CS 3750 Machine Learning

Probabilistic PCA & extensions

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Principal Component Analysis

- Used to transform observed data matrix **X** (N x d) into **Y** (N x q) (find the q principal components)
  - Fairly simple solution:
    1. Centralize the **X**
    2. Calculate the covariance matrix **C** of **X**
    3. Calculate the eigenvectors of the **C**
    4. Select the dimensions that correspond to the q highest eigenvalues
  - Big win for linear algebra.

Limitations of PCA

- PCA is a simple linear algebra transformation, it does not produce a probabilistic model for the observed data.
  - A probabilistic model can be very useful
- The variance-covariance matrix needs to be calculated
  - Can be very computation-intensive for large datasets with a high # of dimensions
- Does not deal properly with missing data
  - Incomplete data must either be discarded or imputed using ad-hoc methods
- Outlying data observations can unduly affect the analysis

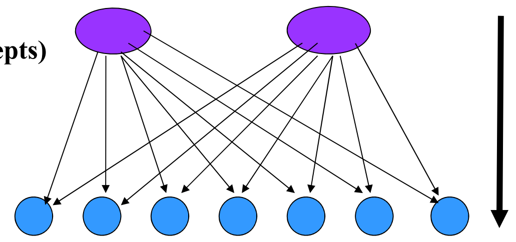
Probabilistic PCA model

- Enables comparison with other probabilistic techniques
- Facilitates statistical testing
- Maximum-likelihood estimates can be computed for elements associated with principal components
- Permits the application of Bayesian methods
- Extends the scope of PCA
  - Multiple PCA models can be combined as a probabilistic mixture
  - PCA projections can be obtained when some data values are missing
- Can be utilized as a constrained Gaussian density model
  - Classification
  - Novelty detection

Latent variable models

- Offer a lower dimensional representation of the data and their dependencies
- Latent variable model:
  - ***y:*** observed variables (d-dimensions)
  - ***x:*** latent variables (q-dimensions)
  - $q<d$

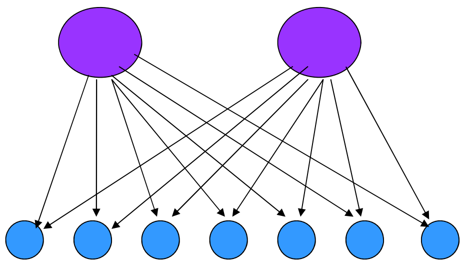
Latent variables (x) q = 2
(hidden variables, underlying concepts)



Observed variables (y) d = 7
(data)

Latent variable models

Latent variables (x) q = 2
(hidden variables, underlying concepts)



Note: Observed variables become independent of each other given latent factors

Observed variables (y) d = 7
(data)

Factor analysis

- **Latent variable model with a linear relationship:**

  $$y \sim Wx + \mu + \varepsilon$$

  - **W** is a d x q matrix that relates observed variables $\boldsymbol{y}$ to the latent variables $\boldsymbol{x}$
  - Latent variables: $\boldsymbol{x \sim N(0, I)}$
  - Error (or noise): $\boldsymbol{\varepsilon \sim N(0, \psi)}$ – Gaussian noise
  - Location term (mean): $\boldsymbol{\mu}$

  Then: $\boldsymbol{y \sim N(\mu, C_y)}$
  - where $\boldsymbol{C_y = WW^T + \psi}$ is the covariance matrix for observed variables $\boldsymbol{y}$
  - the model's parameters $\boldsymbol{W, \mu}$ and $\boldsymbol{\psi}$ can be found using maximum likelihood estimate

Probabilistic PCA (PPCA)

- A special case of the factor analysis model
  - Noise variances constrained to be equal ($\psi_i = \sigma^2$)

  $$y \sim Wx + \mu + \varepsilon$$

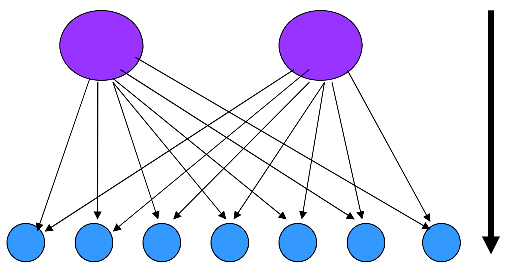
  - Latent variables: $\boldsymbol{x \sim N(0, I)}$
  - Error (or noise): $\boldsymbol{\varepsilon \sim N(0, \sigma^2 I)}$ (isotropic noise model)
  - Location term (mean): $\boldsymbol{\mu}$

  - $\boldsymbol{y|x \sim N(WX + \mu, \sigma^2 I)}$
  - $\boldsymbol{y \sim N(\mu, C_y)}$
  - where $\boldsymbol{C_y = WW^T + \sigma^2 I}$ is the covariance matrix of $\boldsymbol{y}$

- Normal PCA is a limiting case of probabilistic PCA, taken as the limit as the covariance of the noise becomes infinitesimally small ($\boldsymbol{\psi} = \lim_{\sigma^2 \to 0} \sigma^2 \boldsymbol{I}$)

Illustration of probabilistic PCA

Latent variables (x) q = 2
(hidden variables, underlying concepts)



Observed variables (y) d = 7
(data)

x ~ N(0, I)

$\downarrow$

Remapping: ***Wx***
(Weight matrix: ***W****)*

+

μ (location parameter)

+

Random error (noise): ε
ε ~ N(0, σ²I)

$\boldsymbol{y = Wx + \mu + \varepsilon}$
$\boldsymbol{y \sim N(\mu, WW^T + \sigma^2 I)}$

