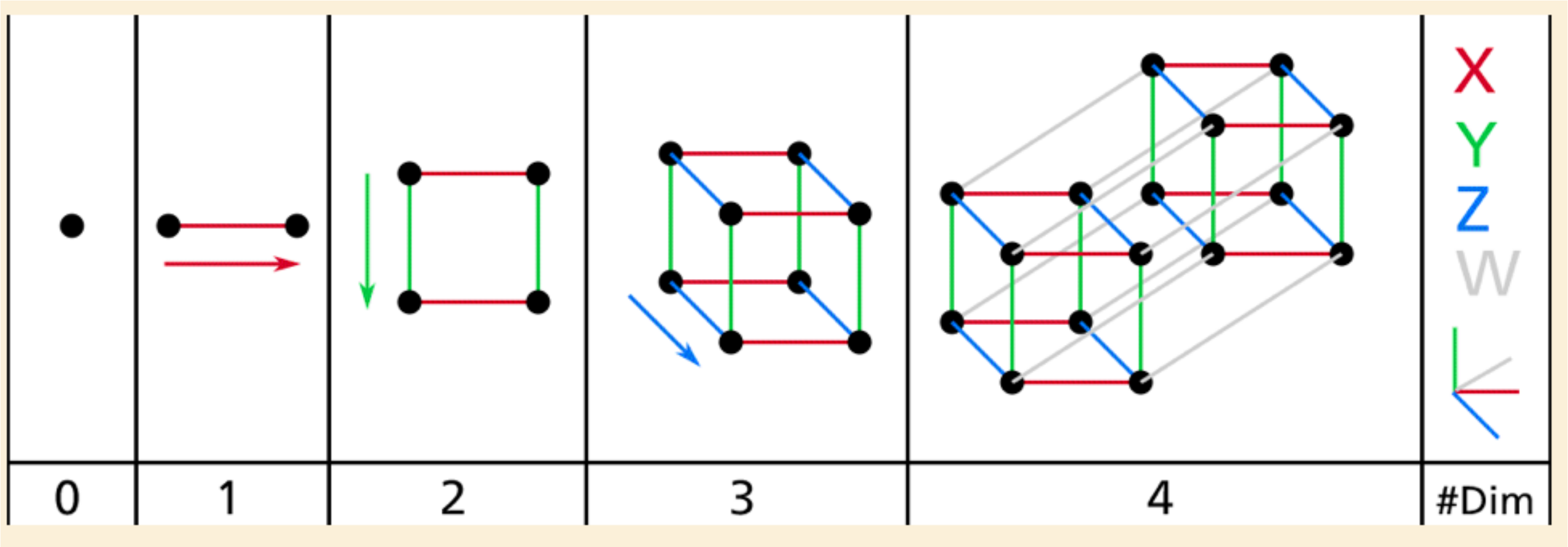
Machine Learning

Clustering and Dimensionality Reduction

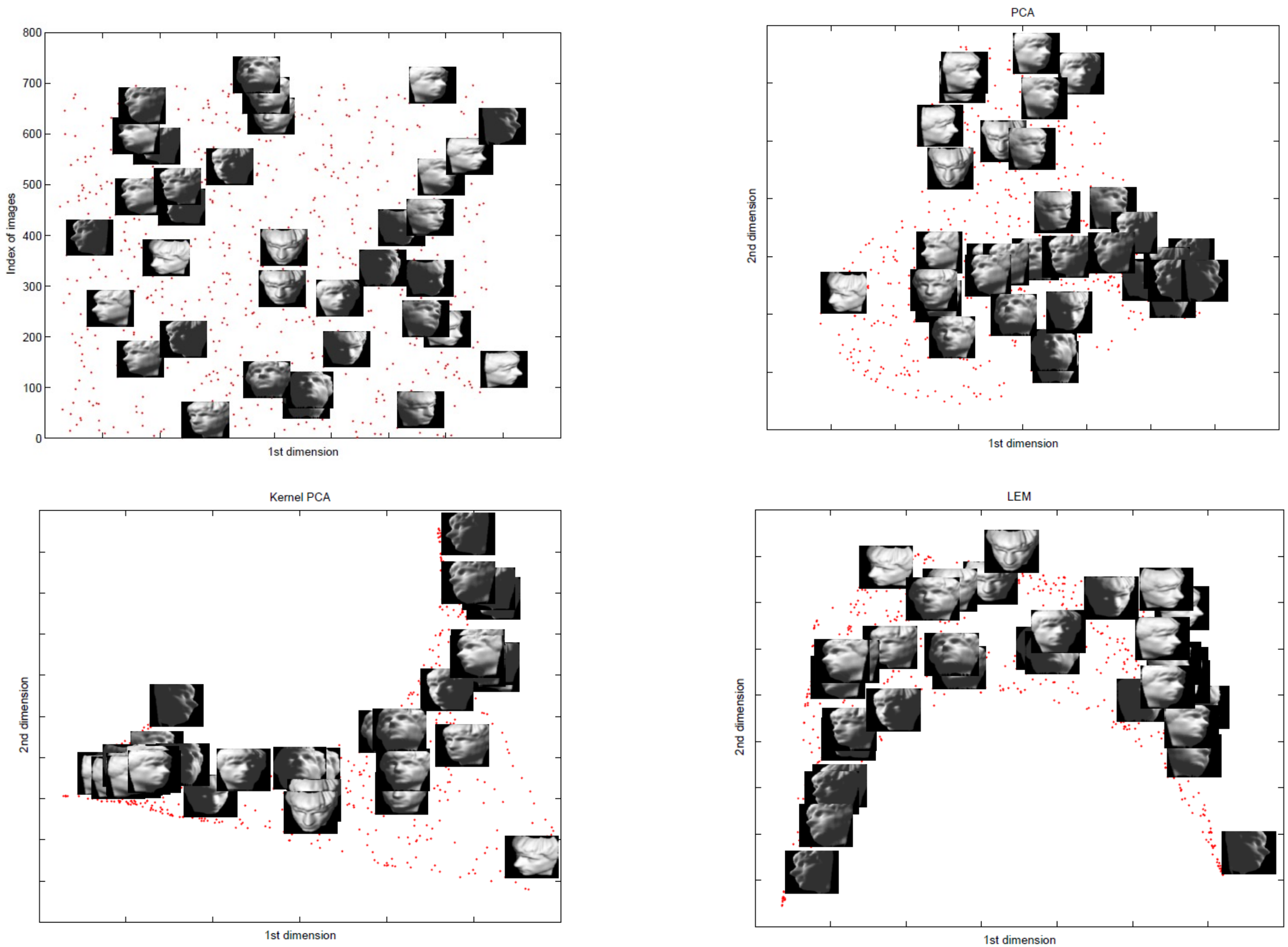
Acknowledge: The slides are partially referred to coursera online machine learning course by Prof. Andrew Ng, and NYU machine learning course.

