A variational-Bayes technique for aggregating probabilistic PCA mixtures from their parameters

Pierrick Bruneau, Marc Gelgon, Fabien Picarougne

A variational-Bayes technique for aggregating probabilistic PCA mixtures from their parameters

Pierrick Bruneau, Marc Gelgon, Fabien Picarougne

Abstract

This paper proposes a solution to the problem of aggregating versatile probabilistic models, namely mixtures of probabilistic principal component analyzers. These models are a powerful generative form for capturing a high-dimensional, non Gaussian, data set. They simultaneously perform mixture fitting and dimensionality reduction. We demonstrate how such models may be advantageously aggregated by accessing mixture parameters only, rather than original data. Aggregation is carried out through Bayesian estimation with a specific prior and an original variational scheme. Experimental results illustrate the effectiveness of the proposal.

Index Terms

Mixture models, aggregation, variational Bayes.

I. Introduction

This paper proposes an effective and original solution to the problem of aggregating versatile probabilistic models, namely mixtures of probabilistic principal component analyzers (MPPCA hereafter [22]). The contribution of the paper is a demonstration of how this aggregation can be conducted by accessing only the parameters of the models to merge, rather than the original data.

Probabilistic PCA (PPCA) is a dimensionality reduction technique that 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 capture high dimensional data sets supported by non linear manifolds. Let us emphasize that Bayesian MPPCAs 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 with the aggregation of such models, providing a central tool for performing data analysis on distributed data sources. There is a growing research interest on this matter, as peer-to-peer infrastructures, grids and sensor networks are increasingly hosting data sets to which statistical learning comes useful. Our scheme focuses on statistical learning of global data

models through the aggregation of a set of local, parametric models. Its main features are as follows:

- motivated by low computational or network-load cost, or for protecting confidentiality of individual data entries, our proposal is designed so that only model parameters need be accessed to aggregate mixtures of PPCA. In other words, the scheme operates on the components of the mixtures to aggregate, rather than original data.
- 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 EM-like variational inference can address it. While this generalizes recent work [10] 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 [6] ; or, as mixture model posterior memberships define a clustering structure, we may seek to build cluster ensembles from distributed sources [21].

In this paper, we focus on reducing the weighted sum of input mixture model parameters, thus optimizing a KL divergence-based loss. We do not address the important issue of weighing mixtures to be added, in a manner that would varying over the feature space (to specialize them, as mixture of experts [8]). Further, the clustering target does not drive the definition of our optimality criterion, even if clustering experiments are used to illustrate the performance of our technique.

A distributed social network architecture is presented in [15], 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 [25]. 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 [14]. In [18], a similar approach is used, jointly with a variational Gaussian mixture setting.

Section II recalls Probabilistic PCA and its extension to mixtures of PPCA; we then sketch an associated variational-Bayes estimation procedure. Section III first discloses how mixtures of

PPCA may be extended to handle components and presents a novel estimation scheme for this model. Section IV provides experimental results. Section V draws conclusions and perspectives.

II. Mixtures of Probabilistic Principal Components Analysers (MPPCA)

A. A probabilistic view to PCA

Principal Component Analysis is a popular, baseline technique for dimensionality reduction. Given a d-dimensional data set, the principal subspace is generally obtained by diagonalizing the sample covariance, i.e. by seeking an eigendecomposition of this $d \times d$ matrix. Tipping [23] proposed an alternative, probabilistic framework to PCA, based on the assumption that every data item y is generated by transforming a zero mean unit variance q-dimensional variable x ($q < d$) with additive isotropic noise.

$$y = \Lambda x + \mu + \epsilon \tag{1}$$

Let us define the associated probability density functions (*pdf*) :

$$p(y|x) = \mathcal{N}(y|\Lambda x + \mu, \sigma^2 I_d) \tag{2}$$

$$p(x) = \mathcal{N}(x|0, I_q) \tag{3}$$

$$p(\epsilon) = \mathcal{N}(\epsilon|0, \sigma^2 I_d) \tag{4}$$

Results for linear Gaussian models [8] provide the following marginal distribution for y :

$$p(y) = \mathcal{N}(y|\mu, \Lambda\Lambda^T + \tau^{-1} I_d) \tag{5}$$

where $\tau = \sigma^{-2}$. Λ is a $d \times q$ matrix, usually known as the *factor* matrix. In other words, each data item y is constructed by the addition of μ, a linear combination of the columns of Λ (columns denoted later as *factors*), and isotropic noise. x contains the coefficients of the linear combination. Let us define $C = \Lambda\Lambda^T + \tau^{-1} I_d$, for further use.

ML estimates for Λ were proven to span the principal subspace of the data sample [23]. This estimate has no closed-form solution, but may be obtained through an iterative scheme. More precisely, update formulas can be derived for parameters $\theta = \{\mu, \Lambda, \tau\}$ and latent variables x by differentiation, implementing an EM algorithm [23].

The ML solution obtained for the PPCA model is up to an arbitrary rotation matrix. Still, this matrix can be recovered by diagonalizing $\Lambda_{ML}^T \Lambda_{ML}$ [23], with limited computational overhead as this matrix is $q \times q$. Post-multiplying Λ_{ML} by this rotation matrix allows us to obtain the scaled eigenvectors for our subspace, ordered by decreasing magnitude.

B. Handling a mixture of PPCA

1) ML approach: The framework presented in paragraph II-A is naturally extended by introducing a latent variable z indicating the membership of a data item to a PPCA model (called component hereafter). A set of weights $\{\omega_k\}$ is associated with K components to describe the relative importance of components. z is a binary 1-of-K variable, meaning that if any item y belongs to the component k, then $z_k = 1$ and $z_j = 0, \forall j \neq k$. Thus, a multimodal density is fitted on the data set, and each component of the mixture density determines its principal subspace. For a data item, the associated *pdf* is:

$$p(z) = \prod_k^K {\omega_k}^{z_k} \quad p(y|z) = \prod_k^K \mathcal{N}(y|\mu_k, C_k)^{z_k} \tag{6}$$

$$p(y) = \sum_k^K \omega_k \mathcal{N}(y|\mu_k, C_k) \tag{7}$$

Consequently, a data set $\mathbf{y} = \{y_1, \ldots, y_N\}$ has a likelihood function defined as :

$$p(\mathbf{y}) = \prod_n^N \sum_k^K \omega_k \mathcal{N}(y_n|\mu_k, C_k) \tag{8}$$

Associated latent variables may be collectively denoted by $\mathbf{z} = \{z_1, \ldots, z_N\}$ and $\mathbf{x} = \{x_1, \ldots, x_N\}$. In eqn. (8), $\mathbf{x}$ and $\mathbf{z}$ are left implicit. Expanding this equation using (6), (3) and (2) makes our latent variables set explicit. Differentiation can then be performed over parameters and latent variables, leading to update equations for an EM algorithm. A ML estimate to the posterior distribution over a data set is output. However, the target number of components K and the number of factors in each component q have to be known in advance. In an unsupervised learning context this is seldom possible, so this problem is traditionally addressed with model selection criteria (eg. BIC [19], AIC [1] and variations thereof [26]). Models with various K are used, and the criterion helps decide which complexity should be retained. However, this approach suffers with several limitations :

- EM returns a distribution associated to a local maximum of the likelihood function. This maximum may be associated with strongly overlapping components, which is undesirable in certain contexts, even if some selection criteria take this property into account [26].
- The likelihood function for the mixture case is not bounded, especially when considering a component being fitted on a single point (i.e. precision then tends to infinity).
- Learning has to be performed numerous times to build a set of models from which the best is picked, according to the criterion. In distributed or online contexts, this may not be appropriate.

