Dimensionality reduction

© Victor Kitov

v.v.kitov@yandex.ru

Summer school on Machine Learning in High Energy Physics

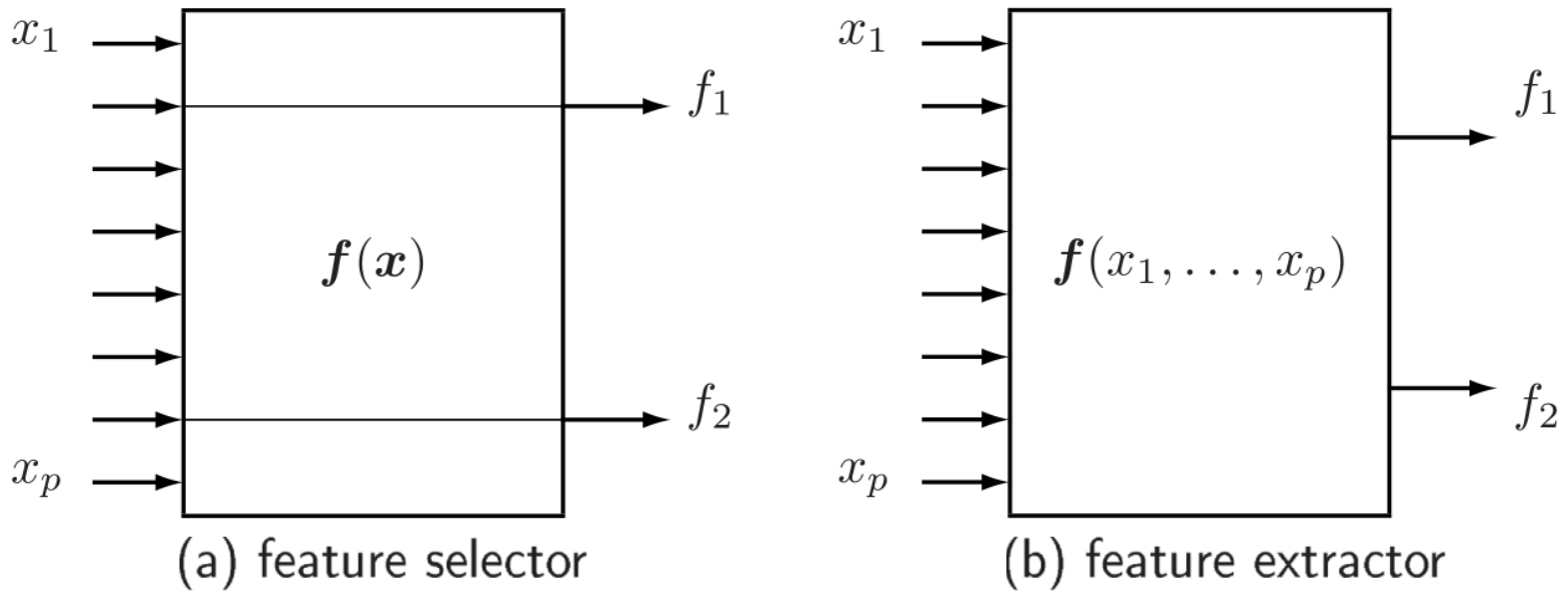
in partnership with

August 2015

Table of Contents

Definition

Feature selection / Feature extraction



(a) feature selector (b) feature extractor

Dimensionality reduction: find transformation of original data to low dimensional space which extracts most relevant information for machine learning task.

Applications

Applications:

- visualization in 2D or 3D
- reduce operational costs (less memory, disc, CPU usage on data transfer)
- remove multi-collinearity to improve performance of machine-learning models

Categorization

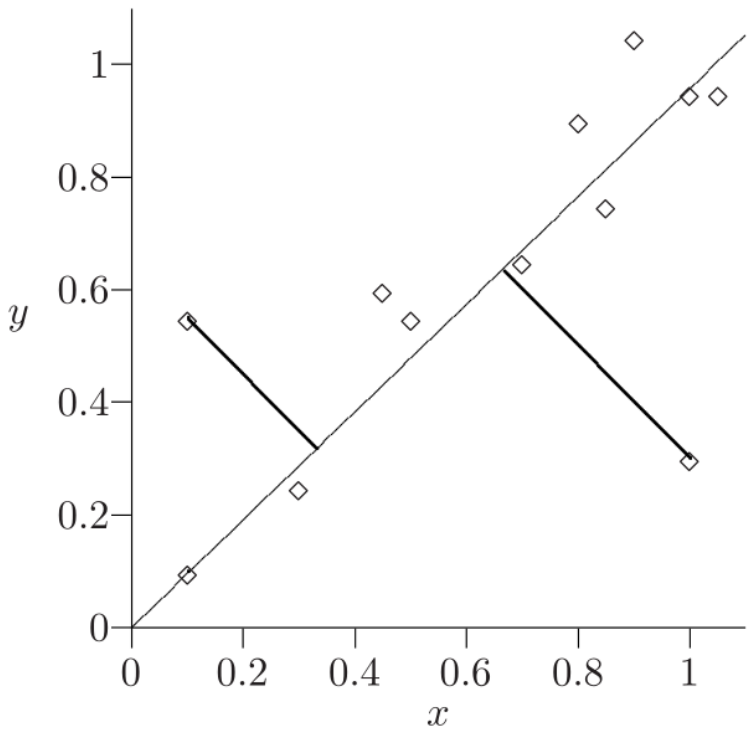
Supervision in dimensionality reduction:

- supervised (such as LDA)
- unsupervied

Mapping to reduced space:

- linear
- non-linear

Example: line of best fit



- In PCA sum of squared of perpendicular distances to line is minimized.
- Not invariant to scale

Table of Contents

Definition

Linear transformation of data:

$$\xi_i = \sum_{j=1}^{D} a_{ij} x_j, \quad i = 1, 2, ...\tilde{D}.$$

Three equivalent ways to derive PCA:

- Find orthogonal transform A yielding new variables ξ_i having stationary values for their variance
- Find orthogonal transform, yielding uncorrelated ξ_j
- Find line of best fit, plane of best fit, etc. where fit is the sum of squares of perpendicular distances.

Method works for $\mathbb{E}x = 0$. So sample observations should be shifted to guarantee: $\sum_{n=1}^{N} x_{nd} = 0, \ d = 1, 2, ..D$.

Covariance matrix properties

$\Sigma = cov[x] \in \mathbb{R}^{D \times D}$ is symmetric positive semidefinite matrix

- has $\lambda_1, \lambda_2, ...\lambda_D$ eigenvalues, satisfying: $\lambda_i \in \mathbb{R}$, $\lambda_i \geq 0$.
- if eigenvalues are unique, corresponding eigenvectors are also unique
- always exists a set of orthogonal eigenvectors $z_1, z_2, ...z_D$: $\Sigma z_i = \lambda_i z_i$.

later we will assume that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_D \geq 0$.

