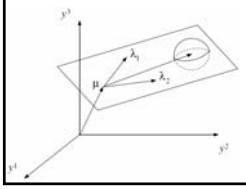
Machine Learning

10-701/15-781, Spring 2008

Factor Analysis and Metric Learning

Eric Xing



Lecture 26, April 23, 2008

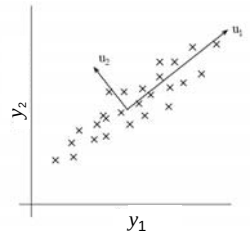
Reading: Chap. 12.3-4, C.B book

Outline

- Probabilistic PCA (breif)
- Factor Analysis (somewhat detail)
- ICA (will skip)
- Distance metric learning from very little side info (a very cool method)

Recap of PCA

- Popular dimensionality reduction technique
- Project data onto directions of greatest variation



$$u = \arg\max \frac{1}{m}\sum_{i=1}^{m}\left(\vec{y}_i^T u\right)^2$$
$$= \arg\max u^T\left(\frac{1}{m}\sum_{i=1}^{m}\vec{y}_i\vec{y}_i^T\right)u$$
$$= \arg\max\left(u^T Cov(y)u\right)$$

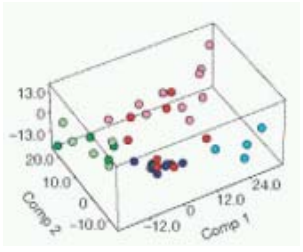
$$\vec{x}_i = \begin{bmatrix} u_1^T\vec{y}_i \\ u_2^T\vec{y}_i \\ \vdots \\ u_q^T\vec{y}_i \end{bmatrix} = U_q^T\vec{y}_i \in \mathbb{R}^q$$

$$\vec{y}_i = U\vec{x}_i$$

- Consequence:
  - x_i are uncorrelated such that the covariance matrix $\frac{1}{m}\sum_{i=1}^{m}\vec{x}_i\vec{x}_i^T$ is $\begin{bmatrix}\gamma_1 & & \\ & \ddots & \\ & & \gamma_q\end{bmatrix}$
  - Truncation error $\Sigma_y = \sum_{k=1}^{K}\gamma_k\left(u_k u_k^T\right) \approx \sum_{k=1}^{q}\gamma_k\left(u_k u_k^T\right) = \Sigma_x$

Recap of PCA

- Popular dimensionality reduction technique
- Project data onto directions of greatest variation

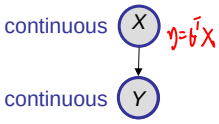


Useful tool for visualising patterns and clusters within the data set, but …

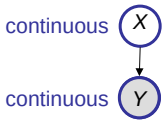
Need centering

Does not explicitly model data noise

Probabilistic Interpretation?



regression



?

Probabilistic PCA

- PCA can be cast as a probabilistic model

$$y_n = \Lambda x_n + \mu + \varepsilon_n \qquad \varepsilon_n \sim \mathcal{N}(0, \sigma^2 I)$$

  with q-dimensional latent variables $x_n \sim \mathcal{N}(0, I)$

- The resulting data distribution is

$$y_n \sim \mathcal{N}(\mu, \Lambda\Lambda^T + \sigma^2 I)$$

- Maximum likelihood solution is equivalent to PCA

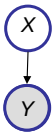
$$\mu^{ML} = \frac{1}{N}\sum_n y_n \qquad \Lambda^{ML} = U_q(\Gamma_q - \sigma^2 I)^{1/2}$$

  Diagonal Γ_q contains the top q sample covariance eigen-values and U_q contains associated eigenvectors

Tipping and Bishop, *J. Royal Stat. Soc.* **6**, 611 (1999).

Factor analysis

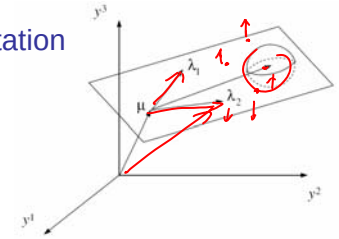
- An unsupervised linear regression model



$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; 0, I)$$
$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}; \mu + \Lambda\mathbf{x}, \Psi)$$

where Λ is called a factor loading matrix, and Ψ is diagonal.

- Geometric interpretation



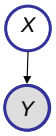
- To generate data, first generate a point within the manifold then add noise. Coordinates of point are components of latent variable.

Relationship between PCA and FA

- Probabilistic PCA is equivalent to factor analysis with equal noise for every dimension, i.e., ε_n~ isotropic Gaussian $\mathcal{N}(0, \sigma^2 I)$
- In factor analysis $\varepsilon_n \sim \mathcal{N}(0, \Psi)$ for a diagonal covariance matrix Ψ
- An iterative algorithm (eg. EM) is required to find parameters if precisions are not known in advance

Factor analysis

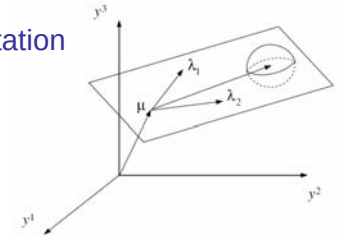
- An unsupervised linear regression model



$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; 0, I)$$
$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}; \mu + \Lambda\mathbf{x}, \Psi)$$

where Λ is called a factor loading matrix, and Ψ is diagonal.

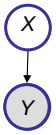
- Geometric interpretation



  - To generate data, first generate a point within the manifold then add noise. Coordinates of point are components of latent variable.

