CS49000-VIZ - Fall 2020

Introduction to Data Visualization

Dimensionality Reduction

Lecture 16

November 12, 2020

Outline

High-dimensional data

Dimensionality reduction

Manifold learning

Computational aspects

Why High-Dimensional?

- Data samples with many attributes
  - Tables with many columns, medical records, ...
- Amalgamate individual properties into high-dimensional *feature* vectors

Challenges

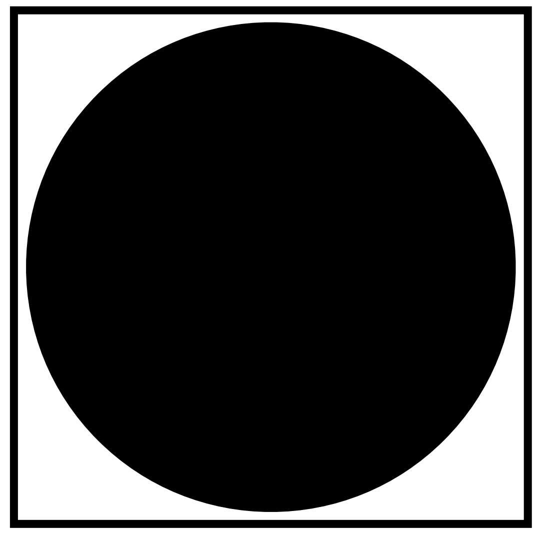
- "Curse of dimensionality"
  - Sampling becomes exponentially costly
  - Space accumulates in"corners" of hypercube
  - Data processing becomes extremely expensive / intractable
- How to visualize?
  - Missing intuition

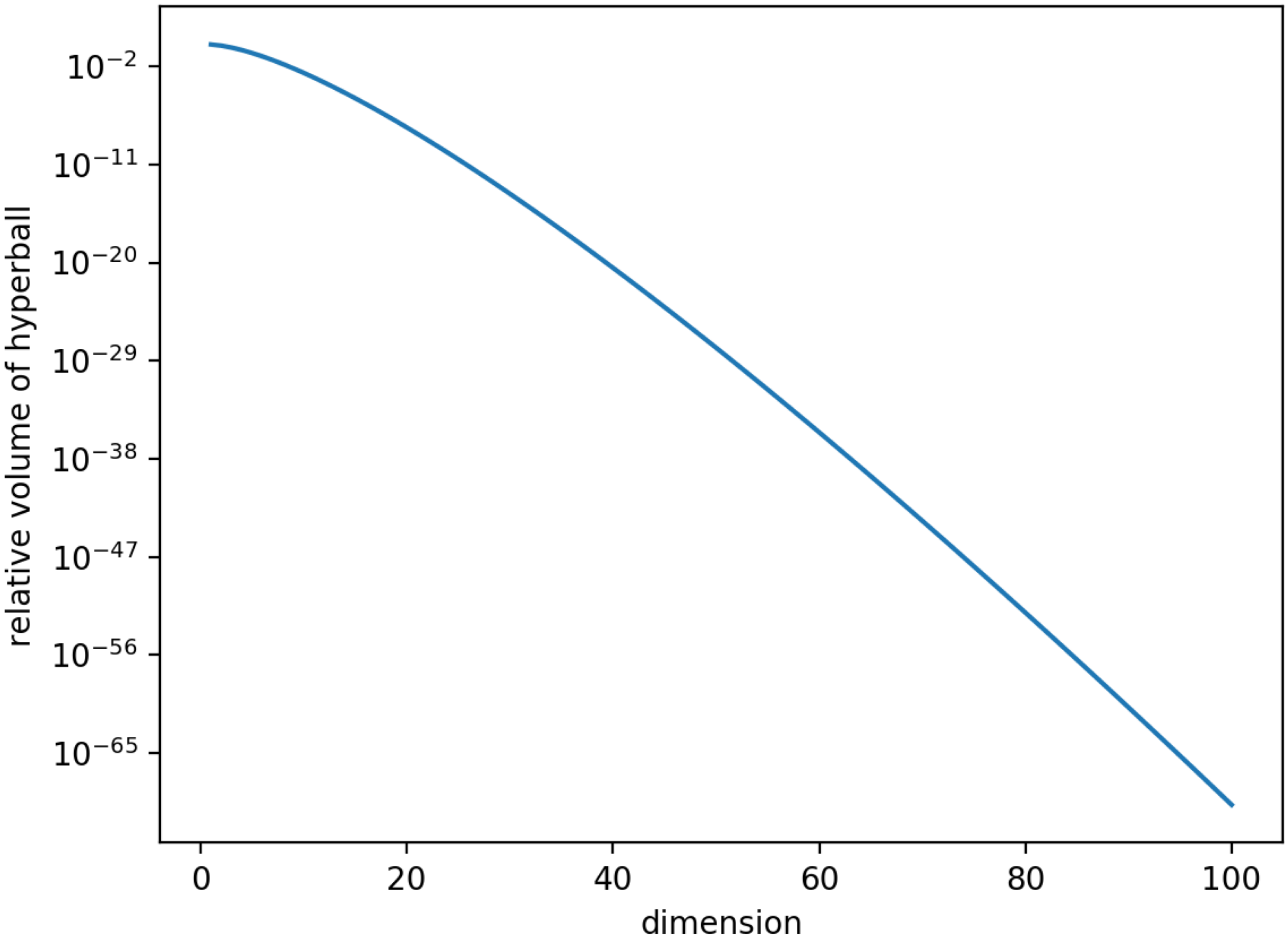
Curse of Dimensionality

Volume of unit hyper-ball in n-dimensions: $V_n = \dfrac{\pi^{n/2}}{\Gamma(\frac{n}{2}+1)}$

$$V_{2k} = \frac{\pi^k}{k!} \qquad V_{2k+1} = \frac{2(k!)(4\pi)^k}{(2k+1)!}$$

Ratio to volume of bounding hyperbox $\dfrac{V_{2k}}{2^k} = \left(\dfrac{\pi}{4}\right)^k \dfrac{1}{k!}$





relative volume of hyperball
10^{-2}
10^{-11}
10^{-20}
10^{-29}
10^{-38}
10^{-47}
10^{-56}
10^{-65}
0
20
40
60
80
100
dimension

