Dimensionality Reduction

TTT4185 Machine Learning for Signal Processing

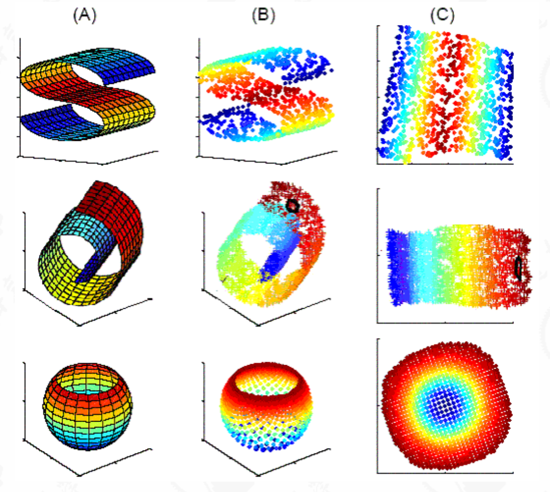
Giampiero Salvi

Department of Electronic Systems
NTNU

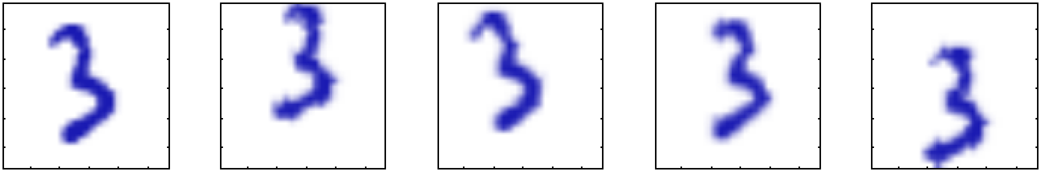
HT2020

Real Dimensionality of Data: Manifolds

- intrinsic dimension of the data
- independent of representation (features)
- manifold: low dimensional topological space embedded in feature space
- can be non-linear (but locally Euclidean)
- if linear they are subspaces

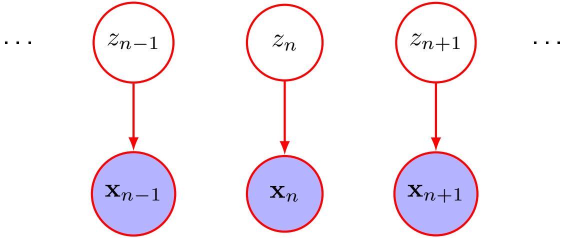


Example: Images



- one single digit example from MNIST
- tranlsations (2 degrees of freedom)
- rotations (1 degree of freedom)
- $100 \times 100 = 10{,}000$ pixels (dimensions)

Continuous Latent Variables



- discrete $z \rightarrow$ mixture models
- continuous $z \rightarrow$ dimensionality reduction

Different Models

Principal Component Analysis (PCA)

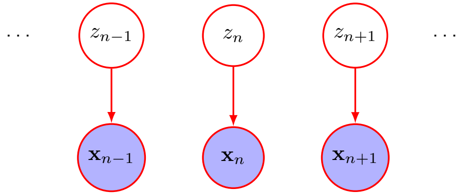
- z and x are Gaussian
- linear Gaussian dependency between x and z

Independent Component Analysis (ICA)

- non Gaussian

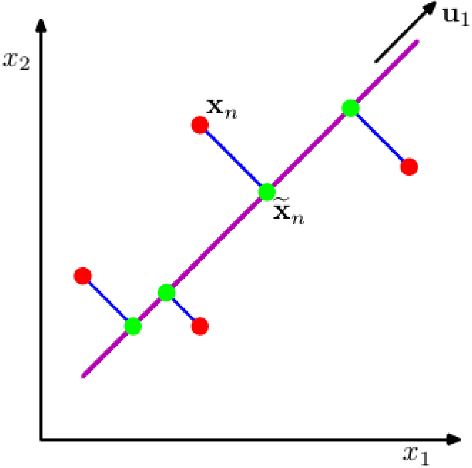
Autoencoders, Isomap, t-SNE, ...

