ISTA 421 + INFO 521
Introduction to Machine Learning

Lecture 29:
Projection
Principle Components Analysis

Clay Morrison

claytonm@email.arizona.edu

Harvill 437A

Phone 621-6609

6 December 2016

Finishing the Gaussian Mixture Model

$$p(\mathbf{X}|\Delta,\boldsymbol{\pi}) = \prod_{n=1}^{N}\sum_{k=1}^{K}\pi_k p(\mathbf{x}_n|\Delta_k) \quad \geq \sum_{n=1}^{N}\mathbf{E}_{q_{nk}}\left\{\log\frac{\pi_k p(\mathbf{x}_n|\boldsymbol{\Delta}_k)}{q_{nk}}\right\}$$

Maximizing the Mixture Model Likelihood

- Now we'll look at an instance of the **EM** algorithm for Gaussian mixtures: a **G**aussian **M**ixture **M**odel.
- We'll want to do maximization, so easier to work with the logarithm of the likelihood

$$L = \log p(\mathbf{X}|\Delta,\boldsymbol{\pi}) = \sum_{n=1}^{N}\log\sum_{k=1}^{K}\pi_k p(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)$$

- **Immediate problem!**: the summation inside the log makes finding the optimal parameters challenging

  We want to take partial derivatives w.r.t. $\boldsymbol{\mu}_k,\ \boldsymbol{\Sigma}_k,\ \boldsymbol{\pi}_k$

- **Trick**: derive a lower-bound on *L* and maximize ***that***

Expectation Maximization (EM) for GMM

$$\mathcal{B} = \sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk} \log \pi_k + \sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk} \log p(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) - \sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk} \log q_{nk}$$

The final expressions:

The mean value of q_{nk} for a particular cluster k :
Average of all posterior probabilities of belonging to cluster k
i.e., the expected proportion of the data belonging to cluster k

$$\pi_k = \frac{1}{N}\sum_{n=1}^{N} q_{nk}$$

Average of the data objects weighted by q_{nk} :
When all cluster membership is such that posterior prob's are 0 or 1, then this is just the proportion of the data assigned to component k

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} q_{nk}\mathbf{x}_n}{\sum_{n=1}^{N} q_{nk}} \qquad \boldsymbol{\mu}_k = \frac{\sum_n z_{nk}\mathbf{x}_n}{\sum_n z_{nk}}$$

weighted covariance :

$$\boldsymbol{\Sigma}_k = \frac{\sum_{n=1}^{N} q_{nk}(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^\mathsf{T}}{\sum_{n=1}^{N} q_{nk}}$$

Looks like Bayes' Rule! :

$$q_{nk} = \frac{\pi_k p(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j p(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

Posterior probability of object n belonging to cluster k

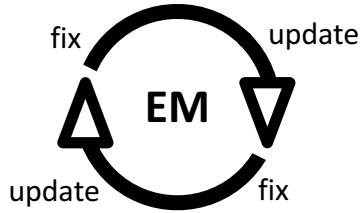
$$p(z_{nk} = 1|\mathbf{x}_n, \boldsymbol{\pi}, \Delta) = \frac{p(z_{nk} = 1|\pi_k)p(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} p(z_{nj} = 1|\pi_j)p(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} = q_{nk}$$

Cluster membership probability

q_{nk}

E step

(expected value of unknown assignments z_{nk})



All about the model

$\boldsymbol{\mu}_k,\ \boldsymbol{\Sigma}_k,\ \boldsymbol{\pi}$

M step

(maximizing w.r.t cluster membership)

Running EM with a GMM

- Initialize the parameters:
  - Mixture paramters: random means and covariances
  - Mixture priors: could choose uniform $\pi_k = 1/K$

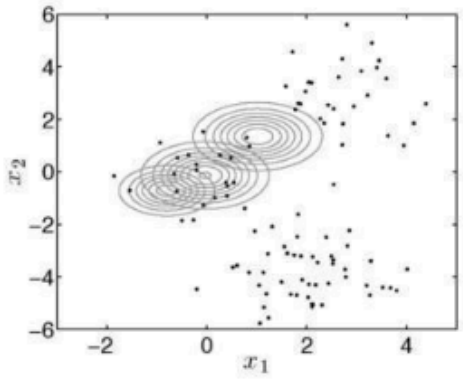
To find cluster membership:

q_{nk} = posterior prob of n belonging to k

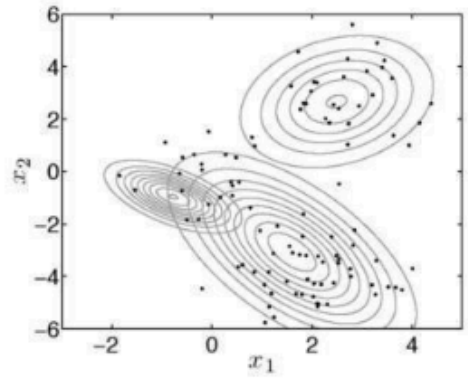
Hard assignment: for each indiv. n, choose the k with the highest q_{nk}

Or, consider the distribution – reveals interesting relationships of individual to more than one cluster; e.g.,

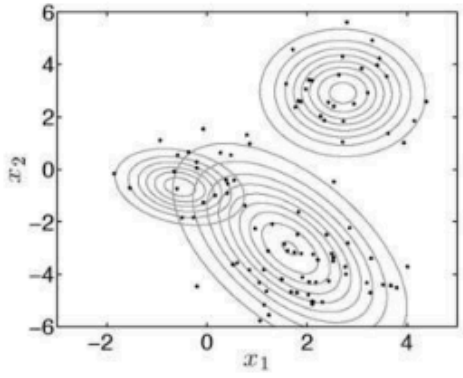
$$q_{n1} = 0.53, \quad q_{n2} = 0.45, \quad q_{n3} = 0.02$$



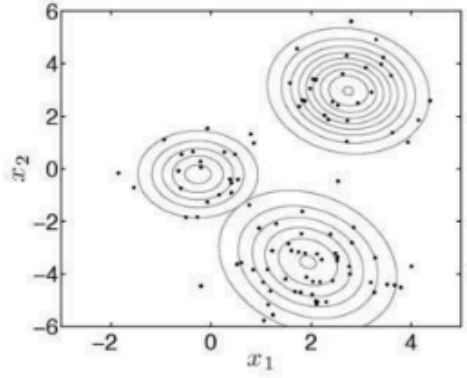
(a) The three randomly initialised Gaussian mixture components



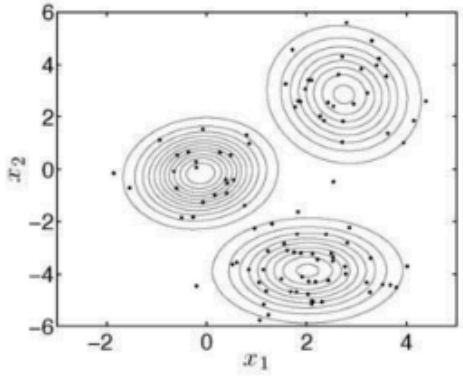
(b) The three components after one iteration of the EM algorithm



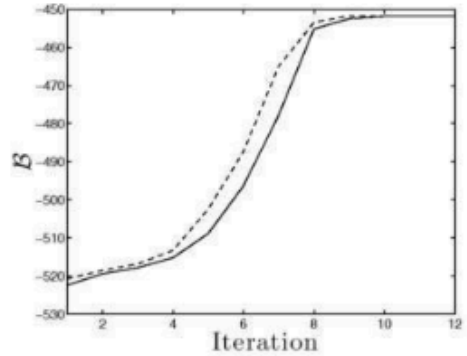
(c) The three components after five iterations of the EM algorithm



(d) The three components after seven iterations of the EM algorithm

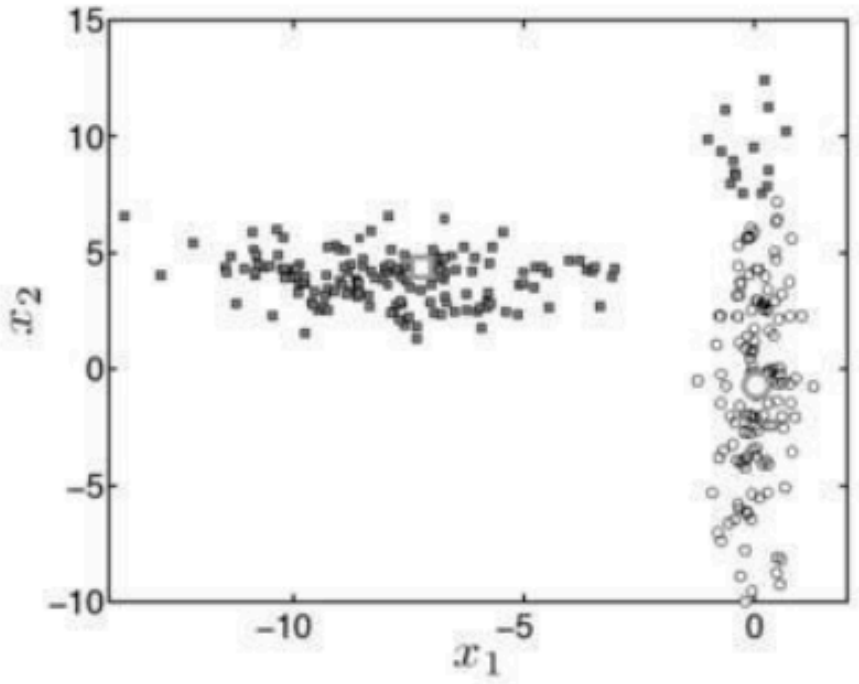


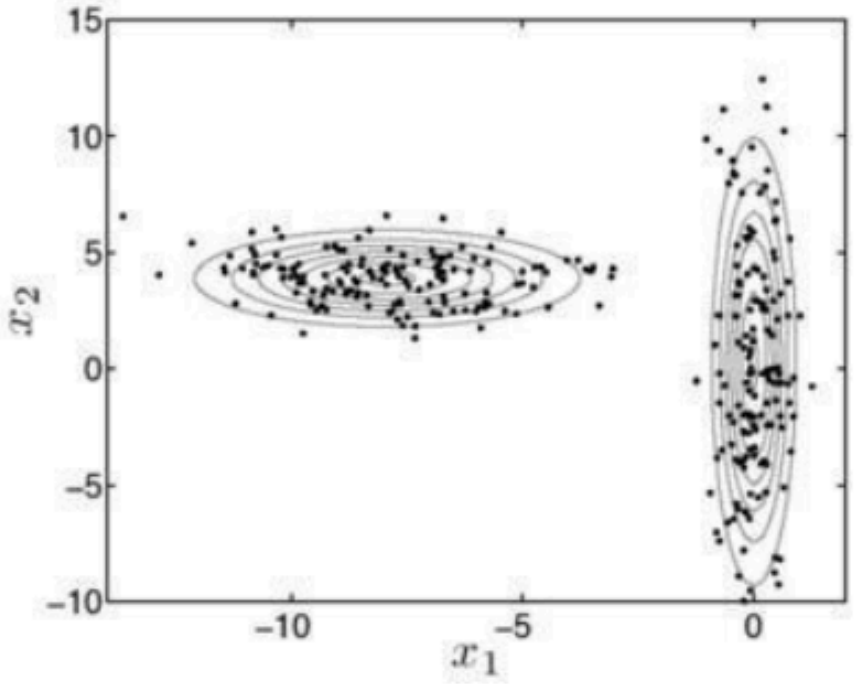
(e) The three components at convergence of the EM algorithm



(f) The evolution of the bound $\mathcal{B}$ (solid line, Equation 6.8) and log likelihood L (dashed line, Equation 6.5)

GMM does well on problem hard for *K*-means

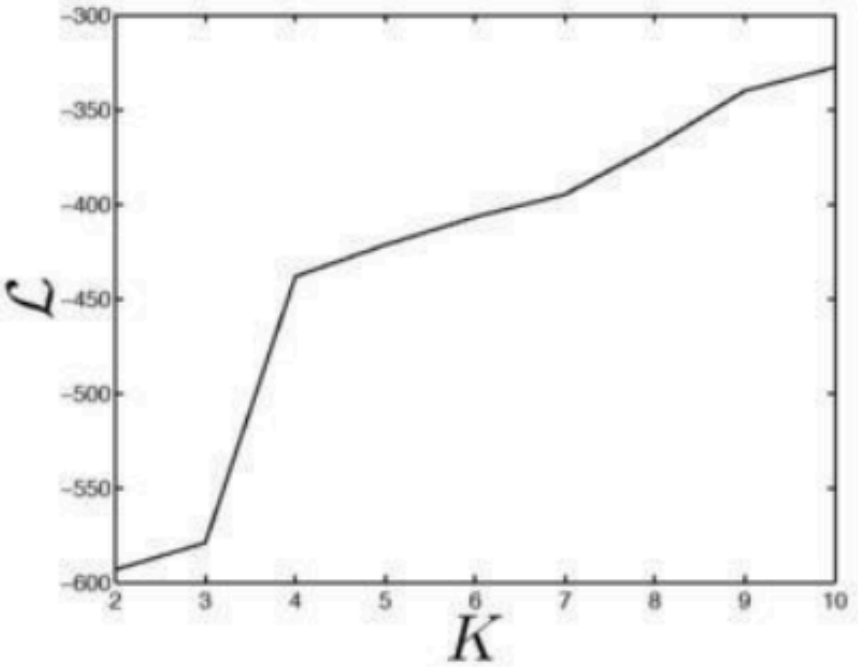




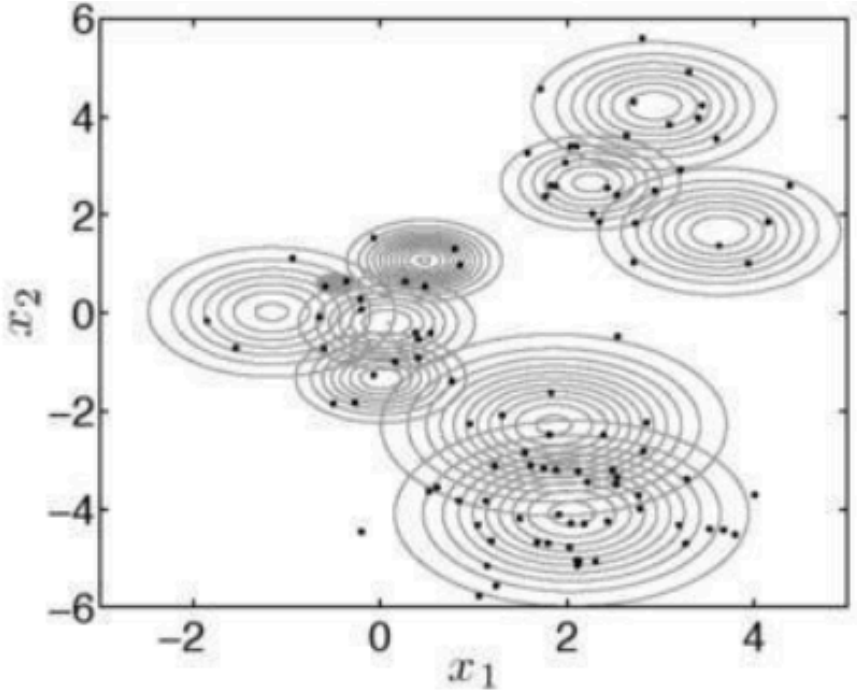
GMM with K=2

Choosing K

- Similar to *K*-means, where we can't just choose the *K* that minimizes the total distances (*K*=*N* → *D*=0), we can't just choose the *K* that maximizes the log likelihood *L* (or bound *B*); it increases with more mixtures:



(a) The increase in model likelihood as the number of components increases

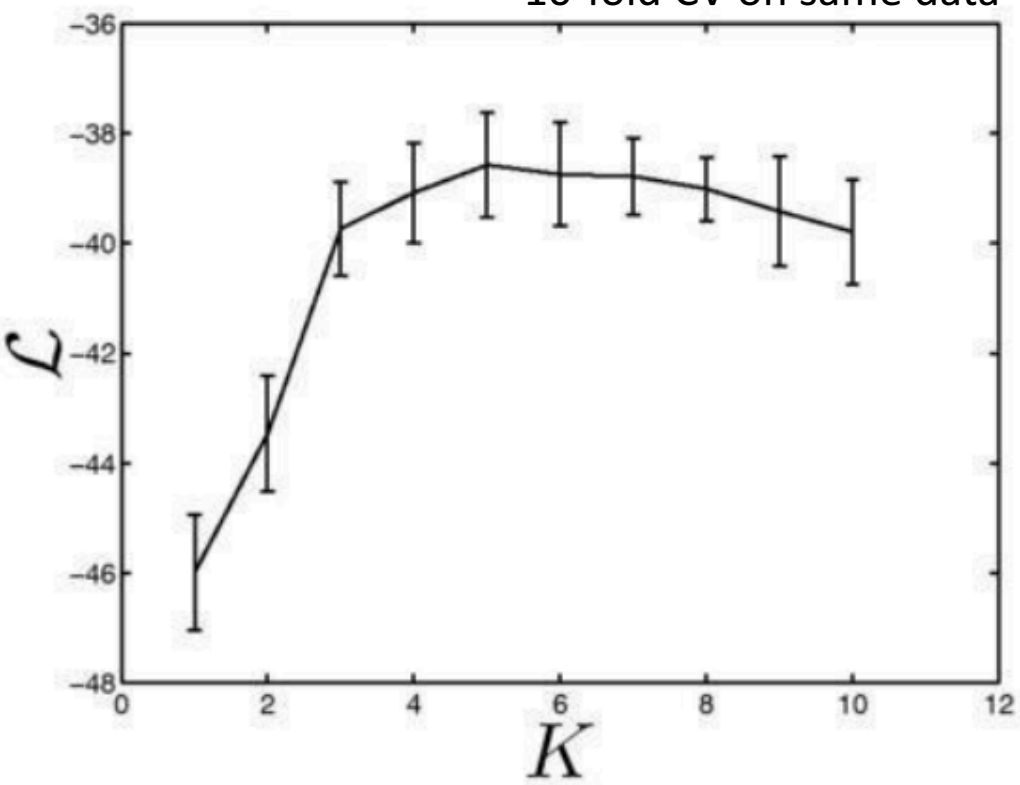


(b) An example of the model at convergence for $K = 10$

The power of a generative model

- **However**: because mixture models are *generative* models, we can run **cross-validation**:
  - For each potential *K*: Hold out data, fit mixtures, then measure likelihood of held-out data

10-fold CV on same data



It's not perfect (somewhere between 3 and 8)

But with non-generative K-means, we have no direct basis for choosing K

Common GMM Simplifications

- Assume **independent** (diagonal) covariance matrix: $\boldsymbol{\Sigma}_k = \sigma_{kd}^2 \mathbf{I}$
- **Isotropic** (iso = same, tropos = way) covariance: $\boldsymbol{\Sigma}_k = \sigma_k^2 \mathbf{I}$

Other Mixtures besides Gaussians

- Could be any probability density $p(\mathbf{x}_n|\boldsymbol{\Delta}_k)$

$$p(\mathbf{X}|\Delta,\boldsymbol{\pi}) = \prod_{n=1}^{N}\sum_{k=1}^{K}\pi_k p(\mathbf{x}_n|\Delta_k)$$

- For D=10 dimensional binary data, e.g.,

$$\mathbf{x}_n = [0,1,0,1,1,1,0,0,0,1]$$

  - Represent as product of (i.e., independent) Bernoulli distributions:

$$p(\mathbf{x}_n|\mathbf{p}_k) = \prod_{d=1}^{D} p_{kd}^{x_{nd}}(1-p_{kd})^{1-x_{nd}}$$

Dimension-specific probabilities for *k*th components

$$\mathbf{p}_k = [p_{k1},\ldots,p_{kD}]^{\mathsf{T}}$$

Product of Bernoulli Mixture Model

Bernoulli likelihood

$$p(\mathbf{x}_n|\mathbf{p}_k) = \prod_{d=1}^{D} p_{kd}^{x_{nd}}(1-p_{kd})^{1-x_{nd}}$$

Probability of mixture component *k*

$$\pi_k = \frac{1}{N}\sum_{n=1}^{N} q_{nk}$$

…stays the same

$$q_{nk} = \frac{\pi_k p(\mathbf{x}_n|\mathbf{p}_k)}{\sum_{j=1}^{K}\pi_j p(\mathbf{x}_n|\mathbf{p}_j)}$$

q_{nk} now in terms of Bernoulli likelihood

Deriving the update of the parameters p_{kd}

$$\begin{aligned}\mathcal{B} &= \sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk}\log p(\mathbf{x}_n|\mathbf{p}_k) + \ldots \\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk}\log\prod_{d=1}^{D} p_{kd}^{x_{nd}}(1-p_{kd})^{1-x_{nd}} + \ldots \\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk}\sum_{d=1}^{D}(x_{nd}\log p_{kd} + (1-x_{nd})\log(1-p_{kd})) + \ldots\end{aligned}$$

$$\mathcal{B} = \sum_{n=1}^{N} q_{nk}\,(x_{nd}\log p_{kd} + (1-x_{nd})\log(1-p_{kd})) + \ldots$$

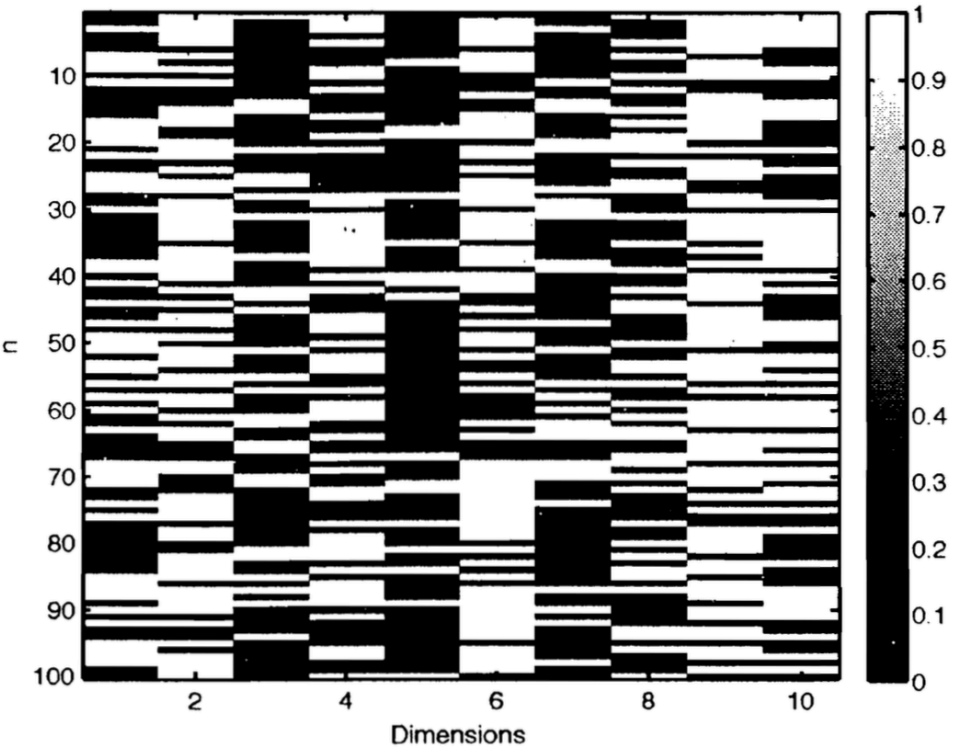
$$\frac{\partial\mathcal{B}}{\partial p_{kd}} = \sum_{n=1}^{N} q_{nk}\left(\frac{x_{nd}}{p_{kd}} - \frac{1-x_{nd}}{1-p_{kd}}\right)$$

$$p_{kd} = \frac{\sum_{n=1}^{N} q_{nk}x_{nd}}{\sum_{n=1}^{N} q_{nk}}$$

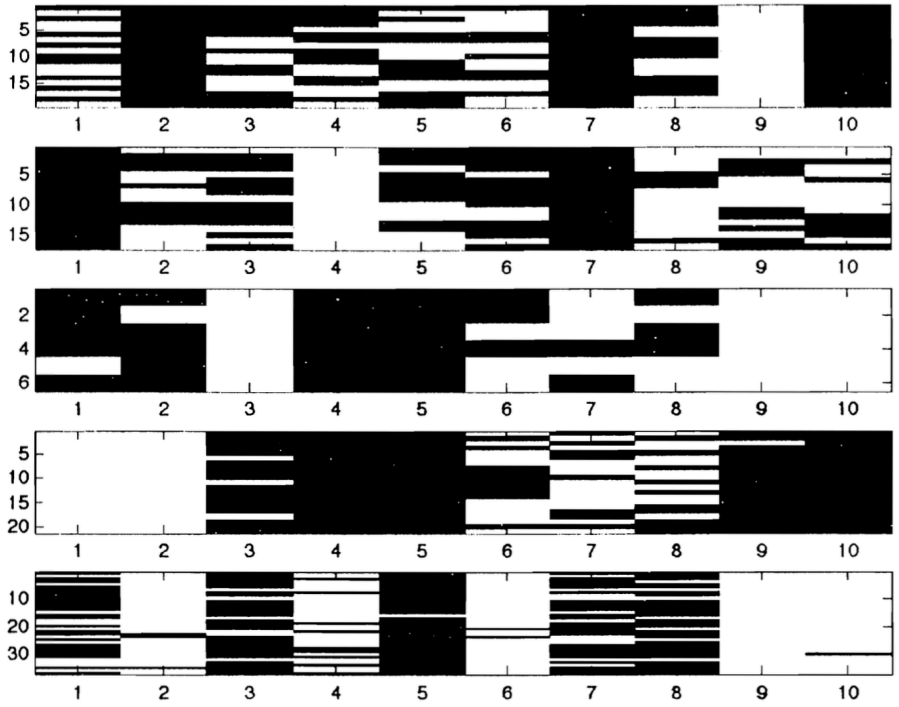
Weighted average of dth data dimension (just like $\boldsymbol{\mu}_k$)

E: update q_{nk} **M**: update $\boldsymbol{\pi}_k$, p_{kd}

Example 10-dimension BinaryClustering



100 data points, each 10-dimensional Binary vector



K=5 clusters

EM is General!

A general pattern:

$\mathbf{X}$ Observed data (may be discrete or continuous; possibly vector-valued for each observation)

$\mathbf{Z}$ Unobserved (latent, missing) data/values (one per observed data point; discrete indicator fn in *hard* EM, continuous probability of indicator in *soft* EM)

$\boldsymbol{\theta}$ Unknown parameters (continuous)

$L(\boldsymbol{\theta}; \mathbf{X}, \mathbf{Z}) = p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ A likelihood function

The maximum likelihood estimate (MLE) of the unknown parameters is determined by the marginal likelihood of the observed data: $L(\boldsymbol{\theta}; \mathbf{X}) = p(\mathbf{X}|\boldsymbol{\theta}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$

Typically this is intractable (***Z*** may be exponential or worse in size)

EM algorithm seeks to find the MLE of the marginal likelihood by iteratively applying the following two steps:

(E) Expectation step: Calculate the expected value of the log likelihood function with respect to the conditional distribution of ***Z*** given ***X*** under the current estimate of the parameters $\boldsymbol{\theta}^{(t)}$ $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) = \mathrm{E}_{\mathbf{Z}|\mathbf{X},\boldsymbol{\theta}^{(t)}}[\log L(\boldsymbol{\theta}; \mathbf{X}, \mathbf{Z})]$ (What typically happens here is updating ***Z*** under fixed $\boldsymbol{\theta}^{(t)}$ and data ***X***.)

(M) Maximization step: Find the parameter(s), $\boldsymbol{\theta}^{(t+1)}$, that maximize(s) this quantity

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$$

Dempster, A.P.; Laird, N.M.; Rubin, D.B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm". *Journal of the Royal Statistical Society*, Series **B** 39 (1): 1–38.

Main parametric modeling frameworks

- Minimizing a Loss function
  - Linear model
  - Linear least mean squares
- Maximum Likelihood
  - Probabilistic model of uncertainty (noise, error)
  - Maximize the likelihood w.r.t. parameters
  - Linear model with additive Gaussian noise
- Bayesian Approach
  - Treat parameters as random variables
  - Use Bayes Theorem to combine likelihood & prior to find posterior distribution

Approaches to Avoiding Over-fitting
i.e., how to achieve ***generalization***
Regularization
Cross Validation (estimating the generalization error)

- Estimation Techniques (often used in Bayesian approaches)
  - Gradient methods (Widrow-Hoff (1st), Newton-Rhapson (2nd))
  - Laplace Approximation
  - Monte Carlo estimation of expectation; Metropolis-Hastings
- Classification (& Regression)
- Clustering
- **Projection**

predicting output

Main algorithmic families of Machine Learning

Dimension Reduction

- Projection methods are part of a broader class of ***Dimension Reduction*** methods.

- Recall that a dimension is a “feature”, and the value along the dimension is the feature value.

- What problems are dimension reduction methods used to address?

Recall:
The Curse of Dimensionality

- A major problem!
- If the data **x** lie in high dimensional space, then an enormous amount of data is required to learn distributions or decision rules.
- Example:
  - 50 dimensions. Each dimension has 20 "levels"
  - Total of 20^{50} cells!
  - But the number of data samples will be far less.
  - There will not be enough data samples to estimate the parameters.

(Adapted from Alan L. Yuille Stat 231, Fall 2004)

The Curse of Dimensionality

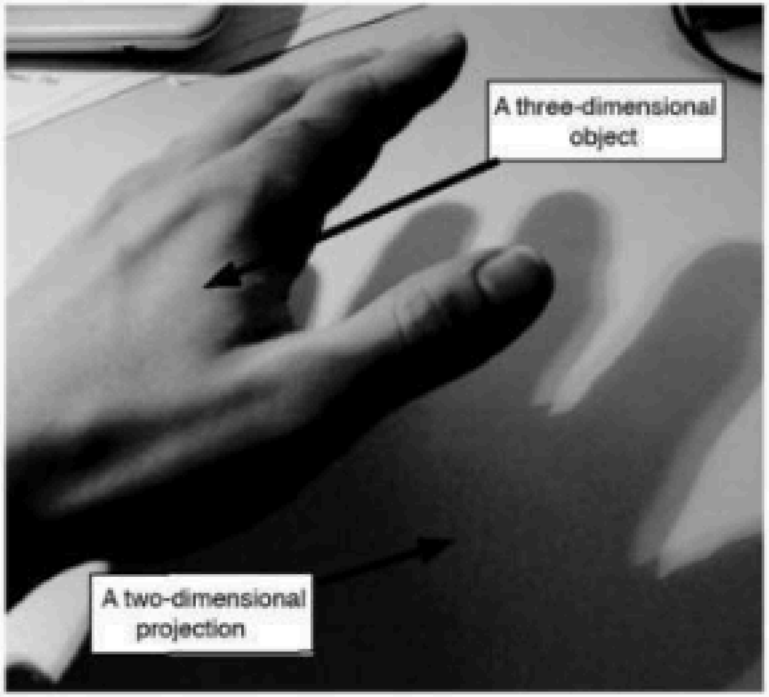
- One way to deal with high dimensionality is to assume that we know the form of the probability distribution.
- For example: a Gaussian model with N dimensions has $N+N(N-1)/2$ parameters to estimate. Requires $O(N^2)$ data to learn reliably. *May* be practical.