V

F

Dimension Reduction

PCA

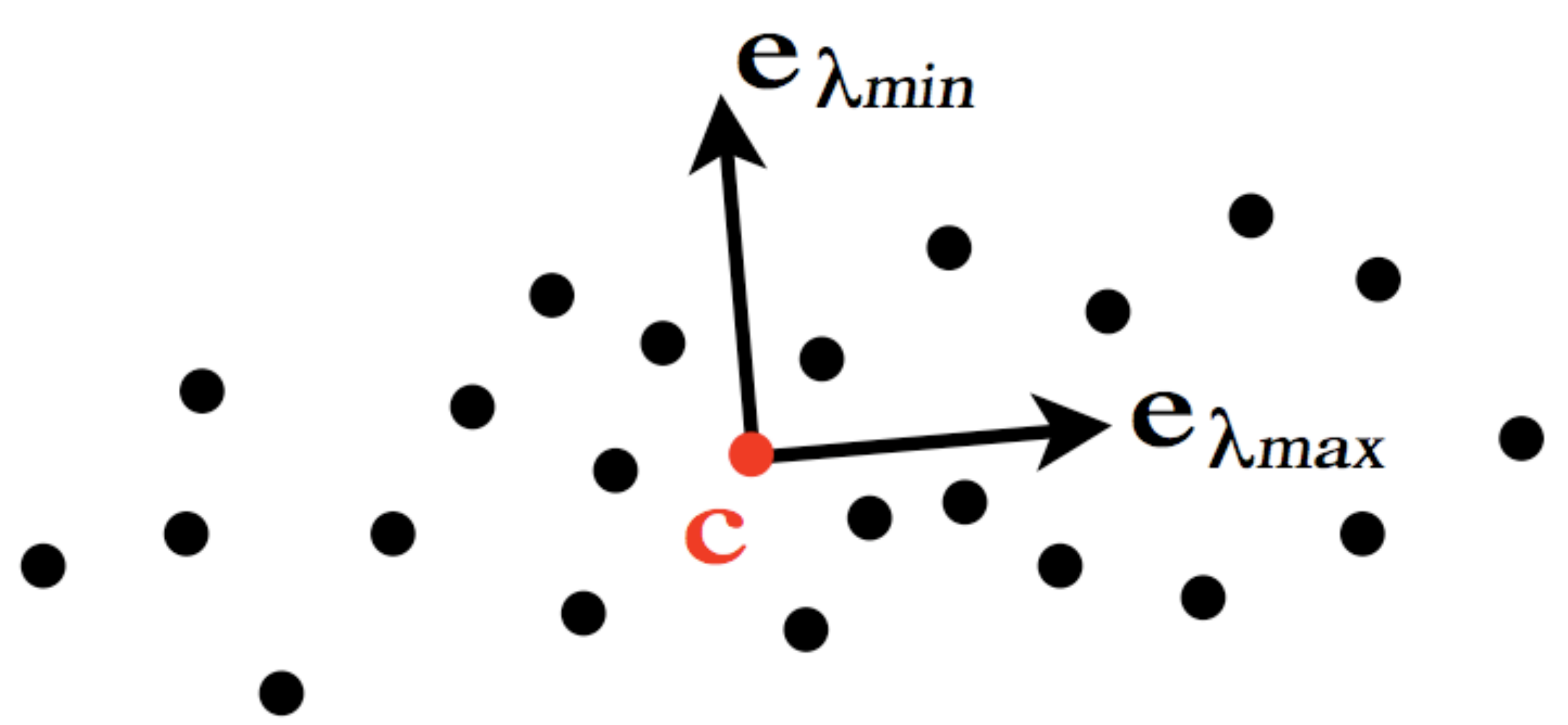
- Assume a point cloud $(\mathbf{x}_i)_{i=1,..,n} \in I\!R^k$
- Interpret these points as observations of a random variable $\mathbf{X} \in I\!R^k$
- The empirical mean (centroid) is $\mathbf{c} = \frac{1}{n}\sum_i \mathbf{x}_i$
- The covariance matrix is given by $\mathbf{A} = \bar{\mathbf{X}}\bar{\mathbf{X}}^T$

$$A_{jl} = \frac{1}{n}\sum_i (x_{ij} - c_j)(x_{il} - c_l)$$

PCA

- The eigenvectors of the covariance matrix form a data-dependent coordinate system

- The first *m* eigenvectors (in decreasing order of associated eigenvalues) span the *m* principal dimensions of the point cloud.



MDS

Multidimensional Scaling

- Input: dissimilarity matrix $\Delta \in I\!R^{n \times n}$

$\Delta_{ij} = \delta_{i,j}$ with $\Delta_{ij} > 0$ and $\Delta_{ii} = 0$

- Goal: find $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in I\!R^d$ such that

$$\forall(i, j)\ ||\mathbf{x}_i - \mathbf{x}_j|| \approx \delta i, j$$

MDS

Method:

- Center(*): $B = -\frac{1}{2}H\Delta H$ where $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$
- Spectral decomposition: $B = U\Lambda U^T$
- Clamp(*): $(\Lambda_+)_{ij} = \max(\Lambda_{ij}, 0)$
- Solution: first d columns of $X = U\Lambda_+^{\frac{1}{2}}$

This is a PCA!

