CSC2515 Fall 2015
Introduction to Machine Learning

Lecture 7: Continuous Latent Variable Models

All lecture slides will be available as .pdf on the course website:
http://www.cs.toronto.edu/~urtasun/courses/CSC2515/CSC2515_Winter15.html

Many of the figures are provided by Chris Bishop
from his textbook: "Pattern Recognition and Machine Learning"

Mixture models and Distributed Representations

- One problem with mixture models: each observation assumed to come from one of K prototypes
- Can think of as graphical model with K binary hidden variables
- But constraint that only one active (responsibilities sum to one) limits representational power
- Alternative: Distributed representation, with several latent variables relevant to each observation
- Can be several binary/discrete variables, or continuous

Example: continuous underlying variables

- What are the intrinsic latent dimensions in these two datasets?

- How can we find these dimensions from the data?

Principal Components Analysis

PCA: most popular instance of second main class of unsupervised learning methods, projection methods, aka dimensionality-reduction methods

Aim: find small number of "directions" in input space that explain correlations in input data; re-represent data by projecting along those directions

Data is assumed to be continuous: linear relationship between data and learned representation

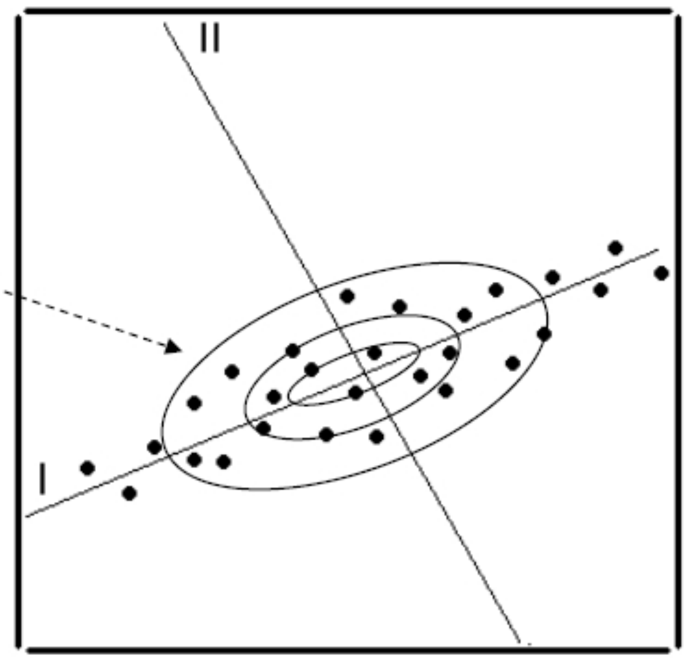
PCA: Common tool

- Handles high-dimensional data – if data has thousands of dimensions, can be difficult for classifier to deal with
- Often can be described by much lower dimensional representation

- Useful for
  - Visualization
  - Preprocessing
  - Modeling – prior for new data
  - Compression

PCA: Intuition

- Assume start with N data vectors, of dimensionality D
- Aim to reduce dimensionality – linearly project (multiply by matrix) to much lower dimensional space, M << D
- Search for orthogonal directions in space w/ highest variance – project data onto this subspace
- Structure of data vectors is encoded in sample covariance



Finding principal components

To find the principal component directions, we center the data (subtract the sample mean from each variable)

Calculate the empirical covariance matrix:

$$C = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$$

Find the M eigenvectors with largest eigenvalues of C – these are the principal components

Assemble these eigenvectors into a DxM matrix U

We can now express *D*-dimensional vectors x by projecting them to M-dimensional z: $z = U^T x$

Standard PCA

- Algorithm: to find *M* components underlying *D*-dimensional data
  - select the top *M* eigenvectors of ***C*** (data covariance matrix):

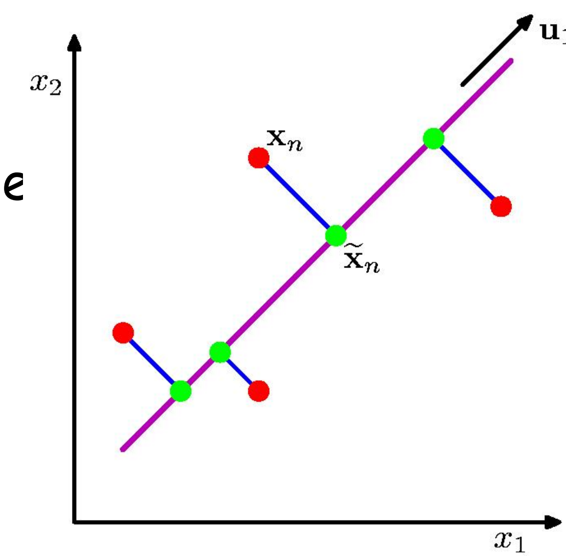
$$C = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T = U \Lambda U^T \qquad C \approx U_{1:M}\Lambda_{1:M}U_{1:M}{}^T$$