Prof. Yi Fang (yfang@nyu.edu)

Data Examples



Data Examples



Curse of Dimensionality

- Cursed phenomena occur in domains such as numerical analysis, sampling, combinatorics, machine learning, data mining and databases. The common theme of these problems is that when the dimensionality increases, the volume of the space increases so fast that the available data become sparse. This **sparsity** is problematic for any method that requires statistical significance. In order to obtain a statistically sound and reliable result, the amount of data needed to support the result often grows exponentially with the dimensionality. Also, organizing and searching data often relies on detecting areas where objects form groups with similar properties; in high dimensional data, however, all objects appear to be sparse and dissimilar in many ways, which prevents common data organization strategies from being efficient.

Why dimensionality reduction (DR)?

- Some features may be irrelevant
- We want to visualize high dimensional data
- “Intrinsic” dimensionality may be smaller than the number of features

DR Models

- Principal Component Analysis (PCA)
- Kernel PCA
- Locally Linear Embedding (LLE)
- Laplacian Eigenmaps (LEM)
- Multidimensional Scaling (MDS)
- ISOMAP
- Semidefinite Embedding (SDE)
- Unified Framework

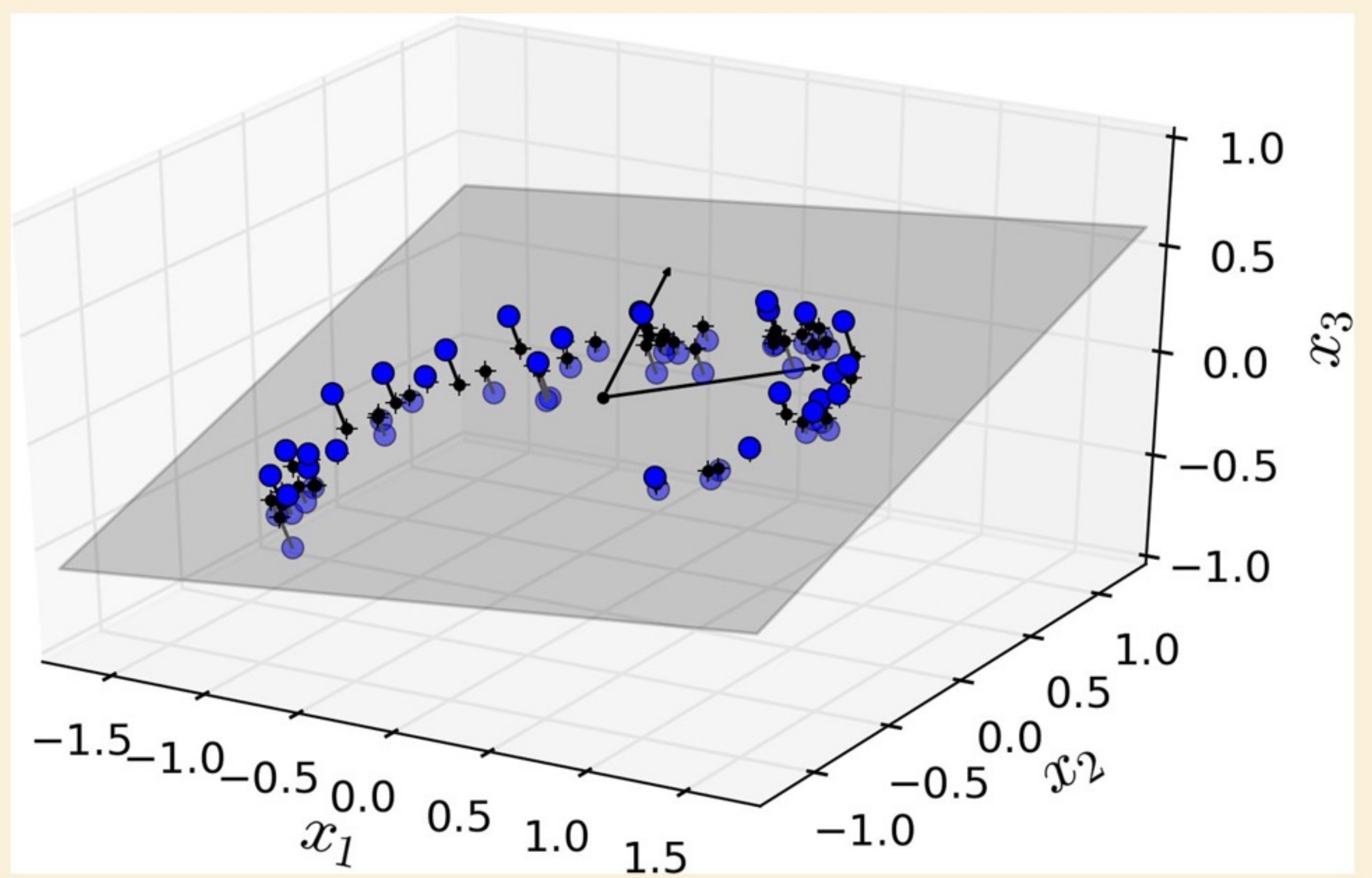
DR Models

- Linear methods
  - Principal component analysis (PCA)
  - Multidimensional scaling (MDS)
  - Independent component analysis (ICA)
- Nonlinear methods
  - Kernel PCA
  - Locally linear embedding (LLE)
  - Laplacian eigenmaps (LEM)
  - Semidefinite embedding (SDE)