Derivation: 1st component

Consider first component:

$$\xi_1 = \sum_{j=1}^{D} a_{1j} x_j$$

Optimization problem:

$$\begin{cases} \mathrm{Var}\xi_1 \to \max_a \\ |a_1|^2 = a_1^T a_1 = 1 \end{cases}$$

Variance is equal:

$$\begin{aligned} Var[\xi_1] &= \mathrm{E}[\xi_1^2] - (\mathrm{E}\xi_1)^2 = \mathrm{E}[a_1^T x x^T a_1] - \mathrm{E}[a_1^T x]\mathrm{E}[x^T a_1] \\ &= a_1^T \left(\mathrm{E}[xx^T] - \mathrm{E}[x]\mathrm{E}[x^T] \right) a_1 = a_1^T \Sigma a_1 \end{aligned}$$

Derivation: 1st component

Optimization problem is equivalent to finding unconditional stationary value of

$$L(a_1, \nu) = a_1^T \Sigma a_1 - \nu(a_1^T a_1 - 1) \rightarrow extr_{a_1, \nu}$$

$$\frac{\partial L}{\partial a_1} = 0 : \; 2\Sigma a_1 - \nu a_1 = 0$$

a_1 is selected from a set of eigenvectors of A. Since

$$Var[\xi_1] = a_1^T \Sigma a_1 = \lambda_i a_1^T a_1$$

a_1 is the eigenvector, corresponding to largest eigenvalue λ_i. Eigenvector is not unique if λ_{max} is a repeated root of characteristic equation: $|\Sigma - \nu I|$=0.

Derivation: 2nd component

$$\xi_2 = a_2^T x$$

$$\begin{cases} \mathrm{Var}[\xi_2] = a_2^T \Sigma a_2 \to \max_{a_2} \\ a_2^T a_2 = |a_2|^2 = 1 \\ cov[\xi_1, \xi_2] = a_2^T \Sigma a_1 = \lambda_1 a_2^T a_1 = 0 \end{cases}$$

Lagrangian (assuming $\lambda_1 > 0$)

$$L(a_2, \nu, \eta) = a_2^T \Sigma a_2 - \nu(a_2^T a_2 - 1) - \eta a_2^T a_1 \to extr_{a_2, \nu, \eta}$$

$$\frac{\partial L}{\partial a_2} = 0 : \; 2\Sigma a_2 - 2\nu a_2 - \eta a_1 = 0$$

$$a_1^T \frac{\partial L}{\partial a_2} = 2 a_1^T \Sigma a_2 - \eta = 0$$

Derivation: 2nd component

Since $a_1^T \Sigma a_2 = a_2^T \Sigma a_1 = 0$, we obtain $\eta = 0$. Then we have that:

$$\Sigma a_2 = \nu a_2$$

so a_2 is eigenvector of Σ, and since we maximize

$$\mathrm{Var}[\xi_2] = a_2^T \Sigma a_2 = \lambda_i a_2^T a_2$$

this should be eigenvector, corresponding to second largest eigenvalue λ_2.

Derivation: k-th component

$$\xi_k = a_k^T x$$

$$\begin{cases} Var[\xi_k] = a_k^T \Sigma a_k \to \max_{a_k} \\ a_k^T a_k = |a_k|^2 = 1 \\ cov[\xi_k, \xi_j] = a_k^T \Sigma a_j = \lambda_j a_k^T a_j = 0, \quad j = 1, 2, ...k-1. \end{cases}$$

Lagrangian (assuming $\lambda_j > 0$, $j = 1, 2, ...k-1$)

$$L(a_k, \nu, \eta) = a_k^T \Sigma a_k - \nu(a_k^T a_k - 1) - \sum_{i=1}^{k-1} \eta_i a_k^T a_i \to extr_{a_2, \nu, \eta}$$

$$\frac{\partial L}{\partial a_k} = 0 : \ 2\Sigma a_k - 2\nu a_k - \sum_{i=1}^{k-1} \eta_i a_i = 0$$

$$\forall j = 1, 2, ...k-1 : \ a_j^T \frac{\partial L}{\partial a_2} = 2a_j^T \Sigma a_k - \eta_j = 0$$

Derivation: k-th component

Since $a_1^T \Sigma a_k = a_k^T \Sigma a_1 = 0$, we obtain $\eta_j = 0$ for all $j = 1, 2, ...k-1$, so

$$\Sigma a_k = \nu a_k$$

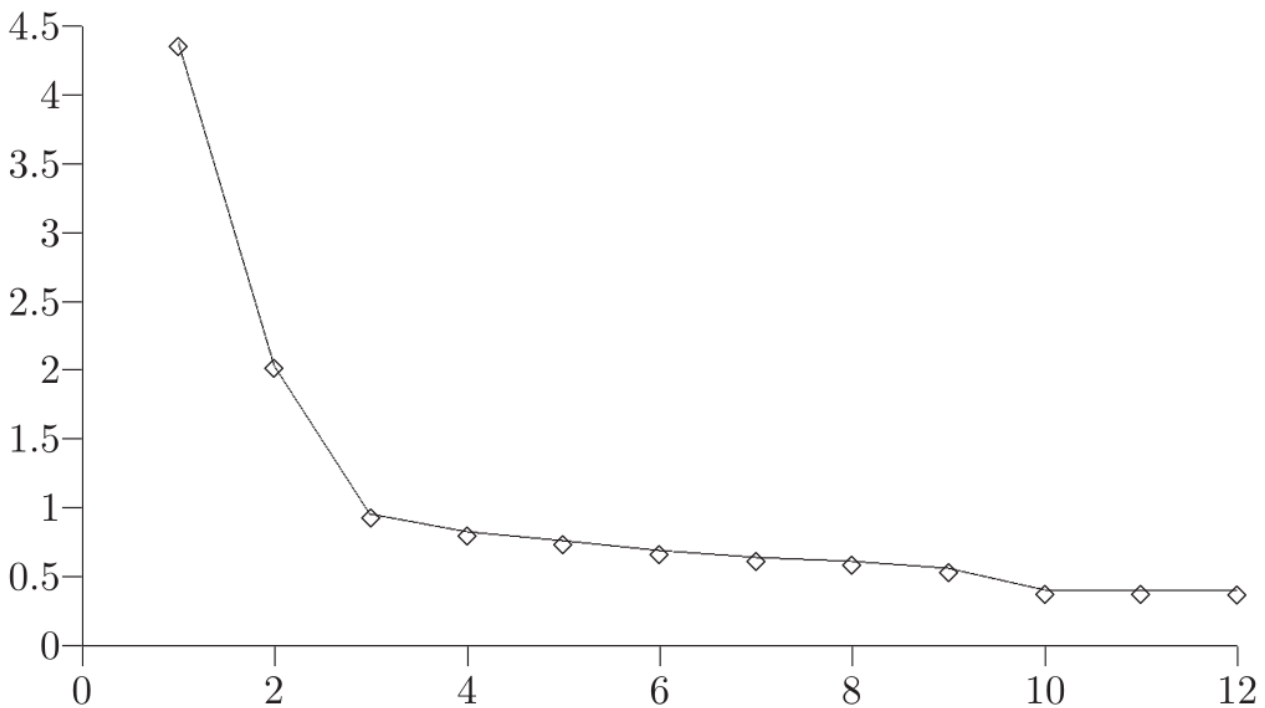
a_k is then the eigenvector.
Variance of ξ_i is

$$Var[\xi_k] = a_k^T \Sigma a_k = \lambda_i a_k^T a_k = \lambda_i$$

so a_k should be the eigenvector corresponding to the k-th largest eigenvalue λ_k.