  - U: orthogonal, columns = unit-length eigenvectors ($U^T U = UU^T = 1$)
  - Λ: eigenvalue = variance in direction of eigenvector

$$\Lambda_{1:3} = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_1 \end{bmatrix}$$

  - project each input vector **x** into this subspace

$$z_{ij} = \mathbf{u}_j^T \mathbf{x}_i \qquad \mathbf{z}_i = U_{1:M}^T \mathbf{x}_i$$



- Two views/derivations:
  - Maximize variance (scatter of green points)
  - Minimize error (red-green distance per datapoint)

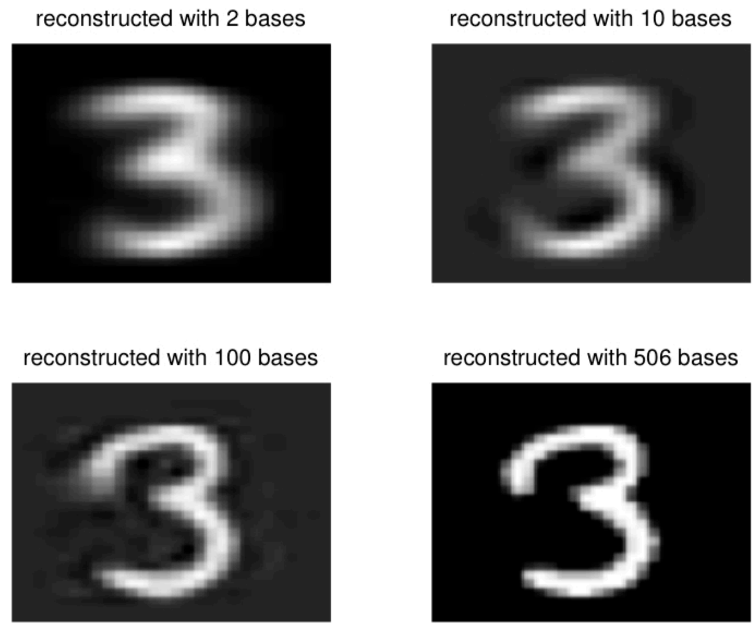
Applying PCA to faces

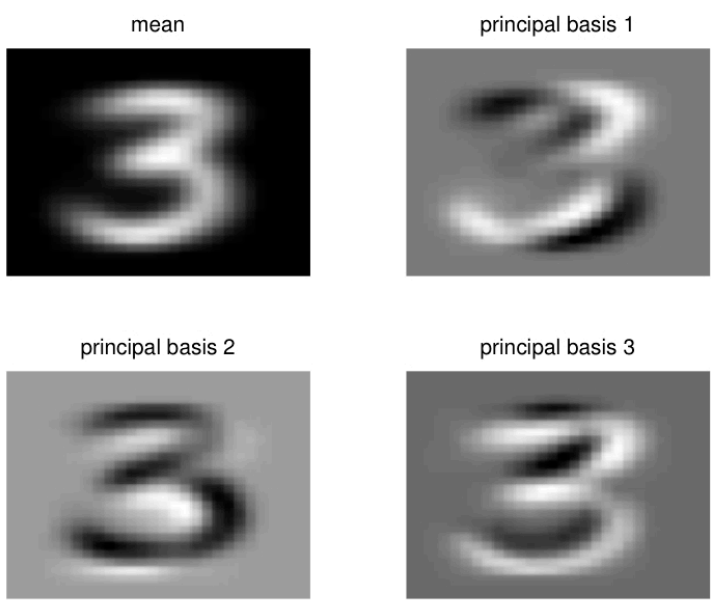
- Run PCA on 2429 19x19 grayscale images (CBCL data)
- Compresses the data: can get good reconstructions with only 3 components

- PCA for pre-processing: can apply classifier to latent representation -- PPCA w/ 3 components obtains 79% accuracy on face/non-face discrimination in test data vs. 76.8% for m.o.G with 84 states
- Can also be good for visualization

Applying PCA to faces: Learned basis

Applying PCA to digits





PCA: Details & Derivations

PCA can be viewed as finding a low-dimensional hyperplane on which to project the data

Ideally this projection will preserve information in the data while reducing dimensionality

Idea: first basis vector of the hyperplane points in the direction of maximum variance of data

The second basis vector points in the direction of maximum variance given that it is orthogonal to first

Each subsequent basis vector, or principal component, is the direction of maximum variance that is orthogonal to all previous principal components

Standard PCA: Variance Maximization

Start with one dimension

Aim: maximize projected variance: find $\mathbf{w}_1$ that maximizes

$$\mathrm{var}(z_1) = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{w}_1^T\mathbf{x}_i - \mathbf{w}_1^T\bar{\mathbf{x}})^2 = \mathbf{w}_1^T C\mathbf{w}_1$$

where C is the data covariance, sample mean $\bar{\mathbf{x}} = \sum_{i=1}^{N}\mathbf{x}_i / N$