Parameters of interest: W (weight matrix), σ^2 *(variance of noise)*

PPCA (Maximum likelihood PCA)

- Log-likelihood for the Gaussian noise model:
  - $L = -\frac{N}{2}\{d\ln(2\pi) + \ln|\boldsymbol{C}_y| + \mathrm{tr}(\boldsymbol{C}_y{}^{-1}\mathbf{S})\}$

    $\boldsymbol{C_y = WW^T + \sigma^2}$
- Maximum likelihood estimates for the above:
  - ***μ:*** mean of the data
  - ***S*** (sample covariance matrix of the observations ***Y***):

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{Y}_n - \boldsymbol{\mu})(\mathbf{Y}_n - \boldsymbol{\mu})^{\mathrm{T}}$$

- MLE's for ***W*** and $\boldsymbol{\sigma^2}$ can be solved in two ways:
  - closed form (Tipping and Bishop)
  - EM algorithm (Roweis)

Tr(A) = sum of diagonal elements of A

Probabilistic PCA

The likelihood is maximized when:

$$\mathbf{W}_{\mathrm{ML}} = \mathbf{U}_q(\sqrt[2]{\boldsymbol{\Lambda}_q - \sigma^2\mathbf{I}})\ \mathbf{R}$$

- For $\mathbf{W} = \mathbf{W}_{\mathrm{ML}}$ the maximum $\mathbf{U}_q$ is a d x q matrix where the q column vectors are the principal eigenvectors of **S.**
- $\boldsymbol{\Lambda}_q$ is a q x q diagonal matrix with corresponding eigenvalues along the diagonal.
- **R** is an arbitrary q x q orthogonal rotation matrix
- Max likelihood estimate for σ^2 is:

$$\sigma^2{}_{\mathrm{ML}} = \frac{1}{d-q}\sum_{j=q+1}^{d}\lambda_j$$

- To find the most likely model given **S**, estimate $\sigma^2{}_{\mathrm{ML}}$ and then $\mathbf{W}_{\mathrm{ML}}$ with $\mathbf{R} = \mathbf{I}$, or you can employ the EM algorithm

Derivation of MLEs

– $L = -N/2\ \{d \ln(2\pi)+\ln|C_y|+\mathrm{tr}(\boldsymbol{C}^{-1}{}_y\boldsymbol{S})\}$

The 1st derivative of LL w/ respect to $\boldsymbol{W}$:

– dL/dW =N$(\boldsymbol{C}^{-1}\boldsymbol{S}\boldsymbol{C}^{-1}\boldsymbol{W}-\boldsymbol{C}^{-1}\boldsymbol{W})$, where $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{L}\boldsymbol{V}^T = \sigma^2\boldsymbol{I}+\boldsymbol{W}\boldsymbol{W}^T$

– The stationary points are $\boldsymbol{S}\boldsymbol{C}^{-1}\boldsymbol{W} = \boldsymbol{W}$.

– Non-trivial case: $\boldsymbol{W} \neq \boldsymbol{0}, \boldsymbol{C} \neq \boldsymbol{S}$

– SVD: $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{L}\boldsymbol{V}^T$, $\boldsymbol{U}$: $d\ x\ q$ orthonormal vectors, $\boldsymbol{L}$: $q\ x\ q$ matrix of singular values, $\boldsymbol{V}$: $q\ x\ q$ orthogonal matrix,
  - $\boldsymbol{C}^{-1}\boldsymbol{W} = \boldsymbol{W}(\sigma^2\boldsymbol{I}+\boldsymbol{W}^T\boldsymbol{W})^{-1} = \boldsymbol{U}\boldsymbol{L}(\sigma^2\boldsymbol{I} + \boldsymbol{L}^2)^{-1}\boldsymbol{V}^T$

– At the stationary points:
  - $\boldsymbol{S}\boldsymbol{U}\boldsymbol{L}(\sigma^2\boldsymbol{I} + \boldsymbol{L}^2)\boldsymbol{V}^T = \boldsymbol{U}\boldsymbol{L}\boldsymbol{V}^T$
  - $\boldsymbol{S}\boldsymbol{U}\boldsymbol{L} = \boldsymbol{U}(\sigma^2\boldsymbol{I} + \boldsymbol{L}^2)\boldsymbol{L}$

– Column vectors of $\boldsymbol{U}$, $\boldsymbol{u}_j$, are eigenvectors of $\boldsymbol{S}$, with eigenvalue λ_j, such that $\sigma^2 + l_j{}^2 = \lambda_j$
  - $l_j{}^2 = (\lambda_j - \sigma^2)^{1/2}$

– (substitute into SVD) $\boldsymbol{W} = \boldsymbol{U}_q(\boldsymbol{\Lambda}_q - \boldsymbol{\sigma}^2\boldsymbol{I})\ \boldsymbol{R}$
  - $\boldsymbol{U}_q$: $d\ x\ q$ with q column eigenvectors $\boldsymbol{u}_j$ of $\boldsymbol{S}$
  - $\boldsymbol{\Lambda}_j$: $\lambda_1 \ldots \lambda_q$, ($q$ eigenvalues of $\boldsymbol{u}_j$), or $\boldsymbol{\sigma}^2$ (corresponding d-q “discarded” rows of $\boldsymbol{W}$)
  - $\boldsymbol{R}$: arbitrary orthogonal matrix, equivalent to a rotation in principal subspace (or a re-parametrization)

Derivation of MLEs (cont)

