MLiT: Mixtures of Gaussians under Linear Transformations

Massimo Piccardi · Hatice Gunes · Ahmed Fawzi Otoom · Oscar Perez Concha

Received: date / Accepted: date

Abstract The curse of dimensionality hinders the effectiveness of density estimation in high dimensional spaces. Many techniques have been proposed in the past to discover embedded, locally-linear manifolds of lower dimensionality, including the mixture of Principal Component Analyzers, the mixture of Probabilistic Principal Component Analyzers and the mixture of Factor Analyzers. In this paper, we propose a novel mixture model for reducing dimensionality based on a linear transformation which is not restricted to be orthogonal nor aligned along the principal directions. For experimental validation, we have used the proposed model for classification of five "hard" data sets and compared its accuracy with that of other popular classifiers. The performance of the proposed method has outperformed that of the mixture of Probabilistic Principal Component Analyzers on four out of the five compared data sets with improvements ranging from 0.5% to 3.2%. Moreover, on all data sets, the accuracy achieved by the proposed method outperformed that of the Gaussian mixture model with improvements ranging from 0.2% to 3.4%.

Keywords Dimensionality reduction · regularized maximum-likelihood · mixture models · linear transformations · object classification.

This paper is dedicated to the enduring and inspiring memory of our friend and colleague Nicola Bruti-Liberati.

Massimo Piccardi · Hatice Gunes · Ahmed Fawzi Otoom · Oscar Perez Concha
School of Computing and Communications
Faculty of Engineering and IT
University of Technology, Sydney (UTS)
Sydney, Australia
E-mail: {massimo, haticeg, afaotoom, oscarpc}@it.uts.edu.au

1 Introduction

Recognition of visual objects into classes of interest is a long-explored area of computer vision and pattern recognition. Despite the enormous progress made to date, recognition accuracy is still unsatisfactory on certain "hard" data sets. One of the main causes of such an elusive accuracy lies in the high dimensionality of the feature sets used to represent the objects. Typical feature sets consist of several, heterogeneous features including intensity and gradient values, appearance histograms, number and position of edge segments, corners, scale-invariant features; and many others. Conversely, the number of pre-classified samples available for training the classifiers is typically limited, often in the order of tens or hundreds (manual segmentation and annotation are tedious and time consuming). It is such a combination of high feature space dimensionality and scarcity of training data that positions many recognition problems under the "curse of dimensionality" [1]. In order to mollify the curse, classification is often preceded by a dimensionality reduction step where the original features are selected or combined into a significantly smaller set. Possibly the most widely known technique, Principal Component Analysis (PCA) operates an orthogonal projection of the original data onto a space of lower dimensionality. This lower-dimensional space is spanned by the principal eigenvectors of the sample covariance typically learned over an unlabelled training set. Linear Discriminant Analysis (LDA), instead, is a discriminative technique using labelled training data to exploit the notion of between-class variance explicitly. Probabilistic PCA (PPCA) makes a significant step forward from standard PCA by providing a precise probabilistic model for the transformation [2, 3]. The model, showing partial analogies with that of Fac-

tor Analysis (FA) [4, 5], posits the existence of a latent linear space and the presence of a noise component on the sample values. The explicit probabilistic model of PPCA enables the measurement of probabilistic distances between data and classes and the use of Bayesian classification. Non-linear techniques for dimensionality reduction have also been proposed such as Kernel PCA (KPCA) [6] and Incremental KPCA (IKPCA) [7]. An appealing alternative to discovering non-linear manifolds in high-dimensional spaces is the use of mixture of locally-linear models. This idea has motivated the extension of the methods above to mixture models such as mixture of PCA [8], mixture of PPCA [9], mixture of FA [10]. A mixture of t-distribution subspaces was also proposed to increase robustness to outliers [11]. For all these mixture models, training is performed by Expectation-Maximization algorithms [12].

In this paper, we present a novel mixture model for dimensionality reduction inspired by an analogy with sensor fusion. Each component in the mixture consists of a linear transformation projecting the original data onto a subspace and a Gaussian distribution fitted on the projected data. For this reason and for immediacy, we have named the proposed method MLiT - mixture of Gaussians under Linear Transformations. Parameters of all the transformations and Gaussian distributions are learned by a specific Expectation-Maximization algorithm. Regularization of the maximum-likelihood solution is proposed in two alternative ways: a) by imposing a constraint on the transformation's parameters; b) by normalizing the transformation's parameters after each maximization step. Our method mostly resembles a mixture of PCA, with the main differences that our transformation is not restricted by orthogonality and that the densities in the projected spaces are learned jointly with the transformations in a maximum-likelihood setting. In this paper, the proposed technique is used to learn class-conditional likelihoods for maximum-likelihood classification of five "hard" data sets from the UCI repository and the authors' own.

The rest of the paper is organized as follows. Section 2 describes the main related techniques with their strengths and limitations. Section 3 presents the proposed method (MLiT), its regularization by a constraint and by normalization, and the initialization procedure. Section 4 describes the experiments conducted to evaluate MLiT over multiple data sets, comparing the results with state-of-the-art classifiers. Section 5 concludes the paper.

2 Related work

In this section, we contrast the proposed method with the main related techniques (PCA, PPCA and FA) in order to more clearly illustrate its rationale. From the comparison, it emerges that none of these techniques can be regarded as superior to the others irrespectively of the problem under consideration and therefore experimental validation will be addressed in the following sections.

PCA maps a y sample from a high-, P-dimensional space to a point $x = W^T(y - \bar{y})$ in a D-dimensional space, with D typically $<< P$. From x, an approximation of y is then obtained as $\tilde{y} = Wx + \bar{y}$, with the reconstruction error defined as $e = \tilde{y} - y$. The D-dimensional space can be conveniently represented embedded in the original, P-dimensional space, showing the D directions that are retained and the $(P - D)$ that are discarded. The parameters of PCA are the P x D transformation matrix, W, and offset $\bar{y}$. Both parameters are learned based on a given set of training data, $Y = \{y_i\}_{i=1..N}$: W is given by the D "largest eigenvectors" (the eigenvectors of the largest eigenvalues) of their sample covariance and $\bar{y}$ by their sample mean. This choice for W has the effect of minimizing the total squared reconstruction error over the training set. Correspondingly, it maximizes the sample covariance in x-space, hoping to retain information useful for later classification. Therefore, the training set is composed of unlabelled data from all classes.

PCA can also be used on class-labelled data to approximate density estimation and maximum-likelihood classification. In this case, a separate PCA model (W, μ) is learned from data of each class. At classification time, a sample is classified as belonging to the class whose model provides the least squared reconstruction error. Either way, PCA models cannot be learned with maximum likelihood or other, fuller Bayesian methods due to their incomplete probabilistic formulation.

