Non-linear feature extraction by the coordination of mixture models

Jakob Verbeek, Nikos Vlassis, Ben Krose

To cite this version:

Jakob Verbeek, Nikos Vlassis, Ben Krose. Non-linear feature extraction by the coordination of mixture models. 9th Annual Conference of the Advanced School for Computing and Imaging (ASCI '03), Jun 2003, Heijen, Netherlands. pp.287–293. inria-00321490v2

HAL Id: inria-00321490
https://inria.hal.science/inria-00321490v2

Submitted on 8 Mar 2011

Non-linear Feature Extraction by the Coordination of Mixture Models

J.J. Verbeek N. Vlassis B.J.A. Kröse

Intelligent Autonomous Systems Group, Informatics Institute,
Faculty of Science, University of Amsterdam,
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
jverbeek@science.uva.nl vlassis@science.uva.nl krose@science.uva.nl

Keywords: Canonical Correlation Analysis, Principal Component Analysis, Self-organizing Maps.

Abstract

We present a method for non-linear data projection that offers non-linear versions of Principal Component Analysis and Canonical Correlation Analysis. The data is accessed through a probabilistic mixture model only, therefore any mixture model for any type of data can be plugged in. Gaussian mixtures are one example, but mixtures of Bernoulli's to model discrete data might be used as well. The algorithm minimizes an objective function that exhibits one global optimum that can be found by finding the eigenvectors of some matrix. Experimental results on toy data and real data are provided.

1 Introduction

Much research has been done in the field of unsupervised non-linear feature extraction to obtain a low dimensional description of high dimensional data that preserves important properties of the data. In this paper we are interested in data that lies on or close to a low dimensional manifold embedded (possibly non-linearly) in a Euclidean space of much higher dimension. A typical example of such data are images of an object under different conditions (e.g. pose and lighting) obtained with a high resolution camera. A low dimensional representation may be desirable for different reasons like compression for storage and communication, for visualization of high dimensional data, and as a preprocessing step for further data analysis or prediction tasks.

An example is given in Fig. 1, we have data in $\mathbb{R}^3$ which lies on a two dimensional manifold (top plot). We want to recover the structure of the data manifold, so that we can 'unroll' the data manifold and work with the data expressed in the 'latent coordinates', i.e. coordinates on the manifold (second plot).

Here, we consider a method to integrate several local feature extractors into a single global representation. The idea is to learn a mixture model on the data where each mixture component acts as a local feature extractor. A mixture of Probabilistic Principal Component Analyzers [1] is a typical example. After this mixture has been learned, the local feature extractors are 'coordinated' by finding for each local feature extractor a suitable linear map into a single 'global' low-dimensional coordinate system. The collective of the local feature extractors together with the linear maps into the global latent space provides a global non-linear projection from the data space into the latent space.

This differs from other non-linear feature extraction methods like: Generative Topographic Map (GTM) [2], Locally Linear GTM [3], Kohonen's Self-Organizing Map [4] and a Variational Expectation Maximization algorithm similar to SOM [5]. These algorithms are actually quite opposite in nature to the method presented here: they start with a given configuration of mixture components in the latent space and adapt the mixture configuration in the data space to optimize some objective function. Here we start with a given and fixed mixture in the data space and find an optimal configuration in the latent space. The advantage of the latter method is that it is not prone to local optima in the objective function.[1] Moreover, the solution for a d-dimensional latent space can be found by finding the eigenvec-

[1] Of course, fitting the mixture model on the data might be susceptible to local optima.

tors of a matrix corresponding to the 2^{nd} smallest up to the $(d+1)^{st}$ smallest eigenvalues. The size of this matrix is given by the total number of locally extracted features plus the number of mixture components, i.e. if all k components extract d features then the size is $k(d+1)$.

The work presented in this paper can be categorized together with other recent algorithms for unsupervised non-linear feature extraction like IsoMap [6], Local Linear Embedding [7], Kernel PCA [8], Kernel CCA [9] and Laplacian Eigenmaps [10]. All these algorithms perform non-linear feature extraction by minimizing an objective function that exhibits a limited amount of stable points that can be characterized as eigenvectors of some matrix. These algorithms are generally simple to implement, since it is only needed to construct some matrix, then the eigenvectors are found by standard methods.

In this paper we build upon recent work of Brand [11]. In the next section we review the work of Brand, we derive the objective function by a slight modification of the free-energy used in [12] and we discuss how we can find its stable points. In Section 3 we present a method to obtain better results by using several mixtures for the data in parallel. Secondly, we show in Section 4 that finding the locally valid linear maps between the data space and the latent space can be regarded as a weighted form of Canonical Correlation Analysis (CCA), and we show how we can generalize the work of Brand to perform non-linear CCA. A general discussion and conclusions can be found in Section 5.

2 Linear Embedding of Local Feature Extractors

In this section we describe the work of Brand [11], in a notation that allows us to draw links with CCA in section 4.