Marginal data distribution

- A marginal Gaussian (e.g., $p(\mathbf{x})$) times a conditional Gaussian (e.g., $p(\mathbf{y}|\mathbf{x})$) is a joint Gaussian
- Any marginal (e.g., $p(\mathbf{y})$ of a joint Gaussian (e.g., $p(\mathbf{x},\mathbf{y})$) is also a Gaussian
  - Since the marginal is Gaussian, we can determine it by just computing its mean and variance. (Assume noise uncorrelated with data.)



$$\begin{aligned} E[\mathbf{Y}] &= E[\mu + \Lambda\mathbf{X} + \mathbf{W}] \quad \text{where } \mathbf{W} \sim \mathcal{N}(0, \Psi) \\ &= \mu + \Lambda E[\mathbf{X}] + E[\mathbf{W}] \\ &= \mu + 0 + 0 = \mu \end{aligned}$$

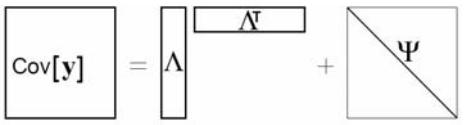
$$\begin{aligned} Var[\mathbf{Y}] &= E\left[(\mathbf{Y} - \mu)(\mathbf{Y} - \mu)^T\right] \\ &= E\left[(\mu + \Lambda\mathbf{X} + \mathbf{W} - \mu)(\mu + \Lambda\mathbf{X} + \mathbf{W} - \mu)^T\right] \\ &= E\left[(\Lambda\mathbf{X} + \mathbf{W})(\Lambda\mathbf{X} + \mathbf{W})^T\right] \\ &= \Lambda E\left[\mathbf{X}\mathbf{X}^T\right]\Lambda^T + E\left[\mathbf{W}\mathbf{W}^T\right] \\ &= \Lambda\Lambda^T + \Psi \end{aligned}$$

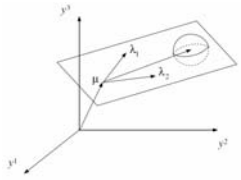
FA = Constrained-Covariance Gaussian

- Marginal density for factor analysis ($\mathbf{y}$ is p-dim, $\mathbf{x}$ is k-dim):

$$p(\mathbf{y} \mid \theta) = \mathcal{N}(\mathbf{y}; \mu, \Lambda\Lambda^T + \Psi)$$

- So the effective covariance is the low-rank outer product of two long skinny matrices plus a diagonal matrix:



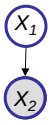


- In other words, factor analysis is just a constrained Gaussian model. (If Ψ were not diagonal then we could model any Gaussian and it would be pointless.)

Review: A primer to multivariate Gaussian

- Multivariate Gaussian density:

$$p(\mathbf{x} \mid \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left\{-\tfrac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1}(\mathbf{x}-\mu)\right\}$$



- A joint Gaussian:

$$p\left(\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} \mid \mu, \Sigma\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} \middle| \begin{bmatrix}\mu_1\\ \mu_2\end{bmatrix}, \begin{bmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{bmatrix}\right)$$

- How to write down $p(\mathbf{x}_1)$, $p(\mathbf{x}_1|\mathbf{x}_2)$ or $p(\mathbf{x}_2|\mathbf{x}_1)$ using the block elements in μ and Σ?
  - Formulas to remember:

$$p(\mathbf{x}_2) = \mathcal{N}(\mathbf{x}_2 \mid \mathbf{m}_2^m, \mathbf{V}_2^m) \qquad p(\mathbf{x}_1|\mathbf{x}_2) = \mathcal{N}(\mathbf{x}_1 \mid \mathbf{m}_{1|2}, \mathbf{V}_{1|2})$$

$$\mathbf{m}_2^m = \mu_2 \qquad \mathbf{m}_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \mu_2)$$

$$\mathbf{V}_2^m = \Sigma_{22} \qquad \mathbf{V}_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

Review: Some matrix algebra

- Trace and derivatives
  - Cyclical permutations
  - Derivatives

$$\mathrm{tr}[A] \overset{\text{def}}{=} \sum_i a_{ii}$$

$$\mathrm{tr}[ABC] = \mathrm{tr}[CAB] = \mathrm{tr}[BCA]$$

$$\frac{\partial}{\partial A}\mathrm{tr}[BA] = B^T$$

$$\frac{\partial}{\partial A}\mathrm{tr}[x^T Ax] = \frac{\partial}{\partial A}\mathrm{tr}[xx^T A] = xx^T$$

- Determinants and derivatives

$$\frac{\partial}{\partial A}\log|A| = A^{-T}$$

FA joint distribution

- Model

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; 0, I)$$
$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}; \mu + \Lambda\mathbf{x}, \Psi)$$

- Covariance between $\mathbf{x}$ and $\mathbf{y}$

$$\begin{aligned} Cov[\mathbf{x}, \mathbf{y}] &= E\left[(\mathbf{x} - 0)(\mathbf{y} - \mu)^T\right] = E\left[\mathbf{x}(\mu + \Lambda\mathbf{x} + \mathbf{w} - \mu)^T\right] \\ &= E\left[\mathbf{x}\mathbf{x}^T\Lambda^T + \mathbf{x}\mathbf{w}^T\right] \\ &= \Lambda^T \end{aligned}$$

- Hence the joint distribution of $\mathbf{x}$ and $\mathbf{y}$:

$$p\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix} \middle| \begin{bmatrix}0\\ \mu\end{bmatrix}, \begin{bmatrix}I & \Lambda^T\\ \Lambda & \Lambda\Lambda^T + \Psi\end{bmatrix}\right)$$

- Assume noise is uncorrelated with data or latent variables.