DR Approaches

- Projection
- Manifold Learning

Projection



After Projection

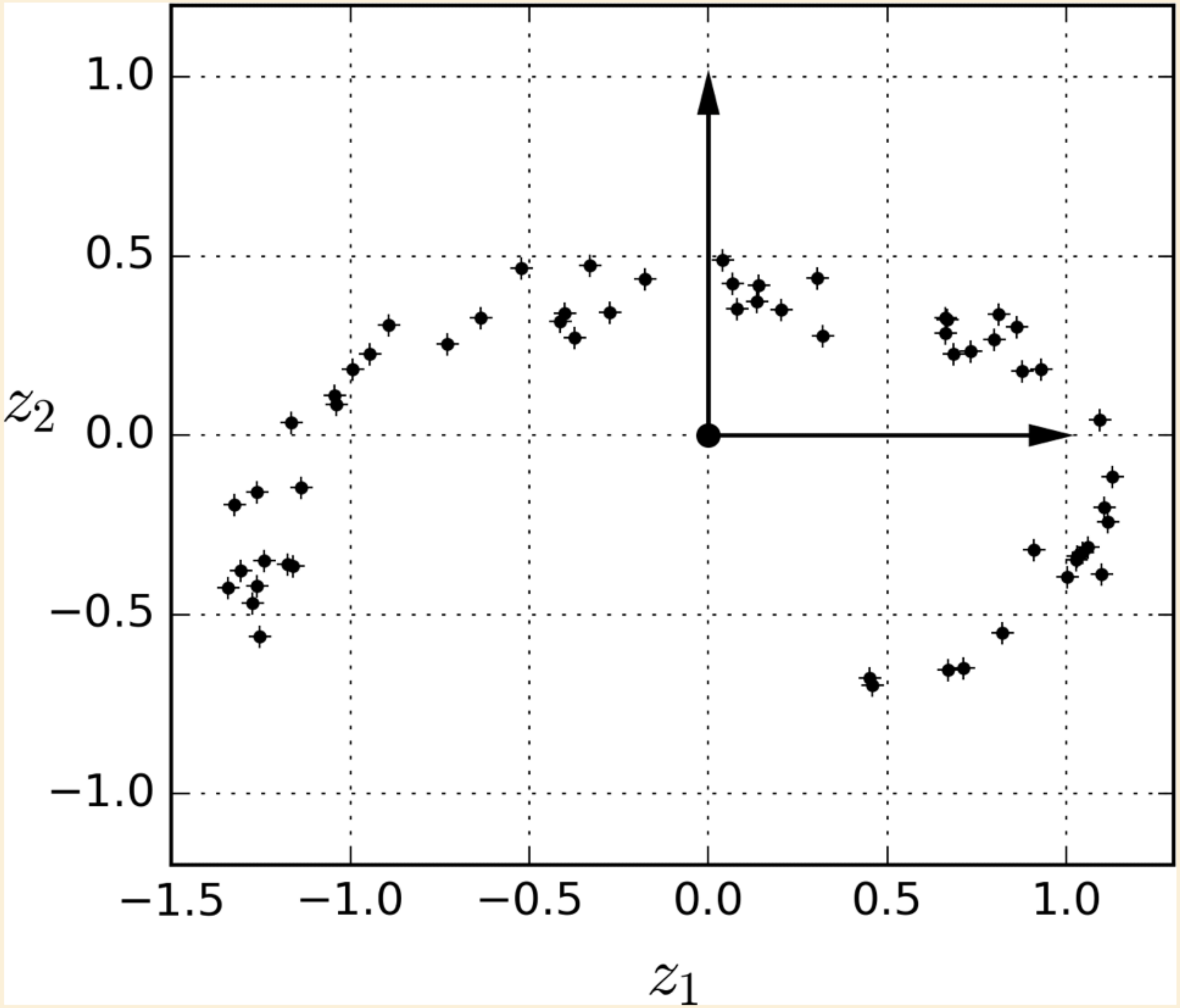


Figure 8-3. The new 2D dataset after projection

Projection Algorithm: PCA

- Principal Component Analysis (PCA) is by far the most popular dimensionality reduction algorithm. First it identifies the hyperplane that lies closest to the data, and then it projects the data onto it.