- Substitute above results into the original *likelihood* expression
- $-L = -N/2\{d \ln(2\pi) + \sum \ln(\lambda_j) + \sum \lambda_j + (d - q)\ln \sigma^2 + q\}$
    - $\lambda_1 \ldots \lambda_q$, are q non-zero eigenvalues of $\boldsymbol{u}_j$ and $\lambda_{q+1} \ldots \lambda_d$, are zero
- Taking derivative of above with respect to σ^2 and solving for zero gives:

$$\sigma^2{}_{\mathrm{ML}} = \frac{1}{d-q} \sum_{j=q+1}^{d} \lambda_j$$

Dimensionality Reduction in pPCA

- So, how do we use this to reduce the dimensionality of data?
- Consider the dimensionality reduction process in terms of the distribution of latent variables, conditioned on the observation:

$$\mathbf{x}|\mathbf{y} \sim N\left(\mathbf{M}^{-1}\mathbf{W}^{\mathrm{T}}(\mathbf{y} - \boldsymbol{\mu}), \sigma^2\mathbf{M}^{-1}\right), \text{ where}$$
$$\mathbf{M} = \mathbf{W}^{\mathrm{T}}\mathbf{W} + \sigma^2\mathbf{I}, \mathbf{M} \text{ is a } q \text{ x } q \text{ matrix}$$

- This can be summarized by its mean:

$$\langle \mathbf{x}_n | \mathbf{y}_n \rangle = \mathbf{M}^{-1}\mathbf{W}_{\mathrm{ML}}{}^{\mathrm{T}}(\mathbf{y}_n - \boldsymbol{\mu})$$

- Intuitively, the optimal reconstruction of $\mathbf{y}_n$ should be $\mathbf{W}_{\mathrm{ML}}\langle \mathbf{x}_n | \mathbf{y}_n \rangle + \boldsymbol{\mu}$. However, it is not. For $\sigma^2 > 0$ it is not an orthogonal projection of $\mathbf{y}_n$.
- If we consider the limit as $\sigma^2 \rightarrow 0$, the projection $\mathbf{W}_{\mathrm{ML}}\langle \mathbf{x}_n | \mathbf{y}_n \rangle$ does become orthogonal and is equivalent to conventional PCA, but then the density model is singular and thus undefined.
- Optimal reconstruction of the observed data may still be obtained from conditional latent mean:
    - $\mathbf{y}_n = \mathbf{W}_{\mathrm{ML}}(\mathbf{W}_{\mathrm{ML}}{}^{\mathrm{T}}\mathbf{W}_{\mathrm{ML}})^{-1}\mathbf{W}_{\mathrm{ML}}{}^{\mathrm{T}} <\mathbf{x}_n|\mathbf{y}_n> + \boldsymbol{\mu}$

Motivation behind using E-M for PCA

- Naive PCA and MLE PCA computation-heavy for high dimensional data or large data sets
- PCA does not deal properly with missing data
  - E-M algorithm estimates ML values of missing data at each iteration
- Naïve PCA uses simplistic way (distance^2 from observed data) to access covariance
  - Sensible PCA (SPCA) defines a proper covariance structure whose parameters can be estimated through the E-M algorithm

E-M algorithm (review)

- Iterative process to estimate parameters consisting of two steps for each iteration
  - Expectation (data step): complete all hidden and missing variables $\mathbf{\Theta}$ (or latent variables) from current set of parameters $\mathbf{\Theta}$
  - Maximization (likelihood step): Update set of parameters $\mathbf{\Theta}`$, using MLE, from complete set of data from previous step $\mathbf{\Theta}$
- Likelihood obtained from MLEs guaranteed to improve in successive iterations
- Continue iterations until negligible improvement is found in likelihood

EM algorithm for normal PCA

- Amounts to an iterative procedure for finding subspace spanned by the q leading eigenvectors without computing covariance
- E-step: $\boldsymbol{X} = (\boldsymbol{W}^T\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{Y}$
  – Fix subspace and project data, ***y***, into it to give values of hidden states ***x***
  – Known: ***Y***: *d*-dimensional observed data
  – Unknown (latent): ***X***: *q*-dimensional unknown states
- M-step: $W_{new} = \boldsymbol{YX}^T(\boldsymbol{XX}^T)^{-1}$
  – Fix values of hidden states and choose subspace orientation that minimizes squared reconstruction errors

EM algorithm and missing data

Data with missing obs filled out: ***x***, Complete data (with blanks not filled out): ***y***

E-step (fill in missing variables):

- If data point ***y*** is complete, then $\boldsymbol{y}^*=\boldsymbol{y}$ and ***x**** is found as usual
- If the data point ***y*** is not complete, ***x**** and ***y**** are the solution to the least squares problem. Compute ***x*** by projecting the observed data ***y*** into the current subspace.
  – For each (possibly incomplete) point ***y***, find the unique pair of points **(x*,y*)** that minimize the norm $||\boldsymbol{W}\boldsymbol{x}^*-\boldsymbol{y}^*||$.
  – Constrain ***x**** to be in the current principal subspace and ***y**** in the subspace defined by known info about ***y***
    - If ***y*** can be completely solved in system of equations, set corresponding column of ***X*** to ***x**** and the corresponding column of ***Y*** to ***y****
    - Otherwise, QR factorization can be used on a particular constraint matrix to find least squares solution