- non-linear



Principal Component Analysis

- data in D dimensions
- sub-space with $M < D$ dimensions
- start with $M = 1$
- unit vector $\mathbf{u}_1$ $(\mathbf{u}_1^\mathsf{T}\mathbf{u}_1 = 1)$
- $\mathbf{x}_n$ is projected onto $\tilde{\mathbf{x}}_n = \mathbf{u}_1^\mathsf{T}\mathbf{x}_n$



Principal Component Analysis

- mean: $\mathbf{u}_1^\mathsf{T}\bar{\mathbf{x}}_n$, with

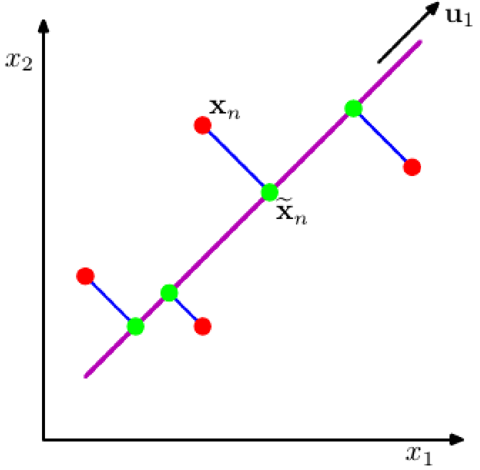
$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

- projected variance:

$$\frac{1}{N}\sum_{n=1}^{N}\left\{\mathbf{u}_1^\mathsf{T}\mathbf{x}_n - \mathbf{u}_1^\mathsf{T}\bar{\mathbf{x}}_n\right\}^2 = \mathbf{u}_1^\mathsf{T}\mathbf{S}\mathbf{u}_1$$

- with covariance matrix $\mathbf{S}$:

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\mathsf{T}$$



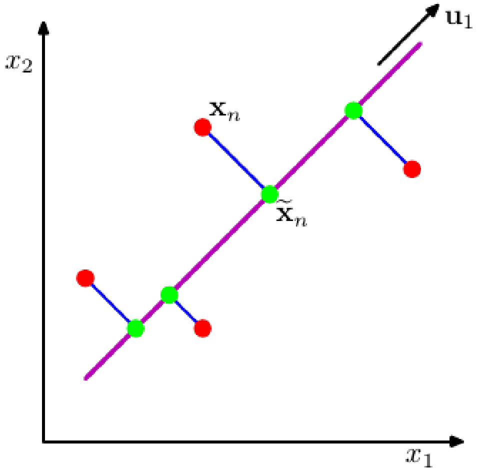
Principal Component Analysis

- maximize projected variance $\mathbf{u}_1^\mathsf{T}\mathbf{S}\mathbf{u}_1$
- with constraint that $\mathbf{u}_1^\mathsf{T}\mathbf{u}_1 = 1$
- solution:

$$\mathbf{S}\mathbf{u}_1 = \lambda_1 \mathbf{u}_1$$

- $\mathbf{u}_1$ is eigenvector of $\mathbf{S}$
- left-multiply by $\mathbf{u}_1^\mathsf{T}$:

$$\mathbf{u}_1^\mathsf{T}\mathbf{S}\mathbf{u}_1 = \lambda_1 \mathbf{u}_1^\mathsf{T}\mathbf{u}_1 = \lambda_1$$

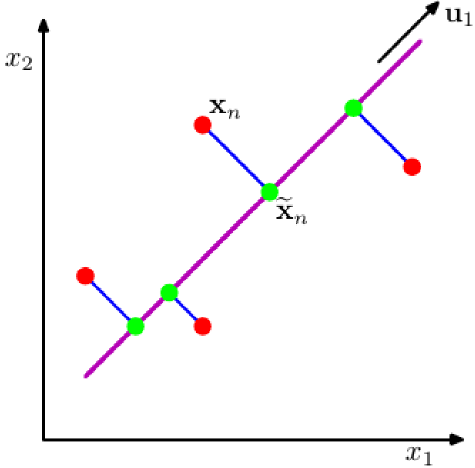


Principal Component Analysis

- find maximum eigenvalue of $\mathbf{S}$
- the corresponding eigenvector is the principal component
- find M principal components incrementally

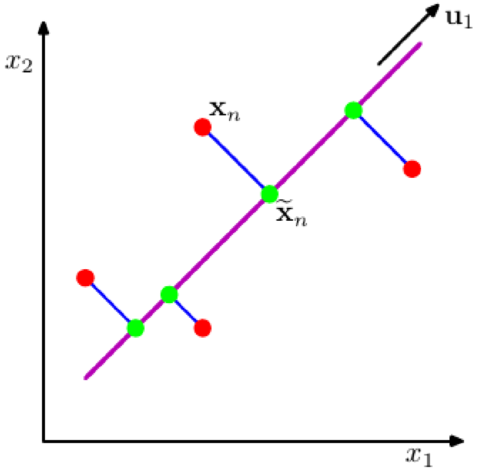
$$\mathbf{u}_1, \ldots, \mathbf{u}_M,$$
$$\mu_1, \ldots, \mu_M$$