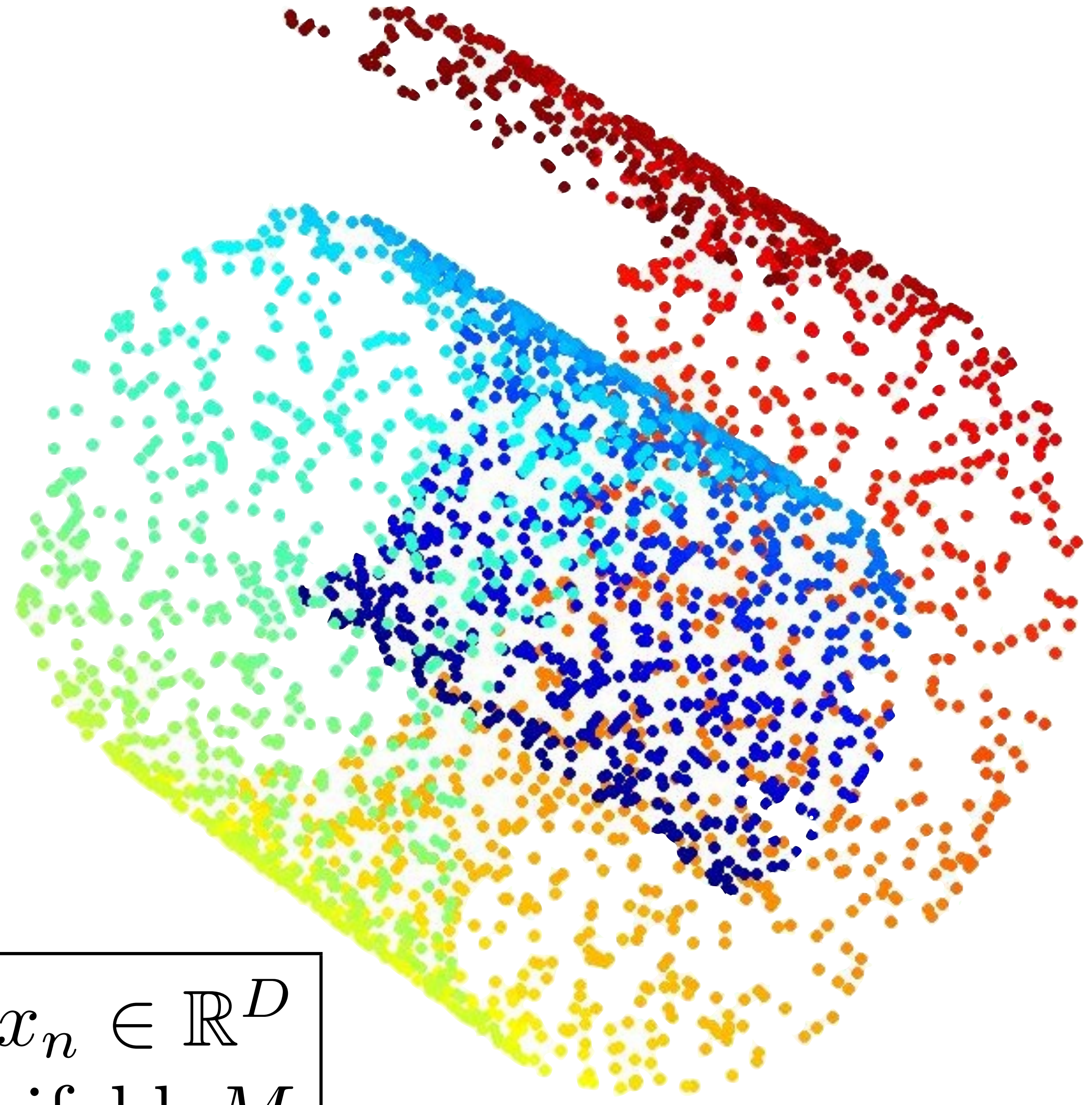
(*): if Δ *measures Euclidean distances,* B *is positive semidefinite*

Manifold Learning

- Move away from linear assumption
- Assume curved low-dimensional geometry
- Assume smoothness

➡ MANIFOLD

- Manifold learning



Problem: Given points $x_1, \ldots, x_n \in \mathbb{R}^D$ that lie on a d-dimensional manifold M that can be described by a single coordinate chart $f : M \to \mathbb{R}^d$, find $y_1, \ldots, y_n \in \mathbb{R}^d$, where $y_i \stackrel{\text{def}}{=} f(x_i)$.