Key Mechanism: Preserving the Variance of data

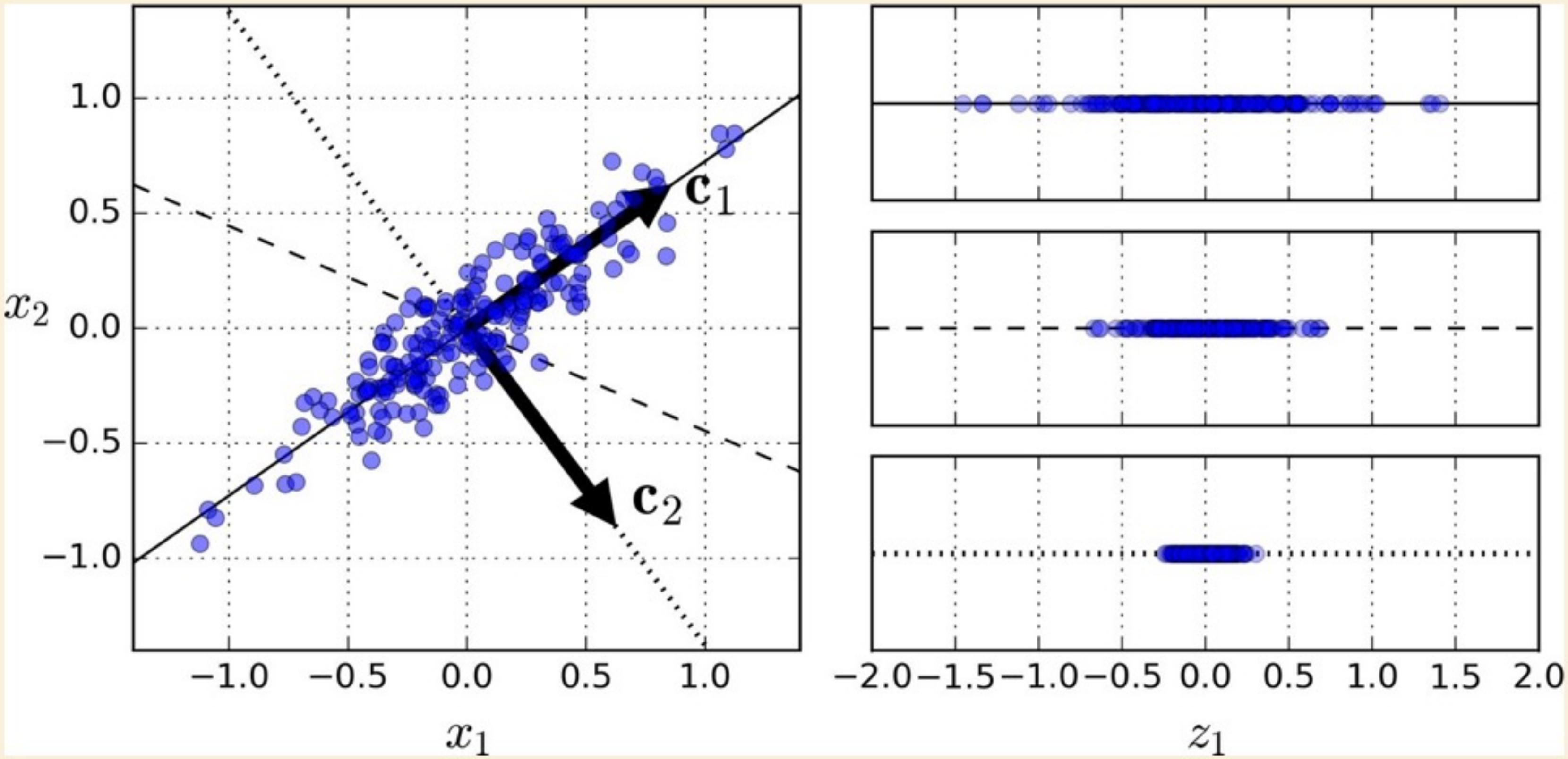


Figure 8-7. Selecting the subspace onto which to project

Preserving the Variance

- Before you can project the training set onto a lower-dimensional hyperplane, you first need to choose the right hyperplane. For example, a simple 2D dataset is represented on the left of Figure above , along with three different axes (i.e., one-dimensional hyperplanes). On the right is the result of the projection of the dataset onto each of these axes. As you can see, the projection onto the solid line preserves the maximum variance, while the projection onto the dotted line preserves very little variance, and the projection onto the dashed line preserves an intermediate amount of variance.
- It seems reasonable to select the axis that preserves the maximum amount of variance, as it will most likely lose less information than the other projections. Another way to justify this choice is that it is the axis that minimizes the mean squared distance between the original dataset and its projection onto that axis.

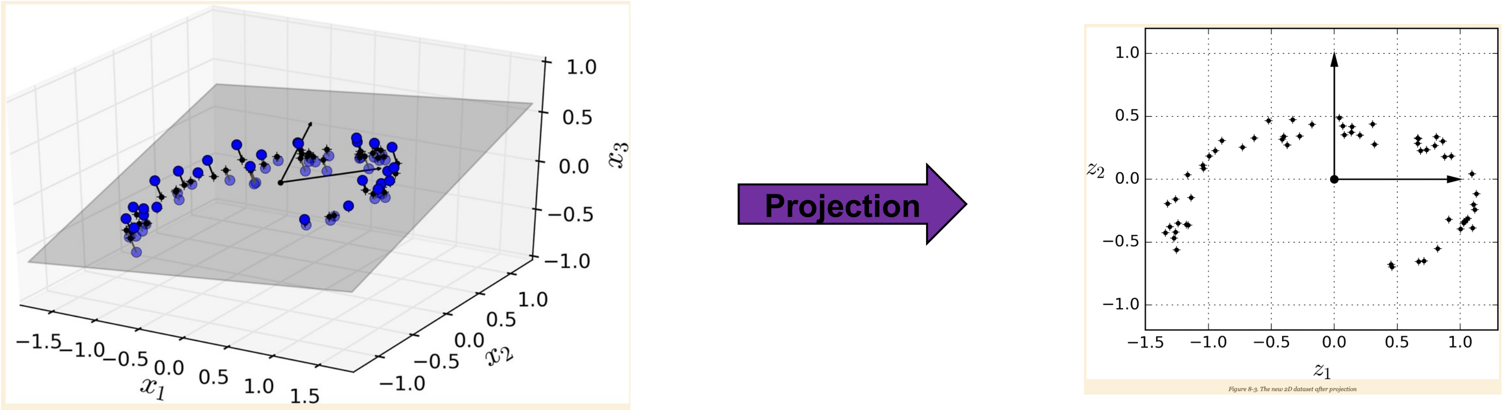
PCA Algorithm

- PCA algorithm:
  - 1. X ← Create N x d data matrix, with one row vector x_n per data point
  - 2. X subtract mean x from each row vector x_n in X
  - 3. Σ ← covariance matrix of X
  - Find eigenvectors and eigenvalues of Σ (through a standard matrix factorization Singular Value Decoposition (SVD))
  - PC's ← the M eigenvectors with largest eigenvalues