Inference in Factor Analysis

- Apply the Gaussian conditioning formulas to the joint distribution we derived above, where

$$\Sigma_{11} = I$$
$$\Sigma_{12} = \Sigma_{12}^{\ T} = \Lambda^T$$
$$\Sigma_{22} = \left(\Lambda\Lambda^T + \Psi\right)$$

we can now derive the posterior of the latent variable $\mathbf{x}$ given observation $\mathbf{y}$, $p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x} \mid \mathbf{m}_{1|2}, \mathbf{V}_{1|2})$, where

$$\mathbf{m}_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{y} - \mu_2) = \Lambda^T\left(\Lambda\Lambda^T + \Psi\right)^{-1}(\mathbf{y} - \mu) \qquad \mathbf{V}_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} = I - \Lambda^T\left(\Lambda\Lambda^T + \Psi\right)^{-1}\Lambda$$

Applying the matrix inversion lemma $\left(E - FH^{-1}G\right)^{-1} = E^{-1} + E^{-1}F\left(H - GE^{-1}F\right)^{-1}GE^{-1}$

$$\Rightarrow \quad \mathbf{V}_{1|2} = \left(I + \Lambda^T\Psi^{-1}\Lambda\right)^{-1} \qquad \mathbf{m}_{1|2} = \mathbf{V}_{1|2}\Lambda^T\Psi^{-1}(\mathbf{y} - \mu)$$

- Here we only need to invert a matrix of size |**x**|×|**x**|, instead of |**y**|×|**y**|.

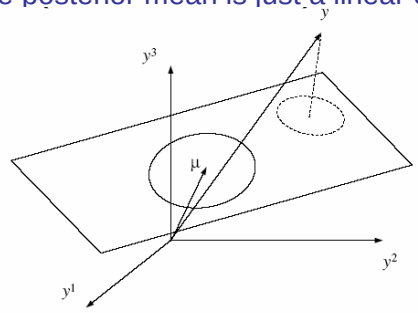
Geometric interpretation: inference is linear projection

- The posterior is:

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}; \mathbf{m}_{1|2}, \mathbf{V}_{1|2})$$

$$\mathbf{V}_{1|2} = \left(I + \Lambda^T\Psi^{-1}\Lambda\right)^{-1} \qquad \mathbf{m}_{1|2} = \mathbf{V}_{1|2}\Lambda^T\Psi^{-1}(\mathbf{y} - \mu)$$

- Posterior covariance does not depend on observed data $\mathbf{y}$!
- Computing the posterior mean is just a linear operation:



EM for Factor Analysis

- Incomplete data log likelihood function (marginal density of y)

$$\ell(\theta, D) = -\frac{N}{2}\log\left|\Lambda\Lambda^T + \Psi\right| - \frac{1}{2}\sum_n (y_n - \mu)^T \left(\Lambda\Lambda^T + \Psi\right)^{-1}(y_n - \mu)$$

$$= -\frac{N}{2}\log\left|\Lambda\Lambda^T + \Psi\right| - \frac{1}{2}\mathrm{tr}\left[\left(\Lambda\Lambda^T + \Psi\right)^{-1}\mathbf{S}\right], \qquad \text{where } \mathbf{S} = \sum_n (y_n - \mu)(y_n - \mu)^T$$

  - Estimating m is trivial: $\hat{\mu}^{ML} = \frac{1}{N}\sum_n y_n$
  - Parameters Λ and Ψ are coupled nonlinearly in log-likelihood

- Complete log likelihood

$$\ell_c(\theta, D) = \sum_n \log p(x_n, y_n) = \sum_n \log p(x_n) + \log p(y_n \mid x_n)$$

$$= -\frac{N}{2}\log|I| - \frac{1}{2}\sum_n x_n^T x_n - \frac{N}{2}\log|\Psi| - \frac{1}{2}\sum_n (y_n - \Lambda x_n)^T \Psi^{-1}(y_n - \Lambda x_n)$$

$$= -\frac{N}{2}\log|\Psi| - \frac{1}{2}\sum_n \mathrm{tr}\left[x_n x_n^T\right] - \frac{N}{2}\mathrm{tr}\left[\mathbf{S}\Psi^{-1}\right], \qquad \text{where } \mathbf{S} = \frac{1}{N}\sum_n (y_n - \Lambda x_n)(y_n - \Lambda x_n)^T$$

E-step for Factor Analysis

- Compute $\langle \ell_c(\theta, D) \rangle_{p(x|y)}$

$$\langle \ell_c(\theta, D) \rangle = -\frac{N}{2}\log|\Psi| - \frac{1}{2}\sum_n \mathrm{tr}\left[\langle X_n X_n^T \rangle\right] - \frac{N}{2}\mathrm{tr}\left[\langle \mathbf{S} \rangle \Psi^{-1}\right]$$

$$\langle \mathbf{S} \rangle = \frac{1}{N}\sum_n \left(y_n y_n^T - y_n \langle X_n^T \rangle \Lambda^T - \Lambda \langle X_n^T \rangle y_n^T + \Lambda \langle X_n X_n^T \rangle \Lambda^T\right)$$

$$\langle X_n \rangle = E[X_n \mid y_n]$$

$$\langle X_n X_n^T \rangle = Var[X_n \mid y_n] + E[X_n \mid y_n] E[X_n \mid y_n]^T$$

  - Recall that we have derived:

$$\mathbf{V}_{1|2} = \left(I + \Lambda^T \Psi^{-1} \Lambda\right)^{-1} \qquad \mathbf{m}_{1|2} = \mathbf{V}_{1|2} \Lambda^T \Psi^{-1}(\mathbf{y} - \mu)$$

$$\Rightarrow \quad \langle X_n \rangle = \mathbf{m}_{x_n|y_n} = \mathbf{V}_{1|2} \Lambda^T \Psi^{-1}(y_n - \mu) \quad \text{and} \quad \langle X_n X_n^T \rangle = \mathbf{V}_{1|2} + \mathbf{m}_{x_n|y_n} \mathbf{m}_{x_n|y_n}^T$$

