On Dimensionality Reduction

CS189/289A: Introduction to Machine Learning

Stella Yu

UC Berkeley

28 November 2017

Outline

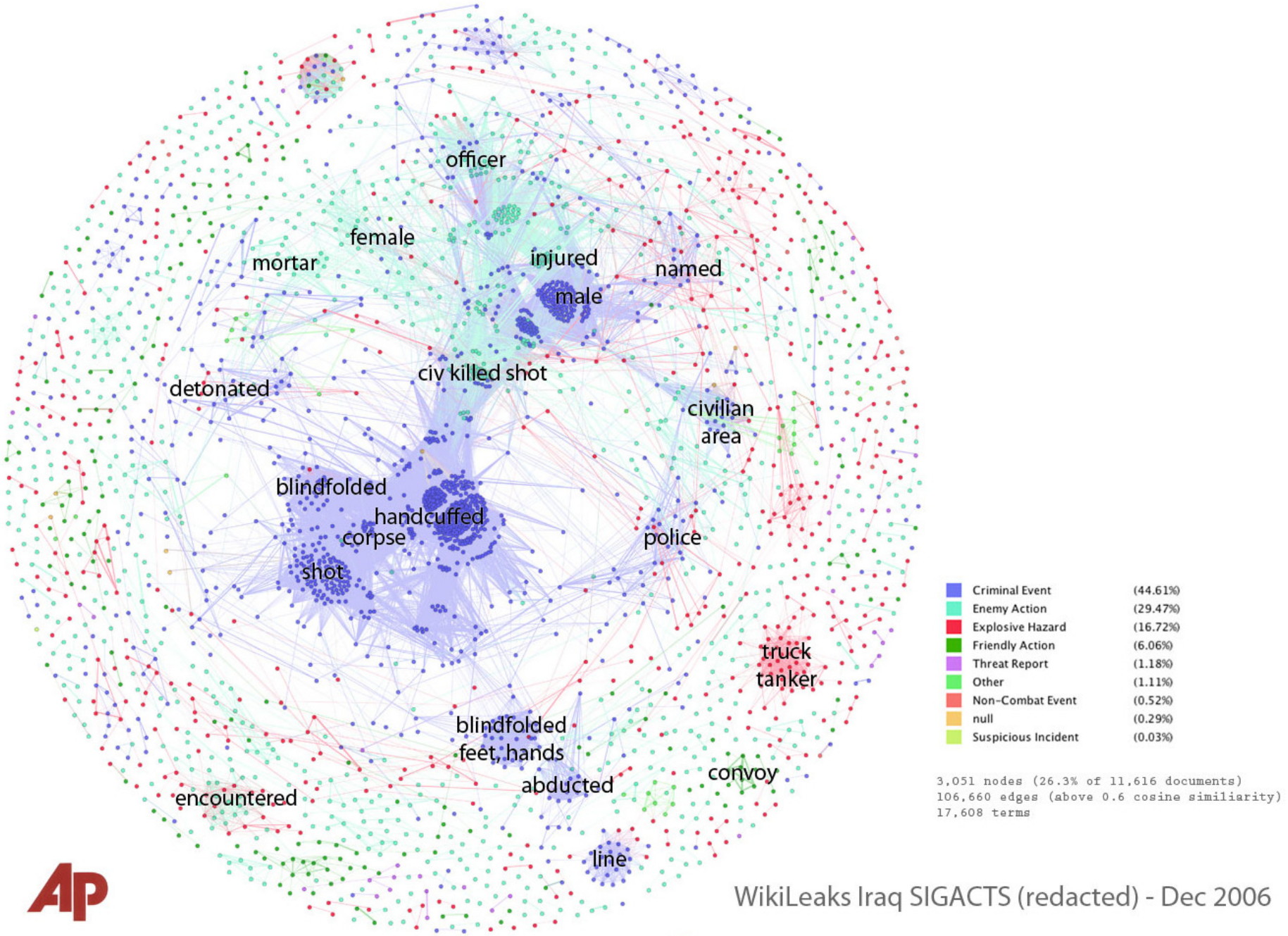
1. Why dimensionality reduction?

2. Linear methods – factorized latent analysis
   - PCA
   - ICA
   - NMF
   - MDS

3. Nonlinear kernel methods – manifold learning
   - IsoMap
   - Laplacian Eigenmap
   - tSNE

4. Clustering
   - Spectral clustering
   - K-means

High-Dimensional Data: Document Classification

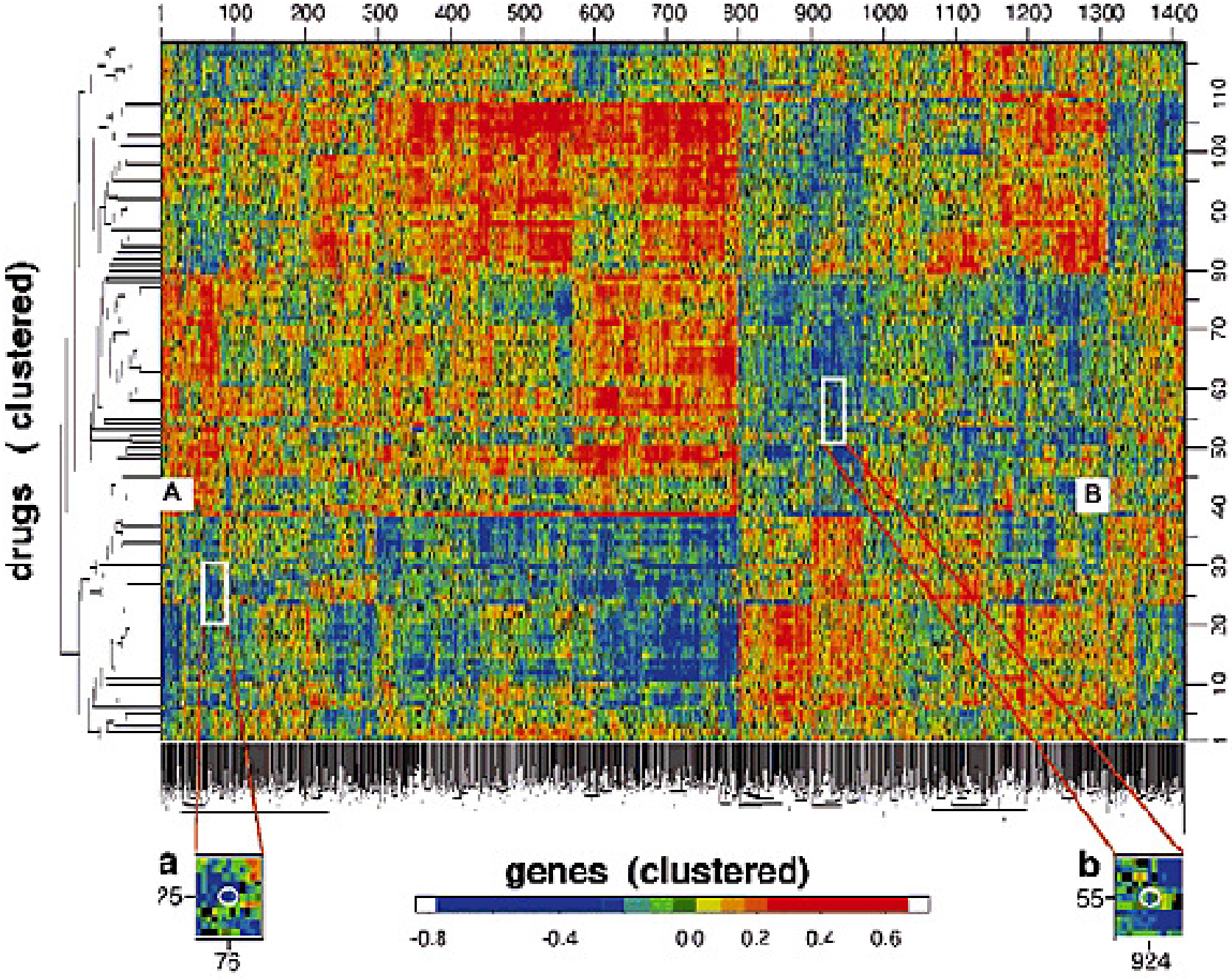


WikiLeaks Iraq SIGACTS (redacted) - Dec 2006

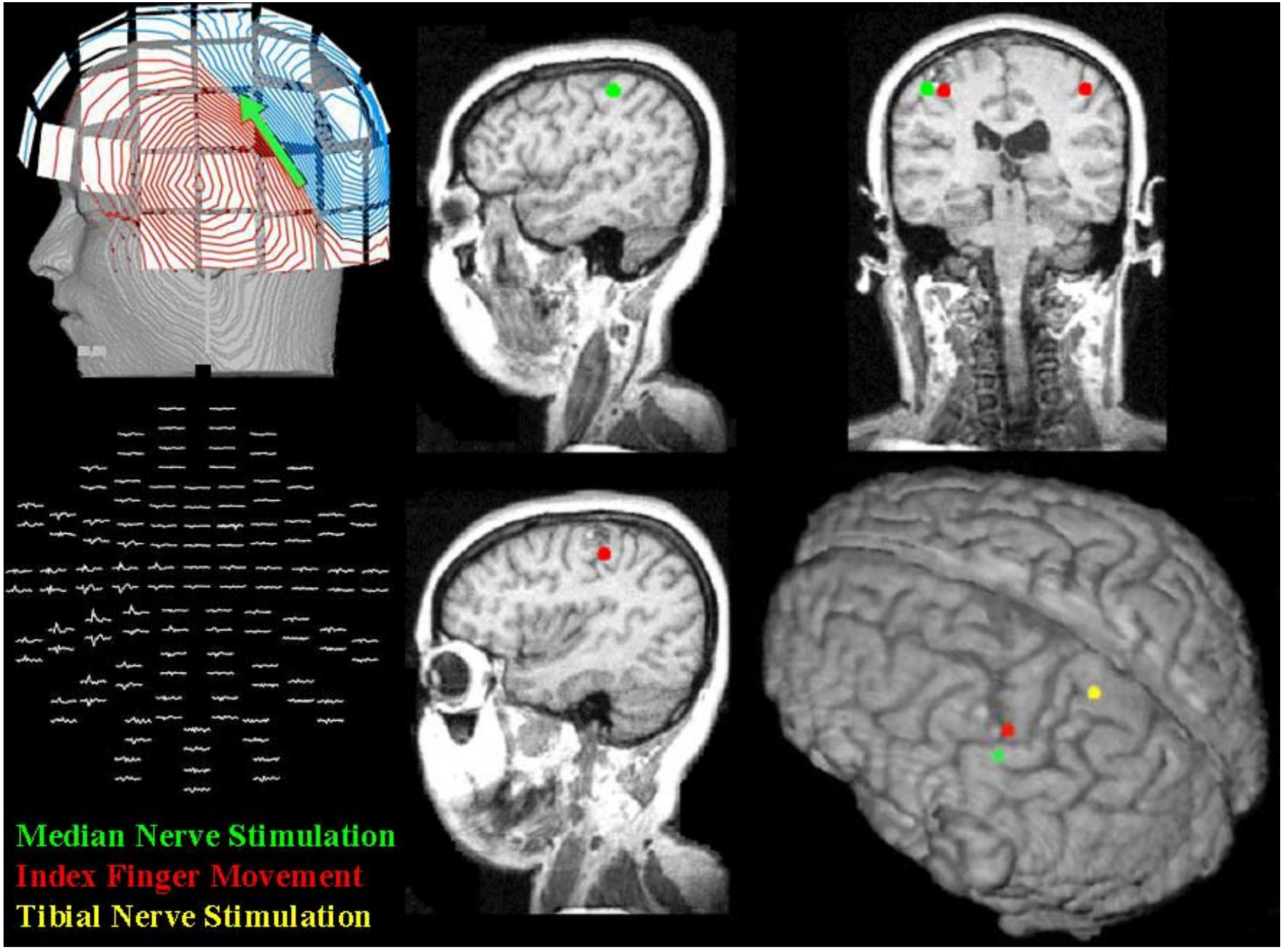
High-Dimensional Data: User Preference Surveys

	The Godfather	Memento	Titanic	Star Wars	$\cdots$
John	5	1	2	5	$\cdots$
Dave	4		2	2	$\cdots$
Susan	5	1	2		$\cdots$
Mark	2	2		4	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

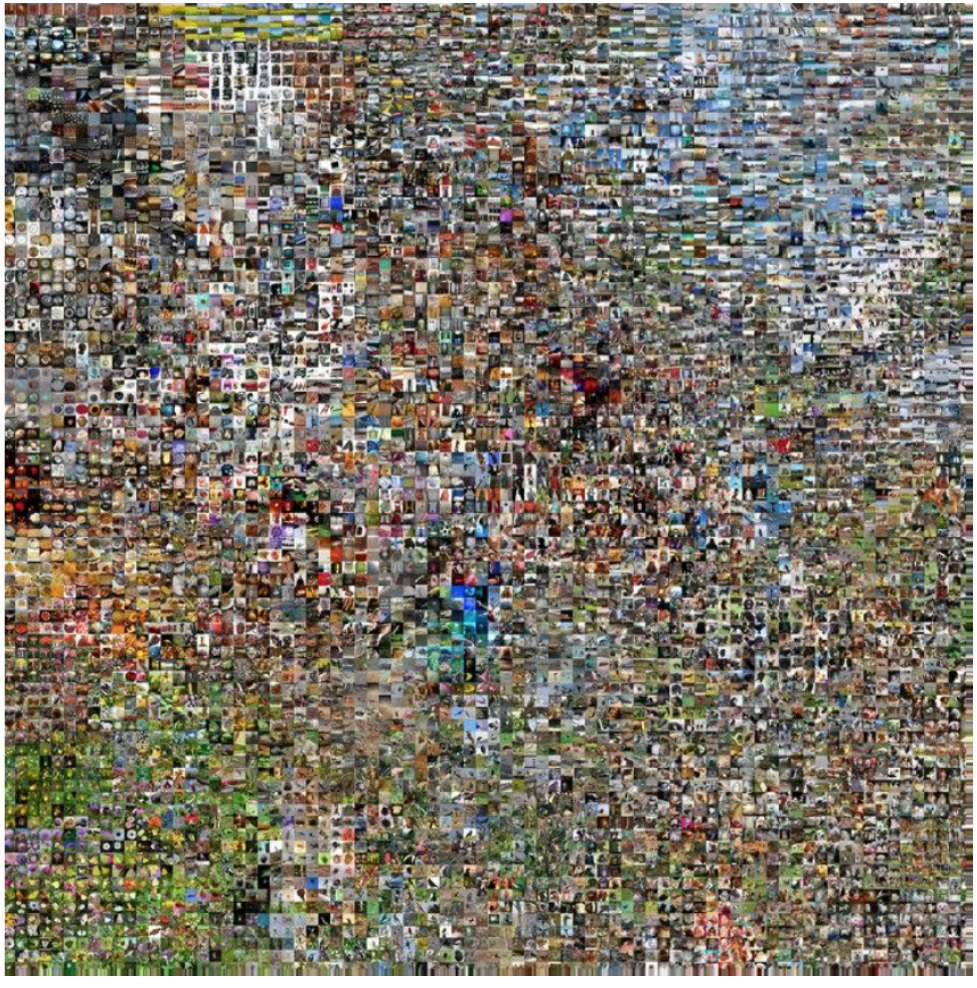
High-Dimensional Data: Gene Discovery



High-Dimensional Data: Brain Imaging



High-Dimensional Data: CNN Features



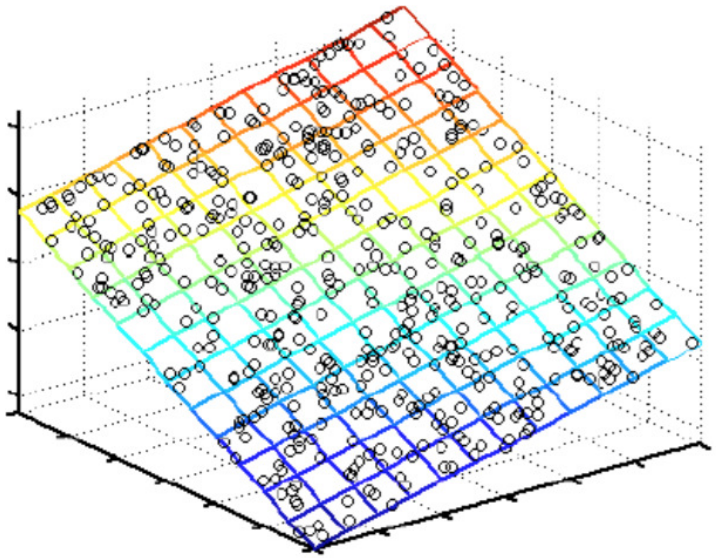
Curse of Dimensionality: Why More Is Bad?