- computational cost of eigenvector decomposition $D \times D$ is $O(D^3)$
- power method $O(MD^2)$



Principal Component Analysis

- alternative view: minimize projection square error
- same solution



PCA Applications

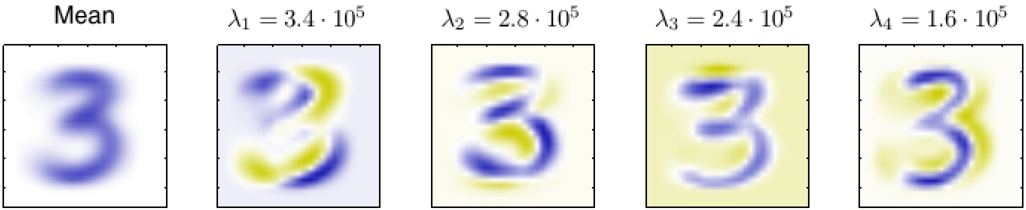
Compression:

- principal components $\mathbf{u}_1, \ldots, \mathbf{u}_M$ ($M \times D$ parameters)
- mixing weights: $\tilde{\mathbf{x}}_n = \sum_{i=1}^{M} \alpha_{ni} \mathbf{u}_i$ (M parameters)
- if N points, $M \times D + N \times M$ instead of $N \times D$

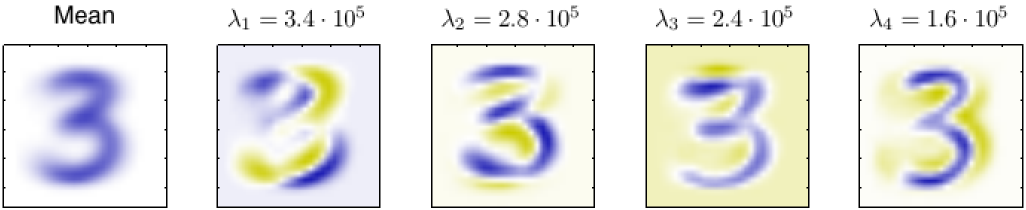
Visualization:

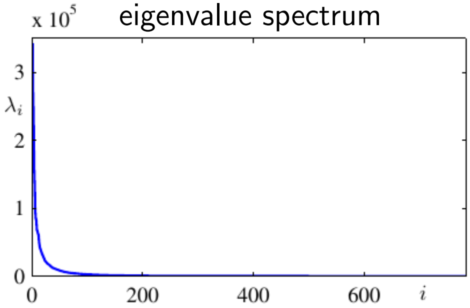
- usually $M = 2$, sometimes $M = 3$
- no big concern on reconstruction error

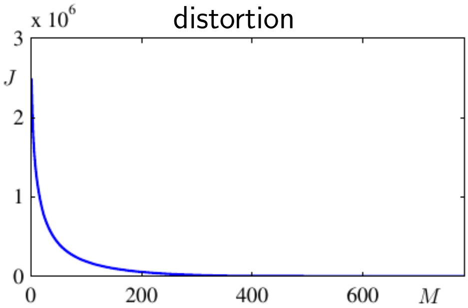
PCA Applications



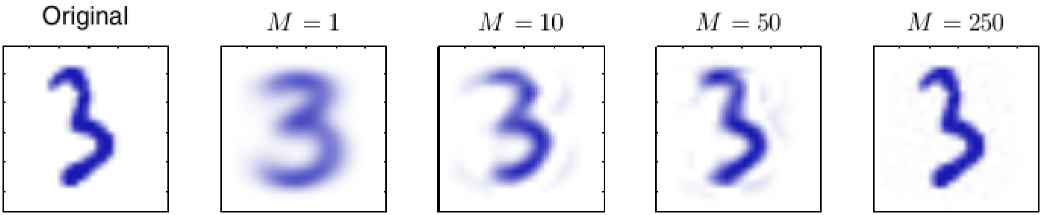
PCA Applications







PCA Reconstruction



- example $D = 10,000$, $N = 1,000,000$
- original: $N \times D = 10,000,000,000$ parameters
- PCA $(M = 10)$):
  $M \times D + N \times M = 100,000 + 10,000,000 = 10,100,000$, reduction 990 times
- PCA $(M = 250)$):
  $M \times D + N \times M = 2,500,000 + 250,000,000 = 252,500,000$, reduction 39 times