Projecting Down to d Dimensions

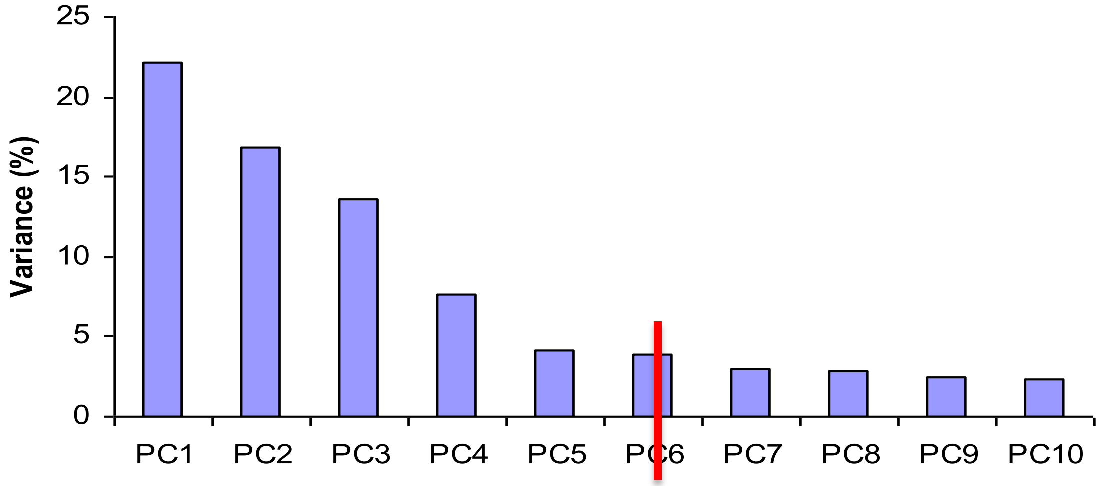
- Once we have identified all the principal components, you can reduce the dimensionality of the dataset down to d dimensions by projecting it onto the hyperplane defined by the first d principal components.

Projection: $\mathbf{X}_{d\text{-proj}} = \mathbf{X}\mathbf{W}_d$



How many components?

- Distribution of Eigen-values (energy percentage)
- Choose the first few eigen-vectors to cover 80-90% of the variance of the data



DR Models

- Principal Component Analysis (PCA)
- Kernel PCA
- Locally Linear Embedding (LLE)
- Laplacian Eigenmaps (LEM)
- Multidimensional Scaling (MDS)
- ISOMAP
- Semidefinite Embedding (SDE)
- Unified Framework

Kernel PCA

- KPCA algorithm:
  - 0: Kernel Tricks: Nonlinear mapping

$$\Phi : x \to \mathcal{H} \qquad x \mapsto \Phi(x)$$

  - 1. Update X with new mapped X ⟵ $\Phi(x)$
  - 2. X ← Create N x d data matrix, with one row vector x_n per data point
  - 3. X subtract mean *x* from each row vector x_n in X
  - 4. Σ ← covariance matrix of X
  - Find eigenvectors and eigenvalues of Σ (through a standard matrix factorization Singular Value Decoposition (SVD))
  - PC's ← the M eigenvectors with largest eigenvalues

Kernel PCA

- History: S. Mika et al, NIPS, 1999
- Data may lie on or near a nonlinear manifold, not a linear subspace
- Find principal components that are nonlinearly to the input space via nonlinear mapping
- Objective

- Solution found by SVD:

  U contains the eigenvectors of

Approach: Manifold Learning

DR Models

- Principal Component Analysis (PCA)
- Kernel PCA
- Locally Linear Embedding (LLE)
- Laplacian Eigenmaps (LEM)
- Multidimensional Scaling (MDS)
- ISOMAP
- Semidefinite Embedding (SDE)
- Unified Framework

