Pattern Recognition 45 (2012) 2804–2816

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Maximum likelihood estimation of Gaussian mixture models using stochastic search

Çağlar Arı [a], Selim Aksoy [b,*], Orhan Arıkan [a]

[a] Department of Electrical and Electronics Engineering, Bilkent University, Ankara 06800, Turkey
[b] Department of Computer Engineering, Bilkent University, Ankara 06800, Turkey

ARTICLE INFO

Article history:
Received 28 March 2011
Received in revised form
16 December 2011
Accepted 30 December 2011
Available online 11 January 2012

Keywords:
Gaussian mixture models
Maximum likelihood estimation
Expectation–maximization
Covariance parametrization
Identifiability
Stochastic search
Particle swarm optimization

ABSTRACT

Gaussian mixture models (GMM), commonly used in pattern recognition and machine learning, provide a flexible probabilistic model for the data. The conventional expectation–maximization (EM) algorithm for the maximum likelihood estimation of the parameters of GMMs is very sensitive to initialization and easily gets trapped in local maxima. Stochastic search algorithms have been popular alternatives for global optimization but their uses for GMM estimation have been limited to constrained models using identity or diagonal covariance matrices. Our major contributions in this paper are twofold. First, we present a novel parametrization for arbitrary covariance matrices that allow independent updating of individual parameters while retaining validity of the resultant matrices. Second, we propose an effective parameter matching technique to mitigate the issues related with the existence of multiple candidate solutions that are equivalent under permutations of the GMM components. Experiments on synthetic and real data sets show that the proposed framework has a robust performance and achieves significantly higher likelihood values than the EM algorithm.

1. Introduction

Gaussian mixture models (GMMs) have been one of the most widely used probability density models in pattern recognition and machine learning. In addition to the advantages of parametric models that can represent a sample using a relatively small set of parameters, they also offer the ability of approximating any continuous multi-modal distribution arbitrarily well like nonparametric models by an appropriate choice of its components [1,2]. This flexibility of a convenient semiparametric nature has made GMMs a popular choice for both density models in supervised classification and cluster models in unsupervised learning problems.

The conventional method for learning the parameters of a GMM is maximum likelihood estimation using the expectation–maximization (EM) algorithm. Starting from an initial set of values, the EM algorithm iteratively updates the parameters by maximizing the expected log-likelihood of the data. However, this procedure has several issues in practice [1,2]. One of the most important of these issues is that the EM algorithm easily gets trapped in a local maximum as the objective being a non-concave optimization problem. Moreover, there is also the associated problem of initialization as it influences which local maximum of the likelihood function is attained.

The common approach is to run the EM algorithm many times from different initial configurations and to use the result corresponding to the highest log-likelihood value. However, even with some heuristics that have been proposed to guide the initialization, this approach is usually far from providing an acceptable solution especially with increasing dimensions of the data space. Furthermore, using the results of other algorithms such as *k*-means for initialization is also often not satisfactory because there is no mechanism that can measure how different these multiple initializations are from each other. In addition, this is a very indirect approach as multiple EM procedures that are initialized with seemingly different values might still converge to similar local maxima. Consequently, this approach may not explore the solution space effectively using multiple independent runs.

Researchers dealing with similar problems have increasingly started to use population-based stochastic search algorithms where different potential solutions are allowed to interact with each other. These approaches enable multiple candidate solutions to simultaneously converge to possibly different optima by making use of the interactions. Genetic algorithm (GA) [3–7], differential evolution (DE) [8], and particle swarm optimization (PSO) [9–12] have been the most common population-based stochastic search algorithms used for the estimation of some form of GMMs. Even though these approaches have been shown to perform better than non-stochastic alternatives such as *k*-means and fuzzy *c*-means, the interaction

* Corresponding author. Tel.: +90 312 2903405; fax: +90 312 2664047.
E-mail addresses: cari@ee.bilkent.edu.tr (Ç. Arı), saksoy@cs.bilkent.edu.tr (S. Aksoy), oarikan@ee.bilkent.edu.tr (O. Arıkan).

0031-3203/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2011.12.023

mechanism that forms the basis of the power of the stochastic search algorithms has also limited the use of these methods due to some inherent assumptions in the candidate solution parametrization. In particular, the interactions in the GA, DE, and PSO algorithms are typically implemented using randomized selection, swapping, addition, and perturbation of the individual parameters of the candidate solutions. For example, the crossover operation in GA and DE randomly selects some parts of two candidate solutions to create a new candidate solution during the reproduction of the population. Similarly, the mutation operation in GA and DE and the update operation in PSO perturb an existing candidate solution using a vector that is created using some combination of random numbers and other candidate solutions. However, randomized modification of individual elements of a covariance matrix independently does not guarantee the result to be a valid (i.e., symmetric and positive definite) covariance matrix. Likewise, partial exchanges of parameters between two candidate solutions lead to similar problems. Hence, these problems confined the related work to either use no covariance structure (i.e., implicitly use identity matrices centered around the respective means) [7–10,12] or constrain the covariances to be diagonal [3,11]. Consequently, most of these approaches were limited to the use of only the mean vectors in the candidate solutions and to the minimization of the sum of squared errors as in the *k*-means setting instead of the maximization of a full likelihood function. Full exploitation of the power of GMMs involving arbitrary covariance matrices estimated using stochastic search algorithms benefits from new parametrizations where the individual parameters are *independently modifiable* so that the resulting matrices remain valid covariance matrices after the stochastic updates and have finite limits so that they can be searched within a *bounded* solution space. In this paper, we present a new parametrization scheme that satisfies these criteria and allows the estimation of generic GMMs with arbitrary covariance matrices.