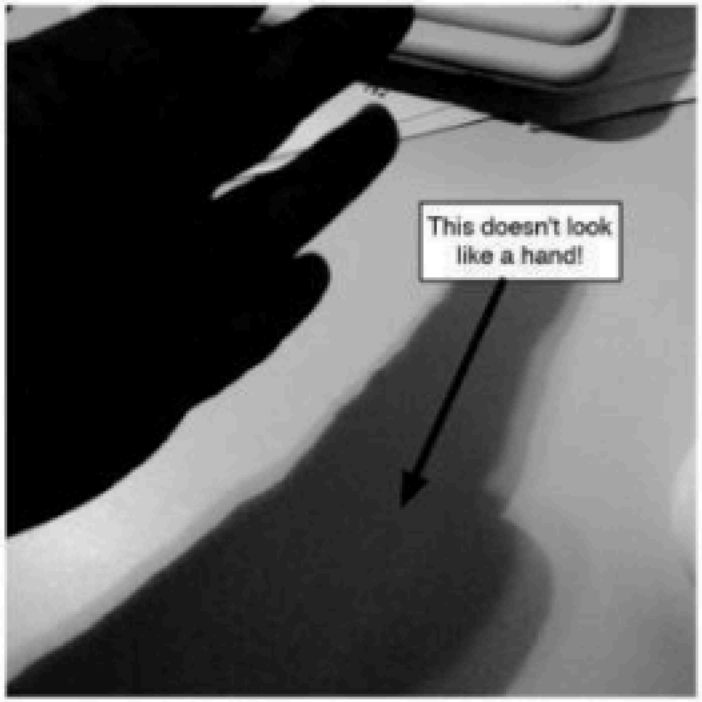
(Adapted from Alan L. Yuille Stat 231, Fall 2004)

Dimension Reduction

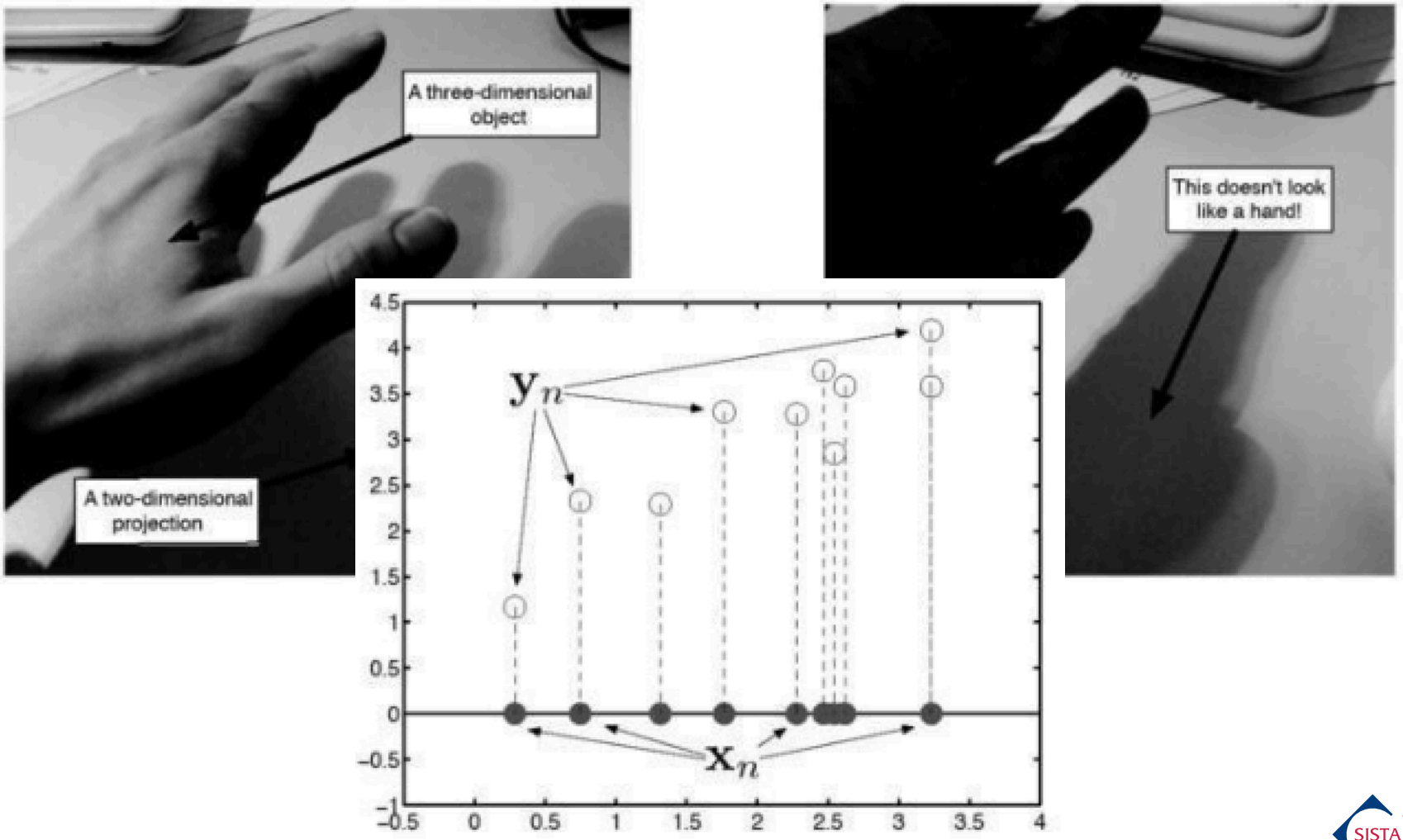
- Another problem
  - Can't visualize data in more than 3 dimensions!
- **Techniques for dimension reduction**:
  - Principle Components Analysis (PCA)
  - Fisher's Linear Discriminant
  - Multi-dimensional Scaling
  - Independent Component Analysis
  - t-SNE (t-Distributed Stochastic Neighbor Embedding)
- **Approach**: Project data into a lower-dimensional space.
- **Goal**: Preserve as much "interesting" structure in data as possible

Intuition of Projection





Intuition of Projection

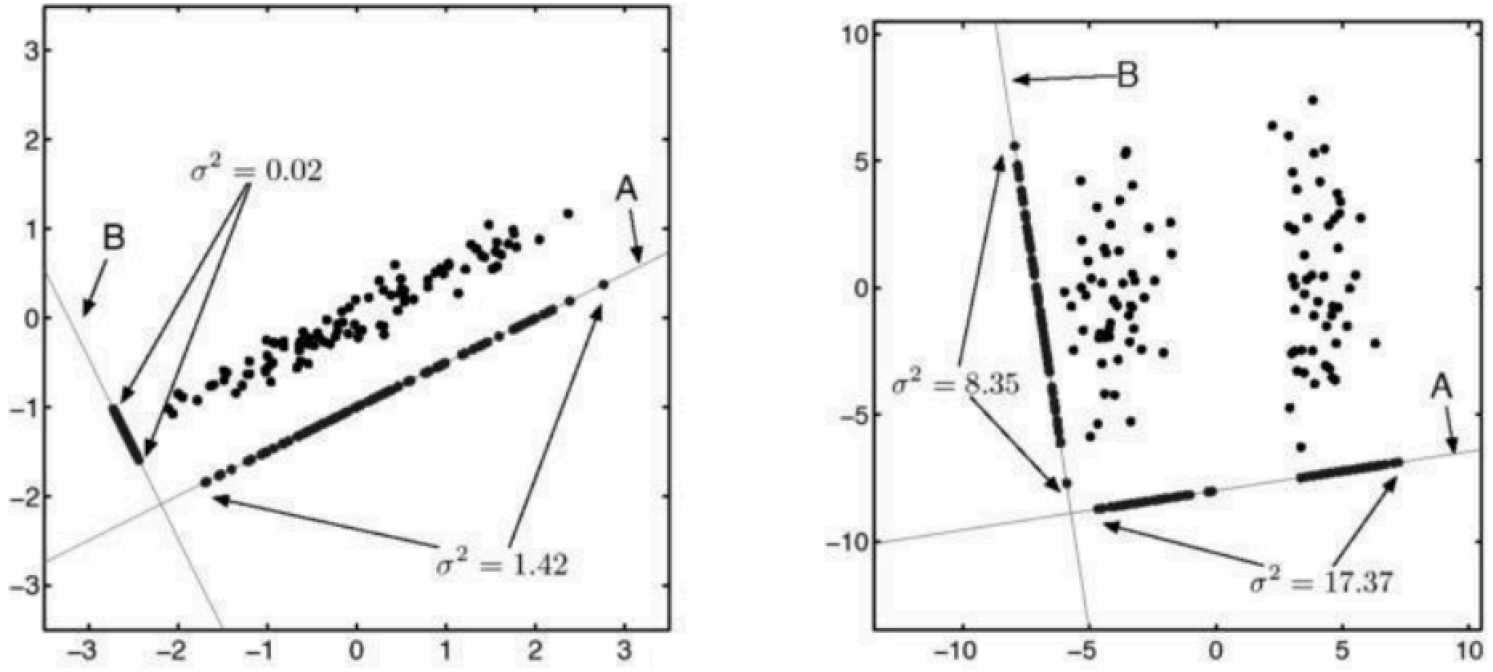


Principle Components Analysis (PCA)

- PCA is the most commonly used dimension reduction technique.
- (Also called the Karhunen-Loeve transform)
- Principle:
  - **Linear**, **orthogonal** projection method to reduce the number of parameters
  - Criteria of interestingness: **maximize variability** along new axes
- Properties
  - Can be viewed as a rotation of the existing axes to new positions in the space defined by the original variables (typically followed by selection/removal)
  - New axes are orthogonal and represent the directions with maximum variability.
  - Can also be viewed as a way to transfer a set of correlated variables into a new set of uncorrelated variables

Variables ≡ features ≈ dimension ≡ axes

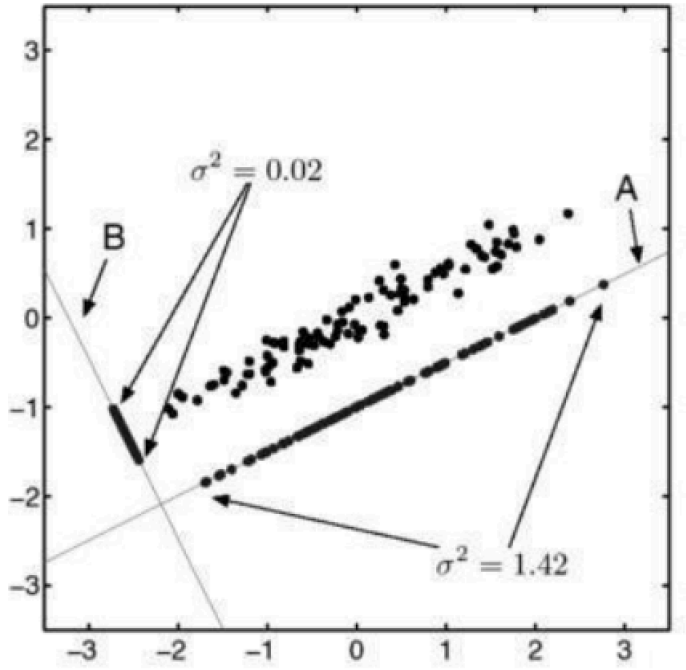
Why care about variance?

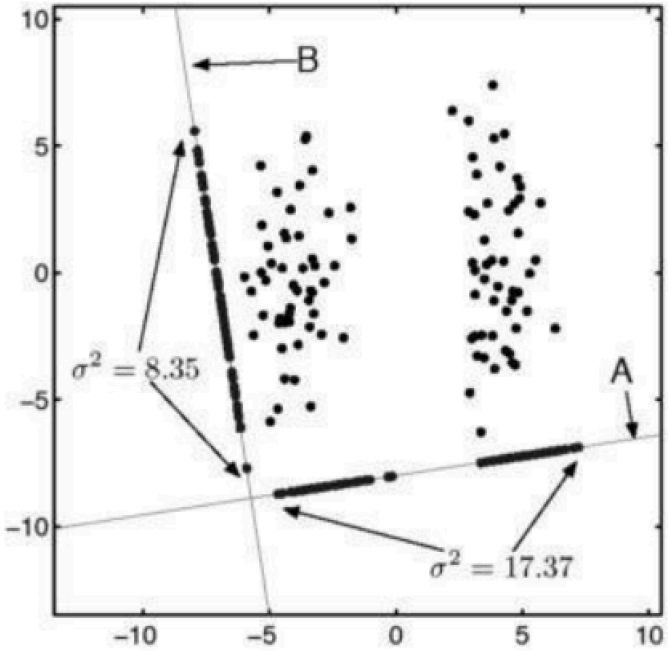


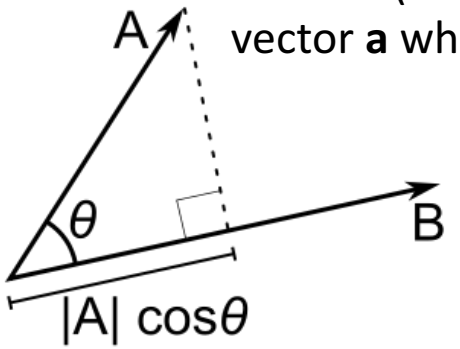
- We don't know the structure of interest ahead of time. But structure tends to be associated with variance.
- PCA uses variance as a proxy for interest.

$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu_x)^2$$

Projection!







The dot (inner) project is a measure of the length of vector **a** when we *project* it onto **b**

$$\mathbf{y}_n = [y_{n1},\ y_{n2}]^\mathsf{T}$$ Datum in original spatial coordinates

Datum in new spatial coordinates

$$x_n = w_1 y_{n1} + w_2 y_{n2}$$

$$x_n = \mathbf{w}^\mathsf{T} \mathbf{y}_n$$

Map from one space to the other

$$\mathbf{w} = [w_1,\ w_2]^\mathsf{T}$$

Principle Components Analysis

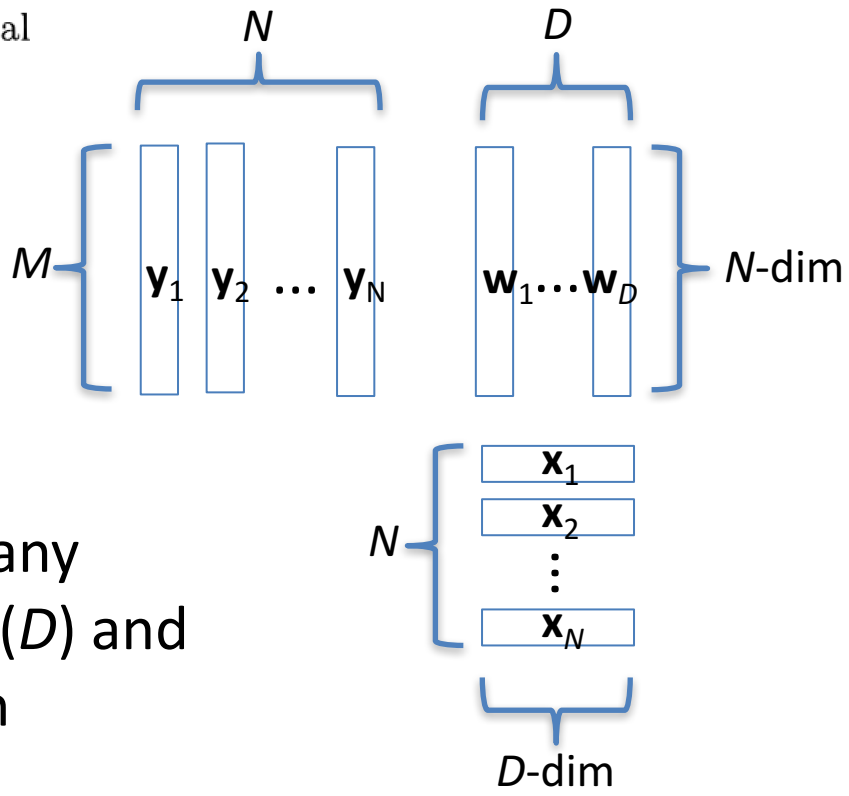
large *small*

projecting from M to D dimensions.