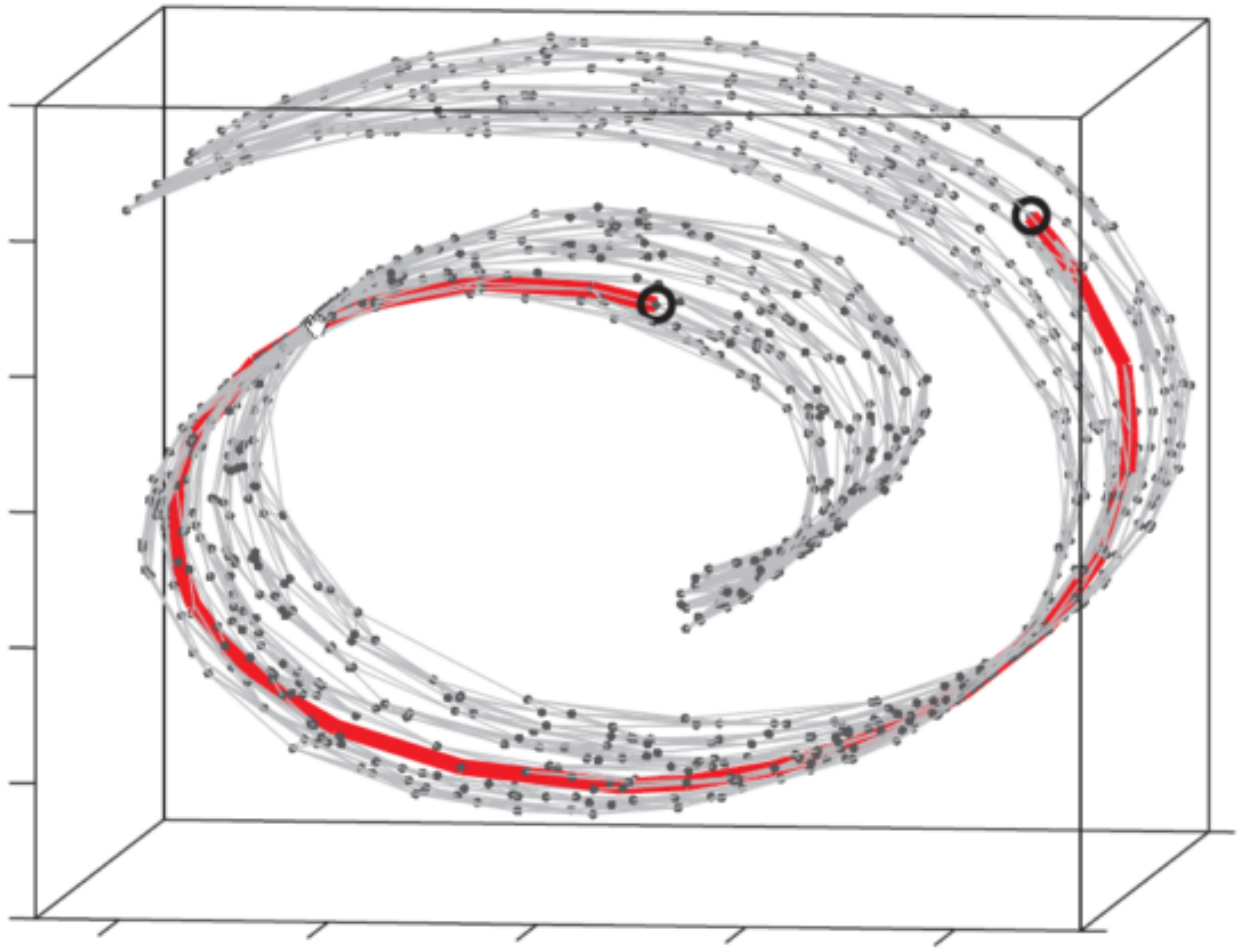
ISOMAP

Tenenbaum et al., Science, 2000

- Form k-NN graph of input points
- Form dissimilarity matrix Δ as square of approximated geodesic distance between points
- Compute MDS!

ISOMAP

- Computing geodesic distance
  - standard graph problem in CS

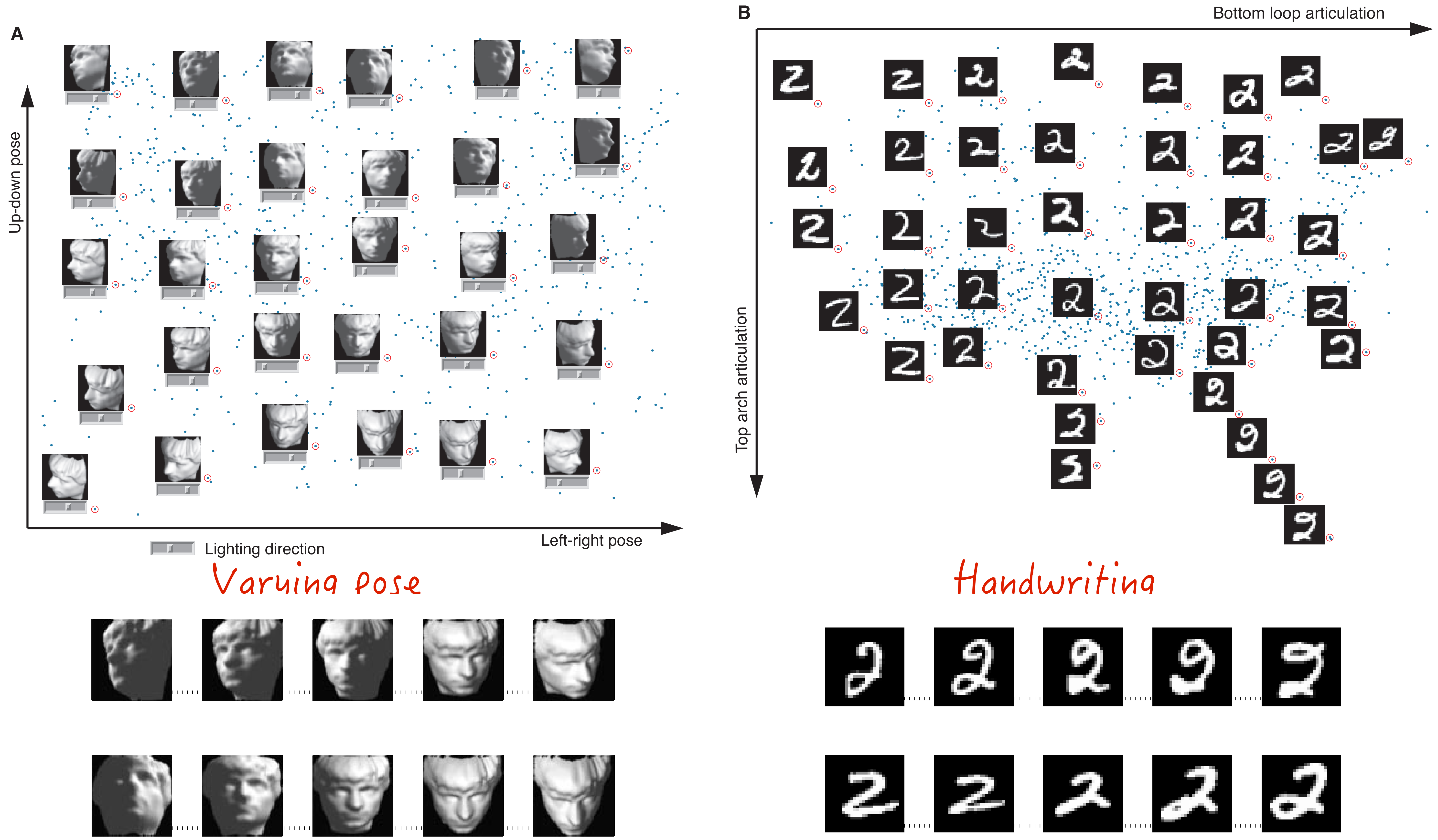


Dijkstra Algorithm $O(|E| + |V| \log |V|)$

```
function Dijkstra(Graph, source):
    for each vertex v in Graph:           // Initializations
        dist[v] := infinity               // Unknown distance function from source to v
        previous[v] := undefined          // Previous node in optimal path from source
    dist[source] := 0                     // Distance from source to source
    Q := the set of all nodes in Graph    // All nodes in the graph are unoptimized - thus are in Q
    while Q is not empty:                 // The main loop
        u := vertex in Q with smallest dist[]
        remove u from Q
        for each neighbor v of u:         // where v has not yet been removed from Q.
            alt := dist[u] + dist_between(u, v)       // be careful in 1st step - dist[u] is infinity yet
            if alt < dist[v]              // Relax (u,v,a)
                dist[v] := alt
                previous[v] := u
    return previous[]
```