Number of components

- Data visualization: 2 or 3 components.
- Take most significant components until their variance falls sharply down:



Number of components

Remind that $A = [a_1|a_2|...|a_D]$, $A^T A = I$, $\xi = A^T x$.
Denote $S_k = [\xi_1, \xi_2, ...\xi_k, 0, 0, ..., 0] \in \mathbb{R}^D$

$$\mathbb{E}[\|S_k\|^2] = \mathbb{E}[\xi_1^2 + \xi_2^2 + ... + \xi_k^2] = \sum_{i=1}^{k} \text{var}\, \xi_i = \sum_{i=1}^{k} \lambda_i$$

$$\begin{aligned} \mathbb{E}[\|S_D\|^2] &= \mathbb{E}[\xi^T \xi] = \\ &= \mathbb{E} x^T A A^T x = \mathbb{E}\left[x^T x\right] = \mathbb{E}[\|x\|^2] \end{aligned}$$

Select such k^* that

$$\frac{\mathbb{E}[\|S_k\|^2]}{\mathbb{E}[\|x\|^2]} = \frac{\mathbb{E}[\|S_k\|^2]}{\mathbb{E}[\|S_D\|^2]} = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{D} \lambda_i} > threshold$$

We may select k^* to account for 90%, 95% or 99% of total variance.

Transformation $\xi \rightleftarrows x$

Dependence between original and transformed features:

$\xi = A^T(x-\mu)$, $x = A\xi+\mu$, μ is the mean of original non-shifted data

Taking first r components - $A_r = [a_1|a_2|...|a_r]$, we get the image of the reduced transformation:

$$\xi_r = A_r^T(x-\mu)$$

ξ_r will correspond to

$$x_r = A\begin{pmatrix} \xi_r \\ 0 \end{pmatrix} + \mu = A_r\xi_r + \mu$$

$$x_r = A_rA_r^T(x-\mu) + \mu$$

$A_rA_r^T$ is projection matrix with rank r.

Properties of PCA

- Covariance matrix replaced with sample-covariance.
- Depends on scaling (unit deviation transforms covariance matrix to correlation matrix).
- Does not require distribution assumptions
- Eigenvectors may be obtained from SVD decomposition of X.

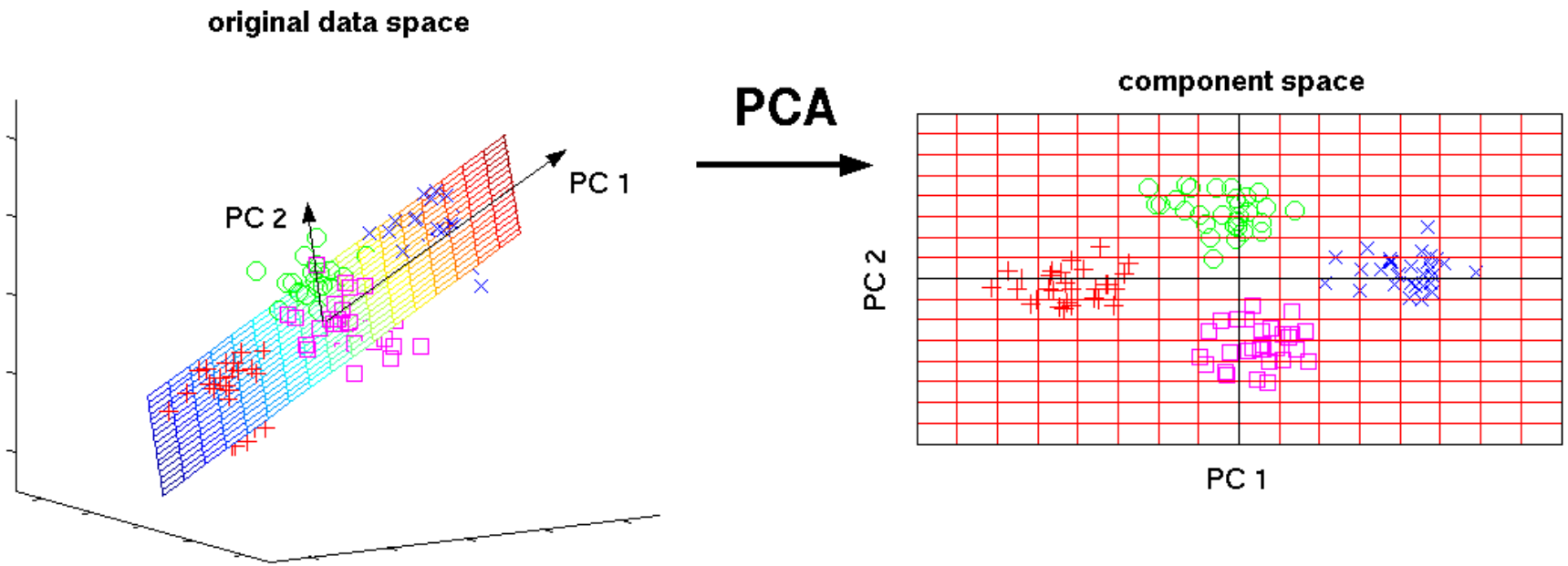
Example

Faces database:

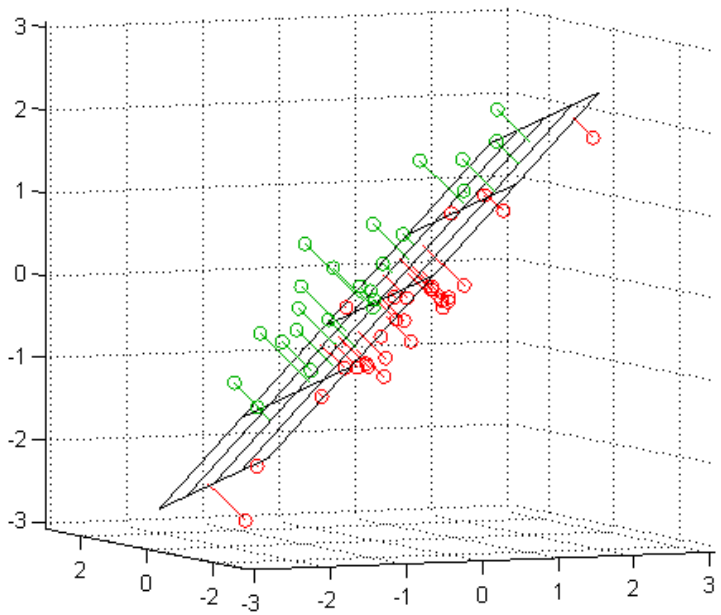
Eigenfaces

PCA for visualization

Uncorrelatedness does not imply independence.