PCA will define D vectors, $\mathbf{w}_d$ N-dimensional

$$\mathbf{x}_n = [\, x_{n1}, \ldots, x_{nD}]^\mathsf{T}$$

$$x_{nd} = \mathbf{w}_d^\mathsf{T}\mathbf{y}_n$$



The learning task: to choose how many dimensions we want to project into (D) and find a projection vector, $\mathbf{w}_d$, for each

Principle Components Analysis

large *small*

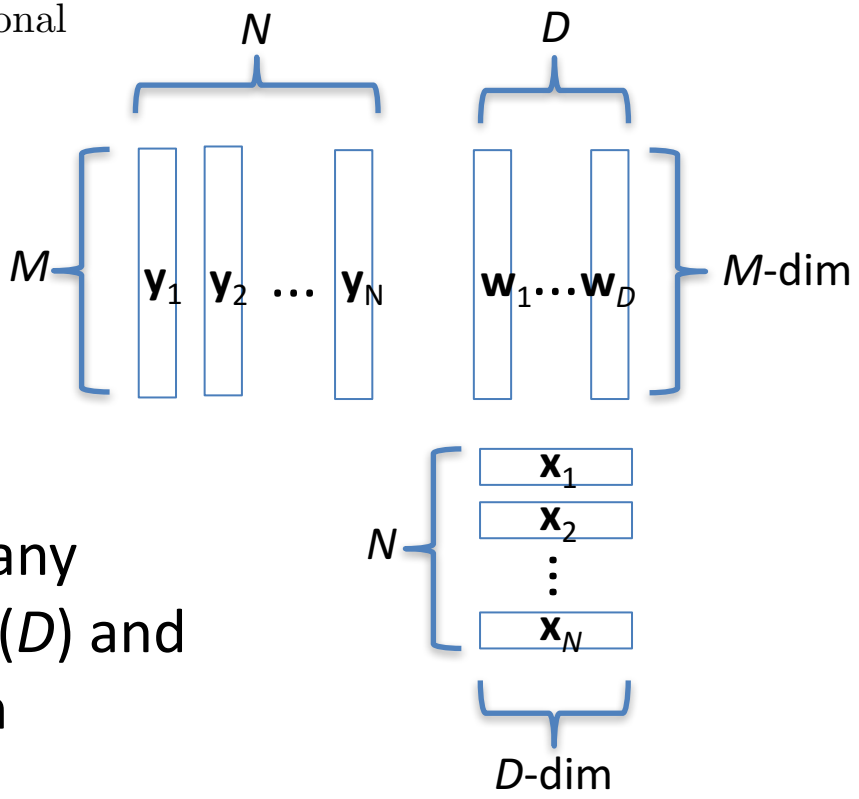
projecting from M to D dimensions.

M-dimensional

PCA will define D vectors, $\mathbf{w}_d$

$$\mathbf{x}_n = [\, x_{n1}, \ldots, x_{nD}]^{\mathsf{T}}$$

$$x_{nd} = \mathbf{w}_d^{\mathsf{T}} \mathbf{y}_n$$



The learning task: to choose how many dimensions we want to project into (D) and find a projection vector, $\mathbf{w}_d$, for each

2 Constraints

1 PCA uses **variance** in the projected space as the criterion to choose $\mathbf{w}_d$, but all $\mathbf{w}_d$'s must be **mutually orthogonal**.

$$\mathbf{w}_i^T \mathbf{w}_j = 0, \ \forall j \neq i$$

For example:
$\mathbf{w}_1$ is the projection that makes the variance in x_{n1} as high as possible
$\mathbf{w}_2$ will also maximize the variance, but must be orthogonal to $\mathbf{w}_1$ ($\mathbf{w}_1^T\mathbf{w}_2=0$)
$\mathbf{w}_3$ will also maximize the variance, but must be orthogonal to $\mathbf{w}_1$ and $\mathbf{w}_2$ …

2 PCA also imposes the constraint that each $\mathbf{w}_i$ **must have length 1** (not a restriction on the technique, rather, its needed to properly define the optimization)

$$\mathbf{w}_i^T \mathbf{w}_i = 1$$

Following FCML, we'll derive an expression for the variance of x_n for one dimension (D=1), which we'll then maximize

$$x_n = \mathbf{w}^T \mathbf{y}_n$$

scalar!

Defining the Variance of x_n

Assume **y** has 0 mean: $\bar{\mathbf{y}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{y}_n = 0$

(Enforce this by subtracting $\bar{\mathbf{y}}$ from every $\mathbf{y}_n$.)

The variance of x is $\sigma_x^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})^2$

$$\sigma_x^2 = \frac{1}{N}\sum_{n=1}^{N} x_n^2 \qquad x_n = \mathbf{w}^\mathsf{T}\mathbf{y}_n$$

We can then simplify this expression using the assumption $\bar{\mathbf{y}} = 0$

$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{w}^\mathsf{T}\mathbf{y}_n$$
$$= \mathbf{w}^\mathsf{T}\left(\frac{1}{N}\sum_{n=1}^{N}\mathbf{y}_n\right)$$
$$= \mathbf{w}^\mathsf{T}\bar{\mathbf{y}} = 0.$$

Plug in definition of x_n as inner product

$$\sigma_x^2 = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{w}^\mathsf{T}\mathbf{y}_n)^2$$
$$= \frac{1}{N}\sum_{n=1}^{N}\mathbf{w}^\mathsf{T}\mathbf{y}_n\mathbf{y}_n^\mathsf{T}\mathbf{w}$$
$$= \mathbf{w}^\mathsf{T}\left(\frac{1}{N}\sum_{n=1}^{N}\mathbf{y}_n\mathbf{y}_n^\mathsf{T}\right)\mathbf{w}$$

C is the sample covariance matrix
(keeping in mind $\bar{\mathbf{y}}$=0 here)

$$\mathbf{C} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{y}_n - \bar{\mathbf{y}})(\mathbf{y}_n - \bar{\mathbf{y}})^\mathsf{T} \qquad \sigma_x^2 = \mathbf{w}^\mathsf{T}\mathbf{C}\mathbf{w},$$

Maximizing the Variance of x_n

Goal is to maximize $\sigma_x^2 = \mathbf{w}^\mathsf{T}\mathbf{C}\mathbf{w}$ $\quad \mathbf{C} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{y}_n - \bar{\mathbf{y}})(\mathbf{y}_n - \bar{\mathbf{y}})^\mathsf{T}$

Remember constraint 2: $\mathbf{w}_i^\mathsf{T}\mathbf{w}_i = 1$

If we did not have this, we could just keep increasing **w** to increase σ^2 !

Incorporate constraint as a Lagrange multiplier!

$$L = \mathbf{w}^\mathsf{T}\mathbf{C}\mathbf{w} - \lambda(\mathbf{w}^\mathsf{T}\mathbf{w} - 1)$$

Now, maximize: take partial derivative, set to 0, rearrange:

$$\frac{\partial L}{\partial \mathbf{w}} = 2\mathbf{C}\mathbf{w} - \lambda\mathbf{w} = \mathbf{0}$$

$$\mathbf{C}\mathbf{w} = \lambda\mathbf{w}$$

But we want to solve for **w**!
What to do?

The eigenvector/eigenvalue equation for square matrix **A**

$$\lambda_i \mathbf{u}_i = \mathbf{A}\mathbf{u}_i$$

Eigenvectors & Eigenvalues

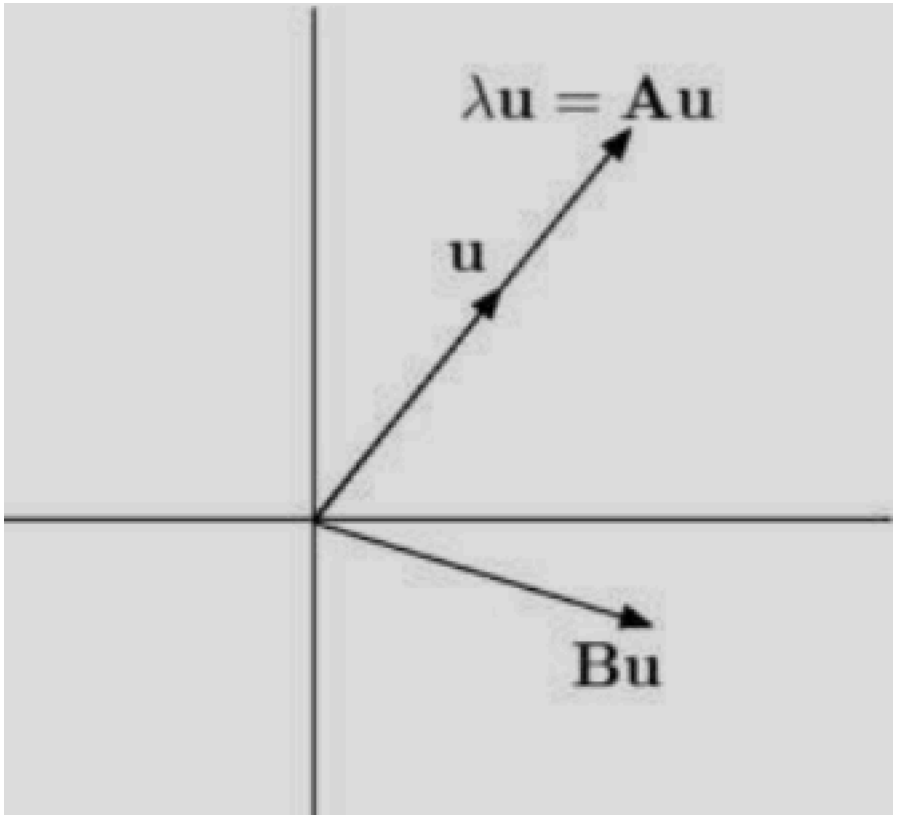
$$\lambda_i \mathbf{u}_i = \mathbf{A}\mathbf{u}_i$$

A is a square, symmetric, positive definite NxN matrix
Solving this equation results in pairs of eigenvalues (λ_i) and eigenvectors ($\mathbf{u}_i$)

- There will be N pairs.
- The N eigenvectors will be mutually orthogonal.

The geometric view:
Multiplying an N-dimensional vector **u** by an NxN square matrix, **B**, generally results in a "rotation" (and scaling) of **u**.

The eigenvector/-value pairs for a square matrix **A** are the vectors **u** for which applying the "rotation" **A** only results in a change in length of **u**; the magnitude of this change is given by the scalar λ.



Maximizing the Variance of x_n

Goal is to maximize $\sigma_x^2 = \mathbf{w}^\mathsf{T}\mathbf{C}\mathbf{w}$

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

$$\frac{\partial L}{\partial \mathbf{w}} = 2\mathbf{C}\mathbf{w} - \lambda\mathbf{w} = \mathbf{0}$$

$$\mathbf{C}\mathbf{w} = \lambda\mathbf{w}$$

The eigenvector/eigenvalue equation for square matrix **A**

$$\lambda_i \mathbf{u}_i = \mathbf{A}\mathbf{u}_i$$

But what is the interpretation of λ ?

$$\sigma_x^2 = \mathbf{w}^\mathsf{T}\mathbf{C}\mathbf{w}$$

$$\mathbf{w}^\mathsf{T}\mathbf{w} = 1$$

Constraint 2 (again)

$$\sigma^2\mathbf{w}^\mathsf{T}\mathbf{w} = \mathbf{w}^\mathsf{T}\mathbf{C}\mathbf{w}$$

$$\sigma^2\mathbf{w} = \mathbf{C}\mathbf{w}$$

So λ corresponds to the variance of the data in the projected space defined by **w**

If we find the *M* eigenvector/eigenvalue pairs of the covariance matrix **C**, the pair with the highest eigenvalue corresponds to the projection with the maximal variance $\mathbf{w}_1$. The 2nd highest eigenvalue corresponds to $\mathbf{w}_2$, then $\mathbf{w}_3$, etc.

The PCA Algorithm

PCA on a set of data objects, $\mathbf{y}_1, \ldots, \mathbf{y}_N$

$$\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N]^\mathsf{T}$$

1. Transform the M-dimensional data to have zero mean by subtracting $\bar{\mathbf{y}}$ from each object wher e $\bar{\mathbf{y}} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{y}_n$.

2. Compute the sample covariance matrix $\mathbf{C} = \frac{1}{N}\sum_{n=\ 1}^{N} \mathbf{y}_n\mathbf{y}_n^\mathsf{T}$ (or $\mathbf{C} = \frac{1}{N}\ \mathbf{Y}^\mathsf{T}\mathbf{Y}$)

3. Find the M eigenvector/eigenvalue pairs of the covariance matrix.

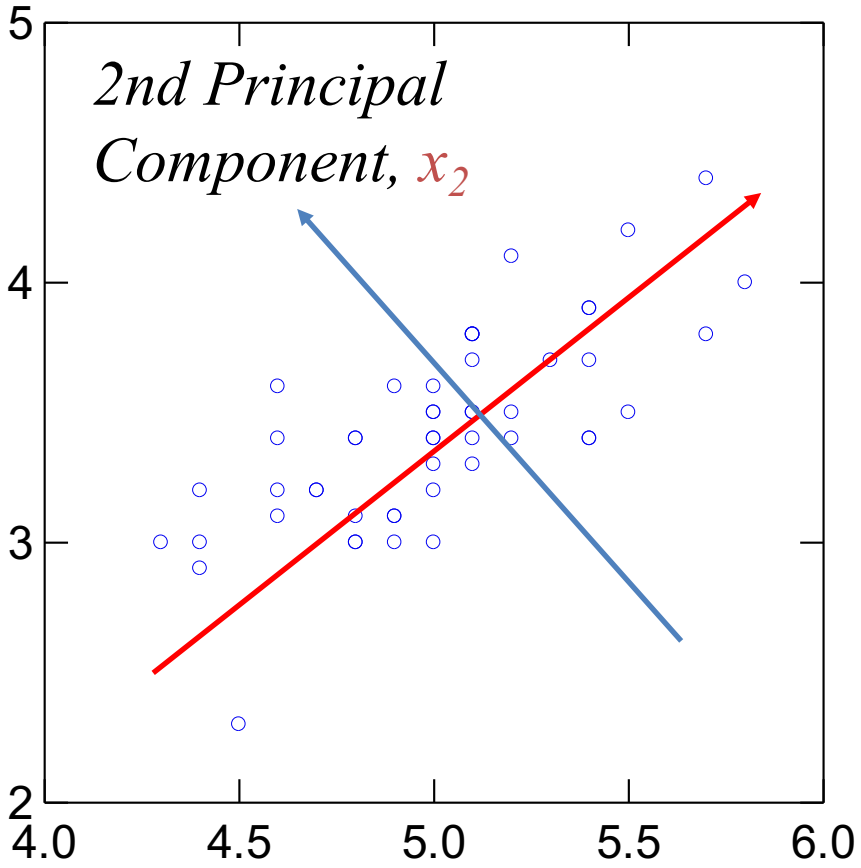
`numpy.linalg.eig` in python, `eigs` in Matlab