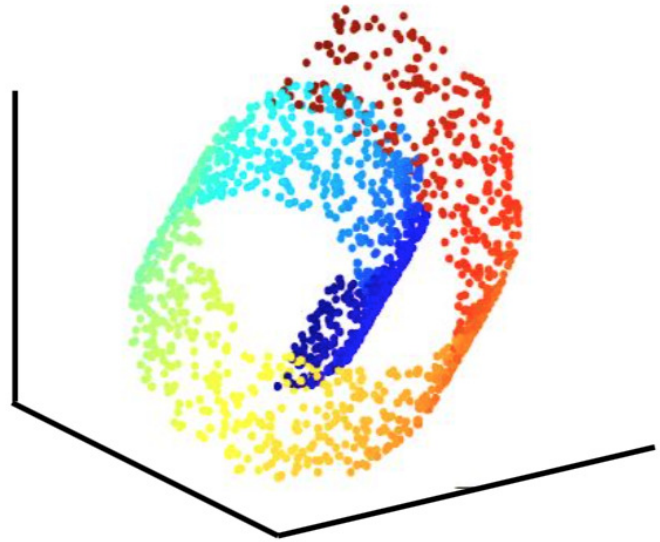
- More noise added than signal
  Redundant features, not all useful
- Computationally challenging
  Hard to store and process data
- Statistically challenging
  Hard to learn complex decision rules (grow with # features)
- Semantically challenging
  Hard to interpret and visualize

Dimensionality Reduction

- Feature Selection: Only a few features are relevant
  Approaches: subset selection and regularization (covered)

- Latent Features: Some combination of features provides a more efficient representation than observed features

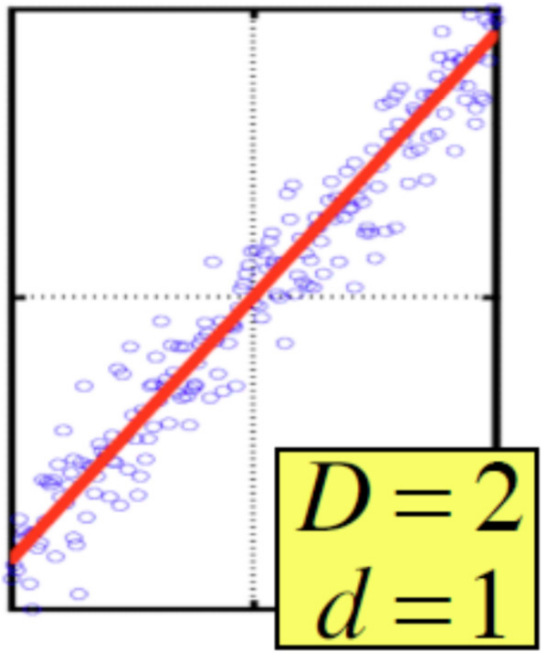


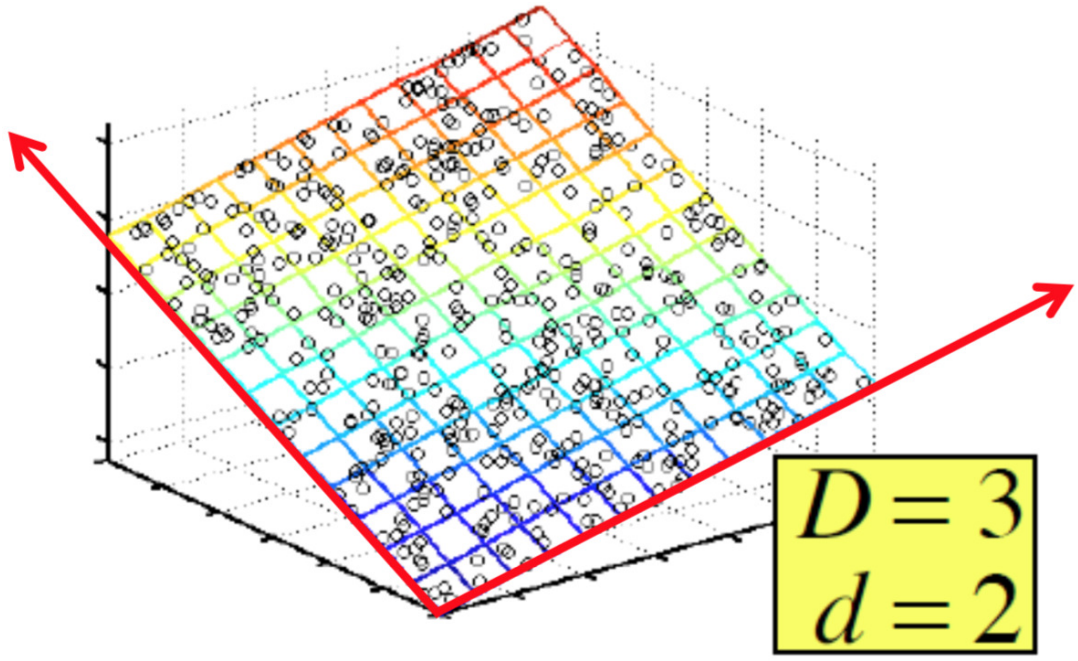


Latent Feature Extraction

- Combinations of features provide more efficient representation, capture underlying relations that govern the data
- Ego, personality and intelligence are hidden attributes that characterize human behavior instead of survey questions
- Topics such as sports, science, news, etc. instead of documents
- Often may not have any physical meaning
- Linear factorization methods:
  PCA, ICA, NMF, MDS, ...
- Nonlinear manifold learning:
  IsoMap, Laplacian Eigenmaps, tSNE, ...

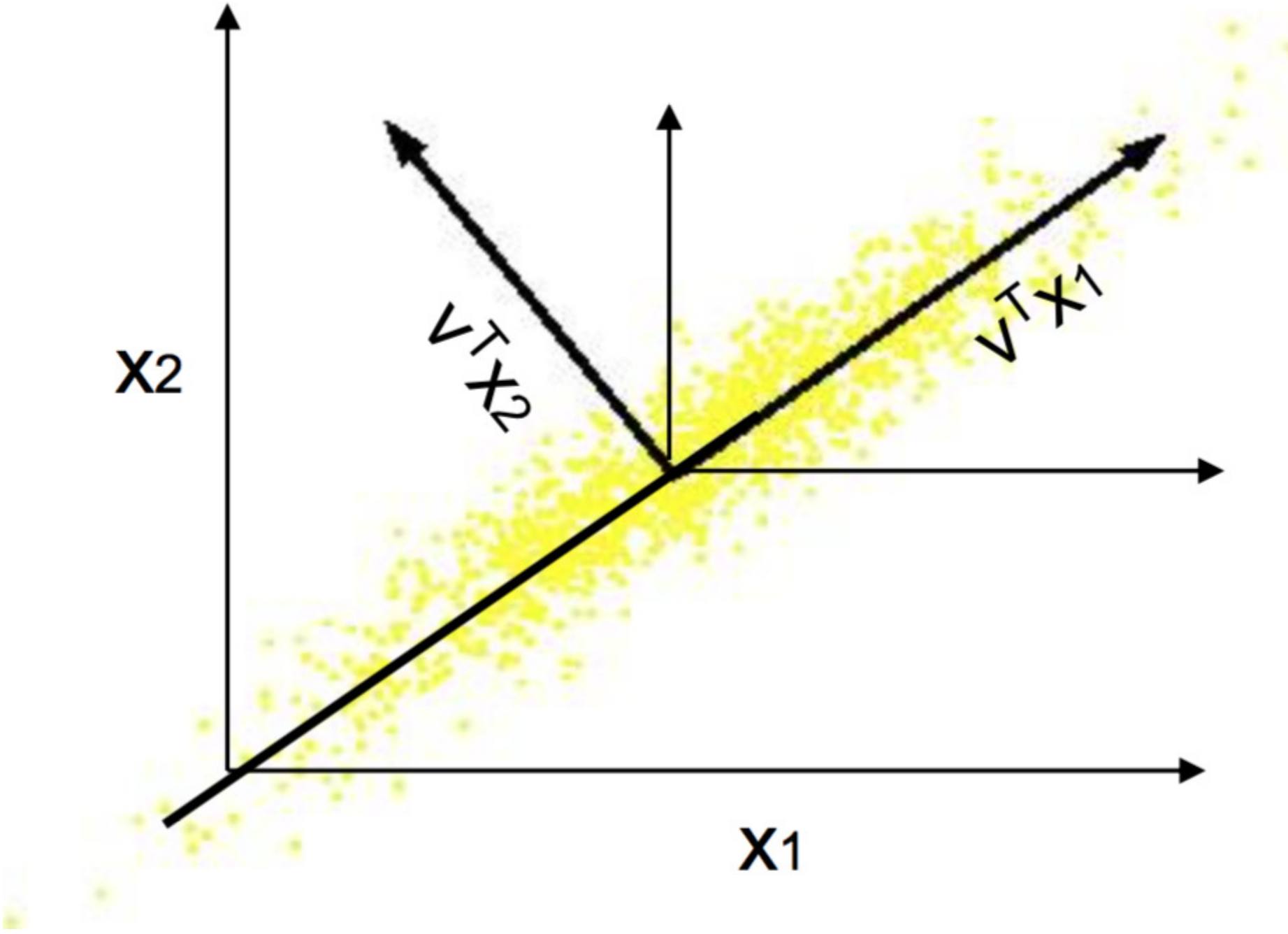
Principal Component Analysis (PCA)





- Assumption: Data lies on a low dimensional linear subspace.
- Axes of this subspace are an effective representation.
- Identifying the axes is known as PCA.
- It can be obtained by eigen or singular value decomposition.
- Geometrically, centering followed by rotation.

PCA Geometry: Centering Followed by Rotation



PCA: Projection for Dimensionality Reduction

- Original representation $\begin{bmatrix} x_i^1 & x_i^2 & \dots & x_i^D \end{bmatrix}'$

$$x_i = \sum_{j=1}^{D} x_i^j e_j = \sum_{j=1}^{D} (e_j' x_i)\, e_j \tag{1}$$

$$e_j = \begin{bmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots \end{bmatrix}' \tag{2}$$

- Transformed representation $\begin{bmatrix} v_1' x_i & v_2' x_i & \dots & v_D' x_i \end{bmatrix}'$

$$x_i = \sum_{j=1}^{D} (v_j' x_i)\, v_j \tag{3}$$

$$\lambda_j = \sum_{i=1}^{n} (v_j' x_i)^2 = \text{variance of all points along } v_j \tag{4}$$

- Dimensionality reduction $\begin{bmatrix} v_1' x_i & v_2' x_i & \dots & v_d' x_i \end{bmatrix}'$

$$x_i \approx \sum_{j=1}^{d} (v_j' x_i)\, v_j, \qquad \lambda_1 \geq \cdots \geq \lambda_D \tag{5}$$