Closely related to our proposal, a variational Bayes scheme was proposed for the single component PPCA [7] and the mixture of Factor Analysers [5], [13]. Factor Analysers (FA) and PPCA approaches differ only by the noise model involved ; from (4), it is chosen isotropic for PPCA, as for FA a diagonal covariance matrix is used. Consequently, the rotational invariance mentioned in section II-A does not hold any more for FA, replaced by invariance regarding individual rescaling of the input variables [23].

2) Variational approach: The design of variational algorithms enables them to overcome the ML defects indicated above. First, good probabilistic priors have to be chosen ; uninformative (i.e. not shading off the data we fit the model to), and yet reflecting desired properties for our target model (i.e. low complexity model, well separated components). The problem is then fully integrated from the Bayesian perspective, thus guaranteeing to obtain a local minimum that matches the desirable properties highlighted in [26]. A local maximum is still obtained, but w.r.t. to a better defined optimization ; this means we will always output an acceptable solution among a set of possible ”good” local maxima, lessening the necessity of multiple learnings.
We define the prior distributions set as :

$$p(\omega|\alpha_0) = \prod_k^K p(\omega_k|\alpha_0) = \prod_k^K \text{Dir}(\omega_k|\alpha_0) \tag{9}$$

$$\begin{aligned} p(\Lambda|\nu) &= \prod_k^K p(\Lambda_k|\nu_k) \\ &= \prod_k^K \prod_j^q p(\Lambda_k^{\cdot j}|\nu_{kj}) = \prod_k^K \prod_j^q \mathcal{N}(\Lambda_k^{\cdot j}|\mathbf{0}, \nu_{kj}^{-1} I_d) \end{aligned} \tag{10}$$

$$p(\nu|a_0, b_0) = \prod_k^K p(\nu_k|a_0, b_0) = \prod_k^K \prod_j^q \text{Ga}(\nu_{kj}|a_0, b_0) \tag{11}$$

$$p(\mu|\mu_0, \nu_0) = \prod_k^K p(\mu_k|\mu_{0k}, \nu_0) = \prod_k^K \mathcal{N}(\mu_k|\mu_{0k}, \nu_0^{-1} I_d) \tag{12}$$

Where $\text{Dir}(.|\alpha)$ denotes the Dirichlet distribution parametrized by α, ν_{kj} are precision parameters for the column factors, and $\text{Ga}(.|a, b)$ denotes the Gamma distribution parametrized by a and b which are respectively the scale and inverse shape of the distribution. Let us give further details about the parametrization that may be used in practice :

- α_0 is set to 10^{-3} to favor the minimal number of components effectively used,
- a_0 and b_0 are set to 10^{-3} to enforce that every factor column is a priori as important as any other,
- μ_{0k} are uniformly randomly chosen in the data space domain, and ν_0 is set to 10^{-3}. Doing so allows proper data space exploration, while not biasing each component significantly.

We did not define a prior over the parameter τ (see eqn. (5) e.g.). This value reflects the lowest eigenvalues of the sample covariance matrix [23]. This information being stable, we chose not to optimize this variable, leaving it as a parameter. Moreover this would have involved significant computational overhead, and experimental results happened to be more stable and accurate with τ statically set to some arbitrary value, 1 in our experiments.

Using our set of priors, we define a Bayesian formulation of the problem :

$$p(\mathbf{y}) = \int p(\theta) p(\mathbf{y}|\theta) d\theta \tag{13}$$

$$\text{with } p(\theta) = p(\omega|\alpha_0) p(\Lambda|\nu) p(\nu|a_0, b_0) p(\mu|\mu_0, \nu_0)$$

In this paper our purpose is to find the only one true posterior distribution $p(\theta|\mathbf{y})$ associated with this integrand. This solution embodies the idea of a trade-off between prior information

and data likelihood. But practically its exact determination is intractable, and we have to resort to some approximation scheme.

Let us take the log and expand (13) :

$$\begin{aligned}
\mathcal{L} &= \ln p(\mathbf{y}) \\
&= \ln\Bigg(\int p(\omega|\alpha_0)d\omega \int p(\nu|a_0,b_0)d\nu \\
&\cdot \int p(\Lambda|\nu)d\Lambda \int p(\mu|\mu_0,\nu_0)d\mu \\
&\cdot \prod_n^N \Bigg[\sum_k^K p(z_{nk}=1|\omega) \int p(x_n)p(y_n|z_{nk}=1,x_n,\Lambda_k,\mu_k)dx_n \Bigg] \Bigg)
\end{aligned} \tag{14}$$

$p(\mathbf{y}|\theta)$ has been rewritten in order to make the dependence on $\mathbf{x}$ explicit. $p(x_{nk})$ is as given in (3), and we used (8) along with noticing that $p(z_{nk}=1|\omega)=\omega_k$.

We consider a set of arbitrary distributions (often referred to as *variational* distributions) $q(\omega,\mu,\Lambda,\nu)$ and $\prod_n^N q(x_n,z_n)$, and, with Jensen's inequality applied to probabilistic measures, use them to lower bound the expression (14). $q(\omega,\mu,\Lambda,\nu)$ is assumed to be factorized, i.e. :

$$\begin{aligned}
q(\theta) = q(\omega,\mu,\Lambda,\nu) &= q(\omega)\prod_k^K q(\mu_k)q(\Lambda_k|\nu_k)q(\nu_k) \\
&= q(\omega|\alpha)\prod_k^K q(\mu_k)\Bigg[q(\Lambda_k|\nu_k)\prod_j^q q(\nu_{kj}|a_{kj},b_{kj}) \Bigg]
\end{aligned} \tag{15}$$

Where $\alpha=\{\alpha_k\}$ and $\{a_{kj},b_{kj}\}$ are variational parameters. Under this assumed factorization, we can write the lower bound as follows :

$$
\begin{aligned}
\mathcal{L} \geq & \int q(\omega) \ln \frac{p(\omega|\alpha_0)}{q(\omega)} d\omega \\
& + \sum_k^K \int q(\nu_k) \left[\ln \frac{p(\nu_k|a_0, b_0)}{q(\nu_k)} + \int q(\Lambda_k) \ln \frac{p(\Lambda_k|\nu_k)}{q(\Lambda_k)} d\Lambda_k \right] d\nu_k \\
& + \sum_k^K \int q(\mu_k) \ln \frac{p(\mu_k|\mu_{0k}, \nu_0)}{q(\mu_k)} d\mu_k \\
& + \sum_n^N \sum_k^K q(z_{nk} = 1) \Bigg[\int q(\omega) \ln \frac{p(z_{nk} = 1|\omega)}{q(z_{nk} = 1)} d\omega \\
& + \int q(x_n|z_{nk} = 1) \ln \frac{p(x_n)}{q(x_n|z_{nk} = 1)} dx_n \\
& + \int q(\Lambda_k) q(\mu_k) d\Lambda_k d\mu_k \\
& \int q(x_n|z_{nk} = 1) \ln p(y_n|z_{nk} =, x_n, \Lambda_k, \mu_k) dx_n \Bigg] = \mathcal{F}
\end{aligned} \tag{16}
$$