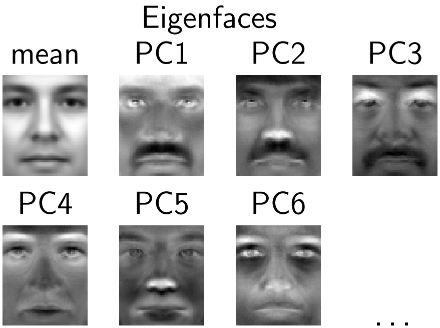
Eigenfaces

Faces from the FERET database

64×73 pixels
$= 4672$ dimensions!

Eigenfaces

Faces from the FERET database



from 4672 dimensions to a small basis

PCA for high-dimensional data

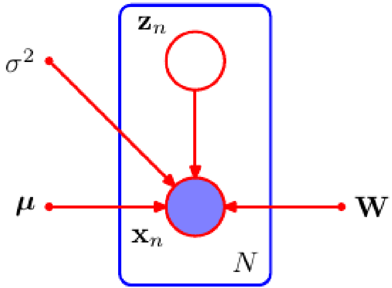
- N points in D-dimensional space, with $N < D$
- they define a subspace of at most $N - 1$ dimensions
- example: 2 points always on a line, 3 points always on a plane...
- $D - N + 1$ eigenvalues are zero!
- in the direction of the corresponding eigenvector: zero variance
- we can reformulate the eigenvector equation with a $N \times N$ matrix

Probabilistic PCA

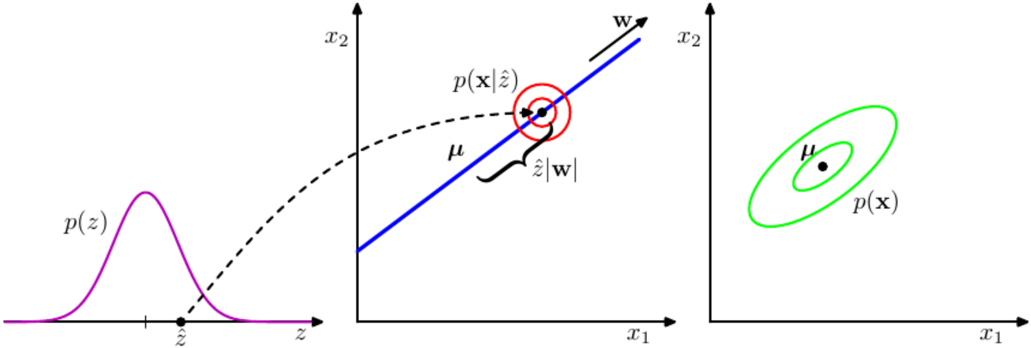
- probabilistic latent variable model
- solve with maximum likelihood

Model:

- $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$
- $p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \sigma^2\mathbf{I})$
- with $\mathbf{W}$ $D \times M$ matrix spanning the linear (principal) subspace



Probabilistic PCA: Generative View



- $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$
- $p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \sigma^2\mathbf{I})$
- $\mathbf{x} = \mathbf{W}\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}$

Probabilistic PCA: Advantages

- can be used to constrain the number of parameters in multivariate Gaussian
- can be solved with EM (computationally efficient)
- can deal with missing values
- we can extend it to mixture of PCA models
- Bayesian version can estimate the number of principal components
- likelihood function: points that are close to principal subspace but far from data distribution
- can create class-conditional densities (classification)
- can be used to generate (sample) data.