M-step for Factor Analysis

- Take the derivates of the expected complete log likelihood wrt. parameters.
  - Using the trace and determinant derivative rules:

$$\frac{\partial}{\partial \Psi^{-1}}\langle \ell_c \rangle = \frac{\partial}{\partial \Psi^{-1}}\left(-\frac{N}{2}\log|\Psi| - \frac{1}{2}\sum_n \mathrm{tr}\left[\left\langle X_n X_n^T \right\rangle\right] - \frac{N}{2}\mathrm{tr}\left[\langle \mathbf{S} \rangle \Psi^{-1}\right]\right)$$

$$= \frac{N}{2}\Psi - \frac{N}{2}\langle \mathbf{S} \rangle \quad \Rightarrow \quad \Psi^{t+1} = \langle \mathbf{S} \rangle$$

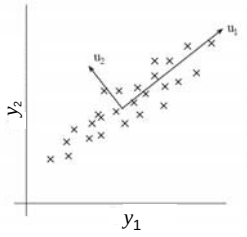
$$\frac{\partial}{\partial \Lambda}\langle \ell_c \rangle = \frac{\partial}{\partial \Lambda}\left(-\frac{N}{2}\log|\Psi| - \frac{1}{2}\sum_n \mathrm{tr}\left[\left\langle X_n X_n^T \right\rangle\right] - \frac{N}{2}\mathrm{tr}\left[\langle \mathbf{S} \rangle \Psi^{-1}\right]\right) = -\frac{N}{2}\Psi^{-1}\frac{\partial}{\partial \Lambda}\langle \mathbf{S} \rangle$$

$$= -\frac{N}{2}\Psi^{-1}\frac{\partial}{\partial \Lambda}\left(\frac{1}{N}\sum_n \left(y_n y_n^T - y_n \left\langle X_n^T \right\rangle \Lambda^T - \Lambda \left\langle X_n^T \right\rangle y_n^T + \Lambda \left\langle X_n X_n^T \right\rangle \Lambda^T\right)\right)$$

$$= \Psi^{-1}\sum_n y_n \left\langle X_n^T \right\rangle - \Psi^{-1}\Lambda \sum_n \left\langle X_n X_n^T \right\rangle \quad \Rightarrow \quad \Lambda^{t+1} = \left(\sum_n y_n \left\langle X_n^T \right\rangle\right)\left(\sum_n \left\langle X_n X_n^T \right\rangle\right)^{-1}$$

Comparison of PCA and FA

- PCA

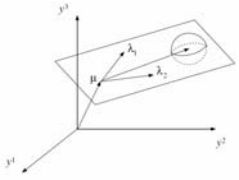


$$y_n = Ux_n$$

$$u = \arg\max\left(u^T Cov(y) u\right)$$

$$\vec{x}_i = \begin{bmatrix} u_1^T \vec{y}_i \\ u_2^T \vec{y}_i \\ \vdots \\ u_k^T \vec{y}_i \end{bmatrix} = U_q^T \vec{y}_i \in \mathbb{R}^q$$

- FA



$$y_n = \Lambda x_n + \mu + \varepsilon_n$$

$$\varepsilon_n \sim \mathcal{N}(0, \Psi)$$

$$\langle X_n \rangle = \mathbf{m}_{x_n|y_n} = \mathbf{V}_{1|2}\Lambda^T \Psi^{-1}(y_n - \mu)$$

and $$\left\langle X_n X_n^T \right\rangle = \mathbf{V}_{1|2} + \mathbf{m}_{x_n|y_n}\mathbf{m}_{x_n|y_n}^T$$

$$\Lambda^{t+1} = \left(\sum_n y_n \left\langle X_n^T \right\rangle\right)\left(\sum_n \left\langle X_n X_n^T \right\rangle\right)^{-1}$$

Comparison of PCA and FA

- PCA

$$u = \arg\max\left(u^T Cov(y) u\right)$$

$$\vec{x}_i = \begin{bmatrix} u_1^T \vec{y}_i \\ u_2^T \vec{y}_i \\ \vdots \\ u_k^T \vec{y}_i \end{bmatrix} = U_q^T \vec{y}_i \in \mathbb{R}^q$$

  - SVD on a $K \times K$ matrix
  - Covariant under rotation: Ay
  - Principle axis can be found incrementally

- FA

$$\langle X_n \rangle = \mathbf{m}_{x_n|y_n} = \mathbf{V}_{1|2} \Lambda^T \Psi^{-1} (y_n - \mu)$$

and $$\langle X_n X_n^T \rangle = \mathbf{V}_{1|2} + \mathbf{m}_{x_n|y_n} \mathbf{m}_{x_n|y_n}^T$$

$$\Lambda^{t+1} = \left(\sum_n y_n \langle X_n^T \rangle \right) \left(\sum_n \langle X_n X_n^T \rangle \right)^{-1}$$

$$\Psi^{t+1} = \langle \mathbf{S} \rangle$$

  - Invert a $q \times q$ matrix
  - Covariant under rescaling: $diag(\alpha)y$
  - Neither of the factors found by a two-factor model is necessarily the same as that found by a single factor model, and …

Example:

Original data matrix → Correlation matrix → Factor matrix

Variables

Obs	v_1	v_2	.	.	.	v_{10}
o_1						
o_2						
.						
.						
.						
o_n						

Observational data

Variables

	v_1	v_2	.	.	.	v_{10}
v_1						
v_2						
.						
.						
.						
v_{10}						

Correlation coefficients

Factors

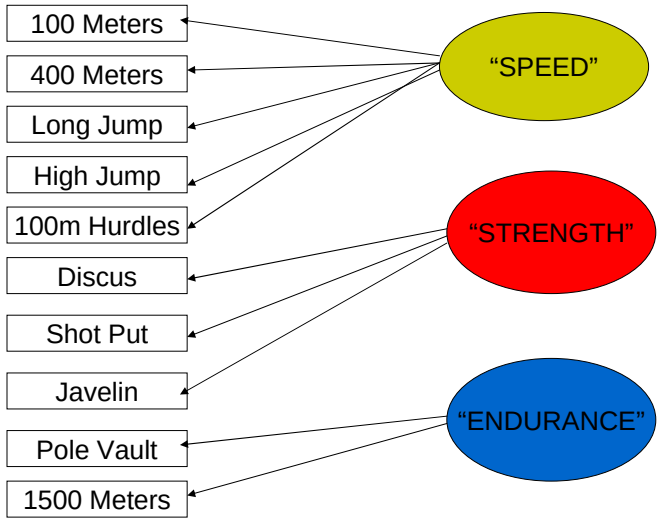
	F_1 Spd	F_2 Str	F_3 End
100m	.87	.07	.14
Long	.77	.27	.13
High	.51	.34	.34
110m	.63	.32	.05
400m	.74	.06	.38
Discus	.22	.79	.06
Shot	.31	.82	.10
Javelin	.02	.70	.15
Pole	.24	.40	.50
1500m	.02	.12	.89

Factor loadings

Decisions

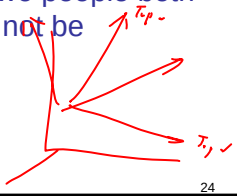
- Factoring method
- # of factors to retain
- Factor rotation

Decathlon example



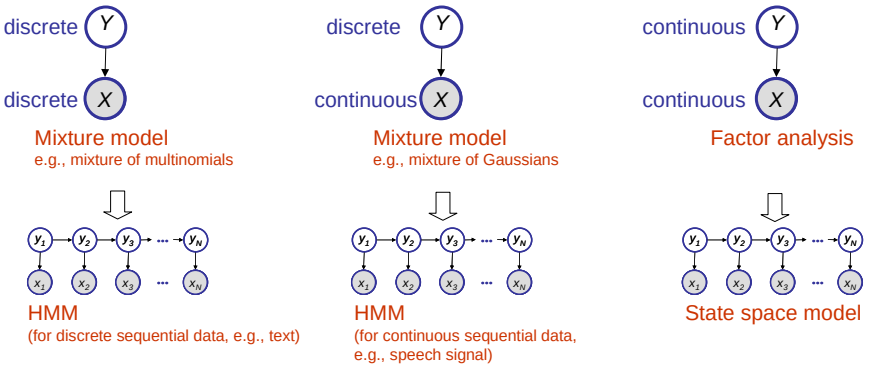
Model Invariance and Identifiability

- There is *degeneracy* in the FA model.
- Since Λ only appears as outer product $\Lambda\Lambda^T$, the model is invariant to rotation and axis flips of the latent space.
- We can replace Λ with ΛQ for any orthonormal matrix Q and the model remains the same: $(\Lambda Q)(\Lambda Q)^T=\Lambda(QQ^T)\Lambda^T=\Lambda\Lambda^T$.
- This means that there is no "one best" setting of the parameters. An infinite number of parameters all give the ML score!
- Such models are called un-identifiable since two people both fitting ML parameters to the identical data will not be guaranteed to identify the same parameters.



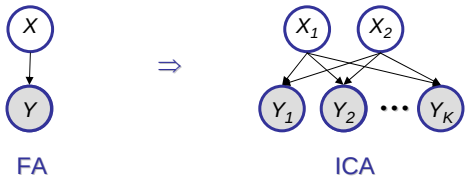
Why FA

- Latent trajectories

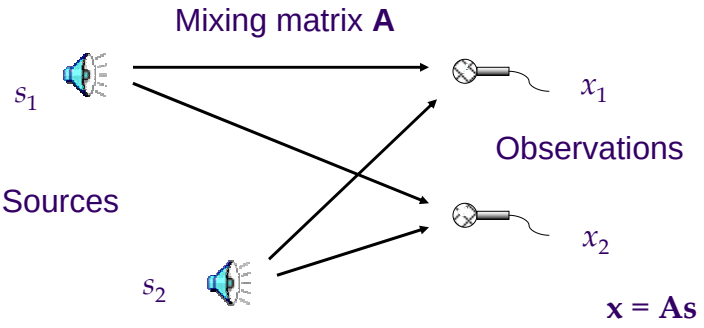


Independent Components Analysis (ICA)

- ICA is similar to FA, except it assumes the latent source has non-Gaussian density.
- Hence ICA can extract higher order moments (not just second order).
- It is commonly used to solve blind source separation (cocktail party problem).



The simple "Cocktail Party" Problem

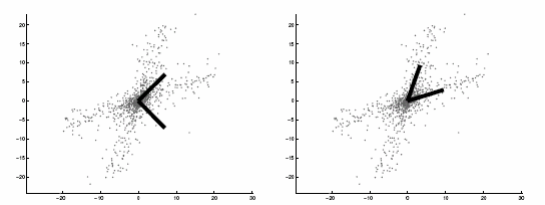


n sources, m=*n* observations

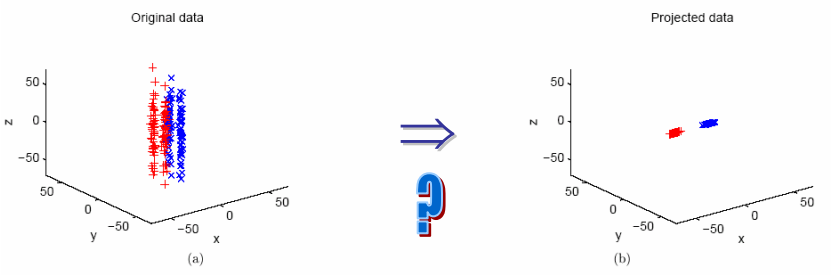
We skip more details and next introduce a more interesting new algorithm!

