Dimensionality Reduction

Unsupervised Learning
Dimensionality Reduction
Principal Component Analysis (PCA)
Applications

Machine Learning

- **Supervised learning:** generate a function from training data ((input, output) pairs).
  - regression: predict a continuous value as a function of the input
  - classification: predict the class label of the input object
- **Unsupervised learning:** fit a model to observations without a priori output.
  - clustering: natural groupings
  - identifying patterns in the data

Unsupervised Learning

- Input: unlabeled data samples $\{x(t)\}_{t=1..m}$
- Why study unlabeled data?
  - Collecting labeled data can be costly
  - Cluster first, label later
  - Changing pattern characteristics
  - Identify features that will be useful for categorization
  - Exploratory data analysis

Dimensionality Reduction

- Reducing the number of random variables under consideration.
- A technique for simplifying a high-dimensional data set by reducing its dimension for analysis.
- Projection of high-dimensional data to a low-dimensional space that preserves the "important" characteristics of the data.

Principal Component Analysis (PCA)

- An orthogonal linear transformation.
- Transforms into a new coordinate system
  - Greatest variance along the first axis
  - Second greatest variance along the second
  - etc
- Also known as:
  - Karhunen-Loève Transform (KLT)
  - Hotelling Transform

PCA (1)

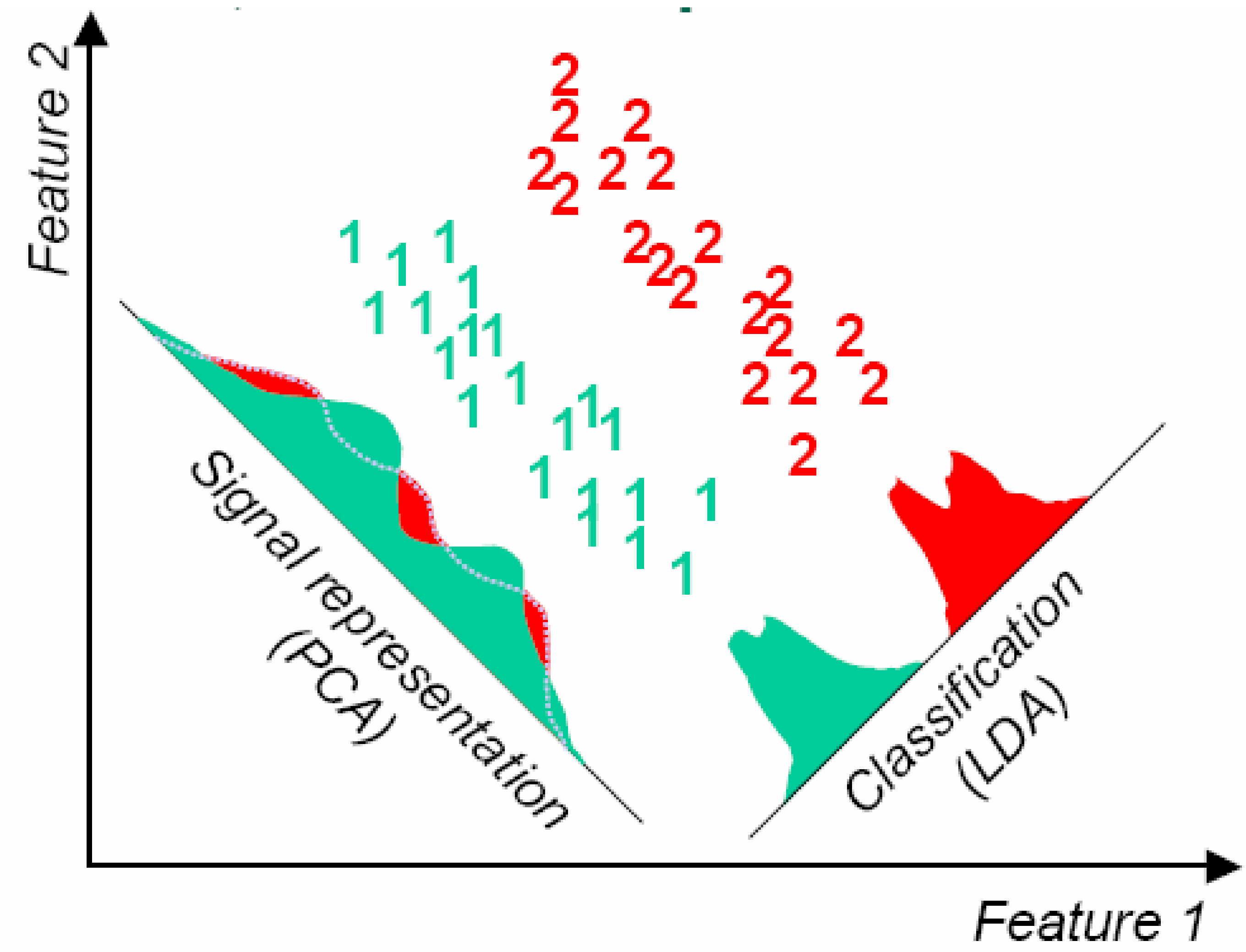
- Look for the unit vector w that maximizes the variance of the projected data items, $w^T x$:

$$w^* = \arg\max_{\|w\|=1} Var\left(w^T x\right) = \arg\max_{\|w\|=1} w^T \Sigma_x w$$

$$\Sigma_x = \sum_{t=1}^{m} \left(x(t) - \mu_x\right)\left(x(t) - \mu_x\right)^T$$

- Solution: w is the dominant eigenvector of the covariance matrix Σ_x.

PCA vs. LDA


Feature 2
Feature 1
Signal representation
(PCA)
Classification
(LDA)

PCA (2)

- What about projecting onto two vectors?
- Maximize:

$$Var\left(w_1^{\mathrm{T}}x\right) + Var\left(w_2^{\mathrm{T}}x\right)$$

$$\text{s.t. } \|w_1\| = \|w_2\| = 1 \text{ and } w_1^{\mathrm{T}}w = 0$$

- w_1 and w_2 are the two dominant eigenvectors of Σ_x.
- This idea generalizes to any dimension k.

PCA (3)

- Let X denote the $n \times m$ data matrix (each column is a centered vector $x(t)-\mu_x \in R^n$)
- Definition: the principle directions (axes) of $\{x(t)\}_{t=1..T}$ are the eigenvectors of the covariance matrix XX^T.
- Let $W = \{w_1,\ldots,w_k\}$ be the k leading principle axes, then the projection W^TX maximizes:

$$J(W) = \sum_{i=1}^{k} Var\left(w_i^{\mathrm{T}} x\right)$$

$$\text{s.t. } \|w_i\| = 1 \text{ and } w_i^{\mathrm{T}} w_j = 0$$

PCA (4)

- Let X denote the centered $n \times m$ data matrix as before.

- Consider the Singular Value Decomposition of X:

$$X = W\Sigma V^{\mathrm{T}}$$

- The PCA transform projects X down into the reduced subspace spanned only by the first L singular vectors, W_L:

$$Y = W_L^{\mathrm{T}} X = \Sigma_L V_L^{\mathrm{T}}$$

Computing the Principle Axes

- Let A = XX^T
- Bad idea: compute all eigenvectors of A and keep the leading k.
- Power method:
  - Compute (v,λ), the leading eigenvector of A
  - Update

$$B = A - \lambda v v^{\mathrm{T}}$$

  - Repeat the above with A = B, until we have k vectors.

Power method (continued)

- If v_1 is the leading eigenvector of A, then it is an eigenvector of B (with eigenvalue 0):

$$\begin{aligned} B v_1 &= \left(A - \lambda_1 v_1 v_1^{\mathrm{T}}\right) v_1 \\ &= A v_1 - \lambda_1 v_1 \left(v_1^{\mathrm{T}} v_1\right) \; = \; \lambda_1 v_1 - \lambda_1 v_1 \end{aligned}$$

- Any other eigenvector v_k of A is also an eignevector of B:

$$\begin{aligned} B v_k &= \left(A - \lambda_1 v_1 v_1^{\mathrm{T}}\right) v_k \\ &= A v_k - \lambda_1 v_1 \left(v_1^{\mathrm{T}} v_k\right) \; = \; \lambda_k v_k \end{aligned}$$