PCA for High-Dimensional Data: Data Matrix SVD

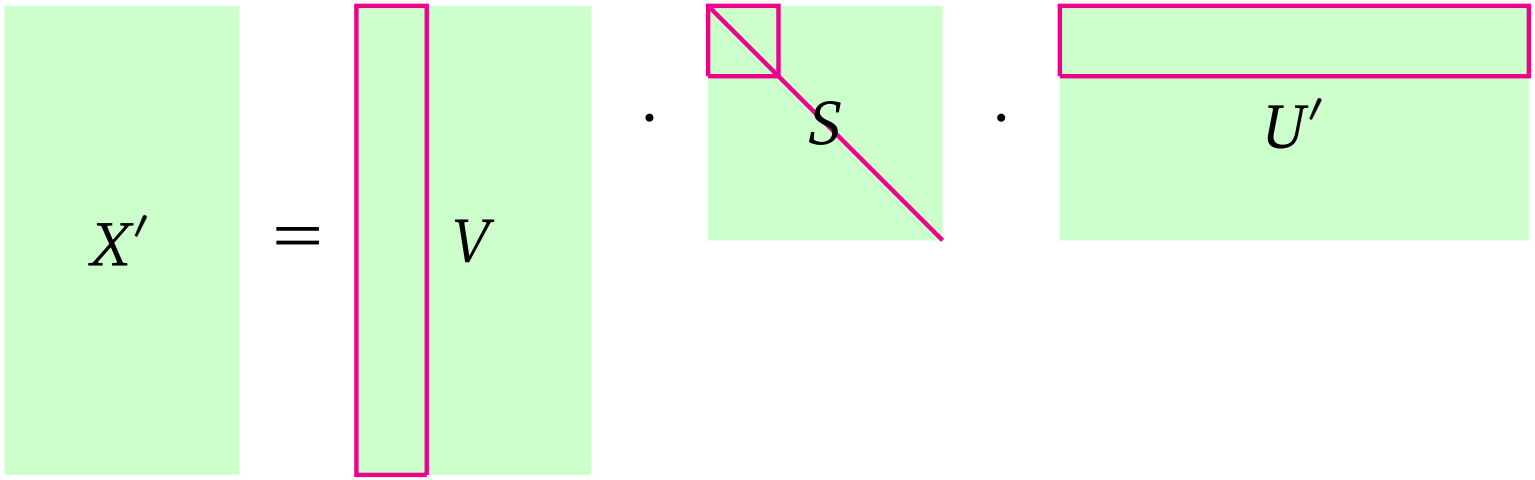
- PCA finds projection direction v with minimum LS error:

$$v = \arg\min \sum_{i=1}^{n} \|x_i - (v'x_i)v\|^2 \tag{6}$$

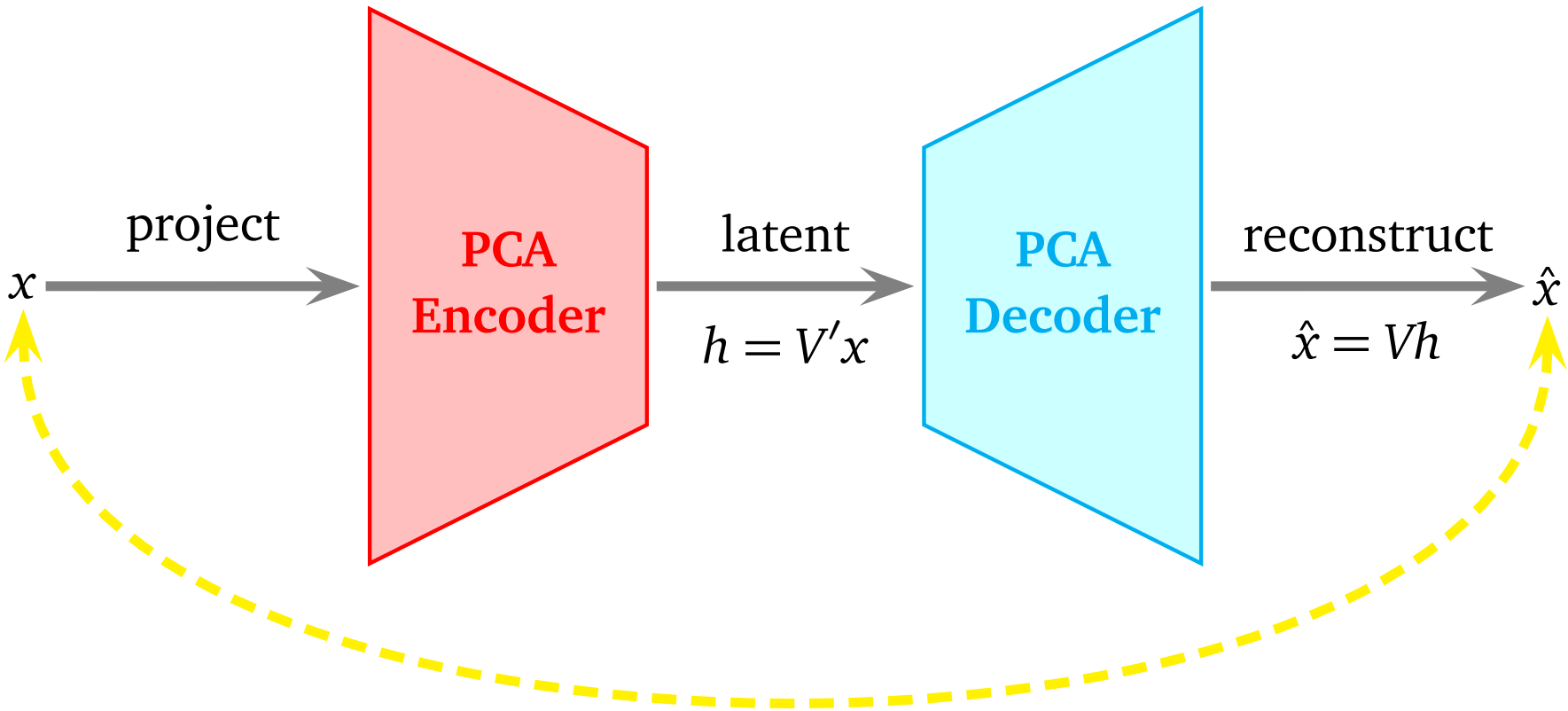
- SVD solution when $D \gg n$:

$$X_{n\times D} = U_{n\times D}\, S_{D\times D}\, V'_{D\times D} \tag{7}$$

$$X'_{D\times n} = V_{D\times D}\, S_{D\times D}\, U'_{D\times n} \approx V'_{D\times d} S_{d\times d} U'_{d\times n} \tag{8}$$

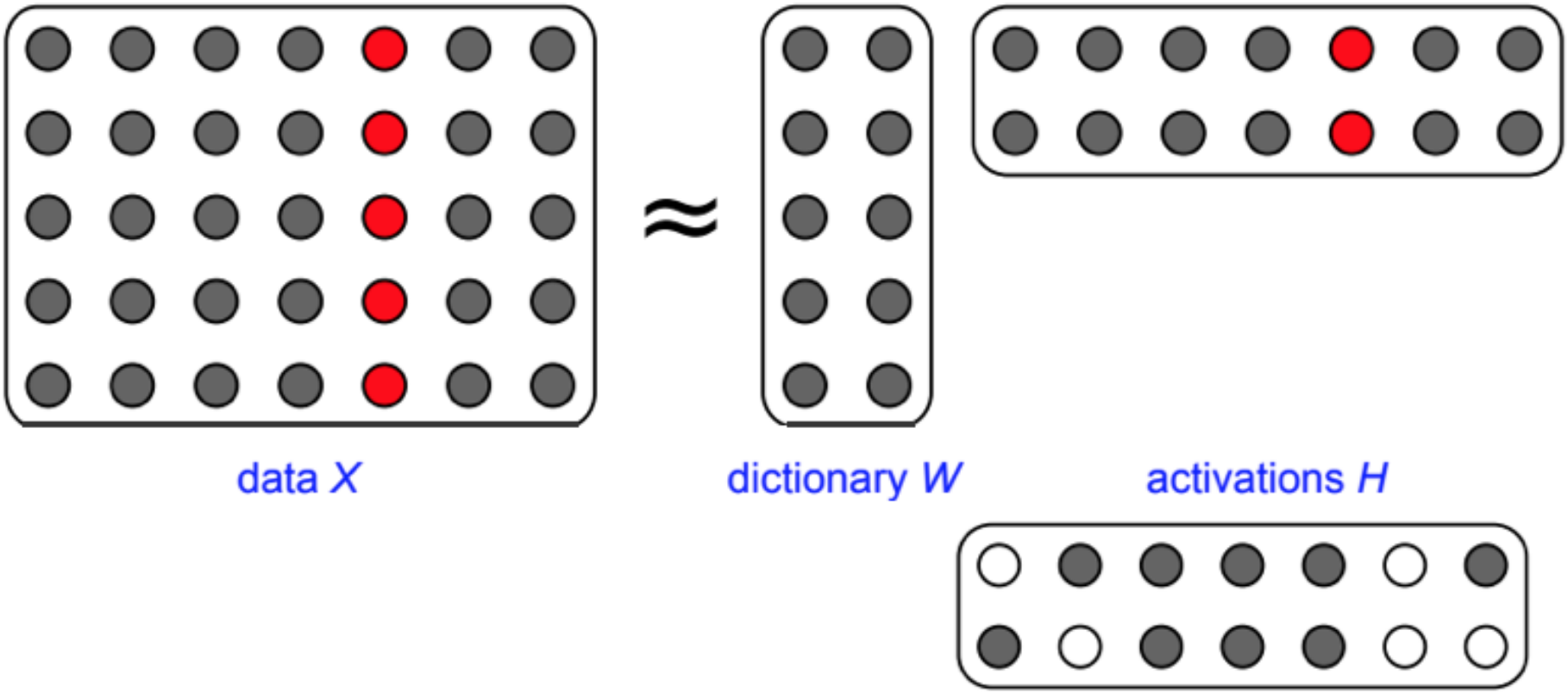


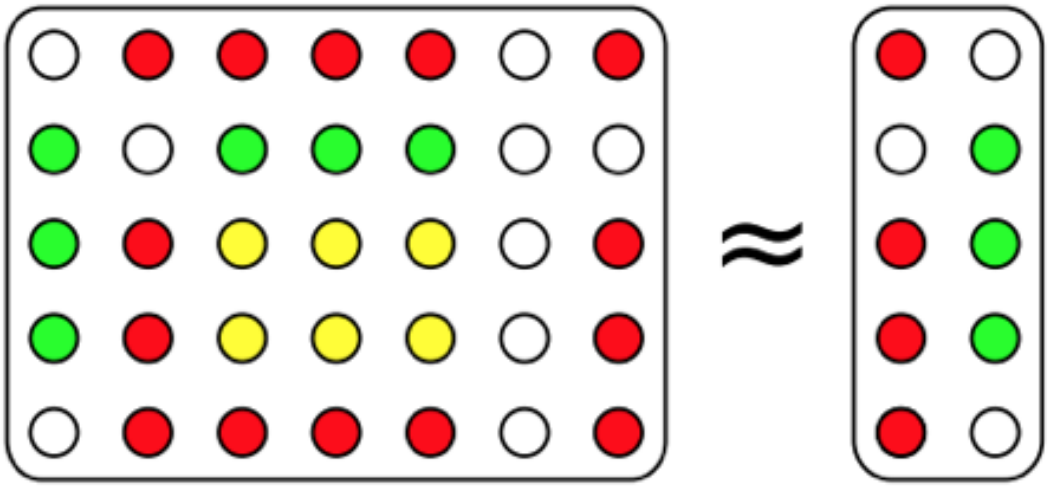
PCA as An Encoder-Decoder Pipeine



$$\min \quad \text{loss}(\hat{x}, x) = E[\|x - \hat{x}\|^2]$$

Dictionary Representation In General

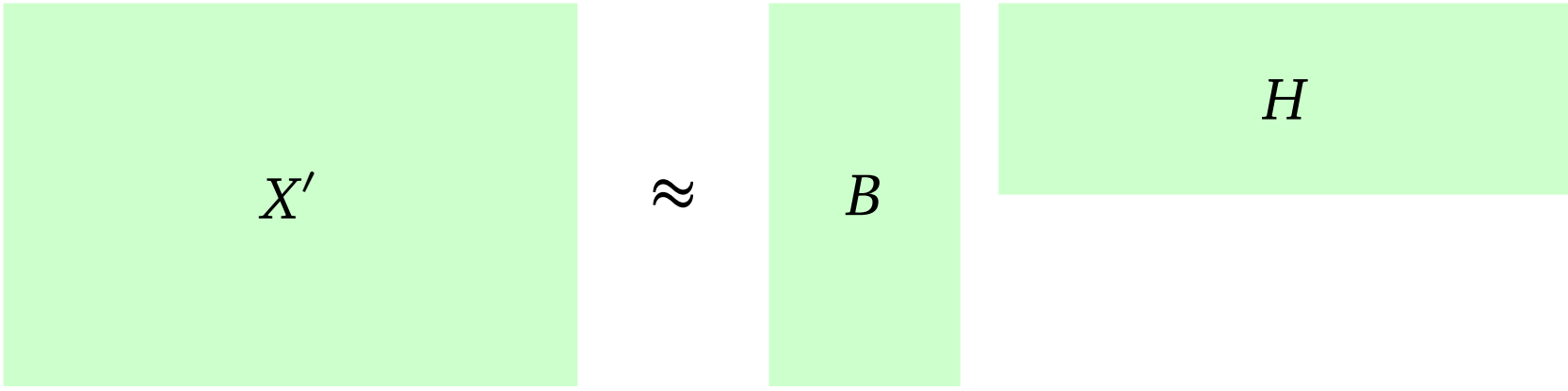




Independent Component Analysis (ICA)

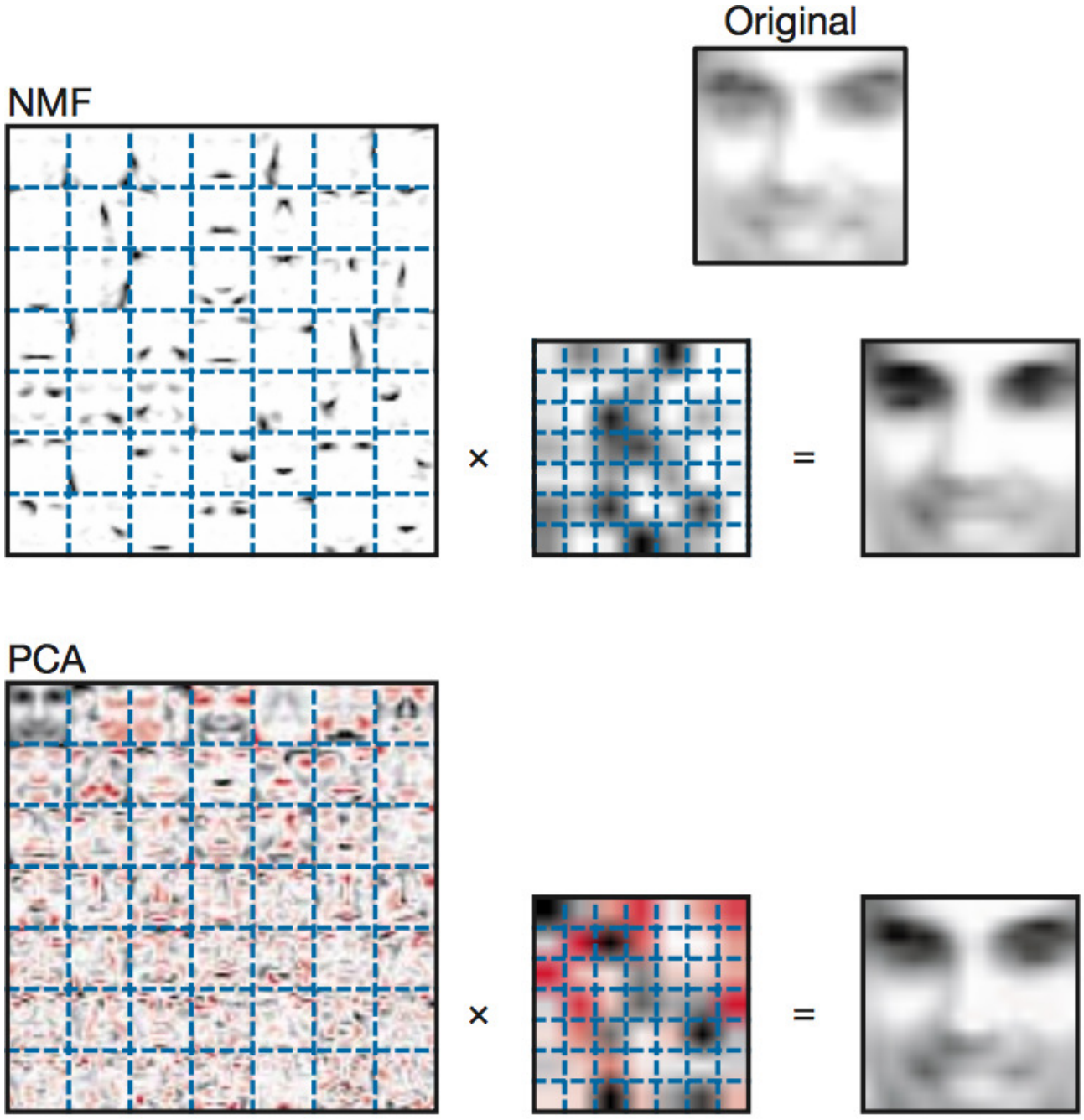
- PCA seeks orthogonal directions that capture maximum variance in data, or that minimize reconstruction error.
- ICA seeks statistically independent directions in the data.