4. Find the eigenvectors corresponding to the D highest eigenvalues, $\mathbf{w}_1, \ldots, \mathbf{w}_D$.

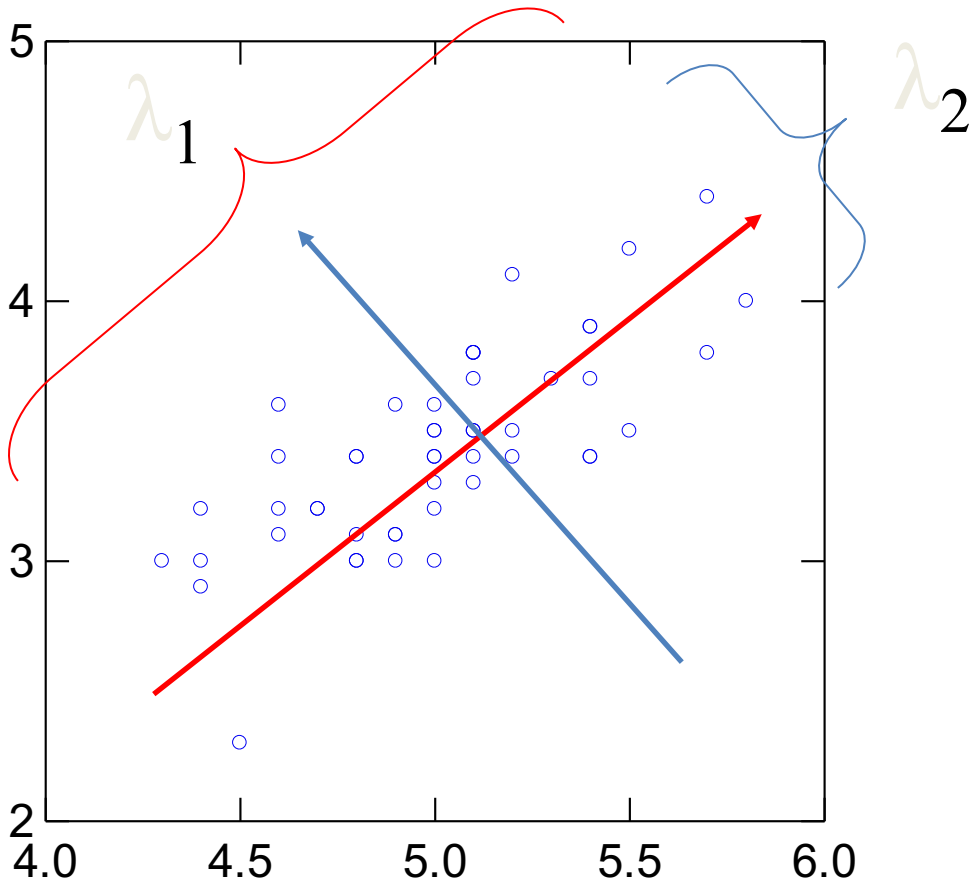
5. Create the dth dimension for object n in the projection, $x_{nd} = \mathbf{w}_d^\mathsf{T}\mathbf{y}_n$ (or $\mathbf{X} = \mathbf{YW}$, where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_D]$, i.e. the $M \times D$ matrix created by placing the D eigenvectors alongside one another and $\mathbf{X}$ is the $N \times D$ matrix defined as $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\mathsf{T}$)

projection

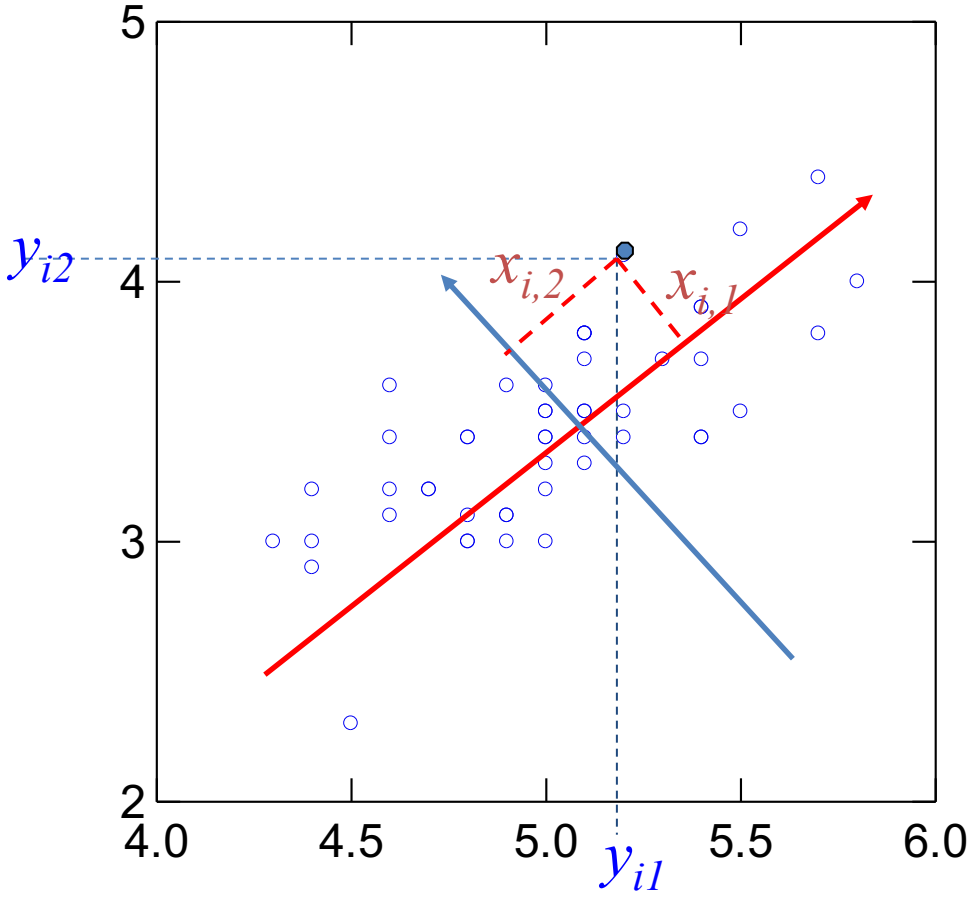
PCA Eigenvectors/Principle Components



PCA Eigenvalues



PCA Projections of data

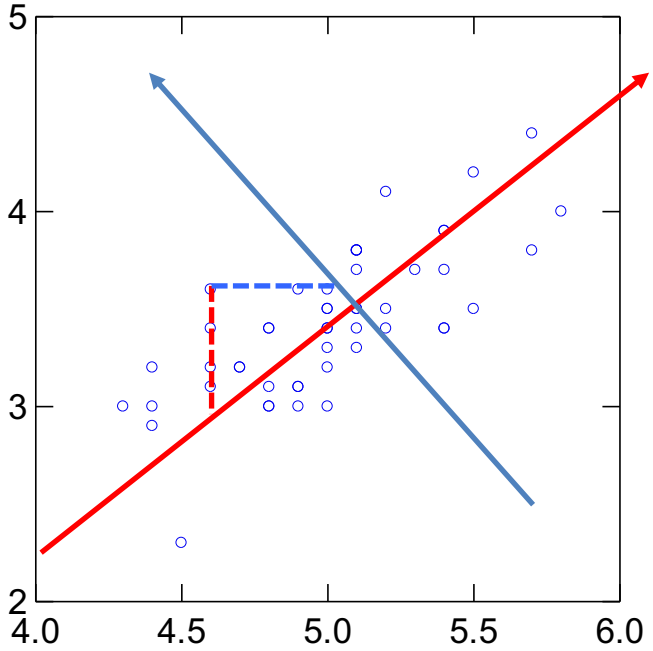


The Least-Squares View

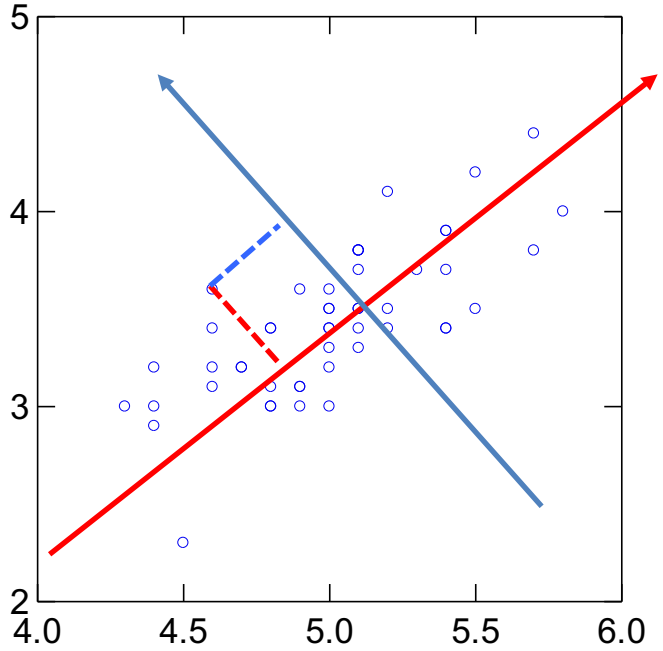
Principle Components are a series of linear least-squares fits to a sample, each orthogonal to all previous ones.

HOWEVER: Unlike linear regression, where the data are projected down to the line perpendicularly to the original axes, instead they're projected perpendicularly to the principle component axis.

Least Squares

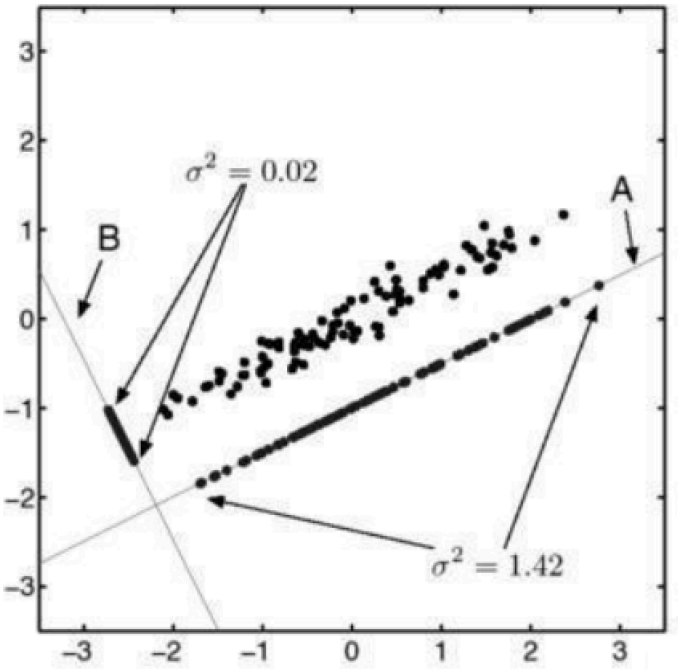


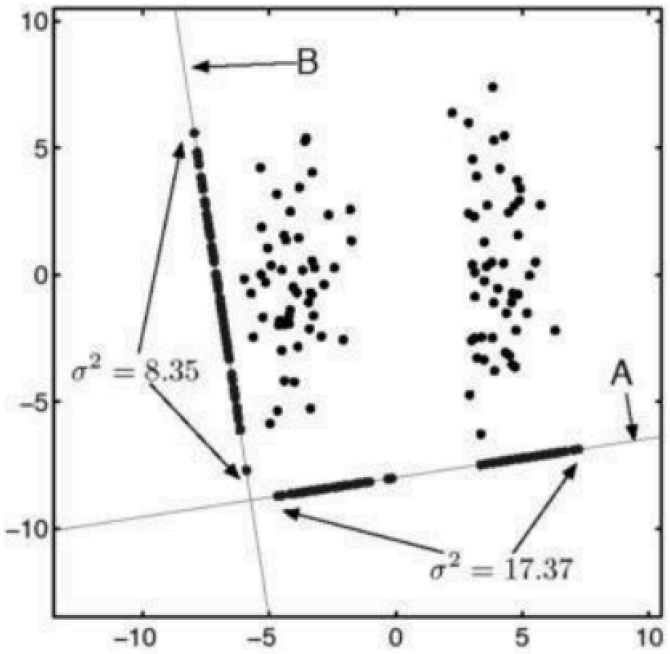
Total least squares (PCA-version)



The "PCA version" is called "Total least squares" or "rigorous least squares" or "orthogonal regression"; in this model, observational errors (deviations from the line) are w.r.t. both dependent and independent variables, rather than the dependent variable(s) alone.

Simple Examples



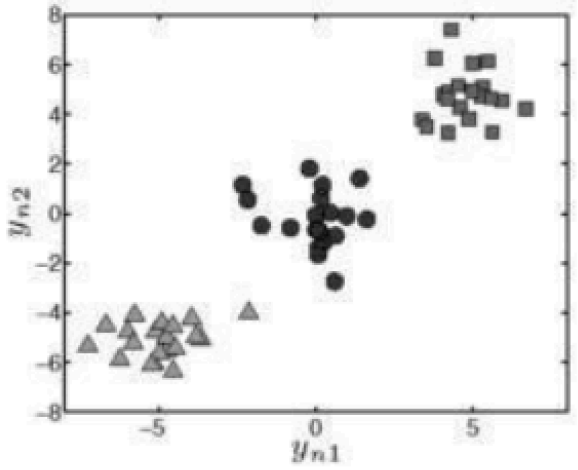


This is 2d data, so there are 2 eigenvectors (2 possible principle components)

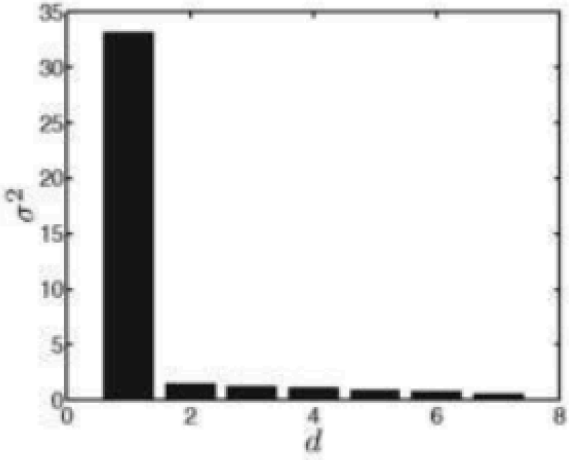
Note: the principle component eigenvectors only give directions -- they're plotted as offset axes for convenience.

Another Example

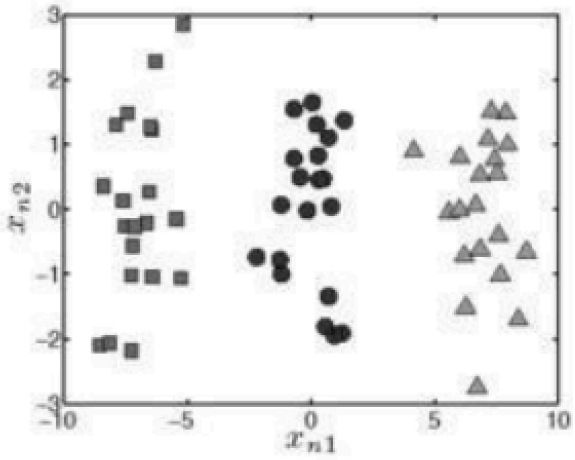
eigen-spectrum



(a) First two dimensions of the data objects $\mathbf{y}_n$



(b) The seven eigenvalues (variances of the projected dimensions)

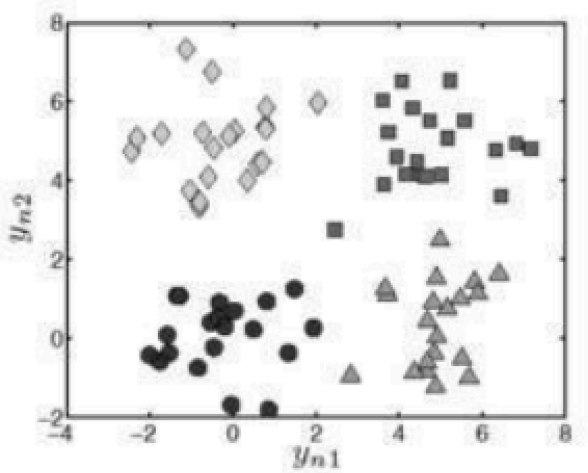


(c) The data projected onto the first two principal components

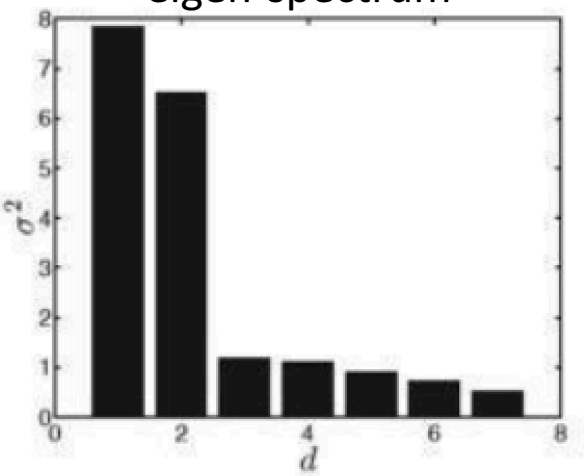
These are actually 7-dimensional data;
the other 5 dimensions are sampled from $\mathcal{N}(0,1)$.
So structure in first two dimensions and symmetric noise in rest.