Alternative method

- Let $X_{n \times m}$, where m << n. The matrix XX^T is very large, but X^TX is much smaller!
- Idea: compute eigenvectors of X^TX instead!
- Let (v,λ) be an eigenpair of X^TX, then (Xv,λ) is an eigenpair of XX^T.
- Proof:

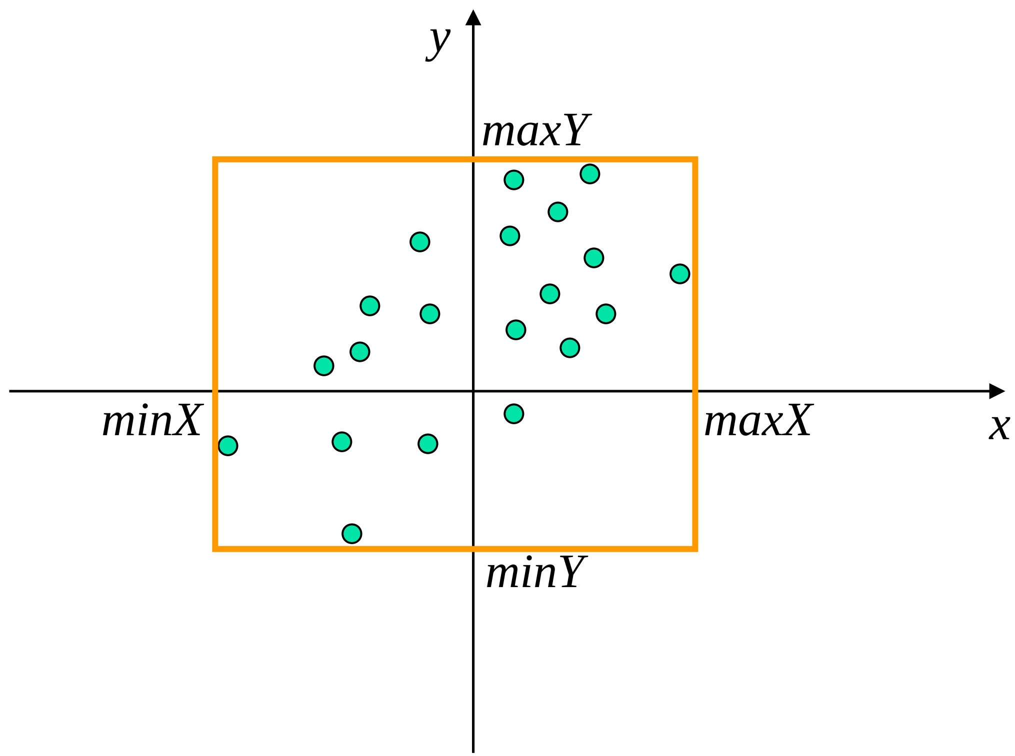
$$X^{\mathrm{T}} X v = \lambda v$$

