Component-level aggregation of probabilistic PCA mixtures using variational-Bayes

Pierrick Bruneau, Marc Gelgon, Fabien Picarougne

Component-level aggregation of probabilistic PCA mixtures using variational-Bayes

Pierrick Bruneau, Marc Gelgon and Fabien Picarougne
LINA (UMR CNRS 6241) - Nantes university, France.
firstname.lastname@univ-nantes.fr

Abstract

This paper proposes a technique for aggregating mixtures of probabilistic principal component analyzers, which are a powerful probabilistic generative model for coping with a high-dimensional, non linear, data set. Aggregation is carried out through Bayesian estimation with a specific prior and an original variational scheme. We demonstrate how such models may be aggregated by accessing model parameters only, rather than original data, which can be advantageous for learning from distributed data sets. Experimental results illustrate the effectiveness of the proposal.

I. Introduction

Probabilistic PCA (PPCA) is an established dimensionality reduction technique [29] which extends standard PCA with the following advantages. First, since a probabilistic model is fit to the data, Bayesian inference may be applied, in particular to determine the appropriate model complexity. Second, mixtures of such PPCA components may be built and estimated, to approximate high dimensional data sets supported by non linear manifolds. Let us emphasize that Bayesian mixtures of PPCA enable the number of parameters to grow only as required by intrinsic data complexity. This is typically much lower than the number of parameters of Gaussian mixture models with high-dimensional covariance matrices. As a result, MPPCA models have better resilience to the curse of dimensionality.

This paper deals proposes an original and efficient technique for aggregating mixtures of PPCA models, providing a central tool for performing data analysis on distributed data sources. The key property is that only parameters of the original models, rather than original data, are required for computing the aggregated model. Therefore, the data itself does not need to be communicated over the network, ensuring a low network load and preserving the confidentiality of individual data entries.

In our scheme, aggregation of a set of MMPCA models consists in their addition, followed by a *compression* phase that seeks an optimal combination of mixture components. A central issue to mixture aggregation is the determination of the number of components. We formulate it as Bayesian estimation and show how variational inference can address it, involving the sole parameters. While this generalizes recent work [11] from Gaussian mixtures to MPPCA, iterative update equations have to be largely reconsidered.

The aggregation task might be understood in various ways. For example summaries can be merged in order to build a hierarchical representation [5] or, as mixture model posterior memberships define a clustering structure, we may seek to build cluster ensembles from distributed sources [28]. In this paper we focus on reducing the weighted sum of input mixture model parameters, by optimizing a KL divergence-based loss. Therefore, clustering information preservation is not the main criterion, even if some clustering experiments are used to illustrate the performance of our technique. A distributed social network architecture is presented in [18], to which the present contribution may give building blocks. Variational mixture learning, extensively used throughout this paper, is generalized to the whole exponential family distributions in [33]. However it differs from the present scheme in that it learns only one common subspace for all mixture components. In the context of sensor networks, the usage of shared sufficient statistics, in place of model parameters as in the present work, is emphasized in [16]. In [25], a similar approach is used, jointly with a variational Gaussian mixture setting. Finally, let us point our a growing interest in learning probabilistic models over distributed data sets [23].

Section II recalls Probabilistic PCA, and an EM algorithm for its resolution. The extension to mixtures of PPCA is exposed in section III. This model is considered along with the variational approach in section IV. Section V first discloses how mixtures of PPCA may be extended to handle components and presents the variational scheme for this new model. Section VI provides experimental results. After some discussion in section VII, conclusions and perspectives are drawn in section VIII.

II. Probabilistic PCA

A. Model

Principal Component Analysis (PCA) is a popular, baseline technique for dimensionality reduction. It outputs a linear transform $\mathcal{P}:\mathbb{R}^d \rightarrow \mathbb{R}^q$, $q < d$. For any d-dimensional observed data set $\mathbf{Y} = \{\mathbf{y}_1, \dots, \mathbf{y}_n, \dots, \mathbf{y}_N\}$, $\mathcal{P}(\mathbf{Y})$ is defined to be a reduced dimensionality representation, that captures the maximal variance of $\mathbf{Y}$.

The q first eigenvectors (i.e. associated to the q first eigenvalues, in descending order) of the $d \times d$ sample covariance matrix of $\mathbf{Y}$ were proven to be an orthogonal basis for $\mathcal{P}(\mathbf{Y})$. Let us remark that the eigenvalues denote the amount of variance captured by the transformation.

Tipping [30] proposed an alternative, probabilistic framework to determine this subspace:

Theorem 2.1: Let $\mathbf{x}_n$ and ϵ be two normally distributed random variables:

$$\begin{aligned} &p(\mathbf{x}_n) = \mathcal{N}(\mathbf{x}_n|\mathbf{0}, \mathbf{I}_q) \\ &p(\epsilon|\tau) = \mathcal{N}(\epsilon|\mathbf{0}, \tau^{-1}\mathbf{I}_d) \\ &\text{with } \mathcal{N} \text{ the multivariate normal distribution,} \\ &\text{and } \mathbf{I}_q \text{ the } q \times q \text{ identity matrix} \end{aligned} \tag{1}$$

Then $\mathbf{y}_n = \Lambda\mathbf{x}_n + \mu + \epsilon$ is marginally and conditionally normally distributed:

$$\begin{aligned} p(\mathbf{y}_n|\mu, \Lambda, \tau) &= \mathcal{N}(\mathbf{y}_n|\mu, \Lambda\Lambda^T + \tau^{-1}\mathbf{I}_d) \\ p(\mathbf{y}_n|\mathbf{x}_n, \mu, \Lambda, \tau) &= \mathcal{N}(\mathbf{y}_n|\Lambda\mathbf{x}_n + \mu, \tau^{-1}\mathbf{I}_d) \end{aligned} \tag{2}$$

This model will be further denoted as Probabilistic PCA (PPCA). Indeed, Maximum Likelihood (ML) estimates for the parameters $\{\Lambda, \tau\}$ were proven to be [30]:

$$\Lambda_{\text{ML}} = \mathbf{U}_q(\mathbf{L}_q - \tau_{\text{ML}}^{-1}\mathbf{I}_q)^{\frac{1}{2}}\mathbf{R} \tag{3}$$

$$\tau_{\text{ML}}^{-1} = \frac{1}{d-q}\sum_{j=q+1}^{d}\lambda_j \tag{4}$$

with $\{\lambda_1, \dots, \lambda_d | \lambda_i \geq \lambda_{i+j},\ \forall j \geq 0\}$ the eigenvalues of $\mathbf{S}$, the sample covariance matrix of $\mathbf{Y}$, $\mathbf{L}_q$ the diagonal matrix defined with $\{\lambda_1, \dots, \lambda_q\}$, $\mathbf{U}_q$ the matrix whose columns contain the corresponding eigenvectors, and $\mathbf{R}$ an arbitrary rotation matrix. Thus, determining ML estimates for this model is equivalent to performing the traditional PCA.

In the description above, we notice that ML solutions for the factor matrix Λ are up to a rotation matrix; if the purpose is to have the columns of Λ_{ML} matching the basis of the principal subspace, then $\mathbf{R}$ should equal $\mathbf{I}_q$, which is generally not the case. Fortunately, since:

$$\Lambda_{\text{ML}}^T\Lambda_{\text{ML}} = \mathbf{R}^T(\mathbf{L}_q - \tau_{\text{ML}}^{-1}\mathbf{I}_q)\mathbf{R} \tag{5}$$

$\mathbf{R}^T$ is recognized as the matrix of eigenvectors of the $q \times q$ matrix $\Lambda_{\text{ML}}^T\Lambda_{\text{ML}}$. Post-multiplying (3) with $\mathbf{R}^T$ then recovers the properly aligned ML solution. Let us notice that the solution is made with unitary basis vectors, scaled by decreasing eigenvalues. In other words, the columns of Λ_{ML} are ordered by decreasing magnitude.

B. EM algorithm for Maximum Likelihood solution

Unfortunately, there is no closed form solution to obtain these estimates. By inspection of the model presented in theorem 2.1, we recognize a latent variable problem, with parameters $\{\mu, \Lambda, \tau\}$, and latent variables $\mathbf{X} = \{\mathbf{x}_n\}$. Thus, using (1) and (2), the log-likelihood of the complete data (i.e. observed and latent variables) $\mathcal{L}_C = \sum_{n=1}^{N} \ln p(\mathbf{x}_n, \mathbf{y}_n|\mu, \Lambda, \tau)$ is formulated, and an EM algorithm can be proposed for the maximization of this value [13], [30].

In the remainder of this paper, τ, as a reflection of the minor eigenvalues of the sample covariance matrix, will be considered of less importance, and rather set as a static parameter, using a low value (e.g. 1). Consequently, probability distributions exposed in this paper will generally be implicitly conditioned on τ, e.g. $p(\mathbf{x}_n, \mathbf{y}_n|\mu, \Lambda, \tau)$ will be written $p(\mathbf{x}_n, \mathbf{y}_n|\mu, \Lambda)$.

The algorithm starts with initializing Λ with some random estimate (e.g. a random orthogonal matrix). The ML estimate for μ only depends on $\mathbf{Y}$, and equals the sample mean. The E step consists in finding the moments of the posterior distribution over the latent variables:

$$\mathbb{E}[\mathbf{x}_n]_{p(\mathbf{x}_n|\mathbf{y}_n,\mu,\Lambda)} = \mathbf{M}^{-1}\Lambda^T(\mathbf{y}_n - \mu) \tag{6}$$

$$\mathbb{E}[\mathbf{x}_n\mathbf{x}_n^T]_{p(\mathbf{x}_n|\mathbf{y}_n,\mu,\Lambda)} = \tau^{-1}\mathbf{M}^{-1} + \mathbb{E}[\mathbf{x}_n]\mathbb{E}[\mathbf{x}_n]^T \tag{7}$$

with $\mathbf{M} = \tau^{-1}\mathbf{I}_q + \Lambda^T\Lambda$. From now on, we will omit the probability distribution subscript for moments in the context of an EM algorithm, implicitly assuming that expectations are taken w.r.t. posterior distributions. The M step is obtained by maximizing $\mathcal{L}_C$ w.r.t. Λ:

$$\Lambda = \left[\sum_{n=1}^{N} (\mathbf{y}_n - \mu)\mathbb{E}[\mathbf{x}_n]\right]\left[\sum_{n=1}^{N} \mathbb{E}[\mathbf{x}_n\mathbf{x}_n^T]\right] \tag{8}$$

Cycling E and M step until convergence of the observed value for $\mathcal{L}_C$ to a local maximum provides us with ML values for parameters, and posterior distributions over latent variables. After convergence, and post-processing as suggested by equation (5), Λ contains a scaled principal subspace basis of the data set $\mathbf{Y}$.