Best hyperplane fit



Subspace L_k or rank k best fits points $x_1, x_2, ...x_D$ if sum of squared distances of these points to this plane is maximized over all planes of rank k.

Best hyperplane fit

For point x_i denote p_i the projection on plane L_k and h_i - orthogonal component. Then $\|x_i\|^2 = \|p_i\|^2 + \|h_i\|^2$.
For set of points:

$$\sum_i \|x_i\|^2 = \sum_i \|p_i\|^2 + \sum_i \|h_i\|^2$$

Since sum of squares is constant, minimization of $\sum_i \|h_i\|^2$ is equivalent to maximization of $\sum_i \|p_i\|^2$.

Another view on PCA directions

k-th step optimization problem for $\xi_k = a_k^T x$:

$$\begin{cases} Var[\xi_k] = a_k^T \Sigma a_k \to \max_{a_k} \\ a_k^T a_k = |a_k|^2 = 1 \\ cov[\xi_k, \xi_j] = a_k^T \Sigma a_j = \lambda_j a_k^T a_j = 0, \quad j = 1, 2, ...k-1. \end{cases}$$

can be equivalently represented as:

$$\begin{cases} \|Xa_k\| \to \max_{a_k} \\ \|a_k\| = 1 \\ a_k \perp a_1, a_k \perp a_2, ...a_k \perp a_{k-1} \text{ if } k \geq 2 \end{cases} \tag{1}$$

since maximization of $\|Xa_k\|$ is equivalent to maximization of $\frac{1}{N}\|Xa_k\|^2 = \frac{1}{N}(Xa_k)^T(Xa_k) = \frac{1}{N}a_k^T X^T X a_k = a_k^T \Sigma a_k$.

Property of PCA

Theorem 1

For $1 \leq k \leq r$ let L_r be the subspace spanned by $a_1, a_2, ...a_r$. Then for each k L_k is the best-fit k-dimensional subspace for X.

Proof: use induction. For $r = 1$ the statement is true by definition since projection maximization is equivalent to distance minimization.
Suppose theorem holds for $r - 1$. Let L_r be the plane of best-fit of dimension with $\dim L = r$. We can always choose a orthonormal basis of L_r $b_1, b_2, ...b_r$ so that

$$\begin{cases} \|b_r\| = 1 \\ b_r \perp a_1, b_r \perp a_2, ...b_r \perp a_{r-1} \end{cases}$$

by setting b_r perpendicular to projections of $a_1, a_2, ...a_{r-1}$ on L_r.

Property of PCA

Consider the sum of squared projections:

$$\|Xb_1\|^2 + \|Xb_2\|^2 + ... + \|Xb_{r-1}\|^2 + \|Xb_r\|^2$$

By induction proposition:

$$\|Xb_1\|^2+\|Xb_2\|^2+...+\|Xb_{r-1}\|^2 \leq \|Xa_1\|^2+\|Xa_2\|^2+...+\|Xa_{r-1}\|^2$$

and

$$\|Xb_r\| \leq \|Xa_r\|$$

since b_r by (28) satisfies constraints of optimization problem (27) and a_r is its optimal solution.

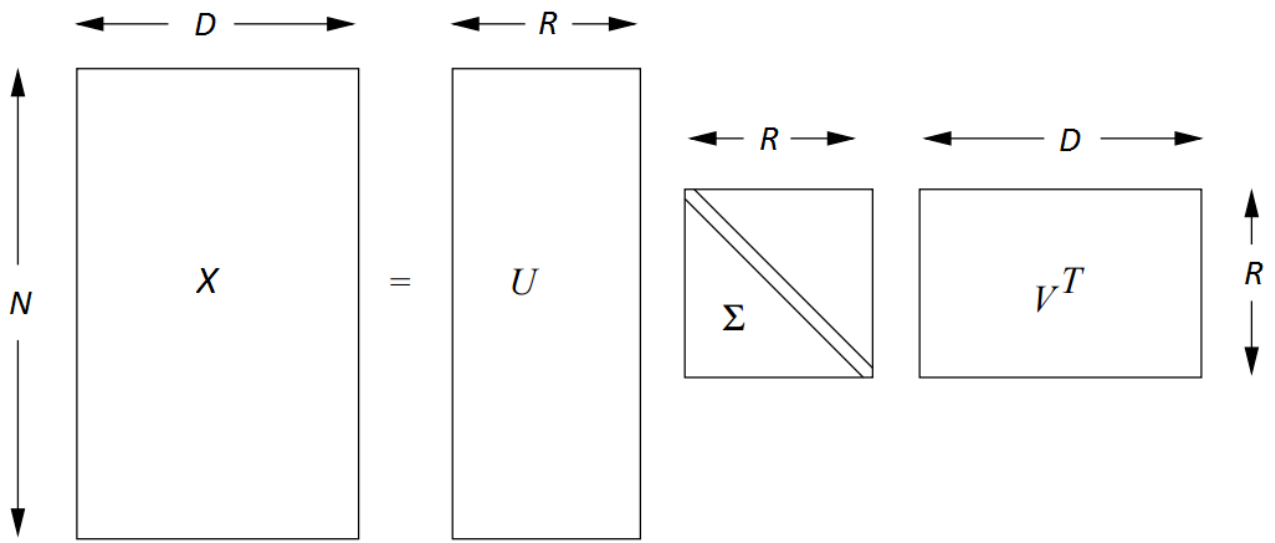
Table of Contents

SVD decomosition

Every matrix $X \in \mathbb{R}^{NxD}$ of rank R can be decomposed into the product of three matrices:

$$X = U\Sigma V^T$$

where $U \in \mathbb{R}^{NxD}$, $\Sigma \in \mathbb{R}^{RxR}$, $V \in \mathbb{R}^{RxD}$, and $\Sigma = diag\{\sigma_1, \sigma_2, ...\sigma_R\}$, $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_R \geq 0$, $U^T U = I$, $V^T V = I$. I denotes identity matrix.