$$X X^{\mathrm{T}} X v = X \lambda v$$

$$X X^{\mathrm{T}} (X v) = \lambda (X v)$$

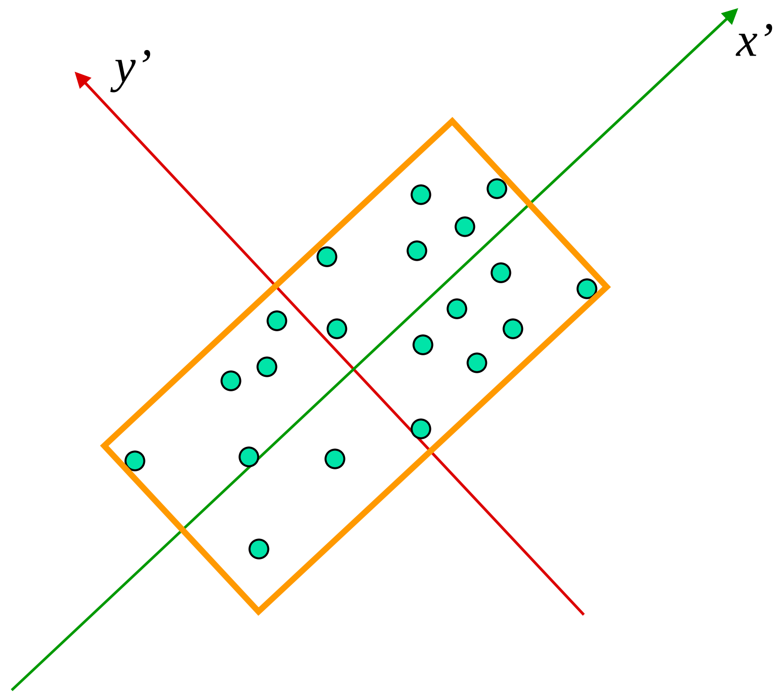
Applications in Comp. Graphics

- Bounding box computation



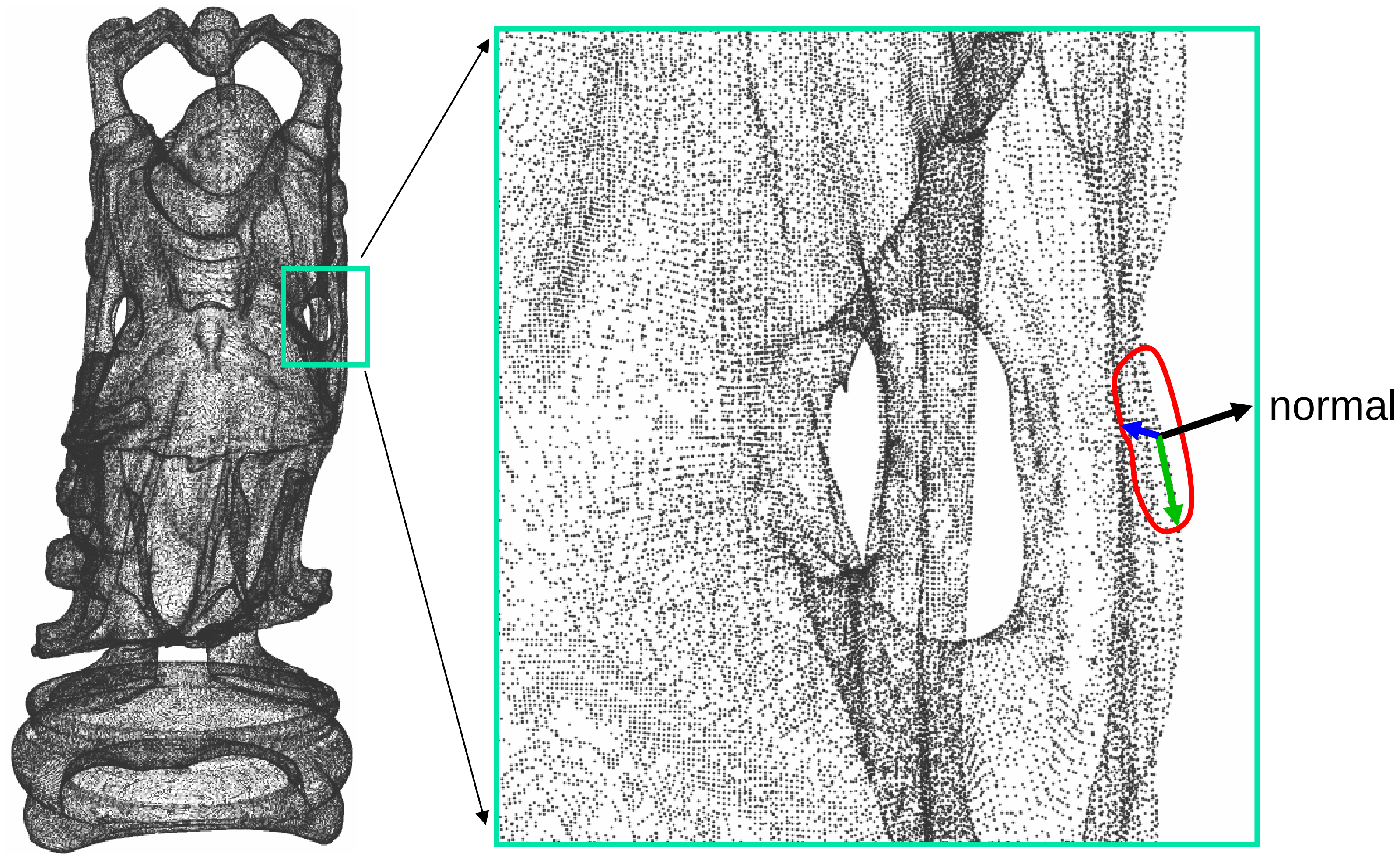
Applications in Comp. Graphics

- Bounding box computation