Another important problem that has been largely ignored in the application of stochastic search algorithms to GMM estimation problems in the pattern recognition literature is identifiability. In general, a parametric family of probability density functions is identifiable if distinct values of the parameters determine distinct members of the family [1,2]. For mixture models, the identifiability problem exists when there is no prior information that allows discrimination between its components. When the component densities belong to the same parametric family (e.g., Gaussian), the mixture density with K components is invariant under the $K!$ permutations of the component labels (indices). Consequently, the likelihood function becomes invariant under the same permutation, and this invariance leads to $K!$ equivalent modes, corresponding to equivalence classes on the set of mixture parameters. This lack of *uniqueness* is not a cause for concern for the iterative computation of the maximum likelihood estimates using the EM algorithm, but can become a serious problem when the estimates are iteratively computed using simulations when there is the possibility that the labels (order) of the components may be switched during different iterations [1,2]. Considering the fact that most of the search algorithms depend on the designed interaction operations, performances of the operations that assume continuity or try to achieve diversity cannot work as intended, and the discontinuities in the search space will make it harder for the search algorithms to find directions of improvement. In an extreme case, the algorithms will fluctuate among different solutions in the same equivalence class, hence, among several equivalent modes of the likelihood function, and will have significant convergence issues. In this paper, we propose an optimization framework where the optimal correspondences among the components in two candidate solutions are found so that desirable interactions become possible between these solutions.

It is clear that a formulation that involves *unique*, *independently modifiable*, and *bounded* parameters is highly desired for effective utilization of stochastic search algorithms for the maximum likelihood estimation of unrestricted Gaussian mixture models. Our major contributions in this paper are twofold: we present a novel parametrization for arbitrary covariance matrices where the individual parameters can be independently modified in a stochastic manner during the search process, and describe an optimization formulation for resolving the identifiability problem for the mixtures. Our first contribution, the parametrization, uses eigenvalue decomposition, and models a covariance matrix in terms of its eigenvalues and Givens rotation angles extracted using QR factorization of the eigenvector matrices via a series of Givens rotations. We show that the resulting parameters are independently modifiable and are bounded so they can be naturally used in different kinds of stochastic global search algorithms. We also describe an algorithm for ordering the eigenvectors so that the parameters of individual Gaussian components are uniquely identifiable.

As our second major contribution, we propose an algorithm for ordering of the Gaussian components within a candidate solution for obtaining a unique correspondence between two candidate solutions during their interactions for parameter updates throughout the stochastic search. The correspondence identification problem is formulated as a minimum cost network flow optimization problem where the objective is to find the correspondence relation that minimizes the sum of Kullback–Leibler divergences between pairs of Gaussian components, one from each of the two candidate solutions. We illustrate the proposed parametrization and identifiability solutions using PSO for density estimation. An early version of this paper [13] presented initial experiments on clustering.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 establishes the notation and defines the estimation problem. Section 4 summarizes the EM approach for GMM estimation. Section 5 presents the details of the proposed covariance parametrization and the solution for the identifiability problem. Section 6 describes the PSO framework and its adaptation as a stochastic search algorithm for GMM estimation. Section 7 presents the experiments and discussion using both synthetic and real data sets. Finally, Section 8 provides the conclusions of the paper.

2. Related work

As discussed in the previous section, existing work on the use of stochastic search algorithms for GMM estimation typically uses only the means [7–10,12] or means and standard deviations alone [3,11] in the candidate solutions. Exceptions where both mean vectors and full covariance matrices were used include [4,5] where EM was used for the actual local optimization by fitting Gaussians to data in each iteration and GA was used only to guide the global search by selecting individual Gaussian components from existing candidate solutions in the reproduction steps. However, treating each Gaussian component as a whole in the search process and fitting it locally using the EM iterations may not explore the whole solution space effectively especially in higher dimensions. Another example is [6] where two GA alternatives for the estimation of multidimensional GMMs were proposed. The first alternative encoded the covariance matrices for d-dimensional data using $d+d^2$ elements where d values corresponded to the standard deviations and d^2 values represented a correlation matrix. The second alternative used d runs of a GA for estimating 1D GMMs followed by d runs of EM starting from the results of the GAs. Experiments using 3D synthetic data

showed that the former alternative was not successful and the latter performed better. The parametrization proposed in this paper allows the use of full covariance matrices in the GMM estimation.

The second main problem, identifiability, that we investigate in this paper is known as "label switching" in the statistics literature for the Bayesian estimation of mixture models using Markov chain Monte Carlo (MCMC) strategies. The label switching corresponds to the interchanging of the parameters of some of the mixture components and the invariance of the likelihood function as well as the posterior distribution for a prior that is symmetric in the components under such permutations [2]. Proposed solutions to label switching include artificial identifiability constraints that involve relabeling of the output of the MCMC sampler based on some component parameters (e.g., sorting of the components based on their means for 1D data) [2], deterministic relabeling algorithms that select a relabeling at each iteration that minimizes the posterior expectation of some loss function [14,15], and probabilistic relabeling algorithms that take into consideration the uncertainty in the relabeling that should be selected on each iteration of the MCMC output [16].

Even though the label switching problem also applies to the population-based stochastic search procedures, only a few pattern recognition studies (e.g., only [6,7] among the ones discussed above) mention its existence during GMM estimation. In particular, Tohka et al. [6] ensured that the components in a candidate solution were ordered based on their means in each iteration. This ordering was possible because 1D data were used in the experiments but such artificial identifiability constraints are not easy to establish for multivariate data. Since they have an influence on the resulting estimates, these constraints are also known to lead to over- or under-estimation [2] and create a bias [14]. Chang et al. [7] proposed a greedy solution that sorted the components of a candidate solution based on the distances of the mean vectors of that solution to the mean vectors of a reference solution that achieved the highest fitness value. However, such heuristic orderings depend on the ordering of the components of the reference solution that is also arbitrary and ambiguous. The method proposed in this paper can be considered as a deterministic relabeling algorithm according to the categorization of label switching solutions as discussed above. It allows controlled interaction of the candidate solutions by finding the optimal correspondences among their components, and enables more effective exploration of the solution space.

In addition to the population-based stochastic search techniques, alternative approaches to the basic EM algorithm also include methods for reducing the complexity of a GMM by trying to estimate the number of components [17,18] or by forcing a hierarchical structure [19,20]. This paper focuses on the conventional problem with a fixed number of components in the mixture. However, the above-mentioned techniques will also benefit from the contributions of this paper as it is still important to be able to find the best possible set of parameters for a given complexity because of the existing multiple local maxima problem. There are also other alternatives that use iterative simulation techniques such as Monte Carlo EM, imputation-posterior algorithm for data augmentation, and Markov chain Monte Carlo EM that define priors for the unknown parameters and replace the E and M steps by draws from conditional distributions computed using these priors [21]. Since these algorithms are not population-based methods and are generally used for more complicated mixture models rather than the standard GMMs, they are out of the scope of this paper. However, our proposed parametrization can also be used in these approaches by providing alternative choices for defining the priors.

