Probabilistic PCA & extensions

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

Principal Component Analysis

- Used to transform observed data matrix $\mathbf{X}$ ($N \times d$) into $\mathbf{Y}$ ($N \times q$) (find the $q$ principal components)
  - Fairly simple solution:
    1. Centralize the $\mathbf{X}$
    2. Calculate the covariance matrix $\mathbf{C}$ of $\mathbf{X}$
    3. Calculate the eigenvectors of the $\mathbf{C}$
    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)
  - $q < d$

Observed variables ($y$) $d = 7$
(Data)

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:**
  \[ y \sim Wx + \mu + \epsilon \]
  - \( W \) is a \( d \times q \) matrix that relates observed variables \( y \) to the latent variables \( x \)
  - Latent variables: \( x \sim N(0, I) \)
  - Error (or noise): \( \epsilon \sim N(0, \psi) \) – Gaussian noise
  - Location term (mean): \( \mu \)

Then:
\[ y \sim N(\mu, Cy) \]
- where \( Cy = WW^T + \psi \) is the covariance matrix for observed variables \( y \)
- the model’s parameters \( W, \mu \) and \( \psi \) can be found using maximum likelihood estimate

Probabilistic PCA (PPCA)

- A special case of the factor analysis model
  - Noise variances constrained to be equal (\( \psi_i = \sigma^2 \))
  \[ y \sim Wx + \mu + \epsilon \]
  - Latent variables: \( x \sim N(0, I) \)
  - Error (or noise): \( \epsilon \sim N(0, \sigma^2 I) \) (isotropic noise model)
  - Location term (mean): \( \mu \)

Then:
\[ y|x \sim N(WX + \mu, \sigma^2 I) \]
- \[ y \sim N(\mu, Cy) \]
  - where \( Cy = WW^T + \sigma^2 I \) is the covariance matrix of \( y \)

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

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

\[ x \sim N(\theta, I) \]

\[ \text{Remapping: } Wx \]
\[ (\text{Weight matrix: } W) \]
\[ + \]
\[ \mu \] (location parameter)
\[ + \]
\[ \text{Random error (noise): } \epsilon \]
\[ \epsilon \sim N(0, \sigma^2 I) \]

\[ y = Wx + \mu + \epsilon \]
\[ y \sim N(\mu, WW^T + \sigma^2 I) \]

Parameters of interest: \( W \) (weight matrix), \( \sigma^2 \) (variance of noise)

PPCA (Maximum likelihood PCA)

- Log-likelihood for the Gaussian noise model:
  \[ L = -\frac{N}{2} \left\{ d \ln(2\pi) + \ln|C_y| + \text{tr}(C_y^{-1}S) \right\} \]
  \[ C_y = WW^T + \sigma^2 \]

- Maximum likelihood estimates for the above:
  - \( \mu \): mean of the data
  - \( S \) (sample covariance matrix of the observations \( Y \)):
  \[ S = \frac{1}{N} \sum_{n=1}^{N} (Y_n - \mu)(Y_n - \mu)^T \]

- MLE’s for \( W \) and \( \sigma^2 \) can be solved in two ways:
  - closed form (Tipping and Bishop)
  - EM algorithm (Roweis)

\( \text{Tr}(A) = \text{sum of diagonal elements of } A \)
**Probabilistic PCA**

The likelihood is maximized when:

\[ W_{ML} = U_q (\sqrt{\Lambda_q - \sigma^2 I}) R \]

- For \( W = W_{ML} \) the maximum \( U_q \) is a \( d \times q \) matrix where the \( q \) column vectors are the principal eigenvectors of \( S \).
- \( \Lambda_q \) is a \( q \times q \) diagonal matrix with corresponding eigenvalues along the diagonal.
- \( R \) is an arbitrary \( q \times q \) orthogonal rotation matrix.
- Max likelihood estimate for \( \sigma^2 \) is:

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

- To find the most likely model given \( S \), estimate \( \sigma^2_{ML} \) and then \( W_{ML} \) with \( R = I \), or you can employ the EM algorithm.

