Nonlinear Component Analysis Based on Correntropy

Jian-Wu Xu, Puskal P. Pokharel, António R. C. Paiva, and José C. Príncipe

Computational NeuroEngineering Laboratory,
University of Florida

July 18, 2006

Supported in part by NSF grant ECS-0300340.
A.R.C. Paiva supported by FCT grant SFRH/BD/18217/2004.

Outline

Motivation

- Why do we need nonlinear component analysis?

Motivation

- Why do we need nonlinear component analysis?
  - Linear PCA only fully describes Gaussian distributed data!
  - In all other cases the principal components are, in general, nonlinear and depend on higher order moments.

Motivation

- Why do we need nonlinear component analysis?
  - Linear PCA only fully describes Gaussian distributed data!
  - In all other cases the principal components are, in general, nonlinear and depend on higher order moments.
- Are there methods for nonlinear component analysis currently available?

Motivation

- Why do we need nonlinear component analysis?
  - Linear PCA only fully describes Gaussian distributed data!
  - In all other cases the principal components are, in general, nonlinear and depend on higher order moments.
- Are there methods for nonlinear component analysis currently available?
  - Iterative methods (Hastie and Stuetzle, 1989; De'ath, 1999)
  - Kernel principal component analysis – Kernel PCA (Schölkopf et al., 1998)
  - ...

Motivation

- Why do we need another nonlinear component analysis method?

Motivation

- Why do we need another nonlinear component analysis method?
    - Iteratives methods are time consuming and prone to search problems (local minima, etc.)
    - Kernel PCA needs to solve the eigendecomposition of the Gram matrix, which has the dimensionally of the data (1000 data points $\Longrightarrow$ 1000×1000 Gram matrix.)
    - Difficult interpretation

Motivation

- Why do we need another nonlinear component analysis method?
  - Iteratives methods are time consuming and prone to search problems (local minima, etc.)
  - Kernel PCA needs to solve the eigendecomposition of the Gram matrix, which has the dimensionally of the data (1000 data points $\Longrightarrow$ 1000 $\times$ 1000 Gram matrix.)
  - Difficult interpretation
- CORRENTROPY PCA:
  - Solves nonlinear component analysis
  - Incorporates higher order statistics
  - Constrained to input dimensionality

Definition of correntropy

- The correntropy of two random variables X and Y is defined as

$$V_{XY} \triangleq E\left[\kappa(x, y)\right].$$

where

  - $E\left[\cdot\right]$ denotes mathematical expectation over X and Y
  - κ is a symmetric positive definite kernel that obeys the Mercer's conditions.

Properties of correntropy

- Correntropy depends on higher order monents. For example, using a Gaussian kernel the series expansion is

$$V_{XY} = \frac{1}{\sqrt{2\pi}\sigma} \sum_{n=0}^{\infty} \frac{(-1)^n}{2^n \sigma^{2n} n!} E\left[\|x - y\|^{2n} \right]$$

- Given any symmetric and positive definite kernel $\kappa(x, y)$, the correntropy kernel is symmetric and positive definite.

Kernel mapping

- Since the correntropy kernel is symmetric and positive definite, the Moore-Aronszajn theorem states that a unique *reproducing kernel Hilbert space (RKHS)* – $\mathcal{H}$ – exists.
- From Mercer's theorem, the correntropy kernel can be decomposed in a sequence of non-negative eigenvalues, $\{\lambda_k : k = 1, 2, \ldots\}$, and corresponding (normalized) eigenfunctions, $\{\varphi_k(x) : k = 1, 2, \ldots\}$.
- This is,

$$V_{XY} = \sum_{k=0}^{\infty} \lambda_k \varphi_k(x)\varphi_k(y) = \sum_{k=0}^{\infty} (\sqrt{\lambda_k}\varphi_k(x))(\sqrt{\lambda_k}\varphi_k(y))$$
$$= \langle \Pi(x), \Pi(y) \rangle$$

Outline

Mapping input data to feature space

- Given a set of zero mean vectors $\mathbf{x}_i \in \mathbb{R}^L, i = 1, \ldots, N$, CORRENTROPY PCA maps the data component-wise in feature space, i.e.:

$$\begin{aligned} \Pi(\mathbf{x}) : \mathbb{R}^L &\longmapsto \mathcal{F} \\ \mathbf{x} &\longmapsto [\Pi(x_1), \Pi(x_2), \ldots, \Pi(x_L)] \end{aligned}$$

  where x_i denotes the ith component of the input sample $\mathbf{x}$.
- This leads to the following definition:

$$\begin{aligned} V_{ij} &\triangleq E\left[\kappa(x_i, x_j)\right] = \langle \Pi(x_i), \Pi(x_j) \rangle \\ &\approx \frac{1}{N} \sum_{k=1}^{N} \kappa(x_{ik}, x_{jk}), \quad \forall i, j = 1, \ldots, L \end{aligned}$$

Feature space component analysis (1)

- The covariance matrix of the transformed data is given by

$$\mathbf{C} = \frac{1}{L}\sum_{i=1}^{L} \Pi(x_i)\Pi(x_i)^T$$

- Then, we can compute the eigendecomposition of $\mathbf{C}$,

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

- Since all solutions lie in the span of $\Pi(x_1), \ldots, \Pi(x_L)$, we have that

$$\mathbf{q} = \sum_{j=1}^{L} \beta_j \Pi(x_j)$$

Feature space component analysis (2)