It is possible to show that maximizing $\mathcal{F}$ is equivalent to minimizing KL$[q(\theta) \parallel p(\theta|\mathbf{y})]$ (see [3], [8] chapter 11 for proof). KL$[q||p]$ denotes the Kullback-Leibler divergence of p w.r.t. q. Thus, when $\mathcal{F}$ is maximized, our variational distribution will equal the true, intractable, posterior we try to determine. On the one hand, as we assumed a factorization for $q(\theta)$, this maximization may not lead to a tight bound ($\mathcal{F} \leq \mathcal{L}$). But on the other hand, 1) now functional derivatives can be performed w.r.t. individual terms in (15), forming a set of update equations, and 2) optimizing a lower bound to a constant unknown value means we do not optimize an unbounded function as is done with the traditional EM algorithm (see paragraph II-B.1). Iterating our update equations set implements a pseudo-EM algorithm (E and M steps being respectively about updating the variational posterior on latent variables, and parameters ; this general scheme is sometimes referred to as Iterative Variational Bayes [20] in the literature ; for the present specific setting, it will be further denoted as VBMPPCA), and the produced variational distribution will approximate the true, unknown, posterior distribution. The update equations are obtained using standard functional calculus over (16), and are given in appendix I-B. Detailed derivations may be found in [5]. Each iteration should increase $\mathcal{F}$, thus convergence is easily assessed by iterating until the computed value for $\mathcal{F}$ has plateaued. Let us recall that, while standard EM algorithms optimize a lower bound w.r.t. a point estimate, variational algorithms optimize w.r.t. a distribution function.

Also, through the optimization, functional forms of variational posteriors and their respective priors happen to be identical. These are general properties for the variational methods involving distribution functions from the exponential family [3].

3) Discussion on parametrization: Values for prior parameters were already discussed in the previous section, and these can serve as initial values for the variational parameters. Earlier we mentioned that using $\alpha_0 = 10^{-3}$ favors the smallest sensible number of components ; indeed, the modes of the associated Dirichlet distribution are thus located at ω values where some of the ω_k are 0. The variational parameter α_k may then become significantly different from 0 only if its component reflects the density of the data set ; components for which $\alpha_k \backsimeq \alpha_0$ can be pruned. Algorithm 1 below may be derived from these observations. A prior component set that covers as much as possible the data space is used. Factor matrices are initialized with random orthogonal matrices, so as to reflect the fact that principal subspaces are made of orthogonal vectors.

Data: A Data Set, An upper bound K for the number of significant components
Result: The set $\{\theta'\}$ of significant components
Define $\{\theta\}$ the set of K prior components ;
while *not convergence* **do**
    update latent variables $\mathbf{x}$ and $\mathbf{z}$;
    update sufficient statistics ;
    update $\{\theta\}$;
end
$\{\theta'\} \leftarrow$ empty set ;
for *each component* θ_k *in* $\{\theta\}$ **do**
    if $\alpha_k > \alpha_0$ **then**
        add θ_k to $\{\theta'\}$;
    end
end

Algorithm 1: VBMPPCA designed to capture the most sensible number of components

This algorithm needs to be fed with an upper bound to the sensible number of significant components. For real world problems, we often have little information about this value. One of the following strategies may be employed :

- Set K to a very high value. This guarantees a good solution, but at high computational cost.
- Set K to a very high value, and prune components at each iteration (i.e. incorporate a pruning step after line 5 in algorithm 1). Computational cost will be high only for the first iterations.
- Adopt an elaborate birth and death strategy, as carried out in [5].

The prior introduced by equations (10) and (11) implements the idea of Automatic Relevance Determination (ARD) [16] ; a precision parameter is associated with each column vector of the factor matrix, and when a column may play no role (i.e. if the data its factor matrix supports has a principal subspace already fit by a complementary set of other columns), the associated variational mode will be driven to high values, while supported columns while have low values. Thus using this principle, the variational approximation to Bayesian inference embodies our preference towards low dimensional subspaces, and an objective criterion for selecting the latent dimensionality of each component.

Yet, an initial value for q has to be chosen ; it is constrained to being less than d, so the naive choice would be to set $q = d - 1$. But update equations complexity scales with q^2 ; so for high d this choice would not be recommended.

After optimization, each factor may be re-arranged by post-multiplying with the rotation matrix (see end of paragraph II-A). After some rank q', the ARD parameter associated with the respective column will potentially exceed some threshold, and so will those of the columns of superior index. Thus we may output Λ as the $d \times q'$ sub-matrix, whose columns have a low ARD precision posterior expected value. This threshold will be scale dependent, so we can employ a more elaborate scheme. For all significant components and, as factors are ordered by decreasing magnitude, we build equivalent covariances with an increasing number of factors using the formula for C in section II-A. We measure then the variation in terms of KL or Jensen-Shannon (JS) divergence. We may then determine the appropriate local dimensionality when this variation falls below a threshold. As divergences are scale independent, this threshold can then be used for any data set.

4) Convergence and bias issues: In the Iterative Variational Bayes scheme, each parameter update equation relies on moments evaluated with respect to other parameters and latent variables. For the Gaussian mixture implementation (see [3] and [8], chap. 10. This algorithm will be

further denoted as VBGMM), variational updates in the M step rely solely on sufficient statistics computed over latent variables moments and input data. This has the following consequences :

- the ordering chosen within the parameter set is not constrained ; to complete the M step, each variational parameter has to be updated once;
- in other words, needed moments do not evolve during a single step.

Inspection of the VBMPPCA update equations (see appendix I-B) leads us to contrasting conclusions :

- within the E step, a sequence is naturally set by remarking $\mathbf{x}$ variables do not depend on $\mathbf{z}$, whereas the contrary is true. Thus updating $\mathbf{x}$ and then $\mathbf{z}$ provides us with consistent estimators. These are used to build sufficient statistics that summarize the influence of latent variables and input data for the subsequent M step.
- M step is much more problematic. We see couples of mutually dependent parameters sets (see fig. 1). This means that no natural sequence may be set ; for example, if μ is updated first, and then is Λ, the subsequent E step may not use a consistent estimator for μ, as its moment is implicitly changed when Λ update is performed.

From a statistical point of view, this means we use biased estimators. Due to the lack of a natural sequence highlighted above, no systematic procedure exists to remove this bias. Properties of variational methods hold (i.e. $\mathcal{F}$ is strictly monotonic and bounded), but in some cases the posterior model distribution may reflect a poor local minimum. This problem has already been briefly stated by Beal [5], suggesting that performing repetitive updates of $\mathbf{z}$, μ, $\mathbf{x}$ and Λ, before updating ω and ν, within a single iteration, may more efficiently increase $\mathcal{F}$. We chose a slightly different approach, which leads to algorithm 2. Briefly stated, we used repetitive updates (from 3 to 5 is a good compromise), but for μ, Λ and $\mathbf{x}$. This new version breaks the usual EM scheme, and burdens each iteration with repeated operations scaling with data size ($\mathbf{x}$), but increases $\mathcal{F}$ much more efficiently, and towards higher values on the likelihood surface. This adaptation can be seen as an empirical way of stabilizing our estimators before the next iteration.