3. Problem definition: GMM estimation

The paper uses the following notation. $\mathbb{R}$ denotes the set of real numbers, $\mathbb{R}_+$ denotes the set of nonnegative real numbers, $\mathbb{R}_{++}$ denotes the set of positive real numbers, $\mathbb{R}^d$ denotes the set of d-dimensional real vectors, and $\mathbb{S}^d_{++}$ denotes the set of symmetric positive definite $d \times d$ matrices. Vectors and matrices are denoted by lowercase and uppercase bold letters, respectively.

We consider a family of mixtures of K multivariate Gaussian distributions in $\mathbb{R}^d$ indexed by the set of parameters $\boldsymbol{\Theta} = \{\alpha_1,\ldots,\alpha_K,\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_K\}$. Each $\boldsymbol{\theta}_k = \{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k\}$ represents the parameters of the kth Gaussian distribution $p_k(\mathbf{x}|\boldsymbol{\theta}_k)$ such that $\boldsymbol{\mu}_k \in \mathbb{R}^d$ and $\boldsymbol{\Sigma}_k \in \mathbb{S}^d_{++}$ are the means and the covariance matrices, respectively, for $k=1,\ldots,K$. Mixing probabilities $\alpha_k \in [0,1]$ are constrained to sum up to 1, i.e., $\sum_{k=1}^K \alpha_k = 1$. Given a set of N data points $\mathcal{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$ where $\mathbf{x}_j \in \mathbb{R}^d$ are independent and identically distributed (i.i.d.) according to the mixture probability density function $p(\mathbf{x}|\boldsymbol{\Theta}) = \sum_{k=1}^K \alpha_k p_k(\mathbf{x}|\boldsymbol{\theta}_k)$, the objective is to obtain the maximum likelihood estimate $\hat{\boldsymbol{\Theta}}$ by finding the parameters that maximize the log-likelihood function

$$\log L(\boldsymbol{\Theta}|\mathcal{X}) = \log p(\mathcal{X}|\boldsymbol{\Theta}) = \sum_{j=1}^{N} \log\left(\sum_{k=1}^{K} \alpha_k p_k(\mathbf{x}_j|\boldsymbol{\theta}_k)\right). \quad (1)$$

Since the log-likelihood function typically has a complicated structure with multiple local maxima, an analytical solution for $\hat{\boldsymbol{\Theta}}$ that corresponds to the global maximum of (1) cannot be obtained by simply setting the derivatives of $\log L(\boldsymbol{\Theta}|\mathcal{X})$ to zero. The common practice for reaching a local maximum of the log-likelihood function is to use the expectation–maximization (EM) algorithm that iteratively updates the parameters of individual Gaussian distributions in the mixture. For completeness and to set up the notation for the rest of the paper, we briefly present the EM algorithm in the next section. The proposed solution to the maximum likelihood estimation problem is described in the following section.

4. GMM estimation using expectation–maximization

In this section we present a review of the EM algorithm and its application to GMM estimation. Details of this review can be found in [1,2]. Since the log-likelihood in (1) is not a concave function, gradient descent-based algorithms typically converge to a local optimum. One of the commonly used techniques for efficient search of a local optimum is provided by the EM algorithm. In the EM approach to the GMM estimation problem, the given data, $\mathcal{X}$, is considered as incomplete data, and a set of N latent variables $\mathcal{Y} = \{y_1,\ldots,y_N\}$ are defined where each y_j indicates which Gaussian component generated the data vector $\mathbf{x}_j$. In other words, $y_j = k$ if the jth data vector was generated by the kth mixture component. Instead of the log-likelihood function, the EM algorithm maximizes an auxiliary function $Q(\boldsymbol{\Theta},\boldsymbol{\Phi})$. $Q(\boldsymbol{\Theta},\boldsymbol{\Phi})$ is a function of both the parameters $\boldsymbol{\Theta}$ and the assignments $\boldsymbol{\Phi} = \{w_{jk}\}$ of the data vectors to the Gaussian components for $j=1,\ldots,N$ and $k=1,\ldots,K$.

This auxiliary function

$$Q(\boldsymbol{\Theta},\boldsymbol{\Phi}) = \sum_{j=1}^{N}\sum_{k=1}^{K} w_{jk}\log(\alpha_k p_k(\mathbf{x}_j|\boldsymbol{\theta}_k)) - \sum_{j=1}^{N}\sum_{k=1}^{K} w_{jk}\log(w_{jk}) \quad (2)$$

is a lower bound to the log-likelihood function for any parameters $\boldsymbol{\Theta}$ and assignments $\boldsymbol{\Phi}$, i.e., $\log L(\boldsymbol{\Theta}|\mathcal{X}) \geq Q(\boldsymbol{\Theta},\boldsymbol{\Phi})$. When $Q(\boldsymbol{\Theta},\boldsymbol{\Phi})$ is maximized over assignments $\boldsymbol{\Phi}$ that are set to be the posterior probabilities $\tilde{\boldsymbol{\Phi}}$ where $w_{jk} = P(y_j = k|\mathbf{x}_j,\boldsymbol{\Theta})$, it has the same value as the log-likelihood function, i.e., $\log L(\boldsymbol{\Theta}|\mathcal{X}) = Q(\boldsymbol{\Theta},\tilde{\boldsymbol{\Phi}})$.

On the other hand, when $Q(\boldsymbol{\Theta},\boldsymbol{\Phi})$ is maximized over parameters $\tilde{\boldsymbol{\Theta}}$, we have $Q(\tilde{\boldsymbol{\Theta}},\boldsymbol{\Phi}) \geq Q(\boldsymbol{\Theta},\boldsymbol{\Phi})$.