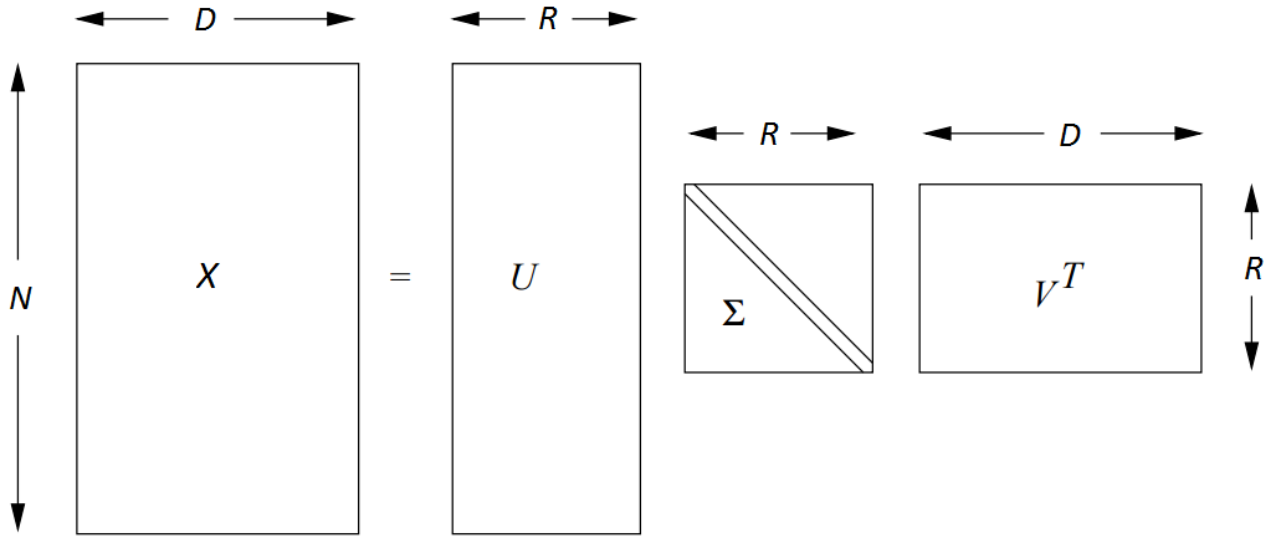


Applications of SVD

For square matrix X:

- U, V^T represent rotations, Σ represents scaling (with projection and reflection),
  every square matrix may be represented as superposition of rotation, scaling and another rotation.
- For full rank X:
$$X^{-1} = V\Sigma^{-1}U^T,$$
  since $XX^{-1} = U\Sigma V^T V\Sigma^{-1}U^T = I$.
- For not-full-rank X: generalized inverse matrix equals
$$X^{+} = V\Sigma^{+}U^T$$
  (A^{+} is such matrix that $AA^{+}A = A$)

Interpretation of SVD



For X_{ij} let i denote objects and j denote properties.

- U represents standardized coordinates of concepts
- V^T represents standardized coordinates of concepts representations
- Σ shows the magnitude of standardized concept presence in X.

Example

	The lord of the rings	Harry Potter	Avatar	Titanic	Love story	A walk to remember
Andrew	4	5	5	0	0	0
John	4	4	5	0	0	0
Matthew	5	5	4	0	0	0
Anna	0	0	0	5	5	5
Maria	0	0	0	5	5	4
Jessika	0	0	0	4	5	4

Example

$$U = \begin{pmatrix} 0. & 0.6 & -0.3 & 0. & 0. & -0.8 \\ 0. & 0.5 & -0.5 & 0. & 0. & 0.6 \\ 0. & 0.6 & 0.8 & 0. & 0. & 0.2 \\ 0.6 & 0. & 0. & -0.8 & -0.2 & 0. \\ 0.6 & 0. & 0. & 0.2 & 0.8 & 0. \\ 0.5 & 0. & 0. & 0.6 & -0.6 & 0. \end{pmatrix}$$

$$\Sigma = \text{diag}\{\begin{pmatrix} 14. & 13.7 & 1.2 & 0.6 & 0.6 & 0.5 \end{pmatrix}\}$$

$$V^T = \begin{pmatrix} 0. & 0. & 0. & 0.6 & 0.6 & 0.5 \\ 0.5 & 0.6 & 0.6 & 0. & 0. & 0. \\ 0.5 & 0.3 & -0.8 & 0. & 0. & 0. \\ 0. & 0. & 0. & -0.2 & 0.8 & -0.6 \\ -0. & -0. & -0. & 0.8 & -0.2 & -0.6 \\ 0.6 & -0.8 & 0.2 & 0. & 0. & 0. \end{pmatrix}$$

Example (excluded insignificant concepts)

$$U_2 = \begin{pmatrix} 0. & 0.6 \\ 0. & 0.5 \\ 0. & 0.6 \\ 0.6 & 0. \\ 0.6 & 0. \\ 0.5 & 0. \end{pmatrix}$$

$$\Sigma_2 = \text{diag}\{\begin{pmatrix} 14. & 13.7 \end{pmatrix}\}$$

$$V_2^T = \begin{pmatrix} 0. & 0. & 0. & 0.6 & 0.6 & 0.5 \\ 0.5 & 0.6 & 0.6 & 0. & 0. & 0. \end{pmatrix}$$

Concepts may be

- patterns among movies (along j) - fantasy/romance
- patterns among people (along i) - boys/girls

Dimensionality reduction case: patterns along j axis.

Applications

- Example: new movie rating by new person

$$x = \begin{pmatrix} 5 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

- **Dimensionality reduction:** map x into concept space:

$$y = V_2^T x = \begin{pmatrix} 0 & 2.7 \end{pmatrix}$$

- **Recommendation system:** map y back to original movies space:

$$\widehat{x} = yV_2^T = \begin{pmatrix} 1.5 & 1.6 & 1.6 & 0 & 0 & 0 \end{pmatrix}$$

Fronebius norm

- Fronebius norm of matrix X is $\|X\|_F \stackrel{df}{=} \sqrt{\sum_{n=1}^{N}\sum_{d=1}^{D} x_{nd}^2}$
- Using properties $||X||_F = \text{tr}\, XX^T$ and $\text{tr}\, AB = \text{tr}\, BA$, we obtain:

$$\begin{aligned} \|X\|_F &= \text{tr}[U\Sigma V^T V\Sigma U^T] = \text{tr}[U\Sigma^2 U^T] = \\ &= \text{tr}[\Sigma^2 U^T U] = \text{tr}[\Sigma^2] = \sum_{r=1}^{R} \sigma_r^2 \end{aligned} \tag{2}$$

Matrix approximation

Consider approximation $X_k = U\Sigma_k V^T$, where $\Sigma_k = \text{diag}\{\sigma_1, \sigma_2, ...\sigma_k, 0, 0, ..., 0\} \in \mathbb{R}^{RxR}$.

Theorem 2

X_k is the best approximation of X retaining k concepts.

Proof: consider matrix $Y_k = U\Sigma' V^T$, where Σ' is equal to Σ except some $R - k$ elements set to zero: $\sigma'_{i_1} = \sigma'_{i_2} = ... = \sigma'_{i_{R-k}} = 0$. Then, using (2)

$$\|X - Y_k\|_F = \left\|U(\Sigma - \Sigma')V^T\right\|_F = \sum_{p=1}^{R-k} \sigma_{i_p}^2 \leq \sum_{p=1}^{R-k} \sigma_p^2 = \|X - X_k\|_F$$

since $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_R \geq 0$.

Matrix approximation

How many components to retain?