E-M algorithm and missing data (E-step)

$$\boldsymbol{W} = \begin{pmatrix} 1 & 1 \\ 1 & 0.5 \\ 2 & 1 \end{pmatrix} \quad \boldsymbol{X} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \quad \boldsymbol{Y} = \begin{pmatrix} 3 \\ 1 \\ ? \end{pmatrix}$$

$$\boldsymbol{Wx} = \boldsymbol{y}$$

$$\begin{aligned} x_1 + \quad x_2 &= 3 \\ x_1 + 0.5\,x_2 &= 1 \\ 2x_1 + \quad x_2 &= y \end{aligned} \quad \xrightarrow{\textit{solve}} \quad \boldsymbol{X^*} = \begin{pmatrix} -1 \\ 4 \end{pmatrix} \quad \boldsymbol{Y^*} = \begin{pmatrix} 3 \\ 1 \\ 2 \end{pmatrix}$$

Set $\boldsymbol{x} = (-1, 4)$, $\boldsymbol{y} = (3, 1, 2)$, proceed to M-step

If two elements are missing in $\boldsymbol{Y}$, then we use QR factorization to find the pair (x^*, y^*) with the least squares of the norm $||\boldsymbol{Wx^*} - \boldsymbol{y^*}||$, according to the constraints specified in the set of equations $\boldsymbol{Wx} = \boldsymbol{y}$.

EM for probabilistic PCA (Sensible PCA - SPCA)

- Probabilistic PCA model:
  - $\boldsymbol{Y} \sim N(\mu, \boldsymbol{WW}^T + \sigma^2\boldsymbol{I})$
- Similar to normal PCA model, the differences are:
  - We do not take the limit as σ^2 approaches 0
  - During EM iterations, data can be directly generated from the SPCA model, and the likelihood estimated from the test data set
  - Likelihood much lower for data far away from the training set, even if they are near the principal subspace
- EM algorithm steps implemented as follows:
  - E: $\boldsymbol{\beta} = \boldsymbol{W}^T(\boldsymbol{WW}^T + \sigma^2\boldsymbol{I})^{-1}$, $<x_n|y_n> = \boldsymbol{\beta}(\boldsymbol{Y} - \mu)$, $\boldsymbol{\Sigma_x} = n\boldsymbol{I} - n\boldsymbol{\beta W} + <x_n|y_n><x_n|y_n>^T$
    - Log-likelihood in terms of weight matrix $\boldsymbol{W}$, and a *centered* observed data matrix $\boldsymbol{Y} - \mu$, noise covariance $\sigma^2\boldsymbol{I}$, and conditional latent mean $<x_n|y_n>$
  - M: $\boldsymbol{W}^{new} = (\boldsymbol{Y} - \mu) <x_n|y_n>^T \boldsymbol{\Sigma_x}^{-1}$, $\sigma^{2\,new} = trace[\boldsymbol{XX}^T - \boldsymbol{W}<x_n|y_n>(\boldsymbol{Y} - \mu)^T]/n^2$
    - Differentiate LL in terms of $\boldsymbol{W}$ and σ^2 and set to zero.

Advantages of using EM algorithm in probabilistic PCA models

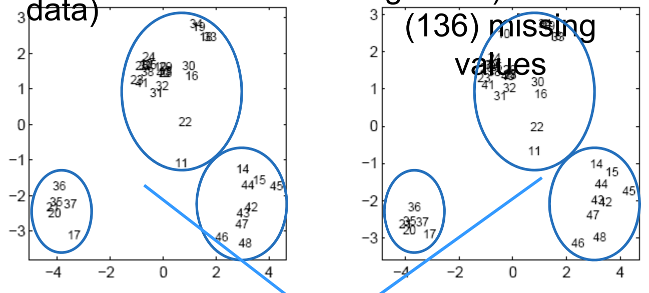
- Convergence:
  - Tipping and Bishop showed (1997) that the only stable local extremum is the *global maximum* at which the true principal subspace is found
- Complexity:
  - Methods that explicitly compute the sample covariance matrix have complexities *O(nd*^{2}*)*
  - EM algorithm does not require computation of sample covariance matrix, *O(dnq)*
    - Huge advantage when $q << d$ (# of principal components is much smaller than original # of variabes)

EM algorithm for PPCA (illustration)

Example: 38 observations (with 18 data points each) from *Tobamovirus* data set (Ripley, 1996)

Standard PCA (on complete data)

Probabilistic PCA (using EM algorithm) with 20% (136) missing values



3 clusters

Other methods for PCA

- Power iteration methods
  - Iteratively update eigenvector estimates through repeated multiplication by matrix to be diagonalized
  - Extremely inefficient to calculate explicitly ($O(nq^2)$)
  - E-M algorithm provides efficient way to obtain sample covariance matrix, without explicitly calculating it
  - Iterative methods to compute SVD are closely related to the EM algorithm
- Learning methods for the principal subspace
  - Sanger's and Oja's rule
  - Typically require more iterations and the learning parameter to be set by hand

Mixtures of probabilistic PCAs

- A combination of local probabilistic PCA models
- Multiple plots may reveal more complex data structures than a PCA projection alone
- Applications:
  - Image compression (Dony and Haykin 1995)
  - Visualization (Bishop and Tipping, 1998)
- Clustering mechanisms of mixture PPCA:
  - Local linear dimensionality reduction
  - Semi-parametric density estimation