Constrain $||\mathbf{w}_1||=1$, via Lagrange multipliers – find that optimal $\mathbf{w}_1 = \mathbf{u}_1$, the first eigenvector of C (eigenvector with maximal eigenvalue), and $\mathbf{w}_1^T C\mathbf{w}_1 = \lambda_1$

Can extend to multiple dimensions – maximize |Cov(Z)|, find that optimal $\mathbf{W}_{1:M} = \mathbf{U}_{1:M}$

$$\mathrm{cov}(\mathbf{Z}) = \frac{1}{N}\sum_{i=1}^{N}\mathbf{z}_i\mathbf{z}_i^T = \frac{1}{N}\sum_{i=1}^{N}U_{1:M}^T\mathbf{x}_i\mathbf{x}_i^T U_{1:M}$$

$$= U_{1:M}^T\left(\frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i^T\right)U_{1:M} = U_{1:M}^T U\Lambda U^T U_{1:M} = \Lambda_{1:M}$$

Standard PCA: Extending to higher dimensions

- Can consider forming components sequentially: find variance-maximizing directions orthogonal to previous ones
- Equivalent to Gaussian approximation to data
- Think of Gaussian as football (hyperellipsoid)
  - Mean is center of football
  - Eigenvectors of covariance matrix are axes of football
  - Eigenvalues are lengths of axes
- PCA can be thought of as fitting the football to the data: maximize volume of data projections in M-dimensional subspace
- Alternative formulation: minimize error, equivalent to minimizing average distance from datapoint to its reconstruction from its projection in subspace

Standard PCA: Error minimization

- Data points represented by projection onto *M*-dimensional subspace, plus some distortion:
- Objective: minimize distortion w.r.t. $\mathbf{U}_1$ (*reconstruction error* of $\mathbf{x}_n$)

$$J = \frac{1}{N}\sum_{n=1}^{N} ||\mathbf{x}_n - \tilde{\mathbf{x}}_n^T||^2$$

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{M} z_{ni}\mathbf{u}_i + \sum_{i=M+1}^{D} b_i\mathbf{u}_i$$

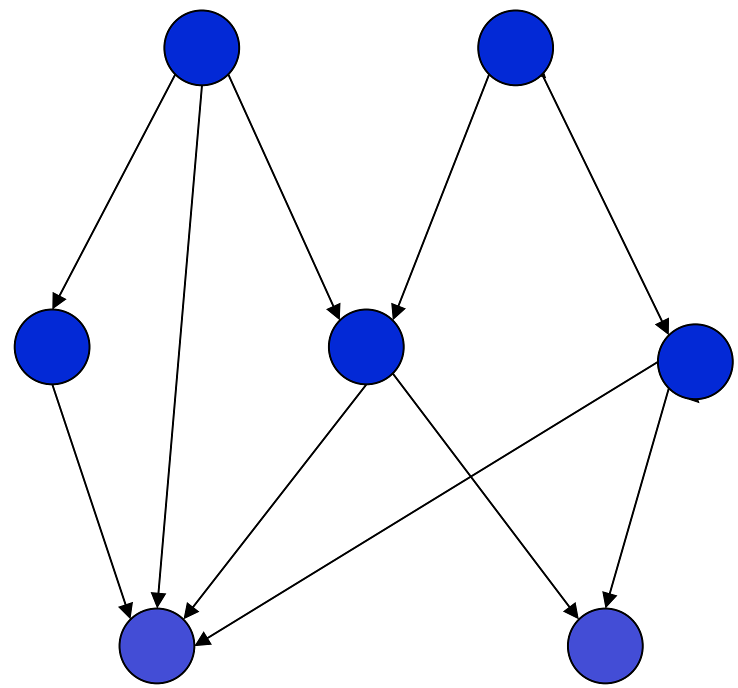
$$\begin{aligned} z_{nj} &= \mathbf{x}_n^T\mathbf{u}_j \\ b_j &= \bar{\mathbf{x}}^T\mathbf{u}_j \end{aligned}$$

$$J = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=M+1}^{D} b_i(\mathbf{x}_n^T\mathbf{u}_i - \bar{\mathbf{x}}^T\mathbf{u}_i)^2 = \sum_{i=M+1}^{D}\mathbf{u}_i^T\mathbf{S}\mathbf{u}_i$$

- The objective is minimized when the *D-M* components are eigenvectors of **S** with *lowest* eigenvalues ! → same result