ICA versus PCA (and FA)

- Similarity
  - Feature extraction
  - Dimension reduction
- Difference
  - PCA uses up to second order moment of the data to produce uncorrelated components
  - ICA strives to generate components as independent as possible



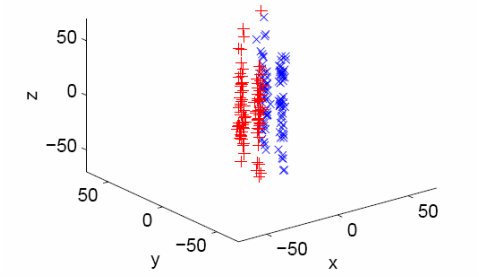
Semi-supervised Metric Learning



Xing et al, NIPS 2003

What is a good metric?

- What is a good metric over the input space for learning and data-mining



  - How to convey metrics sensible to a human user (e.g., dividing traffic along highway lanes rather than between overpasses, categorizing documents according to writing style rather than topic) to a computer data-miner using a systematic mechanism?

Issues in learning a metric

- Data distribution is self-informing (E.g., lies in a sub-manifold)
  - Learning metric by finding an embedding of data in some space.
    - Con: does not reflect (changing) human subjectiveness.
- Explicitly labeled dataset offers clue for critical features
  - Supervised learning
    - Con: needs sizable homogeneous training sets.
- What about side information? (E.g., x and y look (or read) similar ...)
  - Providing small amount of qualitative and less structured side information is often much easier than stating explicitly a metric (what should be the metric for writing style?) or labeling a large set of training data.
- Can we learn a distance metric more informative than Euclidean distance using a small amount of side information?

Distance Metric Learning

Side information:

Suppose for some set of points $\{x_i\}_{i=1}^m \subseteq \mathbb{R}^n$, we are given:

$\mathcal{S}$: $(x_i, x_j) \in \mathcal{S}$ if x_i and x_j are similar

$\mathcal{D}$: $(x_i, x_j) \in \mathcal{D}$ if x_i and x_j are dissimilar

Distance metric learning:

Learn a distance metric of the form

$$d(x, y) = d_A(x, y) = ||x - y||_A = \sqrt{(x - y)^T A (x - y)},$$

such that pairs of points (x_i, x_j) in $\mathcal{S}$ have small squared distance.

- In general, A parameterizes a family of Mahalanobis distances over $\mathbb{R}^n$.
- Learning A is equivalent to finding a rescaling of a data: $x \to A^{1/2}x$.

Optimal Distance Metric

- Learning an optimal distance metric with respect to the side-information leads to the following optimization problem:

$$\min_A \quad \sum_{(x_i,x_j)\in\mathcal{S}} ||x_i - x_j||_A^2 \tag{1}$$

$$\text{s.t.} \quad \sum_{(x_i,x_j)\in\mathcal{D}} ||x_i - x_j||_A \geq 1, \tag{2}$$

$$A \geq 0. \tag{3}$$

  - This optimization problem is *convex*. Local-minima-free algorithms exist.
  - Xing et al 2003 provided an efficient gradient descent + iterative constraint-projection method

Examples of learned distance metrics

- Distance metrics learned on three-cluster artificial data:

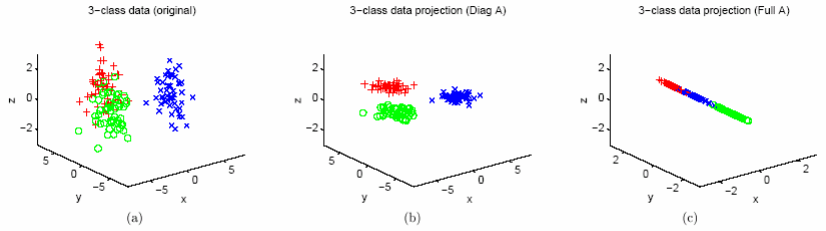
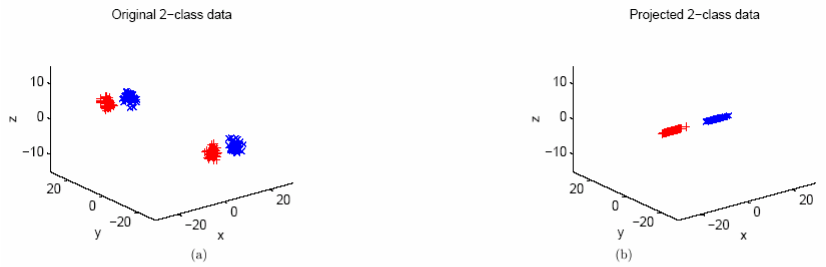


Figure 2: (a) Original data. (b) Rescaling corresponding to learned diagonal A. (c) Rescaling corresponding to full A.