Another Example

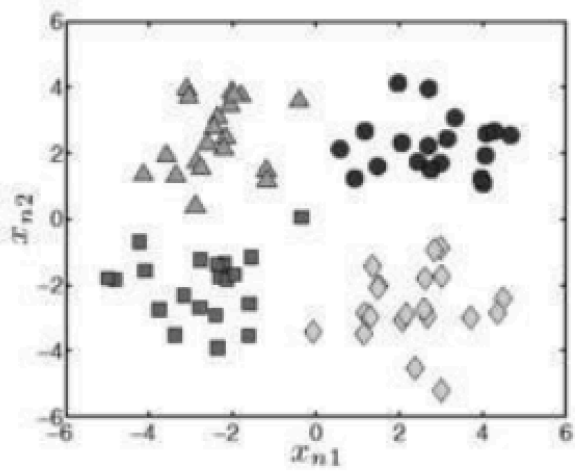
eigen-spectrum



(a) First two dimensions of the data objects $\mathbf{y}_n$



(b) The seven eigenvalues (variances of the projected dimensions)

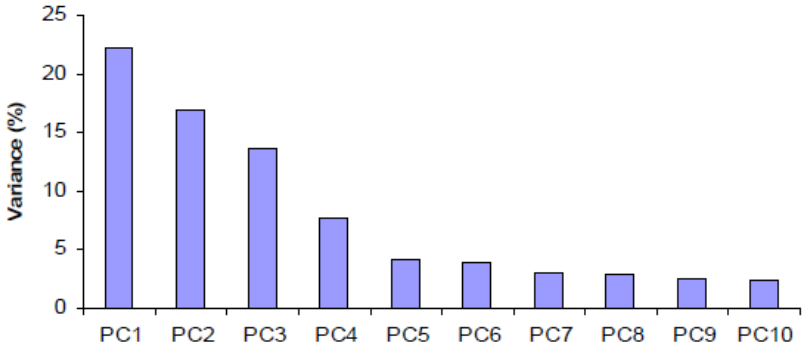


(c) The data projected onto the first two principal components

Note:
(1) The data labels are for visualization, PCA does NOT discover them
(2) Although we generate the data (same as before) with dimensions 3-7 random, the order of the dimensions makes no difference (WHY?)

Choosing D

- If for visualization, choose the top 1-3 eigen-spectra, but can't really visualize in higher than 3 dimensions.
- More generally, the eigen-spectrum provides information about potential interestingness/informativeness, but is subjective.



- Try to find objective measure: if PCA is first step of classification or regression (supervised method), then utility of different D values (cross-validation!)

Eigenfaces!

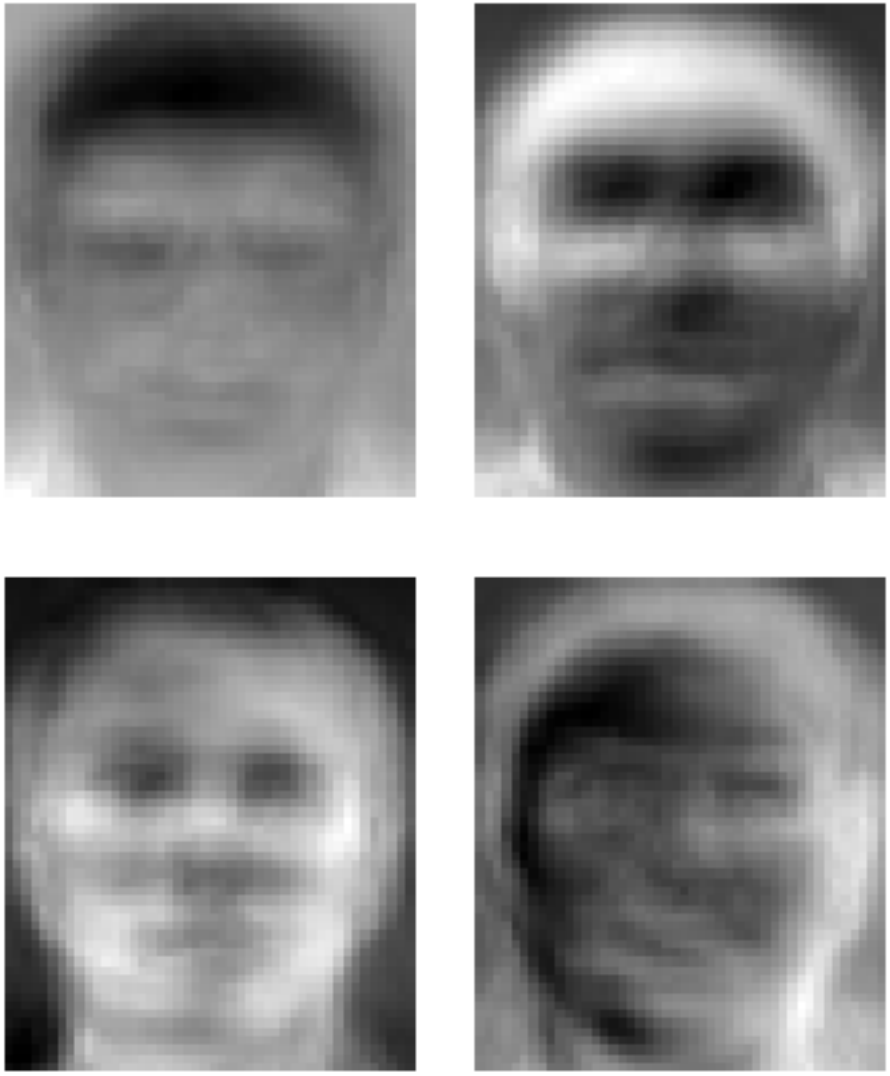
A dataset of M face images, where each image is a set of N pixel intensities.

Derive the top principle components:
Find the eigenvectors for the components, project the M face images to D-dimensional eigenfaces.

To then use for classification, the original database of images is saved as collections of weights describing the contribution each eigenface makes to that image. New face is presented for classification, its own weights are found by projecting the image into the collection of eigenfaces. A method like Nearest Neighbors can then be used to find the distance between the new image vector and vectors of existing images

Reported result: M=10,000 images, each 100x100 pixels, obtain 10,000 eigenvectors; but typically most faces can be identified using a projection on between 100 and 150 eigenfaces!

M. Turk and A. Pentland (1991). "Face recognition using eigenfaces". Proc. IEEE Conference on Computer Vision and Pattern Recognition. pp. 586–591.



A different eigenfaces example from Alan Yuille, using PCA for analysis

- The images of an object under different lighting lie in a low-dimensional space.
- The original images are 256x 256. But the data lies mostly in 3-5 dimensions.
- First show the PCA for a face under a range of lighting conditions. The PCA components have simple interpretations.
- Then plot $\frac{\sum_{i=1}^{M} \lambda_i}{\sum_{i=1}^{N} \lambda_i}$ as a function of M for several objects under a range of lighting.

Example of Alan Yuille's face

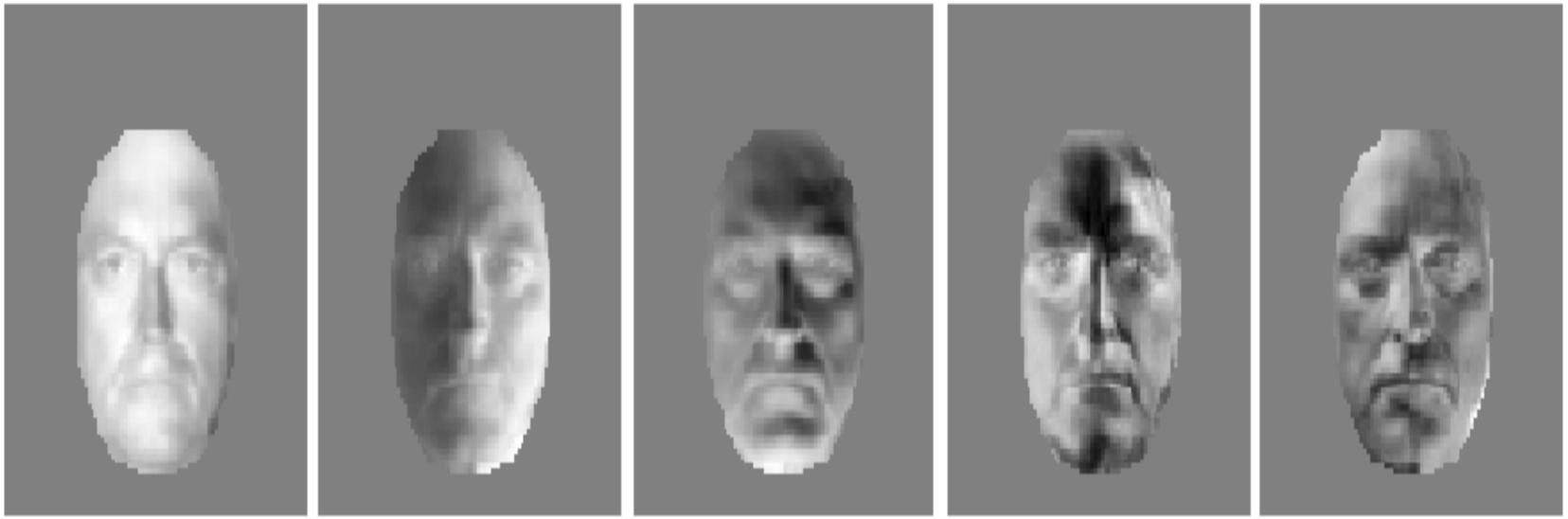


Figure 4: The eigenvectors calculated from the sparse set for the human face. Note that the images were only lit from the right so the eigenvectors are not perfectly symmetric. Observe also that the first three eigenvectors appear to be images of the face illuminated from three orthogonal lighting conditions in agreement with the orthogonal lighting conjecture.

Additional Properties of PCA

- PCA is commonly used to project data X onto the first k principle components, giving the k-dimensional view of the data that best preserves the first two moments (mean, variance).
- PCA is commonly used for lossy compression of high dimensional data.
- Emphasis on variance is where the weaknesses of PCA stem from:
  - The PCs depend heavily on the units of measurement. Where the data matrix contains measurements of vastly differing orders of magnitude, the PC will be greatly biased in the direction of larger measurements. It is therefore recommended to calculate PCs from $Corr(X)$ instead of $Cov(X)$.
  - Robustness to outliers is also an issue. Variance is affected by outliers, therefore so are PCs

Limitations of PCA

1. Data are real-valued
2. Assumes no missing values in the data

- Scientific data often has missing values
- Movie ratings: 1-5 scale (integer) and most people don't watch and rate EVERY movie!

- We'd like to construct the analysis provided by PCA but that can overcome these limitations

Latent Variable Models

- Mixture models!
- Characteristics of objects not provided in the data: ***latent*** (hidden) variables
- Two types of latent variables:
  1. Real feature of the object, but not measured
  2. "Abstract qualities" that may not correspond directly to any particular real thing (entity, process, relation) but instead arise from our modeling assumptions and might be useful
- Example of type 2: Indicator variables; enabled us to build mixture models but do not necessarily correspond to anything in reality. $q_{nk} = p(z_{nk} = 1|\mathbf{x}_n, \pi, \Delta)$
- PCA is also a type of latent variable model: the dimensions identified are (potentially) useful ways of relating/describing relationships among data; do they correspond to real properties?

Extending PCA

- We want to extend PCA beyond its limitations.
- Extend: create a probabilistic version
- Inference, will require approximation of the posterior; direct solution is intractable
- Ch.4 methods: iterative gradient methods (Newton-Rhapson), Laplace approximation, MCMC sampling.
- Here we'll introduce another method: **Variation Bayes**.

Variational Bayes (VB)

- Has some similarity to Laplace estimation (finding a friendlier but approximate distribution family) and the trick we saw in EM of using Jensen's Inequality (maximizing a lower-bound to the log likelihood) to find an estimate the posterior.
- Consider a very general case:
  - Some data **Y**
  - Model with parameters and latent variables, **θ**
- In a Bayesian framework, distinction between how to deal with latent variables and parameters becomes blurred: they are all things we don't know, so treat as random variables.

Variational Bayes (VB)

- The marginal likelihood: $p(\mathbf{Y}) = \int p(\mathbf{Y}, \boldsymbol{\theta}) d\boldsymbol{\theta} = \int p(\mathbf{Y}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}$

(Here we've omitted conditioning on all things that are constant: model type, prior parameters, hyper-parameters, etc.; these are all model-specific, but we first want to develop the general VB method)

- This integral is almost always very difficult
- Log marginal likelihood: $\log p(\mathbf{Y}) = \log \int p(\mathbf{Y}, \boldsymbol{\theta}) d\boldsymbol{\theta}$
- Apply Jensen's Inequality: $\log \mathbf{E}_{p(z)} \{f(z)\} \geq \mathbf{E}_{p(z)} \{\log f(z)\}$
- Introduce arbitrary distribution over $\boldsymbol{\theta}$, $Q(\boldsymbol{\theta})$, to make the log marginal likelihood characterized as an expectation, to make Jensen's Inequality applicable:

$$\log p(\mathbf{Y}) = \log \int Q(\boldsymbol{\theta}) \frac{p(\mathbf{Y}, \boldsymbol{\theta})}{Q(\boldsymbol{\theta})} d\boldsymbol{\theta}$$

$$\geq \int Q(\boldsymbol{\theta}) \log \frac{p(\mathbf{Y}, \boldsymbol{\theta})}{Q(\boldsymbol{\theta})} d\boldsymbol{\theta} = \mathcal{L}(Q)$$

Variational Bayes (VB)

- Now we'll compute the difference between the true log marginal likelihood and our new bound, in order to approximate the posterior

$$\log p(\mathbf{Y}) - \mathcal{L}(Q) = \log p(\mathbf{Y}) - \int Q(\boldsymbol{\theta}) \log \frac{p(\mathbf{Y}, \boldsymbol{\theta})}{Q(\boldsymbol{\theta})} \mathrm{d}\boldsymbol{\theta}$$

Kullback-Leibler Divergence

$$\begin{aligned}\log p(\mathbf{Y}) - \mathcal{L}(Q) &= -\int Q(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{\theta}|\mathbf{Y})}{Q(\boldsymbol{\theta})} \\ &= -\mathrm{KL}\left[Q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathbf{Y})\right]\end{aligned}$$

- KL divergence is a measure of the dissimilarity between two probability distributions

$$\mathrm{KL}[q(x)||p(x)] = \int q(x) \log \frac{p(x)}{q(x)}\, dx \quad \text{(continuous)}$$

$$\mathrm{KL}[q(x)||p(x)] = \sum_x q(x) \log \frac{p(x)}{q(x)} \quad \text{(discrete)}$$

- KL ≥ 0, but also asymmetric (not a *metric*)

$$\mathrm{KL}[q(x)||p(x)] \neq \mathrm{KL}[p(x)||q(x)]$$

- Maximizing $\mathcal{L}(Q)$ by varying Q reduces the negative KL divergence and therefore has the effect of making $Q(\boldsymbol{\theta})$ more similar to $p(\boldsymbol{\theta}|\mathbf{Y})$.