The observed dependencies among posteriors over parameters emerge from the chosen formalism, i.e. the independency assumption between (μ) and (Λ, ν). This choice is motivated in the literature by the introduction of ARD priors as a key feature for automatic dimensionality reduction. A more sophisticated model, which would not require the above mentioned assump-

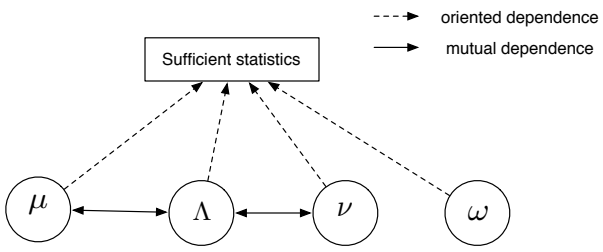

Fig. 1. Moment dependencies between variational parameters. Latent variables are collectively denoted by *sufficient statistics*.

tion, may theoretically be derived. However, this would lead to a highly cluttered formalism; this property led practitioners to the simpler, present solution.

Data: A Data Set, An upper bound K for the number of significant components
Result: The set $\{\theta'\}$ of significant components
Define $\{\theta\}$ the set of K prior components ;
while *not convergence* **do**
    update $\mathbf{z}$;
    update α ;
    perform μ, Λ and $\mathbf{x}$ updates 5 times ;
    update ν ;
end
$\{\theta'\} \leftarrow$ empty set ;
for *each component* θ_k *in* $\{\theta\}$ **do**
    if $\alpha_k > \alpha_0$ **then**
        add θ_k to $\{\theta'\}$;
    end
end

Algorithm 2: Empirically effective VBMPPCA

5) Properties of VBMPPCA vs. VBGMM: VBGMM already addresses the case of highly correlated data, so when using $q = d - 1$ we generally obtain similar solutions for VBGMM and VBMPPCA. The complexity is almost the same for the 2 algorithms (i.e. asymptotically linear w.r.t the number of input items), but for the PPCA mixture each iteration is burdened with the update of the $\mathbf{x}$ latent variables. Thus, with additional regard to the initialization and

convergence issues highlighted above, it might seem advantageous to fit a Gaussian mixture, and eigen-decompose each component afterwards.

However, for very high dimensional data where we strongly expect the local dimensionalities to be much lower than d, we may initialize q with some upper bound to these dimensionalities, so that $q < d - 1$. The gain is quadratic w.r.t the difference between q and $d - 1$. When the hypothesis regarding the lower dimensional manifolds roughly holds, we may then obtain valid models with much lower computational and memory requirements.

III. Aggregating MPPCA models from their parameters

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. Then, we incorporate this representation in the algorithm proposed in section II-B in replacement of an ordinary data set. 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. Virtual sample likelihood

A complete sample (i.e. data and labels) originating from an arbitrary input PPCA mixture with L components may be denoted as $(\mathbf{y}, \mathbf{z}) = \{y_i, z_i\}$, with the usual 1-of-L notation for z_i (see paragraph II-B.1 for definition). Data items in this set may be regrouped according to values taken by z_i : let us note $\hat{y}_l = \{y_i | z_{il} = 1\}$ and $\hat{z}_l = \{z_i | z_{il} = 1\}$. We will define the likelihood of this data set w.r.t a new, unknown mixture of PPCA, and propose an appropriate learning algorithm. This model will be further denoted as target, or output, model. We avoid further ambiguities by defining some indexing conventions and shorthand notations : elements of the input and of the output mixtures will be respectively indexed by $l \in \{1 \dots L\}$ and $k \in \{1 \dots K\}$. The whole output model will be collectively identified by θ, and its full weights set by ω. Also, in order to keep notations as uncluttered as possible, and without loss of generality, we will assume that all input PPCA components have the same number of factors, q. As the output

model should not be more complex than the input, this same value may be used to parametrize the output factor matrices. The likelihood of our complete sample is defined as :

$$\mathcal{L}(\mathbf{y}|\theta) = \mathcal{L}(\mathbf{y}|\mathbf{z},\theta)\mathcal{L}(\mathbf{z}|\theta) \tag{17}$$

$$\text{with } \mathcal{L}(\mathbf{y}|\mathbf{z},\theta) = \prod_k \prod_l (\mathcal{N}(\hat{y}_l|\mu_k, C_k))^{z_{lk}} \tag{18}$$

$$\text{and } \mathcal{L}(\mathbf{z}|\theta) = \prod_k \prod_l (\omega_k^{|\hat{z}_l|})^{z_{lk}} \tag{19}$$

where we transferred the formulation in eqn. (6) to an i.i.d data set. $\mathbf{y}$, the latent PPCA variable $\mathbf{x}$ being intentionally marginalized (see eqn. (5)) at this point. This set of formulas is valid under the assumption that z_{lk} is common for all y_i in $\hat{y}_l$. In most cases it will hold, as aggregating models is mostly about regrouping components.

Let us notice that

$$|\hat{z}_l| = |\hat{y}_l| \simeq N\omega_l \tag{20}$$

where N denotes the size of the full sample discussed above. By incorporating this approximation, (19) now depends on the input data set only through model parameters and the data set size.

(18) may be written as follows :

$$\mathcal{L}(\mathbf{y}|\mathbf{z},\theta) = \prod_k \prod_l (\mathcal{N}(\hat{y}_l|\mu_k, C_k))^{z_{lk}} = \prod_k \prod_l \mathcal{L}_{lk}^{z_{lk}} \tag{21}$$

Now we consider a single $\mathcal{L}_{lk}$ term. We first take its log, expand $\hat{y}_l$ and use the approximation (20) :

$$\ln \mathcal{L}_{lk} = \sum_j^{N\omega_l} \ln \mathcal{N}(y_{lj}|\omega_k, \mu_k, C_k) \tag{22}$$

$$\simeq N\omega_l \left[-\mathrm{KL}(\mathcal{N}(\mu_l, C_l) \parallel \mathcal{N}(\mu_k, C_k) - \mathrm{H}(\mathcal{N}(\mu_l, C_l)\right] \tag{23}$$

where we used the law of large numbers to approximate KL divergence and entropy integrals by a finite sum. Thus N should be sufficiently large to ensure validity of expressions (20) and (23).

Let us highlight a remarkable fact : our approximated likelihood expressions do not depend on the complete input data set any more, but solely on the parameters of the model originating it. Therefore, a large value can be arbitrarily chosen for N. This trick was presented as *virtual sampling* in [24], and used in [10] to perform the aggregation of Gaussian mixtures.