The EM algorithm presented in this section was reported for consistency of the latter developments; if desirable, details can be found in [30].

III. Mixtures of PPCA

A. Model

The probabilistic formulation of the previous section allows an easy extension to mixtures of such models [30]. Indeed, as stated by theorem 2.1, a sample $\mathbf{Y}$ originating from a PPCA model is normally distributed:

$$p(\mathbf{y}_n|\mu, \Lambda) = \mathcal{N}(\mathbf{y}_n|\mu, \mathbf{C}) \tag{9}$$

where, for convenience, we defined $\mathbf{C} = \tau^{-1}\mathbf{I}_d + \Lambda\Lambda^T$.

From now on, let $\mathbf{Y}$ originate from a mixture of such distributions (hereafter called *components* of the mixture model):

$$p(\mathbf{y}_n|\omega, \mu, \Lambda) = \sum_{k=1}^{K} \omega_k \mathcal{N}(\mathbf{y}_n|\mu_k, \mathbf{C}_k) \tag{10}$$

where we used shorthand notations $\omega = \{\omega_k\}$, $\mu = \{\mu_k\}$ and $\Lambda = \{\Lambda_k\}$. These last conventions are not to confuse with that used in equation (9), where only one component is involved. In the present formulation, for simplicity we assume that the number of columns per factor matrix equals q for all components. This might not be the case (i.e. the principal subspace specific to each component might differ in terms of rank), but then the model and the algorithm for its estimation would be easily extended to specific q_k values. τ is set statically to a small value common to all components, and thus, without ambiguity, will not require subscripting.

Distribution (10) implicitly refers to $\mathbf{z}_n$, a binary 1-of-K latent variable (i.e. if $\mathbf{y}_n$ originates from the k-th component, $z_{nk} = 1$, and $z_{nj} = 0 \; \forall j \neq k$). Let us make this latter variable explicit, by expanding (10):

$$p(\mathbf{y}_n|\mathbf{z}_n, \mu, \Lambda) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{y}_n|\mu_k, \mathbf{C}_k)^{z_{nk}} \tag{11}$$

$$p(\mathbf{z}_n|\omega) = \prod_{k=1}^{K} \omega_k^{z_{nk}} \tag{12}$$

or equivalently:

$$p(\mathbf{y}_n|z_{nk} = 1, \mu, \Lambda) = \mathcal{N}(\mathbf{y}_n|\mu_k, \mathbf{C}_k) \tag{13}$$

$$p(z_{nk} = 1|\omega) = \omega_k \tag{14}$$

These binary latent variables may be collectively denoted by $\mathbf{Z} = \{\mathbf{z}_n\}$. Now let us expand (11) with the latent variables $\mathbf{X}$, thus expressing completely the mixture of PPCA components:

$$p(\mathbf{y}_n|\mathbf{z}_n, \mathbf{x}_n, \mu, \Lambda) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{y}_n|\Lambda_k \mathbf{x}_n + \mu_k, \tau^{-1}\mathbf{I}_d)^{z_{nk}} \tag{15}$$

$$p(\mathbf{x}_n|\mathbf{z}_n) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}_n|\mathbf{0}, \mathbf{I}_q)^{z_{nk}} \tag{16}$$

The mixture of PPCA (MPPCA) model is defined by the set of distributions (15), (16) and (12). These distributions can be used to form a graphical representation, shown in figure 1.

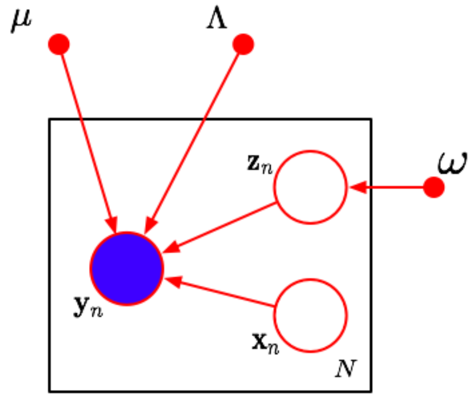

Fig. 1. Directed Acyclic Graph of the MPPCA model. Circles denote random variables (respectively known and unknown for full and empty circles). Points denote parameters to be estimated.

B. *EM algorithm for Maximum Likelihood solution*

The model presented in section III-A, and summarized in figure 1, is a latent variable model, and an EM algorithm for the estimation of its parameters $\{\omega, \mu, \Lambda\}$ and latent variables $\{\mathbf{X}, \mathbf{Z}\}$ can be derived [30]. The complete data log-likelihood $\mathcal{L}_C = \sum_{n=1}^{N} \ln p(\mathbf{y}_n, \mathbf{x}_n, \mathbf{z}_n|\omega, \mu, \Lambda)$ is formed with expressions (15), (16) and (12).

The algorithm starts with the initialization of the model parameters. All component weights ω_k are initialized to $\frac{1}{K}$, the component means μ are scattered randomly in the data domain, and the factor matrices Λ are set with random orthogonal vectors.

The E step consists in finding the moments of the posterior distributions over latent variables $\mathbf{X}$ and $\mathbf{Z}$ given the current model parameters.

The posterior moments for $\mathbf{Z}$ can be interpreted as component responsibilities, and determined straightforwardly using Bayes' rule:

$$\mathbb{E}[z_{nk} = 1] = r_{nk} \propto p(z_{nk=1}|\omega)p(\mathbf{y}_n|z_{nk} = 1, \mu, \Lambda) \tag{17}$$

with probabilities given by equations (14), (13). These estimators are then normalized so that $\sum_{k=1}^{K} r_{nk} = 1$.

As reflected by equation (16), posterior moments for $\mathbf{X}$ are now conditioned on a specific component. Actually, their expressions remain very similar to (6) and (7), but are now derived for each component:

$$\mathbb{E}[\mathbf{x}_n|z_{nk} = 1] = \mathbb{E}[\mathbf{x}_{nk}] = \mathbf{M}_k^{-1}\Lambda_k^T(\mathbf{y}_n - \mu_k) \tag{18}$$

$$\mathbb{E}[\mathbf{x}_n\mathbf{x}_n^T|z_{nk} = 1] = \mathbb{E}[\mathbf{x}_{nk}\mathbf{x}_{nk}^T] = \tau^{-1}\mathbf{M}_k^{-1} + \mathbb{E}[\mathbf{x}_{nk}]\mathbb{E}[\mathbf{x}_{nk}]^T \tag{19}$$

with $\mathbf{M}_k = \tau^{-1}\mathbf{I}_q + \Lambda_k^T\Lambda_k$.

The M step consists in maximizing the complete data log-likelihood $\mathcal{L}_C$ w.r.t. the model parameters :

$$\omega_k = \frac{1}{N}\sum_{n=1}^{N} r_{nk} \tag{20}$$

$$\mu_k = \frac{\sum_{n=1}^{N} r_{nk}(\mathbf{y}_n - \Lambda_k\mathbb{E}[\mathbf{x}_{nk}])}{\sum_{n=1}^{N} r_{nk}} \tag{21}$$

$$\Lambda_k = \left[\sum_{n=1}^{N} r_{nk}(\mathbf{y}_n - \mu_k)\mathbb{E}[\mathbf{x}_{nk}]^T\right]\left[\sum_{n=1}^{N} r_{nk}\mathbb{E}[\mathbf{x}_{nk}\mathbf{x}_{nk}^T]\right]^{-1} \tag{22}$$

Again, E and M steps are iterated until convergence of $\mathcal{L}_C$. Local maxima for scaled principal subspaces basis, specific to each component, are recovered using (5). The EM algorithm presented in this section was reported for consistency of the latter developments, as detailed in [30].

IV. Estimating a MPPCA model with the variational principle

A. Motivation

The subspace dimensionality of a PCA transform is usually determined empirically, for example by using a 80% threshold for the variance captured:

$$q_{\text{threshold}} = \arg\min_q \left\{ \left(\sum_{i=1}^{q} \lambda_q\right) > 80\% \right\} \tag{23}$$

Minka [22] proposed an alternative Bayesian criterion for the automatic estimation of q for a given PPCA model. The Laplace method is used in order to derive an asymptotical approximation to the posterior probability of q. Efficiency is achieved though the availability of the criterion in closed form, depending only on the set of eigenvalues. This can be seen as a rigorous alternative to the usual empirical procedure.

The algorithm presented in section III-B relies on the explicit knowledge of the number of components K and the number of factors per component q_k. This information is generally unknown by advance: but having defined a probabilistic formulation provides us with tools for the inference of these values. As for the single PPCA case, Bayesian methods may be employed for this purpose.

The classical solution is to train models of various complexities (i.e. varying K and q_k), and use selection criteria and validation data sets to compare these models. For example, BIC [26], AIC, or other criteria dedicated to mixture models [34] may be employed. However, this approach requires multiple trainings before selecting the adequate model structure, which might be prohibitive under some system designs.

In this section, we rather propose to adapt the variational Bayesian framework to the MPPCA model. This general purpose technique, specifically designed for rigorous approximation of posterior probability models, was proposed in [17], [2], and reviewed in [7], [27]. It was successfully applied to the Gaussian Mixture Model (GMM) case [17], [2], [7], the PPCA case [6] and mixtures of factor analyzers case [15], [4]. To our knowledge, it has not been extended to the MPPCA case yet. The mixture of factor analyzers and MPPCA models are closely related, differing only by the employed noise model (diagonal for factor analyzers, isotropic for PPCA).

In the remainder of this section, we first propose an extended probabilistic model, necessary to apply the variational Bayesian framework. Then, an EM-like algorithm for a locally optimal MPPCA model is presented.

B. Model

The model exposed in section III-A is extended, additionally taking into consideration parameters as random variables, rather than as point estimates. From the variational point of view, all parameters and latent variables are random variables, for which we want to determine posterior distributions.

Thus, expressions (15), (16) and (12) are included as such in this new model, forming the *observation*, or *likelihood* model [27]. Let us define a probabilistic model for the parameters:

$$
\begin{aligned}
p(\omega|\alpha_0) &= \mathrm{Dir}(\omega|\alpha_0) && (24)\\
p(\Lambda|\nu) &= \prod_{k=1}^{K} p(\Lambda_k|\nu_k) \\
&= \prod_{k=1}^{K}\prod_{j=1}^{q} p(\Lambda_k^{\cdot j}|\nu_{kj}) = \prod_{k=1}^{K}\prod_{j=1}^{q} \mathcal{N}(\Lambda_k^{\cdot j}|\mathbf{0}, \nu_{kj}^{-1} I_d) && (25)\\
p(\nu|a_0, b_0) &= \prod_{k=1}^{K} p(\nu_k|a_0, b_0) = \prod_{k=1}^{K}\prod_{j=1}^{q} \mathrm{Ga}(\nu_{kj}|a_0, b_0) && (26)\\
p(\mu|\mu_0, \nu_0) &= \prod_{k=1}^{K} p(\mu_k|\mu_{0k}, \nu_0) = \prod_{k=1}^{K} \mathcal{N}(\mu_k|\mu_{0k}, \nu_0^{-1} I_d) && (27)
\end{aligned}
$$

with Dir() and Ga() respectively the Dirichlet and gamma distributions, and hyper-parameters subscripted by 0. This model for the parameters may be interpreted as a prior, i.e. some clue about the range of values that the real parameters may take. The objective is then to infer a posterior for these parameters, that trades off prior information for evidence from data. The full model is summarized under a graphical form in figure 2.