Consider a given data set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and a collection of k local feature extractors of the data $\mathbf{f}_s(\mathbf{x})$ for $s = 1, \ldots k$. Thus, $\mathbf{f}_s(\mathbf{x})$ is a vector containing the features extracted from $\mathbf{x}$ by feature extractor s. Each feature extractor gives zero or more (linear) features of the data. With each feature extractor, we associate a mixture component in a probabilistic k-component mixture model. We write the density on high dimensional data items $\mathbf{x}$ as:

$$p(\mathbf{x}) = \sum_{s=1}^{k} p(s)p(\mathbf{x}|s), \qquad (1)$$

where s is a variable ranging over the mixture components and $p(s)$ is the a-priori probability on component s (sometimes also termed 'mixing weight' of component s).

Next, we consider the relationship between the given representation of the data (denoted with $\mathbf{x}$) and the representation of the data in the latent space, which we have to find. Throughout, we will use $\mathbf{g}$ to denote latent coordinates for data items. The $\mathbf{g}$ is for 'Global' latent coordinate as opposed to locally extracted features, which we will denote by $\mathbf{z}$. For the unobserved latent coordinate $\mathbf{g}$ corresponding to a data point $\mathbf{x}$ and conditioned on s, we assume the density:

$$p(\mathbf{g}|\mathbf{x}, s) = \mathcal{N}(\mathbf{g}; \boldsymbol{\kappa}_s + \mathbf{A}_s\mathbf{f}_s(\mathbf{x}), \sigma^2\mathbf{I}) \qquad (2)$$

where $\mathcal{N}(\mathbf{g}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a Gaussian distribution on $\mathbf{g}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. The mean, $\mathbf{g}_{ns}$, of $p(\mathbf{g}|\mathbf{x}_n, s)$ is the sum of the component mean $\boldsymbol{\kappa}_s$ in the latent space and a linear transformation, implemented by $\mathbf{A}_s$, of $\mathbf{f}_s(\mathbf{x}_n)$. From now on we will use homogeneous coordinates and write: $\mathbf{L}_s = [\mathbf{A}_s\boldsymbol{\kappa}_s]$ and $\mathbf{z}_{ns} = [\mathbf{f}_s(\mathbf{x}_n)^\top 1]^\top$, and thus $\mathbf{g}_{ns} = \mathbf{L}_s\mathbf{z}_{ns}$.

Consider the marginal (over s) distribution on latent coordinates given data:

$$p(\mathbf{g}|\mathbf{x}) = \sum_s p(s, \mathbf{g}|\mathbf{x}) = \sum_s p(s|\mathbf{x})p(\mathbf{g}|\mathbf{x}, s). \qquad (3)$$

Given a fixed set of local feature extractors and a corresponding mixture model, we are interested in finding linear maps $\mathbf{L}_s$ that give rise to 'nice' projections of the data in the latent space. By 'nice', we mean that components that have a high posterior probability to have generated $\mathbf{x}$ (i.e. $p(s|\mathbf{x})$ is large) should give similar projections of $\mathbf{x}$ in the latent space. So we would like the $p(\mathbf{g}|\mathbf{x}, s)$ to look similar for the components with large posterior. Since the marginal $p(\mathbf{g}|\mathbf{x})$ is a Mixture of Gaussians, we can measure the similarity between the predictions by looking how much the mixture resembles a single Gaussian. If the predictions are all the same, the mixture is a single Gaussian.

We are now ready to formally define the objective function $\Phi(\{\mathbf{L}_1, \ldots, \mathbf{L}_k\})$, which sums the data log-likelihood and a Kullback-Leibler divergence $\mathcal{D}$ which measures how uni modal the distribution $p(\mathbf{g}|\mathbf{x}) = \sum_s p(s|\mathbf{x})p(\mathbf{g}|\mathbf{x}, s)$ is.

$$\begin{aligned}\Phi &= \sum_{n=1}^{N} \log p(\mathbf{x}_n) \\ &\quad - \sum_n \mathcal{D}(Q_n(\mathbf{g}, s) \parallel p(\mathbf{g}, s|\mathbf{x}_n)), \qquad (4)\end{aligned}$$

where Q_n distributes independently over $\mathbf{g}$ and s:

$$Q_n(\mathbf{g}, s) = q_{ns}Q_n(\mathbf{g}) = q_{ns}\mathcal{N}(\mathbf{g}; \mathbf{g}_n, \boldsymbol{\Sigma}_n). \qquad (5)$$

This objective function is similar to the one used in [12, 13]. The difference here is that the covariance matrix of $p(\mathbf{g}|\mathbf{x}, s)$ does not depend on the linear maps $\mathbf{L}_s$. If we fix the data space model, turning the data log-likelihood into a constant, and set $q_{ns} = p(s|\mathbf{x}_n)$, then the objective sums for each data point $\mathbf{x}_n$ the Kullback-Leibler divergence between $Q_n(\mathbf{g})$ and $p(\mathbf{g}|\mathbf{x}, s)$, weighted by the posterior $p(s|\mathbf{x}_n)$:

$$\Phi = \sum_{n,s} q_{ns}\mathcal{D}(Q_n(\mathbf{g}) \parallel p(\mathbf{g}|\mathbf{x}_n, s)). \tag{6}$$