Mixtures of probabilistic PCAs

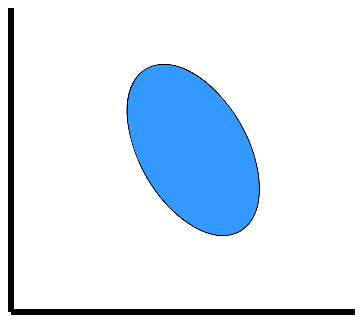
- $\mathrm{LL} = \sum_{n=1}^{N} \ln\{p(\boldsymbol{y}_n)\} = \sum_{n=1}^{N} \ln\{\sum_{i=1}^{M} \boldsymbol{\pi}_i\, p(\boldsymbol{y}_n|i)\}$
  - $p(\boldsymbol{y}|i)$ is a single PPCA model and $\boldsymbol{\pi}_i$ is the corresponding mixing proportion
  - Different mean vectors μ_i, weighting matrices $\boldsymbol{W}_i$, and noise error parameters σ_i^2 for each of M probabilistic PCA models
- An iterative EM algorithm can be used to solve for parameters
- Guaranteed to find a *local* maximum of the log-likelihood

Information Recovery

- PCA minimizes the sum of squared distances from x to its back-projection from the lower dimensional space.
- However,
  - This loss function is not a good fit when the data are not real-valued
  - Using standard PCA will do a bad job reconstructing these types of data

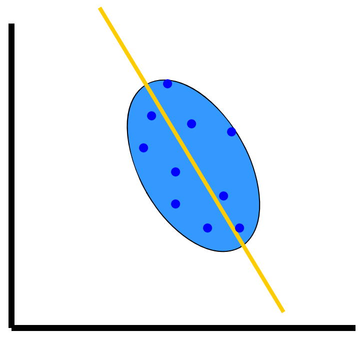
PCA’s weakness

- PCA assumes a Gaussian distribution for the random variable x



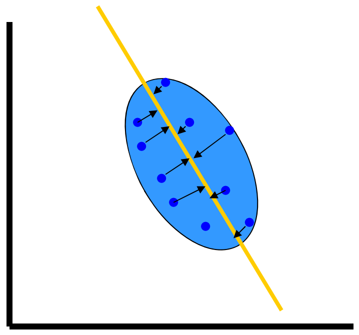
PCA’s weakness

- Gaussian noise is added to the samples from the Gaussian distribution.



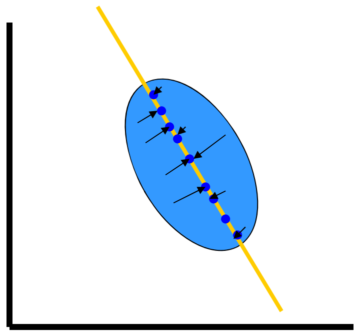
PCA's weakness

- For real-valued data this is not a problem in general.



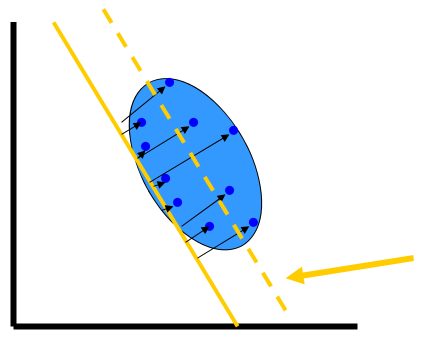
PCA's weakness

- The loss function is appropriately measured in both directions



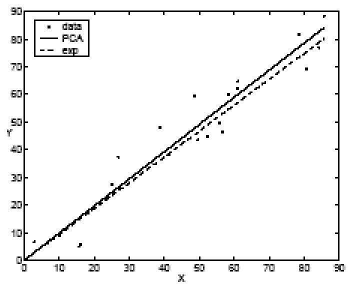
PCA’s weakness

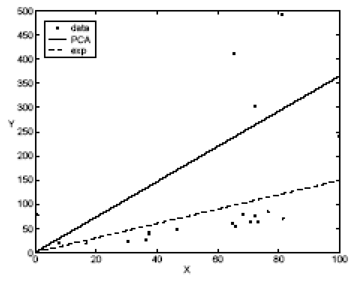
- What if the noise is known to be all positive?



Which loss function to use?

- Maybe a different loss function is better, but which?





Exponential PCA

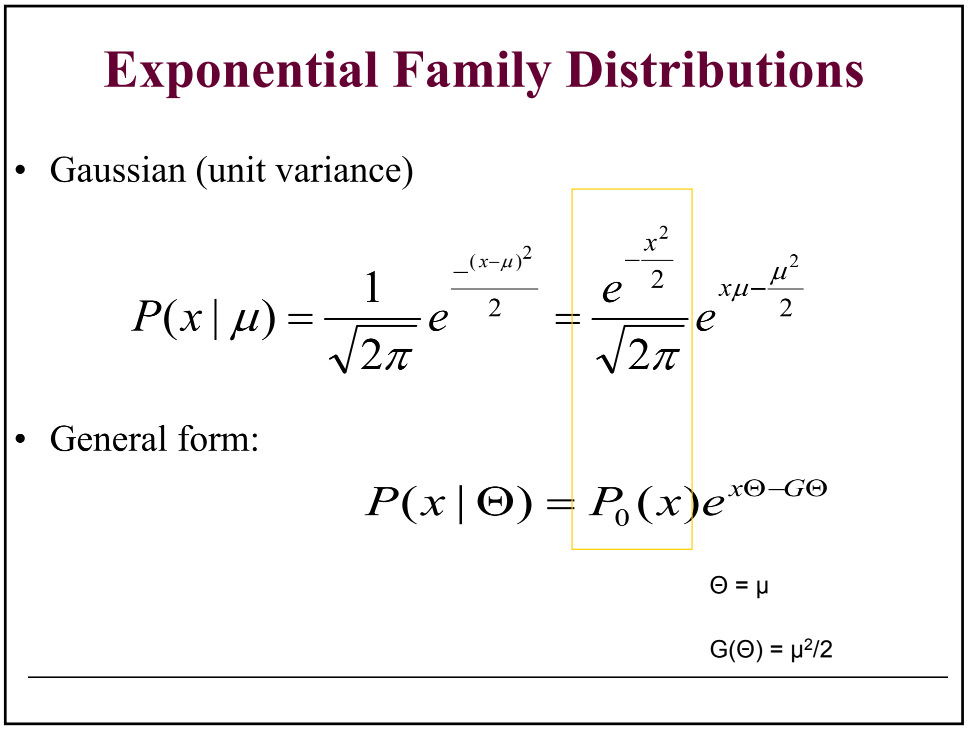
- **General idea:**
  - Extend PCA to include the entire family of exponential family distributions.
  - The unique properties of the modelling distribution for features determines the loss function for that data component automatically.
  - There's a trick which allows easy optimization of the loss function.