Application to Clustering

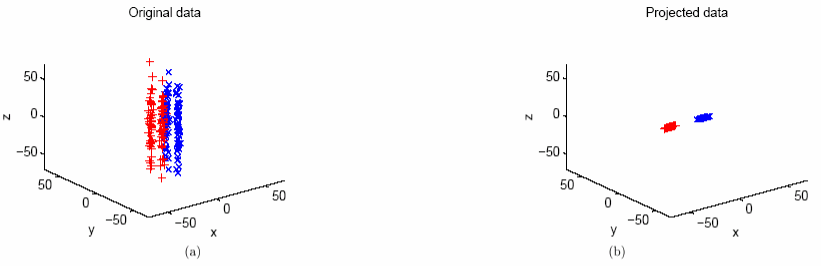
- Artificial Data I: a difficult two-class dataset



1. K-means: Accuracy = 0.4975
2. Constrained K-means: Accuracy = 0.5060
3. K-means + metric: Accuracy = 1
4. Constrained K-means + metric: Accuracy = 1

Application to Clustering

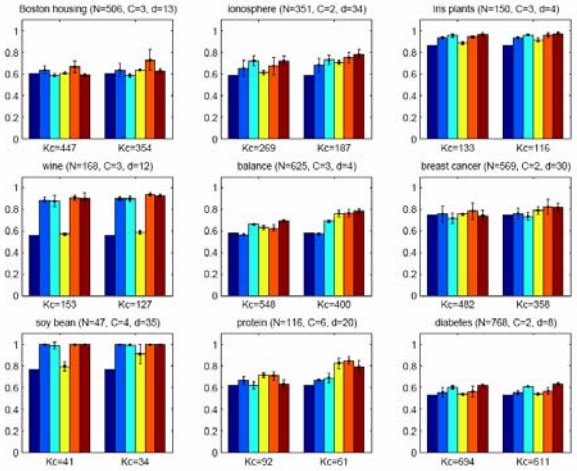
- Artificial Data II: two-class data with strong irrelevant feature



1. K-means: Accuracy = 0.4993
2. Constrained K-means: Accuracy = 0.5701
3. K-means + metric: Accuracy = 1
4. Constrained K-means + metric: Accuracy = 1

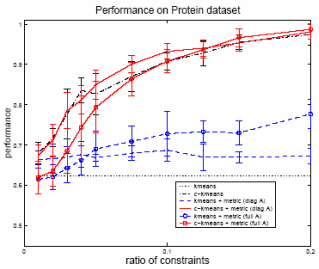
Application to Clustering

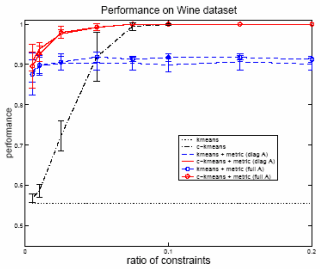
- 9 datasets from the UC Irvine repository



Accuracy vs. amount of side-information

- Two typical examples of how the quality of the clusters found increases with the amount of side-information.



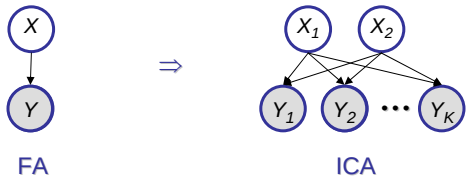


Take home message

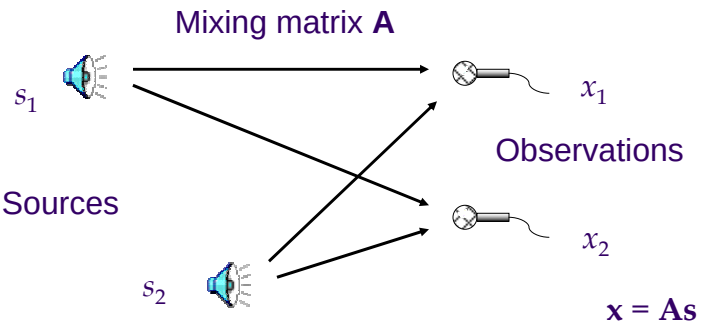
- Distance metric learning is an important problem in machine learning and data mining.
- A good distance metric can be learned from small amount of side-information in the form of similarity and dissimilarity constraints from data by solving a convex optimization problem.
- The learned distance metric can identify the most significant direction(s) in feature space that separates data well, effectively doing implicit Feature Selection.
- The learned distance metric can be used to improve clustering performance.

Additional Details:
Independent Components Analysis (ICA)

- ICA is similar to FA, except it assumes the latent source has non-Gaussian density.
- Hence ICA can extract higher order moments (not just second order).
- It is commonly used to solve blind source separation (cocktail party problem).

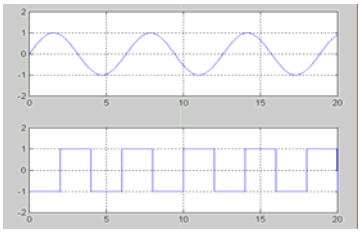


The simple "Cocktail Party" Problem

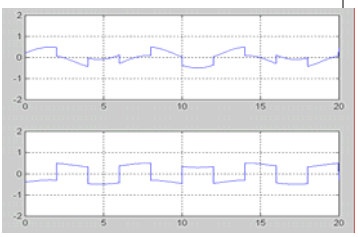


n sources, m=*n* observations

Motivation



Two Independent Sources

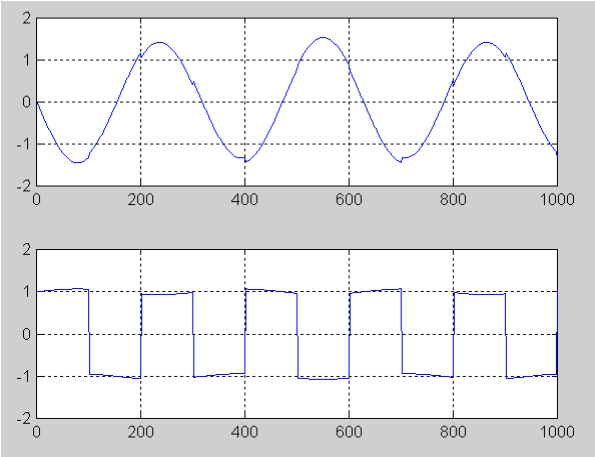


Mixture at two Mics

$$x_1(t) = a_{11}s_1(t) + a_{12}s_2(t)$$
$$x_2(t) = a_{21}s_1(t) + a_{22}s_2(t)$$