KL divergence between Gaussians, and entropy for a Gaussian distribution, have closed form expressions, enabling rewriting (23) as :

$$\begin{aligned}\ln \mathcal{L}_{lk} = N\omega_l[&-\frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln\det(\Lambda_k\Lambda_k^T + \tau_k^{-1}I_d)\\ &-\frac{1}{2}\mathrm{Tr}((\Lambda_k\Lambda_k^T + \tau_k^{-1}I_d)^{-1}\\ &[\Lambda_l\Lambda_l^T + \tau_l^{-1} + (\mu_l-\mu_k)(\mu_l-\mu_k)^T)])]\end{aligned}$$

In the remainder of the paper, we discard the influence of τ_l^{-1}, as the ML value of this term embodies the smallest eigenvalues of the respective components. Since $\Lambda_l\Lambda_l^T = \sum_j \Lambda_l^{:j}\Lambda_l^{:jT}$, we may describe $\ln \mathcal{L}_{lk}$ as the combined likelihood of the means and factors of our input components (after renormalization, and with respective means μ_k and 0).

$$\mathcal{L}_{lk} = \left[\mathcal{N}(\mu_l|\mu_k, C_k)\prod_j \mathcal{N}(\Lambda_l^{:j}|0, C_k)\right]^{N\omega_l} \tag{24}$$

Using (24) along with (21) and (19) gives as an approximation for the complete likelihood of our virtual sample :

$$\begin{aligned}\mathcal{L}(\mathbf{y}|\theta) = \prod_l\prod_k (\omega_k^{N\omega_l})^{z_{lk}} \\ \left[\left(\mathcal{N}(\mu_l|\mu_k, C_k)\prod_j \mathcal{N}(\Lambda_l^{:j}|0, C_k)\right)^{N\omega_l}\right]^{z_{lk}}\end{aligned} \tag{25}$$

In (25) all the terms belong to the exponential family. Taken to an exponent, such a distribution is still in the exponential family, so $N\omega_l$ may be incorporated in the parametrizations, giving :

$$\mathcal{L}(\mathbf{y}|\theta) = \prod_l \sum_k \left(p(z_{lk}=1|\omega^{N\omega_l})p(\mu_l)\prod_j p(\Lambda_l^{:j})\right) \tag{26}$$

In order to keep the notations uncluttered in (26), we used :

$$p(z_{lk} = 1|\omega^{N\omega_l}) = \omega_k^{N\omega_l} \tag{27}$$

$$p(\mu_l) = \mathcal{N}(\mu_l|\mu_k, (N\omega_l)^{-1}C_k) \tag{28}$$

$$p(\Lambda_l) = \mathcal{N}(\Lambda_l^j|0, (N\omega_l)^{-1}C_k) \tag{29}$$

The Gaussian terms in our likelihood still depend on C_k, so these may be expanded with $\mathbf{x}$ variables using linear Gaussian models properties. In the classical scheme [5], [22], there is a single variable x per item. Now, each input component is associated with $1+q$ items, so $\mathbf{x}$ scales accordingly. To avoid ambiguities among latent variables, we define $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$, $\mathbf{x}_1 = \{x_{1l}\}$ and $\mathbf{x}_2 = \{x_{2lj}|j \in 1 \dots q\}$. The complete likelihood is then defined as :

$$\begin{aligned}
\mathcal{L}(\mathbf{y}|\theta) = \prod_l^L \sum_k^K p(z_{lk} = 1|\omega^{N\omega_l}). \\
\int dx_{1l} p(x_{1l}) p(\mu_l|x_{1l}). \\
\prod_j^q \int dx_{2lj} p(x_{2lj}) p(\Lambda_l^j|x_{2lj}) \\
\text{with } p(x_{.l.}) = \mathcal{N}(x_{.l.}|0, (N\omega_l)^{-1}I_q) \\
p(\mu_l|x_{1l}) = \mathcal{N}(\mu_l|\Lambda_k x_{1l} + \mu_k, (N\omega_l\tau_k)^{-1}I_d) \\
p(\Lambda_l^j|x_{2lj}) = \mathcal{N}(\Lambda_l^j|\Lambda_k x_{2lj}, (N\omega_l\tau_k)^{-1}I_d)
\end{aligned} \tag{30}$$

B. Lower bound derivation

The class of algorithms presented in this paper optimize a lower bound to the marginal likelihood, obtaining an estimate to the posterior MPPCA model as a result. We also defined a set of variational distributions in eqn. (15). This set is augmented with terms associated to the supplementary latent variables introduced in eqn. (30). Consequently the lower bound derivation given in eqn. (16) is slightly modified. The structure of the output model, a mixture of PPCA, is unchanged, then so are the first three lines. The last four lines are specifically associated to the data likelihood, so expression (30) can be used instead of standard data likelihood.

$$
\begin{aligned}
\mathcal{F} &= \int q(\omega) \ln \frac{p(\omega|\alpha_0)}{q(\omega)} d\omega \\
&+ \sum_k^K \int q(\nu_k) \left[\ln \frac{p(\nu_k|a_0, b_0)}{q(\nu_k)} + \int q(\Lambda_k) \ln \frac{p(\Lambda_k|\nu_k)}{q(\Lambda_k)} d\Lambda_k \right] d\nu_k \\
&+ \sum_k^K \int q(\mu_k) \ln \frac{p(\mu_k|\mu_{0k}, \nu_0)}{q(\mu_k)} d\mu_k \\
&+ \sum_l^L \sum_k^K q(z_{lk} = 1) \Bigg[\int q(\omega) \ln \frac{p(z_{lk} = 1|\omega^{N\omega_l})}{q(z_{lk} = 1)} d\omega \\
&+ \int q(x_{1l}|z_{lk} = 1) \ln \frac{p(x_{1l})}{q(x_{1l}|z_{lk} = 1)} dx_{1l} \\
&+ \sum_j \int q(x_{2lj}|z_{lk} = 1) \ln \frac{p(x_{2lj})}{q(x_{2lj}|z_{lk} = 1)} dx_{2lj} \\
&+ \int q(\Lambda_k) q(\mu_k) d\Lambda_k d\mu_k \\
&\quad \cdot \Bigg(\int q(x_{1l}|z_{lk} = 1) \ln p(\mu_l|\Lambda_k x_{1l} + \mu_k) dx_{1l} \qquad (31) \\
&+ \sum_j \int q(x_{2lj}|z_{lk} = 1) \ln p(\Lambda_l^j|\Lambda_k x_{2lj}) dx_{2lj} \Bigg) \Bigg] \qquad (32)
\end{aligned}
$$

Functional calculus may be again employed, with no greater difficulty than in section II-B.2, to perform derivatives, leading to coupled update equations, with the same theoretic properties. This new set of updates is given in appendix I-C. Thus we define algorithm VBMPPCA-A as an extension of algorithm 2, designed to obtain our posterior output mixture.

C. Parametrization

In section II-B, we mentioned the usage of uninformative priors. These are still used here, but we may also jointly exploit some strong prior knowledge. Indeed in paragraph II-A we noticed that the standard estimation procedure was able to recover the scaled eigenvectors ordered by decreasing magnitude in the columns of Λ_{ML}. We also remark that the additional latent variables are associated with the columns of the Λ input matrices. Under the assumption of appropriately ordered input Λ, intuitively we would associate the first column of the input Λ to the first column of the output Λ and so on. As x variables denote the combination of columns of Λ, we therefore choose to initialize $\mathbf{x}_2$ estimates to canonical vectors, so as to reflect this belief. Experimentally

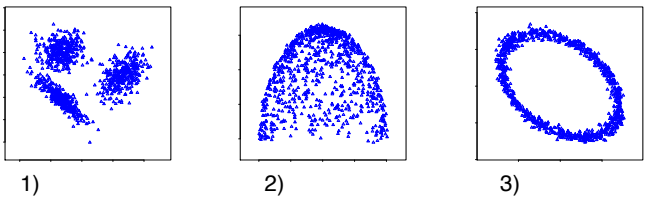

Fig. 2. Synthetic data sets representations (2D projections). 1) Gaussian 2) semisphere 3) circle

this principle was found to improve the results very significantly. The $\mathbf{x}_2$ variables updates burden our E step, but but this is largely outweighed by noticing that the complexity of our algorithm now scales with the number of input components instead of the size of a data set.