In this model, hyper-parameters shape our priors, and are statically set with uninformative values. As the number of output components may be potentially high, K is set to some high value, and α_0 is set to a low value reflecting our preference towards simple models; indeed, for $\alpha_0 < 1$, the modes of the Dirichlet distribution nullify at least one of the ω_k. a_0 and b_0 are set to small values, and $a_0 = b_0$, so as to assign equal prior roles to all columns in factor matrices. These serve as prior information for $\{\nu_{kj}\}$, which are precision parameters for the columns of the factor matrices. This model setting is inspired by the Automatic Relevance Determination [20]; in brief, maximizing the likelihood of data under this model will conduct some of the ν_{kj} to high values. The associated factor columns will then tend to be void, leaving only the effective ones with low ν_{kj} values. μ_{0k} are scattered in the data domain.

Let us remark that the priors over ω and Λ are the keys to Bayesian automatic complexity determination. In other words, a posterior under the model presented in this section would naturally have the appropriate number $K' < K$ of non-zero ω_k posterior moments, and for each of the associated factor matrices, the appropriate number $q'_k < q_k$ of columns with their associated posterior ν_{kj} set to a low value. Thus q_k may be all set to $q = d - 1$, and only relevant factors are then extracted.

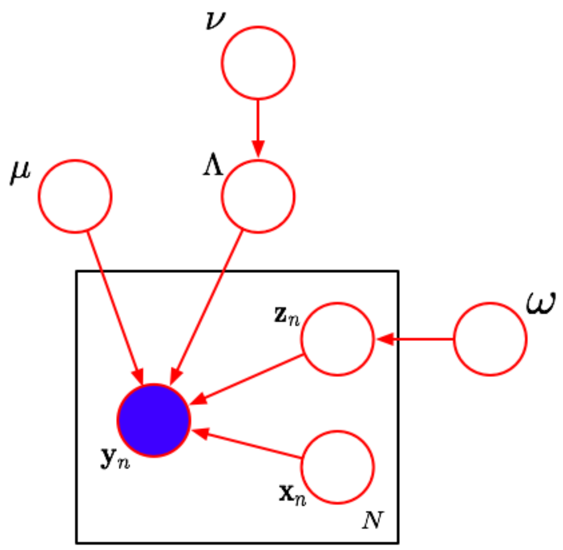

Fig. 2. Directed Acyclic Graph of the MPPCA model, with all data and parameters modeled by random variables.

C. The Variational Bayesian EM algorithm for the MPPCA

As an EM-like approach, the variational method starts with the formulation of the complete data likelihood:

$$\begin{aligned} & p(\mathbf{Y}, \mathbf{X}, \mathbf{Z}, \omega, \mu, \Lambda, \nu | \alpha_0, a_0, b_0, \mu_0, \nu_0) \\ & = p(\mathbf{Z}|\omega) p(\mathbf{Y}|\mathbf{Z}, \mathbf{X}, \Lambda, \mu) p(\mathbf{X}|\mathbf{Z}). \\ & p(\omega|\alpha_0) p(\Lambda|\nu) p(\nu|a_0, b_0) p(\mu|\nu_0) \end{aligned} \tag{28}$$

Hyper-parameters are only used as a static parametrization, and may be omitted in this expression. Thus the complete data log-likelihood may be written:

$$\mathcal{L}_C = \ln p(\mathbf{Y}, \mathbf{X}, \mathbf{Z}, \omega, \mu, \Lambda, \nu) \tag{29}$$

The true posterior $p(\mathbf{X}, \mathbf{Z}, \omega, \mu, \Lambda, \nu|\mathbf{Y})$ that maximizes $\mathcal{L}_C$ requires intractable integration. Therefore, there exists no systematic way to obtain it, and its functional form is undetermined. The purpose of the variational approach is to find an approximation to this true posterior within a simpler and well-defined family of distributions, so that the tractability is guaranteed.

A variational distributions set is defined, and factorizes exactly as does the prior distribution in the model:

$$\begin{aligned} q(\mathbf{X}, \mathbf{Z}, \omega, \mu, \Lambda, \nu) &= q(\mathbf{X}, \mathbf{Z}) q(\omega, \mu, \Lambda, \nu) \\ q(\mathbf{X}, \mathbf{Z}) &= \prod_{n=1}^{N} q(\mathbf{x}_n|\mathbf{z}_n) q(\mathbf{z}_n) \\ q(\omega, \mu, \Lambda, \nu) &= q(\omega) \prod_{k=1}^{K} q(\mu_k) q(\Lambda_k|\nu_k) q(\nu_k) \end{aligned} \tag{30}$$

Let us remark that these variational distributions have the same functional form as their respective prior. Then, the following theorem holds [27] :

Theorem 4.1: Let $\theta = \{\theta_1, \ldots, \theta_m\}$ be any factorized parameter and latent variables set, and $\mathbf{Y}$ an observed data set. The approximate distribution verifying:

$$q^*(\theta) = \arg\min_{q(\theta)} \mathrm{KL}\big[q(\theta)||p(\theta|\mathbf{Y})\big] \tag{31}$$

is obtained when:

$$q^*(\theta_i) \propto \exp\Big(\mathbb{E}_{q^*(\theta_{/i})}[\mathcal{L}_C]\Big),\ i = 1, \dots, q \tag{32}$$

or equivalently, (33)

$$\ln q^*(\theta_i) = \mathbb{E}_{q^*(\theta_{/i})}[\mathcal{L}_C] + \text{const},\ i = 1, \dots, q \tag{34}$$

with KL(.||.) the KL divergence between two probability distributions,
and $\theta_{/i}$ the set $\{\theta_j\}$ in θ for which $j \neq i$,
and $q^*(\theta_{/i}) = \prod_{j=1, j\neq i}^{m} q^*(\theta_j)$

This theorem provides us with a systematical way to obtain an approximation $q^*(\mathbf{X}, \mathbf{Z}, \omega, \mu, \Lambda, \nu)$ to the true and unknown posterior. The algorithm starts with the initialization of $m-1$ elements of $q^*()$, e.g. with their respective prior. The m-th element is updated using theorem 4.1, followed by the successive updates of the $m-1$ first elements. Each such step (i.e. succession of m updates) was proven to increase the expected value for $\mathcal{L}_C$ [7]. This property lead to the EM-like denomination. Variational updates are performed until convergence. In other words, an estimate for the optimal $q^*()$ is progressively refined.

Let us remark that $\mathbb{E}[\mathcal{L}_C]$ can be viewed as a lower bound to a constant, marginalized likelihood (i.e. integrated over all unknowns). Thus it can be computed and used to detect the convergence of the variational algorithm.

Similarly to EM, this scheme performs a local optimization, yet variational EM algorithms perform a functional optimization, as opposed to point-wise for EM. Thus, some regularity is guaranteed for the solutions, and usual degeneracies of EM schemes for mixture models are avoided [21], [7].

Using expression (53) for $\mathcal{L}_C$, and theorem 4.1, we obtain the formulae for the update of the variational distributions (i.e. each individual term in (54)):

- **E step**

$$\begin{aligned}
&\Sigma_{\mathbf{x}_k} = \left(I_q + \tau \mathbb{E}[\Lambda_k^T \Lambda_k]\right)^{-1} \\
&\mathbb{E}_{q(\mathbf{x}_n | z_{nk}=1)}[\mathbf{x}_n] = \mathbb{E}[\mathbf{x}_{nk}] = \tau \Sigma_{\mathbf{x}_k} \mathbb{E}[\Lambda_k^T](\mathbf{y}_n - \mathbb{E}[\mu_k]) \\
&\mathbb{E}_{q(\mathbf{x}_n | z_{nk}=1)}[\mathbf{x}_n \mathbf{x}_n^T] = \mathbb{E}[\mathbf{x}_{nk} \mathbf{x}_{nk}^T] = \Sigma_{\mathbf{x}_k} + \mathbb{E}[\mathbf{x}_{nk}]\mathbb{E}[\mathbf{x}_{nk}]^T \\
&r_{nk} \propto \exp\Bigg(\psi(\alpha_k) - \psi\Big(\sum_{j=1}^{K} \alpha_j\Big) - \frac{1}{2}\text{Tr}(\mathbb{E}[\mathbf{x}_{nk}\mathbf{x}_{nk}^T]) \\
&- \frac{\tau}{2}\Big[||\mathbf{y}_n||^2 + \mathbb{E}[||\mu_k||^2] - 2(\mathbf{y}_n - \mathbb{E}[\mu_k])^T \mathbb{E}[\Lambda_k]\mathbb{E}[\mathbf{x}_{nk}] \\
&- 2\mathbf{y}_n^T \mathbb{E}[\mu_k] + \text{Tr}(\mathbb{E}[\Lambda_k^T \Lambda_k]\mathbb{E}[\mathbf{x}_{nk}\mathbf{x}_{nk}^T])\Big]\Bigg)
\end{aligned} \tag{35}$$

- **Sufficient statistics**

$$\begin{aligned}
N_k &= \sum_{n=1}^{N} r_{nk} & \mathbf{y}_k^2 &= \sum_{n=1}^{N} r_{nk} ||\mathbf{y}_n||^2 \\
\mathbf{y}_k &= \sum_{n=1}^{N} r_{nk} \mathbf{y}_n & \mathbf{s}_k &= \sum_{n=1}^{N} r_{nk} \mathbb{E}[\mathbf{x}_{nk}] \\
\mathbf{sy}_k &= \sum_{n=1}^{N} r_{nk} \mathbf{y}_n \mathbb{E}[\mathbf{x}_{nk}]^T & \mathbf{S}_k &= \sum_{n=1}^{N} r_{nk} \mathbb{E}[\mathbf{x}_{nk} \mathbf{x}_{nk}^T]
\end{aligned} \tag{36}$$