Factor Analysis and Probabilistic PCA amend the limitations of PCA by proposing a full probabilistic model that can be trained with maximum likelihood. Both posit the existence of a latent, low-dimensional space where a point, x, is in correspondence with a y sample in the original space. The relationship between samples and latent points is given by:

$$y = Wx + \mu + \epsilon \tag{1}$$

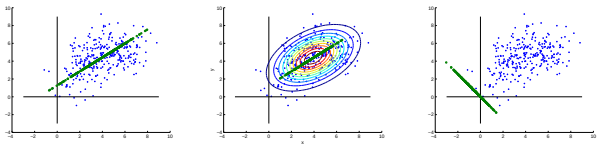
where W is a P x D matrix describing a linear transformation and ϵ is an additive noise component. Both x and ϵ are treated as random variables and assumed normally distributed, with $p(x) = \mathcal{N}(x|0, I)$ and

$p(\epsilon) = \mathcal{N}(\epsilon|0, \Psi)$, and independent. It follows immediately that $p(y) = \mathcal{N}(y|\mu, C)$, with $C = WW^T + \Psi$. In Factor Analysis, Ψ in (1) is assumed to be diagonal. Probabilistic PCA further restricts $p(\epsilon)$ to be spherical i.e. $\Psi = \sigma^2\mathbf{I}$ [2]. This restriction on the noise covariance is needed to uniquely separate the covariance attributed to the noise from that of the data [13]. In other terms, we could say that PPCA models the data along the retained directions through a full Gaussian model, whereas it models them along the discarded directions only through a Gaussian model with spherical covariance matrix. Figure 1(b) shows an example of a component of a PPCA mixture.

It may be interesting to note that in PPCA the maximum-likelihood solution for W is, like in PCA, based on the largest eigenvectors. This equates to aligning the model along the directions of maximal dispersion: given that $p(y)$ is Gaussian, this solution may appear counterintuitive as it seems to confer low likelihood to the model. However, one must take into account that the maximum-likelihood solution for σ^2 is given by the average of the discarded eigenvalues: choosing to discard larger eigenvalues would cause a larger overall covariance for $p(y)$, and worse likelihood. Therefore, within PPCA, it is the assumptions on the noise that determine the retained and the discarded subspaces. For FA, no closed-form solution is available and the determination of the retained and discarded subspaces depends on the initialization of iterative solvers. A physical interpretation of FA is therefore more elusive.

It is relatively straightforward to combine multiple individual models into a mixture model [8]. The rationale for this is to obtain multiple, locally-linear models which can well approximate a nonlinear manifold. When mixtures of M component distributions are considered, class-labelled PCA, PPCA and FA naturally extend by adding the mixing parameters, $\alpha_l, l = 1..M$, and fitting the other parameters individually for each component [9, 10]. Given that closed-form solutions for the direct maximization of the likelihood are either impossible or simply less practical, Expectation-Maximization (EM) algorithms are commonly used for maximum-likelihood estimation of these parameters.

The above review of PCA, PPCA and FA highlights the assumptions in their respective models. There are, obviously, limitations deriving from such assumptions: for instance, the Gaussian assumption in the high dimensional space made by PPCA and FA makes the estimates sensitive to outliers. In addition, models based on minimum reconstruction error and orthogonal transformations such as PCA may not necessarily lead to high classification accuracy. The rationale of our model is therefore to relax some of these assumptions and look to experimental accuracy as the criterion for model validation.



(a) PCA mixture component (b) MPPCA component (c) MLiT component

Fig. 1 An illustration of a component in three different mixture models. The original two-dimensional data is in blue and the transformed data is in green.

3 Approach and Methodology

In this section, we describe MLiT, a method for generating a mixture distribution in a dimensionally reduced space that can be useful for density modelling and classification. We first describe the model in the next subsection. We then present the Expectation-Maximization algorithm devised to learn the model from a set of samples with maximum likelihood in Subsection 3.2. In Subsections 3.3.1 and 3.3.2, we present two regularized solutions based on constrained optimization and on normalization, respectively. Finally, in Subsection 3.4, we discuss the initialization procedure.

3.1 Mixture of Gaussians under linear transformations (MLiT)

Let us consider a multivariate random variable, y, in a high, P-dimensional space. We define the lower, D-dimensional space through a compressive linear model

$$x = \Omega y \tag{2}$$

where Ω is a $D \times P$ real matrix, with $D <= P$ and typically $D << P$. We also posit a density function, $p(x)$, in x-space and consider

$$f(y) = p(\Omega y) = p(x); \tag{3}$$

$f(y)$ is not a proper density in y-space: rather, a probability function that repeats the probability density $p(x)$ for all y points satisfying $x = \Omega y$. As such, $f(y)$ expresses the probability of the combination of two distributions: a distribution modelled by $p(x)$ in the D-dimensional subspace spanned by the rows of Ω (the *retained* dimensions); and a uniform distribution along the $(P - D)$-dimensional subspace satisfying

equation $x = \Omega y$ for any given x (the *discarded* dimensions). For instance, if $p(x)$ is Gaussian, $f(y)$ has the shape of a Gaussian "ridge" i.e. a D-dimensional Gaussian function which repeats itself along the direction of $x = \Omega y$ in y-space. When referring to its distributional properties hereafter, we will refer to this distribution as Gaussian-uniform. We derive the motivation for this model from an analogy with sensor measurements, where x can be seen as a view of y made available by a sensor. If the representation power of x is adequate, it will permit to successfully study properties of y e.g. classify measurements in classes of interest.

In general, exploiting an array of M sensors can offer a richer representation of y than a single sensor. By calling $f(y|l)$ the probability function for the l-th sensor in the array, $l = 1..M$, it holds that:

$$f(y|l) = p(\Omega_l y|l) \tag{4}$$

where we have assumed that each sensor has its own independent view of y, expressed by Ω_l [14].

Let us now assume that we have a way to estimate a discrete distribution, $p(l)$, stating the *quality* of the l-th sensor at explaining the y sample. From Bayes theorem, we obtain:

$$f(y, l) = f(y|l)p(l) = p(\Omega_l y|l)p(l) \tag{5}$$

By marginalizing over l, we obtain the probability function $f(y)$ for the sensor array case:

$$f(y) = \sum_{l=1}^{M} f(y, l) = \sum_{l=1}^{M} f(y|l)p(l) = \sum_{l=1}^{M} p(\Omega_l y|l)p(l) \tag{6}$$

which closely recalls the general density of a mixture distribution. However, probabilities are computed in subspaces spanned by linear transformations and such transformations differ for each component. For simplicity of treatment, we further assume that the individual sensor densities are Gaussian, and note $\alpha_l = p(l)$, obtaining:

$$f(y) = \sum_{l=1}^{M} \alpha_l \mathcal{N}(\Omega_l y|\mu_l, \Sigma_l) \tag{7}$$

where the $\mathcal{N}(\Omega_l y|\mu_l, \Sigma_l)$ terms are the densities in the subspaces; means μ_l, and covariance matrices Σ_l are the parameters of each Gaussian component in the l-th subspace for $l = 1..M$; weights α_l are the mixing coefficients.

When the parameters are estimated with maximum likelihood, Ω_l will have to take values such that the Gaussian model for the l-th component, μ_l and Σ_l, would maximize the likelihood i.e. small values for Σ_l would be privileged. This is equivalent to creating a maximally invariant view for the class by choosing a corresponding linear transformation of the original space. The Gaussian-uniform distribution retains the dimensions of minimum variance as an "invariant" for the class and models them in terms of location, scale and directions by a full Gaussian model. Conversely, it renounces to model the data in the subspace spanned by the discarded dimensions since its high (P-D) dimensionality yield extremely low data density. However, the scale of Ω_l in the Gaussian-uniform distribution needs to be constrained to prevent the obvious yet degenerate solution $\Omega_l = 0$. We further note that this model makes no attempt at positioning the subspaces over clusters of data in y-space or minimizing reconstruction errors. As such, the number of views is not in correspondence with the number of clusters in the sample set. Rather, each view is justified by a good likelihood fit i.e. providing high within-class invariance. Figure 1(c) shows an example of $f(y)$ for a view.