The GMM-EM algorithm is based on these two properties of the Q function. Starting from a set of initial parameters, the algorithm finds a local maximum for the log-likelihood function by alternatingly maximizing the Q function over the assignments $\boldsymbol{\Phi}$ and the parameters $\boldsymbol{\Theta}$. Maximization over the assignments is called the expectation step as the assignments

$$w_{jk}^{(t)} = P(y_j = k|\mathbf{x}_j,\boldsymbol{\Theta}^{(t)}) = \frac{\alpha_k^{(t)} p_k(\mathbf{x}_j|\boldsymbol{\theta}_k^{(t)})}{\sum_{i=1}^{K} \alpha_i^{(t)} p_i(\mathbf{x}_j|\boldsymbol{\theta}_i^{(t)})} \tag{3}$$

make the log-likelihood function, that is also referred to as the incomplete likelihood, equal to the expected complete likelihood. Maximization of the Q function over the parameters is referred to as the maximization step, and results in the parameter estimates

$$\hat{\alpha}_k^{(t+1)} = \frac{1}{N}\sum_{j=1}^{N} w_{jk}^{(t)} \tag{4}$$

$$\hat{\boldsymbol{\mu}}_k^{(t+1)} = \frac{\sum_{j=1}^{N} w_{jk}^{(t)}\mathbf{x}_j}{\sum_{j=1}^{N} w_{jk}^{(t)}} \tag{5}$$

$$\hat{\boldsymbol{\Sigma}}_k^{(t+1)} = \frac{\sum_{j=1}^{N} w_{jk}^{(t)}(\mathbf{x}_j-\hat{\boldsymbol{\mu}}_k^{(t+1)})(\mathbf{x}_j-\hat{\boldsymbol{\mu}}_k^{(t+1)})^T}{\sum_{j=1}^{N} w_{jk}^{(t)}} \tag{6}$$

where t indicates the iteration number.

5. GMM estimation using stochastic search

Since the EM algorithm converges to a local optimum, in its application to the GMM parameter estimation problem, the common practice is to use multiple random initializations to find different local maxima, and to use the result corresponding to the highest log-likelihood value. As discussed in Section 1, an alternative is to use population-based stochastic search algorithms where different candidate solutions are allowed to interact with each other. However, the continuation of the iterations that search for better candidate solutions assume that the parameters remain valid both in terms of the requirements of the GMM and with respect to the bounds enforced by the data. The validity and boundedness of the mean vectors are relatively easy to implement but direct use of covariance matrices introduce problems. For example, one might consider to use $d(d+1)/2$ potentially different entries of a real symmetric $d \times d$ covariance matrix as a direct parametrization of the covariance matrix. Although this ensures the symmetry property, it cannot guarantee the positive definiteness where arbitrary modifications of these entries may produce non-positive definite matrices. This is illustrated in Table 1 where a new covariance matrix is constructed from three valid covariance matrices in a simple arithmetic operation. Even though the input matrices are positive definite, the output matrix is often not positive definite for increasing dimensions. Another possible parametrization is to use Cholesky factorization but the resulting parameters are unbounded (real numbers in the $(-\infty,\infty)$ range). Therefore, lack of a suitable parametrization for arbitrary covariance matrices has limited the flexibility of the existing approaches in modeling the covariance structure of the components in the mixture.

In this section, first, we propose a novel parametrization where the parameters of an arbitrary covariance matrix are independently modifiable and can have upper and lower bounds. We also describe an algorithm for unique identification of these parameters from a valid covariance matrix. Then, we describe a new solution to the mixture identifiability problem where different orderings of the Gaussian components in different candidate solutions can significantly affect the convergence of the search procedure. The proposed approach solves this issue by using a two-stage interaction between the candidate solutions. In the first stage, the optimal correspondences among the components of two candidate solutions are identified. Once these correspondences are identified, in the second stage, desirable interactions such as selection, swapping, addition, and perturbation can be performed. Both the proposed parametrization and the solutions for the two identifiability problems allow effective use of population-based stochastic search algorithms for the estimation of GMMs.

5.1. Covariance parametrization

The proposed covariance parametrization is based on eigenvalue decomposition of the covariance matrix. For a given d-dimensional covariance matrix $\boldsymbol{\Sigma} \in \mathbb{S}_{++}^{d}$, let $\{\lambda_i,\boldsymbol{v}_i\}$ for $i=1,\ldots,d$ denote the eigenvalue–eigenvector pairs in a particular order where $\lambda_i \in \mathbb{R}_{++}$ for $i=1,\ldots,d$ correspond to the eigenvalues and $\boldsymbol{v}_i \in \mathbb{R}^d$ such that $\|\boldsymbol{v}_i\|_2 = 1$ and $\boldsymbol{v}_i^T\boldsymbol{v}_j = 0$ for $i \neq j$ represent the eigenvectors. A given d-dimensional covariance matrix $\boldsymbol{\Sigma}$ can be written in terms of its eigenvalue–eigenvector pairs as $\boldsymbol{\Sigma} = \sum_{i=1}^{d} \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^T$. Let the diagonal matrix $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1,\ldots,\lambda_d)$ denote the eigenvalue matrix, and the unitary matrix $\mathbf{V} = (\boldsymbol{v}_1,\ldots,\boldsymbol{v}_d)$ denote the corresponding eigenvector matrix where the normalized eigenvectors are placed into the columns of $\mathbf{V}$ in the order determined by the corresponding eigenvalues in $\boldsymbol{\Lambda}$. Then, the given covariance matrix can be written as $\boldsymbol{\Sigma} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$.

Due to its symmetric structure, an arbitrary d-dimensional covariance matrix has $d(d+1)/2$ degrees of freedom; thus, at most $d(d+1)/2$ parameters are needed to represent this matrix. The proposed parametrization is based on the following theorem.

Theorem 1. *An arbitrary covariance matrix with $d(d+1)/2$ degrees of freedom can be parametrized using d eigenvalues in a particular order and $d(d-1)/2$ Givens rotation matrix angles $\phi^{pq} \in [-\pi/4, 3\pi/4]$ for $1 \leq p < q \leq d$ computed from the eigenvector matrix whose columns store the eigenvectors in the same order as the corresponding eigenvalues.*

The proof is based on the following definition, proposition, and lemma. An example parametrization for a 3×3 covariance matrix is given in Fig. 1.