In order to minimize the objective Φ with respect to $\mathbf{g}_n$ and $\mathbf{\Sigma}_n$, it is easy to derive that we get:

$$\mathbf{g}_n = \sum_s q_{ns}\mathbf{g}_{ns} \text{ and } \mathbf{\Sigma}_n = \sigma^2\mathbf{I}, \tag{7}$$

where $\mathbf{I}$ denotes the indentity matrix. Skipping some additive and multiplicative constants with respect to the linear maps $\mathbf{L}_s$, the objective Φ then simplifies to:

$$\Phi = \sum_{n,s} q_{ns} \parallel \mathbf{g}_n - \mathbf{g}_{ns} \parallel^2 \geq 0, \tag{8}$$

with $\mathbf{g}_{ns} = \mathbf{L}_s\mathbf{z}_{ns}$ as before. Note that we can rewrite the objective as:

$$\Phi = \frac{1}{2}\sum_{n,s,t} q_{ns}q_{nt} \parallel \mathbf{g}_{nt} - \mathbf{g}_{ns} \parallel^2 \geq 0. \tag{9}$$

The objective is a quadratic function of the linear maps $\mathbf{L}_s$ [11]. At the expense of some extra notation, we obtain a clearer form of the objective as a function of the linear maps. Let:

$$\begin{aligned} \mathbf{u}_n &= [q_{n1}\mathbf{z}_{n1}^\top \dots q_{nk}\mathbf{z}_{nk}^\top], & (10)\\ \mathbf{U} &= [\mathbf{u}_1^\top \dots \mathbf{u}_N^\top]^\top, & (11)\\ \mathbf{L} &= [\mathbf{L}_1 \dots \mathbf{L}_k]^\top. & (12) \end{aligned}$$

Note that from (7), (10) and (12) we have: $\mathbf{g}_n = (\mathbf{u}_n\mathbf{L})^\top$. The expected projection coordinates can thus be computed as:

$$\mathbf{G} = [\mathbf{g}_1 \dots \mathbf{g}_N]^\top = \mathbf{UL}. \tag{13}$$

We define the block-diagonal matrix $\mathbf{D}$ as:

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{D}_k \end{pmatrix} \tag{14}$$

where $\mathbf{D}_s = \sum_n q_{ns}\mathbf{z}_{ns}\mathbf{z}_{ns}^\top$. The objective then can be written as:

$$\Phi = \mathrm{Tr}\{\mathbf{L}^\top(\mathbf{D} - \mathbf{U}^\top\mathbf{U})\mathbf{L}\}. \tag{15}$$

The objective function is invariant for translation and rotation of the global latent space, as can be easily seen from (8). Furthermore, re-scaling the latent space changes the objective monotonically. To make solutions unique with respect to translation, rotation and scaling, we put two constraints:

$$(\mathit{trans.}): \ \bar{\mathbf{g}} = \frac{1}{N}\sum_n \mathbf{g}_n = 0, \tag{16}$$

$$\begin{aligned} (\mathit{rot.} + \mathit{sc.}): \ \mathbf{\Sigma}_\mathbf{g} &= \sum_n (\mathbf{g}_n - \bar{\mathbf{g}})(\mathbf{g}_n - \bar{\mathbf{g}})^\top & (17)\\ &= \mathbf{L}^\top\mathbf{U}^\top(\mathbf{I} - \mathbf{1}/N)\mathbf{UL} = \mathbf{I}, \end{aligned}$$

where we used $\mathbf{1}$ to denote the matrix with ones everywhere. Solutions are found by differentiation using Lagrange multipliers, and the columns $\mathbf{v}$ of $\mathbf{L}$ are found to be generalized eigenvectors:

$$(\mathbf{D} - \mathbf{U}^\top\mathbf{U})\mathbf{v} = \lambda\mathbf{U}^\top(\mathbf{I} - \mathbf{1}/n)\mathbf{Uv} \tag{18}$$

Since the solutions are zero mean, it follows that $\mathbf{1Uv} = \mathbf{0}$, and hence that:

$$\begin{aligned} (\mathbf{D} - \mathbf{U}^\top\mathbf{U})\mathbf{v} &= \lambda\mathbf{U}^\top\mathbf{Uv} & (19)\\ &\Leftrightarrow \\ \mathbf{Dv} &= (\lambda + 1)\mathbf{U}^\top\mathbf{Uv}. & (20) \end{aligned}$$