Nonnegative Matrix Factorization (NMF)



- Data X and factors B and H all have nonnegative entries.
- Nonnegativity of B ensures interpretabilty of the dictionary, as patterns and data belong to the same space.
- Nonnegativity of H tends to produce part-based representation, as subtractive combinations are forbidden.
- Early work by (Paatero & Tapper, 1994)
  Landmark Nature paper by (Lee & Seung, 1999).

NMF Learns Parts vs. PCA Learns the Whole



Multidimensional Scaling (MDS)

- Input: A matrix of pairwise distances, W.
  For metric MDS, it can be computed from the distances between points X in D-dimensional space.

- Output: Find Y in d-dimensional space, $d \leq D$,
  such that distances between Y's match those between X's.

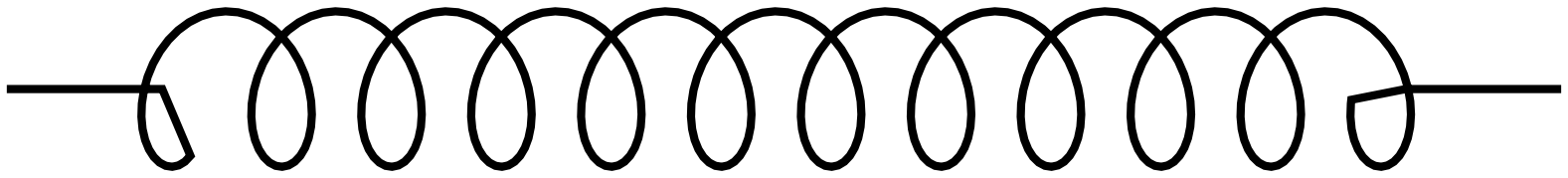
$$\text{Given } W: \qquad W_{ij} = \|X_i - X_j\|^2, \qquad X_i \in R^D \tag{9}$$

$$\text{Find } Y: \qquad \|Y_i - Y_j\|^2 \approx W_{ij}, \qquad Y_i \in R^d. \tag{10}$$

- Metric MDS is equivalent to PCA for Euclidean distances.

- Non-metric MDS: W could be general dissimilarity measures.

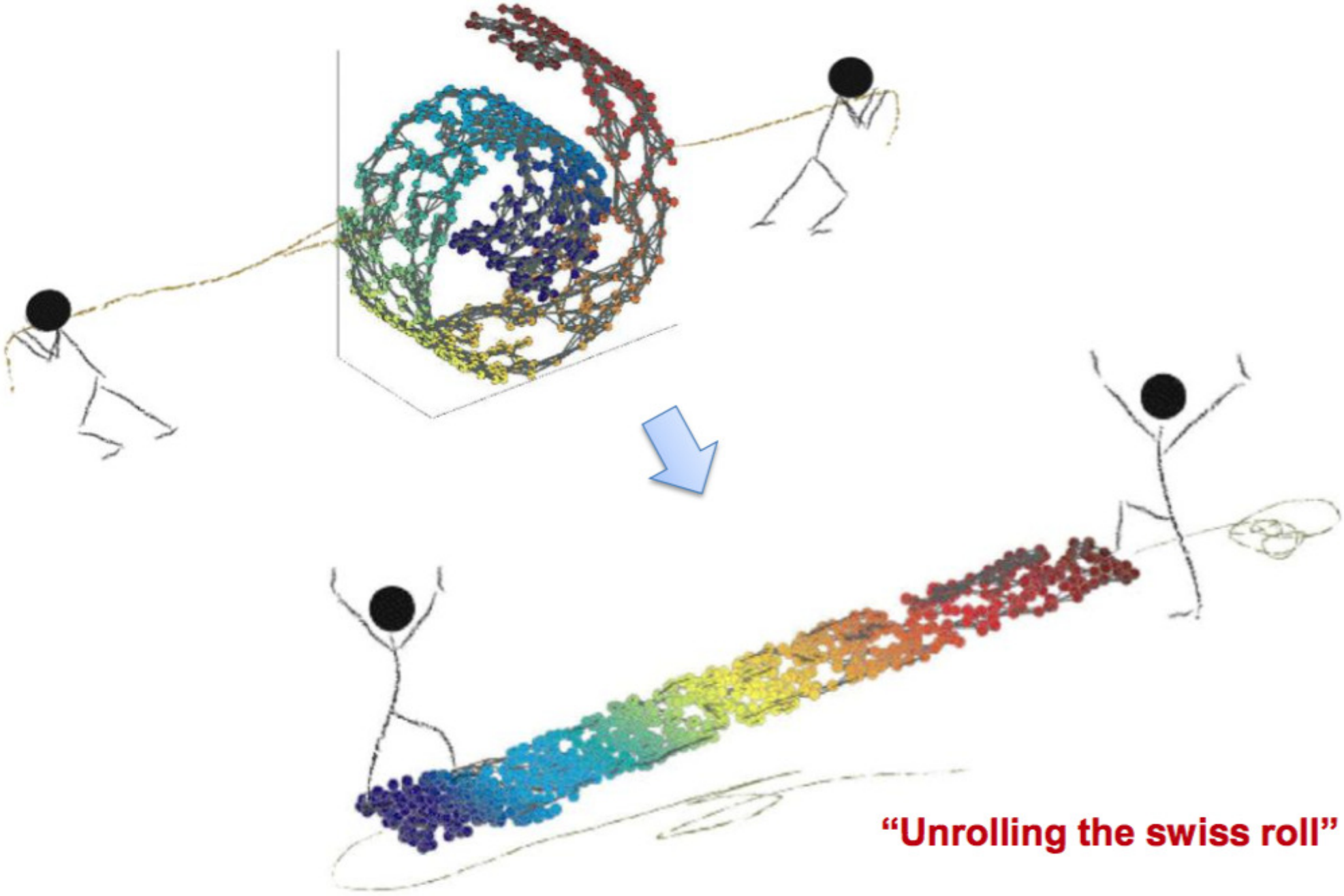
Linear Methods Cannot Discover Structures



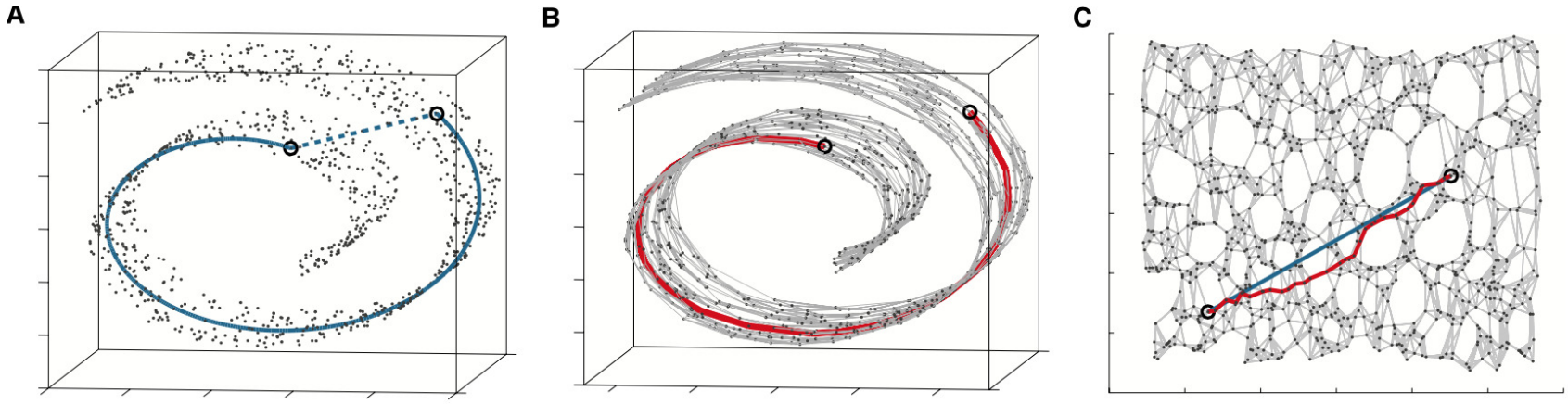
1. Linear methods such as MDS preserve interpoint distanecs and may fail to capture inherent local geometric structures

2. Nonlinear dimensionality reduction
   - Kernel methods: data independent
   - Manifold learning: data dependent kernels

Unsupervised Low-Dimensional Embedding

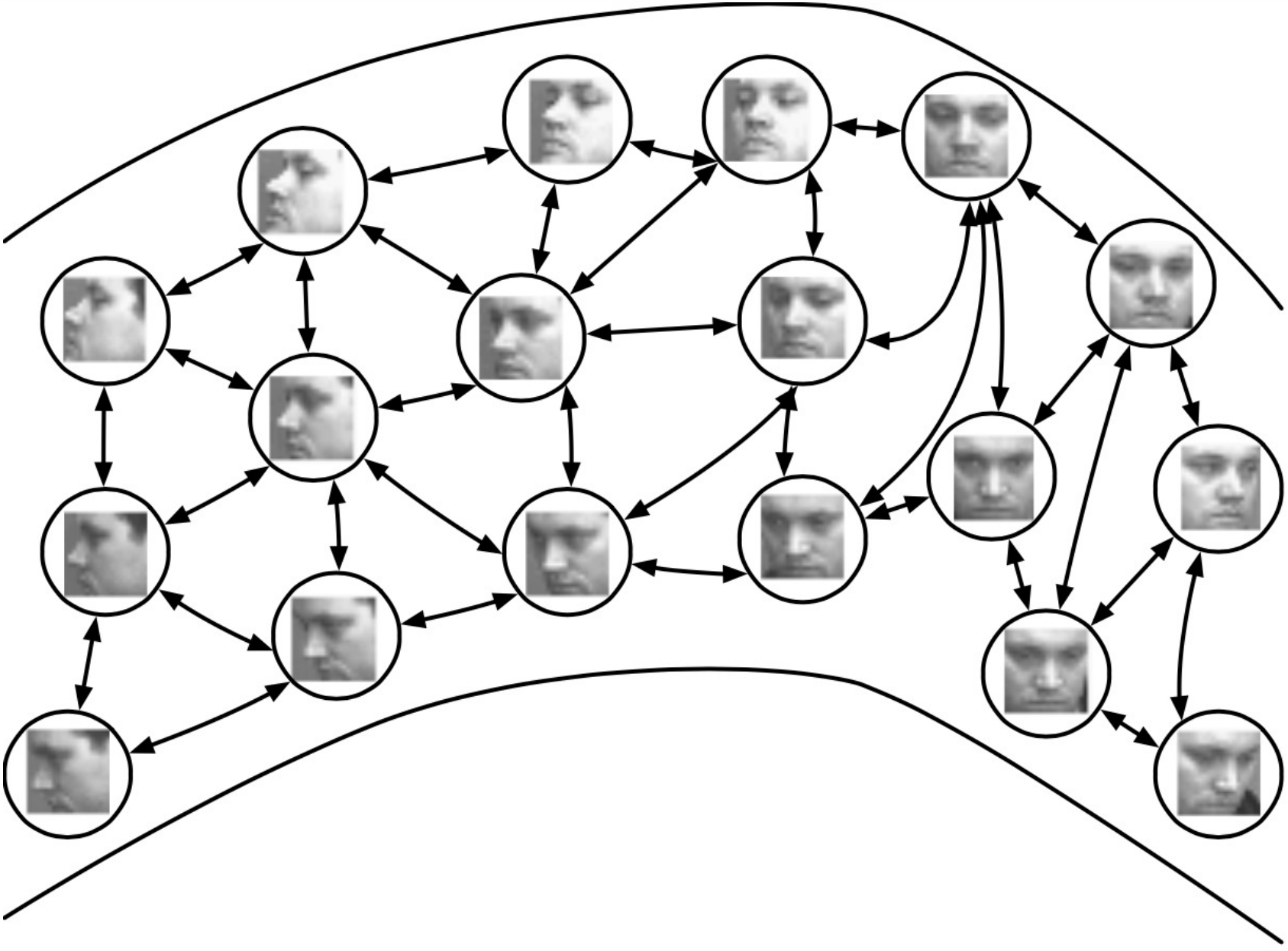


Isometric Feature Mapping (IsoMap)