Definition 1. A Givens rotation matrix $\mathbf{G}(p,q,\phi^{pq})$ with three input parameters corresponding to two indices p and q that satisfy $p < q$,

Table 1
Simulation of the construction of a covariance matrix from three existing covariance matrices. Given the input matrices Σ_1, Σ_2, and Σ_3, a new matrix is constructed as $\Sigma_{\text{new}} = \Sigma_1 + (\Sigma_2 - \Sigma_3)$ in an arithmetic operation that is often found in many stochastic search algorithms. This operation is repeated for 100,000 times for different input matrices at each dimensionality reported in the first row. As shown in the second row, the number of Σ_{new} that is positive definite, i.e., a valid covariance matrix, decreases significantly at increasing dimensions. This shows that the entries in the covariance matrix cannot be directly used as parameters in stochastic search algorithms.

Dimension	3	5	10	15	20	30
# valid	44,652	27,443	2882	103	1	0

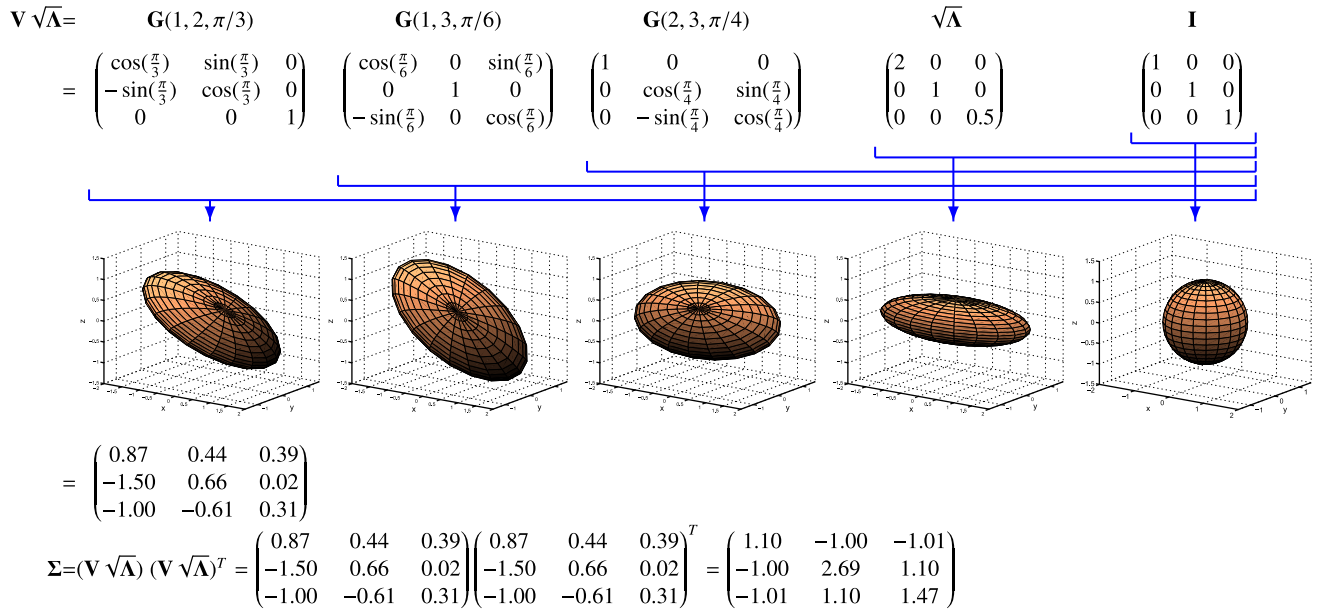

Fig. 1. Example parametrization for a 3×3 covariance matrix. The example matrix can be parametrized using $\{\lambda_1,\lambda_2,\lambda_3,\phi^{12},\phi^{13},\phi^{23}\} = \{4, 1, 0.25, \pi/3, \pi/6, \pi/4\}$. The ellipses from right to left show the covariance structure resulting from each step of premultiplication of the result of the previous step, starting from the identity matrix.

and an angle ϕ^{pq} has the form

$$\mathbf{G}(p,q,\phi^{pq}) = \begin{pmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & \cos(\phi^{pq}) & \cdots & \sin(\phi^{pq}) & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & & -\sin(\phi^{pq}) & \cdots & \cos(\phi^{pq}) & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{pmatrix}. \quad (7)$$

Premultiplication by $\mathbf{G}(p,q,\phi^{pq})^T$ corresponds to a counterclockwise rotation of ϕ radians in the plane spanned by two coordinate axes indexed by p and q [22].

Proposition 1. *A Givens rotation can be used to zero a particular entry in a vector. Given scalars a and b, the $c = \cos(\phi)$ and $s = \sin(\phi)$ values in (7) that can zero b can be computed as the solution of*

$$\begin{pmatrix} c & s \\ -s & c \end{pmatrix}^T \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} h \\ 0 \end{pmatrix} \quad (8)$$

using the following algorithm [22]

if $b = 0$ ***then***
  $c = 1;\ s = 0$
else
  if $|b| > |a|$ ***then***
    $\tau = -a/b;\ s = 1/\sqrt{1+\tau^2};\ c = s\tau$
  else
    $\tau = -b/a;\ c = 1/\sqrt{1+\tau^2};\ s = c\tau$
  end if
end if

where ϕ can be computed as $\phi = \arctan(s/c)$. The resulting Givens rotation angle ϕ is in the range $[-\pi/4, 3\pi/4]$ by definition (because of the absolute values in the algorithm).

Lemma 1. *An eigenvector matrix $\mathbf{V}$ of size $d \times d$ can be written as a product of $d(d-1)/2$ Givens rotation matrices whose angles lie in the interval $[-\pi/4, 3\pi/4]$ and a diagonal matrix whose entries are either $+1$ or -1.*