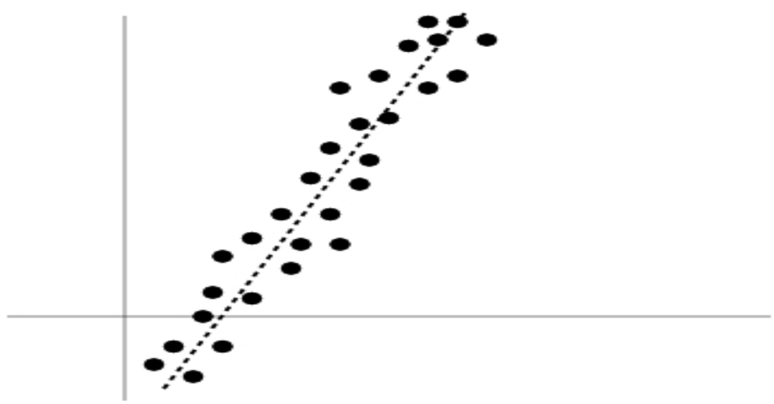
Return to Graphical Model View

- Last time we discussed latent variable models
- The latent variables in mixture models are multinomials (referring to cluster identity).
- Today we've been considering continuous latent variables



Dimensionality Reduction vs. Clustering

- Training continuous latent variable models often called *dimensionality reduction*, since there are typically many fewer latent dimensions
- Examples: Principal Components Analysis, Factor Analysis, Independent Components Analysis
- Continuous causes often more efficient at representing information than discrete
- For example, if there are two factors, with about 256 settings each, we can describe the latent causes with two 8-bit numbers
- If we try to cluster the data, we need $2^{16} \sim= 10^5$ numbers



Generative View

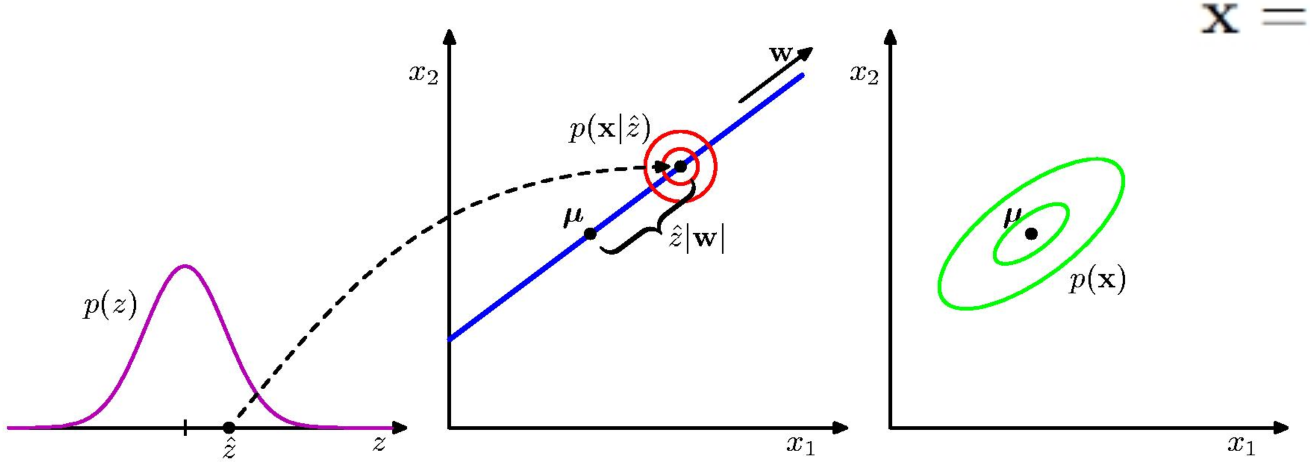
- Each data example generated by first selecting a point from a distribution in the latent space, then generating a point from the conditional distribution in the input space
- Mixture models have multinomial latents
- For continuous latents, and inputs, now looking at simple models: Gaussian distributions in both latent and data space, linear relationship betwixt
- This view underlies Probabilistic PCA, Factor Analysis

Probabilistic PCA

- Probabilistic, generative view of data
- Assumptions:
  - underlying latent variable has a Gaussian distribution
  - linear relationship between latent and observed variables
  - isotropic Gaussian noise in observed dimensions

$$
\begin{aligned}
p(\mathbf{z}) &= \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}) \\
p(\mathbf{x}|\mathbf{z}) &= \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \mu, \sigma^2\mathbf{I})
\end{aligned}
$$

$$\mathbf{x} = \mathbf{W}\mathbf{z} + \mu + \epsilon$$



Probabilistic PCA: Marginal data density

- Columns of $\mathbf{W}$ are the *principal components*, σ^2 is *sensor noise*
- Product of Gaussians is Gaussian: the joint p($\mathbf{z}$,$\mathbf{x}$), the marginal data distribution p($\mathbf{x}$) and the posterior p($\mathbf{z}$|$\mathbf{x}$) are also Gaussian
- Marginal data density (predictive distribution):

$$p(\mathbf{x}) = \int_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\mathbf{z})d\mathbf{z} = \mathcal{N}(\mathbf{x}|\mu, \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I})$$