Exponential Family Distributions

- Exponential Family distributions can be rewritten as:

$$P(x \mid \Theta) = P_0(x) e^{x\Theta - G(\Theta)}$$

- X is your data in the high-dimensional space
- Θ is the natural (or canonical) parameterization of the distribution
- $P_0(x)$ is a constant (not dependent on Θ)
- $G(\Theta)$ is the partition function (assures a valid distribution)



Exponential Family Distributions

- Gaussian (unit variance)

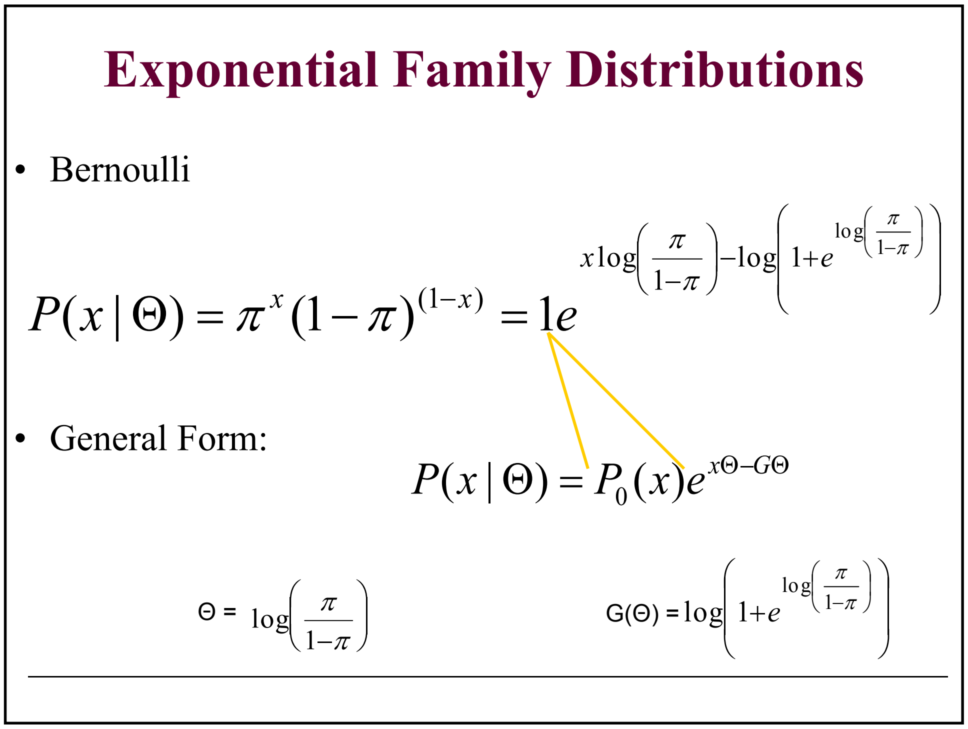
$$P(x\mid\mu)=\frac{1}{\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2}}=\boxed{\frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}}}e^{x\mu-\frac{\mu^2}{2}}$$

- General form:

$$P(x\mid\Theta)=\boxed{P_0(x)}e^{x\Theta-G\Theta}$$

$\Theta = \mu$

$G(\Theta) = \mu^2/2$



Exponential Family Distributions

- Bernoulli

$$P(x\mid\Theta)=\pi^x(1-\pi)^{(1-x)}=1e^{x\log\left(\frac{\pi}{1-\pi}\right)-\log\left(1+e^{\log\left(\frac{\pi}{1-\pi}\right)}\right)}$$

- General Form:

$$P(x\mid\Theta)=P_0(x)e^{x\Theta-G\Theta}$$

$\Theta = \log\left(\frac{\pi}{1-\pi}\right)$

$G(\Theta) = \log\left(1+e^{\log\left(\frac{\pi}{1-\pi}\right)}\right)$

Exponential Family Distributions

- Basic idea: With manipulation, you only need $P_0(x)$, Θ and $G(\Theta)$ to define an exponential distribution.
- Now take the log of $P(x|\Theta)$:

$$P(x \mid \Theta) = P_0(x)e^{x\Theta - G\Theta}$$

$$\log P(x \mid \Theta) = \log(P_0(x)) + x\Theta - G(\Theta)$$

- $G(\Theta)$ is the cummulant function of $P(x|\Theta)$
- This means that $\nabla G(\Theta)$ is the expected value of x.

So what?

- For any model Θ, we can find the expectation of the data x given Θ.
- We compare the expectation to the observed data to measure how much our model is losing in the representation.
- In this way, $G(\Theta)$ can be seen as a sort of information loss function.

Optimization

- If we want a better model, we need the information loss from that model to be lower.
- It would be cool if we could maximize log(p(x| Θ)), since it gets penalized for loss.
- Turns out that a dual problem exists for optimizing the loglikelihood.