The value of the objective function is given by:

$$\begin{aligned} \Phi &= \sum_i \mathbf{v}_i^\top(\mathbf{D} - \mathbf{U}^\top\mathbf{U})\mathbf{v}_i & (21)\\ &= \sum_i \lambda_i\mathbf{v}_i^\top\mathbf{U}^\top\mathbf{Uv}_i & (22)\\ &= (N-1)\sum_i \lambda_i \mathrm{var}(\mathbf{g}^i), & (23) \end{aligned}$$

where $\mathrm{var}(\mathbf{g}^i)$ denotes the variance in the i^{th} coordinate of the projected points. So rescaling to get unit variance makes the objective equal to $(N-1)\sum_i \lambda_i$. The smallest eigenvalue is always zero, corresponding to projections that collapses all projections in the point $\boldsymbol{\kappa}$. This projection has for $s = 1, \dots, k$: $\mathbf{A}_s = 0$ and $\boldsymbol{\kappa}_s = \boldsymbol{\kappa}$. This is embedding tells us nothing about the data, therefore we need the eigenvectors corresponding to the second up to the $(d+1)^{st}$ smallest eigenvalues to obtain the best embedding in d dimensions. Note that there is no problem if different feature extractors extract a different number of features.

In Fig. 1 we give an illustration of the above. The top two images show the original data presented to the algorithm and the 2-dimensional representation found by the algorithm. The blue and yellow sticks indicate the local feature extractors which are centered on the red dots. The two lower scatter plots compare the coordinates on the data generating manifold with the discovered latent coordinates. For both coordinates there is high correlation with the true latent coordinates.

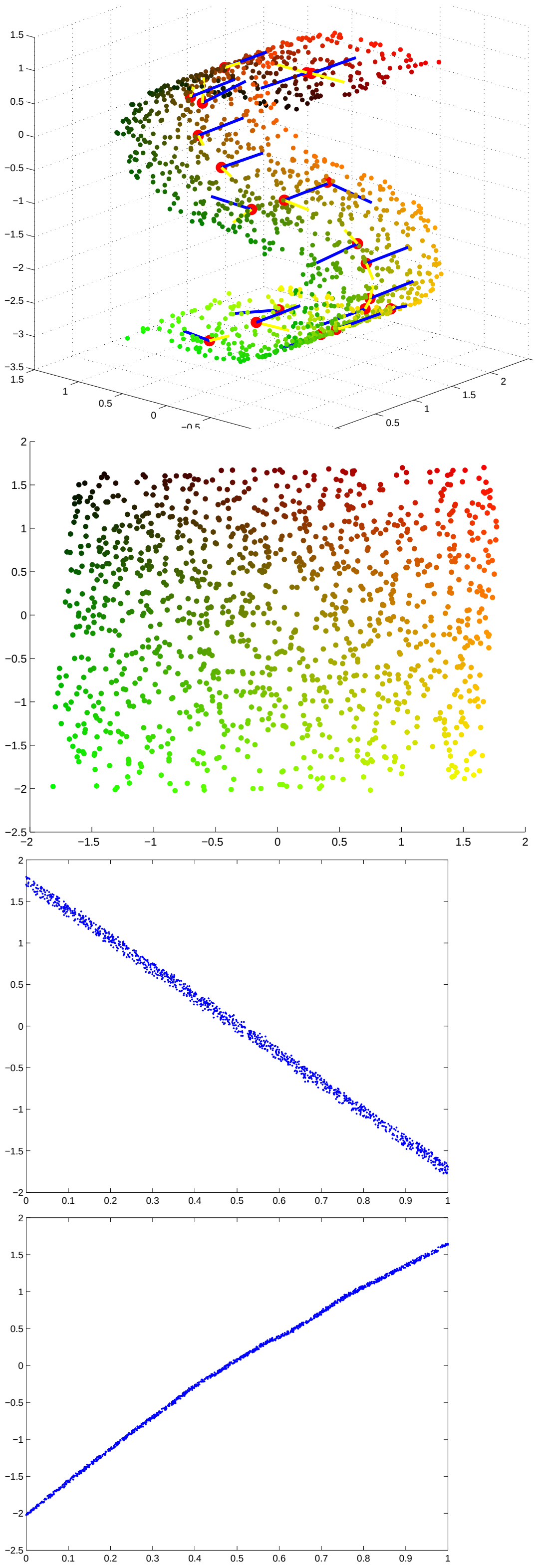

Figure 1: From top to bottom. (1) Data in $\mathbb{R}^3$ with charts indicated by the blue and yellow sticks. (2) Data representation in $\mathbb{R}^2$. (3) First original coordinate on manifold against first found latent coordinate. (4) Second original coordinate on manifold against second found latent coordinate.

3 Incorporating more points in the objective function

In this section we discuss how using multiple mixtures for the data can help obtain more stable results. The next section then discusses how this method can be exploited to define a non-linear form of Canonical Correlation Analysis.

Sometimes the method of the previous section collapses several linear feature extractors in some part of the underlying data manifold onto a point or a line. Whereas the Automatic Alignment (AA) method [14] was less sensitive to such phenomena.