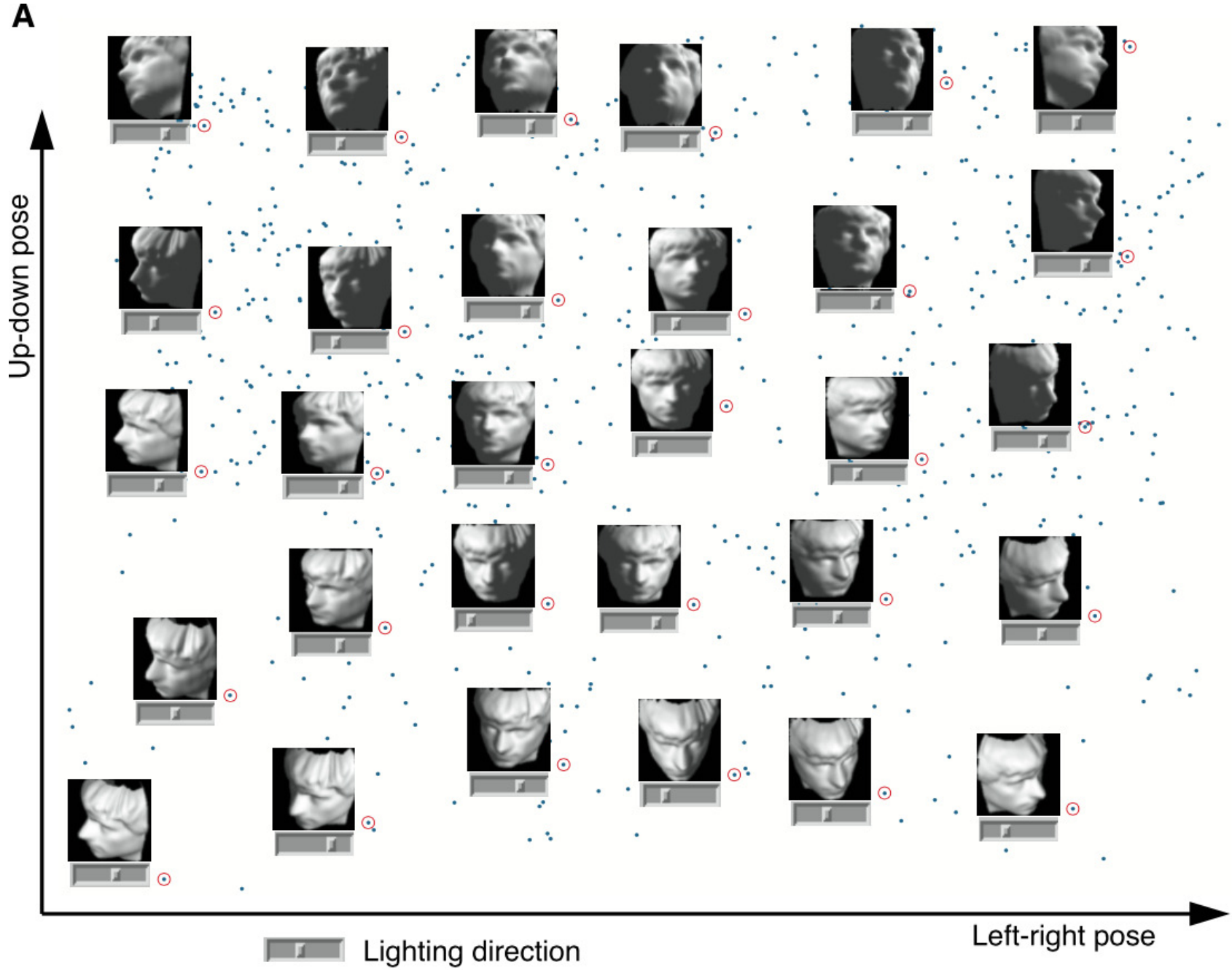


1. Construct local neighbourhood graph.
2. Compute shortest paths for each pair of points in the graph.
3. Apply MDS on geodesic instead of Euclidean distances.

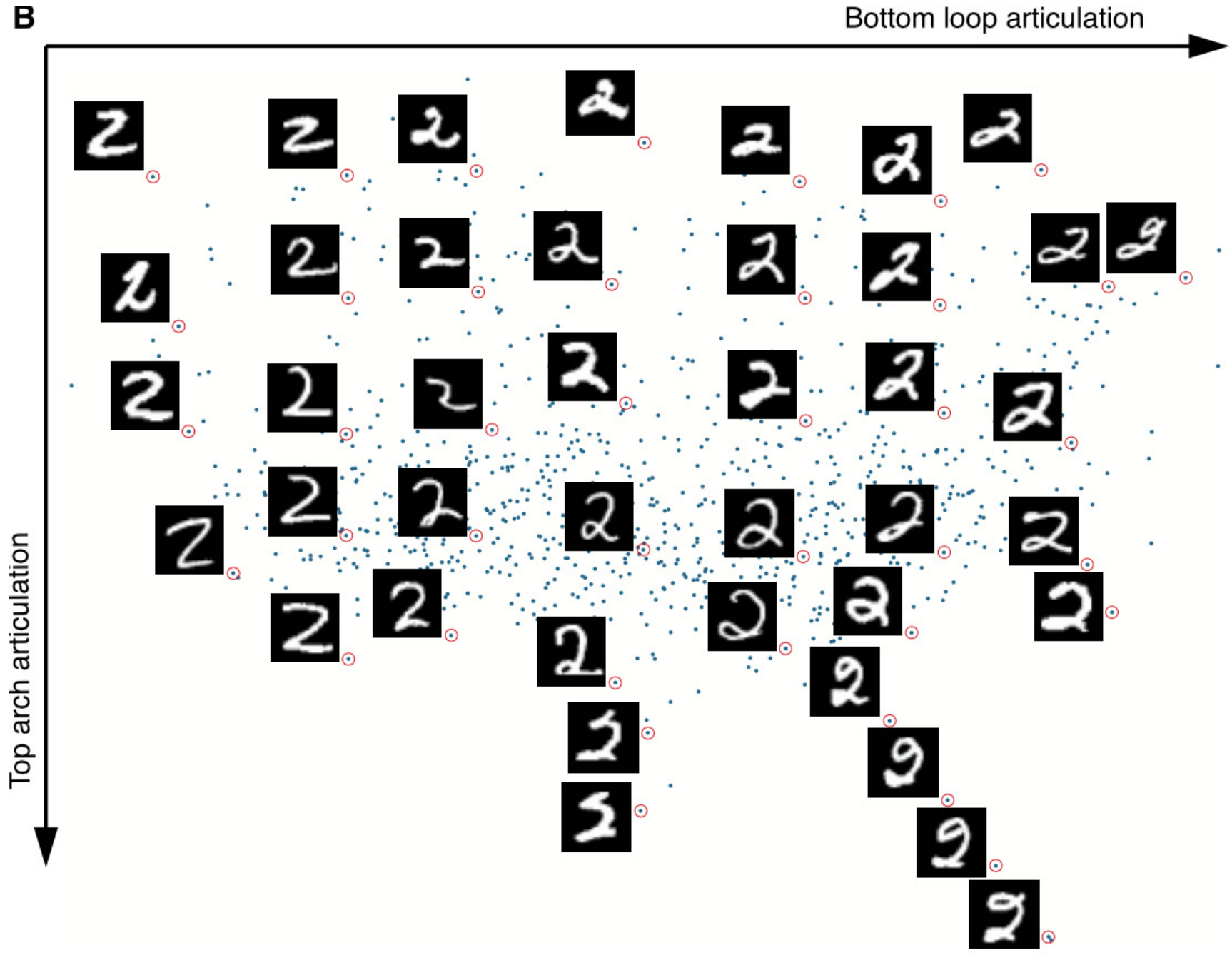
Nearest Neighbour Graph for Manifold Learning



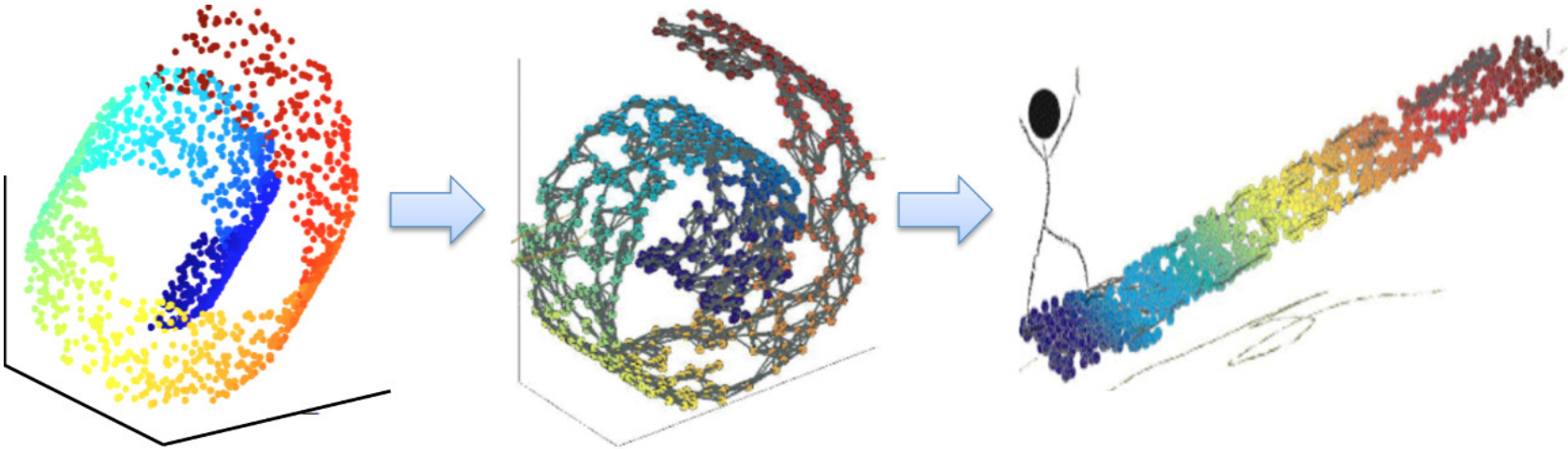
IsoMap Face Embedding Results



IsoMap Digit Embedding Results



Laplacian Eigenmaps: Nearby Points Stay Close



1. Construct a local neighbourhood graph.
2. Compute the weights between nodes.

$$W_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}} \tag{11}$$

3. Compute the eigenvectors of the graph Laplacian.

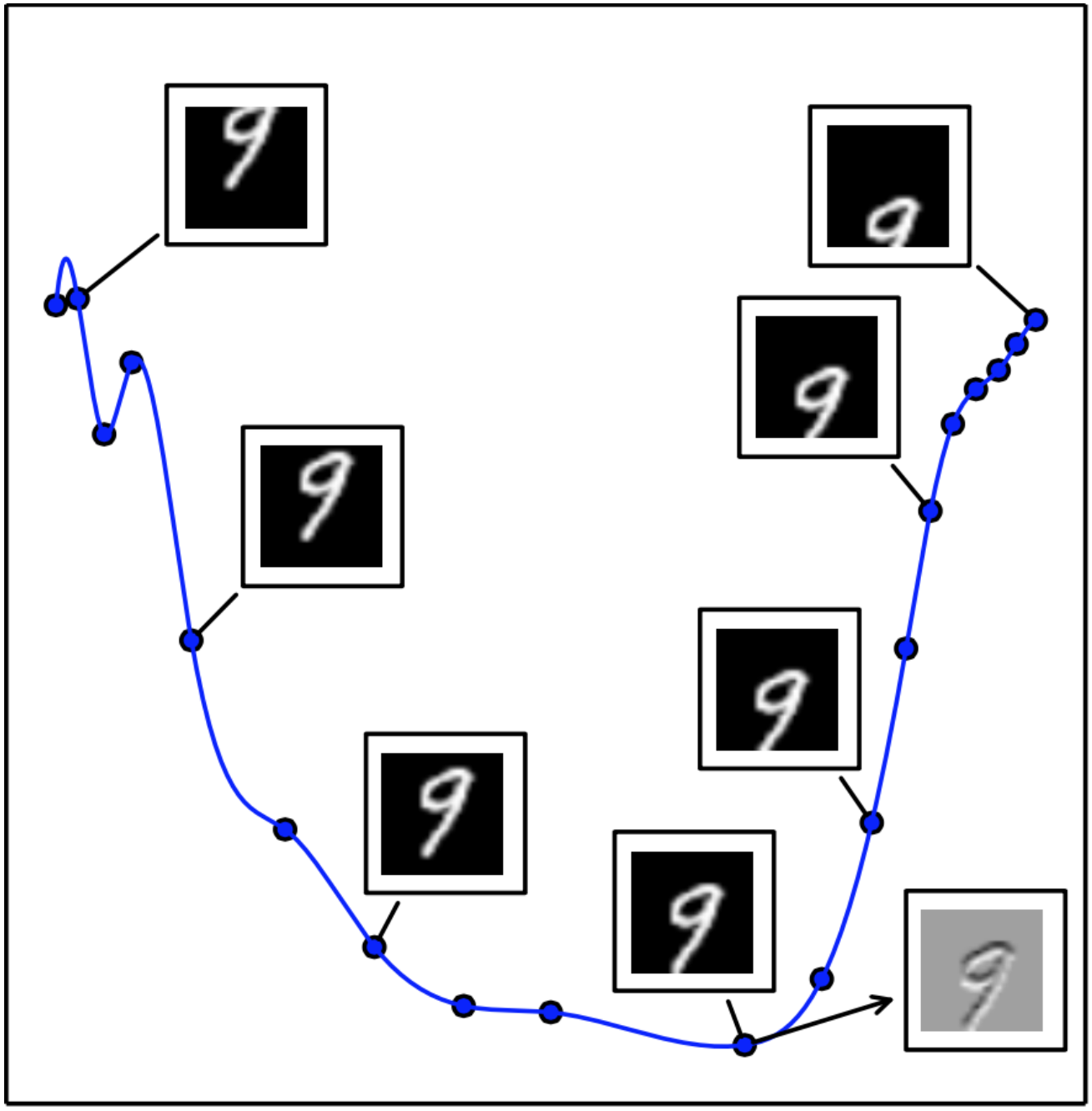
criterion: $$\min_y \varepsilon(y; W) = \sum_{i,j} W_{ij}(y_i - y_j)^2 \tag{12}$$