ISOMAP

- Successful in computer vision problems



Locally Linear Embedding

Roweis et al., Science, 2000

Intuition

- smooth manifold is locally close to linear

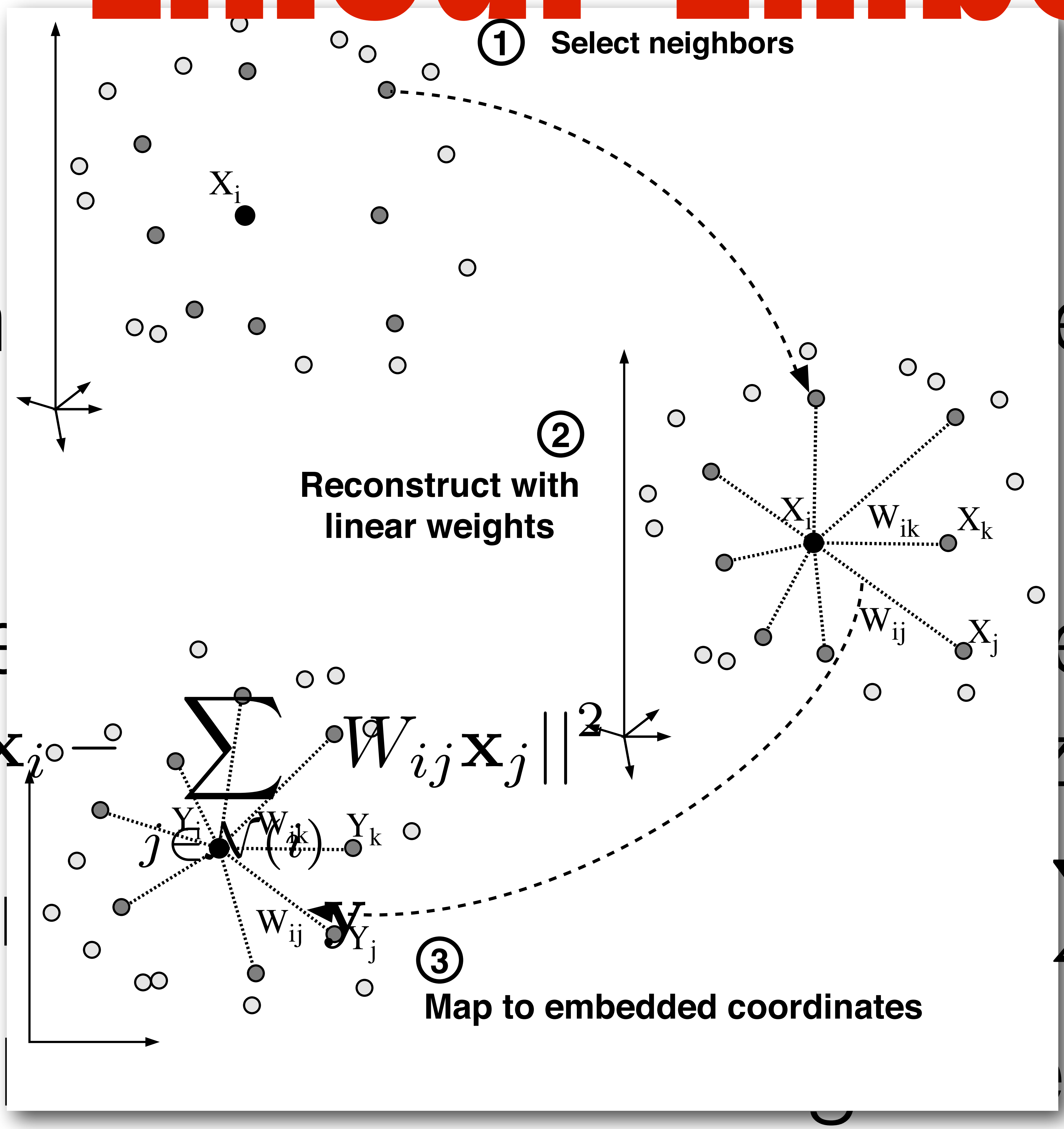
Method

- Characterize local linearity through weight matrix W such that $||\mathbf{x}_i - \sum_{j \in \mathcal{N}(i)} W_{ij}\mathbf{x}_j||^2$ is minimized
- Find d-dimensional $\mathbf{y}$'s that minimize $\sum_i ||\mathbf{y}_i - \sum_j W_{ij}\mathbf{y}_j||^2$
- Solution is obtained as first d eigenvectors of

$$(I - W)^T (I - W)$$

Locally Linear Embedding

Roweis et al., Science, 2000



Intuition

- smooth man ... ear

Method

- Characterize ... eight matrix W such that $||\mathbf{x}_i - \sum_{j \in \mathcal{N}(i)} W_{ij}\mathbf{x}_j||^2$ ... zed
- Find d-dime ... $\sum_i ||\mathbf{y}_i - \sum_j W_{ij}\mathbf{y}_j||^2$
- Solution is o ... ectors of