- **M step**

$$
\begin{aligned}
\alpha_k &= \alpha_{0k} + N_k & a_{kj} &= a_0 + \frac{d}{2} \\
b_{kj} &= b_0 + \frac{1}{2}\sum_{i=1}^{d} \mathbb{E}[{\Lambda_k^{ij}}^2] & \Sigma_{\Lambda_k} &= (\mathrm{diag}(\mathbb{E}[\nu_k]) + \tau \mathbf{S}_k)^{-1} \\
\mathbb{E}[{\Lambda_k^{ij}}^2] &= \mathbb{E}[\Lambda_k^{i\cdot} {\Lambda_k^{i\cdot}}^T]^{jj} & \Sigma_{\mu_k} &= \Big[\nu_0 + \tau N_k\Big]^{-1} I_d
\end{aligned}
\tag{37}
$$

$$
\begin{aligned}
\mathbb{E}[\Lambda_k^T \Lambda_k] &= \sum_{i=1}^{d} \mathbb{E}[\Lambda_k^{i\cdot} {\Lambda_k^{i\cdot}}^T] \\
\mathbb{E}[\Lambda_k^{i\cdot}] &= \Sigma_{\Lambda_k} \tau \Big[\mathbf{sy}_k^{i\cdot} - \mathbb{E}[\mu_k^i] \mathbf{s}_k\Big] \\
\mathbb{E}[\mu_k] &= \Sigma_{\mu_k} \Big[\nu_0 \mu_{0k} + \tau(\mathbf{y}_k - \mathbb{E}[\Lambda_k]\mathbf{s}_k)\Big]
\end{aligned}
$$

Let us note that variational distributions can be characterized either by their first and second order moments (e.g. $\mathbb{E}[\mathbf{x}_{nk}]$ or $\mathbb{E}[{\Lambda_k^{ij}}^2]$), or their shaping hyper-parameters (e.g. α_k, b_{kj}). All are deduced by the usage of theorem 4.1.

We regrouped updates for the variational distributions over latent variables in the E step. Sufficient statistics are accumulators that are built using the current estimators for latent variables. Variational distributions over model parameters are updated using these accumulators. Updates for model parameters are regrouped in an M step, by analogy to the classic EM formulation. Practically, the algorithm starts with the initialization of the variational distributions over parameters (i.e. $q(\omega, \mu, \Lambda, \nu)$) with their respective prior, and proceeds with the E step.

Convergence may be assessed by monitoring the value of $\mathcal{L}_C$, expected w.r.t. the current estimate for the variational distribution $q^*()$. The obtained estimates for Λ_k are post-processed using (5).

The algorithm outputs a posterior MPPCA:

$$
\text{output MPPCA} = \{\omega_k, \mu_k, \Lambda_k, \tau | \alpha_k > \alpha_0\} \tag{38}
$$

The variational Bayesian algorithm for the mixture of PPCA will be further identified by VBMPPCA. An analogous application of theorem 4.1 to the GMM case was already proposed in the literature [7]. The associated algorithm will be further known as VBGMM.

As proposed in section IV-B, the algorithm is run with factor matrices of the maximal possible size, i.e. $q = d - 1$ columns. In the same section, we argued that the model ensures dimensionality reduction with precision parameters over the columns of the factor matrices. After post-processing, factor matrices are ordered by decreasing magnitude (see section II-A): so a systematic way to infer the proper dimensionality specific to each output component can be proposed:

```
Data: An output MPPCA
Result: output MPPCA with dimensionality reduced factor matrices
1  for each component $k$ in the model do
2      $currentDivergence \leftarrow +\infty$ ;
3      $curQ \leftarrow 0$ ;
4      while $currentDivergence > threshold$ do
5          $curQ \leftarrow curQ + 1$ ;
6          $\Lambda_{\text{temp}} \leftarrow$ first $curQ$ columns of $\Lambda_k$ ;
7          $currentDivergence \leftarrow \text{KL}\Big(\mathcal{N}(\mathbf{0}, \Lambda_k \Lambda_k^T + \tau^{-1}\mathbf{I}_d) || \mathcal{N}(\mathbf{0}, \Lambda_{\text{temp}} \Lambda_{\text{temp}}^T + \tau^{-1}\mathbf{I}_d)\Big)$ ;
8      end
9      $\Lambda_k \leftarrow$ first $(curQ - 1)$ columns of $\Lambda_k$ ;
10 end
```

Algorithm 1: Inference of dimensionality for output factor matrices

The KL divergence between two Gaussians is defined in closed form. The algorithm 1 tries to form the covariance of the marginal for data $\mathbf{Y}$ with a limited number of factors, and stops when the last added factor did not change significantly the shape of the distribution. This means that factors columns with index greater or equal to $curQ$ are close to zero, and can be pruned from output factor matrices. Subspaces with reduced dimensionality are thus built.

V. Aggregating mixtures of PPCA

A straightforward way of aggregating MPPCA models separately estimated on different data sources is to add them. However if sources reflect the same underlying process, the resulting mixture will generally be unnecessarily complex, i.e. redundant, with regard to that would have been estimated on the reunion of the data sets. In this section, we show first how such an input can be seen as the limit representation of a virtual data set [10]. Then, we incorporate this representation in the model described in section III in replacement of an ordinary data set. The variational EM algorithm for this model is the given. As a result, we obtain the low complexity model that best fits the data which would have been generated from the input mixture, without resorting to the data itself or any sampling scheme.

A. Model

A full sample (i.e. data and indicator variables, all observed) originating from an arbitrary and known MPPCA with L components may be denoted by $(\mathbf{Y}, \mathbf{Z}') = \{\mathbf{y}_n, \mathbf{z}'_n\}$. Let us regroup elements from this sample according to the values taken by $\mathbf{z}'_n$, and define $\hat{\mathbf{y}}_l = \{\mathbf{y}_n | z'_{nl} = 1\}$, and $\hat{\mathbf{z}}'_l = \{\mathbf{z}'_n | z'_{nl} = 1\}$.

We propose to formulate the likelihood of this sample under an unknown MPPCA model. This last model will be further known as the *target*, as opposed to the *input* model, that originated the sample we consider. To prevent ambiguities, components from the input and output models will be indexed respectively by $l \in \{1, \dots, L\}$ and $k \in \{1, \dots, K\}$.

The (known) indicator variables of the sample according to the input model are collectively denoted by $\mathbf{Z}'$; this notation is not to be confused with $\mathbf{Z}$, the (unknown) indicator variables of this same sample according to the output model. In order to keep notations concise, and without loss of generality, we will assume that all components in the input model have factor matrices with the same number of columns, q.

The output model will not be more complex than the input model, so the same q may be used to parametrize the size of factor matrices in the output model. Let us write the likelihood of the full sample under the output MPPCA model:

$$p(\mathbf{Y},\mathbf{Z}|\omega,\mu,\Lambda) = p(\mathbf{Y}|\mathbf{Z},\mu,\Lambda)p(\mathbf{Z}|\omega) \tag{39}$$

$$\text{with } p(\mathbf{Y}|\mathbf{Z},\mu,\Lambda) = \prod_{k=1}^{K}\prod_{l=1}^{L}(\mathcal{N}(\hat{\mathbf{y}}_l|\mu_k,\mathbf{C}_k))^{z_{lk}} \tag{40}$$

$$\text{and } p(\mathbf{Z}|\theta) = \prod_{k=1}^{K}\prod_{l=1}^{L}(\omega_k^{|\hat{\mathbf{z}}'_l|})^{z_{lk}} \tag{41}$$

where we used expressions (11) and (12). ω, μ and Λ collectively denote the output model parameters. Latent variable $\mathbf{X}$ is intentionally left implicit. We assumed that membership variables for all $\mathbf{y}_i$ dans $\hat{\mathbf{y}}_l$ are identical, and equal to $\mathbf{z}_l$. This hypothesis is generally acceptable, mixture models aggregation being often about combining components. Let us notice that:

$$|\hat{\mathbf{z}}'_l| = |\hat{\mathbf{y}}_l| \simeq N\omega_l \tag{42}$$

with N the cardinal of the sample considered above. Using this approximation in (41) replaces the input data by its abstraction (i.e. sample size and input model parameters). Let us rewrite (40) as follows:

$$p(\mathbf{Y}|\mathbf{Z},\theta) = \prod_{k=1}^{K}\prod_{l=1}^{L}(\mathcal{N}(\hat{\mathbf{y}}_l|\mu_k,\mathbf{C}_k))^{z_{lk}} = \prod_{k=1}^{K}\prod_{l=1}^{L}p_{lk}^{z_{lk}} \tag{43}$$

Defining $\mathcal{L}_{lk} = \ln p_{lk}$ for a specific term in (43), expanding $\hat{\mathbf{y}}_l$, and using approximation (42) gives:

$$\mathcal{L}_{lk} \simeq \sum_{j=1}^{N\omega_l}\ln\mathcal{N}(\mathbf{y}_{lj}|\omega_k,\mu_k,\mathbf{C}_k) \tag{44}$$

$$\simeq N\omega_l\Big[-\mathrm{KL}\big(\mathcal{N}(\mu_l,\mathbf{C}_l)||\mathcal{N}(\mu_k,\mathbf{C}_k)\big) - \mathrm{H}\big(\mathcal{N}(\mu_l,\mathbf{C}_l)\big)\Big] \tag{45}$$

where we used the law of large numbers to approximate the KL divergence and the entropy by a finite sum. As a consequence, N should be sufficiently large for expressions (42) and (45) to be roughly valid.

Let us underline that the likelihood (i.e. (40) and (41)) has been transformed so that it depends on the input sample only through the input model parameters. N can then be chosen arbitrarily. In other words, we use a *virtual sample*, but only process input model parameters. This idea was proposed in [32], [31]. This approximation significantly reduces the complexity of the problem, as it now only depends on the size of the input mixture L.

