinite Embedding



Experimental Results



Spiral



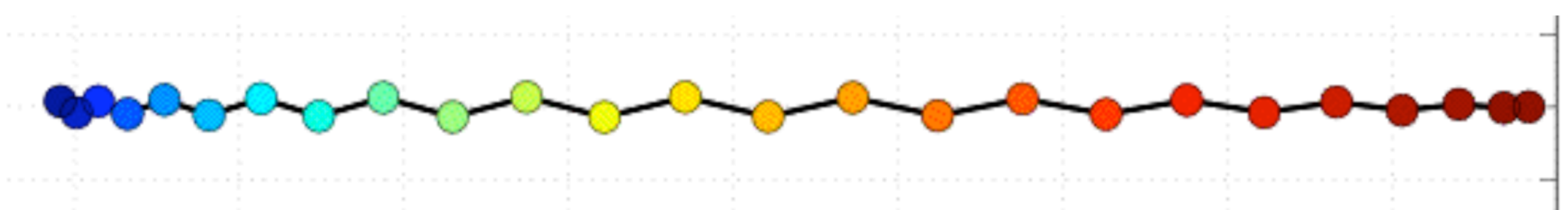
$$N = 25$$

$$k = 2$$

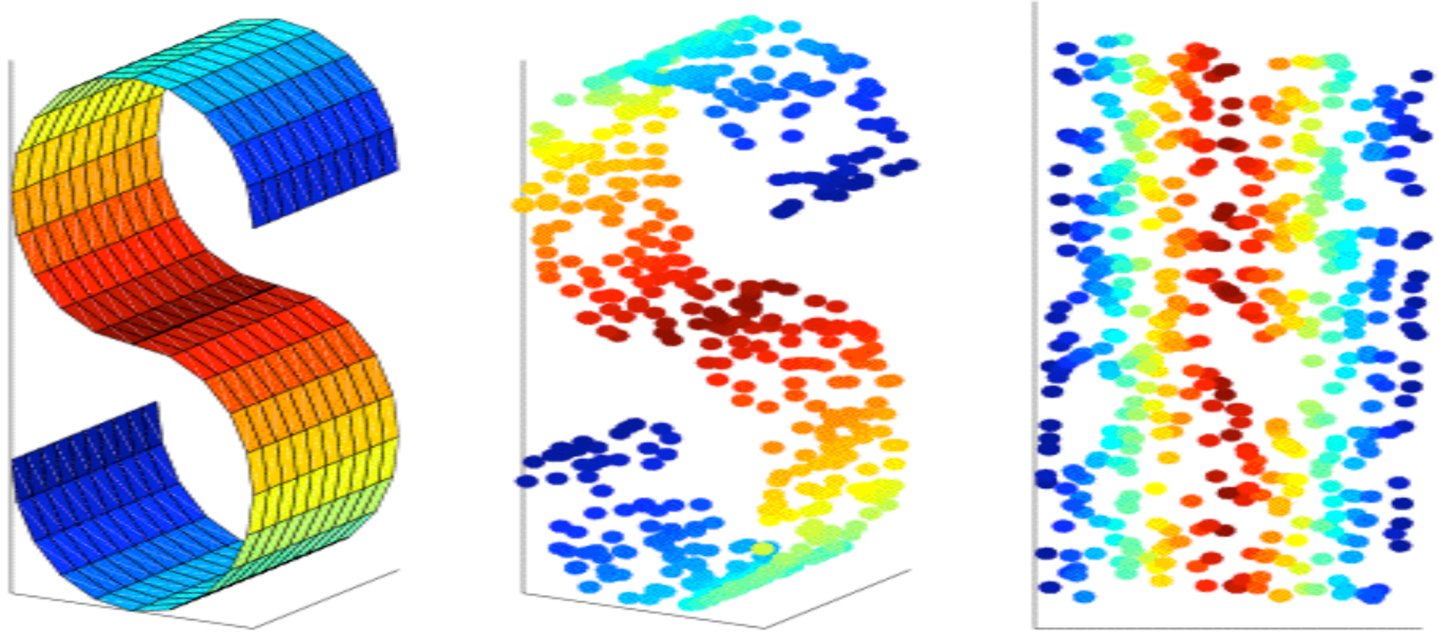
$$D = 2$$

$$d = 1$$