Choosing Q(θ)

- If we maximize the bound with respect to Q(**θ**), we are making Q(**θ**) a better and better approximation to the posterior.
- Need to choose a form of Q(**θ**) that makes it relatively easy(ier) to maximize the bound:

$$\mathcal{L}(Q) = \int Q(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{\theta}, \mathbf{Y})}{Q(\boldsymbol{\theta})} \mathrm{d}\boldsymbol{\theta}$$

- Tradeoff: more complex Q(**θ**) is better approximation, but makes bound harder to compute; less complex Q(**θ**) is poorer approximation but easier to optimize.

Choosing Q(θ)

- Independence assumption: $Q(\boldsymbol{\theta}) = \prod_{l=1}^{L} Q_l(\boldsymbol{\theta}_l)$

  l is set of parameters or latent variables

- Could be split many ways:
  - W parameters, X latent vectors

$$Q(\mathbf{W}, \mathbf{X}) = Q_{\mathbf{W}}(\mathbf{W}) Q_{\mathbf{X}}(\mathbf{X})$$

$$Q_{\mathbf{W}}(\mathbf{W}) = \prod_{m=1}^{M} Q_{\mathbf{w}_m}(\mathbf{w}_m), \quad \text{and/or} \quad Q_{\mathbf{X}}(\mathbf{X}) = \prod_{n=1}^{N} Q_{\mathbf{x}_n}(\mathbf{x}_n)$$

$$Q_{\mathbf{X}}(\mathbf{X}) = \prod_{n=1}^{N} \prod_{d=1}^{D} Q_{x_{nd}}(x_{nd})$$

- All parameters are likely to be dependent; more independence likely worsens approx.

Choosing & Optimizing Q(θ)

- Independence assumption: $Q(\boldsymbol{\theta}) = \prod_{l=1}^{L} Q_l(\boldsymbol{\theta}_l)$

  l is set of parameters or latent variables

- Optimizing Q(**θ**) (given the indep. assumption):

$$Q_l(\boldsymbol{\theta}_l) = \frac{\exp\left(\mathbf{E}_{k \neq l}\left\{\log p(\mathbf{Y}, \boldsymbol{\theta})\right\}\right)}{\int \exp\left(\mathbf{E}_{k \neq l}\left\{\log p(\mathbf{Y}, \boldsymbol{\theta})\right\}\right) d\boldsymbol{\theta}_l}$$

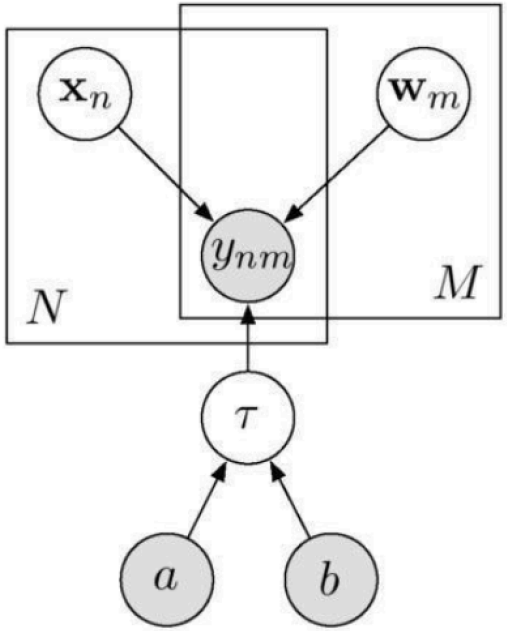
- Algorithm for computing each $Q_l(\boldsymbol{\theta}_l)$ is much like EM: take expectation with respect to each other $Q_k(\boldsymbol{\theta}_k)$.

Probabilistic model for PCA

- Observe $n = 1 \ldots N$ M-dimensional input $\mathbf{y}_n$
- Want to find D-dimensional representation of $\mathbf{x}_n$ (where D < M)

$$\mathbf{y}_n = \mathbf{W}\mathbf{x}_n + \mathbf{v}$$

- **W** is an MxD matrix, **v** is an Mx1 noise vector



$$p(\mathbf{x}_n) = \mathcal{N}(\mathbf{0}, \mathbf{I}_D)$$

$$p(\mathbf{W}) = \prod_{m=1}^{M} p(\mathbf{w}_m)$$

$$p(\mathbf{w}_m) = \mathcal{N}(\mathbf{0}, \mathbf{I}_D)$$

$$p(y_{nm}) = \mathcal{N}(\mathbf{w}_m^{\mathsf{T}}\mathbf{x}_n, \tau^{-1})$$

$$p(\tau|a,b) = \Gamma(a,b) = \frac{b^a \tau^{a-1} e^{-b\tau}}{\Gamma(a)}$$

Precision (as opposed to variance) $(\tau^{-1} = \sigma^2)$

VB for Prob. PCA Inference

- Use Variational Bayes to infer an approximate posterior over $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top$.
- Need to decompose $Q(\mathbf{W}, \mathbf{X}, \tau)$

$$Q(\boldsymbol{\theta}) = \prod_{l=1}^{L} Q_l(\boldsymbol{\theta}_l)$$

$$Q(\mathbf{W}, \mathbf{X}, \tau) = Q_\tau(\tau) \left[\prod_{n=1}^{N} Q_{\mathbf{x}_n}(\mathbf{x}_n)\right] \left[\prod_{m=1}^{M} Q_{\mathbf{w}_m}(\mathbf{w}_m)\right]$$

$$Q_l(\boldsymbol{\theta}_l) = \frac{\exp\left(\mathbf{E}_{k \neq l}\left\{\log p(\mathbf{Y}, \boldsymbol{\theta})\right\}\right)}{\int \exp\left(\mathbf{E}_{k \neq l}\left\{\log p(\mathbf{Y}, \boldsymbol{\theta})\right\}\right) d\boldsymbol{\theta}_l}$$

$$p(\mathbf{Y}, \mathbf{X}, \mathbf{W}, \tau) = p(\tau|a, b) \left[\prod_{m=1}^{M} p(\mathbf{w}_m)\right] \left[\prod_{n=1}^{N} p(\mathbf{x}_n) p(\mathbf{y}_n|\mathbf{W}, \mathbf{x}_n, \tau)\right]$$

$$\log p(\mathbf{Y}, \mathbf{X}, \mathbf{W}, \tau) = \log p(\tau|a, b) + \sum_{m=1}^{M} \log p(\mathbf{w}_m) + \sum_{n=1}^{N} \log p(\mathbf{x}_n) + \sum_{n=1}^{N} \log p(\mathbf{y}_n|\mathbf{W}, \mathbf{x}_n, \tau).$$

VB for Prob. PCA Inference

$$\log p(\mathbf{Y}, \mathbf{X}, \mathbf{W}, \tau) = \log p(\tau|a,b) + \sum_{m=1}^{M} \log p(\mathbf{w}_m) + \sum_{n=1}^{N} \log p(\mathbf{x}_n) + \sum_{n=1}^{N} \sum_{m=1}^{M} \log p(y_{nm}|\mathbf{w}_m, \mathbf{x}_n, \tau)$$

Assume independent noise vector (diagonal matrix)

$$p(y_{nm}|\mathbf{w}_m, \mathbf{x}_n, \tau) = \mathcal{N}(\mathbf{w}_m^{\mathsf{T}}\mathbf{x}_n, \tau^{-1})$$

Now need expressions for:

$$Q_\tau(\tau) \qquad Q_{\mathbf{x}_n}(\mathbf{x}_n) \qquad Q_{\mathbf{w}_m}(\mathbf{w}_m)$$

$$\log p(\mathbf{Y}, \mathbf{X}, \mathbf{W}, \tau) = \log p(\tau|a, b) + \sum_{m=1}^{M} \log p(\mathbf{w}_m) + \sum_{n=1}^{N} \log p(\mathbf{x}_n) + \sum_{n=1}^{N} \sum_{m=1}^{M} \log p(y_{nm}|\mathbf{w}_m, \mathbf{x}_n, \tau)$$

Derive $Q_\tau(\tau)$

$$Q_l(\boldsymbol{\theta}_l) = \frac{\exp\left(\mathbf{E}_{k \neq l}\left\{\log p(\mathbf{Y}, \boldsymbol{\theta})\right\}\right)}{\int \exp\left(\mathbf{E}_{k \neq l}\left\{\log p(\mathbf{Y}, \boldsymbol{\theta})\right\}\right) d\boldsymbol{\theta}_l}$$

$$Q_\tau(\tau) \propto \exp\left(\mathbf{E}_{Q_\mathbf{X}(\mathbf{X}) Q_\mathbf{W}(\mathbf{W})}\left\{\log p(\mathbf{Y}, \mathbf{X}, \mathbf{W}, \tau)\right\}\right)$$

$$\log p(\mathbf{Y}, \mathbf{X}, \mathbf{W}, \tau) \propto a \log b + (a-1) \log \tau - b\tau - \log \Gamma(a) - \frac{NM}{2} \log 2\pi + \frac{NM}{2} \log \tau - \frac{\tau}{2} \sum_n \sum_m (y_{nm} - \mathbf{w}_m^\mathsf{T} \mathbf{x}_n)^2$$

$$
\begin{aligned}
Q_\tau(\tau) &\propto \exp\left(\mathbf{E}_{Q_{\mathbf{X}}(\mathbf{X})Q_{\mathbf{W}}(\mathbf{W})}\left\{\log p(\mathbf{Y},\mathbf{X},\mathbf{W},\tau)\right\}\right) \\
&\propto \exp\left((a-1)\log\tau - b\tau + \frac{NM}{2}\log\tau\right) \\
&\quad \times \exp\left(-\frac{\tau}{2}\mathbf{E}_{Q_{\mathbf{X}}(\mathbf{X})Q_{\mathbf{W}}(\mathbf{W})}\left\{\sum_n\sum_m (y_{nm} - \mathbf{w}_m^\mathsf{T}\mathbf{x}_n)^2\right\}\right).
\end{aligned}
$$

$$
\mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)Q_{\mathbf{w}_m}(\mathbf{w}_m)}\left\{y_{nm}^2\right\} = y_{nm}^2.
$$

$$
\exp\left(-\frac{\tau}{2}\sum_{n,m}\left(y_{nm}^2 + \mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)Q_{\mathbf{w}_m}(\mathbf{w}_m)}\left\{-2\mathbf{w}_m^\mathsf{T}\mathbf{x}_n + \mathbf{x}_n^\mathsf{T}\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\mathbf{x}_n\right\}\right)\right).
$$

$$
\mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)Q_{\mathbf{w}_m}(\mathbf{w}_m)}\left\{-2\mathbf{w}_m^\mathsf{T}\mathbf{x}_n\right\} = -2\mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)}\left\{\mathbf{x}_n\right\}^\mathsf{T}\mathbf{E}_{Q_{\mathbf{w}_m}(\mathbf{w}_m)}\left\{\mathbf{w}_m\right\}
$$

$$
\mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)Q_{\mathbf{w}_m}(\mathbf{w}_m)}\left\{-2\mathbf{w}_m^\mathsf{T}\mathbf{x}_n\right\} = -2\left\langle\mathbf{x}_n\right\rangle^\mathsf{T}\left\langle\mathbf{w}_m\right\rangle
$$

$$\begin{aligned}\mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)Q_{\mathbf{w}_m}(\mathbf{w}_m)}\left\{\mathbf{x}_n^\mathsf{T}\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\mathbf{x}_n\right\} &= \mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)}\left\{\mathbf{x}_n^\mathsf{T}\left\langle\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\right\rangle\mathbf{x}_n\right\} \\ &= \left\langle\mathbf{x}_n^\mathsf{T}\left\langle\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\right\rangle\mathbf{x}_n\right\rangle.\end{aligned}$$

Putting everything back together, we have

$$\begin{aligned}Q_\tau(\tau) \propto \exp\Bigg((a-1)\log\tau - b\tau + \frac{NM}{2}\log\tau \\ - \frac{\tau}{2}\sum_{n,m}\left(y_{nm}^2 - 2\left\langle\mathbf{w}_n\right\rangle^\mathsf{T}\left\langle\mathbf{x}_n\right\rangle + \left\langle\mathbf{x}_n^\mathsf{T}\left\langle\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\right\rangle\mathbf{x}_n\right\rangle\right)\Bigg).\end{aligned}$$

This can be written as

$$Q_\tau(\tau) \propto \tau^{e-1}\exp\{-\tau f\},$$

where

$$\begin{aligned}e &= a + \frac{NM}{2} \\ f &= b + \frac{1}{2}\sum_{n,m}\left(y_{nm}^2 - 2\left\langle\mathbf{w}_n\right\rangle^\mathsf{T}\left\langle\mathbf{x}_n\right\rangle + \left\langle\mathbf{x}_n^\mathsf{T}\left\langle\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\right\rangle\mathbf{x}_n\right\rangle\right).\end{aligned}$$

$$\log p(\mathbf{Y}, \mathbf{X}, \mathbf{W}, \tau) = \log p(\tau|a, b) + \sum_{m=1}^{M} \log p(\mathbf{w}_m) + \sum_{n=1}^{N} \log p(\mathbf{x}_n) + \sum_{n=1}^{N} \sum_{m=1}^{M} \log p(y_{nm}|\mathbf{w}_m, \mathbf{x}_n, \tau)$$

Derive $Q_{\mathbf{x}_n}(\mathbf{x}_n)$

$$Q_l(\boldsymbol{\theta}_l) = \frac{\exp\left(\mathbf{E}_{k \neq l}\left\{\log p(\mathbf{Y}, \boldsymbol{\theta})\right\}\right)}{\int \exp\left(\mathbf{E}_{k \neq l}\left\{\log p(\mathbf{Y}, \boldsymbol{\theta})\right\}\right) d\boldsymbol{\theta}_l}$$

$$Q_{\mathbf{x}_n}(\mathbf{x}_n) \propto \exp\left(\mathbf{E}_{Q_{\mathbf{W}}(\mathbf{W}) Q_\tau(\tau)}\left\{\log p(\mathbf{x}_n) + \sum_{m=1}^{M} p(y_{nm}|\mathbf{w}_m, \mathbf{x}_n, \tau)\right\}\right)$$

$$
\begin{aligned}
Q_{\mathbf{x}_n}(\mathbf{x}_n) &\propto \exp\left(\mathbf{E}_{Q_{\mathbf{W}}(\mathbf{W})Q_\tau(\tau)}\left\{-\frac{1}{2}\mathbf{x}_n^\mathsf{T}\mathbf{x}_n\right.\right. \\
&\qquad \left.\left.-\frac{1}{2}\tau\sum_m\left(-2y_{nm}\mathbf{x}_n^\mathsf{T}\mathbf{w}_m+\mathbf{x}_n^\mathsf{T}\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\mathbf{x}_n\right)\right\}\right) \\
&= \exp\left(-\frac{1}{2}\mathbf{x}_n^\mathsf{T}\mathbf{x}_n-\frac{1}{2}\langle\tau\rangle\sum_m\left(-2y_{nm}\mathbf{x}_n^\mathsf{T}\langle\mathbf{w}_m\rangle+\mathbf{x}_n^\mathsf{T}\left\langle\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\right\rangle\mathbf{x}_n\right)\right) \\
&= \exp\left(-\frac{1}{2}\left(\mathbf{x}_n^\mathsf{T}\left[\mathbf{I}_D+\langle\tau\rangle\sum_m\left\langle\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\right\rangle\right]\mathbf{x}_n\right.\right. \\
&\qquad \left.\left.-2\langle\tau\rangle\,\mathbf{x}_n^\mathsf{T}\sum_m y_{nm}\langle\mathbf{w}_m\rangle\right)\right).
\end{aligned}
$$