AA is based on the Local Linear Embedding algorithm [7] and finds linear maps to integrate a collection of locally valid feature extractors in a manner similar to what we described in the previous section. In AA the objective function is:

$$\Phi_{AA} = \sum_{m,n} \mathbf{W}_{nm} \parallel \mathbf{g}_n - \mathbf{g}_m \parallel^2, \qquad (24)$$

again we have $\mathbf{g}_n = \sum_s q_{ns} \mathbf{g}_n^s$. The weights, collected in the weight matrix $\mathbf{W}$, are found by first determining for every data point $\mathbf{x}_n$ its nearest neighbors $\mathrm{nn}(n)$ in the data space and use the $\mathbf{W}$ that minimizes:

$$\sum_n \parallel \mathbf{x}_n - \sum_{m \in \mathrm{nn}(n)} \mathbf{W}_{nm} \mathbf{x}_m \parallel^2, \qquad (25)$$

under the constraint that $\sum_m \mathbf{W}_{nm} = 1$ and only nearest neighbors can have non-zero weights, thus: $\forall m \notin \mathrm{nn}(n) : \mathbf{W}_{nm} = 0$.

The objective function (8) of the previous section is in fact only based on points for which the posteriors q_{ns} assigns non-negligible mass on at least two mixture components. This is seen directly from (9), since if one of the q_{ns} takes all mass, all products of the posteriors are zero except for one term where the squared distance is zero. In other words, the objective only focuses on data that is on the overlap of at least two mixture components. So, only points for which the entropy of the posterior is reasonably high contribute to the objective function. The other data, for which the entropy in the posterior is low, is thus completely neglected in the objective function and thus in the coordination of the local feature extractors.

In practice, often the bulk of the data has a low entropy posterior and hardly plays a role in the objective function. To make more use of the available data we should have more entropy in the posteriors. One way to do that is to smoothen the posteriors by taking the closest posterior in KL-divergence sense to the old posterior that has a certain entropy, yielding: $q'_{ns} \propto q^{\alpha}_{ns}$. Another possibility is to smoothen such that the *average* entropy has a certain value. However, to determine the 'right' amount of entropy in the posteriors seems a rather problematic question.

The problem can also be circumvented by using several mixtures, say two, in parallel. The mixtures are trained independently and the final posterior for a specific mixture component from one of the two mixtures is defined as half of the posterior within the mixture from which this component originates. So, the sum of the posteriors on the components originating from the first mixture equals 1/2 and similarly for the second mixture. In this manner we always have an entropy of at least one bit in the posteriors, and hence all points make a significant contribution to the objective function.

Note that we are using just a heuristic here, more principled ways to learn a mixture with high entropy in the posteriors could be divised but do not seem essential. Only in the case that the two independently learned mixtures are the same we have not gained anything. This, however, is not likely to be the case for mixture models fitted by EM algorithms that find local optima of the log-likelihood function. We initialize the EM mixture learning algorithm at different values and in this manner we (probably) find mixtures that are different local optima of the data log-likelihood function. We can think of this method as covering the data manifold not with one set of tiles, but using two coverings of the manifold. In this way every tile is overlapping with several other tiles on its complete extent.

We found in experiments that using two mixtures instead of one gives great improvement in the obtained results. Moreover, we prevent the computation of the weight matrix $\mathbf{W}$ needed by AA. The time needed to compute the weight matrix is in principle quadratic in the number of data points and can be problematic in non-Euclidean spaces. By using several mixtures in parallel, we avoid the quadratic cost and have a method that is based purely on mixture models and that can be readily applied to any type of mixture models.

In Fig. 2 we give some examples of data embeddings obtained on a data set of 1965 pictures of a face.[2] First the images were projected from the original 28×20 pixel coordinates to 10 dimensions with PCA. The projection from 10 to 2 dimensions was done by fitting a 10 component mixture of 2-dimensional probabilistic PCA [1] and aligning them with the method described in the previous section. We show the latent coordinates assigned to the pictures as dots, some pictures are shown next to their latent coordinate.

The two bottom figures, where we used two parallel mixtures, show a nice separation of the 'smiling' faces and the 'angry' faces. Furthermore, the variations in gaze direction are nicely identified. The two top images, where we used single mixtures with respectively ten and twenty mixture components do not show the separation between the two 'moods'.

4 Non-linear Canonical Correlation Analysis

In Canonical Correlation Analysis (CCA) [15, 16] two zero mean sets of points are given: $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subset \mathrm{IR}^{\mathrm{p}}$ and $\mathbf{Y} = \{\mathbf{y_1}, \ldots, \mathbf{y_N}\} \subset \mathrm{IR}^{\mathrm{q}}$. The aim is to find linear maps $\mathbf{a}$ and $\mathbf{b}$, that map members of $\mathbf{X}$ and $\mathbf{Y}$ respectively on the real line, such that the correlation between the linearly transformed variables is maximized. This is easily shown to be equivalent with minimizing:

$$\mathcal{E} = \frac{1}{2} \sum_n \left[\mathbf{a}\mathbf{x}_n - \mathbf{b}\mathbf{y}_n\right]^2 \tag{26}$$

under the constraints that $\mathbf{a}[\sum_n \mathbf{x}_n \mathbf{x}_n^\top]\mathbf{a}^\top + \mathbf{b}[\sum_n \mathbf{y}_n \mathbf{y}_n^\top]\mathbf{b}^\top = 1$. Several generalizations of CCA are possible.