The KL divergence between Gaussians and the entropy of a Gaussian both have closed form expressions. Using these to expand (45) gives:

$$\begin{aligned}\mathcal{L}_{lk} = N\omega_l\Big[&-\frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln\det(\Lambda_k\Lambda_k^T + \tau^{-1}\mathbf{I}_d)\\ &-\frac{1}{2}\mathbf{Tr}\big((\Lambda_k\Lambda_k^T + \tau^{-1}\mathbf{I}_d)^{-1}[\Lambda_l\Lambda_l^T + \tau^{-1}\mathbf{I}_d\\ &+ (\mu_l-\mu_k)(\mu_l-\mu_k)^T]\big)\Big]\end{aligned} \tag{46}$$

τ is generally a low value, and its influence will be discarded in the rest of the section. As $\Lambda_l\Lambda_l^T = \sum_{j=1}^{q}\Lambda_l^{\cdot j}\Lambda_l^{\cdot j^T}$, p_{lk} can be approximated by the combined likelihoods of the input means and factor columns. Thus after renormalization:

$$p_{lk} = \left[\mathcal{N}(\mu_l|\mu_k,\mathbf{C}_k)\prod_{j=1}^{q}\mathcal{N}(\Lambda_l^{\cdot j}|\mathbf{0},\mathbf{C}_k)\right]^{N\omega_l} \tag{47}$$

Using (47) with (43) and (41), an approximation to the input sample likelihood is obtained:

$$p(\mathbf{Y},\mathbf{Z}|\omega,\mu,\Lambda) = \prod_{l=1}^{L}\prod_{k=1}^{K}\Big[\omega_k^{N\omega_l}\Big(\mathcal{N}(\mu_l|\mu_k,\mathbf{C}_k) \\ .\prod_{j=1}^{q}\mathcal{N}(\Lambda_l^{\cdot j}|0,\mathbf{C}_k)\Big)^{N\omega_l}\Big]^{z_{lk}} \tag{48}$$

In (48), all terms are members of the exponential family. Such a probability distribution, taken to any power and normalized, is still a member of the exponential family; thus $N\omega_l$ can be incorporated into the model parameters, giving:

$$p(\mathbf{Y},\mathbf{Z}|\omega,\mu,\Lambda) = p(\mathbf{Z}|\omega)p(\{\mu_l,\Lambda_l\}|\mathbf{Z},\mu,\Lambda) \tag{49}$$

where we defined:

$$p(\mathbf{Z}|\omega) = \prod_{l=1}^{L}\prod_{k=1}^{K}(\omega_k^{N\omega_l})^{z_{lk}} \tag{50}$$

$$p(\{\mu_l,\Lambda_l\}|\mathbf{Z},\mu,\Lambda) = \prod_{l=1}^{L}\prod_{k=1}^{K}\Big(\mathcal{N}(\mu_l|\mu_k,(N\omega_l)^{-1}\mathbf{C}_k) \\ .\prod_{j=1}^{q}\mathcal{N}(\Lambda_l^{\cdot j}|0,(N\omega_l)^{-1}\mathbf{C}_k)\Big)^{z_{lk}} \tag{51}$$

Let us note that on the left hand side of (49), $\mathbf{Y}$ still appears, while on the right hand side, only input model parameters are used. This convention was chosen for homogeneity to the model presented in section IV-B. Notice that the marginal for the virtual sample $p(\mathbf{Y}|\omega,\mu,\Lambda)$, that would be obtained using (49), is a mixture of products of Gaussians.

All Gaussian terms in (49) still depend on $\mathbf{C}_k$, and, using theorem 2.1, can be extended with $\mathbf{x}$ latent variables. As each component of the mixture is made of the product of $1+q$ terms, the size of $\mathbf{X}$ is increased accordingly. To prevent any ambiguity, let us define $\mathbf{X} = (\mathbf{X}_1,\mathbf{X}_2)$, $\mathbf{X}_1 = \{\mathbf{x}_{1l}\}$ and $\mathbf{X}_2 = \{\mathbf{x}_{2lj}|j\in 1\dots q\}$. The likelihood of the complete data can then be written as:

$$p(\mathbf{Y},\mathbf{X},\mathbf{Z}|\omega,\mu,\Lambda) = p(\mathbf{Z}|\omega)p(\{\mu_l,\Lambda_l\}|\mathbf{Z},\mathbf{X}_1,\mathbf{X}_2,\mu,\Lambda) \\ .p(\mathbf{X}_1,\mathbf{X}_2|\mathbf{Z}) \tag{52}$$

with $p(\{\mu_l,\Lambda_l\}|\mathbf{Z},\mathbf{X}_1,\mathbf{X}_2,\mu,\Lambda)$

$$= \prod_{l=1}^{L}\prod_{k=1}^{K}\Big(\mathcal{N}\big(\mu_l|\Lambda_k\mathbf{x}_{1l}+\mu_k,(N\omega_l\tau)^{-1}\mathbf{I}_d\big) \\ .\prod_{j=1}^{q}\mathcal{N}\big(\Lambda_l^{\cdot j}|\Lambda_k\mathbf{x}_{2lj},(N\omega_l\tau)^{-1}\mathbf{I}_d\big)\Big)^{z_{lk}}$$

and $p(\mathbf{X}_1,\mathbf{X}_2|\mathbf{Z})$

$$= \prod_{l=1}^{L}\prod_{k=1}^{K}\Big(\mathcal{N}\big(\mathbf{x}_{1l}|(N\omega_l)^{-1}\mathbf{I}_q\big)\prod_{j=1}^{q}\mathcal{N}\big(\mathbf{x}_{2lj}|(N\omega_l)^{-1}\mathbf{I}_q\big)\Big)^{z_{lk}}$$

B. Variational EM algorithm

In the previous section, we showed how the likelihood of a virtual sample may be expressed in terms of the sole parameters of an input MPPCA model, without any actual sampling scheme. Thus by substituting (52) to the terms $p(\mathbf{Z}|\omega)p(\mathbf{Y}|\mathbf{Z},\mathbf{X},\Lambda,\mu)p(\mathbf{X}|\mathbf{Z})$ in (28), we obtain:

$$\begin{aligned} p(\mathbf{Y},\mathbf{X},\mathbf{Z},\omega,\mu,\Lambda,\nu) &= p(\mathbf{Z}|\omega)p(\{\mu_l,\Lambda_l\}|\mathbf{Z},\mathbf{X}_1,\mathbf{X}_2,\mu,\Lambda) \\ &.p(\mathbf{X}_1,\mathbf{X}_2|\mathbf{Z})p(\omega|\alpha_0)p(\Lambda|\nu)p(\nu|a_0,b_0)p(\mu|\nu_0) \end{aligned} \tag{53}$$

The variational Bayesian algorithm described in section IV-C can then be extended straightforwardly. The associated variational distribution follows the factorization suggested by (53), and uses the same functional forms for its terms:

$$\begin{aligned} q(\mathbf{X}_1,\mathbf{X}_2,\mathbf{Z},\omega,\mu,\Lambda,\nu) &= q(\mathbf{X}_1,\mathbf{X}_2,\mathbf{Z})q(\omega,\mu,\Lambda,\nu) \\ q(\mathbf{X}_1,\mathbf{X}_2,\mathbf{Z}) &= \prod_{l=1}^{L}\left(q(\mathbf{z}_l)q(\mathbf{x}_{1l}|\mathbf{z}_l)\prod_{j=1}^{q}q(\mathbf{x}_{2lj}|\mathbf{z}_l)\right) \\ q(\omega,\mu,\Lambda,\nu) &= q(\omega)\prod_{k=1}^{K}q(\mu_k)q(\Lambda_k|\nu_k)q(\nu_k) \end{aligned} \tag{54}$$

Using theorem 4.1 with (53) and (54), we obtain:

- **E step** :

$$\begin{aligned} \Sigma_{\mathbf{x}_{kl}} &= \left(N\omega_l[\mathbf{I}_q+\tau\mathbb{E}[\Lambda_k^T\Lambda_k]]\right)^{-1} = (N\omega_l)^{-1}\Sigma_{\mathbf{x}_k} \\ \mathbb{E}[\mathbf{x}_{1lk}] &= \tau N\omega_l\Sigma_{\mathbf{x}_{kl}}\mathbb{E}[\Lambda_k^T]\Big(\mu_l-\mathbb{E}[\mu_k]\Big) \\ &= \tau\Sigma_{\mathbf{x}_k}\mathbb{E}[\Lambda_k^T]\Big(\mu_l-\mathbb{E}[\mu_k]\Big) \\ \mathbb{E}[\mathbf{x}_{2lkj}] &= \tau N\omega_l\Sigma_{\mathbf{x}_{kl}}\mathbb{E}[\Lambda_k^T]\Lambda_l^{\cdot j} = \tau\Sigma_{\mathbf{x}_k}\mathbb{E}[\Lambda_k^T]\Lambda_l^{\cdot j} \\ r_{lk} &\propto N\omega_l(\psi(\alpha_k)-\psi(\sum_{j=1}^{K}\alpha_j)) - \frac{N\omega_l}{2}\mathbf{Tr}(\mathbb{E}[\mathbf{x}_{1lk}\mathbf{x}_{1lk}^T] \\ &+\sum_{j=1}^{q}\mathbb{E}[\mathbf{x}_{2lkj}\mathbf{x}_{2lkj}]) - \frac{\tau N\omega_l}{2}\Bigg[\|\mu_l\|^2+\sum_{j=1}^{q}\|\Lambda_l^{\cdot j}\|^2 \\ &+\mathbb{E}[\|\mu_k\|^2]-2\mu_l^T\mathbb{E}[\mu_k]-2(\mu_l-\mathbb{E}[\mu_k])^T\mathbb{E}[\Lambda_k]\mathbb{E}[\mathbf{x}_{1lk}] \\ &-2\sum_{j=1}^{q}(\Lambda_l^{\cdot j^T}\mathbb{E}[\Lambda_k]\mathbb{E}[\mathbf{x}_{2lkj}])+\mathbf{Tr}\Big(\mathbb{E}[\Lambda_k^T\Lambda_k]\big(\mathbb{E}[\mathbf{x}_{1lk}\mathbf{x}_{1lk}^T] \\ &+\sum_{j=1}^{q}\mathbb{E}[\mathbf{x}_{2lkj}\mathbf{x}_{2lkj}^T]\big)\Big)\Bigg] \end{aligned} \tag{55}$$

- **Sufficient statistics** :

$$N_k = \sum_{l=1}^{L}N\omega_l r_{lk} \qquad \mathbf{y}_k = \sum_{l=1}^{L}N\omega_l r_{lk}\mu_l$$

$$\mathbf{s}_k = \sum_{l=1}^{L}N\omega_l r_{lk}\mathbb{E}[\mathbf{x}_{1lk}]$$

$$\mathbf{sy}_k = \sum_{l=1}^{L} N\omega_l r_{lk} \left(\mu_l \mathbb{E}[\mathbf{x}_{1lk}]^T + \sum_{j=1}^{q} \Lambda_l^{\cdot j} \mathbb{E}[\mathbf{x}_{2lkj}]^T \right)$$

$$\mathbf{y}_k^2 = \sum_{l=1}^{L} N\omega_l r_{lk} \left(||\mu_l||^2 + \sum_{j=1}^{q} ||\Lambda_l^{\cdot j}||^2 \right)$$

$$\mathbf{S}_k = \sum_{l=1}^{L} N\omega_l r_{lk} \left(\mathbb{E}[\mathbf{x}_{1lk}\mathbf{x}_{1lk}^T] + \sum_{j=1}^{q} \mathbb{E}[\mathbf{x}_{2lkj}\mathbf{x}_{2lkj}^T] \right) \quad (56)$$

- **M step** : as the output model remains identical, with dependence on the same set of sufficient statistics, update formulae (37) can be used.