3.2 An Expectation-Maximization algorithm for MLiT

Given the above, our aim is to derive the Ω_l matrix for each component of the mixture together with the other parameters in a maximum-likelihood framework. To this aim, we consider a set of i.i.d. observations, $Y = \{y_i\}_{i=1..N}$, in the high dimensional space. Our goal is then that of finding values for parameters of (7) maximizing likelihood

$$L(\theta) = p(Y|\theta) = \prod_{i=1}^{N} f(y_i) = \prod_{i=1}^{N} (\sum_{l=1}^{M} \alpha_l \mathcal{N}(\Omega_l y_i|\mu_l, \Sigma_l)) \tag{8}$$

where $\theta = \{\Omega_l, \alpha_l, \mu_l, \Sigma_l\}$ and $l = 1..M$. As usual in similar cases, rather than attempting maximization of (8) directly, we adopt an EM approach. This requires positing the existence of a set of discrete, M-valued latent variables, $Z = \{z_i\}_{i=1..N}$, whose minimum requirement is that the expression of joint probability function $f(y_i, Z)$ be simpler than $f(y_i)$ itself. The target for maximization is the expected value of the complete-data log-likelihood,

$$Q(\theta, \theta^g) = \sum_{Y} [\ln(p(Y, Z|\theta)p(Z|Y, \theta^g)] \tag{9}$$

where θ and θ^g represent the new and old parameters, respectively. In (9), Z represents a single realization of the entire set of the latent variables and the summation extends over all its possible M^N values. Note that EM does not require $p(Y, Z|\theta)$ to be a normalized density and this suits our case.

We here assume that $p(z_i = l|y_i, \theta^g)$ expresses the probability that the l-th view be the best at explaining the y_i sample. An expression for the two terms in (9), $\ln(p(Y, Z|\theta))$ and $p(Z|Y, \theta^g)$, is derived in Appendix A, leading to

$$Q(\theta, \theta^g) = \sum_{l=1}^{M}\sum_{i=1}^{N} \ln(\alpha_l) p(z_i = l|y_i, \theta^g) + \\ + \sum_{l=1}^{M}\sum_{i=1}^{N} \ln(\mathcal{N}(\Omega_l y_i|\mu_l, \Sigma_l)) p(z_i = l|y_i, \theta^g) \quad (10)$$

where $p(z_i = l|y_i, \theta^g)$, or $p(l|y_i, \theta^g)$ for brevity, is the *responsibility* of the l-th component for the y_i sample [13]. Responsibilities are given by

$$p(l|y_i, \theta^g) = \frac{f(y_i, l|\theta^g)}{f(y_i|\theta^g)} = \frac{\alpha_l^g \mathcal{N}(\Omega_l y_i|\mu_l^g, \Sigma_l^g)}{\sum_{k=1}^{M} \alpha_k^g \mathcal{N}(\Omega_k y_i|\mu_k^g, \Sigma_k^g)} \quad (11)$$

The next step is the maximization of (10) to which we proceed by computing the set of the first-order partial derivatives and equating them to zero. This operation requires an analysis of the interdependencies between parameters. Like in any other mixture, each partial derivative in α_l depends only on α_l itself and can be resolved independently of the other parameters. This leads to the following re-estimation formula

$$\alpha_l = \frac{1}{N}\sum_{i=1}^{N} p(l|y_i, \theta^g) \quad (12)$$

The next parameter which we aim to solve for is μ_l. By equating its first-order partial derivative to zero and with further manipulation, we obtain

$$\mu_l = \frac{\sum_{i=1}^{N} \Omega_l y_i p(l|y_i, \theta^g)}{\sum_{i=1}^{N} p(l|y_i, \theta^g)} \quad (13)$$

Given that Ω_l does not depend on i, it also holds that

$$\mu_l = \Omega_l \frac{\sum_{i=1}^{N} y_i p(l|y_i, \theta^g)}{\sum_{i=1}^{N} p(l|y_i, \theta^g)} = \Omega_l \mu_{yl} \quad (14)$$

where with μ_{yl} we have noted the responsibility-weighted sample mean in y-space.

Equation (13) shows that μ_l depends on Ω_l; therefore, the zeros of their derivatives should be determined jointly to guarantee a maximum. However, to avoid the complexity of the joint optimization, we decided to update these two parameters with a co-ordinate ascent approach where, in the current EM iteration, Ω_l is optimized based on the value of μ_l^g from the previous iteration and μ_l is optimized based on the value so determined for Ω_l. While this does not guarantee a maximum, it guarantees that $Q(\theta, \theta^g)$ monotonically increases along the iterations as in Generalized EM [15].

Solving for Σ_l in an analogous way leads to the following expression:

$$\Sigma_l = \frac{\Sigma_{i=1}^{N} (\Omega_l y_i - \mu_l)(\Omega_l y_i - \mu_l)^T p(l|y_i, \theta^g)}{\sum_{i=1}^{N} p(l|y_i, \theta^g)} \\ = \Omega_l \Sigma_{yl} \Omega_l^T \quad (15)$$

where with Σ_{yl} we have noted the responsibility-weighted sample covariance in y-space. Following the same rationale as for (13), we determine the value for Σ_l in (15) based on the values determined for Ω_l and μ_l in the current iteration.

The maximization of $Q(\theta, \theta^g)$ in Ω_l is formally a weighted, multivariate linear regression problem, with the responsibilities (11) as the weights. A closed-form solution exists for this problem [13]. However, in this section we propose a column-wise maximization since this will later permit us to efficiently impose the required regularization terms. We therefore break Ω_l into its P column vectors, $(w_j)_l$, $j = 1..P$, ignore the first term as it does not depend on Ω_l, and re-write the second term, $Q_2(\theta, \theta^g)$, as

$$Q_2(\theta, \theta^g) = \sum_{l=1}^{M}\sum_{i=1}^{N} \ln(\mathcal{N}(\Omega_l y_i|\mu_l, \Sigma_l)) p(l|y_i, \theta^g) = \\ = \sum_{l=1}^{M}\sum_{i=1}^{N} \big(((-\frac{1}{2}\ln(|\Sigma_l|) - \\ - \frac{1}{2}((w_1)_l y_{i1} ... + (w_j)_l y_{ij} ... + (w_P)_l y_{iP} - \mu_l)^T \\ \times \Sigma_l^{-1}((w_1)_l y_{i1} ... + (w_j)_l y_{ij} ... + (w_P)_l y_{iP} - \\ - \mu_l))p(l|y_i, \theta^g)\big) \quad (16)$$

where the $y_{ij}, j = 1..P$, are the scalar elements of sample y_i. Our approach consists of differentiating (16) with respect to each $(w_j)_l$ while keeping the other columns constant, and equating to zero to obtain the solution for the column. This approach guarantees an increase

in (16) along the lines, again, of Generalized EM. For simplicity of notation, henceforth, we omit the l index and the parentheses when referring to the column vectors.