The presence of the linear and quadratic terms within the expectation tells us that this is a Gaussian:

$$Q_{\mathbf{x}_n}(\mathbf{x}_n)=\mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_n},\boldsymbol{\Sigma}_{\mathbf{x}_n}).$$

Equating coefficients lets us read off expressions for $\boldsymbol{\mu}_{\mathbf{x}_n}$ and $\boldsymbol{\Sigma}_{\mathbf{x}_n}$:

$$
\begin{aligned}
\mathbf{x}_n^\mathsf{T}\boldsymbol{\Sigma}_{\mathbf{x}_n}^{-1}\mathbf{x}_n &\equiv \mathbf{x}_n^\mathsf{T}\left[\mathbf{I}_D+\langle\tau\rangle\sum_m\left\langle\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\right\rangle\right]\mathbf{x}_n \\
\boldsymbol{\Sigma}_{\mathbf{x}_n} &= \left[\mathbf{I}_D+\langle\tau\rangle\sum_m\left\langle\mathbf{w}_m\mathbf{w}_m^\mathsf{T}\right\rangle\right]^{-1} \\
-2\mathbf{x}_n^\mathsf{T}\boldsymbol{\Sigma}_{\mathbf{x}_n}^{-1}\boldsymbol{\mu}_{\mathbf{x}_n} &\equiv -2\langle\tau\rangle\,\mathbf{x}_n^\mathsf{T}\sum_m y_{nm}\langle\mathbf{w}_m\rangle \\
\boldsymbol{\mu}_{\mathbf{x}_n} &= \langle\tau\rangle\,\boldsymbol{\Sigma}_{\mathbf{x}_n}\sum_m y_{nm}\langle\mathbf{w}_m\rangle.
\end{aligned}
$$

$$\log p(\mathbf{Y}, \mathbf{X}, \mathbf{W}, \tau) = \log p(\tau|a,b) + \sum_{m=1}^{M} \log p(\mathbf{w}_m) + \sum_{n=1}^{N} \log p(\mathbf{x}_n) + \sum_{n=1}^{N}\sum_{m=1}^{M} \log p(y_{nm}|\mathbf{w}_m, \mathbf{x}_n, \tau)$$

Derive $Q_{\mathbf{w}_m}(\mathbf{w}_m)$

$$Q_l(\boldsymbol{\theta}_l) = \frac{\exp\left(\mathbf{E}_{k\neq l}\left\{\log p(\mathbf{Y}, \boldsymbol{\theta})\right\}\right)}{\int \exp\left(\mathbf{E}_{k\neq l}\left\{\log p(\mathbf{Y}, \boldsymbol{\theta})\right\}\right) d\boldsymbol{\theta}_l}$$

$$Q_{\mathbf{w}_m}(\mathbf{w}_m) \propto \exp\left(\mathbf{E}_{Q_{\mathbf{X}}(\mathbf{X})Q_\tau(\tau)}\left\{\log p(\mathbf{w}_m) + \sum_{n=1}^{N} p(y_{nm}|\mathbf{w}_m, \mathbf{x}_n, \tau)\right\}\right)$$

$$
\begin{aligned}
Q_{\mathbf{w}_m}(\mathbf{w}_m) &\propto \exp\left(\mathbf{E}_{Q_{\mathbf{X}}(\mathbf{X})Q_\tau(\tau)}\left\{-\frac{1}{2}\mathbf{w}_m^\mathsf{T}\mathbf{w}_m\right.\right. \\
&\quad \left.\left.-\frac{1}{2}\tau\sum_n\left(-2y_{nm}\mathbf{w}_m^\mathsf{T}\mathbf{x}_n+\mathbf{w}_m^\mathsf{T}\mathbf{x}_n\mathbf{x}_n^\mathsf{T}\mathbf{w}_m\right)\right\}\right) \\
&= \exp\left(-\frac{1}{2}\mathbf{w}_m^\mathsf{T}\mathbf{w}_m-\frac{1}{2}\langle\tau\rangle\sum_n\left(-2y_{nm}\mathbf{w}_m^\mathsf{T}\langle\mathbf{x}_n\rangle+\mathbf{w}_m^\mathsf{T}\left\langle\mathbf{x}_n\mathbf{x}_n^\mathsf{T}\right\rangle\mathbf{w}_m\right)\right) \\
&\propto \exp\left(-\frac{1}{2}\left(\mathbf{w}_m^\mathsf{T}\left[\mathbf{I}_D+\langle\tau\rangle\sum_n\left\langle\mathbf{x}_n\mathbf{x}_n^\mathsf{T}\right\rangle\right]\mathbf{w}_m\right.\right. \\
&\quad \left.\left.-2\langle\tau\rangle\mathbf{w}_m^\mathsf{T}\sum_n y_{nm}\langle\mathbf{x}_n\rangle\right)\right).
\end{aligned}
$$

$$
\begin{aligned}
Q_{\mathbf{w}_m}(\mathbf{w}_m) &= \mathcal{N}(\boldsymbol{\mu}_{\mathbf{w}_m},\boldsymbol{\Sigma}_{\mathbf{w}_m}) \\
\boldsymbol{\Sigma}_{\mathbf{w}_m} &= \left[\mathbf{I}_D+\langle\tau\rangle\sum_n\left\langle\mathbf{x}_n\mathbf{x}_n^\mathsf{T}\right\rangle\right]^{-1} \\
\boldsymbol{\mu}_{\mathbf{w}_m} &= \langle\tau\rangle\boldsymbol{\Sigma}_{\mathbf{w}_m}\sum_n y_{nm}\langle\mathbf{x}_n\rangle.
\end{aligned}
$$

Moving forward …

- With the posterior factors' families identified, we can derive the required expectations:

$$\langle \mathbf{x}_n \rangle = \boldsymbol{\mu}_{\mathbf{x}_n} \qquad \left\langle \mathbf{x}_n \mathbf{x}_n^{\mathsf{T}} \right\rangle = \boldsymbol{\Sigma}_{\mathbf{x}_n} + \boldsymbol{\mu}_{\mathbf{x}_n} \boldsymbol{\mu}_{\mathbf{x}_n}^{\mathsf{T}}$$

$$\langle \mathbf{w}_m \rangle = \boldsymbol{\mu}_{\mathbf{w}_m} \qquad \left\langle \mathbf{w}_m \mathbf{w}_m^{\mathsf{T}} \right\rangle = \boldsymbol{\Sigma}_{\boldsymbol{\mu}_m} + \boldsymbol{\mu}_{\mathbf{w}_m} \boldsymbol{\mu}_{\mathbf{w}_m}^{\mathsf{T}} .$$

$$\langle \tau \rangle = \frac{e}{f} .$$

Putting it together

We now have everything we need to obtain an approximate posterior $Q(\mathbf{W}, \mathbf{X}, \tau)$ using VB. We must first initialise the various parameters. We will start by initialising $\langle \tau \rangle = a/b$ (its expected prior value) and then sample each $\langle \mathbf{w}_m \rangle$ from $\mathcal{N}(\mathbf{0}, \mathbf{I}_D)$ and compute $\left\langle \mathbf{w}\mathbf{w}^\mathsf{T} \right\rangle = \mathbf{I}_D + \langle \mathbf{w}_m \rangle \langle \mathbf{w}_m \rangle^\mathsf{T}$. We can now compute $\boldsymbol{\mu}_{\mathbf{x}_n}$ and $\boldsymbol{\Sigma}_{\mathbf{x}_n}$ and hence $\langle \mathbf{x}_n \rangle$ and $\left\langle \mathbf{x}_n \mathbf{x}_n^\mathsf{T} \right\rangle$. We proceed as follows:

1. For all n, compute $\boldsymbol{\Sigma}_{\mathbf{x}_n}$ and $\boldsymbol{\mu}_{\mathbf{x}_n}$ and update $\langle \mathbf{x}_n \rangle$ and $\left\langle \mathbf{x}_n \mathbf{x}_n^\mathsf{T} \right\rangle$.
2. Using the new values of $\langle \mathbf{x}_n \rangle$ and $\left\langle \mathbf{x}_n \mathbf{x}_n^\mathsf{T} \right\rangle$, compute $\boldsymbol{\mu}_{\mathbf{w}_m}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\mu}_w}$ and update $\langle \mathbf{w}_m \rangle$ and $\left\langle \mathbf{w}_m \mathbf{w}_m^\mathsf{T} \right\rangle$ for all m.
3. Compute $\left\langle \mathbf{x}_n^\mathsf{T} \left\langle \mathbf{w}_m \mathbf{w}_m^\mathsf{T} \right\rangle \mathbf{x}_n \right\rangle$ for all n and m.
4. Compute e and f and update $\langle \tau \rangle$.
5. If not converged, return to 1.

To check convergence, we can either monitor how much the various parameters are changing, or compute the bound, $\mathcal{L}(\boldsymbol{\theta})$ (Equation 7.6), which will increase until convergence and then remain unchanged. The bound is given by

$$\begin{aligned} \mathcal{L}(\mathbf{X}, \mathbf{W}, \tau) &= \int Q(\mathbf{X}, \mathbf{W}, \tau) \log \frac{p(\mathbf{Y}, \mathbf{X}, \mathbf{W}, \tau)}{Q(\mathbf{X}, \mathbf{W}, \tau)} \, dQ(\mathbf{X}, \mathbf{W}, \tau) \\ &= \int Q(\cdot) \log p(\cdot) \, dQ(\cdot) - \int Q(\cdot) \log Q(\cdot) \, dQ(\cdot). \end{aligned}$$

$$\int Q(\cdot)\log p(\cdot)\; dQ(\cdot) \;=\; \mathbf{E}_{Q_\tau(\tau)}\left\{\log p(\tau|a,b)\right\} + \sum_{n=1}^{N}\mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)}\left\{\log p(\mathbf{x}_n)\right\} + \sum_{m=1}^{M}\mathbf{E}_{Q_{\mathbf{w}_m}(\mathbf{w}_m)}\left\{\log p(\mathbf{w}_m)\right\} + \sum_{n=1}^{N}\sum_{m=1}^{M}\mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)}\left\{\log p(y_{nm}|\mathbf{x}_n,\mathbf{w}_m,\tau)\right\},$$

$$\int Q(\cdot)\log Q(\cdot)\; dQ(\cdot) \;=\; \mathbf{E}_{Q_\tau(\tau)}\left\{\log Q_\tau(\tau)\right\} + \sum_{n=1}^{N}\mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)}\left\{\log Q_{\mathbf{x}_n}(\mathbf{x}_n)\right\} + \sum_{m=1}^{M}\mathbf{E}_{Q_{\mathbf{w}_m}(\mathbf{w}_m)}\left\{\log Q_{\mathbf{w}_m}(\mathbf{w}_m)\right\}.$$

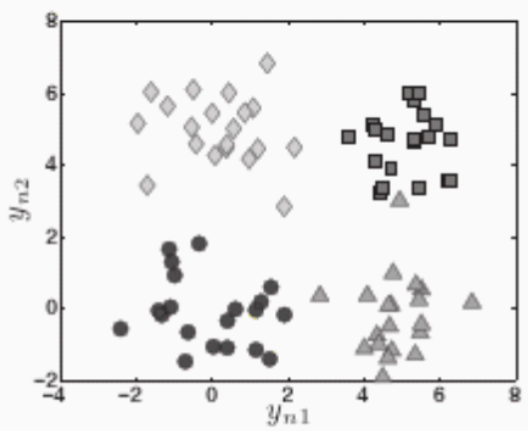
$$
\begin{aligned}
\mathcal{L}(\mathbf{X}, \mathbf{W}, \tau) \quad = \quad & a \log b + (a-1) \langle \log \tau \rangle - b \langle \tau \rangle - \log \Gamma(a) \\
& - \frac{ND}{2} \log 2\pi - \frac{1}{2} \sum_n \left(\mathrm{Tr}(\boldsymbol{\Sigma}_{\mathbf{x}_n}) + \boldsymbol{\mu}_{\mathbf{x}_n}^\mathsf{T} \boldsymbol{\mu}_{\mathbf{x}_n} \right) \\
& - \frac{MD}{2} \log 2\pi - \frac{1}{2} \sum_m \left(\mathrm{Tr}(\boldsymbol{\Sigma}_{\mathbf{w}_m}) + \boldsymbol{\mu}_{\mathbf{w}_m}^\mathsf{T} \boldsymbol{\mu}_{\mathbf{w}_m} \right) \\
& - \frac{NM}{2} \log 2\pi + \frac{NM}{2} \langle \log \tau \rangle - \frac{1}{2} \langle \tau \rangle \sum_{n,m} \left\langle (y_{nm} - \mathbf{w}_m^\mathsf{T} \mathbf{x}_n)^2 \right\rangle \\
& - (e \log f + (e-1) \langle \log \tau \rangle - f \langle \tau \rangle - \log \Gamma(e)) \\
& - \left(- \frac{ND}{2} \log 2\pi - \frac{ND}{2} - \frac{1}{2} \sum_n \log |\boldsymbol{\Sigma}_{\mathbf{x}_n}| \right) \\
& - \left(- \frac{MD}{2} \log 2\pi - \frac{MD}{2} - \frac{1}{2} \sum_m \log |\boldsymbol{\Sigma}_{\mathbf{w}_m}| \right)
\end{aligned}
$$

$$
\left\langle (y_{nm} - \mathbf{w}_m^\mathsf{T} \mathbf{x}_n)^2 \right\rangle = y_{nm}^2 - 2 y_{nm} \langle \mathbf{x}_n \rangle^\mathsf{T} \langle \mathbf{w}_m \rangle + \left\langle \mathbf{x}_n^\mathsf{T} \left\langle \mathbf{w}_m \mathbf{w}_m^\mathsf{T} \right\rangle \mathbf{x}_n \right\rangle
$$

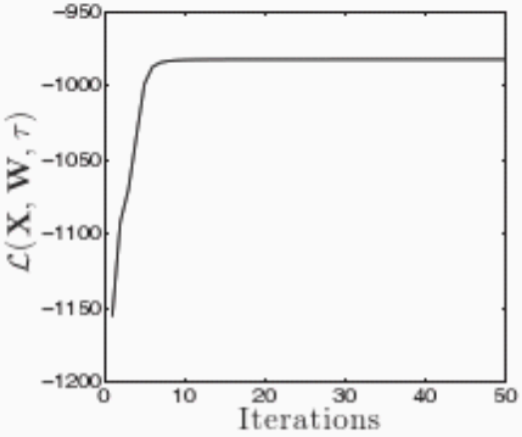
If we take S samples, $\tau^1, \ldots, \tau^S$, the approximation is given by

$$
\langle \log \tau \rangle \approx \frac{1}{S} \sum_{s=1}^{S} \log \tau^s .
$$

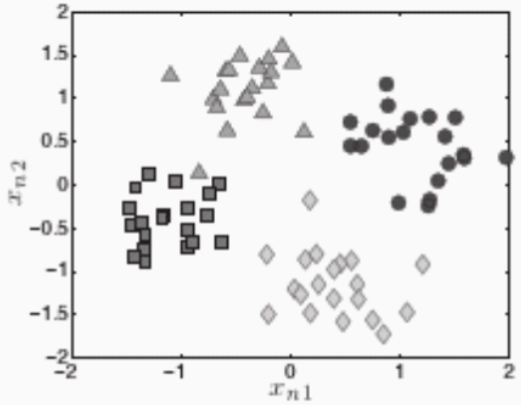
An example of PPCA



(a) First two dimensions of the data objects $\mathbf{y}_n$.



(b) Evolution of the lower bound $\mathcal{L}(\mathbf{X}, \mathbf{W}, \tau)$.



(c) The posterior mean of the latent variables.

PCA vs PPCA (VB)

- PCA is based purely on algebraic manipulations: find a new basis where each orthogonal component maximizes the variance
- PPCA models the vectors in the input space as latent-random variables and finds them by the means of maximizing the likelihood
- VB allows you to deal with missing values by using indicator variables
- VB also allows for non-real-valued data