$$\frac{\lambda_1}{\lambda_2} > 10^3$$



S Manifold



$N = 500$

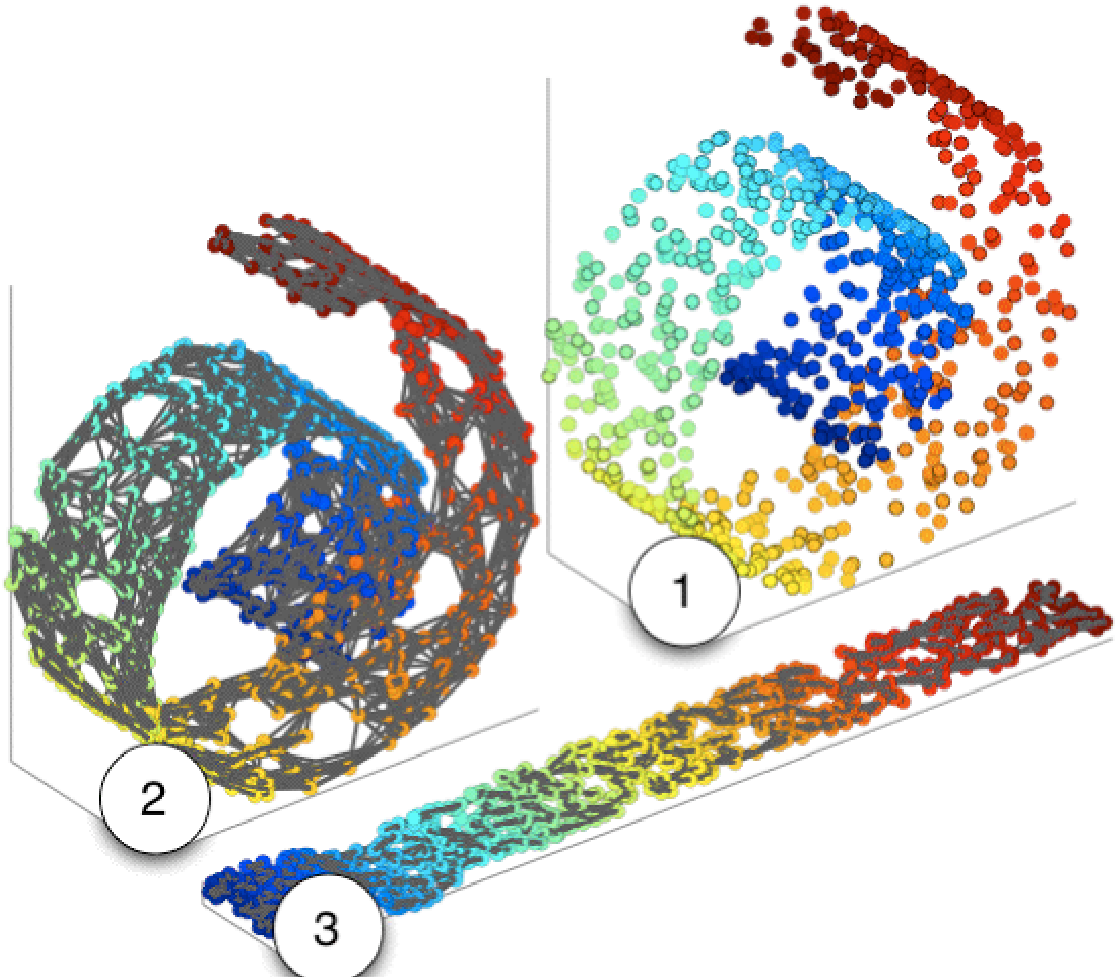
$k = 4$

$D = 3$

$d = 2$

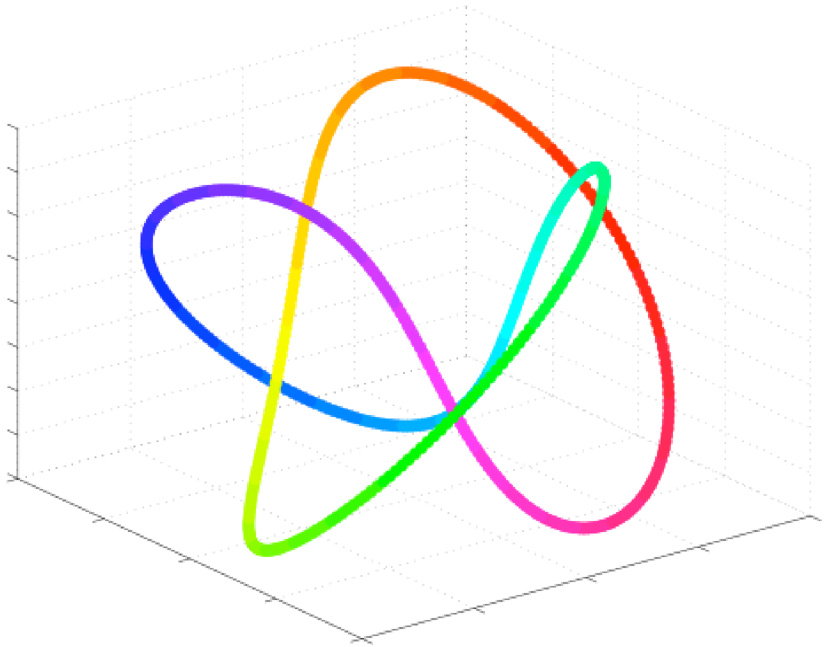