- Can derive by completing square in exponent, or by just computing mean and covariance given that it is Gaussian:

$$\begin{aligned}
E[\mathrm{x}] &= E[\mu + \mathbf{W}\mathbf{z} + \epsilon] = \mu + \mathbf{W}E[\mathrm{z}] + E[\epsilon] \\
&= \mu + \mathbf{W}0 + 0 = \mu \\
\mathbf{C} &= Cov[\mathrm{x}] = E[(\mathbf{z} - \mu)(\mathbf{z} - \mu)^T] \\
&= E[(\mu + \mathbf{W}\mathbf{z} + \epsilon - \mu)(\mu + \mathbf{W}\mathbf{z} + \epsilon - \mu)^T] \\
&= E[(\mathbf{W}\mathbf{z} + \epsilon)(\mathbf{W}\mathbf{z} + \epsilon)^T] \\
&= \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I}
\end{aligned}$$

Probabilistic PCA: Joint distribution

- Joint density for PPCA (**x** is D-dim., **z** is M-dim):

$$p\left(\begin{bmatrix}\mathbf{z}\\ \mathbf{x}\end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{z}\\ \mathbf{x}\end{bmatrix} \,\middle|\, \begin{bmatrix}0\\ \mu\end{bmatrix}, \begin{bmatrix} I & \mathbf{W}^\top \\ \mathbf{W} & \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}\end{bmatrix}\right)$$

  – where cross-covariance terms from:

$$\begin{aligned} Cov[\mathbf{z},\mathbf{x}] &= E[(\mathbf{z}-0)(\mathbf{x}-\mu)^T] = E[\mathbf{z}(\mu + \mathbf{W}\mathbf{z} + \epsilon - \mu)^T] \\ &= E[\mathbf{z}(\mathbf{W}\mathbf{z}+\epsilon)^T] = \mathbf{W}^T \end{aligned}$$

- Note that evaluating predictive distribution involves inverting **C**: reduce O(D^3) to O(M^3) by applying *matrix inversion lemma*:

$$\mathbf{C}^{-1} = \sigma^{-1}\mathbf{I} - \sigma^{-2}\mathbf{W}(\mathbf{W}^T\mathbf{W} + \sigma^2\mathbf{I})^{-1}\mathbf{W}^T$$

Probabilistic PCA: Posterior distribution

- Inference in PPCA produces posterior distribution over latent **z**
- Derive by applying Gaussian conditioning formulas (see 2.3 in book) to joint distribution

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

$$p(\mathbf{x}_1) = \mathcal{N}(\mu_1, \Sigma_{11})$$

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

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

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

$$\begin{aligned} p(\mathbf{z}|\mathbf{x}) &= \mathcal{N}(\mathbf{z}|\mathbf{m}, \mathbf{V}) \\ \mathbf{m} &= \mathbf{W}^T(\mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I})^{-1}(\mathbf{x} - \mu) \\ \mathbf{V} &= \mathbf{I} - \mathbf{W}^T(\mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I})^{-1}\mathbf{W} \end{aligned}$$

- Mean of inferred **z** is projection of centered **x** – linear operation
- Posterior variance does not depend on the input **x** at all!

Standard PCA: Zero-noise limit of PPCA

- Can derive standard PCA as limit of Probabilistic PCA (PPCA) as $\sigma^2 \to 0$.
- ML parameters $\mathbf{W}^*$ are the same
- Inference is easier: orthogonal projection

$$\lim_{\sigma^2 \to 0} \mathbf{W}^T(\mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{W}^T)^{-1} = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T$$