eigensolution: $$Ly = \lambda Dy \tag{13}$$

degree matrix: $$D = \mathrm{Diag}(W1) \tag{14}$$

Laplacian matrix: $$L = D - W \tag{15}$$

Manifold: Tangent Hyperplane



Embedding: Local Tangents → Global Positioning

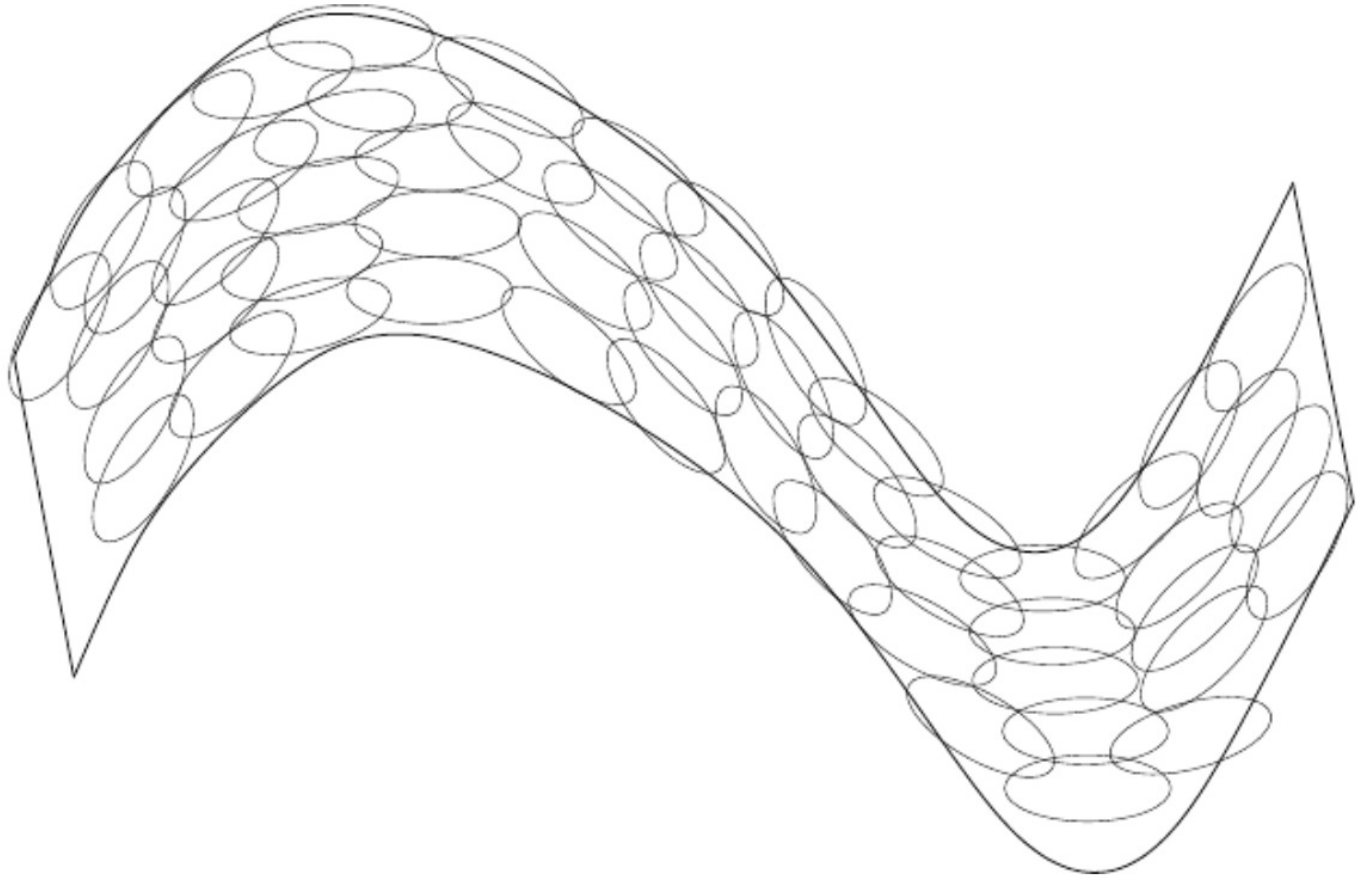


Figure 14.9: If the tangent planes (see figure 14.6) at each location are known, then they can be tiled to form a global coordinate system or a density function. Each local patch can be thought of as a local Euclidean coordinate system or as a locally flat Gaussian, or "pancake," with a very small variance in the directions orthogonal to the pancake and a very large variance in the directions defining the coordinate system on the pancake. A mixture of these Gaussians provides an estimated density function, as in the manifold Parzen window algorithm (Vincent and Bengio, 2003) or its non-local neural-net based variant (Bengio *et al.*, 2006c).

t-Distributed Stochastic Neighbour Embedding (tSNE)

1. Compute pairwise feature similarity as Gaussian probability distributions in the original data space

2. Define a new feature similarity as t-distributions in the low-dimensional space, addressing different point densities in low vs. high dimensional spaces.

3. Minimize the KL divergence between two conditional probability distributions.

4. Iterative optimization via gradient descent.

Feature Similarity: Joint Probability Distributions

- Gaussian distribution in high-dimensional space

$$p_{ij} \propto e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}, \quad p_{ii} = 0 \tag{16}$$

- Minimize KL divergence between p (fixed) and q (flexible):

$$\min \quad KL(p||q) = \sum_{j \neq i} p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{17}$$

  Large p_{ij} modeled by small $q_{ij} \Rightarrow$ large penalty
  Small p_{ij} modeled by large $q_{ij} \Rightarrow$ small penalty

- Ideally, $p_{ij} = q_{ij} \Rightarrow$ how can it be achieved in the embedding?

- What is the choice for the probability distribution of the embedding from high-D to low-d space?

The Geometry of High-Dimensional Spaces

$$\text{volume of outer ball} \propto r^d \tag{18}$$

$$\text{volume of inner ball} \propto (r-\varepsilon)^d \tag{19}$$

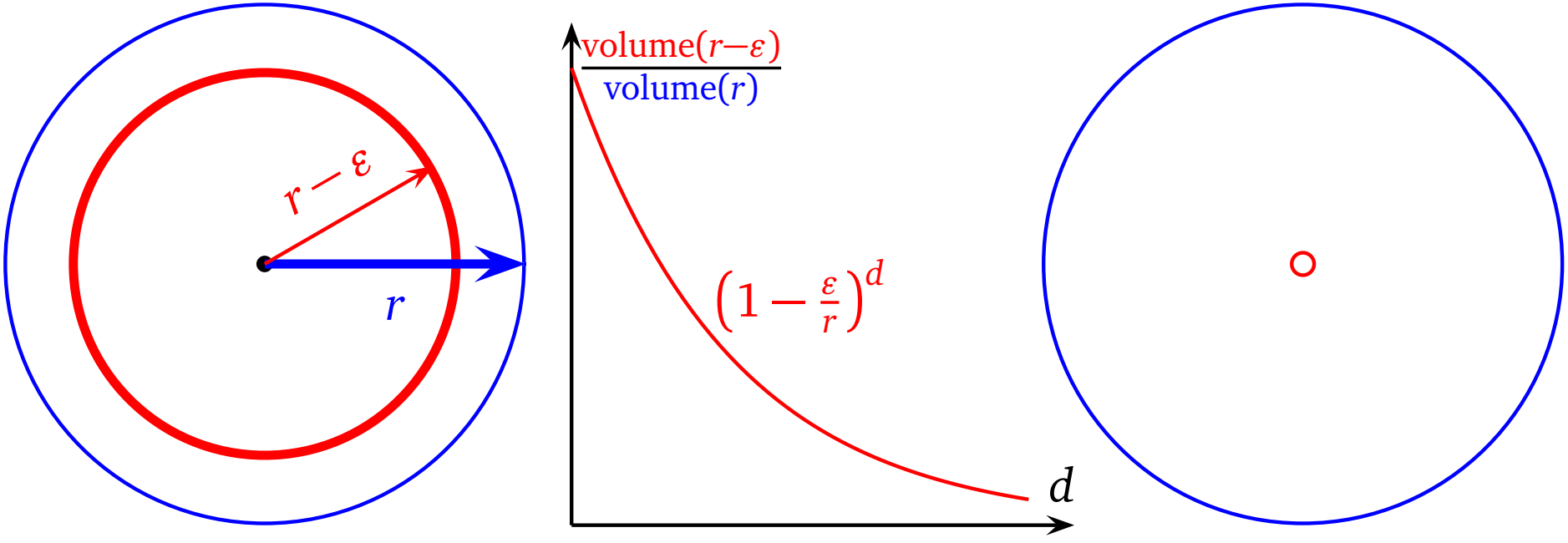
$$\text{ratio of inner-to-outer ball volume} = \tag{20}$$

$$\frac{(r-\varepsilon)^d}{r^d} = \left(1-\frac{\varepsilon}{r}\right)^d \approx \exp\left(-\frac{\varepsilon d}{r}\right) \to 0, \quad \text{as } d \to \infty \tag{21}$$

$\frac{\varepsilon}{r}$	$d=2$	$d=10$	$d=100$	$d=1000$
0.01	0.9801	0.9044	0.3660	$0.99^{1000} \approx 0.0000$
0.25	0.5625	0.0563	0.0000	0.0000
0.5	0.2500	0.0010	0.0000	0.0000

Uniform Distribution in High-Dimensional Balls

$$\lim_{d\to\infty} \frac{\text{volume}(r-\varepsilon)}{\text{volume}(r)} = 0$$

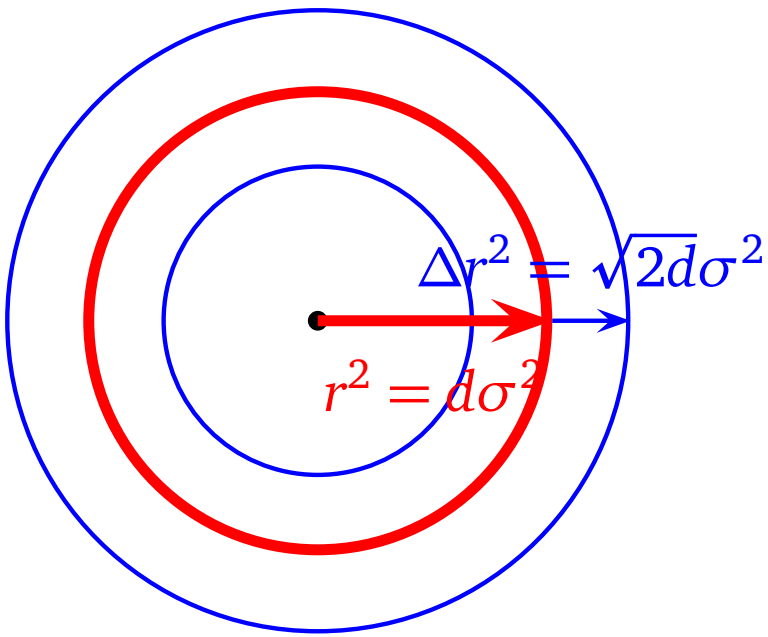


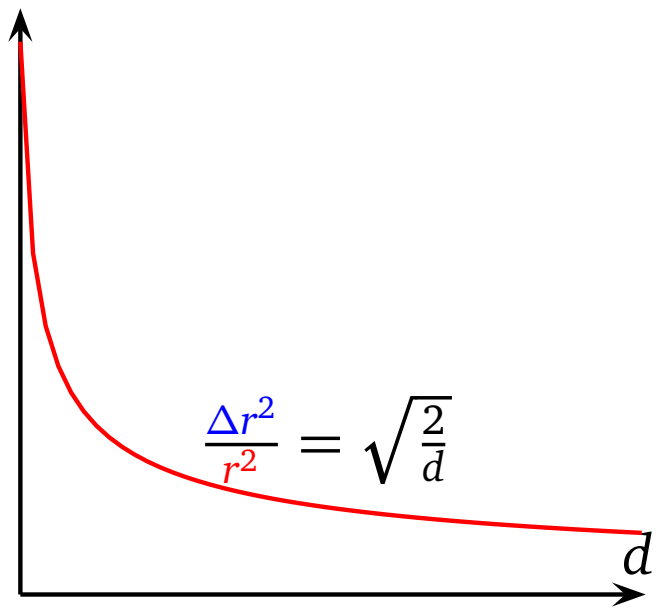
- Random points from uniform distributions in ball: nearly all are in outer shell.

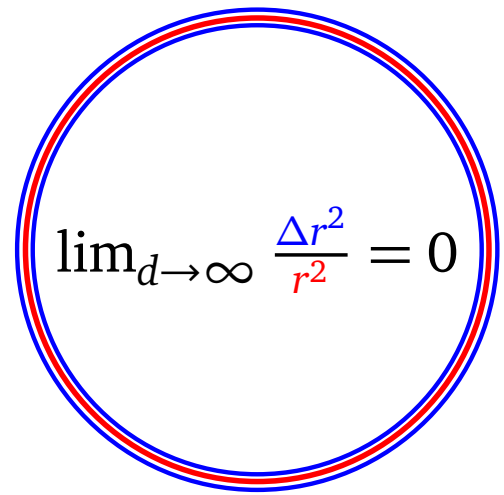
High-Dimensional Gaussians

$$X \sim \mathcal{N}(0, \sigma^2), \quad p(x) = \frac{1}{(2\pi\sigma)^{\frac{d}{2}}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{d} x_i^2\right) \tag{22}$$

$$r^2 = \sum_{i=1}^{d} x_i^2 \sim \mathcal{N}(d\sigma^2, (\sqrt{2d}\sigma^2)^2) \tag{23}$$







- Random points from Gaussian distributions in ball: nearly all are in some thin shell.
- Curse of Dimensionality: All k-NNs are actually not that near!

SNE vs. tSNE: Gaussian vs. Heavy-tail t-Dist.

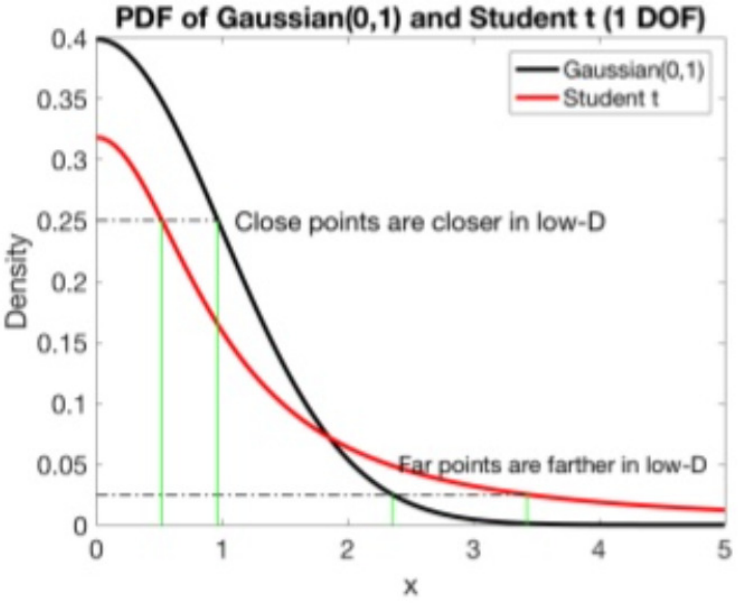
- SNE: Gaussian distribution in low-dimensional space

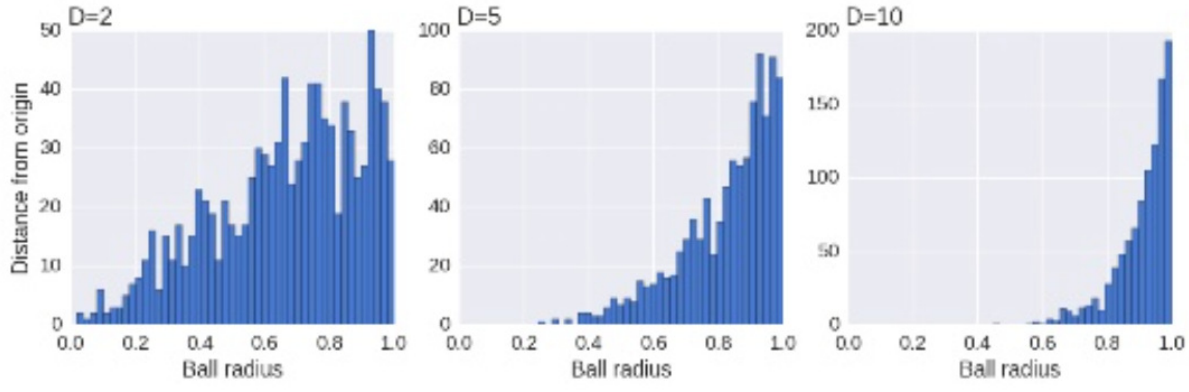
$$q_{ij} \propto e^{-\|y_i - y_j\|^2}, \quad q_{ii} = 0 \tag{24}$$

- tSNE: heavy-tail t-distribution in low-dimensional space:

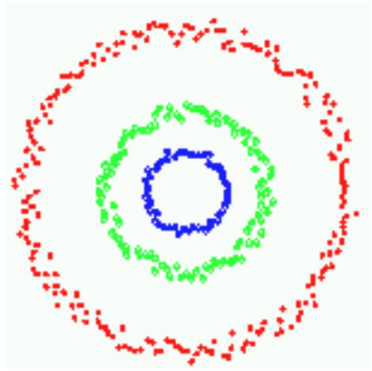
$$q_{ij} \propto (1 + \|y_i - y_j\|^2)^{-1}, \quad q_{ii} = 0 \tag{25}$$

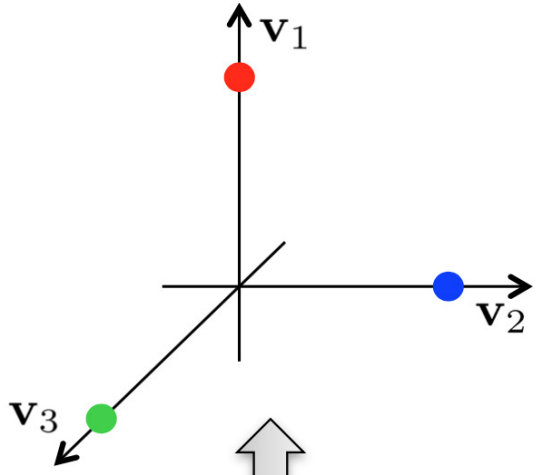
- SNE $\sim t(x, \infty)$, tSNE $\sim t(x, 1)$. $\quad t(x, n) = (1 + \frac{x^2}{n})^{-\frac{n+1}{2}}$.