Thus, in order to obtain the solution for w_j we differentiate (16) and obtain

$$\begin{aligned}\frac{\partial(Q_2(\theta,\theta^g))}{\partial w_j} &= \sum_{i=1}^{N}\big(\Sigma_l^{-1g}(w_1^g y_{i1}... + w_j y_{ij}... \\ &+ w_P^g y_{iP} - \mu_l^g) y_{ij} p(l|y_i,\theta^g)) = \\ &= \Sigma_l^{-1g}\sum_{i=1}^{N}((w_1^g y_{i1}... + w_j y_{ij}... \\ &+ w_P^g y_{iP} - \mu_l^g) y_{ij} p(l|y_i,\theta^g)\big) = \\ &= 0.\end{aligned} \tag{17}$$

where the external sum is ignored since all its terms, but one, are null after the differentiation.

We note that (17) is of the form $Ab = 0$, with A equal to Σ_l^{-1g} and b, the remaining terms. Given that A is a full rank matrix, the only solution is for b to be equal to 0. This gives us the following simplified equation:

$$\begin{aligned}\frac{\partial(Q_2(\theta,\theta^g))}{\partial w_j} &= \sum_{i=1}^{N}((w_1^g y_{i1} + ... + w_j y_{ij} + ... + w_P^g y_{iP} - \\ &- \mu_l^g) y_{ij} p(l|y_i,\theta^g)) = \\ &= 0.\end{aligned} \tag{18}$$

Therefore, w_j can be eventually obtained as

$$w_j = \frac{\sum_{i=1}^{N}(-w_1^g y_{i1}... - w_P^g y_{iP} + \mu_l^g) y_{ij} p(l|y_i,\theta^g)}{\sum_{i=1}^{N} y_{ij}^2 p(l|y_i,\theta^g)} \tag{19}$$

Equation (19) gives the desired re-estimation formula for w_j; analogously, we derive the re-estimation formulas for the other column vectors of Ω_l. During an EM iteration, the computation of the value for w_j uses the values determined so far for $w_k, 0 \leq k < j$, and values from the previous iteration w_k^g for $j < k \leq P$, the columns yet to update.

From a computational complexity perspective, it is generally advantageous to compute values $x_{li} = \Omega_l y_i$, $i = 1..N$, explicitly and use them in place of $\Omega_l y_i$ for both training and evaluation. This makes the computation only linear in the higher dimension, P, rather than quadratic. For instance, evaluation of (7) attracts $O(MPD)$ complexity, which is equivalent to that of MPPCA [2, 9].

3.3 Regularization

The proposed model (7) aims to optimize the transformation matrix, Ω_l, jointly with the fitting of a mixture density in the sub-spaces, thus departing from the orthonormal restriction of PCA projections. Its major drawback is that, in the absence of alternative constraints, it permits a closed-form solution where Ω_l is equal to zero: an identically null Ω_l transforms any y sample to the origin of the sub-space, leading to a null covariance therein and artificially infinite likelihood. The same phenomenon occurs for the iterative solution presented in this section: as iterations proceed, Ω_l tends to zero since Σ_l does correspondingly (as shown by (15)). However, the proposed solution allows us to introduce constraints preventing the norm of Ω_l from becoming smaller than a given value. This can be achieved by an equality constraint imposing a specific norm, or "scale", for Ω_l. Moreover, it is important that the scale of the Ω_l transformations be the same across components and across classes for likelihoods to be comparable. We present a constrained and a normalized solutions in the following subsections.

3.3.1 Constrained solution

Regularization of maximum-likelihood solutions can be achieved by using priors adequate for the type of regularization required. This converts the point estimate objective of maximum likelihood (8) into a maximum a posteriori:

$$\theta^* = \arg\max_{\theta}(p(Y|\theta)p(\theta)) \tag{20}$$

Common examples are the limitation in the parameters' size achieved by L2-norm priors and the enforcement of sparse solutions typical of L1-norm priors [13]. Rather, our goal is to strictly impose the same norm on all the Ω_l transformations in our models (7); we express this constraint as $g(\Omega_l) = \|\Omega_l\| - s = 0$, where s is the chosen value for the norm. Imposing this constraint is equivalent to using prior

$$p(\Omega_l) = \begin{cases} 1 & g(\Omega_l) = 0 \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

This prior retains the shape of the likelihood in certain regions of the parameter space and is therefore simple to impose. Further, it guarantees that the norm of all transformations be the same across components and classes. The full derivation of the constrained solution is presented in Appendix B.

3.3.2 Normalized solution

In this subsection, we present a simpler, alternative approach to the regularization of the solution of (17) that normalizes the transformation matrix, Ω_l, after each M step (MLiT (N)). Normalization is achieved by dividing Ω_l by its norm

$$\Omega_l = \frac{\Omega_l}{norm(\Omega_l)} \tag{22}$$

A number of matrix norms can be used in (22) such as L-2 (the maximum singular value of Ω), L-1, L-infinity, Frobenius and others, or even entrywise norms such as that we used in (34). However, from our experiments, we noticed that the Frobenius norm provided the highest and most stable results. Therefore, in the following, we report results based on this norm

A major difference between the constrained and the normalized solution is as follows: in the former, the M step is only allowed to find parameters satisfying the constraint. In the latter, the M step searches for parameter values in an unconstrained manner, and then normalizes the solution found. The expectation-maximization steps become therefore expectation-maximization-normalization steps. A second major difference derives from the column-wise update of the constrained solution, that restricts the constraint to apply to one column at a time as in (35), rather than on the entire matrix (34). This prevents the evolving solution from exploring variable proportions between the columns' norms. It is therefore evident that the normalized solution is allowed to explore the parameter space more broadly, and possibly produce a more effective final value. Its obvious disadvantage is that the normalized parameters might or might not produce higher likelihood than at the previous iteration. For this reason, we monitor the evolution of the likelihood along the iterations and elicit ad-hoc criteria for convergence.

3.4 Initialization and Convergence

In this subsection, we discuss the initialization strategy for the MLiT EM algorithm. In EM, the parameter values traversed along the iterations and the likelihood value achieved at convergence may strongly depend on the parameters' initial values. Different approaches have been proposed for dealing with this issue, including: initialization by clustering; running multiple starts and choosing the solution that provides the highest log-likelihood; split-and-merge operations; and others [16]. For our approach, we choose to apply a deterministic initialization to ensure repeatable results at each run. Namely, we decided to initialize the projection matrix, Ω, by the orthonormal transformation provided by PCA, selecting either the *largest* or the *smallest eigenvectors* (i.e. the eigenvectors associated with the largest and smallest eigenvalues, respectively). Projecting the data by the largest eigenvectors transforms them into a space where their variance is maximized and, under the hypothesis that their distribution be Gaussian, the likelihood is minimum amongst all orthonormal projections [17]. Conversely, projecting them with the smallest eigenvectors transforms them into a space where their variance is minimized and likelihood is maximum. Initializing with the largest eigenvectors forces EM to explore a large region of the parameter space before convergence. In contrast, initializing with the minimum eigenvectors is likely to start EM near a local maximum of the likelihood and typically permits a faster convergence. In our experiments, we noticed that there is no certain initialization method that always provide the best classification accuracy. Thus, we experiment with both methods and choose that providing greater accuracy over a cross-validation test.