Proof. Existence of such a decomposition can be shown by using QR factorization via a series of Givens rotations. QR factorization decomposes any real square matrix into a product of an orthogonal matrix $\mathbf{Q}$ and an upper triangular matrix $\mathbf{R}$, and can be computed by using Givens rotations where each rotation zeros an element below the diagonal of the input matrix. When the QR algorithm is applied to $\mathbf{V}$, the angle ϕ^{pq} for the given indices p and q is calculated using the values $\mathbf{V}(p,p)$ and $\mathbf{V}(q,p)$ as the scalars a and b, respectively, in Definition 1, and then, $\mathbf{V}$ is premultiplied with the transpose of the Givens rotation matrix as $\mathbf{G}(p,q,\phi^{pq})^T\mathbf{V}$ where $\mathbf{G}$ is defined in Definition 1. This multiplication zeros the value $\mathbf{V}(q,p)$. This process is continued for $p = 1,\ldots,d-1$ and $q = p+1,\ldots,d$, resulting in the orthogonal matrix

$$\mathbf{Q} = \prod_{p=1}^{d-1} \prod_{q=p+1}^{d} \mathbf{G}(p,q,\phi^{pq}) \quad (9)$$

and the triangular matrix

$$\mathbf{R} = \mathbf{Q}^T\mathbf{V}. \quad (10)$$

Since the eigenvector matrix $\mathbf{V}$ is orthogonal, i.e., $\mathbf{V}^T\mathbf{V} = \mathbf{I}$, $\mathbf{R}^T\mathbf{Q}^T\mathbf{Q}\mathbf{R} = \mathbf{I}$ leads to $\mathbf{R}^T\mathbf{R} = \mathbf{I}$ because $\mathbf{Q}$ is also orthogonal. Since $\mathbf{R}$ should be both orthogonal and upper triangular, we conclude that $\mathbf{R}$ is a diagonal matrix whose entries are either $+1$ or -1. □

Proof of Theorem 1. Following Lemma 1, an eigenvector matrix $\mathbf{V}$ in which the eigenvectors are stored in a particular order can be written using $d(d-1)/2$ angle parameters for the $\mathbf{Q}$ matrix and an additional d parameters for the $\mathbf{R}$ matrix. However, since both $\boldsymbol{v}_i$ and $-\boldsymbol{v}_i$ are valid eigenvectors ($\boldsymbol{\Sigma}\boldsymbol{v}_i = \lambda_i\boldsymbol{v}_i$ and $\boldsymbol{\Sigma}(-\boldsymbol{v}_i) = \lambda_i(-\boldsymbol{v}_i)$), we can show that those additional d parameters for the diagonal of $\mathbf{R}$ are not required for the parametrization of eigenvector matrices.

This follows from the invariance of the Givens rotation angles to the rotation of the eigenvectors with π radians such that when any column of the $\mathbf{V}$ matrix is multiplied by -1, only the $\mathbf{R}$ matrix changes, while the $\mathbf{Q}$ matrix, hence the Givens rotation angles, do not change. To prove this invariance, let $\mathcal{P} = \{\mathbf{P} | \mathbf{P} \in \mathbb{R}^{d\times d}, \mathbf{P}(i,j) = 0, \forall i \neq j,$ and $\mathbf{P}(i,i) \in \{+1,-1\}$ for $i = 1,\ldots,d\}$ be a set of modification

matrices. For a given $\mathbf{P}\in\mathcal{P}$, define $\hat{\mathbf{V}}=\mathbf{VP}$. Since $\mathbf{V}=\mathbf{QR}$, we have $\hat{\mathbf{V}}=\mathbf{QRP}$. Then, defining $\hat{\mathbf{R}}=\mathbf{RP}$ gives $\hat{\mathbf{V}}=\mathbf{Q}\hat{\mathbf{R}}$. Since $\mathbf{Q}$ did not change and $\hat{\mathbf{R}}=\mathbf{RP}$ is still a diagonal matrix whose entries are either $+1$ or -1, it is a valid QR factorization. Therefore, we can conclude that there exists a QR factorization $\hat{\mathbf{V}}=\mathbf{Q}\hat{\mathbf{R}}$ with the same $\mathbf{Q}$ matrix as the QR factorization $\mathbf{V}=\mathbf{QR}$.

The discussion above shows that the $d(d-1)/2$ Givens rotation angles are sufficient for the parametrization of the eigenvectors because the multiplication of any eigenvector by -1 leads to the same covariance matrix $\mathbf{\Sigma}$, i.e.,

$$\begin{aligned}\mathbf{\Sigma}&=\sum_{i=1,\ i\neq j}^{d}\lambda_i\mathbf{v}_i\mathbf{v}_i^T+\lambda_j(-\mathbf{v}_j)(-\mathbf{v}_j)^T\\&=\sum_{i=1,\ i\neq j}^{d}\lambda_i\mathbf{v}_i\mathbf{v}_i^T+\lambda_j(\mathbf{v}_j)(\mathbf{v}_j)^T\\&=\sum_{i=1}^{d}\lambda_i\mathbf{v}_i\mathbf{v}_i^T\end{aligned}\tag{11}$$

Finally, together with the d eigenvalues, the covariance matrix can be constructed as $\mathbf{\Sigma}=\mathbf{V\Lambda V}^T$. □

5.2. Identifiability of individual Gaussians

Theorem 1 assumes that the eigenvalue–eigenvector pairs are given in a particular order. However, since any d-dimensional covariance matrix $\mathbf{\Sigma}\in\mathbb{S}_{++}^{d}$ can be written as $\mathbf{\Sigma}=\sum_{i=1}^{d}\lambda_i\mathbf{v}_i\mathbf{v}_i^T$ and there is no inherent ordering of the eigenvalue–eigenvector pairs, it is possible to write this summation in terms of $d!$ different eigenvalue and eigenvector matrices as $\mathbf{\Sigma}=\mathbf{V\Lambda V}^T$ simply by changing the order of the eigenvalues and their corresponding eigenvectors in the eigendecomposition matrices $\mathbf{\Lambda}$ and $\mathbf{V}$. For example, all equivalent parametrizations of the example covariance matrix in Fig. 1 are given in Table 2. Furthermore, multiplying any column of the eigenvector matrix by -1 still gives a valid eigenvector matrix, resulting in 2^d possibilities. Since we showed that there exists a unique $\mathbf{Q}$ matrix and a corresponding set of unique Givens rotation angles can be extracted via QR factorization in the proof of Theorem 1, the result is invariant to these 2^d possibilities. However, for an improved efficiency in the global search, it is one of our goals to pair the parameters between alternate solution candidates before performing any interactions among them. Therefore, the dependence of the results on the $d!$ orderings and the resulting equivalence classes still need to be eliminated.

Table 2
To demonstrate its non-uniqueness, all equivalent parametrizations of the example covariance matrix given in Fig. 1 for different orderings of the eigenvalue–eigenvector pairs. The angles are given in degrees. The parameters in the first row are used in Fig. 1.