$$(I - W)^T(I - W)$$

Locally Linear Embedding

Roweis et al., Science, 2000

Intuition

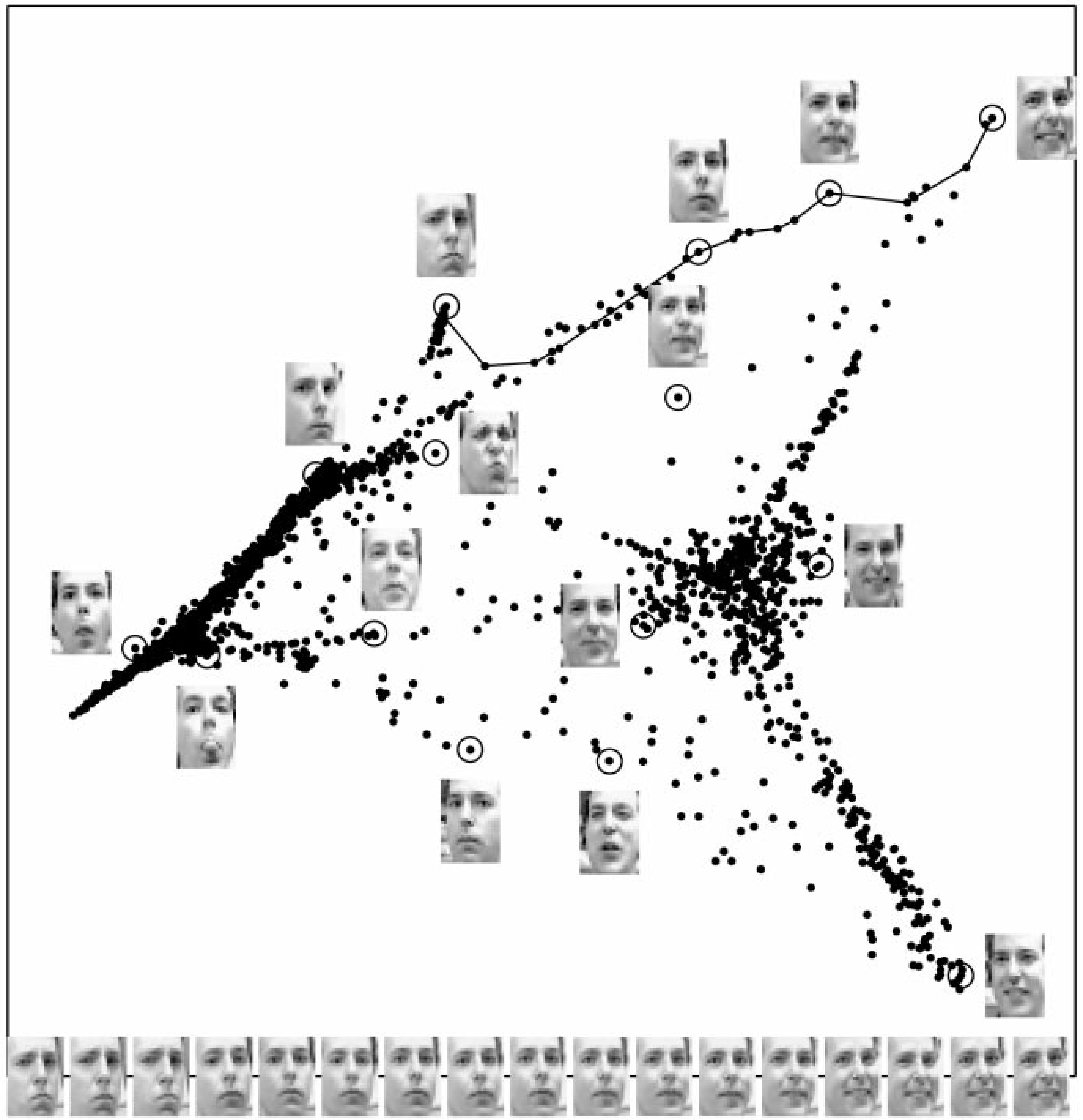
- smooth manifold is locally close to linear

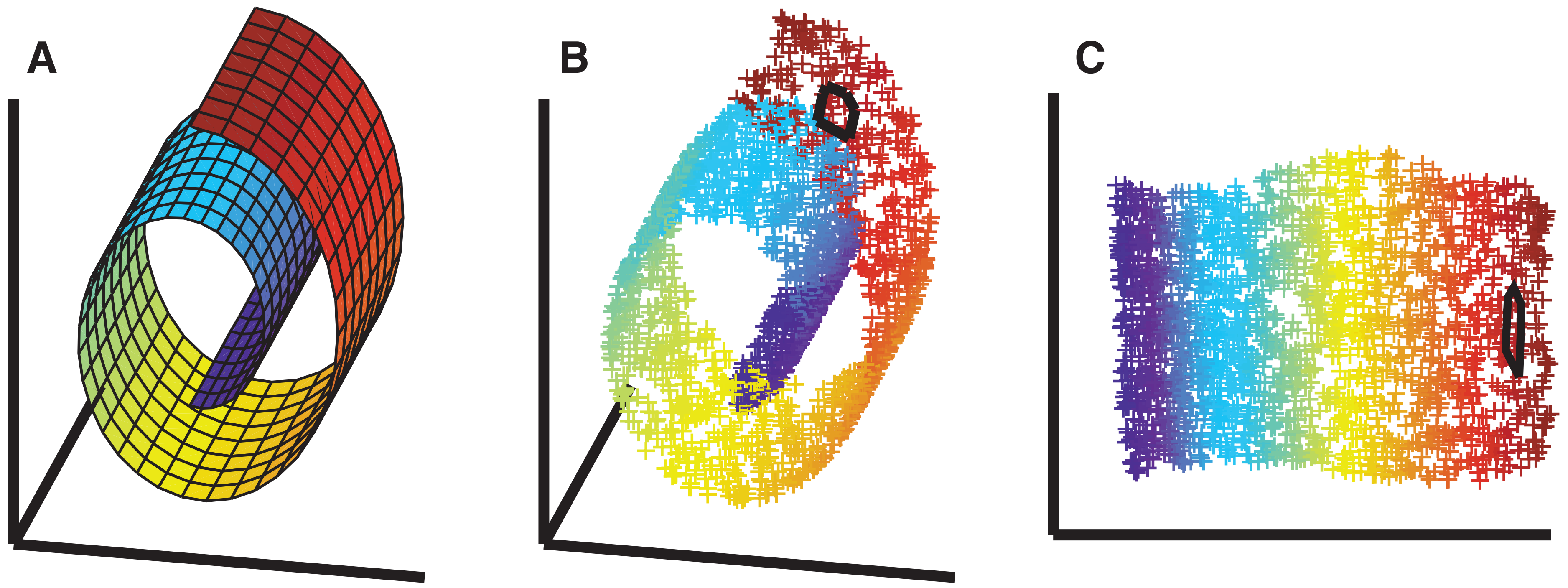
Method

- Characterize local linearity through weight matrix W such that $||\mathbf{x}_i - \sum_{j \in \mathcal{N}(i)} W_{ij}\mathbf{x}_j||^2$ is minimized
- Find d-dimensional $\mathbf{y}$'s that minimize $\sum_i ||\mathbf{y}_i - \sum_j W_{ij}\mathbf{y}_j||^2$
- Solution is obtained as first d eigenvectors of