---

**Derivation of MLEs**

- \( L = -\frac{N}{2} \{ d \ln(2\pi) + \ln|C| + \text{tr}[C^{-1}S]\} \)
  - The first derivative of \( LL \) with respect to \( W \):
  - \( \frac{dL}{dW} = -N(C^{-1}SC^{-1}W - C^{-1}W) \)
  - The stationary points are \( SC^{-1}W = W \).
  - Non-trivial case: \( W \neq 0, C \neq S \)
  - SVD: \( W = ULVT \), \( U \): \( d \times q \) orthonormal vectors, \( L \): \( q \times q \) matrix of singular values, \( V \): \( q \times q \) orthogonal matrix,
    - \( C^{-1}W = W(\sigma^2 I + L^2) \)
  - At the stationary points:
    - \( SUL(\sigma^2 I + L^2)V^T = ULVT \)
    - \( SUL = U(\sigma^2 I + L^2)L \)
  - Column vectors of \( U \), \( u_q \), are eigenvectors of \( S \), with eigenvalue \( \lambda_j \) such that \( \sigma^2 + l_j^2 = \lambda_j \)
    - \( l_j^2 = (\lambda_j - \sigma^2)^{1/2} \)
  - (substitute into SVD) \( W = U_q (A_q - \sigma^2 I) R \)
    - \( U_q \): \( d \times q \) with \( q \) column eigenvectors \( u_q \) of \( S \)
    - \( A_q: \lambda_1, \ldots, \lambda_q \) (eigenvalues of \( u_q \)), or \( \sigma^2 \) (corresponding \( d-q \) “discarded” rows of \( W \))
    - \( R \): 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
• \(-L = -N/2\{d \ln(2\pi) + \Sigma \ln(\lambda_j) + \Sigma j + (d - q)\ln \sigma^2 + q\} \)
  \(\lambda_1\ldots\lambda_q\), are \(q\) non-zero eigenvalues of \(u_j\) and \(\lambda_{q+1}\ldots\lambda_d\), are zero
• Taking derivative of above with respect to \(\sigma^2\) and solving for zero gives:
  \[ \sigma^2_{ML} = \frac{1}{d-q} \sum_{j=q+1}^{d} \lambda_j \]

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:
  \[ x|y \sim N(M^{-1}W^T(y - \mu), \sigma^2 M^{-1}) \]
• This can be summarized by its mean:
  \[ \langle x_n | y_n \rangle = M^{-1}W_{ML}^T(y_n - \mu) \]
• Intuitively, the optimal reconstruction of \(y_n\) should be \(W_{ML} \langle x_n | y_n \rangle + \mu\). However, it is not. For \(\sigma^2 > 0\) it is not an orthogonal projection of \(y_n\).
• If we consider the limit as \(\sigma^2 \to 0\), the projection \(W_{ML} \langle x_n | y_n \rangle\) 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:
  \[ y_n = W_{ML}(W_{ML}^TW_{ML})^{-1}W_{ML}^T <x_n|y_n> + \mu \]
Motivation behind using E-M for PCA

- Naïve 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 (distance² 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 \( \Theta \) (or latent variables) from current set of parameters \( \Theta \)
  - Maximization (likelihood step): Update set of parameters \( \Theta' \), using MLE, from complete set of data from previous step \( \Theta \)
- 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: $X = (WTW)^{-1}WTY$
  - 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: $W_{\text{new}} = YX^TXX^T^{-1}$
  - 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 $y^* = y$ 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 $||Wx^*-y^*||$.
  - 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)

\[
W = \begin{pmatrix}
1 & 1 \\
1 & 0.5 \\
2 & 1
\end{pmatrix} \\
X = \begin{pmatrix}
x_1 \\
x_2
\end{pmatrix} \\
Y = \begin{pmatrix}
3 \\
1 \\
?
\end{pmatrix}
\]