$\frac{□_1}{□_3}$ □ 45

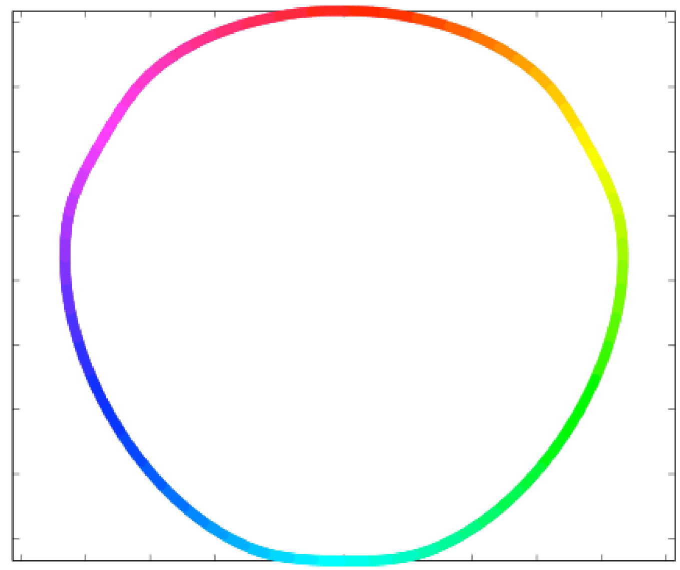
Swiss Roll



$$N = 800$$
$$k = 6$$
$$D = 8$$

Trefoil knot





$$N = 539$$
$$k = 4$$
$$D = 3$$

Teapot (half rotation)