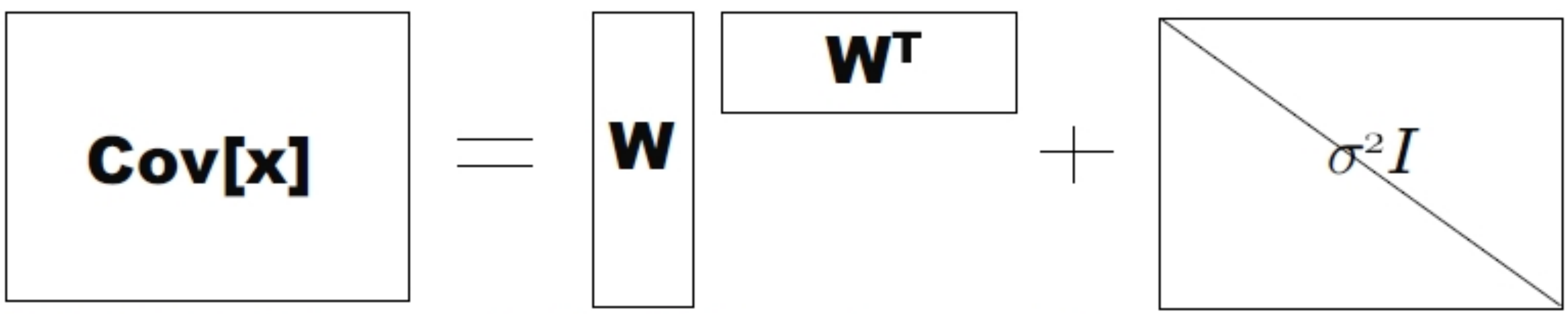
- Posterior covariance is zero

Probabilistic PCA: Constrained covariance

- Marginal density for PPCA (**x** is D-dim., **z** is M-dim):

$$p(\mathbf{x}|\theta) = \mathcal{N}(\mathbf{x}|\mu, \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I})$$

  - where $\theta = \mathbf{W}, \mu, \sigma$
- Effective covariance is low-rank outer product of two long skinny matrices plus a constant diagonal matrix



- So PPCA is just a constrained Gaussian model:
  - Standard Gaussian has D + D(D+1)/2 effective parameters
  - Diagonal-covariance Gaussian has D+D, but cannot capture correlations
  - PPCA: DM + 1 – M(M-1)/2, can represent M most significant correlations

Probabilistic PCA: Maximizing likelihood

$$
\begin{aligned}
L(\theta; \mathbf{X}) &= \log p(\mathbf{X}|\theta) = \sum_n \log p(\mathbf{x}_n|\theta) \\
&= -\frac{N}{2}\log|\mathbf{C}| - \frac{1}{2}\sum_n (\mathbf{x}_n - \mu)\mathbf{C}^{-1}(\mathbf{x}_n - \mu)^T \\
&= -\frac{N}{2}\log|\mathbf{C}| - \frac{1}{2}Tr[\mathbf{C}^{-1}\sum_n (\mathbf{x}_n - \mu)(\mathbf{x}_n - \mu)^T] \\
&= -\frac{N}{2}\log|\mathbf{C}| - \frac{1}{2}Tr[\mathbf{C}^{-1}\mathbf{S}]
\end{aligned}
$$

Fit parameters ($\theta = \mathbf{W}, \mu, \sigma$) to max likelihood: make model covariance match observed covariance; distance is trace of ratio

Sufficient statistics: mean $\mu = (1/N)\sum_n \mathbf{x}_n$ and sample covariance **S**

Can solve for ML params directly: k^{th} column of **W** is the M^{th} largest eigenvalue of **S** times the associated eigenvector; σ^2 is the sum of all eigenvalues less than M^{th} one

Probabilistic PCA: EM

Rather than solving directly, can apply EM

Need complete-data log likelihood

$$\log p(\mathbf{X}, \mathbf{Z} | \mu, \mathbf{W}, \sigma^2) = \sum_n [\log p(\mathbf{x}_n | \mathbf{z}_n) + \log p(\mathbf{z}_n)]$$

E step: compute expectation of complete log likelihood with respect to posterior of latent variables z, using current parameters – can derive $E[\mathbf{z}_n]$ and $E[\mathbf{z}_n \mathbf{z}_n^T]$ from posterior $p(\mathbf{z}|\mathbf{x})$

M step: maximize with respect to parameters W and σ^2

Iterative solution, updating parameters given current expectations, expectations give current parameters

Nice property – avoids direct $O(ND^2)$ construction of covariance matrix, instead involves sums over data cases: $O(NDM)$; can be implemented online, without storing data

Probabilistic PCA: Why bother?

- Seems like a lot of formulas, algebra to get to similar model to standard PCA, but…
- Leads to understanding of underlying data model, assumptions (e.g., vs. standard Gaussian, other constrained forms)
- Derive EM version of inference/learning: more efficient
- Can understand other models as generalizations, modifications
- More readily extend to mixtures of PPCA models
- Principled method of handling missing values in data
- Can generate samples from data distribution

Factor Analysis

Can be viewed as generalization of PPCA

Historical aside – controversial method, based on attempts to interpret factors: e.g., analysis of IQ data identified factors related to race

Assumptions:

- underlying latent variable has a Gaussian distribution
- linear relationship between latent and observed variables
- *diagonal* Gaussian noise in data dimensions

$$
\begin{aligned}
p(\mathbf{z}) &= \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}) \\
p(\mathbf{x}|\mathbf{z}) &= \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \mu, \Psi)
\end{aligned}
$$

W: *factor loading matrix* (*D* x *M*)

Ψ : data covariance (diagonal, or axis-aligned; vs. PCA's spherical)