At the beginning of the section III-A, we stated the same q might be used without loss of generality. To see this, let us consider a set of input factor matrices, with q_i the subspace dimensionalities attached to them. We define $q_{\max} = \max q_i$. We see that completing each factor matrix with $q_{\max} - q_i$ void columns, and setting the associated $x_2..$ variable to 0 instead of a canonical vector reduces to consider a single q for all input factors.

IV. Experimental results

A. Data sets

We report results on the following data sets :

- *Gaussian* : this synthetic data was generated by sampling from 3 well separated 3D Gaussians (see figure 2-1). 6 additional dimensions were then built by random linear combinations of the original 3D signal. 2000 points were sampled from each Gaussian.
- *semisphere* : this synthetic data set was obtained by sampling random angles from a 3D semi-sphere (see figure 2-2). 6000 points were generated.
- *circle* : 6000 points were generated along a 2D circle with additive noise (see figure 2-3). This original 2D signal was then linearly transformed to 6 dimensional using a random orthogonal matrix, with, again, some additive noise.
- *Pen-based recognition of handwritten digits* (further denoted as *Pen data*) : 10992 16-dimensional data points built from the positions taken on a tablet to draw numerical digits [2]. Each point is labeled with the true digit that was drawn (0-9, thus defining 10 classes).

B. Estimating probability density models

We assess the ability of our method for building density models from distributed data sources, which supply only model parameters. To this aim, the 3 above synthetic data sets are used according to the following protocol :

- input models are built using VBMPPCA on random subsamples of the input data. All our synthetic data sets are made of 6000 points, from which we extract 200 points subsamples. 500 models are built using this procedure, forming a pool of inputs. For comparison sake, we also estimate VBGMM models on the same subsamples.
- we apply VBMPPCA-A to aggregate a varying number n of input MPPCA models, randomly chosen in our pool. For each n value, the experiment is performed 20 times, and results are averaged. The Gaussian mixture aggregation scheme presented in [10] (and further denoted as VBGMM-A) supplies a benchmark result. For the comparison to be fair, selected input MPPCA models are converted to Gaussian mixtures using formula (5) in order to feed this reference procedure.
- the ground truth density model is fitted using VBGMM over the whole data set.

The following characteristics are measured :

- the estimated complexities for our input models (i.e. number of components and number of factors per component). This evaluates to which extent VBMPPCA is able to automatically discover the number of components and factors. As a comparison, the number of components discovered by fitting a Gaussian mixture is also given.
- the Jensen-Shannon (JS) divergence between the aggregated mixtures and the sum of the input mixtures, depending on n. Jensen-Shannon divergence is a symmetrized version of KL divergence. This will assess the quality of our aggregation.
- the JS divergence between the aggregated mixtures and the ground truth density model, depending on n.
- the average number of components in the output, depending on n.

Table I summarizes the respective behaviors of VBGMM and VBMPPCA. On average, it can be seen that VBMPPCA tends to produce richer models. But this is not an explicit flaw ; the number of components for the *Gaussian* data set can be seen as over-estimated, whereas 7 components seem more able to fit *circle*. Estimated subspace dimensionalities are, on average,

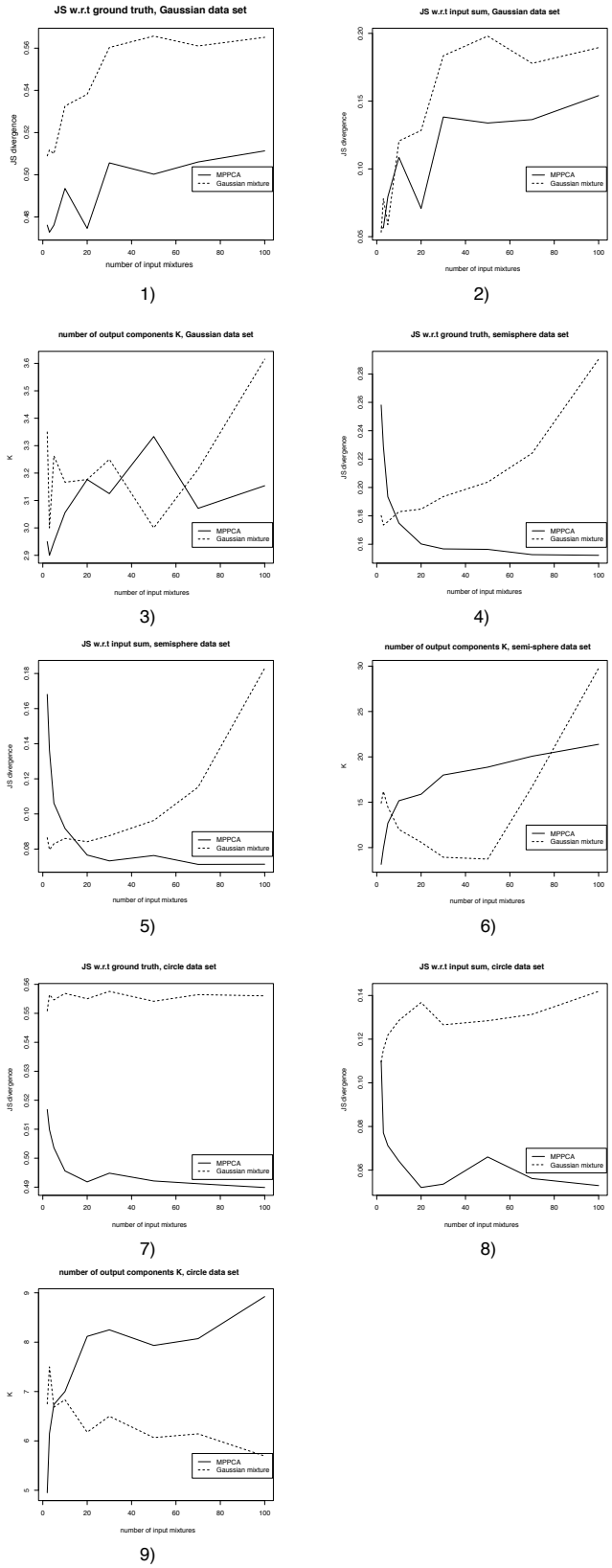

Fig. 3. Aggregation quality evaluation results for the 3 synthetic data sets.

	K (MPPCA)	K (Gaussian mixture)	q
Gaussian	3.3	2.98	2.71
semi-sphere	13.8	11.16	1.64
circle	7.43	4.27	1.25

TABLE I

COMPLEXITIES (NUMBER OF GROUPS K, AND COMPONENT SUBSPACE DIMENSIONALITY q) FOR THE INPUT MPPCA MODELS USED IN OUR PROTOCOL. USED SUBSAMPLES ARE ALSO FITTED WITH GAUSSIAN MIXTURES FOR COMPARISON WITH MPPCA.

conform to the original signals ; 3D Gaussians were used, and in table I $q_{\text{output}} = 2.71$. *Semi-sphere* and *circle* clearly expose 1D to 2D manifolds ; this property is empirically recovered (q_{output} respectively equals 1.64 and 1.25 for these data sets).