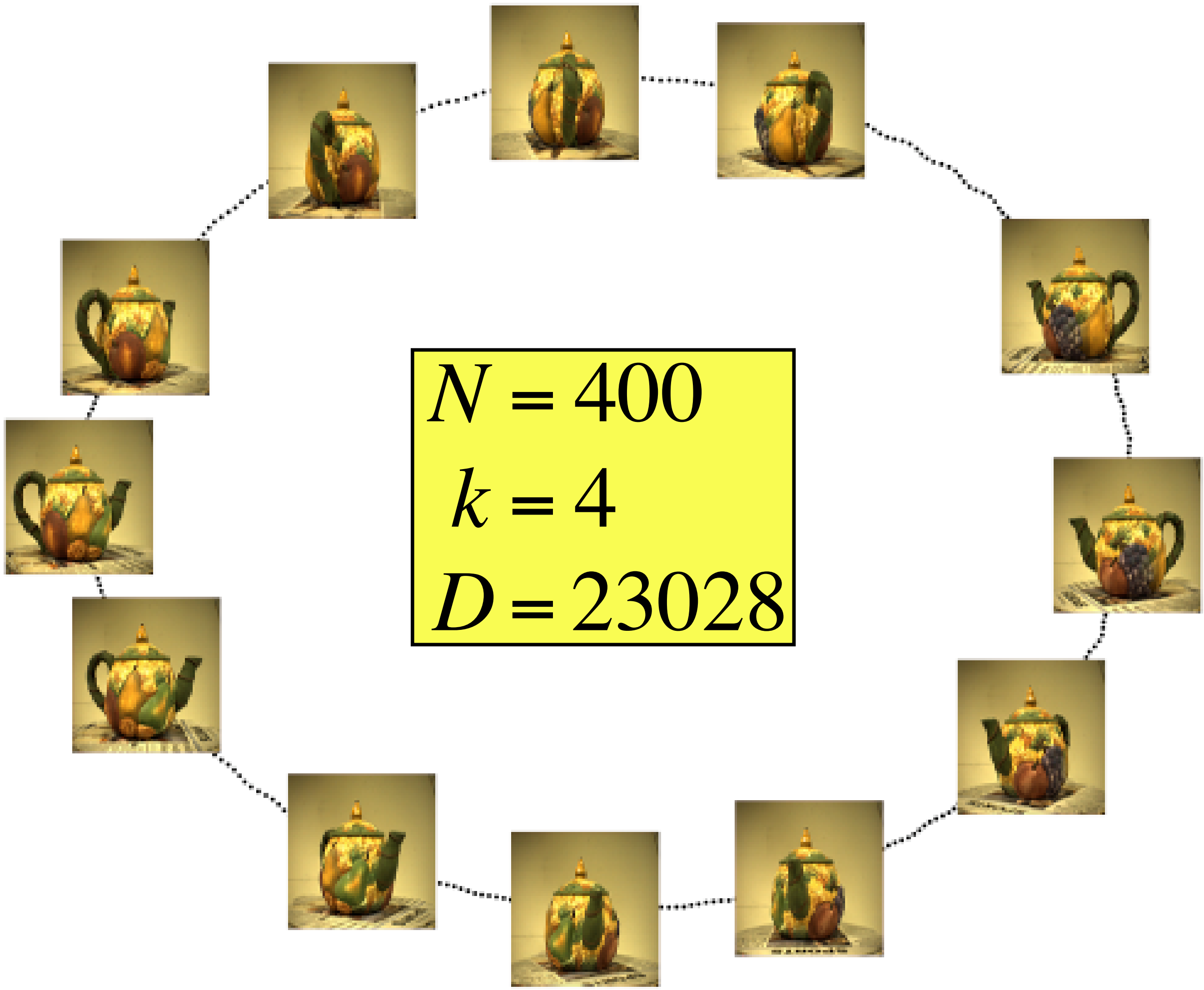
Images ordered by one dimensional embedding