Manifold Learning

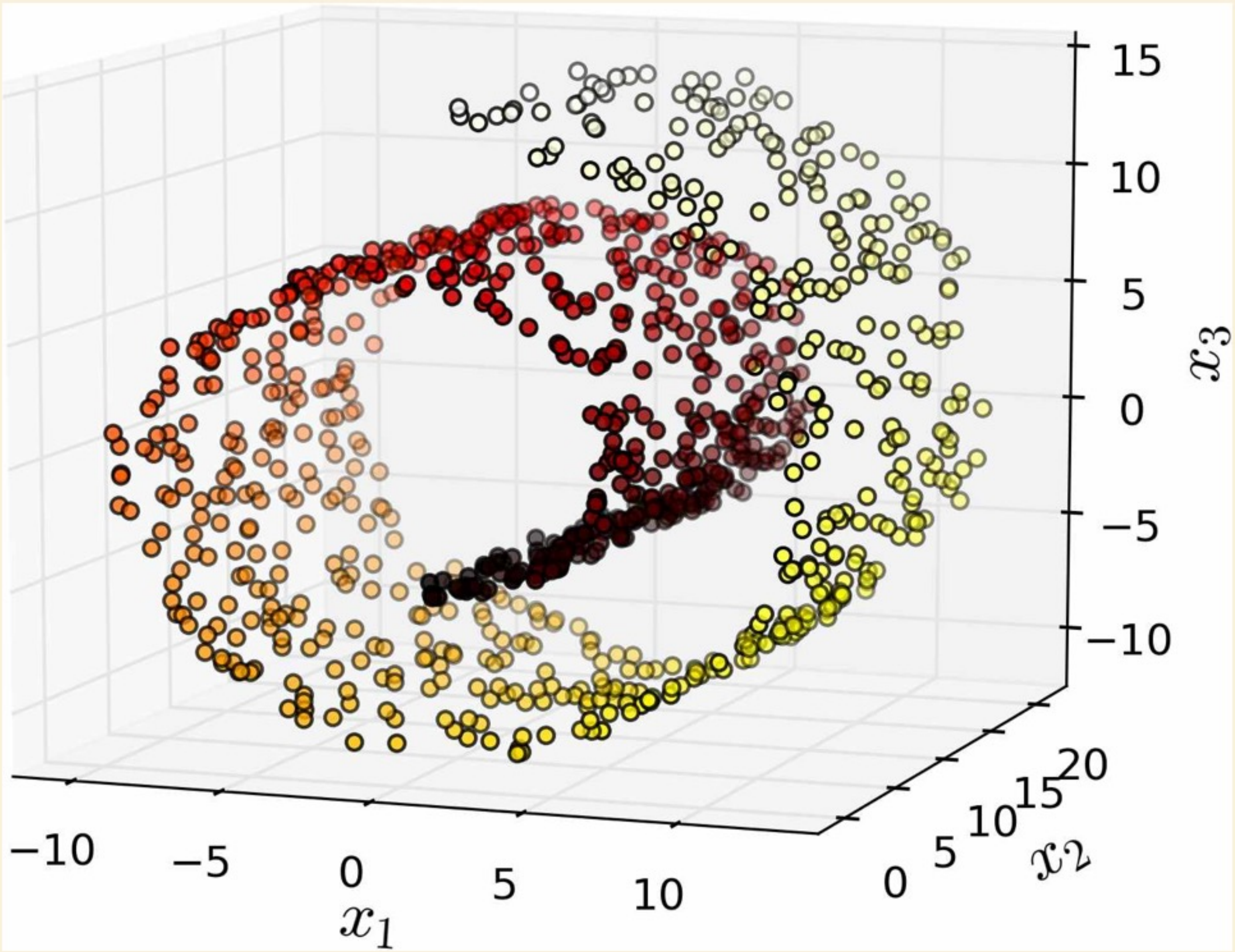


Figure 8-4. Swiss roll dataset

Projection Vs Unrolling

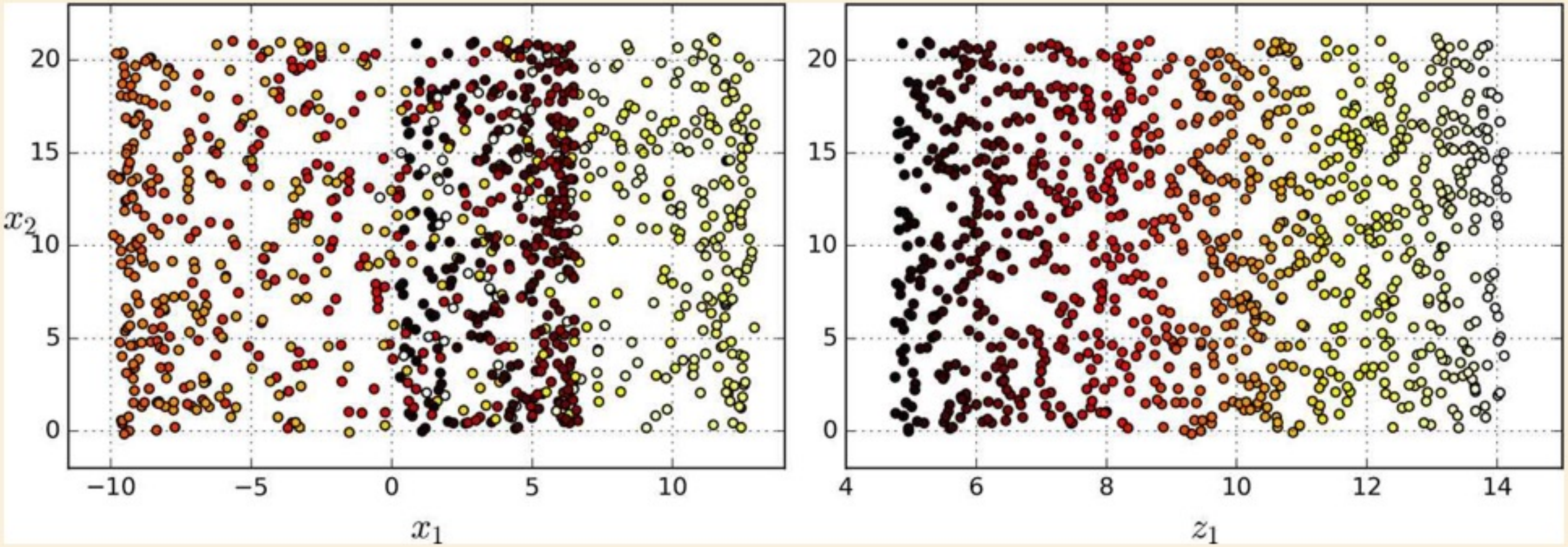


Figure 8-5. Squashing by projecting onto a plane (left) versus unrolling the Swiss roll (right)