These update expressions are used exactly as already suggested in section IV-C, until convergence of the modified expected value for $\mathcal{L}_C$ (i.e. the log of expression (53)). This new variational algorithm will be further known as VBMPPCA-A. The variational algorithm for the aggregation of GMM components, already proposed in [11], is denoted as VBGMM-A, and will be used for comparison in section VI.

Let us consider a single component from the output model. As stated earlier, after convergence, its column vectors are ordered by decreasing magnitude. Let us notice that each dimension of a latent variable $\mathbf{x} \in \mathbf{X}$ can be interpreted as a coefficient for a linear combination of these column vectors. Moreover, it seems natural to combine first columns of input factor matrices to form the first column of a component's output factor matrix, and so on. Therefore, during the first E step of the algorithm, instead of performing the update for $\mathbf{x}_{2lj}$ variables, these are initialized with canonical vectors:

$$\mathbf{x}_{2lj} = \mathbf{e}_j \quad (57)$$

In the subsequent E steps, the moments for these variables may be updated as usual. Updating $(1+q)$ $\mathbf{x}$ variables instead of a single one burdens our E step. But this loss is not critical, as our algorithm now scales with a number of input components, instead of a number of data elements.

Moreover, experimentally VBMPPCA-A happened to perform better with no update of $\mathbf{X}_2$. The results presented in this paper used this implementation.

At the beginning of section V-A, we assumed the same parameter q for all input components. Let us consider the case where this should not be true: then q_l is the factor matrix size associated with input component l. We set $q_{\max} = \max q_l$. Then, appending $q_{\max} - q_l$ void columns to each input factor matrix, and statically setting associated $\mathbf{x}_{2lj}$ to $\mathbf{0}$ is equivalent to using the same q for all components.

VI. Experimental evaluation

We report hereunder results obtained for synthetic and real data sets. We also supply a GPL R software package with implementation of the present proposal [8].

A. Comparison of VBGMM and VBMPPCA

We first propose a thorough comparison of VBGMM and VBMPPCA methods. For this purpose, we designed the 3 following data generating processes:

- **SYN1**: this process is able to sample randomly data elements from 4 well separated 2D Gaussian components. This signal is then linearly transformed with additive noise to form a data set with $d = 9$. An example for the original 2D signal is provided on figure 3.
- **SYN2**: this process generates elements randomly along a semi-sphere. An example sampled from this process is shown on figure 4.

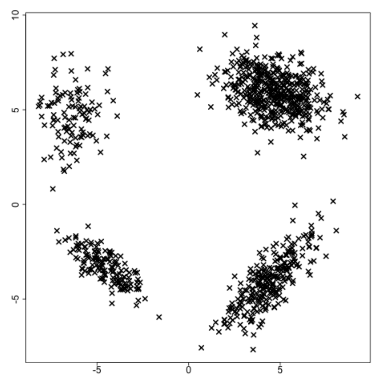
Fig. 3. Sample from SYN1

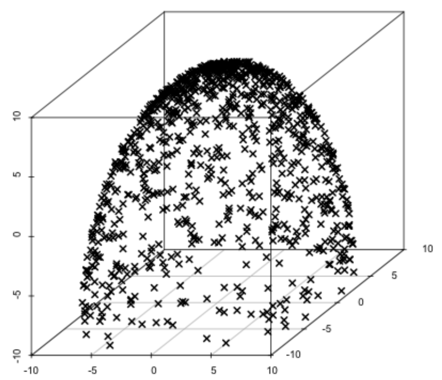
Fig. 4. Sample from SYN2

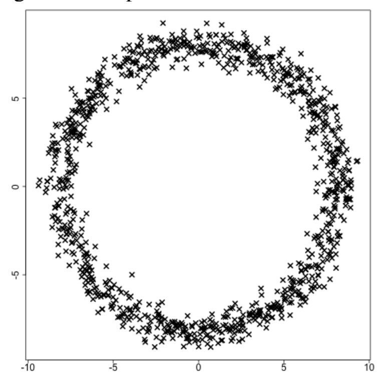
Fig. 5. Sample from SYN3

- **SYN3**: this process samples elements along a 2D circle with additive noise. As for SYN1, this original signal is transformed to 9D, with additional noise, using a linear transform. An example of the 2D process is shown on figure 5.

The linear transforms used to generate samples from SYN1 and SYN3 are designed to produce artificial linearly redundant information. Thus a method like PCA should recognize the original signal in this context. SYN2 is characterized by a 2D manifold within a 3D space, which dimensionality reduction methods should also be able to recognize. Employed dimensionalities (up to 9 in our 3 examples) may seem modest, but were chosen for their clear identifiability, and for tractability of VBGMM (which is $O(d^2)$).

Our processes were used to generate training data sets of various sizes. Then 10 testing sets of 1000 elements were generated, and will serve for validation purposes.

100 training sets were generated for each size, and each was used to estimate a model, either with VBGMM or VBMPPCA. VBMPPCA is parametrized with the maximal potential rank for each component, i.e. $q = d - 1$. The quality of each model was assessed using the following characteristics:

- The effective rank of the component subspaces.
- The effective number of components K'.
- The average log-likelihoods of the validation data sets. In order to perform this operation, MPPCA models are transformed to GMM using the identity $\mathbf{C}_k = \tau^{-1}\mathbf{I}_d + \Lambda_k\Lambda_k^T$. In the specific case of SYN1 and SYN3, only the 2 relevant dimensions were used.

These measures were averaged over the training and testing sets, and are reported in tables I, II and III. Each measure is associated to its standard deviation (bracketed in the tables).

Let us highlight a limitation of the VBMPPCA method: many local minima of the algorithm are models with some of their components fitting very few data elements with pure noise (i.e. posterior Λ_k is close to $\mathbf{0}$ for these

	training sets size				
	50	100	200	500	1000
q_{output} (MPPCA)	0.59 [0.29]	0.85 [0.31]	1.13 [0.31]	1.51 [0.21]	1.66 [0.16]
K' (GMM)	**3.95 [0.36]**	**3.84 [0.37]**	**3.84 [0.37]**	**3.93 [0.26]**	3.95 [0.22]
K' (MPPCA)	5.25 [1.19]	5.37 [1.23]	4.78 [0.91]	4.15 [0.52]	**4.03 [0.36]**
likelihood (GMM)	**-4709 [307]**	-4527 [104]	-4415 [117]	**-4290 [79]**	**-4239 [91]**
likelihood (MPPCA)	-4742 [828]	**-4417 [43]**	**-4380 [17]**	-4379 [48]	-4373 [27]

TABLE I

COMPARISONS BETWEEN VBGMM AND VBMPPCA, SYN1 PROCESS. BEST RESULTS ARE BOLD-FACED.

	training sets size				
	50	100	200	500	1000
q_{output} (MPPCA)	0.13 [0.14]	0.15 [0.08]	0.20 [0.10]	0.52 [0.25]	1.22 [0.36]
K' (GMM)	**7.01 [1.31]**	**9.25 [1.23]**	**10.9 [1.26]**	**13.84 [1.27]**	16.83 [1.44]
K' (MPPCA)	8.75 [1.88]	17.63 [2.51]	25.97 [4.49]	18.72 [7.22]	**8.44 [2.70]**
likelihood (GMM)	**-9321 [583]**	**-8352 [109]**	**-7981 [60]**	**-7581 [29]**	**-7321 [20]**
likelihood (MPPCA)	-19882 [4357]	-11202 [1008]	-8964 [279]	-8352 [151]	-8269 [131]

TABLE II

COMPARISONS BETWEEN VBGMM AND VBMPPCA, SYN2 PROCESS. BEST RESULTS ARE BOLD-FACED.

components). An illustrating example is provided in figure 6. Indeed, PCA is under-determined when applied on an insufficient support or data. In our Bayesian setting, this degeneracy is regularized with void factors.

This tendency for models fitted with VBMPPCA to include an excessive number of small components generally leads to higher K' values (see tables I and II). This phenomenon also biases the average q_{output} value towards $\mathbf{0}$, and may cause poor likelihood values, as observed with the SYN2 process.

This effect is more obvious when the data support is not sufficient (i.e. the training set size is too small, such as in figure I for sets of 50 elements), or when the dimensionality of the input data is too small (e.g. SYN2, see table II). In such cases, using VBGMM is preferable.

In cases where input data is more plentiful, or when the dimensionality is higher, the advantage of using VBMPPCA becomes more apparent. Indeed, VBMPPCA behaves well with SYN1. The correct subspace dimensionalities tend to be recovered (2 for true data, average of 1.66 with training sets of 1000 elements). Likelihood and estimated K' values indicate that both VBGMM and VBMPPCA find the expected results.

The results with SYN3 underline the good behavior of VBMPPCA with shapes lying on manifolds. Indeed, when VBGMM tends to find only 1 group (which is inadequate, especially regarding the mode of the data density), VBMPPCA easily identifies a good set of components to fit the input data (see figure 7).

	training sets size		
	50	100	200
q_{output} (MPPCA)	0.57 [0.19]	0.69 [0.20]	0.84 [0.21]
K' (GMM)	3.66 [1.45]	2.39 [1.36]	1.79 [1.17]
K' (MPPCA)	**7.55 [1.22]**	**7.78 [1.41]**	**7.17 [1.17]**
likelihood (GMM)	-5882 [263]	-6004 [344]	-6112 [354]
likelihood (MPPCA)	**-5546 [420]**	**-5209 [53]**	**-5163 [61]**

	training sets size		
	500	1000	
q_{output} (MPPCA)	1.21 [0.22]	1.49 [0.19]	
K' (GMM)	1.16 [0.65]	1.03 [0.30]	
K' (MPPCA)	**6.46 [0.85]**	**6.50 [0.72]**	
likelihood (GMM)	-6294 [214]	-6335 [106]	
likelihood (MPPCA)	**-5149 [57]**	**-5138 [48]**	

TABLE III

COMPARISONS BETWEEN VBGMM AND VBMPPCA, SYN3 PROCESS. BEST RESULTS ARE BOLD-FACED.