$N = 200$

$k = 4$

$D = 23028$

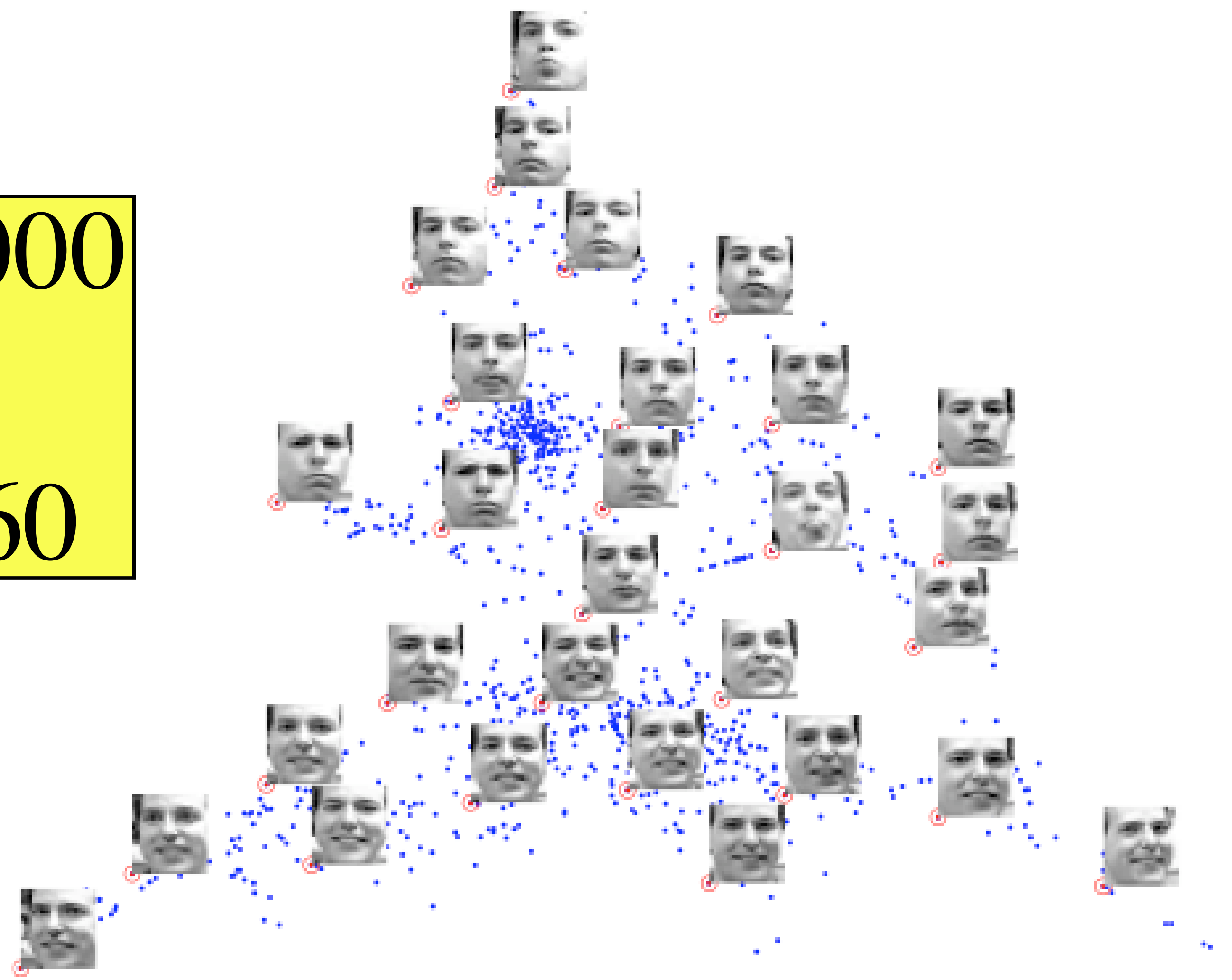
Teapot (full rotation)