General case: Since

$$\|X - X_k\|_F = \left\|U(\Sigma - \Sigma_k)V^T\right\|_F = \sum_{i=k+1}^{R} \sigma_i^2$$

a reasonable choice is k^* such that

$$\frac{\|X - X_{k^*}\|_F}{\|X\|_F} = \frac{\sum_{i=k^*+1}^{R} \sigma_i^2}{\sum_{i=1}^{R} \sigma_i^2} < threshold$$

Visualization: 2 or 3 components.

Theorem 3

For any matrix Y_k *with* rank $Y_k = k$: $\|X - X_k\|_F \leq \|X - Y_k\|_F$

Finding U and V

- **Finding V**
  $X^TX = (U\Sigma V^T)^T U\Sigma V^T = (V\Sigma U^T)U\Sigma V^T = V\Sigma^2 V^T$. It follows that
  $$X^TXV = V\Sigma^2 V^T V = V\Sigma^2$$
  So V consists of eigenvectors of X^TX with corresponding eignvalues $\sigma_1^2, \sigma_2^2, ...\sigma_R^2$.

- **Finding U:**
  $XX^T = U\Sigma V^T (U\Sigma V^T)^T = U\Sigma V^T V\Sigma U^T = U\Sigma^2 U^T$. So
  $$XX^TU = U\Sigma^2 U^T U = U\Sigma^2.$$
  So U consists of eigenvectors of XX^T with corresponding eigenvalues $\sigma_1^2, \sigma_2^2, ...\sigma_R^2$.

Comments

- Denote the average $\bar{X} \in \mathbb{R}^D : \bar{X}_j = \sum_{i=1}^{N} x_{ij}$
- Denote the n-th row of X be $X_n \in \mathbb{R}^D : X_{nj} = x_{nj}$
- For centered X sample covariance matrix $\widehat{\Sigma}$ equals:

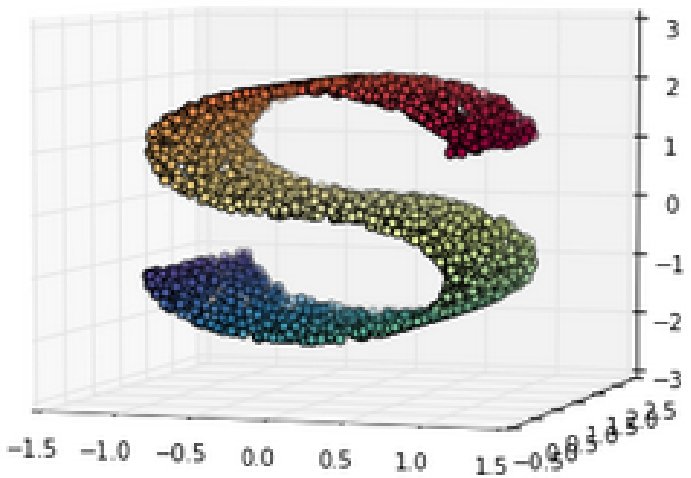
$$\begin{aligned} \widehat{\Sigma} &= \frac{1}{N}\sum_{n=1}^{N}(X_n - \bar{X})(X_n - \bar{X})^T = \frac{1}{N}\sum_{n=1}^{N} X_n X_n^T \\ &= \frac{1}{N} X^T X \end{aligned}$$

- V **consists of principal components** since
  - V consists of eigenvectors of $X^T X$,
  - principal components are eignevectors of $\widehat{\Sigma}$ and
  - $\widehat{\Sigma} \propto X^T X$.

Table of Contents

Non-linear dimensionality reduction

- Based on assumption that original data $x \in \mathbb{R}^D$ is distributed compactly on non-linear surface with dimensionality $d < D$.
- Let $y \in \mathbb{R}^d$ denote the coordinates of x on the surface.
- d is usually unknown.
- Sample dataset:



- Linear dimensionality reduction techniques will fail here.

Categorization

Non-linear approaches of dimensionality reduction:

- preserving global properties
  - kernel PCA, autoencoders, MDS, ISOMAP, diffusion maps, MVU
- preserving local properties
  - LLE, LTSA
- global alignment of local linear models (not considered here)

4 Non-linear dimensionality reduction
- Global methods
- Local methods

Multi-dimensional scaling

Multi-dimensional scaling

Map $x \to y$ preserving distances as much as possible.

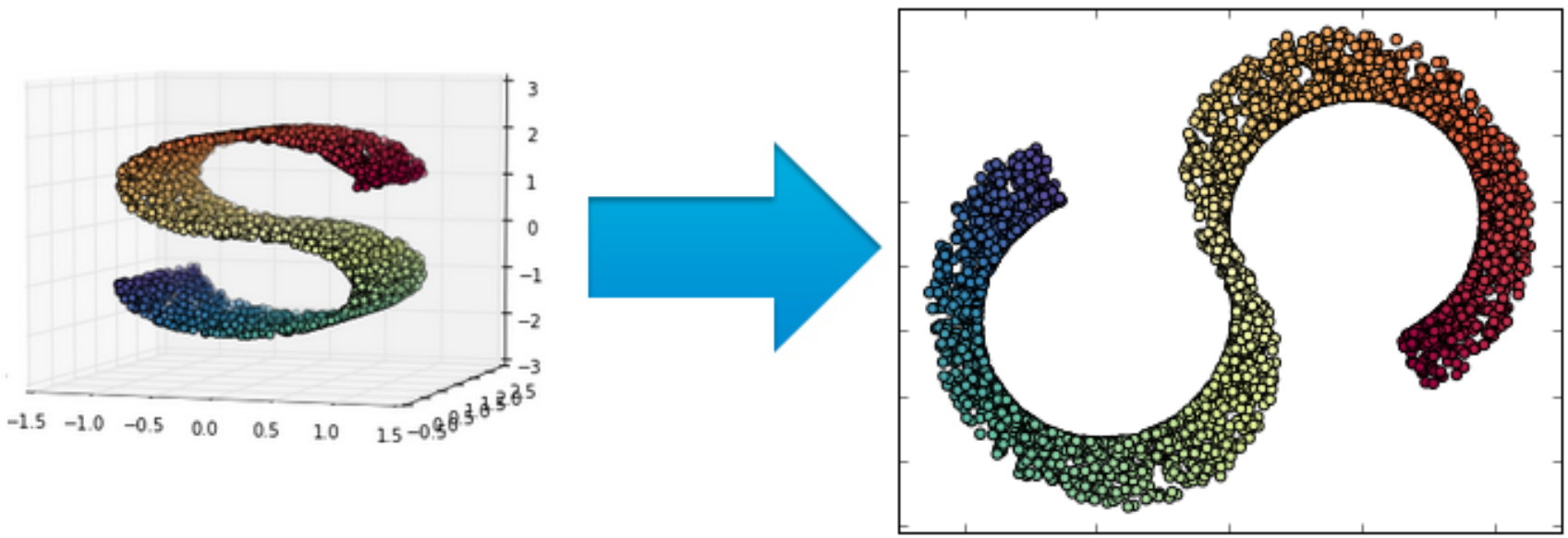
- Approaches:
  - absolute difference

$$\sum_{i,j} \left(\|x_i - x_j\| - \|y_i - y_j\| \right)^2 \to \min_Y$$

  - relative difference (more attention to small distances)

$$\sum_{i,j} \frac{\left(\|x_i - x_j\| - \|y_i - y_j\| \right)^2}{\|x_i - x_j\|} \to \min_Y$$

Example



Issue: small $\|x_i - x_j\|$ should not always imply small $\|y_i - y_j\|$, such as in case of red and yellow points.