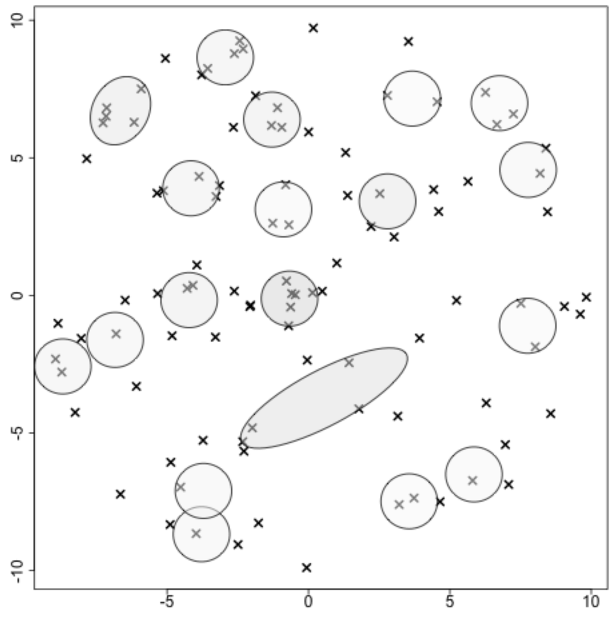

Fig. 6. A set from SYN2 process (projected on 2 dimensions), used to train a MPPCA model.

B. Comparison of VBGMM-A and VBMPPCA-A

The models that were estimated in section VI-A are then used as pools of input models for aggregations with VBGMM-A and VBMPPCA-A algorithms. More specifically, a separate pool is formed with models from each training set size. Pools associated with training set sizes 100 and 1000 are retained in this section.

Each experiment is conducted with varying amounts of input models selected randomly in their pool. 10 runs are performed for each possible configuration (i.e. a specific training set size and amount of input models to aggregate). For a fair comparison, only input MPPCA models are used: the same selected inputs are used with VBMPPCA-A, and simply converted to GMM models before using VBGMM-A.

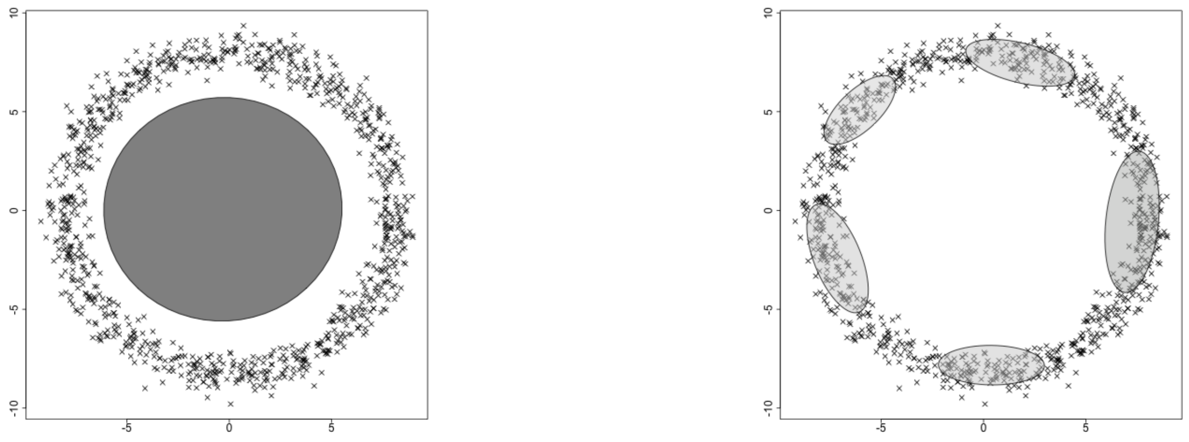

Fig. 7. GMM (left) and MPPCA (right) trained with a data set from SYN3.

MPPCA model fitting was shown to be irrelevant with SYN2 in section VI-A. Thus SYN1 and SYN3 processes were solely considered in this section. In tables IV and V, averaged quality measures and their standard deviations are indicated. The log-likelihood values are estimated using the validation data sets presented in section VI-A.

Even if the difference may seem slight, VBMPPCA-A seems to perform better than VBGMM-A (see tables IV and V). As in the previous section, the rank associated to component subspaces tend to be better recovered when using bigger training data sets. Models estimated with VBMPPCA-A seem to be associated with better likelihoods, at the expense of some complexity (i.e. K' tends to be higher with VBMPPCA-A).

Let us note that while VBMPPCA used $q = d-1$, VBMPPCA-A is parametrized according to its input components (see end of section V-B). Each of these have $q_{\text{output}} < d - 1$ set as a posterior w.r.t. to their respective training set. Thus generally $q < d - 1$ when using VBMPPCA-A. As the complexity of the latter is $O(q^2)$, its computational cost is much lower that VBGMM-A performed over the same inputs, with comparable results.

	# of input models, using training set size = 100		
	2	10	50
q_{output} (MPPCA)	1.11 [0.19]	1.03 [0.16]	1.18 [0.14]
K' (GMM)	4.90 [0.99]	**4.30 [0.48]**	**4.30 [0.68]**
K' (MPPCA)	**4.50 [0.97]**	6.30 [1.25]	7.80 [1.23]
likelihood (GMM)	**-4391 [52]**	-4356 [8]	-4425 [216]
likelihood (MPPCA)	-4425 [186]	**-4334 [6]**	**-4325 [3]**
	# of input models, using training set size = 1000		
	2	10	50
q_{output} (MPPCA)	1.76 [0.02]	1.80 [0.11]	1.74 [0.15]
K' (GMM)	**4.00 [0.00]**	4.10 [0.32]	**4.40 [0.70]**
K' (MPPCA)	4.10 [0.32]	**3.90 [0.32]**	4.50 [0.53]
likelihood (GMM)	-4368 [4]	**-4368 [4]**	-4392 [58]
likelihood (MPPCA)	**-4366 [4]**	-4460 [290]	**-4369 [5]**

TABLE IV
COMPARISONS BETWEEN VBGMM-A AND VBMPPCA-A, SYN1 PROCESS. BEST RESULTS ARE BOLD-FACED.

C. Incremental aggregation of MPPCA models

In section VI-B, the input models, i.e. the selection we mean to aggregate in an experiment, are all available at once. In some architectural contexts, models may become available incrementally, as their production and broadcast over a network progresses. Thus we may consider the possibility to update a reference model (see figure 8).

	# of input models, using training set size = 100		
	2	10	50
q_{output} (MPPCA)	0.91 [0.11]	1.03 [0.08]	1.17 [0.10]
K' (GMM)	6.70 [1.16]	**7.30 [0.67]**	**6.60 [1.26]**
K' (MPPCA)	**6.60 [1.43]**	8.30 [1.34]	8.00 [1.15]
likelihood (GMM)	**-5237 [70]**	-5220 [87]	-5276 [87]
likelihood (MPPCA)	-5343 [247]	**-5156 [47]**	**-5111 [30]**

	# of input models, using training set size = 1000		
	2	10	50
q_{output} (MPPCA)	1.75 [0.22]	1.88 [0.10]	1.80 [0.14]
K' (GMM)	**6.30 [0.48]**	7.60 [1.43]	**6.40 [0.97]**
K' (MPPCA)	6.30 [0.95]	**7.20 [1.03]**	8.60 [1.07]
likelihood (GMM)	**-5259 [90]**	-5262 [80]	-5319 [66]
likelihood (MPPCA)	-5270 [84]	**-5171 [75]**	**-5119 [23]**

TABLE V

COMPARISONS BETWEEN VBGMM-A AND VBMPPCA-A, SYN3 PROCESS. BEST RESULTS ARE BOLD-FACED.

Let us consider again the input model collection of section VI-B. Specifically, we use the sub-collection associated with training sets of 1000 elements, for SYN1 and SYN3 processes. We propose to simulate an incremental aggregation, by updating a reference MPPCA model successively with all 100 input models available for each process. The quality of the model eventually obtained is then assessed. This experiment is performed 10 times. Averaged measures and standard deviations associated with these runs are reported in table VI.

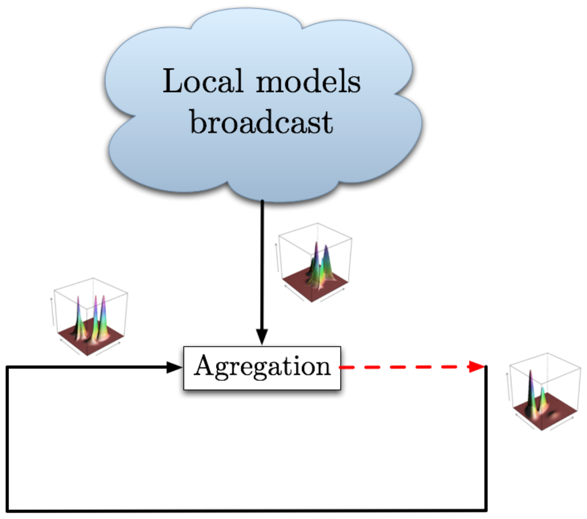

Fig. 8. Incremental aggregation in the context of local models broadcasted on a network

Results shown in table VI indicate that the usage of VBMPPCA-A in an incremental setting challenges the standard described in section VI-B, either regarding to the number of components or the likelihood of the estimated models.

D. *Experiments on real data sets*

In this section, we use our methods along with the following two real data sets:

- *Pen-based recognition of handwritten digits data set* (further denoted as *pen*) [1]: this data set was built with measurements of digit drawings on a tablet. Its 10992 elements are described by 16 numeric attributes. These are pretty naturally gathered in 10 classes (i.e. one for each digit).
- *MNIST database of handwritten digits* (further denoted as *hand*) [19]: the training set of this database (60000 elements) was used for these experiments. Each element is described by a 28x28 image patch of a specific

	SYN1	SYN3
q_{output}	1.76 [0.02]	1.75 [0.13]
K'	4.10 [0.32]	7.10 [0.99]
likelihood	-4379 [46]	-5240 [45]

TABLE VI
RESULTS OF INCREMENTAL AGGREGATIONS WITH SYN1 AND SYN3 PROCESSES.

digit (see figure 9); these can be viewed as numeric 784-dimensional vectors of pixel intensities. As for the *pen* data set, the classes are naturally defined as the digit contained in each image patch.

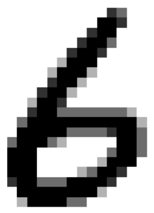

Fig. 9. Sample image patch from the *hand* data set

First, training and testing sets are delimited within these data sets :

- The 5000 first elements constitute the training set for *emph*. The complementary set is used for validation.
- 10000 elements are randomly selected in *hand* to form the validation set. The complementary set of 50000 elements is used for training.

Similarly to our experiments on synthetic data, the training data sets further divided in subsets (25 subsets of 200 elements for *pen*, 50 subsets of 1000 elements for *hand*). Models are estimated on these subsets using either VBGMM or VBMPPCA.

Measures and comparisons are provided in table VII for models estimated on subsets of *pen* and *hand*. As VBGMM and GMM likelihood computation are intractable on 784-dimensional data sets, only relevant measurements are indicated. To limit the computational load with VBMPPCA, the explicit constraint $q < 9$ is used for both data sets.