Factor Analysis: Distributions

- As in PPCA, the joint p(**z**,**x**), the marginal data distribution p(**x**) and the posterior p(**z**|**x**) are also Gaussian
- Marginal data density (predictive distribution):

$$p(\mathbf{x}) = \int_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\mathbf{z})d\mathbf{z} = \mathcal{N}(\mathbf{x}|\mu, \mathbf{W}\mathbf{W}^T + \mathbf{\Psi})$$

- Joint density:

$$p\left(\begin{bmatrix}\mathbf{z}\\ \mathbf{x}\end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{z}\\ \mathbf{x}\end{bmatrix} \mid \begin{bmatrix}0\\ \mu\end{bmatrix}, \begin{bmatrix} I & \mathbf{W}^\top \\ \mathbf{W} & \mathbf{W}\mathbf{W}^\top + \Psi\end{bmatrix}\right)$$

- Posterior, derived via Gaussian conditioning

$$\begin{aligned} p(\mathbf{z}|\mathbf{x}) &= \mathcal{N}(\mathbf{z}|\mathbf{m}, \mathbf{V}) \\ \mathbf{m} &= \mathbf{W}^T(\mathbf{W}\mathbf{W}^T + \mathbf{\Psi})^{-1}(\mathbf{x} - \mu) \\ \mathbf{V} &= \mathbf{I} - \mathbf{W}^T(\mathbf{W}\mathbf{W}^T + \mathbf{\Psi})^{-1}\mathbf{W} \end{aligned}$$

Factor Analysis: Optimization

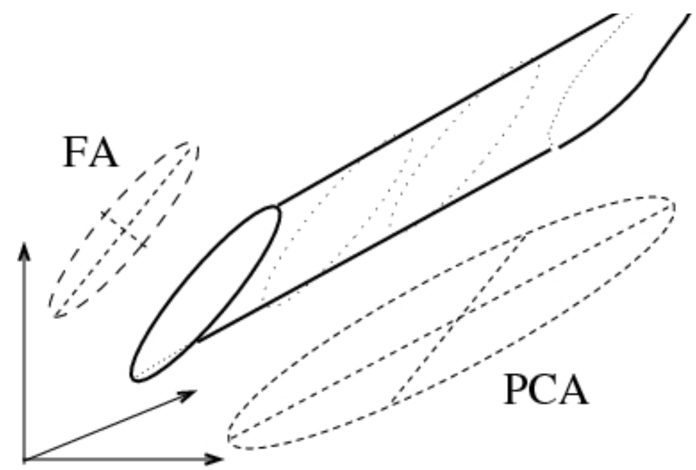
- Parameters are coupled, making it impossible to solve for ML parameters directly, unlike PCA
- Must use EM, or other nonlinear optimization
- E step: compute posterior p(**z**|**x**) – use matrix inversion to convert $D \times D$ matrix inversions to $M \times M$
- M step: take derivatives of expected complete log likelihood with respect to parameters

Factor Analysis vs. PCA: Rotations

- In PPCA, the data can be rotated without changing anything: multiply data by matrix **Q,** obtain same fit to data

$$
\begin{aligned}
\mu &\leftarrow \mathbf{Q}\mu \\
\mathbf{W} &\leftarrow \mathbf{QW} \\
\Psi &\leftarrow \Psi
\end{aligned}
$$

- But the scale is important
- PCA looks for directions of large variance, so it will grab large noise directions



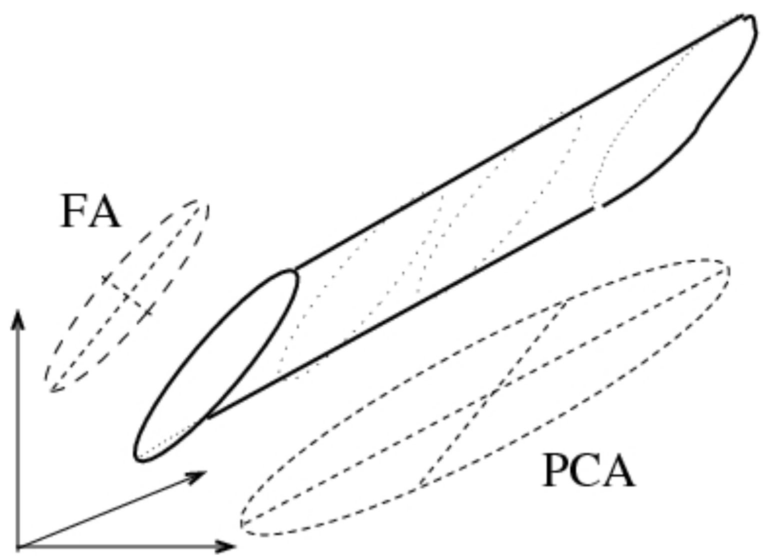
Factor Analysis vs. PCA: Scale

In FA, the data can be re-scaled without changing anything

Multiply x_i by a_i:

$$\begin{aligned} \mu_i &\leftarrow \alpha_i \mu_i \\ \mathbf{W}_{ij} &\leftarrow \alpha_i \mathbf{W}_{ij} \\ \Psi_i &\leftarrow \alpha_i^2 \Psi_i \end{aligned}$$