Generalizing the above such that the sets do not need to be zero mean and allowing a translation as well gives the problem of minimizing:

$$\mathcal{E} = \frac{1}{2} \sum_n \left[(\mathbf{a}\mathbf{x}_n + \mathbf{t}_a) - (\mathbf{b}\mathbf{y}_n + \mathbf{t}_b)\right]^2, \tag{27}$$

again under the constraints that the sum of variances of the projections of $\mathbf{X}$ and $\mathbf{Y}$ are equal to some fixed constant. We can also generalize by mapping to IR^{d} instead of the real line, and then requiring the covariance matrix of the projections to be identity.[3] CCA can also be readily extended to take into account more than two point sets, c.f. [9].

In the generalized CCA with multiple point-sets and allowing translations and mapping to IR^{d}, we minimize the squared distance between all pairs of projections under the constraints that each set of projected points has unit variance. We denote the projection of the n-th point in the s-th point-set as $\mathbf{g}_{ns}$. We thus minimize the error

[2] These images can be obtained from the web page of Sam Roweis at `http://www.cs.toronto.edu/~roweis`.

[3] If the linear maps are constrained to be orthonormal matrices (rotations plus reflection) then this is known as the Procrustes rotation [17].

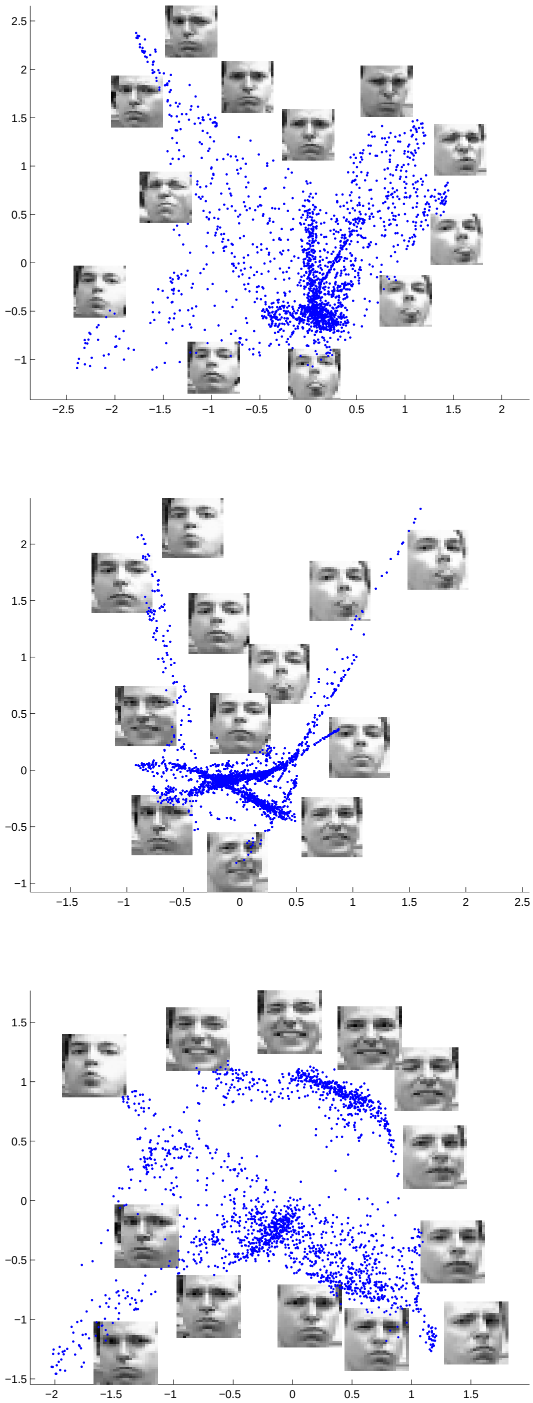

Figure 2: From top to bottom: (1) One mixture of 10 components. (2) One mixture of 20 components. (3) Two parallel mixtures of 10 components each.

function:

$$\Phi_{CCA} = \frac{1}{2k} \sum_n \sum_{s=1}^{k} \sum_{t=1}^{k} \parallel \mathbf{g}_{ns} - \mathbf{g}_{nt} \parallel^2 \quad (28)$$

$$= \sum_n \sum_s \parallel \mathbf{g}_{ns} - \mathbf{g}_n \parallel^2, \quad (29)$$

where $\mathbf{g}_n = \frac{1}{k} \sum_s \mathbf{g}_{ns}$. The constraint is that: $\sum_s \sum_n \parallel \mathbf{g}_{ns} \parallel^2 = 1$.