On figures 3-(2,5,8), we see that the KL (or, equivalently, JS) loss w.r.t. the input models weighted sum is, on average, higher when using VBGMM-A instead of VBMPPCA-A. The only exception occurs when aggregating a very small number of sources for the *semi-sphere* data set (fig. 3-5). Similar facts can be seen when considering ground truth models as a comparison point (figs 3-(1,4,7)). Variational methods optimization is based on a KL divergence minimization (see paragraph II-B.2). In that sense, VBMPPCA-A leads to more accurate estimates. The output number of components seems data-dependent ; with VBMPPCA-A, it is higher for *circle* (fig. 3-9) , and lower only when considering small or large inputs for *semi-sphere* (fig. 3-6) and *Gaussian* (fig. 3-3). This illustrates significantly different behaviors, even if no inconsistency can be highlighted here, as values for both techniques always stay pretty much in the same range.

C. Clustering

Here we evaluate our method on a real data set. Similarly to the previous paragraph, we will measure the quality of our density models w.r.t some standard, but we will also confront them to a ground truth labeling. Mixture models are often employed to cluster a data set ; each data item will have its label inferred using Bayes decision rule.

A n-element data set, when partitioned (or clustered) in k groups, defines a set of n labels,

	K (MPPCA)	K (Gaussian mixture)	q
Pen data	15.97	23.95	6.35
	error (%) (MPPCA)	error (%) (Gaussian mixture)	
Pen data	12.3	9.8	

TABLE II

COMPLEXITIES (NUMBER OF GROUPS K, AND COMPONENT SUBSPACE DIMENSIONALITY q) AND ERRORS FOR THE INPUT MPPCA MODELS USED IN OUR PROTOCOL. USED SUBSAMPLES ARE ALSO FITTED WITH GAUSSIAN MIXTURES FOR COMPARISON WITH MPPCA.

each label taking values in $\{1 \dots k\}$. We will measure the quality of a label set (or equivalently its error w.r.t the true label set) by comparing the inferred labels I_i to the true labels T_i ; the error would increase only if $I_i \neq T_i$.

However this method can be inconsistent ; clustering is unsupervised, so any permutation between label modalities should be equally valid. Furthermore, measuring strict inequalities between labels is problematic when the true k and the inferred k are not the same, which may often happen with real world data.

This motivates the usage of another error measure, suggested in [17], [12]. This relies on processing the labels taken by each possible couple of data items ; two items should have the same inferred labels only if the true ones are identical, and symmetrically. Violations would increase the error measure. This error measure is normalized by $\frac{n(n-1)}{2}$, the number of possible couples. This ensures it lies in $[0, 1]$.

To this end, the protocol of the previous paragraph is again employed, augmented with the suggested measure of error for the models involved. 300 input models were built with VBMPPCA, on larger subsamples (500 data points). Errors are measured w.r.t the full data set, even for input models fitted on partial data. Properties of input models are summarized in table II, and results for aggregation experiments are presented in figure 4. Let us recall that the data set used in this section is defined over 16 dimensions. To illustrate how computational burden can be limited with VBMPPCA, input models were fitted using $q = 8$, instead of the default setting, $q = d - 1$.

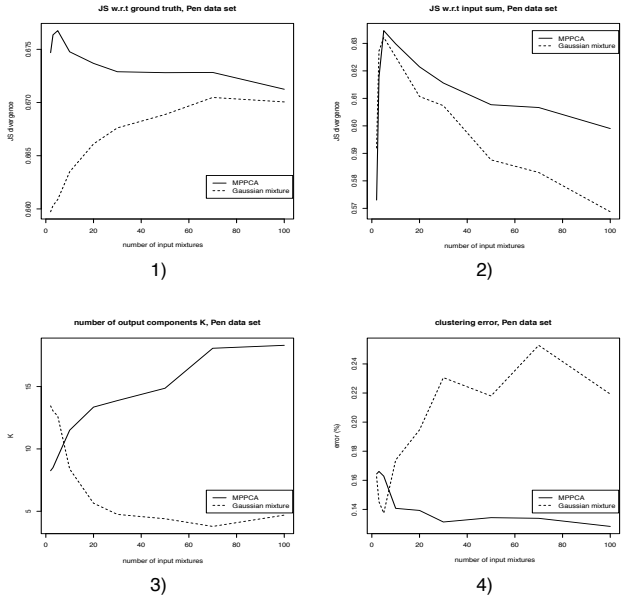

Fig. 4. Aggregation quality evaluation results for *Pen data*.

Using VBMPPCA over subsamples of this data set leads to simpler models on average. The estimated number of groups is closer to the ground truth (16 for MPPCA, 24 for Gaussian mixtures, vs. 10 real groups, see table II). We see that even if we used prior subspace dimensionalities smaller than the maximum that may be used, on average there is an supplementary dimensionality reduction ($q_{\text{output}} = 6.35$). The measured error is significantly higher when using VBMPPCA, but this may be due to the strong difference regarding the estimated number of groups. Indeed, augmenting the number of clusters mechanically reduces our clustering error measure.

JS divergences of aggregations carried out with VBMPPCA-A w.r.t the input models weighted sum tend to be higher, as may be noticed in fig. 4-2. The same fact holds when comparing to the ground truth density (see fig. 4-1). Thus VBMPPCA-A first appears to perform worse than VBGMM-A. This discrepancy may originate from the usage of limited prior subspace dimensionalities. However, this result has to be carefully interpreted ; this real data set is strongly non-Gaussian, which means numerous local optima models may be found for it. All will be equivalently good, i.e. their lower bounds will lie in the same range, but pairwise divergences will often be high.

Despite this apparent flaw, the estimated number of groups seems much more adequate with VBMPPCA-A in fig. 4-3; VBGMM-A tends to strongly underestimate the output number of components when the data sources become numerous, as this number happens to be much more stable when using VBMPPCA-A. Actually, MPPCA input models and aggregated MPPCA always have similar complexities, whereas using Gaussian mixtures tends to produce over-complex inputs and under-complex outputs. It may be seen as an effect of using combinations of subspaces to compensate the noise induced by input models fitted on partial data, and local minima problems highlighted above. Also, VBMPPCA-A produces models with much lower clustering error (see fig. 4-4). Even if this difference may partially be induced by model complexities, we globally see that VBMPPCA-A has better ability to preserve the information carried by input models, even when the subspace dimensionalities are constrained.

V. Conclusion and future works

In this paper we proposed a new technique that performs the aggregation of mixtures of PPCA. 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, with remarks given in paragraphs II-B.3, II-B.5 and III-C, 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 [4], 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 [11] for discussion), we may these as follows :

- *must-link* : for now, each factor matrix is initialized with a random orthogonal matrix (see

section II-B.3). 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 PPCA mixture scheme ; but we might exploit the fact we maintain principal subspace structures.

REFERENCES

[1] H. Akaike. A new look at the statistical model identification. *IEEE Trans. on Automatic Control*, AC-19(6), 1974.

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

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

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

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

[6] M. Bechchi, G. Raschia, and N. Mouaddib. Merging distributed database summaries. *Conference on Information and Knowledge Management*, pages 419–428, 2007.

[7] C. M. Bishop. Variational principal components. *Proceedings of 9th ICANN*, 1:509–514, 1999.

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

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

[10] 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.

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

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

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

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

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

[16] 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.

[17] 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.