Isomap

Isomap

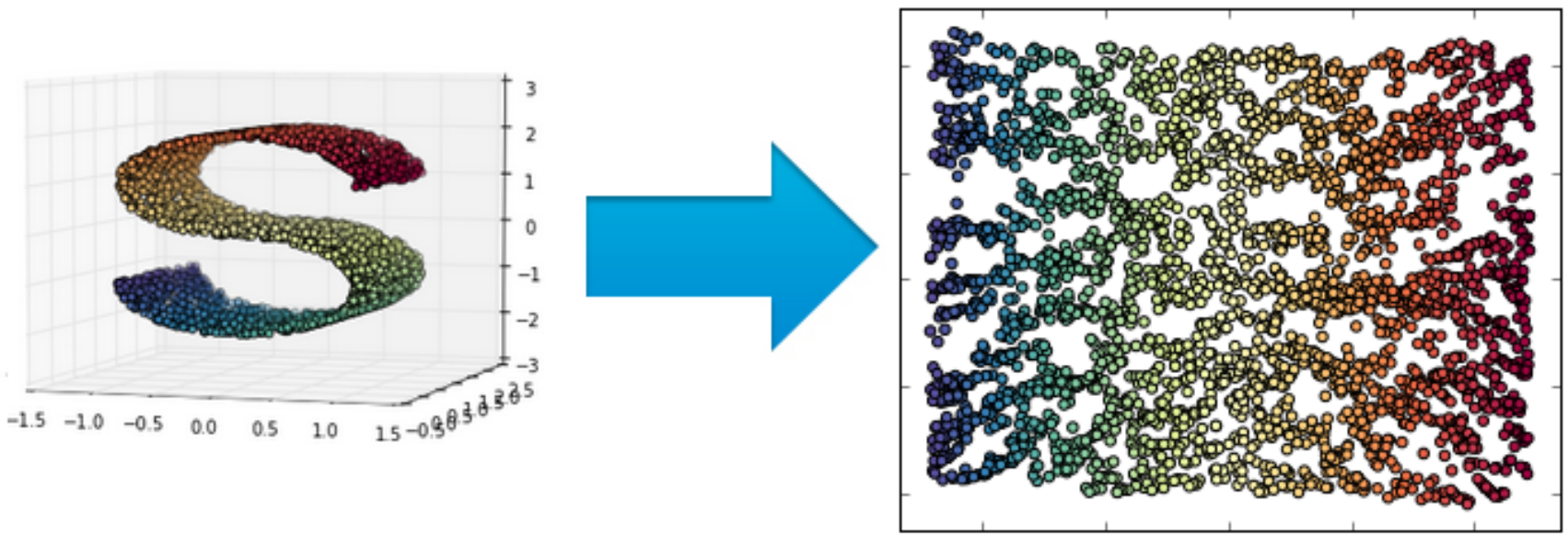
Map $x \to y$ preserving correspondence between distance in transformed space and "geodesic" distance along the surface in original space.

- This apprach solves the previous issue of MDS.
- Geodesic distance calculation:
  1. for each x_n find its K nearest neighbours $x_{n_1, n_2, \ldots n_K}$
  2. build the pairwise distance matrix, filling distance between samples and their k-NN.
  3. calculate all pairwise distances using shortest-path algorithm of Dijkstra or Floyd.
- Finally usual MDS is applied to match $\|x_i - x_j\|_G$ and $\|y_i - y_j\|$, where $\|\cdot\|_G$ is geodesic distance.

Issues of Isomap

- Noisy observations between distant parts of surfaces may make distant parts close
- Solutions:
  - remove observations with large total flows through them
  - remove nearest neighbours that violate local linearity
- Non-convexity of surface may lead to taking K-NN that are far away in the transformed space.
- Selection of K:
  - if too small, then poor approximation of geodesic distance
  - if too large, then increases chance of "short-circuiting" through noisy observations.

Example of ISOMAP



Maximum variance unfolding

Maximum variance unfolding

Maximally unfold the transformations, preserving local geometry of data.

initialize neighbourhood graph G with edges being the samples $x_1, x_2, ...x_N$

for each x_n:
  for $k = 1, 2, ...K$:
    find k-th nearest neighbour x_{n_k} to x_n
    add a link to G between x_n and x_{n_k}

solve the optimization problem:
$\sum_{i,j} ||y_i - y_j|| \rightarrow \min$ subject to: $||y_i - y_j||^2 = ||x_i - x_j|| \; \forall (i,j) \in G$

Issue: noise sample may "short-circuit" leading to redundant constraint, which may prevent manifold unfolding.

Kernel PCA

- Like PCA, but input space is expanded with kernels
- Easy computation of projections of new points
- Issue: kernel selection.
  - linear (reduces to ordinary PCA)
  - Gaussian
  - polynomial

Diffusion maps

1. Construct proximity graph
   - nodes: observations
   - edge weight between x_i and x_j:

$$w_{ij} = e^{-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2}}$$

2. for each x_i outgoing probabilities set to normalized weights:

$$p_{ij}^{(1)} = \frac{w_{ij}}{\sum_k w_{ik}}$$

3. random walk with probabilities $p_{ij}^{(1)}$ stored in matrix $P^{(1)}$ is assumed.
4. based on random walk assumption, the probability of walking from x_i to x_j after T steps is:

$$p_{ij}^{(T)} = \left(P^{(1)}\right)^T$$

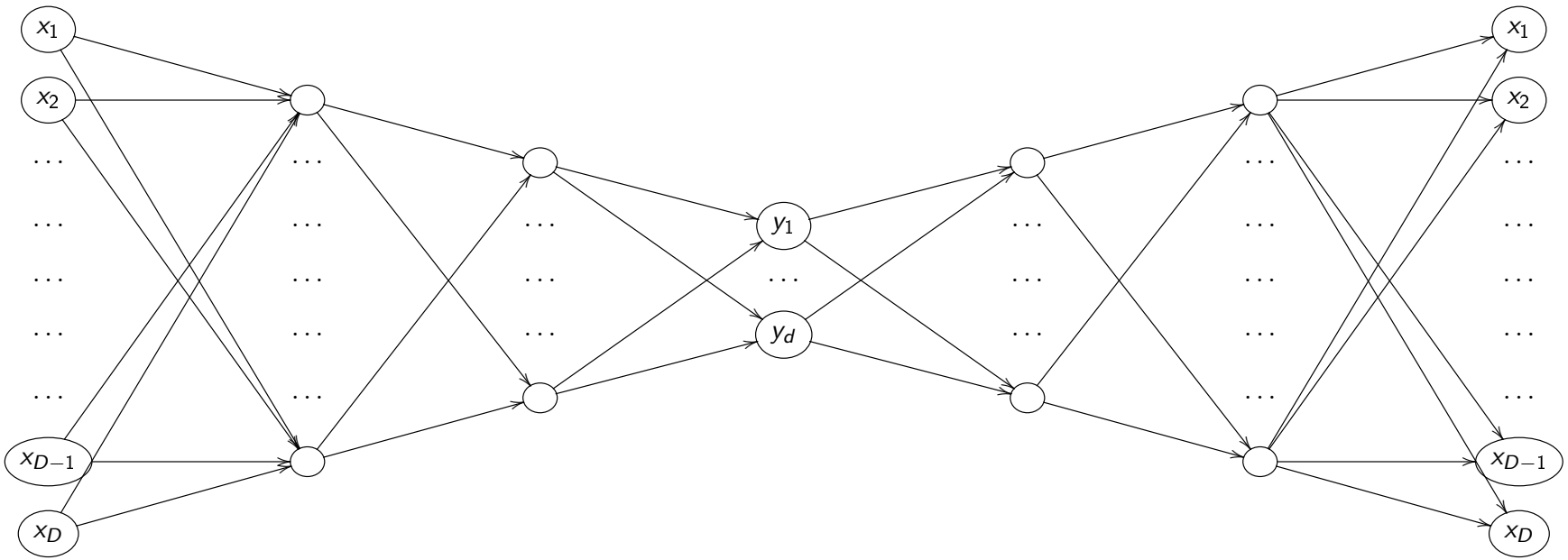
Diffusion maps

- Finally MDS is applied to match $\|y_i - y_j\|$ to *diffusion distance*:

$$D^T(x_i, x_j) = \sqrt{\sum_k \frac{(p_{ik}^{(T)} - p_{jk}^{(T)})}{\psi(x_k)}}$$

  where $\psi(x_k)$ is normalization term, that attributes more weight to high density regions.
- **Benefit:** distance between points is based on multiple paths through the graph - more robust to noise.

Autoencoders



- feed-forward neural network, tranined to reproduce input with MSE loss.
- D input and D output nodes
- d nodes in the central layer
- User-defined number of layers and nodes

Autoencoders

- Benefits: can map new points to reduced space
- Issues:
  - optimization may get stuck in local optima
  - slow convergence (can be cured with specific starting weights)
  - infeasible to apply to high d (too many connections).

4 Non-linear dimensionality reduction
- Global methods
- Local methods

Local linear embedding

Local linear embedding

Method preserves reconstruction weights of objects through their nearest neighbors.

INPUT:
training sample $x_1, x_2, \ldots x_N$
number of neighbours K

ALGORITHM:
for each x_i:
find its K nearest neighbours: $x_{i(1)}, x_{i(2)}, \ldots x_{i(K)}$
find weights to reconstruct x_i using its neighbours:
$x_i \approx \sum_{k=1}^{K} w_{ik} x_{i(k)}$

solve optimization problem: $\sum_{n=1}^{N} (y_i - \sum_{k=1}^{K} w_{ik} x_{ik})^2 \to \min_Y$

OUTPUT: reduced space representation: $y_1, y_2, \ldots y_N$.

Laplacian eigenmaps

Laplacian eigenmaps

Method preserves distances of points with nearest neighbors.

INPUT:
training sample $x_1, x_2, \ldots x_N$
number of neighbours K

ALGORITHM:
for each x_i:
find its K nearest neighbours: $x_{i(1)}, x_{i(2)}, \ldots x_{i(K)}$
for each nearest neightbour j=$i(1), i(2), \ldots i(K)$:

calculate distance-based weights: $w_{ij} = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$

solve optimization problem:
$\sum_{n=1}^{N} \sum_{i=1}^{N} \sum_{j \in \{i(1), \ldots i(K)\}} w_{ij} (y_i - y_j)^2 \to \min_Y$

OUTPUT: reduced space representation: $y_1, y_2, \ldots y_N$.

Comments on local methods

- short-circuiting affects only local points in space
- local method, relying on K-NN => prone to curse of dimensionality
- prone to overfitting on outliers (when they become nearest neighbors)

Properties

Technique	Convex	Parameters	Computational	Memory
PCA	yes	none	$O(D^3)$	$O(D^2)$
MDS	yes	none	$O(N^3)$	$O(N^2)$
Isomap	yes	K	$O(N^3)$	$O(N^2)$
MVU	yes	K	$O((NK)^3)$	$O((NK)^3)$
Kernel PCA	yes	kernel	$O(N^3)$	$O(N^2)$
Diffusion maps	yes	σ, T	$O(N^3)$	$O(N^2)$
Autoencoders	no	network shape	$O(INW)$	$O(W)$
LLE	yes	K	$O(pN^2)$	$O(pN^2)$
Laplacian eigenmaps	yes	K, σ	$O(pN^2)$	$O(pN^2)$

D - input dimension, N - sample size, K - number of nearest neighbors, σ - smoothing parameter of Gaussian kernel, W number of weights in neural network, I - number of iterations (optimization passes), p - the fraction of non-zero entries in the weight matrix.

Other

- **Problem of transforming new previously unobserved samples.**
  - direct for PCA, Kernel PCA, autoencoders
  - only approximations possible for other methods.
- **Intrinsic dimensionality estimation**
  - Cross-validation of the original task (e.g. classification)
  - How many components of local PCA explain most of the variance
  - The growth rate of number of samples falling inside a growing hypersphere with center x:

$$\#\{x_i : \|x_i - x\| \leq R\}$$

  - etc.

Evaluation

- L.J.P. van der Maaten, E.O. Postma, H.J. van den Herik. Dimensionality Reduction:A Comparative Review. Working paper. 2008.
  - Extensive comparison of different dimensionality reduction methods.
  - Non-linear techniques perform better than PCA on simulated data
  - PCA wins most of the time on real data
  - Problems:
    - global methods: short-circuiting
    - nearest neighbours based methods: curse of dimensionality, overfitting to outliers
    - unstable optimization for local methods: they reduce to eigenproblems, frequently $\lambda_{max}/\lambda_{min} \gg 1$.
    - suboptimal local optima for autoencoders.
- Actively developing field!

References


Keith D. Copsey Andrew R. Webb.
Statistical Pattern Recognition.
Wiley, 3rd edition, 2011.


Ravindran Kannan John Hopcroft.
Foundations of data science.
www.cs.cornell.edu/jeh/book11April2014.pdf, 2014.
[Online; accessed 16-August-2015].


Jeff Ullman Jure Leskovec, Anand Rajaraman.
Mining of massive datasets.
www.mmds.org, 2014.
[Online; accessed 16-August-2015].


L.J.P. van der Maaten, E. O. Postma, and H. J. van den Herik.
Dimensionality reduction: A comparative review, 2008.