Note that the objective Φ in equation (8) coincides with Φ_{CCA} in (29) if we set $q_{ns} = 1/k$. The different constraints imposed upon the optimization by CCA and our objective of the previous sections turn out to be equivalent, since the stable points under both constraints coincide. We can regard the features extracted by each local feature extractor as a point set. Hence, we can interpret the method of the Section 2 as a weighted form of CCA that allows translation on multiple point sets.

Viewing the coordination of local feature extractors as a form of CCA suggests using the coordination technique for non-linear CCA. This is achieved quite easily, without modifying the objective function (8). We consider different point sets, each having a mixture of locally valid linear projections into the 'global' (in the sense of shared by all mixture components and point sets) latent space. We minimize the weighted sum of the squared distances between all pairs of projections, i.e. we have pairs of projections due to the same point set and also pairs that combine projections from different point sets. As with the use of parallel mixtures, the weight q_{ns} associated with the s^{th} projection of observation n, is given by the posterior of the mixture component in the corresponding mixture (e.g. the posterior $p(s|\mathbf{x}_n)$ if projection s is due to a component in the mixture for $\mathbf{X}$ and $p(s|\mathbf{y}_n)$ if s refers to a component in the mixture fitted to $\mathbf{Y}$) divided over the number of point sets. It is now clear that the use of parallel mixtures in the previous section was merely a form of non-linear CCA where both point sets are the same.

We again define $\mathbf{g}_n = \sum_s q_{ns}\mathbf{g}_{ns}$, where s now ranges over all mixture components in all mixtures fitted on the C different point sets. We use q^c_{nr} to denote the posterior on component r for observation n in point set c. where the q^c_{nr} are rescaled with a factor C such that all q^c_{nr} for one point set c sum to one. Furthermore, we define $\mathbf{g}^c = \sum_r q^c_{nr}\mathbf{g}^c_{nr}$ as the average projection due to point set c. The objective can be rewritten in different forms:

$$\begin{aligned}
\Phi &= \frac{1}{2}\sum_{n}\sum_{s,t} q_{ns}q_{nt} \parallel \mathbf{g}_{ns} - \mathbf{g}_{nt} \parallel^2 && (30)\\
&= \sum_{n}\sum_{s} q_{ns} \parallel \mathbf{g}_{ns} - \mathbf{g}_{n} \parallel^2 && (31)\\
&= \sum_{c} \frac{1}{C}\left[\sum_{n,s} q^c_{ns} \parallel \mathbf{g}_{n} - \mathbf{g}^c_{ns} \parallel^2\right] && (32)\\
&= \sum_{c} \frac{1}{C}\sum_{n} [\parallel \mathbf{g}_{n} - \mathbf{g}^c_{n} \parallel^2]\\
&+ \sum_{c} \frac{1}{C}\sum_{n}\left[\sum_{s} q^c_{ns} \parallel \mathbf{g}^c_{n} - \mathbf{g}^c_{ns} \parallel^2\right]. && (33)
\end{aligned}$$

Observe how in (33) the objective sums both between point set consistency of the projections (first summand) and within point set consistency of the projections (second summand).

As an illustrative toy application of the non-linear CCA we took two point-sets in $\mathbb{R}^2$ of 600 points each. The first point-set was generated on an S-shaped curve the second point set was generated along a circle segment. To both point sets we added Gaussian noise. We learned a $k = 10$ component mixture model on both point-sets, illustrated in the top figures of Fig. 3. In the bottom figure the discovered latent space representation (vertical axis) is plotted against the coordinate on the generating curve (horizontal axis). The relationship is monotonic and roughly linear.

5 Discussion and Conclusions

We have shown how we can globally maximize a free-energy objective similar to that of [12, 13] with respect to the linear maps $\mathbf{L}_s$ that map homogeneous coordinates of the local feature extractors into a global latent space. This was achieved by making the variance of $p(\mathbf{g}|s, \mathbf{x})$ independent of the linear maps. The linear maps are found by computing the $d+1$ smallest eigenvectors of a matrix. The edge size of the eigenproblem is given by k plus the total number of extracted features. Typically, if all feature extractors give the same number of features, this will be $k(d+1)$. However, it is not necessary to let every mixture component extract an equal number of features.

The way in which we find the latent representation of the data is very similar to the k-segments algorithm [18]. In the latter we first fitted a mixture of PCA's and used a (only locally optimal) combinatorial optimization algorithm to discover how the different PCA's (segments) could be aligned. Also here, first a mixture model is fitted to the data and then an optimization problem is solved to coordinate the mixture components.