Local Linearly Embedding

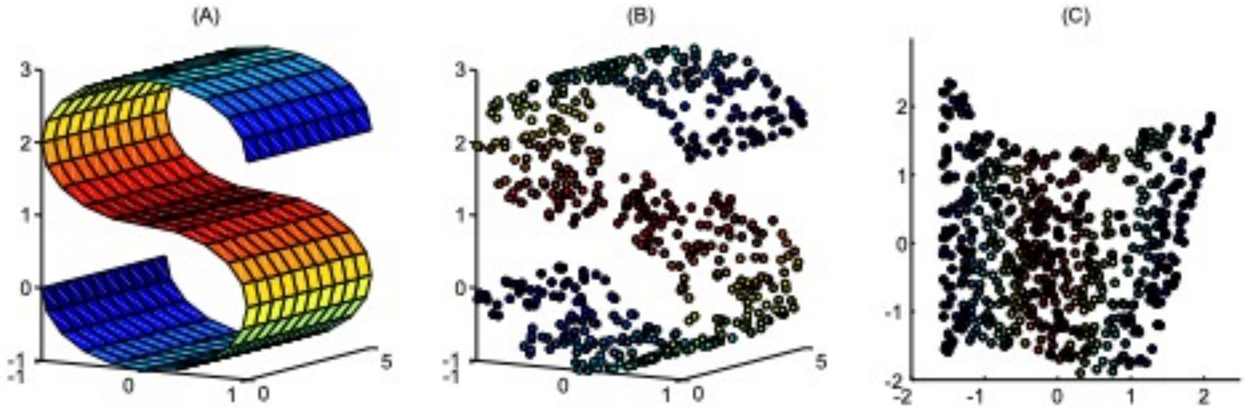
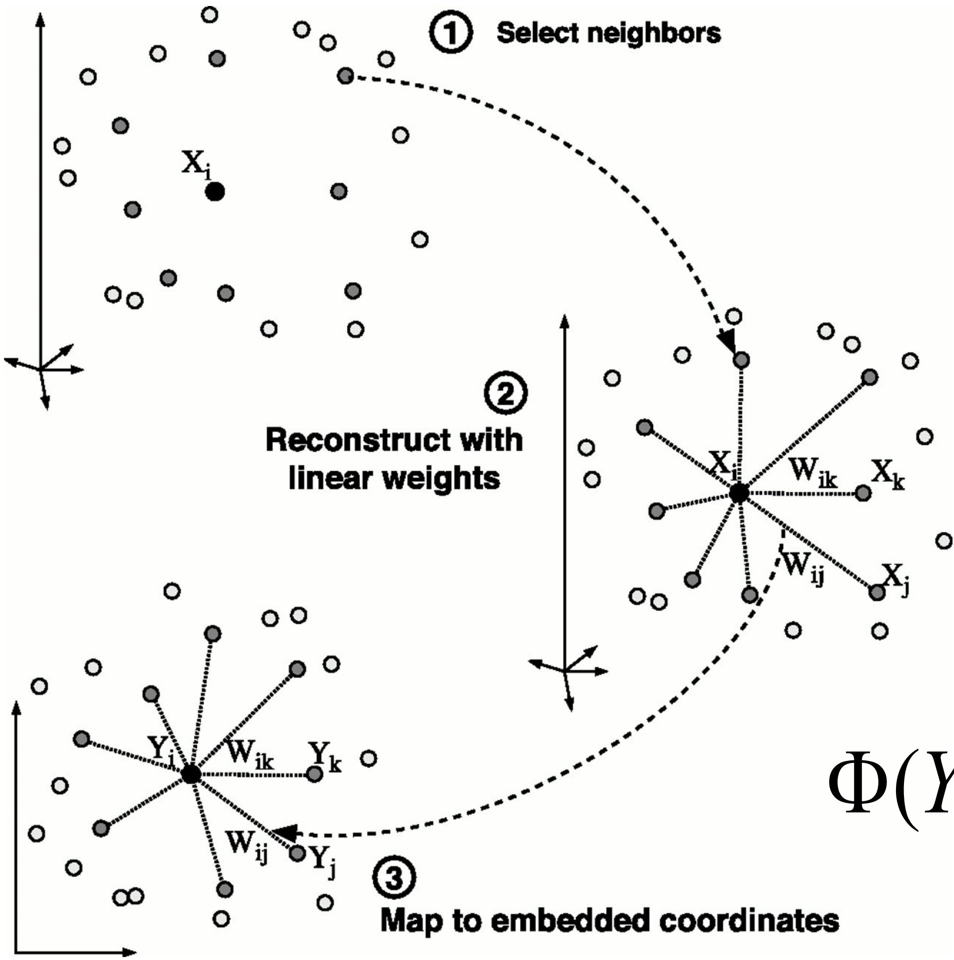


Figure 1: The problem of nonlinear dimensionality reduction, as illustrated for three dimensional data (B) sampled from a two dimensional manifold (A). An unsupervised learning algorithm must discover the global internal coordinates of the manifold without signals that explicitly indicate how the data should be embedded in two dimensions. The shading in (C) illustrates the neighborhood-preserving mapping discovered by LLE.

Reference: An Introduction to Locally Linear Embedding, https://cs.nyu.edu/~roweis/lle/papers/lleintro.pdf
http://statweb.stanford.edu/~tibs/sta306bfiles/isomap.pdf

Fit locally, Think Globally



From Nonlinear Dimensionality Reduction by Locally Linear Embedding

Sam T. Roweis and Lawrence K. Saul

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

Locally Linear Embedding (LLE)

- History: S. Roweis and L. Saul, Science, 2000
- Procedure

1. Identify the neighbors of each data point
2. Compute weights that best linearly reconstruct the point from its neighbors

$$\min_{\mathbf{w}} \sum_{i=1}^{N} \|\mathbf{x}_i - \sum_{j=1}^{k} w_{ij}\mathbf{x}_{N_i(j)}\|^2$$

3. Find the low-dimensional embedding vector which is best reconstructed by the weights determined in Step 2

$$\min_{Y} \sum_{i=1}^{N} \|\mathbf{y}_i - \sum_{j=1}^{k} w_{ij}\mathbf{y}_{N_i(j)}\|^2 \Longleftrightarrow \min_{Y} \ \mathrm{tr}(Y^\top Y L)$$