As the data per class are projected to each of the components, the initial parameters of the EM algorithm are chosen as follows:

- The initial mean μ_l and covariance matrix Σ_l of each component are computed as the sample mean and sample covariance of the projected data.
- The initial priors α_l for $l = 1..M$, are chosen to be equal across all the components.

In Tables 1 and 2, we present a summary of the MLiT learning algorithm and its initialization procedure respectively.

4 Experiments and analysis

The empirical evaluation of a classifier's accuracy requires extensive testing over multiple data sets and a comparative analysis with existing, state-of-the-art classifiers. To this aim, in this section we present details on the data sets used and experiments conducted.

4.1 Data sets

We evaluate the proposed method on five data sets, four of which are selected from the UCI Machine Learning Repository [18], and are widely used by the pattern recognition community for evaluating learning algorithms. These four data sets are the Vehicle data set, Wisconsin Diagnostic Breast Cancer data set (WDBC), Wisconsin Prognostic Breast Cancer (WPBC) data set,

Table 1 Summary of MLiT main algorithm.

Main Algorithm. It is executed for each c class separately, where $c = 1..C$

$[\Omega_l, \theta_l]_{l=1..M}$=**MLiT**$(Method, Y, D, M, S)$

Input:
- $Method$: $'CONS'$ for Constrained and $'NORM'$ for Normalized
- $Y = \{y_n\}_{n=1..N}$: set of N samples, with $dim(Y) = P$xN
- D: number of reduced dimensions
- M: number of components
- S: scale of the transformation

Output:
- Ω_l: final transformation matrix for the l-th reduced space, $l = 1, ...M$, $dim(\Omega_l) = D$xP
- $\theta_l = \{\alpha_l, \mu_l, \sigma_l\}$: final parameters of the Gaussian for the l-th reduced space

Local variables:
- $X_l = \{x_{ln}\}_{n=1..N}$: transformed data in the l-th reduced space, with $dim(X_l) = D$xN

1. $[X_l, \Omega_l, \theta_l]_{l=1..M}$=**INITIALIZE**$(Method, Y, D, M, S)$
2. **IF** $Method = 'CONS'$ **THEN**
   - Column counter for Ω_l: $j = 1$;

   END IF
3. **Do EM until convergence, maximum iterations or stop condition**
4. **E-step**:
   $p(l|y_i, \theta) = \frac{\alpha_l \mathcal{N}(x_{li}|\mu_l, \Sigma_l)}{\sum_{k=1}^{M} \alpha_k \mathcal{N}(x_{ki}|\mu_k, \Sigma_k)}$
5. **M-step**:

   IF $Method = 'CONS'$ **THEN**
   - w_j^{new}: Lagrangian equation for single column update imposing 2^D linear constraints
   - Among the 2^D possible constraints on w_j, select the first such that $norm(w_j) = S/P$
   - IF none THEN stop END IF
   - Increase column counter for updating the next column: $j = j + 1$
   - IF j=P+1 THEN j=1 END IF

   ELSE-IF $Method = 'NORM'$ **THEN**
   - Ω_l^{new}: $\{w_j^{new} = f(w_1^{new}, ..., w_{j-1}^{new}, w_{j+1}, ..., w_P)\}$,$j = 1, .., P$
   - Normalize $\Omega_l = \frac{\Omega_l}{norm(\Omega_l)} S$

   END IF
   - Compute new transformed data: $X_l^{new} = \Omega_l^{new} Y$
   - Compute new values for $\alpha_l^{new}, \mu_l^{new}, \Sigma_l^{new}$
   - IF convergence or maximum number of iterations THEN stop END IF
   - Increase number of iterations counter
6. **Repeat** from 3.

Table 2 Summary of MLiT initialization phase.

Initialization Algorithm:

$[X_l, \Omega_l, \theta_l]_{l=1..M}$=**INITIALIZE**$(Method, Y, D, M, S)$

Input:
- $Method$, Y, D, M and S as in **MLiT**

Output:
- $X_l = \{x_{ln}\}_{n=1..N}$: initial transformed data in the l-th reduced space, with $dim(X_l) = D$xN
- Ω_l: initial transformation matrix for the l-th reduced space, $l = 1, ...M$, $dim(\Omega_l) = D$xP
- $\theta_l = \{\alpha_l, \mu_l, \Sigma_l\}$: initial parameters of the Gaussian for the l-th reduced space

1. Calculate $U = Eig(COV(Y))$, where U are the eigenvectors
2. Initialize Ω_l using consecutive largest or smallest eigenvectors of U
3. Express Ω_l as P, Dx1 column vectors with index $j \rightarrow$ $\rightarrow \Omega_l = \{(w_j)_l\}$, $j = 1..P$
4. **IF** $Method = 'CONS'$ **THEN**
   - Normalize each column vector as $(w_j)_l = \frac{(w_j)_l}{norm((w_j)_l)} \frac{S}{P}$

   ELSE-IF $Method = 'NORM'$ **THEN**
   - Normalize the whole matrix as $\Omega_l = \frac{\Omega_l}{norm(\Omega_l)} S$

   END IF
5. Compute initial transformed data: $X_l = \Omega_l Y$
6. Compute initial values for $\theta_l = \{\mu_l, \Sigma_l, \alpha_l\}$:
   - $\alpha_l = \frac{1}{M}$
   - $\mu_l = \frac{1}{N} \sum_{n=1}^{N} x_{ln}$
   - $\Sigma_l = \frac{1}{N-1} \sum_{n=1}^{N} (x_{ln} - \mu_l)(x_{ln} - \mu_l)^T$

and Optical Handwritten Digits data set (OpticDigit). The last data set, named Public Premises Video Surveillance data set (PPVS), was collected by the authors themselves.

The Vehicle data set involves classification of a given silhouette as one of four types of vehicles, namely, "bus", "Opel", "Saab" and "van". The vehicle silhouettes are described by various shape measurements. The rationale for choosing this data set is that it is the most similar in the UCI repository to our own data set and can offer a comparative insight into the method's performance. The WDBC and WPBC data sets contain various shape features from images of fine needle aspirates (FNA) of breast mass for diagnosis and prognosis of breast cancer. The OpticDigit data set is based on rescaled bitmaps of handwritten digits: the original 32x32 black and white bitmaps are divided into non-overlapping blocks of 4x4 pixels and the number of 'on' pixels counted in each block, resulting in a 64-dimensional feature vector of homogeneous features. The Public Premises Video Surveillance data set (PPVS) is based on video footage provided by an industrial partner. It involves classification of an object in a video surveillance environment into one of four classes: "trolleys", "bags", "single persons", and "groups of people". The images of these objects have been clipped from video footage acquired at a number of airports and train stations world-wide. The feature set consists of statistics of various local features such as line segments, circles, corners, and global shape descriptors such as fitted ellipses and bounding boxes. This feature set is described in detail in [19].

As we can conclude from the previous paragraphs and the data displayed in Table 3, there are major differences between these five data sets in terms of:

- nature of data and application context,
- number of instances available,
- number of features extracted,
- types of features used for representation, and
- number of classes.

Therefore, the chosen data sets offer a suitable basis for comparative analysis.

Table 3 Comparative summary of the data sets used.