But rotation in data space is important

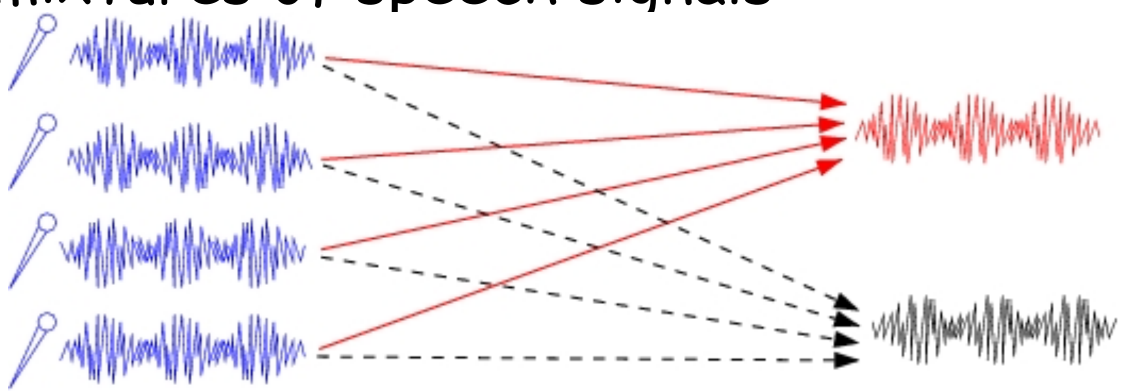


Factor Analysis : Identifiability

- Factors in FA are *non-identifiable:* not guaranteed to find same set of parameters – not just local minimum but invariance
- Rotate **W** by any unitary **Q** and model stays the same – **W** only appears in model as outer product $\mathbf{W}\mathbf{W}^T$
- Replace **W** with **WQ**: $(\mathbf{WQ})(\mathbf{WQ})^T = \mathbf{W}(\mathbf{Q}\,\mathbf{Q}^T)\,\mathbf{W}^T = \mathbf{W}\mathbf{W}^T$
- So no single best setting of parameters
- Degeneracy makes unique interpretation of learned factors impossible

Independent Components Analysis (ICA)

- ICA is another continuous latent variable model, but it has a *non-Gaussian* and *factorized* prior on the latent variables
- Good in situations where most of the factors are small most of the time, do not interact with each other
- Example: mixtures of speech signals



- Learning problem same as before: find weights from factors to observations, infer the unknown factor values for given input
- ICA: factors are called "sources", learning is "unmixing"

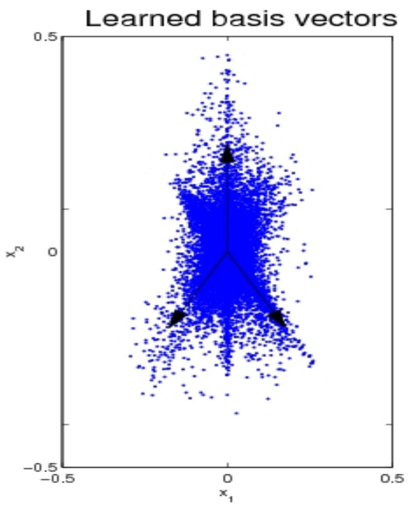
ICA Intuition

Since latent variables assumed to be independent, trying to find linear transformation of data that recovers independent causes

Avoid degeneracies in Gaussian latent variable models: assume non-Gaussian prior distribution for latents (sources)

Often we use *heavy-tailed* source priors, e.g.,

$$p(z_j) = \frac{1}{\pi \cosh(z_j)} = \frac{1}{\pi(\exp(z_j) + \exp(-z_j))}$$



ICA Details

- Simplest form of ICA has as many outputs as sources (square) and no sensor noise on the outputs:

$$\begin{aligned} p(\mathbf{z}) &= \prod_k p(z_k) \\ \mathbf{x} &= \mathbf{V}\mathbf{z} \end{aligned}$$

- Learning in this case can be done with gradient descent (plus some "covariant" tricks to make updates faster and more stable)
- If keep $\mathbf{V}$ square, and assume isotropic Gaussian noise on the outputs, there is a simple EM algorithm
- Much more complex cases have been studied also: non-square, time delays, etc.

Summary of Latent Factor Methods

- Aim to find low-dimensional subspace that captures essential properties of data
- Assumes that even though data is high-dimensional, there are some small number of continuous underlying (latent) factors, whose variability accounts for variations in observations
- Example: latent factors underlying images are lighting, object identities, pose, etc.
- Different methods vary in terms of their assumptions about these factors, and how the observations relate to the factors