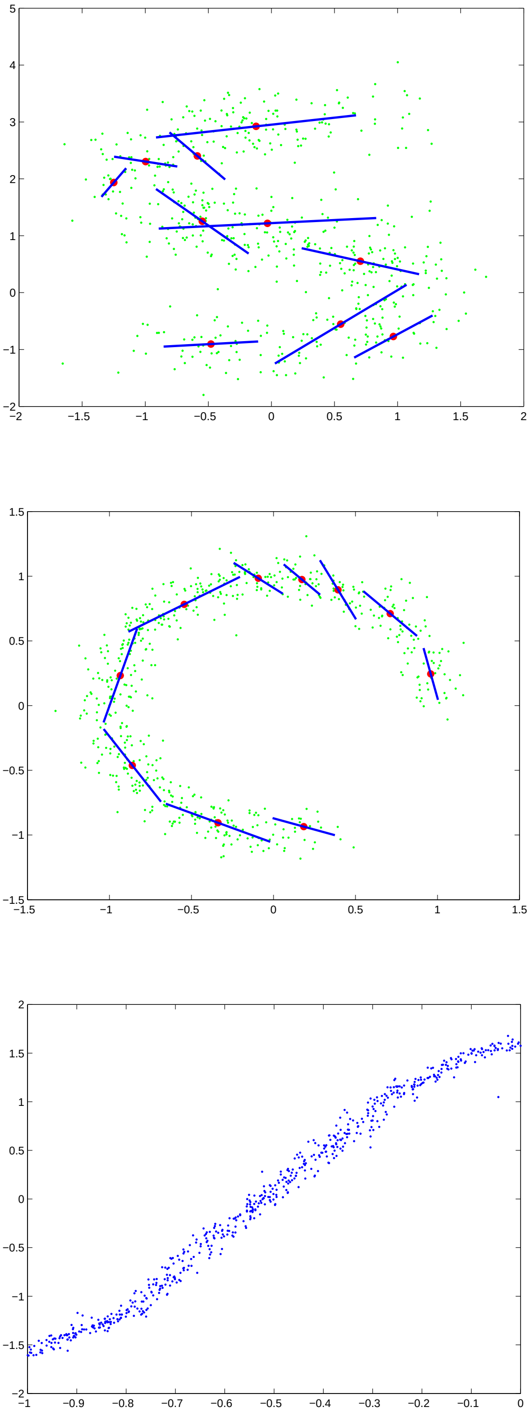

Figure 3: (Top) Two point-sets and the fitted mixture models. Local charts are indicated by the blue sticks. (Bottom) Scatter plot of the found latent coordinates (vertical) and coordinate along the generating curve (horizontal).

However, in the current work the optimization of the alignment is globally optimal and can be applied to PCA's of any dimension and latent spaces of any dimension.

As compared to [14] we do not have to compute neighbors anymore in the data space, which is costly and moreover can be hard when processing data with mixed discrete and continuous features. We only need a mixture model to produce posteriors here. In [11] the same objective and solution are used as here. We showed how using multiple mixtures in parallel can boost performance significantly and how the same technique can be used for non-linear CCA. We illustrated the viability of the presented algorithms on synthetic and real data.

In future work, we want to exploit the link with CCA further by investigating its applicability in classification and regression problems.

Acknowledgment

This research is supported as project AIF 4997 by the Technology Foundation STW, applied science division of NWO and the technology program of the Ministry of Economic Affairs.

References

[1] M.E. Tipping and C.M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11(2):443–482, 1999.

[2] C. M. Bishop, M. Svensén, and C. K. I Williams. GTM: The generative topographic mapping. *Neural Computation*, 10:215–234, 1998.

[3] J.J. Verbeek, N. Vlassis, and B. Kröse. Locally linear generative topographic mapping. In M. Wiering, editor, *Proc. of 12th Belgian-Dutch conf. on Machine Learning*, 2002.

[4] T. Kohonen. *Self-Organizing Maps*. Springer, 2001.

[5] J.J. Verbeek, N. Vlassis, and B.J.A. Kröse. Self-organization by optimizing free-energy. In M. Verleysen, editor, *Proc. of European Symposium on Artificial Neural Networks*. D-side, Evere, Belgium, 2003. To appear, preprint available from `http://www.science.uva.nl/~jverbeek`.

[6] J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, December 2000.

[7] S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.

[8] B. Schölkopf, A.J. Smola, and K. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[9] F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

[10] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[11] M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

[12] S.T. Roweis, L.K. Saul, and G.E. Hinton. Global coordination of local linear models. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, USA, 2002. MIT Press.

[13] J.J. Verbeek, N. Vlassis, and B. Kröse. Coordinating Principal Component Analyzers. In J.R. Dorronsoro, editor, *Proceedings of Int. Conf. on Artificial Neural Networks*, pages 914–919, Madrid, Spain, 2002. Springer.

[14] Y.W. Teh and S.T. Roweis. Automatic alignment of local representations. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

[15] H. Hotelling. Relation betweeen two sets of variates. *Biometrika*, 28:322–377, 1936.

[16] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, 1994.

[17] T.F. Cox and M.A.A. Cox. *Multidimensional Scaling*. Number 59 in Monographs on statistics and applied probability. Chapman & Hall, 1994.

[18] J.J. Verbeek, N. Vlassis, and B. Kröse. A k-segments algorithm for finding principal curves. *Pattern Recognition Letters*, 23(8):1009–1017, 2002.