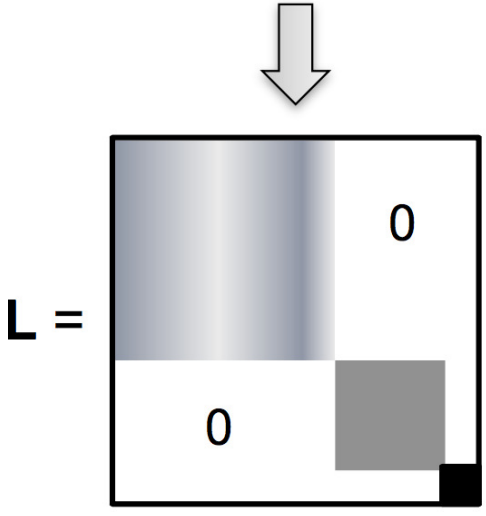


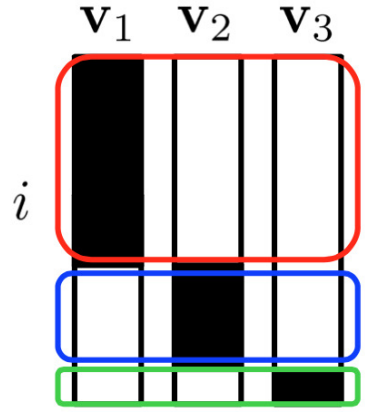
Unsupervised Learning: Spectral Clustering





Points are easy to cluster in embedded space e.g. using k-means





Embedding of point i

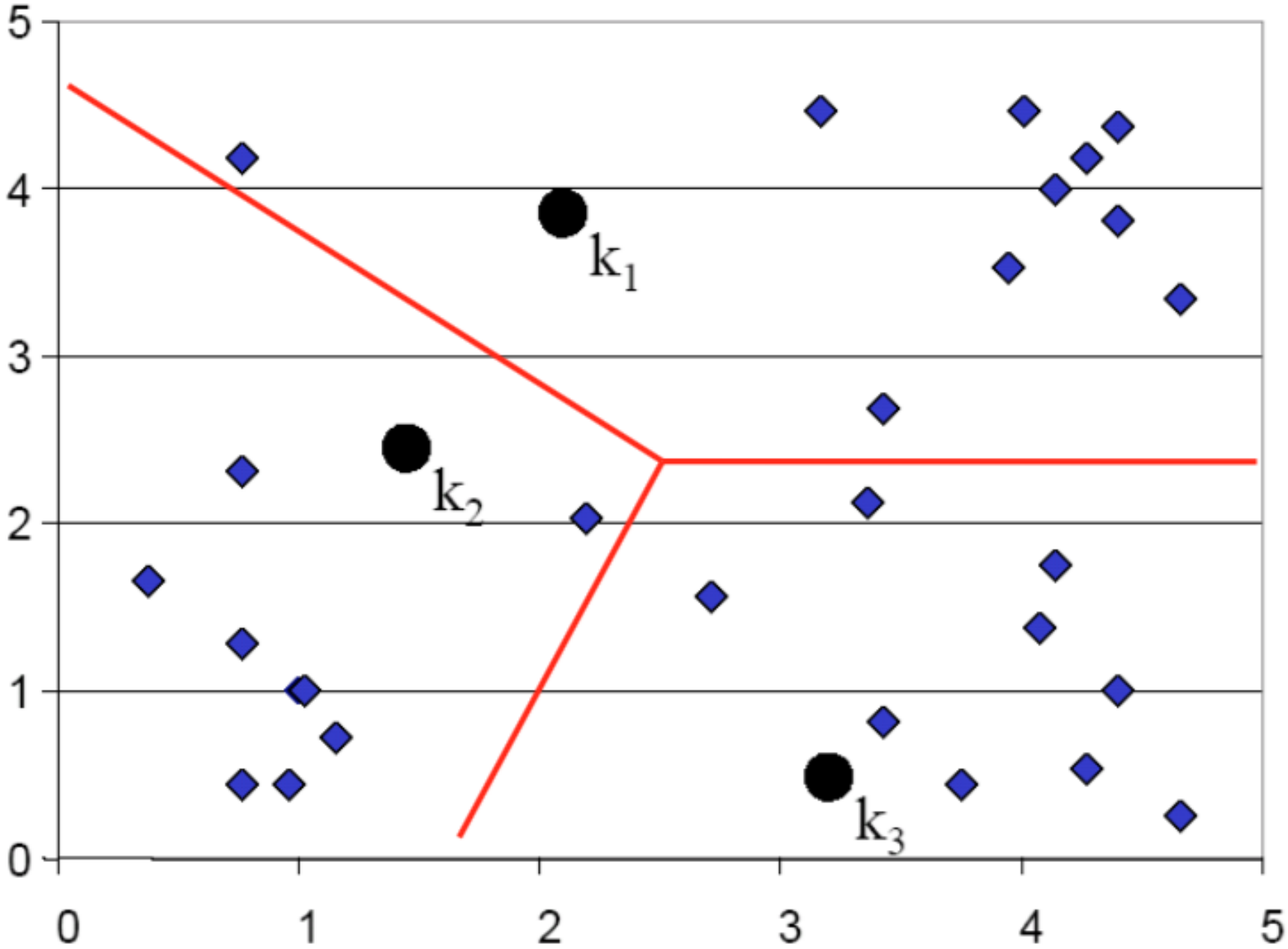
$[\mathbf{v}_1(i), \mathbf{v}_2(i), \mathbf{v}_3(i)]$

1. Compute the spectral embedding via Laplacian eigenmaps.
2. Cluster the embedded points, via e.g. K-means.

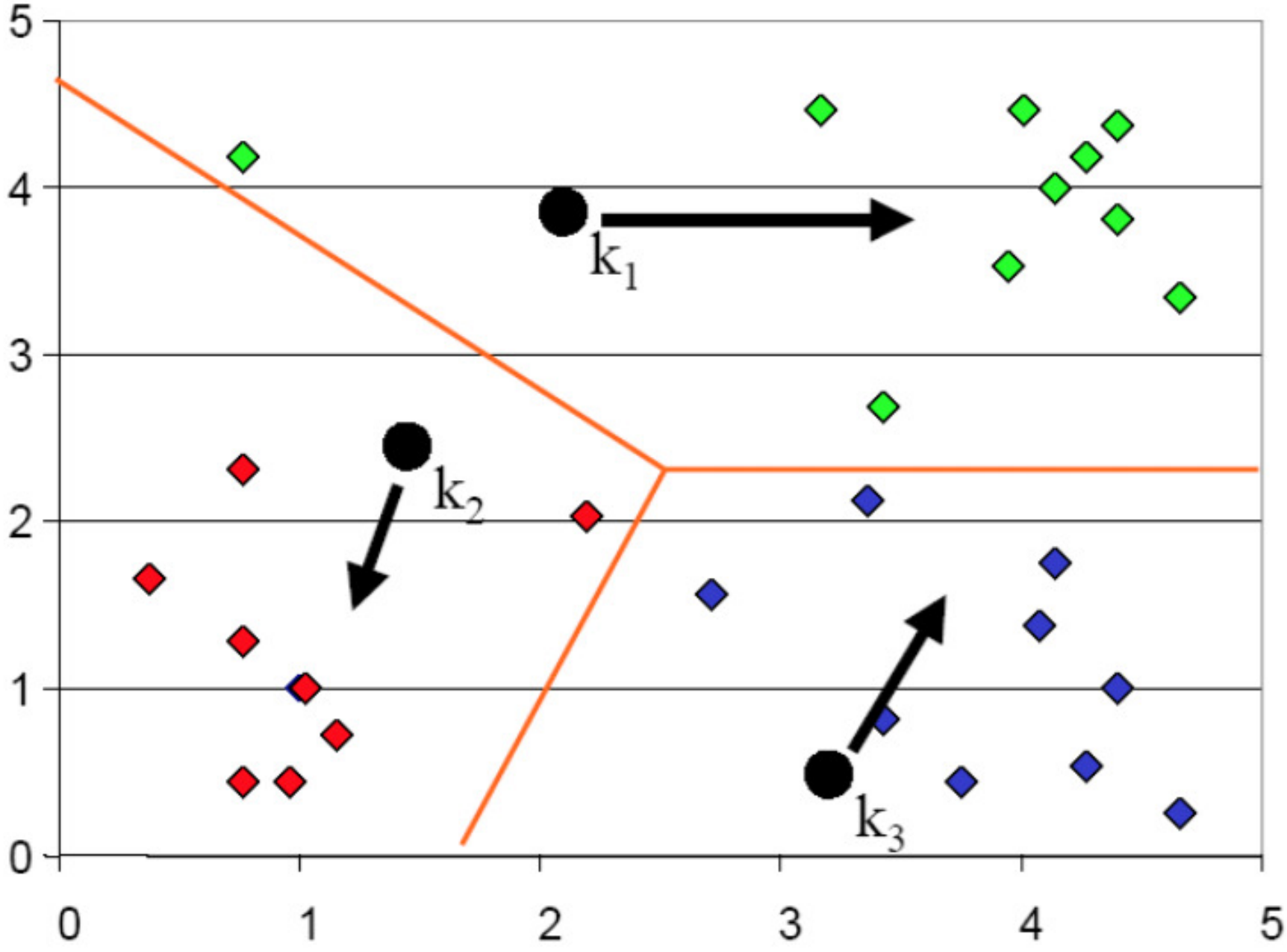
K-Means Clustering Algorithm

- Input: Desired number of clusters, k
- Initalize the k cluster centers (randomly if necessary)
- Iterate the following two steps.
  1. Assign the objects to the nearest cluster centers
  2. Re-estimate the k cluster centers
- Terminate:
  If none of the assignments changed in the last iteration, exit.
  Otherwise go to 1.