Bregman Divergence

- Your model: *p* (a set of parameters)
- You want to know: is *q* (a set of parameters for a similar model) a better fitting model?
- Assume a convex differentiable projection function *F* defined on a convex space → projects to a convex space
- Bregman divergence:

$$D_F^{\mathrm{f}}(p,q) = F(p) - F(q) - \langle \nabla F(q), p - q \rangle.$$

- the difference between the value of *F* at point *p* and the value of the first-order Taylor expansion of *F* around point *q* evaluated at point *p*

Strategy: the distance in the new convex space represents the loss. Optimizing the distance results in better estimates of expectation parameters for the model.

Bregman Divergence

- The function F is derived from G(Θ) as a dual problem (Azoury & Warmuth, 2001):

$$F(g(\Theta)) + G(\Theta) = g(\Theta)\Theta$$
$$g(\Theta) = \nabla_\Theta G(\Theta)$$

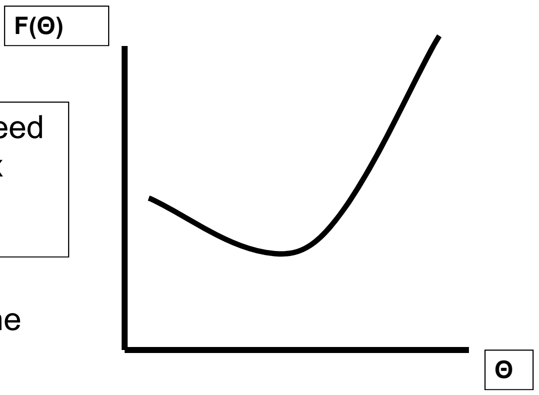
- The dual creates a "link" function g which maps between natural and expectation parameter space

Derivatives:

$$f(x) = g^{-1}(x)$$
$$f(x) = F'(x)$$

Bregman Divergence

$$D_F^q(p,q) = F(p) - F(q) - \langle \nabla F(q), p - q \rangle.$$

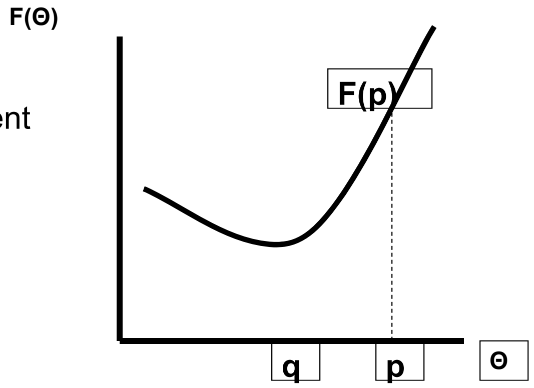


F is guaranteed to be convex due to its construction from a G function in the exponential family.

Bregman Divergence

$$D_F^\mu(p, q) = F(p) - F(q) - \langle \nabla F(q), p - q \rangle.$$

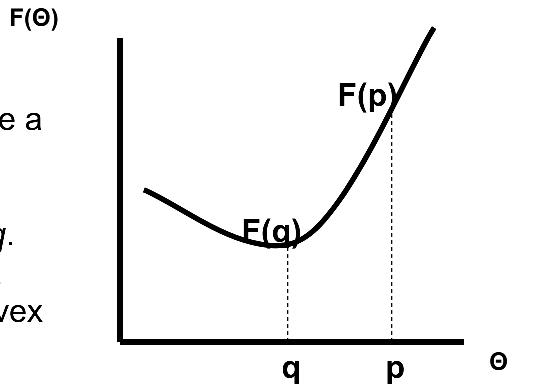
p is our current set of natural parameters.



Bregman Divergence

$$D_F^\mu(p, q) = F(p) - F(q) - \langle \nabla F(q), p - q \rangle.$$

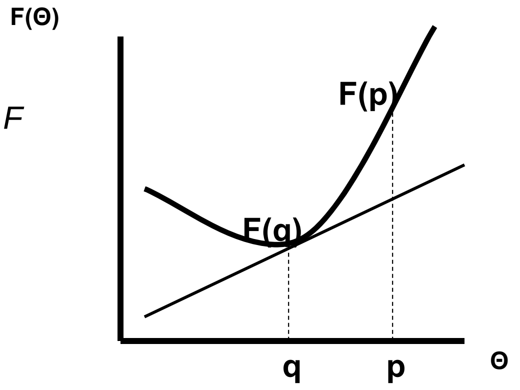
We determine a new set of natural parameters *q*. and project it onto the convex hull.



Bregman Divergence

$$D^{R}_F(p,q) = F(p) - F(q) - \langle \nabla F(q), p - q \rangle.$$

The slope of *F* at *F(q)* is measured.

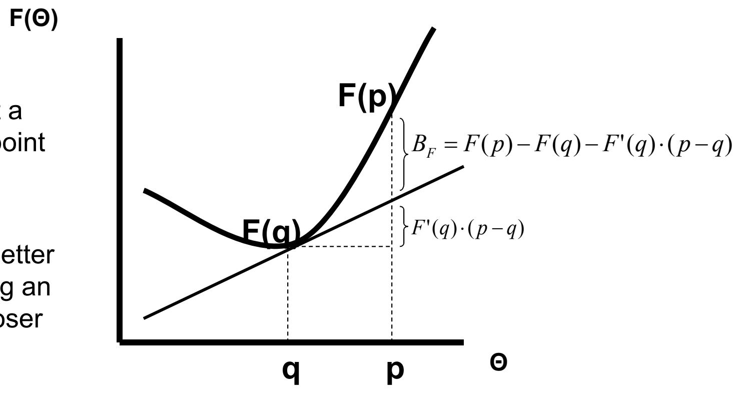


Bregman Divergence

$$D^{R}_F(p,q) = F(p) - F(q) - \langle \nabla F(q), p - q \rangle.$$

The Bregman distance B_F is higher if q is at a more convex point than p.