Data set	# Features	# Instances	# Classes
Vehicle	18	846	4
OpticDigit	64	5620	10
WDBC	30	569	2
WPBC	33	198	2
PPVS	44	600	4

4.2 Experiments

In this subsection, we present classification results for the proposed method on the five aforedescribed data sets. We compare the performance of our approach with that of mixture of PPCA (MPPCA) and Gaussian mixture model (GMM). Experiments with these classifiers were carried out in MATLAB by setting all tunable parameters in the most genuine way to achieve the highest performance. We summarize below the main parameters, and in Table 4, we report the values that achieved the best accuracy results. The parameters are as follows:

- GMM: There is one main parameter, the number of the GMM components (M).
- MPPCA: There are two main parameters, the number of reduced dimensions (D), and the number of mixture components (M).

For MLiT, the initial parameters for the mixture distribution and transformation matrices for each class are as follows:

- The number of the mixture components (M) and reduced dimensions (D) were manually selected as reported in Table 4.
- The initial transformation matrices for each class, $\Omega^{[0]}$, were computed by using either the smallest or the largest consecutive eigenvectors of the covariance matrix of the original data. For example, in the case of largest eigenvectors, two components per class ($M = 2$), and a reduced space of three dimensions ($D = 3$), we select the three first eigenvectors for $\Omega_1^{[0]}$ and the eigenvectors between the third and the fifth for $\Omega_2^{[0]}$.
- Transformed data: $X_l^{[0]} = \Omega_l^{[0]} Y$, $l = 1..M$.
- Covariance type in the reduced space: full.
- Means, $\mu_l^{[0]}$, and covariances, $\Sigma_l^{[0]}$, $l = 1..M$: computed as the sample means and sample covariances of the transformed data.
- Equal priors for all components, $\alpha_l^{[0]} = \frac{1}{M}, l = 1..M$.

A covariance matrix can become singular wherever a component maps one sample only, causing artificially infinite likelihood. With our method, the covariance matrix may also become singular if Ω takes a rank that is less than D. To prevent this singularity from occurring, we simply add a small value of 0.01 to the elements on the principal diagonal of all covariance matrices.

As stopping criteria, for MLiT (C) the EM algorithm continues to iterate until either there are no more solutions satisfying the imposed constraints (36) and (37), or EM converges. We set the convergence threshold on the difference between two consecutive values of the log-likelihood to 0.01. Convergence of MLiT (C) is ensured as the constrained solution guarantees monotonic increase of the likelihood. However, for MLiT (N) the normalization step does not guarantee such a monotonic increase in the likelihood; hence, we elicit an ad-hoc criteria for stopping by running the EM algorithm for 50 iterations and choosing that delivering the maximum accuracy by cross-validation. For MPPCA, we observed that the accuracy stabilized after 200 iterations. For GMM, instead, accuracy stabilization was empirically achieved after 50 iterations.

Classifiers were all trained in a supervised manner. Once an $f(y|c)$ probability function is learned from samples of each c class, $c = 1..C$, maximum likelihood classification is simply given by

$$c^* = \arg\max_c (f(y|c)) \tag{23}$$

For validation, we have chosen 5-fold cross-validation since it offers a good trade off between the large bias of the hold-out method and the large variance of the leave-one-out method [20, 21]. This implies randomly partitioning the data set into five disjoint subsets, training the classifier with four and using the last for testing. Classification accuracy is averaged over five runs by using, in turn, each fold for testing. We express classification accuracy simply as the percentage of correctly classified instances with respect to their total number:

$$\text{accuracy} = \frac{\text{number of correctly classified samples}}{\text{total number of samples}} \tag{24}$$

It is important to note that, in the following, we report the classification results in terms of two statistical measures: average accuracy over the various runs, and standard deviation. However, we chose the average accuracy as the main measure for comparing the different classifiers; nevertheless, the standard deviation is an important measure for the precision of the classification accuracy, and it can be considered together with the accuracy for a better estimate of the classification performance.

Table 4 Results for 5-fold CV in terms of accuracy (%), on five data sets and across different classifiers, with the highest indicated in boldface font. For each dataset, the first row presents the main parameters' values for different classifiers, and the second row shows the achieved accuracy (%).

Classifier	MLiT (C)		MLiT (N)		MPPCA		GMM
Parameters	D	M	D	M	D	M	M
Dataset							
Vehicle	14	2	14	2	10	2	2
(%)	80.6 ±2.1		**85.6 ±1.9**		83.6 ±1.3		82.8 ±1.9
OpticDigit	19	5	29	2	16	1	2
(%)	96.0 ±0.5		98.4 ±0.3		**98.6 ±0.4**		96.9 ±0.3
WDBC	7	4	18	1	20	2	2
(%)	95.4 ±1.5		**96.1 ±1.7**		94.7 ±1.7		95.9 ±1.8
WPBC	6	2	4	4	15	4	4
(%)	76.9 ±0.0		**77.4 ±1.1**		76.9 ±0.0		75.9 ±1.4
PPVS	15	2	33	1	25	2	2
(%)	72.8 ±4.9		**76.7 ±2.4**		73.5 ±4.0		73.3 ±2.1

Table 4 reports the best results of 5-fold cross-validation on the various data sets and across the compared classifiers. We note that in this table, for MLiT, all results are obtained with largest eigenvectors initialization except the cases of MLiT (N) on Vehicle and WPBC data sets.

It is clear from this table that, in all cases, MLiT (N) outperformed MLiT (C). This can be explained by the greater freedom that MLiT (N) enjoys in searching the parameter space during training. Hence, it has higher chance of finding better values.

We can also note from Table 4 that the performance of MLiT outperformed that of MPPCA on four out of the five compared data sets with improvements ranging from 0.5% to 3.2%, proving the strength of MLiT against a state-of-the-art classifier (MPPCA slightly outperformed MLiT by 0.2% only on the OpticDigit data set). Moreover, we can note that, in all cases, the performance of MLiT outperformed that of GMM with improvements ranging from 0.2% to 3.4%. This illustrates the ability of MLiT in overcoming the curse of dimensionality and providing better performance in the reduced space. MPPCA has also provided better classification results than GMM on majority of the data sets.

Overall, the experiments on the five data sets presented in this section showed that MLiT generally reported higher experimental accuracy over both compared classifiers. Interpretation of accuracy results in high dimensional spaces is not immediate: we lean to attribute these improvements to the Gaussian-uniform distribution property of focussing on invariant features. This permits the building of compact models that have proved discriminative when used with the Bayes inversion rule, while it introduces elements of robustness since outliers are ousted to the discarded dimensions during training as much as possible. The non-orthogonality of the transformation adds further degrees of freedom to the model.