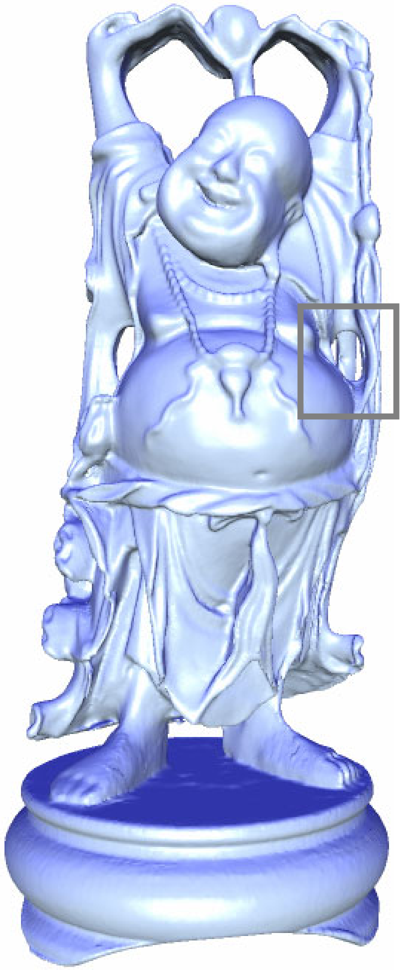
Applications in Comp. Graphics

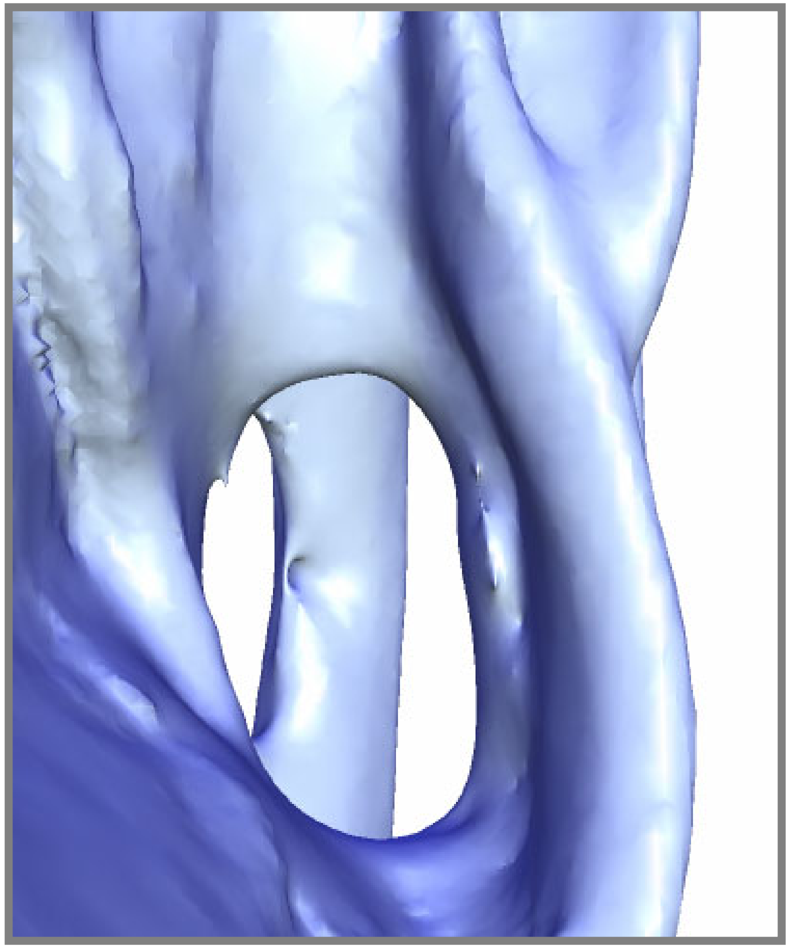
- Normal Estimation in point clouds



Applications in Comp. Graphics

- Normal Estimation in point clouds





Applications in Comp. Graphics

- Morphable face models
  - http://gravis.cs.unibas.ch/Sigg99.html