λ_1	λ_2	λ_3	ϕ^{12}	ϕ^{13}	ϕ^{23}
4	1	0.25	60.00	30.00	45.00
4	0.25	1	60.00	30.00	−45.00
1	4	0.25	123.43	−37.76	39.23
1	0.25	4	123.43	−37.76	129.23
0.25	4	1	−3.43	−37.76	−39.23
0.25	1	4	−3.43	−37.76	50.77

In order to have unique eigenvalue decomposition matrices, we propose an ordering algorithm based on the eigenvectors so that from a given covariance matrix we can obtain uniquely ordered eigenvalue and eigenvector matrices, leading to a unique set of eigenvalues and Givens rotation angles as the parameters. The ordering algorithm uses only the eigenvectors and not the eigenvalues because the eigenvectors correspond to the principal directions of the data whereas the eigenvalues indicate the amount of the extent of the data along these directions. The dependency of the results on the $d!$ orderings can be eliminated by aligning the principal directions of the covariance matrices so that a unique set of angle parameters with similar values for similarly aligned matrices can be obtained. Fig. 2 illustrates two different orderings based on eigenvectors and eigenvalues.

The proposed eigenvalue–eigenvector ordering algorithm uses an orthogonal basis matrix as a reference. In this greedy selection algorithm,the eigenvector among the unselected ones having the maximum absolute inner product with the ith reference vector is put into the ith column in the output matrix. Let $\mathcal{S}^{\text{in}}=\{\{\lambda_1^{\text{in}},\mathbf{v}_1^{\text{in}}\},\ldots,\{\lambda_d^{\text{in}},\mathbf{v}_d^{\text{in}}\}\}$ denote the input eigenvalue–eigenvector pair set, $\mathbf{V}^{\text{ref}}=(\mathbf{v}_1^{\text{ref}},\ldots,\mathbf{v}_d^{\text{ref}})$ denote the reference orthogonal basis matrix, $\mathbf{\Lambda}^{\text{out}}=\text{diag}(\lambda_1^{\text{out}},\ldots,\lambda_d^{\text{out}})$ and $\mathbf{V}^{\text{out}}=(\mathbf{v}_1^{\text{out}},\ldots,\mathbf{v}_d^{\text{out}})$ denote the final output eigenvalue and eigenvector matrices,and $\mathcal{I}$ be the set of indices of the remaining eigenvalue–eigenvector pairs that need to be ordered. The ordering algorithm is defined in Algorithm 1.

$$\mathbf{\Sigma}_1=\begin{pmatrix}1.56&-0.36&-0.70\\-0.36&2.56&0.35\\-0.70&0.35&1.88\end{pmatrix}\Rightarrow\mathbf{\Lambda}_1=\begin{pmatrix}3&0&0\\0&2&0\\0&0&1\end{pmatrix},\quad\mathbf{V}_1=\begin{pmatrix}0.43&0.44&0.79\\-0.75&0.66&0.05\\-0.50&-0.61&0.62\end{pmatrix}\Rightarrow\{\phi_1^{12},\phi_1^{13},\phi_1^{23}\}=\{60.00,30.00,45.00\}$$

$$\mathbf{\Sigma}_2=\begin{pmatrix}1.11&0.79&-1.21\\0.79&2.18&-1.24\\-1.21&-1.24&1.97\end{pmatrix}\Rightarrow\mathbf{\Lambda}_2=\begin{pmatrix}1&0&0\\0&4&0\\0&0&0.25\end{pmatrix},\ \mathbf{V}_2=\begin{pmatrix}0.42&0.44&0.80\\-0.78&0.63&0.06\\-0.47&-0.64&0.60\end{pmatrix}\Rightarrow\{\phi_2^{12},\phi_2^{13},\phi_2^{23}\}=\{61.80,28.20,46.80\}$$

$$\mathbf{\Sigma}_3=\begin{pmatrix}1.11&0.79&-1.21\\0.79&2.18&-1.24\\-1.21&-1.24&1.97\end{pmatrix}\Rightarrow\mathbf{\Lambda}_3=\begin{pmatrix}4&0&0\\0&1&0\\0&0&0.25\end{pmatrix},\ \mathbf{V}_3=\begin{pmatrix}0.44&0.42&0.80\\0.63&-0.78&0.06\\-0.64&-0.47&0.60\end{pmatrix}\Rightarrow\{\phi_3^{12},\phi_3^{13},\phi_3^{23}\}=\{125.09,-39.97,38.07\}$$

Fig. 2. Parametrization of 3×3 covariance matrices by using different orderings of the eigenvectors. Eigendecomposition matrices $\mathbf{\Lambda}_i$ and $\mathbf{V}_i$, and the Givens angles extracted from $\mathbf{V}_i$ as $\{\phi_i^{12},\phi_i^{13},\phi_i^{23}\}$ are given for three cases, $i=1,2,3$. The eigenvectors in $\mathbf{V}_2$ are ordered according to the eigenvectors of $\mathbf{V}_1$ by using the algorithm proposed in this paper, and the eigenvectors in $\mathbf{V}_3$ are ordered in descending order of the eigenvalues in $\mathbf{\Lambda}_3$. The resulting angles $\{\phi_2^{12},\phi_2^{13},\phi_2^{23}\}$ are very similar to $\{\phi_1^{12},\phi_1^{13},\phi_1^{23}\}$, reflecting the similarity of the principal directions in $\mathbf{V}_1$ and $\mathbf{V}_2$, and enabling the interactions to be aware of the similarity between $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$. However, the angles $\{\phi_3^{12},\phi_3^{13},\phi_3^{23}\}$ do not show any indication of this similarity, and interactions between $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_3$ will be very different even though the matrices $\mathbf{\Sigma}_2$ and $\mathbf{\Sigma}_3$ are identical.