5 Conclusions

In this paper, we have presented a novel method for linear dimensionality reduction within mixture distributions. The model that we have proposed for the class-conditional likelihood is a mixture of Gaussian distributions under linear transformations (7). This model equates to a uniform distribution along the discarded dimensions and a full Gaussian model along the retained dimensions. It is important to contrast this model properly to the several existing methods for linear dimensionality reduction in mixture models such as mixtures of PCA, PPCA, FA and t-distributions. The main point of difference is that the linear transformation is not restricted to be orthogonal or related to eigenvectors. Further, the linear model adopted, $x = \Omega y$, does not assume additive noise models and makes x observable. On the ground of that, we can evaluate density $\mathcal{N}(\Omega y|\mu, \Sigma) = \mathcal{N}(x|\mu, \Sigma)$ directly in x-space. However, the non-orthogonality of the transformation make the scale of the compressed x-spaces become arbitrary. Therefore, we have proposed two regularized solutions that impose a common scale to all the transformations based on constrained optimization (MLiT (C)) and normalization (MLiT (N)). However, from the experimental results, we noticed that MLiT (N) delivers higher classification results compared to MLiT (C). As we presented earlier, this is explained by the fact that MLiT (N) enjoys more freedom in exploring the parameter

space during training, possibly finding better solutions compared to MLiT (C).

The experimental performance of MLiT has proved to outperform that of MPPCA and GMM in almost all cases with improvements ranging from 0.2% to 3.4% compared to the runner-up. The only case where MLiT did not deliver the best accuracy is on the OpticDigit data set where MPPCA slightly outperformed MLiT by 0.2%.

In addition to visual object classification, the proposed method permits general application for density modeling and classification of other continuous numerical data requiring dimensionality reduction. Moreover, its re-estimation formulas can be easily extended to suit boosting and other weighted maximum likelihood targets and adapt to a variety of pattern recognition frameworks.

A The expected complete-data log likelihood for MLiT

In this appendix, we show the derivation of the two main terms in (9),

$$Q(\theta, \theta^g) = \sum_{Y} \big[\ln(p(Y, Z|\theta)) p(Z|Y, \theta^g) \big] \tag{25}$$

namely, the complete data log-likelihood, $\ln(p(Y, Z|\theta)$, and the posterior for the latent variables, $p(Z|Y, \theta^g)$. Whereas we report this derivation herewith for completeness, it could be easily derived from the equivalent derivation for conventional Gaussian mixtures.

We begin with the derivation of an expression for $\ln(p(Y, Z|\theta)$. We first apply Bayes theorem

$$\ln(p(Y, Z|\theta) = \ln(p(Y|Z, \theta) p(Z|\theta)). \tag{26}$$

Given the following assumptions: a) the mutual independence of the y_i observations b) the dependence of y_i only on its own latent variable, z_i and c) the mutual independence of the z_i, we have

$$\begin{aligned}
\ln(p(Y|Z, \theta) p(Z|\theta)) &= \ln\big(\prod_{i=1}^{N} (p(y_i|Z, \theta)) p(Z|\theta) \big) \\
&= \ln\big(\prod_{i=1}^{N} (p(y_i|z_i = l, \theta)) p(Z|\theta) \big) \\
&= \ln\big(\prod_{i=1}^{N} (p(y_i|z_i = l, \theta)) \prod_{i=1}^{N} p(z_i = l|\theta) \big) \\
&= \ln\big(\prod_{i=1}^{N} [(p(y_i|z_i = l, \theta)) p(z_i = l|\theta)] \big) \\
&= \sum_{i=1}^{N} \ln\big[(p(y_i|z_i = l, \theta)) p(z_i = l|\theta) \big]
\end{aligned} \tag{27}$$

The argument of the logarithm can be written as

$$p(y_i|z_i = l, \theta) p(z_i = l|\theta) = \mathcal{N}(\Omega_l y|\mu_l, \Sigma_l)\, \alpha_l. \tag{28}$$

Therefore, we have the final equivalence

$$\ln(p(Y, Z|\theta)) = \sum_{i=1}^{N} \ln\big[\alpha_l \mathcal{N}(\Omega_l y_i|\mu_l, \Sigma_l) \big]. \tag{29}$$

The next term needed is posterior $p(Z|Y, \theta^g)$. Under the assumptions above, the probability of any entire assignment, Z, conditioned on the observations is

$$p(Z|Y, \theta^g) = \prod_{i=1}^{N} p(z_i = l|y_i, \theta^g). \tag{30}$$

Let us then apply Bayes theorem, again, to $p(z_i = l|y_i, \theta^g)$

$$p(z_i = l|y_i, \theta^g) = \frac{p(z_i = l|\theta^g) p(y_i|z_i = l, \theta^g)}{p(y_i|\theta^g)} \tag{31}$$

In the above, we have three terms in the right member. In the numerator, there are the terms that we have computed for (28). The denominator is the marginal probability of y over all the components. Therefore, the above equation becomes

$$p(z_i = l|y_i, \theta^g) = \frac{\alpha_l^g \mathcal{N}(\Omega_l y_i|\mu_l^g, \Sigma_l^g)}{\sum_{k=1}^{M} \alpha_l^g \mathcal{N}(\Omega_l y_i|\mu_l^g, \Sigma_l^g)} \tag{32}$$

justifying our formula for the responsibilities. We can now replace both terms (29), (30) in $Q(\theta, \theta^g)$ to obtain

$$Q(\theta, \theta^g) = \sum_{Y} \big[\big(\sum_{i=1}^{N} \ln[\alpha_l \mathcal{N}(\Omega_l y_i|\mu_l, \Sigma_l)] \big) \big(\prod_{i=1}^{N} p(z_i = l|y_i, \theta^g) \big) \big] \tag{33}$$

The following simple steps leading to (10) can be repeated from [22].

B A constrained Expectation-Maximization algorithm for MLiT

In this appendix, we present the regularized solution based on this constrained optimization (MLiT (C)). Again, we update one column vector, w_j, at a time while keeping the others still. As a constraint on Ω_l, $g(\Omega_l) = 0$, we choose an *entrywise L1-norm* constraint, $g(\Omega_l) = \|\Omega_l\|_{L1} - s = 0$; constant s is the chosen value for the norm, or *scale*:

$$g(\Omega_l) = \sum_{i=1}^{D} \sum_{k=1}^{P} |w_{ik}| - s = 0 \tag{34}$$

where $dim(\Omega_l) = D\text{x}P$. However, as we update only one w_j column at a time, we need to impose this constraint in a column-wise manner. Therefore, we turn (34) into the stronger constraint $g(w_j)$

$$g(w_j) = \sum_{i=1}^{D} |w_{ij}| - (c = s/P) = 0 \tag{35}$$

A solution for (17) under constraint (35) may be obtained by using a constrained optimization solver such as CML [23]. However, we prefer to provide an inline solution for reasons of speed and independency. Constraint (35) can be represented in an expanded notation as

$$g(w_j) = \pm w_{1j} \pm w_{2j} \pm ... \pm w_{Dj} - c = 0 \tag{36}$$

under condition

$$|w_{1j}|, |w_{2j}|, ..., |w_{Dj}| \leq c. \tag{37}$$

Eq. (36) shows the 2^D different combinations of signs that the constraint can take, leading to 2^D different linear constraints that do not use absolute values. Whereas the number of such constraints is significant, it is tolerable for typical values of D. In contrast, alternative approaches for computing equality-constrained solutions impose constraints on the rows of Ω [24]. The number of constraints so required is in the order of 2^P, exponential in the high dimension, P, and therefore unmanageable. This is the ultimate justification for our choice of a column-wise solution for the maximization of $Q(\theta, \theta^g)$ in Ω_l. Our constrained approach first solves (17) under each of the (36) linear constraints. If solutions are found, they are then tested post-hoc for satisfaction of (37).