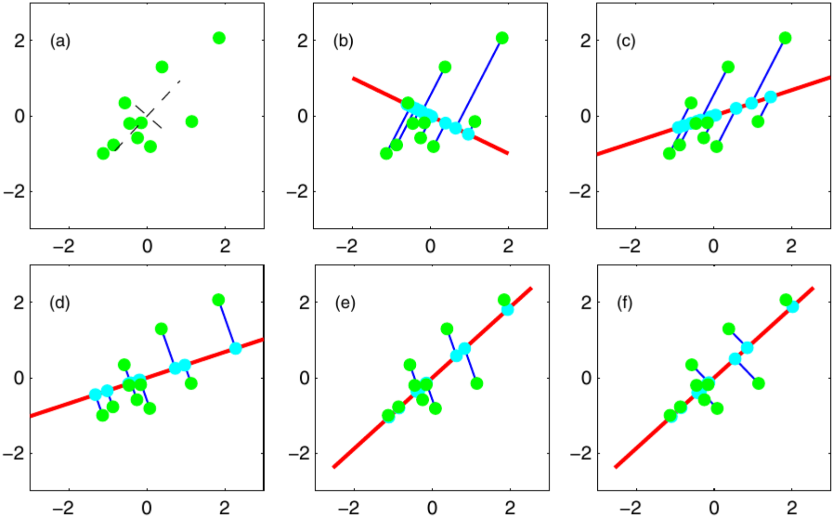
Maximum Likelihood PCA

- there exist a closed form solution to ML
- predictive distribution $p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \sigma^2\mathbf{I})$ is redundant: rotations of $\mathbf{W}$ give the same distribution
- λ_i variance in principal direction i
- σ^2 variance orthogonal to principal subspace
- statistical nonidetifiability

EM algorithm for PCA

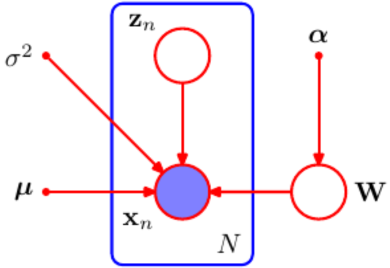
- convenient in high dimensional space (iterative instead of sample covariance matrix)
- missing values (if missing at random)
- works even for sigma square to zero (EM for standard PCA)

EM for PCA: Physical Interpretation (rod and springs)



Bayesian version can find the intrinsic dimensionality

- solution intractable
- can be approximated



Factor Analysis

- $p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \boldsymbol{\Psi})$
- $\boldsymbol{\Psi}$ diagonal (in PCA it was $\boldsymbol{\Psi} = \sigma^2\mathbf{I}$)

Independent Component Analysis (ICA)

- latent distribution $p(\mathbf{z})$ is non-Gaussian
- if $p(\mathbf{z})$ factorizes into $\prod_{j=1}^{M} p(z_j)$ then ICA

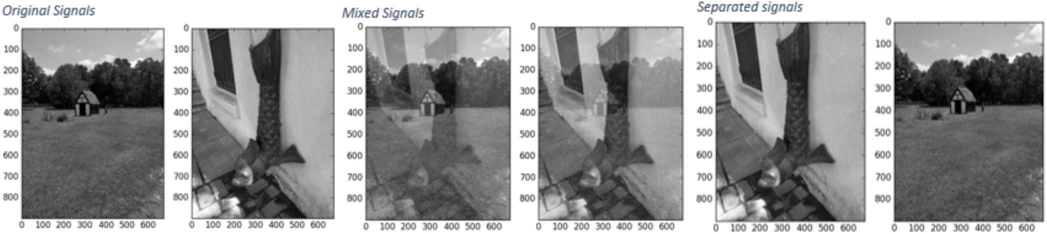
Example: blind source separation

Blind Source Separation (Speech)

- N voices picked up by M microphones
- usually $M = N$
- each microphone picks up a linear combination of the two
- ignoring room acoustic and relative movements of sources and mics
- ICA can separate the voices perfectly

http://www.kecl.ntt.co.jp/icl/signal/sawada/demo/bss2to4/index.html

Blind Source Separation (Images)

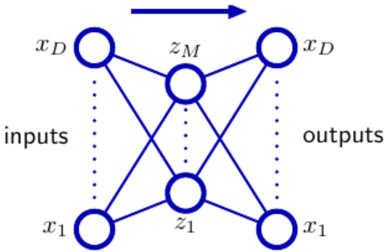


source Wikipedia

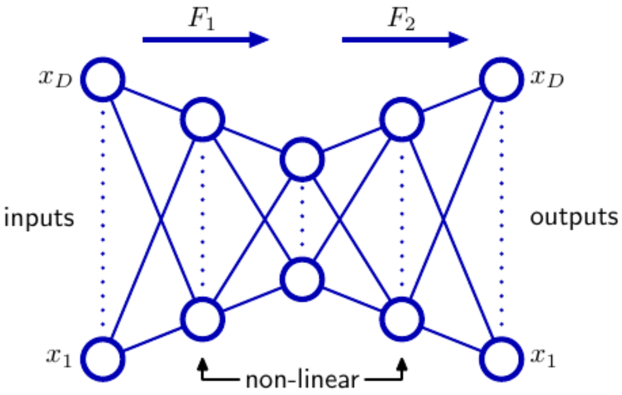
Autoencoders (linear manifolds)