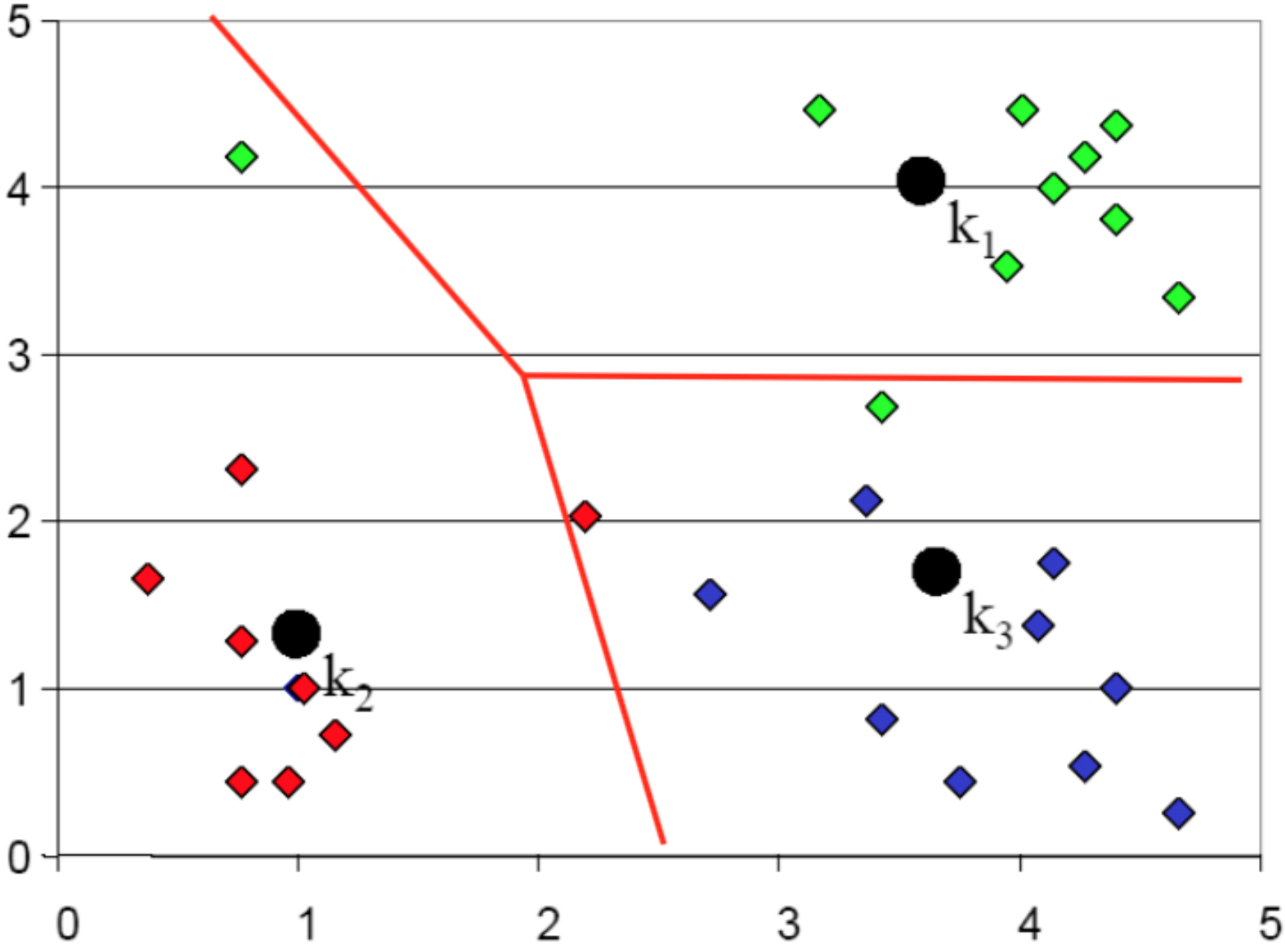
K-means Clustering: Step 1



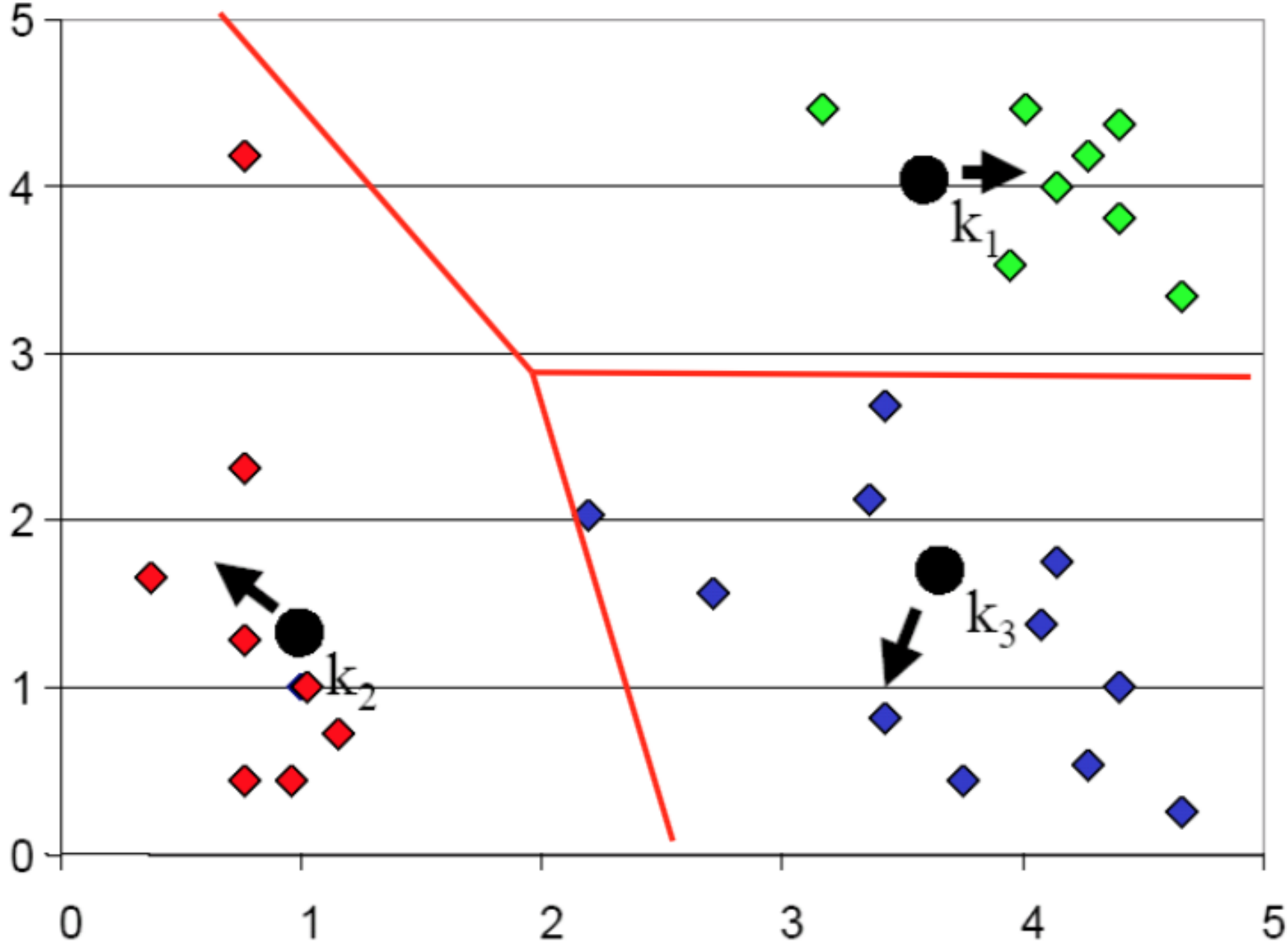
K-means Clustering: Step 2



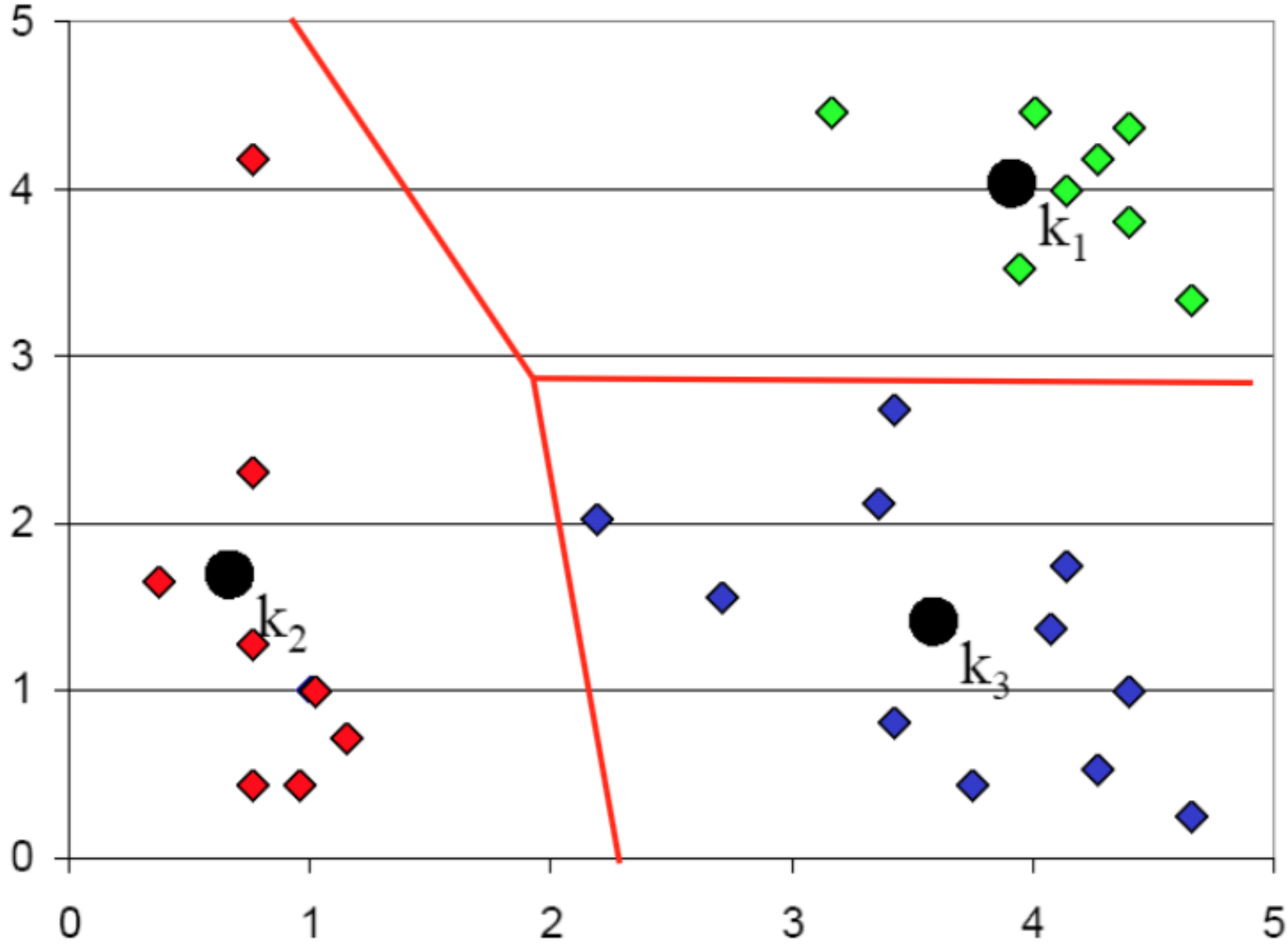
K-means Clustering: Step 3



K-means Clustering: Step 4



K-means Clustering: Step 5

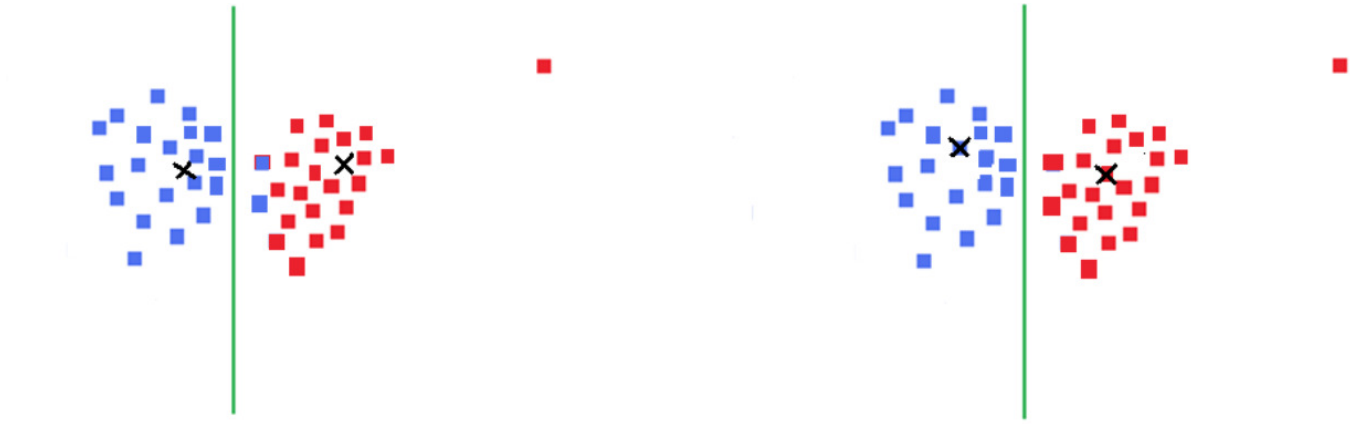


K-means Sensitive to Seed Choice

K-means Sensitive to Seed Choice

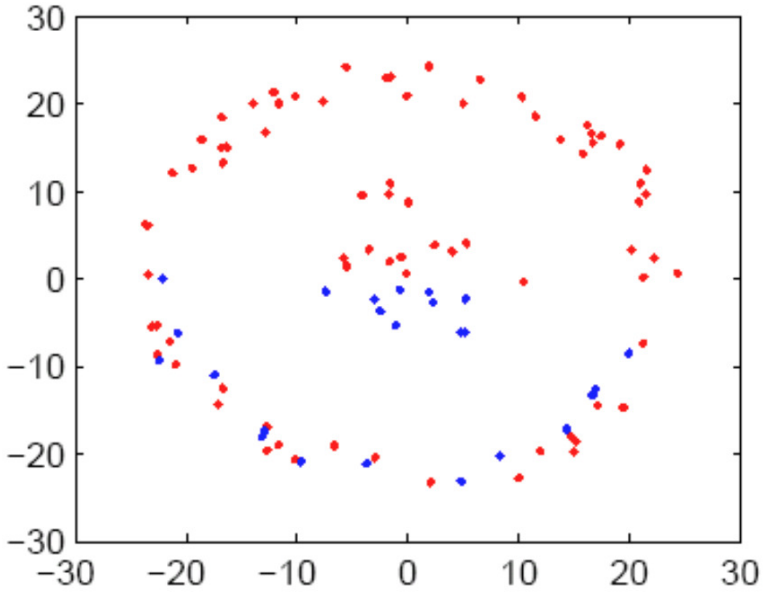
K-means: Pros and Cons

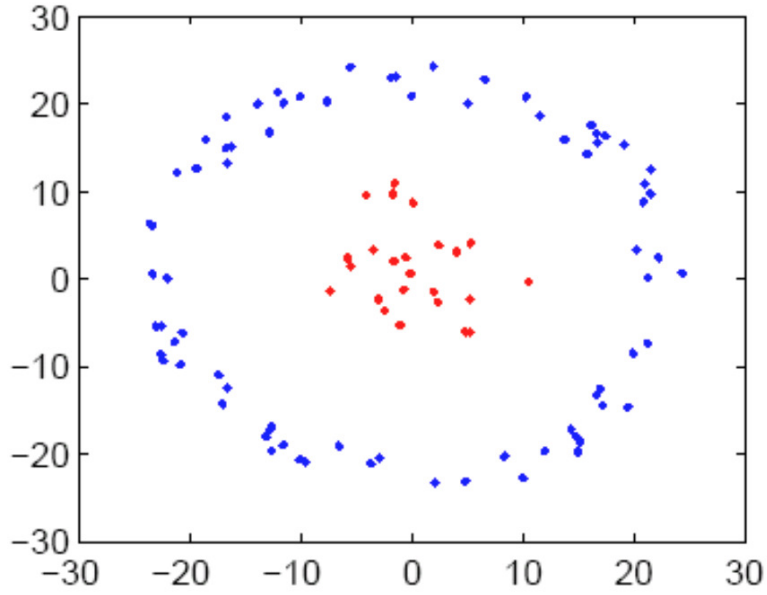
- Simple and easy for exploratory data analysis.
- Results can vary based on random seed selection.
- Need good initializations: e.g. multiple random seeds.
- Sometimes it is hard to select k.
- Sensitive to outliers $\rightarrow$ use k-medoids



- K-means assumes isotropic convex clusters.

K-means vs. Spectral Clustering





Applying k-means to Laplacian eigenvector embeddings allows us to find non-isotropic and non-convex clusters.

Summary

1. Dimensionality reduction is needed not only for more effective and efficient feature learning, but also for feature visualization and unsupervised exploratory data analysis.
2. Linear methods take the data matrix and try to factorize it into either a latent representation with a dictionary for projecting the data (PCA, ICA NMF), or an embedding itself (MDS). Linear methods are global, preserving all inter-point distances. They cannot discover inherent data structures.
3. Nonlinear methods include data-independent nonlinear kernel methods and data-dependent manifold learning. IsoMap, Laplacian Eigenmap, tSNE vary on what and how local relationships are preserved.
4. Manifold learning and clustering are closely related unsupervised learning methods. Spectral clustering is simply k-means applied to the Laplacian embedding, handling non-isotropic and non-convex clusters.