$$N = 400$$
$$k = 4$$
$$D = 23028$$

Images of faces

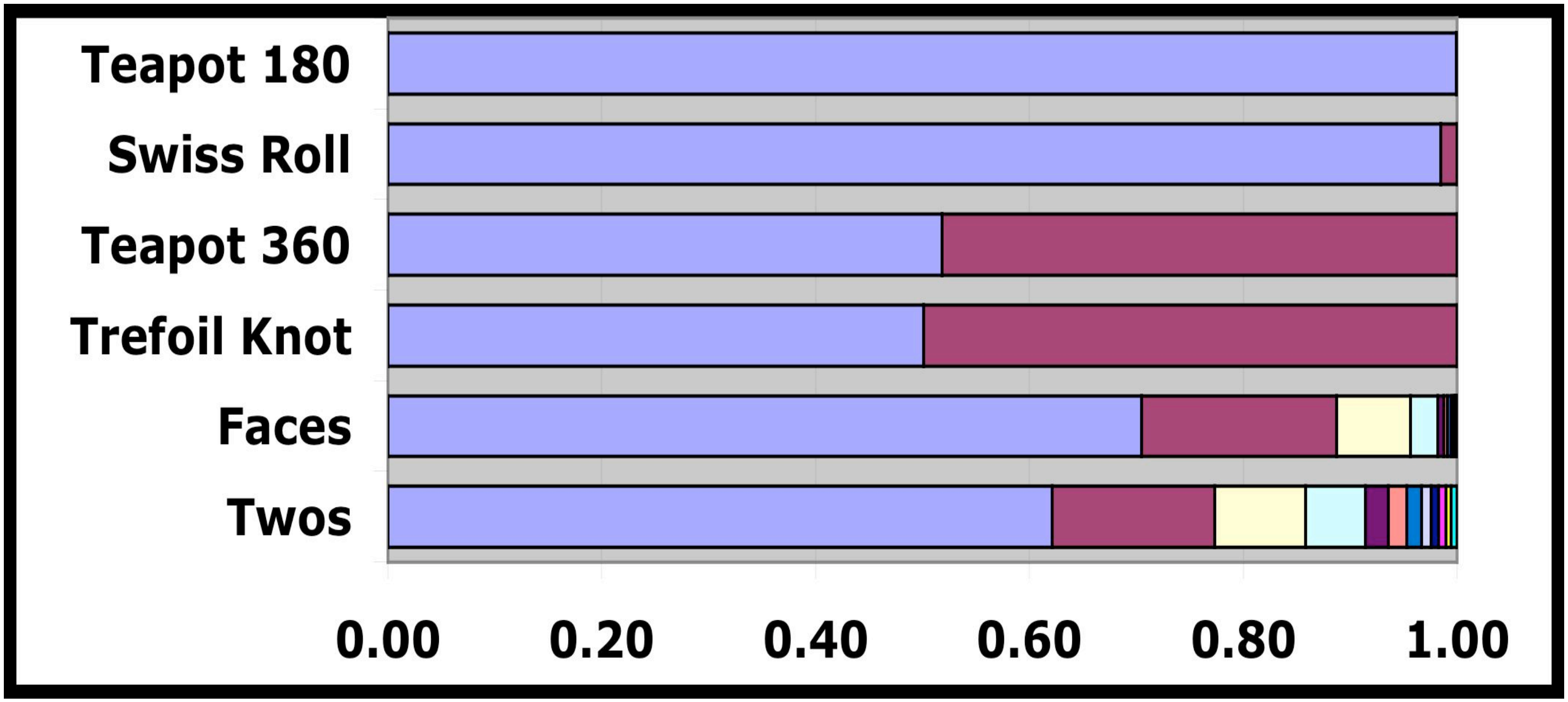
$$N = 1000$$
$$k = 4$$
$$D = 560$$



Handwritten digits

$$N = 638$$
$$k = 4$$
$$D = 256$$

Eigenvalues



(normalized by trace)

Evaluating SDE

- **Pros**
  - Eigenvalues reveal dimensionality.
  - Constraints ensure local isometry.
  - Algorithm tolerates small data sets.
- **Cons**
  - Computation intensive.
  - Currently limited to $N \leq 2000$, $k \leq 6$.

Outline

- Motivation
- Algorithm #1: LLE
- Algorithm #2: SDE
- Comparisons and related work

LLE vs SDE

- **Sparse vs dense**

  LLE constructs a sparse matrix.
  SDE constructs a dense matrix.

- **Bottom vs top**

  LLE computes bottom eigenvectors.
  SDE computes top eigenvectors.

- **Angle vs distance-preserving**

  LLE motivated by conformal maps.
  SDE motivated by isometric maps.

- **Estimating the dimensionality**

  LLE eigenvalues do not reveal *d*.
  SDE eigenvalues do reveal *d*.

Other methods

- **Kernel PCA**

  Map inputs nonlinearly to a new space, then perform PCA.

- **Isomap**

  Measure pairwise distances along manifold, then apply MDS.

- **Laplacian eigenmaps**

  Preserve nearness relations as encoded by graph Laplacian.

Manifold learning

- **1998** Isomap
- **2000** Locally Linear Embedding (LLE)
- **2002** Laplacian eigenmaps
- **2004** Semidefinite Embedding (SDE)

Common framework:

1) Compute nearest neighbors.
2) Construct an N x N matrix.
3) Compute eigenvectors.

Comparison

Algorithm	Mapping	Signature	Matrix
Isomap	isometric	geodesics	dense
SDE	isometric	local distances	dense
hLLE	isometric	hessians	dense
LLE	conformal	tangents	sparse
Laplacian eigenmaps	proximity-preserving	Laplacian	sparse

Conclusion

- Big ideas
  - Manifolds are everywhere.
  - Spectral methods can learn them.
- Ongoing work
  - Scaling up to larger data sets
  - Theoretical guarantees
  - Alternative topologies
  - Extrapolation and functional maps