The bigger the distance, the better *q* is at providing an expectation closer to the data x.



Bregman Divergence

- For exponential family the function *F* is derived from G(Θ) as a dual problem (Azoury & Warmuth, 2001):

$$F(g(\Theta)) + G(\Theta) = g(\Theta)\Theta$$
$$g(\Theta) = \nabla_{\Theta} G(\Theta)$$

- The dual creates a "link" function *g* which maps between natural and expectation parameter space

Derivatives:

$$f(x) = g^{-1}(x)$$
$$f(x) = F'(x)$$

Bregman Divergence & Loglikelihood

- For the exponential family of distributions, the loglikelihood of data given model is related to a Bregman Divergence.
  - The divergence depends on which type of exponential family distribution you pick
  - Different well-known divergences are obtainable with popular choices for G(Θ)

How can it be?!

- The loglikelihood of the data given model can be rewritten as follows:

$$
\begin{aligned}
&-\log P(x\mid\Theta) = -\log(P_0(x)) - x\Theta + G(\Theta)\\
&= -\log(P_0(x)) - x\Theta + [g(\Theta)\Theta - F(g(\Theta))]\\
&= -\log(P_0(x)) - F(g(\Theta)) - x\Theta + g(\Theta)\Theta\\
&= -\log(P_0(x)) - F(g(\Theta)) - \Theta\cdot(x - g(\Theta))\\
&= -\log(P_0(x)) - F(g(\Theta)) - [g^{-1}(g(\Theta))]\cdot(x - g(\Theta))\\
&= -\log(P_0(x)) + [F(x) - F(x)] - F(g(\Theta)) - [g^{-1}(g(\Theta))]\cdot(x - g(\Theta))\\
&= -\log(P_0(x)) - F(x) + F(x) - F(g(\Theta)) - f(g(\Theta))\cdot(x - g(\Theta))\\
&= -\log(P_0(x)) - F(x) + B_F(x\,\|\,g(\Theta))
\end{aligned}
$$

Optimization

- Good news: Loglikelihood can be rewritten in terms of a Bregman divergence

$$-\log(P(x\mid\Theta) = -\log(P_0(x)) - F(x) + B_F(x\,\|\,g(\Theta))$$

- Optimizing negative loglikelihood is commonly done in EM
- Only the Bregman divergence term depends on Θ, the rest can be ignored.

Exponential PCA

- **Problem:** Find Θ's which come close to the observed data points x. (Minimize loss)
- Express the Θ's in a lower dimensionality
- **Solution:** Find a basis with *L* principal axes, represent the Θ's as a linear combination of these axes which most closely approximate x.

Generalized Exponential PCA

- Natural parameters:

$$\Theta = AV$$

- Finally, some dimensions
  - A is n * L
    - (rows of A represent the lower dimensionality representation of a data point)
  - V is L*d
    - (rows of V represent the principal axes of the model's projection basis)
  - Your data X is n*d

Generalized Exponential PCA

- Optimize the negative loglikelihood of a model given the data

$$-\log(P(x\mid\Theta) = -\log(P_0(x)) - F(x) + B_F(x \| g(\Theta))$$
$$\Theta = AV$$

  - This is equivalent to maximizing a series of Bregman divergences over the individual components of data.
  - Changing the distribution which models the loglikelihood….
    - Changes the function G(Θ), which
    - Changes the expectation parameters of the model, which
    - Changes the Bregman divergence which was derived from G(Θ), which means
    - The loss function for the data is different (the Bregman distance between x and the expectation parameters g(Θ))

Example

- Lets choose the Normal distribution
  - For a normal distribution, G(Θ) = $\Theta^2/2$
  - Therefore,
    - g(Θ) = G' (Θ) = Θ ; $g^{-1}(x)$=f(x)= x; F(x) = $x^2/2$
  - Compute the Bregman divergence between x and g(Θ):

$$B_F(p \| q) = F(p) - F(q) - f(q)\cdot(p-q)$$
$$= F(x) - F(g(\Theta)) - f(g(\Theta))\cdot(x - g(\Theta))$$
$$= \frac{x^2}{2} - \frac{\Theta^2}{2} - \Theta\cdot(x-\Theta)$$
$$= \frac{1}{2}x^2 - \frac{2}{2}\Theta x + \frac{1}{2}\Theta^2$$
$$= \frac{1}{2}(x-\Theta)^2$$

$B_F(x\|g(\Theta))$ ends up being Euclidean distance!

Example

- We want to optimize $\Theta = AV$ to fit the loss function.
- Algorithm:
  - Initialize A, V =0
  - For data = 1:n
    - For c = 1:L
      - Initialize V_c randomly
      - Until convergence,

```
For i = 1:n,    $\hat{a}_{ic} = \arg\min_{a\in\Re} \sum_j B_F(x_{ij} \| g(a v_{cj}))$
For j = 1:d,    $\hat{v}_{cj} = \arg\min_{v\in\Re} \sum_i B_F(x_{ij} \| g(\hat{a}_{ic} v))$
```

Summary:

- Use the generative model of PCA
- Extend PCA to use any partition function G(Θ)
- Convert the negative loglikelihood into a Bregman divergence
- Optimize the negative loglikelihood using an alternating update procedure over the natural parameters.