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

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

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

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

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

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

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

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

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

APPENDIX I
UPDATE EQUATIONS

A. Notations used

$\langle X \rangle_{q(X)}$: expectation of X w.r.t. $q(X)$

M^{ij} : matrix element located at i^{th} row and j^{th} column

$M^{i\cdot}$, $M^{\cdot j}$: respectively i^{th} row and j^{th} column of M

v^i : vector v i^{th} element

I_d : d-dimensional identity matrix

diag(X) : diagonal matrix which diagonal is 1) X if X is a vector, 2) X repeated d times if X is a scalar, 3) the diagonal of X if X is a matrix

In general, we will use $\langle f(X) \rangle$ as a shorthand notation for $\langle f(X) \rangle_{q(X)}$, when $f(X)$ is a function of X and no other latent variable or parameter.

B. MPPCA update equations

1) E step:

$$\Sigma_{x_k} = (I_q + \tau\langle\Lambda_k^T\Lambda_k\rangle)^{-1} \tag{33}$$

$$\langle x_n\rangle_{q(x_n|z_{nk}=1)} = \langle x_{nk}\rangle = \tau\Sigma_{x_k}\langle\Lambda_k^T\rangle(y_n - \langle\mu_k\rangle) \tag{34}$$

$$\langle x_n x_n^T\rangle_{q(x_n|z_{nk}=1)} = \langle x_{nk}x_{nk}^T\rangle = \Sigma_{x_k} + \langle x_{nk}\rangle\langle x_{nk}\rangle^T \tag{35}$$

$$\begin{aligned} q(z_{nk}=1) \propto \exp\Bigg(&\psi(\alpha_k) - \psi\left(\sum_j \alpha_j\right) + \frac{1}{2}\ln|\Sigma_{x_k}| \\ &- \frac{1}{2}\langle x_{nk}x_{nk}^T\rangle - \frac{\tau}{2}\Big[\|y_n\|^2 + \langle\|\mu_k\|^2\rangle \\ &- 2(y_n - \langle\mu_k\rangle)^T\langle\Lambda_k\rangle\langle x_{nk}\rangle \\ &- 2y_n^T\langle\mu_k\rangle + \mathrm{Tr}(\langle\Lambda_k^T\Lambda_k\rangle\langle x_{nk}x_{nk}^T\rangle)\Big]\Bigg) \end{aligned} \tag{36}$$

NB : a term was missing in [5] for eqn. (36). The corrected version is given here.

2) Sufficient statistics:

$$N_k = \sum_n q(z_{nk}=1) \tag{37}$$

$$y_k = \sum_n q(z_{nk}=1)y_n \tag{38}$$

$$s_k = \sum_n q(z_{nk}=1)\langle x_{nk}\rangle \tag{39}$$

$$sy_k = \sum_n q(z_{nk}=1)y_n\langle x_{nk}\rangle^T \tag{40}$$

$$S_k = \sum_n q(z_{nk}=1)\langle x_{nk}x_{nk}^T\rangle \tag{41}$$

3) M step:

$$\alpha_k = \alpha_0 + N_k \tag{42}$$

$$a_{kj} = a_0 + \frac{d}{2} \tag{43}$$

$$b_{kj} = b_0 + \frac{1}{2}\sum_{i}^{d} \langle \Lambda_k^{ij2} \rangle \tag{44}$$

$$\Sigma_{\Lambda_k} = (\text{diag}(\langle \nu_k \rangle) + \tau S_k)^{-1} \tag{45}$$

$$\langle \Lambda_k^{i.} \rangle = \Sigma_{\Lambda_k} \tau \left[s y_k^{i.} - \langle \mu_k^i \rangle s_k \right] \tag{46}$$

$$\langle \Lambda_k^T \Lambda_k \rangle = \sum_{i}^{d} \langle \Lambda_k^{i.} \Lambda_k^{i.T} \rangle \tag{47}$$

$$\langle \Lambda_k^{ij2} \rangle = \langle \Lambda_k^{i.} \Lambda_k^{i.T} \rangle^j \tag{48}$$

$$\Sigma_{\mu_k} = \left[\text{diag}(\nu_0) + \tau N_k \right]^{-1} I_d \tag{49}$$

$$\langle \mu_k \rangle = \Sigma_{\mu_k} \left[\text{diag}(\nu_0 \mu_0) + \tau (y_n - \langle \Lambda_k \rangle s_k) \right] \tag{50}$$

C. MPPCA aggregation update equations

When absent, a parameter update should be taken identical as in section I-B.

1) E step:

$$\Sigma_{x_{kl}} = \left(N\omega_l[I_q + \tau\langle\Lambda_k^T\Lambda_k\rangle]\right)^{-1} \tag{51}$$

$$= (N\omega_l)^{-1}\Sigma_{x_k} \tag{52}$$

$$\langle x_{1lk}\rangle = \tau N\omega_l\Sigma_{x_{kl}}\langle\Lambda_k^T\rangle\Big(\mu_l - \langle\mu_k\rangle\Big) \tag{53}$$

$$\langle x_{2lkj}\rangle = \tau N\omega_l\Sigma_{x_{kl}}\langle\Lambda_k^T\rangle\Lambda_l^{\cdot j} \tag{54}$$

$$= \tau\Sigma_{x_k}\langle\Lambda_k^T\rangle\Big(\mu_l - \langle\mu_k\rangle\Big) \tag{55}$$

$$q(z_{lk}=1) \propto N\omega_l(\psi(\alpha_k) - \psi(\sum_j \alpha_j)) + \frac{1+q}{2}\ln|\Sigma_{x_k}| \tag{56}$$

$$- \frac{N\omega_l}{2}\mathrm{Tr}(\langle x_{1lk}x_{1lk}^T\rangle + \sum_j\langle x_{2lkj}x_{2lkj}\rangle) \tag{57}$$

$$- \frac{\tau N\omega_l}{2}\Bigg[\|\mu_l\|^2 + \sum_j\|\Lambda_l^{\cdot j}\|^2 + \langle\|\mu_k\|^2\rangle - 2\mu_l^T\langle\mu_k\rangle \tag{58}$$

$$- 2(\mu_l - \langle\mu_k\rangle)^T\langle\Lambda_k\rangle\langle x_{1lk}\rangle - 2\sum_j \Lambda_l^{\cdot jT}\langle\Lambda_k\rangle\langle x_{1lk}\rangle \tag{59}$$

$$+ \mathrm{Tr}\langle\Lambda_k^T\Lambda_k\rangle[\langle x_{1lk}x_{1lk}^T\rangle + \sum_j\langle x_{2lkj}x_{2lkj}^T\rangle]\Bigg] \tag{60}$$

2) Sufficient statistics:

$$N_k = \sum_l N\omega_l q(z_{lk}=1) \tag{61}$$

$$m_k = \sum_l N\omega_l q(z_{lk}=1)\mu_l \tag{62}$$

$$s_k = \sum_l N\omega_l q(z_{lk}=1)\langle x_{1lk}\rangle \tag{63}$$

$$sm_k = \sum_l N\omega_l q(z_{lk}=1)\Big(\mu_l\langle x_{1lk}\rangle^T + \sum_j \Lambda_l^{\cdot j}\langle x_{2lkj}\rangle^T\Big) \tag{64}$$

$$S_k = \sum_l N\omega_l q(z_{lk}=1)\Big(\langle x_{1lk}x_{1lk}^T\rangle + \sum_j\langle x_{2lkj}x_{2lkj}^T\rangle\Big) \tag{65}$$

3) M step: Using the new set of sufficient statistics, the M step updates are identical to those found in appendix I-B.3, the only modification being the substitution of sy_k by sm_k.