Centering Y with unit variance

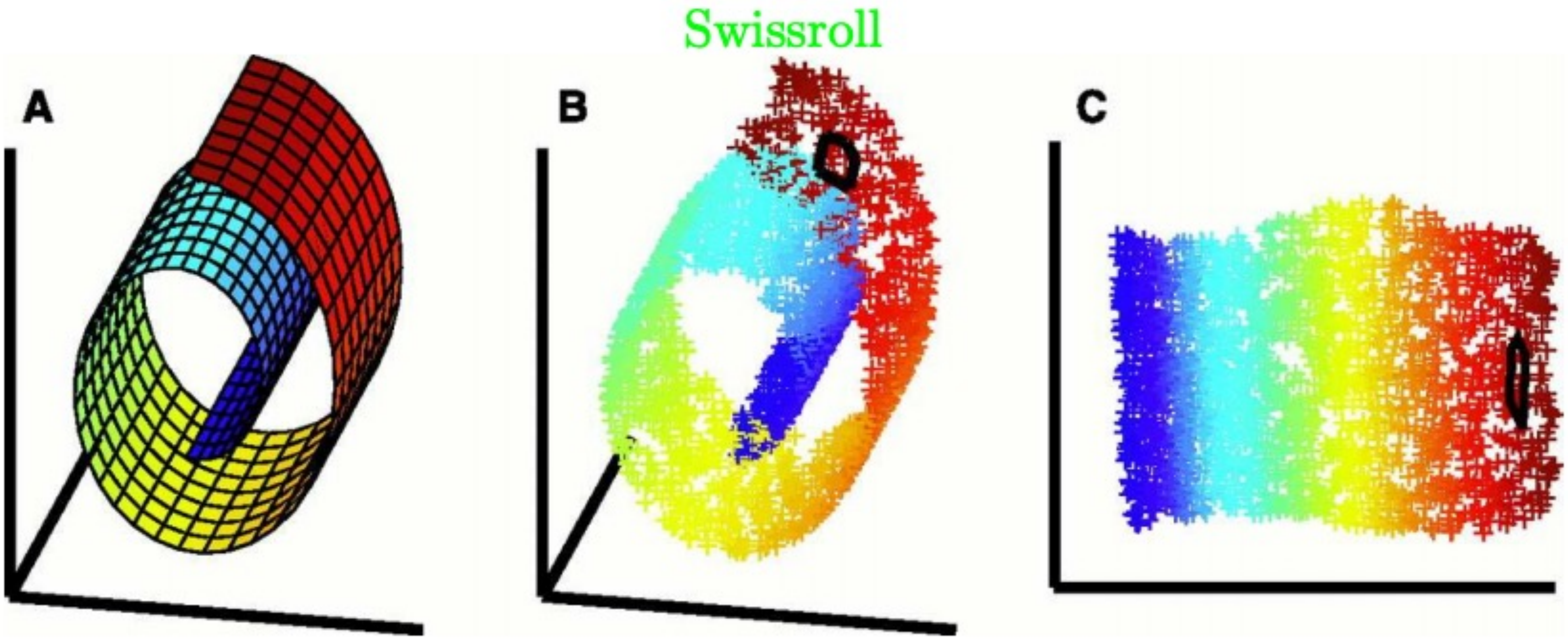
Solution:

- Minimize

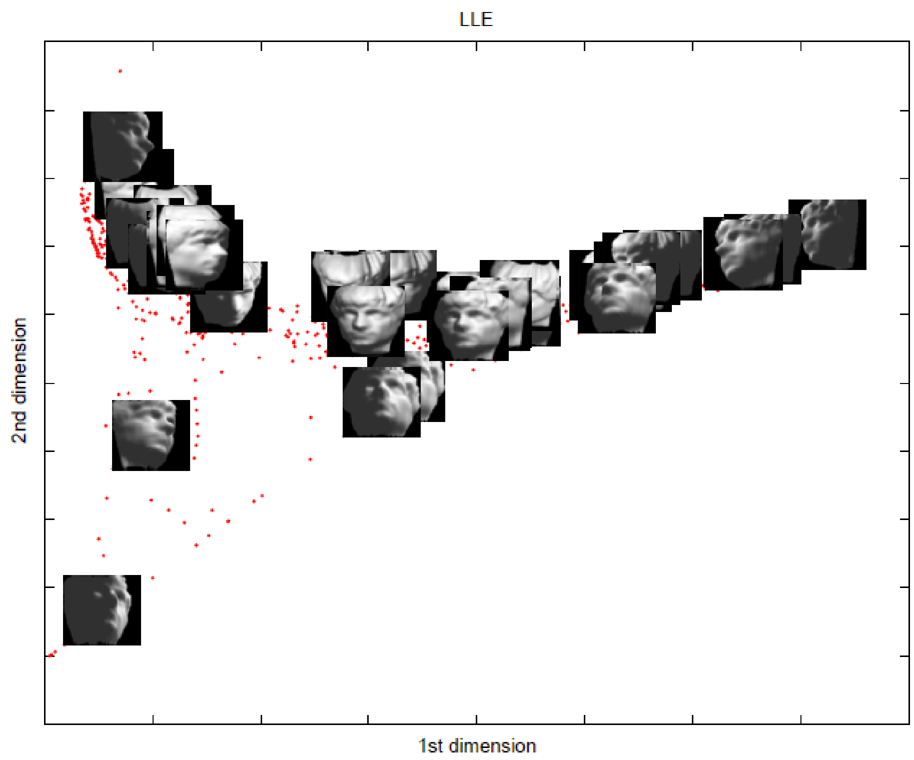
$$\mathrm{tr}[(Y - WY)^T(Y - WY)] = \mathrm{tr}[Y(I - W)^T(I - W)Y]$$

- W is $N \times N$; Y is $N \times d$, for some small $d < p$.
- solutions Y are the bottom eigenvectors of $M = (I - W)^T(I - W)$.
- They also assume $1^T Y = 0$ (i.e. they are centered at the origin), and $(1/N)Y^T Y = I$, the identity matrix in d dimensions. This means they discard the smallest eigenvector of M and keep the next d.

After Embedding



LLE Example



ISOMAP Idea?