\(Wx = y\)

\[
x_1 + x_2 = 3 \\
x_1 + 0.5x_2 = 1 \\
2x_1 + x_2 = y
\]

\[\text{s}olve\]

\[
X^* = \begin{pmatrix}
-1 \\
4
\end{pmatrix} \\
Y^* = \begin{pmatrix}
3 \\
1 \\
2
\end{pmatrix}
\]

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

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

---

EM for probabilistic PCA
(Sensible PCA - SPCA)

- Probabilistic PCA model:
  - \(Y \sim N(\mu, WW^T + \sigma^2 I)\)
- Similar to normal PCA model, the differences are:
  - We do not take the limit as \(\sigma^2\) 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:** \(\beta = W^T(WW^T + \sigma^2 I)^{-1} <x_n|y_n> = \beta(Y-\mu)\), \(\Sigma_x = nI - n\beta W + <x_n|y_n><x_n|y_n>^T\)
    - Log-likelihood in terms of weight matrix \(W\), and a centered observed data matrix \(Y-\mu\), noise covariance \(\sigma^2 I\), and conditional latent mean \(<x_n|y_n>\)
  - **M:** \(W^{new} = (Y-\mu)<x_n|y_n>^T \Sigma_x^{-1}, \sigma^2^{new} = trace[XX^T - W<x_n|y_n>(Y-\mu)^T]/n^2\)
    - Differentiate LL in terms of \(W\) and \(\sigma^2\) 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 variables)

---

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% 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

- $\text{LL} = \sum_{n=1}^{N} \ln \{p(y_n)\} = \sum_{n=1}^{N} \ln \{\sum_{i=1}^{M} \pi_i p(y_n|i)\}$

  • $p(y|i)$ is a single PPCA model and $\pi_i$ is the corresponding mixing proportion
  • Different mean vectors $\mu_i$, weighting matrices $W_i$, and noise error parameters $\sigma_i^2$ 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 $x$.

- 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:
  \[ P(x \mid \Theta) = P_0(x)e^{\Theta G(\Theta)} \]
  - \( x \) is your data in the high-dimensional space
  - \( \Theta \) is the natural (or canonical) parameterization of the distribution
  - \( P_0(x) \) is a constant (not dependent on \( \Theta \))
  - \( G(\Theta) \) is the partition function (assures a valid distribution)
Exponential Family Distributions

• Gaussian (unit variance)

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

• General form:

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

\[ \Theta = \mu \]

\[ G(\Theta) = \mu^2/2 \]

Exponential Family Distributions

• Bernoulli

\[ P(x \mid \Theta) = \pi^x (1 - \pi)^{(1-x)} = 1e \]

• General Form:

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

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

\[ G(\Theta) = \log \left( \frac{1+e^{\log \left( \frac{\pi}{1-\pi} \right)}}{1-e^{\log \left( \frac{\pi}{1-\pi} \right)}} \right) \]
Exponential Family Distributions

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

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

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

• $G(\Theta)$ is the cummulant function of $P(x \mid \Theta)$
• This means that $VG(\Theta)$ is the expected value of $x$.

---

So what?

• For any model $\Theta$, we can find the expectation of the data $x$ given $\Theta$.

• 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 $\log(p(x|\Theta))$, 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:

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

- 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 $G(\Theta)$ 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

\[
D^F_F(p, q) = F(p) - F(q) - \{\nabla F(q), p - q\}.
\]

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

\[ D^F_{\theta}(p, q) = F(p) - F(q) - \langle \nabla F(q), p - q \rangle. \]

\( F(\Theta) \)

\( p \) is our current set of natural parameters.

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

\[ D^F_{\theta}(p, q) = F(p) - F(q) - \langle \nabla F(q), p - q \rangle. \]
Bregman Divergence

\[ D^q_F(p, q) = F(p) - F(q) - \langle \nabla F(q), p - q \rangle. \]

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

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 $G(\Theta)$ 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 $G(\Theta)$
How can it be?!