For examplification, let us consider one of the (36) constraints

$$g(w_j) = w_{1j} + w_{2j}... + w_{Dj} - c = 0 \tag{38}$$

Then, consider the Lagrangian equation

$$\begin{aligned} h(w_j) &= f(w_j) + \lambda g(w_j) \\ &= f(w_j) + \lambda\big(\sum_{i=1}^{D} w_{ij} - c\big) \end{aligned} \tag{39}$$

where $f(w_j)$ is $Q(\theta, \theta^g)$ (10) (or (16) likewise), λ is the Lagrange multiplier, and $g(w_j)$ is the constraint. Eq. (39) can be written as

$$\begin{aligned} h(w_j) = \sum_{i=1}^{N} \big(&(-\frac{1}{2}\ln(|\Sigma_l|) - \\ &- \frac{1}{2}(w_1 y_{i1} + ... + w_j y_{ij} + ... + w_p y_{ip} - \mu_l)^T \\ &\times \Sigma_l^{-1}(w_1 y_{i1} + ... + w_j y_{ij} + ... + w_p y_{ip} - \mu_l)) \\ &\times p(l|y_i, \theta^g)\big) + \lambda\big(\sum_{i=1}^{D} w_{ij} - c\big) \end{aligned} \tag{40}$$

where the external sum is ignored since all its terms, but one, are null after the differentiation.

The derivative of (40) in w_j is

$$\begin{aligned} \frac{\partial(h(w_j))}{\partial w_j} = \sum_{i=1}^{N} \big(&\Sigma_l^{-1g}(w_1^g y_{i1} + ... + w_j y_{ij} + ... + w_p^g y_{ip} \\ &- \mu_l^g) y_{ij} p(l|y_i, \theta^g)\big) + \lambda\overline{1} = \\ &= 0 \end{aligned} \tag{41}$$

where $\overline{1}$ stands for a $D \times 1$ vector of all ones. By pre-multiplying (41) by Σ_l^g, we obtain:

$$\begin{aligned} \frac{\partial(h(w_j))}{\partial w_j} = \sum_{i=1}^{N} \big(&(w_1^g y_{i1} + ... + w_j y_{ij} + ... + w_p^g y_{ip} - \mu_l^g) \\ &\times y_{ij} p(l|y_i, \theta^g) + \lambda \Sigma_l^g \overline{1}\big) = \\ &= 0 \end{aligned} \tag{42}$$

By setting $r = \Sigma_{i=1}^{N} y_{ij}^2 p(l|y_i, \theta^g)$, and collectively naming b all terms in $\left[w_k^g\right]_{k=1..P, k\neq j}$, we obtain

$$\begin{aligned} Rl\frac{\partial(h(w_j))}{\partial w_j} &= r w_j + b + \lambda \Sigma_l^g \overline{1} \\ &= 0 \end{aligned} \tag{43}$$

The solution for w_j can then be written as

$$w_j = \frac{-b - \lambda \Sigma_l^g \overline{1}}{r} \tag{44}$$

Let us now call s_{nm} the $D \times D$ elements of Σ_l^g in row-column notation and make (44) explicit in its D rows

$$\begin{cases} w_{1j} = \frac{-b_1 - \lambda(s_{11} + ... + s_{1D})}{r} \\ ... \\ w_{Dj} = \frac{-b_D - \lambda(s_{D1} + ... + s_{DD})}{r} \end{cases}$$

We are eventually in a position to impose the $g(w_j)$ constraint by adding up all left and right members of (45). On the left, we obtain c, a known value. On the right, a linear function of λ. Therefore, λ can be solved for immediately as

$$\lambda = \frac{-(cr + b_1 + ... + b_D)}{(s_{11} + ... + s_{DD})} \tag{45}$$

With λ thus computed, its value is replaced in (44) to obtain the desired, constrained solution for w_j. For each of the other remaining constraints, λ is also calculated and the corresponding constrained solutions of w_j are obtained. We note that, in order for the EM algorithm to continue iterating, at least one constraint solution for w_j satisfying (36) and (37) is needed. We also note that there is no specific relevance for the choice of s: changing scale would lead to scaled densities for all components and classes and equivalent classification outcomes; possible differences can be imputed to numerical resolution. In the experiments, we have tried different values of s in a logarithmic scale and chosen that corresponding to the highest classification accuracy.

Acknowledgements The authors wish to thank the Australian Research Council and iOmniscient Pty Ltd that have partially supported this work under the Linkage Project funding scheme - grant LP0668325.

References

1. R. Bellman, editor. *Adaptive control processes - A guided tour*. Princeton University Press, Princeton, New Jersey, 1961.
2. Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*,, 61(3):611–622, 1999.

3. Sam Roweis. EM algorithms for PCA and SPCA. In *Advances in neural information processing systems*, volume 10, pages 626 – 632, Colorado, United States. The MIT Press.
4. D. J. Bartholomew, editor. *Latent Variable Models and Factor Analysis*. Charles Griffin & Co. Ltd., London, 1987.
5. A. Basilevsky, editor. *Statistical Factor Analysis and Related Methods*. Wiley, New York, 1994.
6. Smola Alexander Schölkopf, Bernhard and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.
7. Tat-Jun Chin and David Suter. Incremental kernel principal component analysis. *IEEE Transactions on Image Processing*, 16(6):1662–1674, 2007.
8. G. E. Hinton, P. Dayan, and M. Revow. Modeling the manifolds of images of handwritten digits. *IEEE Transactions on Neural Networks*, 8(1):65–74, 1997.
9. M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999.
10. Z. Ghahramani and G.E. Hinton. The EM algorithm for mixtures of factor analyzers. Technical Report CRG-TR-96-1, University of Toronto, 1997. the original paper for the mixture of factor analyzers.
11. D. D. Ridder and V. Franc. Robust subspace mixture models using t-distribution. In *14th British Machine Vision Conference (BMVC)*, pages 319–328, London, UK, 2003.
12. 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*, 39(1):1–38, 1977.
13. Christopher M. Bishop, editor. *Pattern Recognition and Machine Learning*. Springer, 2006.
14. J.V. Kittler. Combining classifiers: A theoretical framework. *Pattern Analysis and Applications*, 1(1):18–27, 1998.
15. R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*. Cambridge, MA: MIT Press, 1999.
16. M. A. F. Figueiredo and A. K. Jain. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):381–396, 2002. avoid singularity by applying deterministic annealing.
17. Richard J. Bolton and Wojtek J. Krzanowski. A characterization of principal components for projection pursuit. *The American Statistician*, 53(2):108–109, 1999.
18. A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.
19. A.F. Otoom, H. Gunes, and M. Piccardi. Towards automatic abandoned object classsifcation in visual surveillance systems. In *Asia-Pacific Workshop on Visual Information Processing*, pages 143–149, Tanian, Taiwan, 2007.
20. A. K. Jain, R. P. W. Duin, and Mao Jianchang. Statistical pattern recognition: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.
21. L. Breiman and P. Spector. Submodel selection and evaluation in regression: The x-random case. *International Statistical Review*, 60(3):291–319, 1992.
22. Jeff A. Bilmes. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian Mixture and Hidden Markov Model, 1998.
23. Ronald Schoenberg. Constrained maximum likelihood. *Computational Economics*, 10:251–266, 1997.
24. G. H. Golub and C. F. van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.