a_{ij} ... Depend on the distances of the microphones from the speakers

Motivation



Get the Independent Signals out of the Mixture

Blind Source Separation

- Suppose that there are *k* unknown independent sources

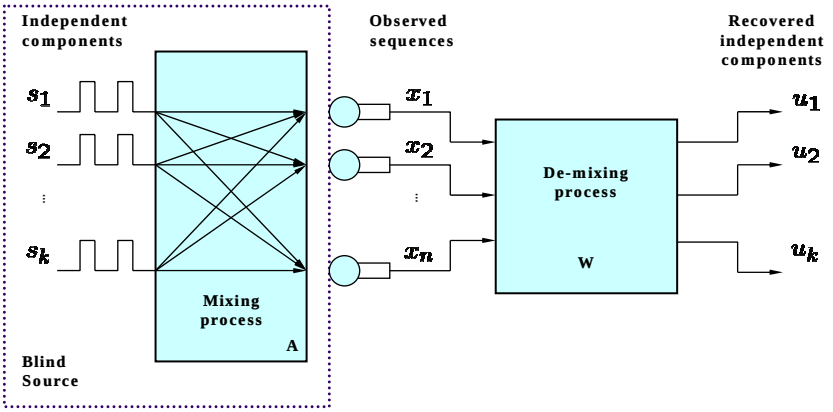
$$\mathbf{s}(t) = [s_1(t), \ldots, s_k(t)]^T \qquad \text{with} \qquad E[\mathbf{s}(t)] = \mathbf{1}$$

- A data vector x(*t*) is observed at each time point *t*, such that

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$$

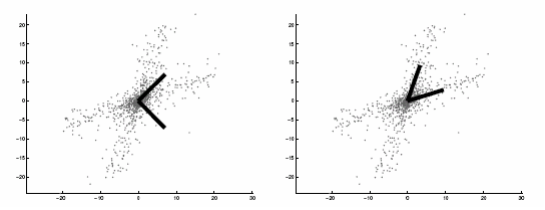
where $\mathbf{A}$ is a $n \times k$ full rank scalar matrix

Blind source separation



ICA versus PCA (and FA)

- Similarity
  - Feature extraction
  - Dimension reduction
- Difference
  - PCA uses up to second order moment of the data to produce uncorrelated components
  - ICA strives to generate components as independent as possible

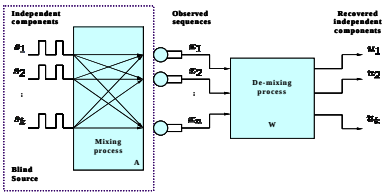


Problem formulation

- The goal of ICA is to find a linear mapping $\mathbf{W}$ such that the unmixed sequences $\mathbf{u}$

$$\mathbf{u}(t) = \mathbf{W}\mathbf{x}(t) = \mathbf{W}\mathbf{A}\mathbf{s}(t)$$

are maximally statistically independent



- Find some

$$\mathbf{V} = \mathbf{W}\mathbf{A} = \mathbf{P}\mathbf{C}$$

where $\mathbf{C}$ is a diagonal matrix and $\mathbf{P}$ is a permutation matrix.

Principle of ICA: Nongaussianity

- The fundamental restriction in ICA is that the independent components must be nongaussian for ICA to be possible.
- This is because gaussianity is invariant under orthogonal transformation and hence make the matrix $\mathbf{A}$ not identifiable for gaussian independent components.

Measures of nongaussianity (1)

- Kurtosis
  - $\text{Kurt}(y) = E\left\{y^4\right\} - 3\left(E\left\{y^2\right\}\right)^2$
  - Kurtosis can be very sensitive to outliers, when its value has to be estimate from a measured sample.
- Mutual information
- Negative Entropy

FastICA – Preprocessing

- Centering:
  - Make the x-s mean 0 variables
- Whitening
  - Transform the observed vector x linearly so that it has unit variance:

$$\mathbf{E}\left\{\tilde{\mathbf{x}}\tilde{\mathbf{x}}^T\right\} = \mathbf{I}$$

  - One can show that:

$$\tilde{\mathbf{x}} = \mathbf{E}\mathbf{D}^{-1/2}\mathbf{E}^T\mathbf{x} = \widetilde{\mathbf{A}}\mathbf{s}$$

where $\mathbf{E}\left\{\mathbf{x}\mathbf{x}^T\right\} = \mathbf{E}\mathbf{D}\mathbf{E}^T$

FastICA algorithm

- Initialize the weight matrix $\mathbf{W}$
- Iteration:
  - $$\mathbf{W}^{+} = \mathbf{W} + \mathrm{diag}(\alpha_i)[\mathrm{diag}(\beta_i) + \mathbf{E}\{\mathbf{g}(\mathbf{u}\mathbf{u}^{\mathsf{T}}\}]\mathbf{W}$$

    where

    $$\beta_i = -\mathbf{E}\{u_i g(u_i)\}, \alpha_i = -1/(\beta_i - \mathbf{E}\{g(u_i)'\})$$

  - Repeat until convergence $\mathbf{W}^{\infty}$
- The ICAs are the components of $\mathbf{W}^{\infty}\mathbf{x}(t)$

Summary

- There has been a wide discussion about the application of Independence Component Analysis (ICA) in Signal Processing, Neural Computation and Finance.
- First introduced as a novel tool to separate blind sources in a mixed signal.
- The Basic idea of ICA is to reconstruct from observation sequences the hypothesized independent original sequences.