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} ||\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{x}_n||^2$$

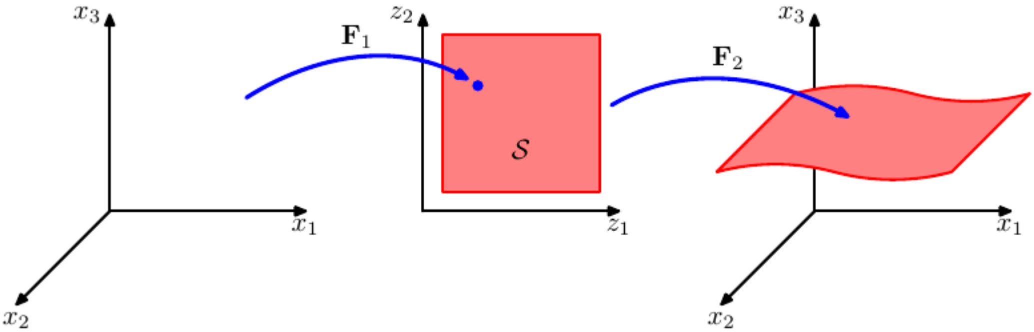
- if linear activations, then global minimum
- similar to PCA, but not orthogonal and normalized PCs
- still linear subspace even for nonlinear activations



Autoencoders (non-linear manifolds)



Autoencoders: mapping illustration

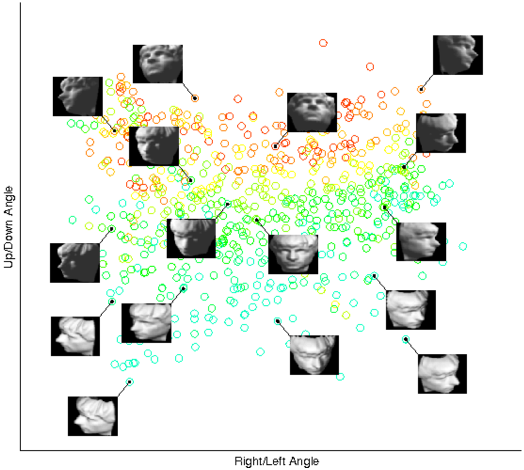


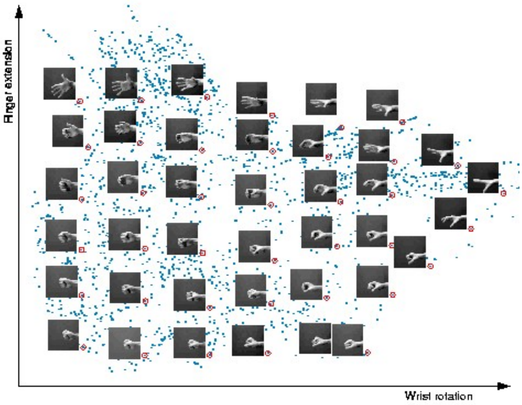
Isometric Feature Mapping (Isomap)

Using geodesic distances

https://chart-studio.plotly.com/~empet/14345.embed

Isomap Examples

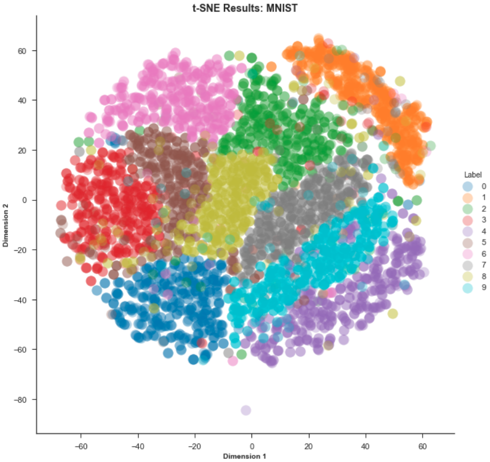




t-SNE (not in the book)

t-distributed stochastic neighbor embedding

- works best for visualization (2-3 dim)
- similar groups of points are close
- probability distribution over pairs of points in high dim (pairs of more similar points have higher probability)
- probability distribution over pairs of points in low dimensions
- minimize KL divergence



van der Maaten, L. and Hinton, G. E. (2008). Visualizing data using t-SNE. J. Machine Learning Res., 9 . 473 , 516