$$(I - W)^T(I - W)$$

LLE





Laplacian Eigenmaps

Belkin, Niyogi, Advances in Neural Information Processing Systems, 2001

- Graph Laplacian of W is $L = D - W$

  with D diagonal and $D_{ii} = \sum_j W_{ij}$

- Define W as either binary indicator of connectivity or through heat kernel
- Solve for $\sum_{i,j} W_{ij} ||\mathbf{y}_i - \mathbf{y}_j||^2 = \mathrm{tr}(Y^T L Y)$
- Compute eigenvectors $\mathbf{y}$ of Laplacian associated with non-zero eigenvalue

Laplacian Eigenmaps

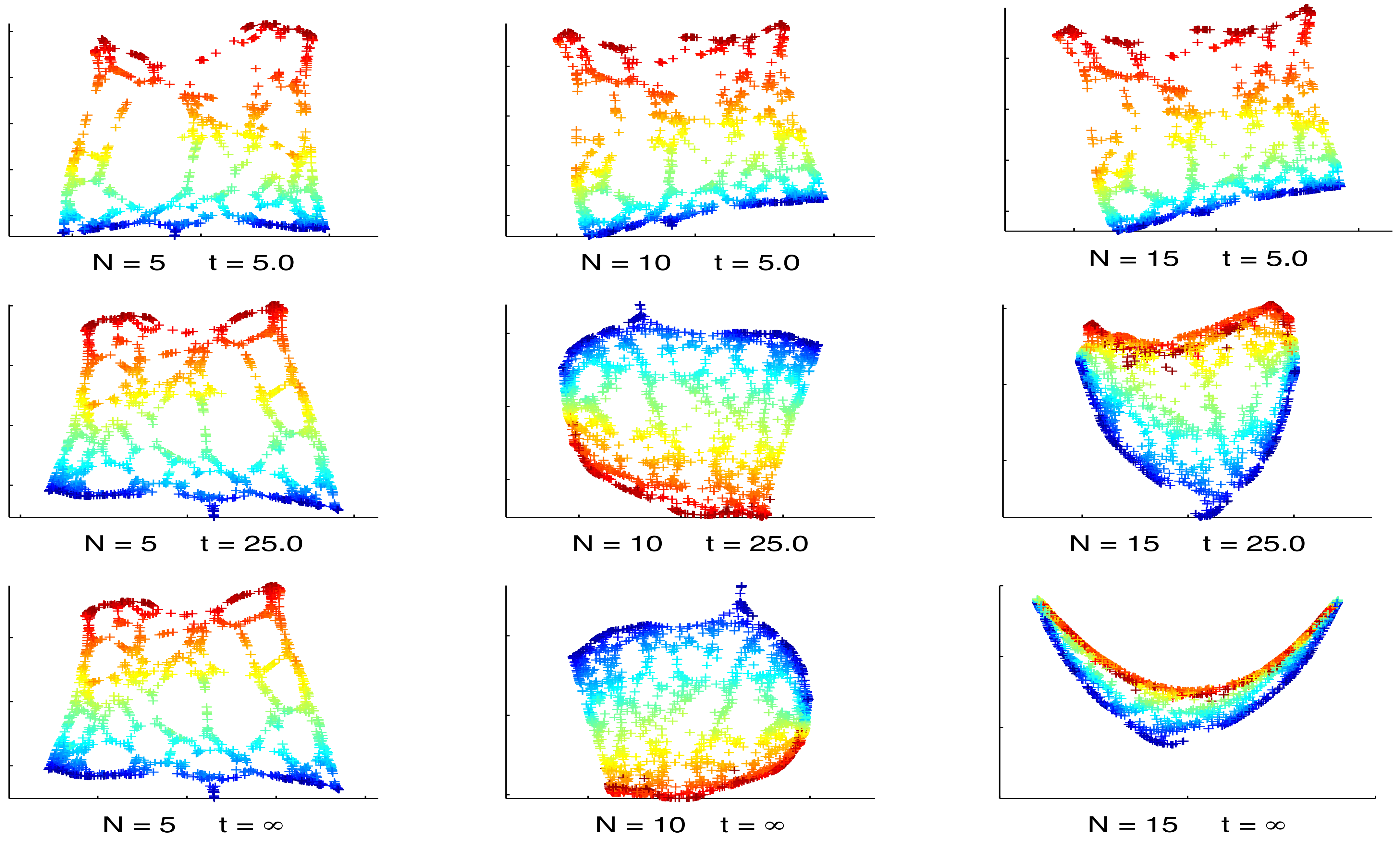
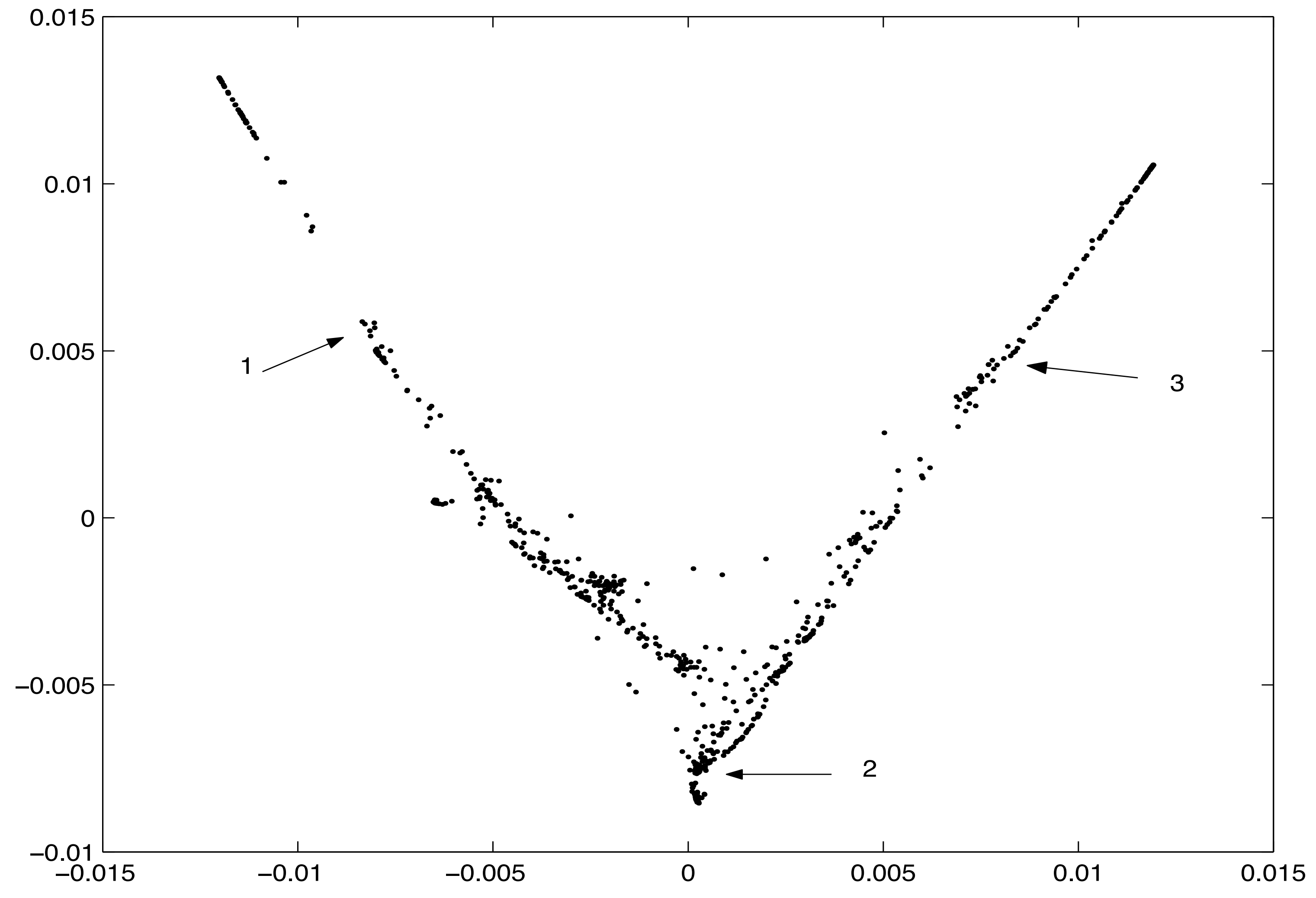
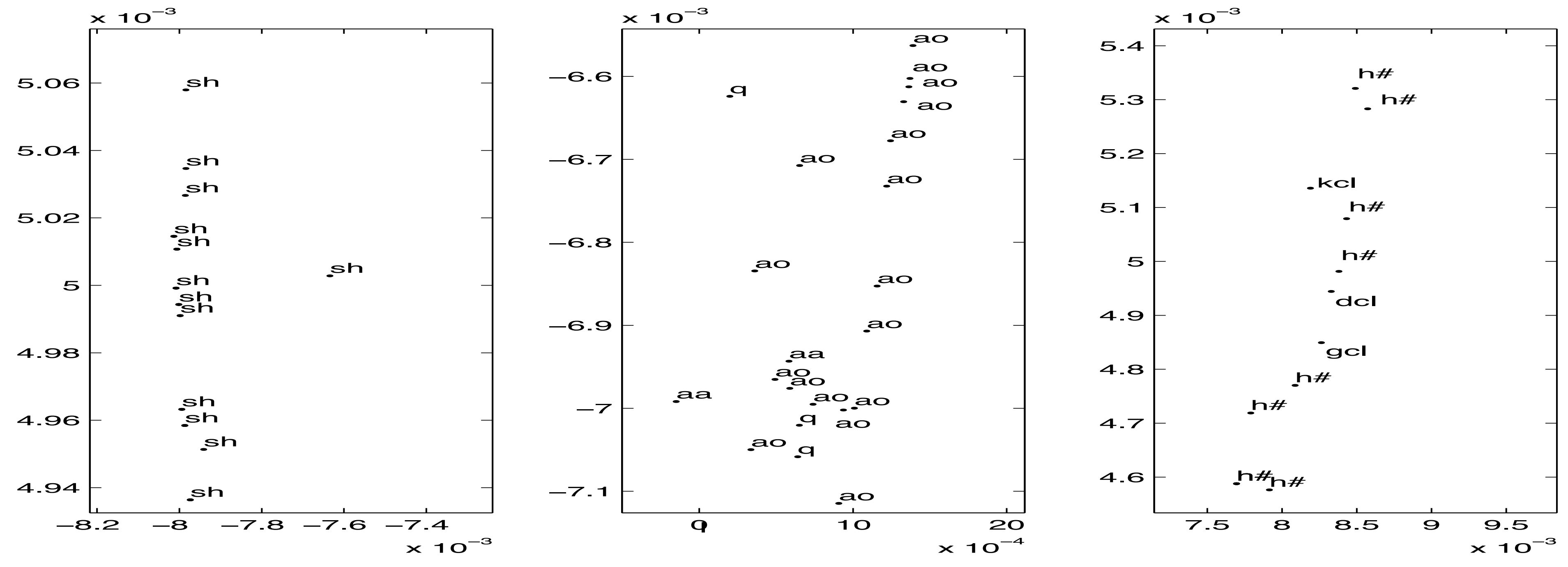


Figure 2: Two-dimensional representations of the "swiss roll" data, for different values of the number of nearest neighbors N and the heat kernel parameter t. $t = \infty$ corresponds to the discrete weights.





Computation

- Spectral decomposition intractable for large problems
- Approximate method can be used
- Nyström method + column sampling