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

\[
- \log P(x | \Theta) = - \log(P_0(x)) - x\Theta + G(\Theta)
\]

\[
= - \log(P_0(x)) - x\Theta + [g(\Theta)\Theta - F(g(\Theta))]
\]

\[
= - \log(P_0(x)) - F(g(\Theta)) - x\Theta + g(\Theta)\Theta
\]

\[
= - \log(P_0(x)) - F(g(\Theta)) - \Theta \cdot (x - g(\Theta))
\]

\[
= - \log(P_0(x)) - F(g(\Theta)) - [g^{-1}(g(\Theta))] \cdot (x - g(\Theta))
\]

\[
= - \log(P_0(x)) + [F(x) - F(x)] - F(g(\Theta)) - [g^{-1}(g(\Theta))] \cdot (x - g(\Theta))
\]

\[
= - \log(P_0(x)) - F(x) + F(x) - F(g(\Theta)) - f(g(\Theta)) \cdot (x - g(\Theta))
\]

\[
= - \log(P_0(x)) - F(x) + B_F(x \| g(\Theta))
\]

Optimization

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

\[
- \log( P(x | \Theta) ) = - \log(P_0(x)) - F(x) + B_F(x \| g(\Theta))
\]

• Optimizing negative loglikelihood is commonly done in EM

• Only the Bregman divergence term depends on \( \Theta \), the rest can be ignored.
Exponential PCA

• **Problem:** Find $\Theta'$s which come close to the observed data points $x$. (Minimize loss)

• Express the $\Theta'$s in a lower dimensionality

• **Solution:** Find a basis with $L$ principal axes, represent the $\Theta'$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 $n \times L$
    • (rows of $A$ represent the lower dimensionality representation of a data point)
  – $V$ is $L \times d$
    • (rows of $V$ represent the principal axes of the model’s projection basis)
  – Your data $X$ is $n \times d$
Generalized Exponential PCA

- Optimize the negative loglikelihood of a model given the data
  \[ \log P(x | \Theta) = - \log (P_0(x)) - F(x) + B_F(x \| g(\Theta)) \]
  \[ \Theta = AV \]
  - 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 \(G(\Theta)\), which
    - Changes the expectation parameters of the model, which
    - Changes the Bregman divergence which was derived from \(G(\Theta)\), which means
    - The loss function for the data is different (the Bregman distance between \(x\) and the expectation parameters \(g(\Theta)\))

Example

- Lets choose the Normal distribution
  - For a normal distribution, \(G(\Theta) = \Theta^2/2\)
  - Therefore,
    - \(g(\Theta) = G'(\Theta) = \Theta\); \(g^{-1}(x) = f(x) = x\); \(F(x) = x^2/2\)
  - Compute the Bregman divergence between \(x\) and \(g(\Theta)\):
    \[ B_F(p \| q) = F(p) - F(q) - f'(q) \cdot (p - q) \]
    \[ = F(x) - F(g(\Theta)) - f'(g(\Theta)) \cdot (x - g(\Theta)) \]
    \[ = \frac{x^2}{2} - \frac{\Theta^2}{2} - \Theta \cdot (x - \Theta) \]
    \[ = \frac{1}{2} x^2 - \frac{2}{2} \Theta x + \frac{1}{2} \Theta^2 \]
    \[ = \frac{1}{2} (x - \Theta)^2 \quad B_F(x \| g(\Theta)) \text{ ends up being Euclidean distance!} \]
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 $i = 1:n$,
          - $A_{ic} = \arg \min_{a \in \mathbb{R}} \sum_j B_F(x_{ij} \| g(A_{ic}v))$
        - For $j = 1:d$,
          - $v_{cj} = \arg \min_{v \in \mathbb{R}} \sum_i B_F(x_{ij} \| g(A_{ic}v))$

Summary:

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