- Instead of solving the eigendecomposition we can solve

$$\langle \Pi(x_k), \mathbf{C}\mathbf{q} \rangle = \langle \Pi(x_k), \lambda \mathbf{q} \rangle \,, \quad \forall k = 1, \ldots, L$$

- Substituting the expressions for $\mathbf{C}$ and $\mathbf{q}$, yields

$$\frac{1}{L} \sum_{i=1}^{L} \sum_{j=1}^{L} \beta_j \langle \Pi(x_k), \Pi(x_i) \rangle \langle \Pi(x_i), \Pi(x_j) \rangle = \lambda \sum_{j=1}^{L} \beta_j \langle \Pi(x_k), \Pi(x_j) \rangle \,, \quad \forall k = 1, \ldots, L$$

Feature space component analysis (3)

- Define the *correntropy matrix* with the ijth entry,

$$V_{ij} = \langle \Pi(x_i), \Pi(x_j) \rangle \approx \frac{1}{N} \sum_{k=1}^{N} \kappa(x_{ik}, x_{jk}), \quad \forall i, j = 1, \ldots, L$$

- Then, the solutions of the previous set of equations can be found through the eigendecomposition of

$$V^2 \bar{\beta} = L \lambda V \bar{\beta}$$

which has the same solutions as, $V\bar{\beta} = L\lambda\bar{\beta}$, where $\bar{\beta} = [\beta_1, \ldots, \beta_L]^T$.

Computing the data projections

- The data projections are given by the inner product of the transformed vector with the eigenvectors:

$$P(\mathbf{a}) = \sum_{i=1}^{L} \beta_i \frac{1}{N} \sum_{j=1}^{N} \kappa(x_{ij}, a_i)$$

Correntropy PCA: Summary

1. Compute the correntropy matrix $\mathbf{V}$
2. Compute the eigendecomposition of the correntropy matrix
3. Project the data points onto the eigenvectors

Data centering in feature space

- So far, the transformed vectors were assumed to be zero mean, which is not true in general.
- The centered data in feature space is given by:

$$\begin{aligned}\overline{\Pi(x_i)} &= \Pi(x_i) - E\left[\Pi(x_i)\right] \\ &= \Pi(x_i) - \frac{1}{N}\sum_{k=1}^{N}\Pi(x_{ik})\end{aligned}$$

where x_{ik} is the ith component of the kth sample vector.

Data centering in feature space: inner product bias adjustment

- In terms of input samples, the inner product between two centered vectors in feature space is given by:

$$\begin{aligned}\left\langle \overline{\Pi(x_i)}, \overline{\Pi(x_j)} \right\rangle &= \langle \Pi(x_i), \Pi(x_j) \rangle - 2 \left\langle \Pi(x_i), \frac{1}{N}\sum_{m=1}^{N} \Pi(x_{jm}) \right\rangle \\ &\quad + \left\langle \frac{1}{N}\sum_{k=1}^{N} \Pi(x_{ik}), \frac{1}{N}\sum_{m=1}^{N} \Pi(x_{jm}) \right\rangle \\ &= E\left[\kappa(x_i - x_j)\right] - \frac{1}{N}\sum_{k=1}^{N}\sum_{m=1}^{N} \kappa(x_{ik} - x_{jm})\end{aligned}$$

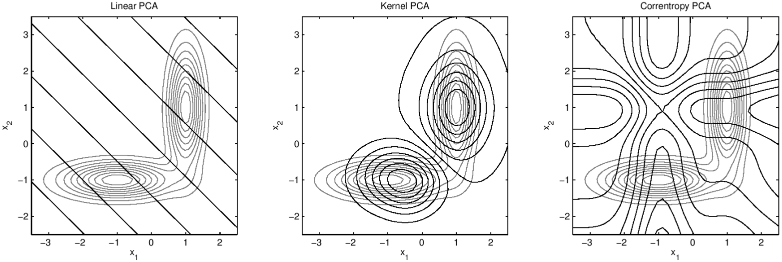
Outline

Example 1: Mixture of two Gaussians

- Generated 200 samples from mixture of two Gaussians:

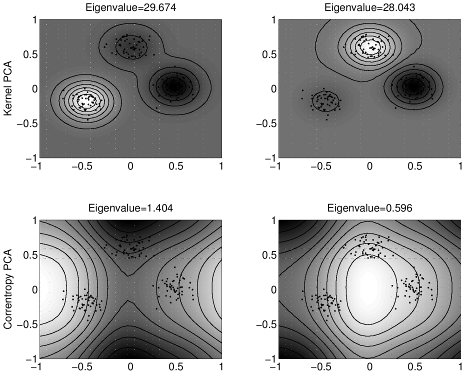
$$f(\mathbf{x}) = (\mathcal{N}(\mathbf{m}_1, \Sigma_1) + \mathcal{N}(\mathbf{m}_2, \Sigma_2))/2$$

- Kernel size: 0.5



Example 2: Mixture of three Gaussians clusters

- Generated 150 samples (50 per cluster) from mixture of three Gaussians clusters with standard deviation 0.1.
- Kernel size: 0.2



Conclusions

- Proposed novel approach for principal component analysis, based on the correntropy cost function.
- Incorporates higher order statistics.
- Problem constrained to the dimensionality of the input.
- Much smaller computational complexity.