Estimating a mixture model on a data set can be interpreted as a clustering task. Thus validation subsets, and their associated true labels, are used to compute the clustering error of each output model. We use the error function proposed in [14], [24]. This function is specially adapted to the clustering task in that it measures how two label sets are related.

VBMPPCA lead to lower likelihood values and more complex models for *pen* data set. This may be due to the constraint imposed on q. However, measured clustering errors indicate sensible estimations. Let us note that the output subspace ranks for components vary with the data set: only 2.76 dimensions are used in average among the 8 possible for *pen*, while almost all are used for *hand*.

In figure 10, we show some examples of means for MPPCA models estimated on subsets of *hand*. A label is estimated for each component using the validation data set, by using a majority vote. The 4 first factors of a component are also presented as an example. Lighter or darker gray values indicate where the influence of this factor is located in the image space. We see how each factor embodies different variations of elements in its associated

	pen	*hand*
q_{output} (MPPCA)	2.76 [0.27]	7.17 [0.29]
K' (GMM)	12.7 [1.8]	
K' (MPPCA)	24.2 [3.0]	34.8 [6.1]
likelihood (GMM)	-245663 [3088]	
likelihood (MPPCA)	-325406 [15228]	
error (%, GMM)	12.6 [3.0]	
error (%, MPPCA)	9.0 [1.5]	10.8 [1.2]

TABLE VII
RESULTS FOR VBGMM AND VBMPPCA ON *pen* AND *hand* DATA SETS

component.

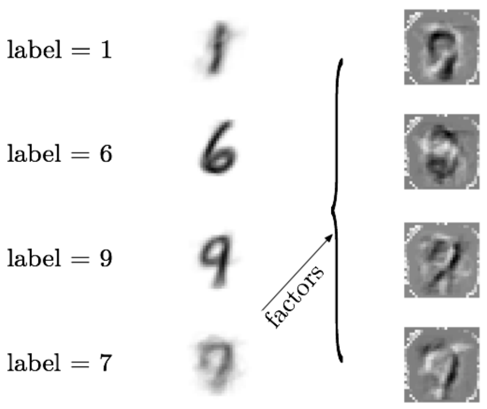

Fig. 10. Mean and factor examples of MPPCA models on the *hand* data set.

Aggregations are then performed using all these input models. Results, when available, are presented in table VIII. As in section VI-B, MPPCA input models are used with both VBMPPCA-A and VBGMM-A. Conversions are performed when appropriate.

	pen	*hand*
q_{output} (MPPCA)	3.27 [0.31]	7.12 [0.37]
K' (GMM)	13.2 [1.7]	
K' (MPPCA)	13.0 [1.9]	27.5 [6.3]
likelihood (GMM)	-270609 [6706]	
likelihood (MPPCA)	-356815 [19803]	
error (%, GMM)	16.9 [3.2]	
error (%, MPPCA)	11.2 [1.4]	11.7 [1.9]

TABLE VIII
RESULTS FOR VBGMM-A AND VBMPPCA-A ON *pen* AND *hand* DATA SETS

Likelihood values in table VIII suggest that VBGMM-A performs better than VBMPPCA-A on the *pen* data set, and that agregation of MPPCA models imply a quality loss relatively to the inputs (see also table VII). Despite this apparent flaw, classification errors for aggregated MPPCA models tend to be much better than what is obtained for GMM aggregation, with similar model complexities. Compared to baseline MPPCA estimation (see table VII), aggregation causes a slight loss in classification error, associated with a much lower model complexity. Thus MPPCA aggregations produce good summaries of the input models.

VII. Discussion on the MPPCA model

In VBMPPCA and VBMPPCA-A algorithms, the moments of variational distributions are strongly coupled (see equations (35), (36), (37), (55) and (56)). This may cause biases, or, from a computational point of view, decreases of the lower bound value associated with VBMPPCA and VBMPPCA-A algorithms (see section IV-C). Practically, after a sufficient number of iterations, this bias eventually disappears; but this compromises the usage of the lower bound as a reliable convergence criterion. Instead, measures of the agitation of $\mathbf{Z}$ may be used [4].

VIII. Conclusion and perspectives

In this paper we proposed a new technique that performs the aggregation of MPPCA models. A fully probabilistic and Bayesian framework, along with the possibility to deal with high dimensional data motivated our approach. Theoretical justifications were developed, and some illustrative results were detailed.

Results obtained in section VI show that processing over subspaces and principal axes provide an interesting guideline to carry out aggregations. Components are fitted according to their intrinsic subspace dimensionality, instead of a crisp covariance structure combination. Furthermore, properties of ML PPCA solutions (and particularly the ability to recover easily the ordered principal axes set), provide us with some strong prior knowledge, and a well-defined initialization scheme.

Besides providing building blocks for distributed, incremental and on-line learning, we believe there should be some interesting derivation of the mixture of PPCA in the domain of semi-supervised clustering. Let us suppose, as formalized and used in [3], that we have a set of ”must-link” constraints (i.e. pairs of data items that should be clustered together), and ”must-not-link” constraints (i.e. data that should not be in the same group). Under the hypothesis of compact clusters (i.e. each cluster should lie on a compact and approximately linear manifold, see [12] for discussion), we may these as follows :

- *must-link* : for now, each factor matrix is initialized with a random orthogonal matrix (e.g. see section III-B). Data items we believe to be in the same component may be used to influence this initialization, and guide the algorithm towards a specific local minimum (as we said earlier, real world data might be strongly non gaussian, so there might be several posterior models with similar likelihoods but significant pairwise KL divergences).
- *must-not-link* : As we employed a Bayesian integration scheme, this kind of constraint might be modeled by some *pdf* (e.g. as in [9]). This remark is not specific to the MPPCA model ; but we might exploit the fact we maintain principal subspace structures.

References

[1] F. Alimoglu. Combining multiple classifiers for pen-based handwritten digit recognition. Technical report, Institute of Graduate Studies in Science and Engineering, 1996.

[2] H. Attias. A variational Bayesian framework for graphical models. In *Advances in Neural Information Processing Systems - MIT Press*, volume 12, 2000.

[3] S. Basu, M. Bilenko, A. Banerjee, and R. J. Mooney. Probabilistic semi-supervised clustering with constraints. In *Semi-Supervised Learning*. MIT Press, 2006.

[4] M. J. Beal. *Variational Algorithms for approximate inference*. PhD thesis, University of London, 2003.

[5] M. Bechchi, G. Raschia, and N. Mouaddib. Merging distributed database summaries. In *ACM Conference on Information and Knowledge Management (CIKM'2007)*, pages 419–428, Lisbon, Portugal, 2007.

[6] C. M. Bishop. Variational principal components. In *Proceedings of 9th ICANN (Int. Conf. on Artificial Neural Networks)*, volume 1, pages 509–514, 1999.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006.

[8] P. Bruneau. Vbmix : a R package for variational-Bayes learning and aggregation of mixtures of probabilistic principal components analyzers. Technical report, Nantes university, HAL report hal-00567289, http://hal.archives-ouvertes.fr/, 2011.

[9] P. Bruneau, M. Gelgon, and F. Picarougne. Parsimonious variational-Bayes mixture aggregation with a Poisson prior. In *Proceedings of European Signal Processing Conference (EUSIPCO'2009)*, 2009.

[10] P. Bruneau, M. Gelgon, and F. Picarougne. Aggregation of probabilistic PCA mixtures with a variational-Bayes technique over parameters. In *International Conference on Pattern Recognition (ICPR'10)*, Istanbul, Turkey, 2010.

[11] P. Bruneau, M. Gelgon, and F. Picarougne. Parsimonious reduction of Gaussian mixture models with a variational-Bayes approach. *Pattern Recognition*, 43:850–858, March 2010.

[12] O. Chapelle, B. Scholkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, 2006.

[13] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society - B Series*, B(39):1–38, 1977.

[14] E. B. Fowlkes and C. L. Mallows. A method for comparing two hierarchical clusterings. *J. Am. Stat. Assoc.*, 78:553–569, 1983.

[15] Z. Ghahramani and M. J. Beal. Variational inference for Bayesian mixtures of factor analysers. In *Advances in Neural Information Processing Systems*, 2000.

[16] D. Gu. Distributed em algorithm for Gaussian mixtures in sensor networks. *IEEE Trans. Neural Networks*, 19(7):1154–1166, 2008.

[17] M. I. Jordan, Z. Ghahramani, Jaakkola T. S., and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

[18] A. Kermarrec. Challenges in personalizing and decentralizing the web: An overview of GOSSPLE. In *Lecture Notes in Computer Science*, volume 5873, pages 1–16, 2009.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[20] D.J.C. MacKay. Probable networks and plausible predictions — a review of practical Bayesian methods for supervised neural networks. *Network: Computation in Neural Systems*, 6:469–505, 1995.

[21] G. J. McLachlan and D. Peel. *Finite Mixture Models*. John Wiley and Sons, 2000.

[22] T. P. Minka. Automatic choice of dimensionality for pca. In *Neural Information Processing Systems*, 2000.

[23] D. Newman, A. Asuncion, P. Smyth, and M. Welling. Distributed algorithms for topic models. *Journal of Machine Learning Research*, 2010.

[24] F. Picarougne, H. Azzag, G. Venturini, and C. Guinot. A new approach of data clustering using a flock of agents. *Evolutionary Computation*, 15(3):345–367, 2007.

[25] B. Safarinejadian, M. Menhaj, and M. Karrari. Distributed variational Bayesian algorithms for gaussian mixtures in sensor network. *Signal Processing*, 90(4):1197–1208, 2010.

[26] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6:461–464, 1978.

[27] V. Smidl and A. Quinn. *The Variational Bayes Method in Signal Processing*. Springer, 2006.

[28] A. Strehl and J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2003.

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

[30] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society - B Series*, 61(3):611–622, 1999.

[31] N. Vasconcelos. Image indexing with mixture hierarchies. In *Proceedings of IEEE Conference in Computer Vision and Pattern Recognition*, volume 1, pages 3–10, 2001.

[32] N. Vasconcelos and A. Lippman. Learning mixture hierarchies. In *Advances in Neural Information Processing Systems - MIT Press*, volume II, pages 606–612, 1998.

[33] K. Watanabe, S. Akaho, and S. Omachi. Variational Bayesian mixture model on a subspace of exponential family distributions. *IEEE Trans. Neural Networks*, 20(11):1783–1796, 2009.

[34] T. Xiang and S. Gong. Model selection for unsupervised learning of visual context. *International Journal of Computer Vision*, 69(2):181–201, 2006.