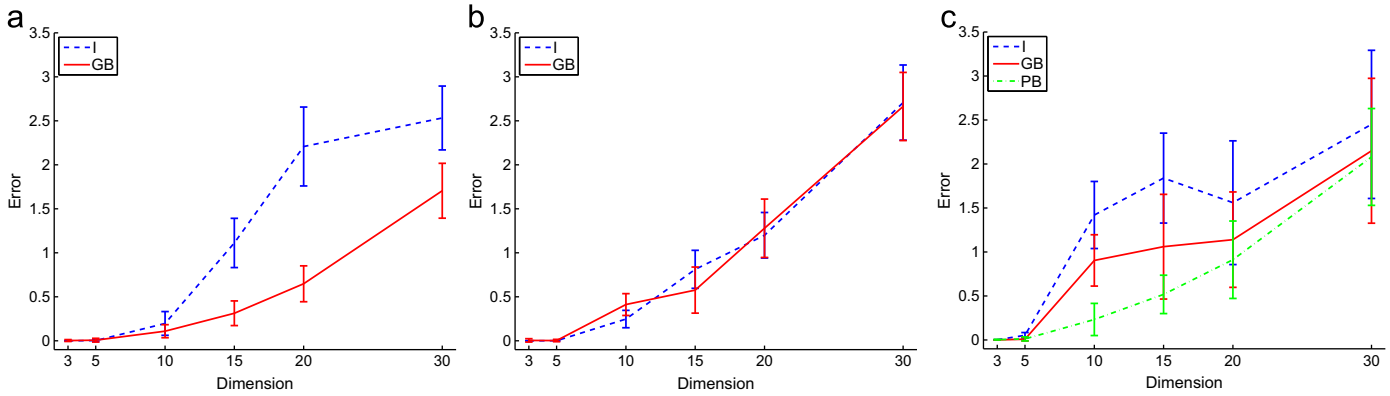

Fig. 3. Average error in log-likelihood and its standard deviation (shown as error bars at one standard deviation) in 1000 trials for different choices of reference matrices in eigenvector ordering during the estimation of the covariance matrix of a single Gaussian using stochastic search. Choices for the reference matrix are I: identity matrix, GB: the eigenvector matrix corresponding to the global best solution, and PB: the eigenvector matrix corresponding to the personal best solution. (a) GA, (b) DE and (c) PSO.

Algorithm 1. Eigenvector ordering algorithm.

Input: $\mathcal{S}^{\text{in}}$, $\mathbf{V}^{\text{ref}}$, $\mathcal{I}=\{1,\ldots,d\}$
Output: $\mathbf{\Lambda}^{\text{out}}$, $\mathbf{V}^{\text{out}}$
1: **for** $i=1$ to d **do**
2: $i^*=\arg\max_{j\in\mathcal{I}}|(\boldsymbol{v}_j^{\text{in}})^T(\boldsymbol{v}_i^{\text{ref}})|$
3: $\lambda_i^{\text{out}}\leftarrow\lambda_{i^*}^{\text{in}}$
4: $\boldsymbol{v}_i^{\text{out}}\leftarrow\boldsymbol{v}_{i^*}^{\text{in}}$
5: $\mathcal{I}\leftarrow\mathcal{I}-\{i^*\}$
6: **end for**

Any reference basis matrix $\mathbf{V}^{\text{ref}}$ in Algorithm 1 will eliminate the dependency on the $d!$ orderings, and will result in a unique set of parameters. However, the choice of $\mathbf{V}^{\text{ref}}$ can affect the convergence of the likelihood during estimation. We performed simulations to determine the most effective reference matrix $\mathbf{V}^{\text{ref}}$ for eigenvector ordering. The maximum likelihood estimation problem in Section 3 was set up to estimate the covariance matrix of a single Gaussian as follows. Given a set of N data points $\mathcal{X}=\{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$ where each $\mathbf{x}_j\in\mathbb{R}^d$ is independent and identically distributed according to a Gaussian with zero mean and covariance matrix $\mathbf{\Sigma}$, the log-likelihood function

$$\log L(\mathbf{\Sigma}|\mathcal{X})=-\frac{Nd}{2}\log(2\pi)-\frac{N}{2}\log(|\Sigma|)-\frac{1}{2}\sum_{j=1}^{N}\mathbf{x}_i^T\Sigma^{-1}\mathbf{x}_i \tag{12}$$

can be rewritten as

$$\log L(\mathbf{\Sigma}|\mathcal{X})=-\frac{Nd}{2}\log(2\pi)-\frac{N}{2}\log(|\Sigma|)-\frac{N}{2}\text{tr}(\Sigma^{-1}\mathbf{X}) \tag{13}$$

where $\mathbf{X}=(1/N)\sum_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i^T$. Thus, the maximum likelihood estimate of $\mathbf{\Sigma}$ can be found as the one that maximizes $\log(|\Sigma^{-1}|)-\text{tr}(\Sigma^{-1}\mathbf{X})$. We solved this maximization problem using GA, DE, and PSO implemented as in [6,23,24], respectively. For GA and DE, candidate reference matrices were the identity matrix and the eigenvector matrix corresponding to the global best solution. For PSO, candidate reference matrices were the identity matrix, the eigenvector matrix corresponding to each particle's personal best, and the eigenvector matrix corresponding to the global best particle. For each case, 100 different target Gaussians ($\mathbf{X}$ in (13)) were randomly generated by sampling the eigenvalues from the uniform distribution Uniform[0.1,1.0] and the Givens rotation angles from the uniform distribution Uniform$[-\pi/4,3\pi/4]$. This was repeated for dimensions $d\in\{3,5,10,15,20,30\}$, and the respective optimization algorithm was used to find the corresponding covariance matrix ($\mathbf{\Sigma}$ in (13)) that maximized the log-likelihood using 10 different initializations. Fig. 3 shows the plots of estimation errors resulting from the 1000 trials. The error was computed as the difference between the target log-likelihood computed from the true Gaussian parameters ($\mathbf{\Sigma}=\mathbf{X}$) and the resulting log-likelihood computed from the estimated Gaussian parameters. Based on these results, we can conclude that the eigenvector matrix corresponding to the personal best solution for PSO, and the eigenvector matrix corresponding to the global best solution for GA and DE (no personal best is available in GA and DE) can be used as the reference matrix in the eigenvector ordering algorithm.

Summary: The discussion above demonstrated that a d-dimensional covariance matrix $\mathbf{\Sigma}\in\mathbb{S}_{++}^d$ can be parametrized using d eigenvalues $\lambda_i\in\mathbb{R}_{++}$ for $i=1,\ldots,d$ and $d(d-1)/2$ angles $\phi^{pq}\in[-\pi/4,3\pi/4]$ for $1\le p<q\le d$. We showed that, for a given covariance matrix, these parameters can be uniquely extracted using eigenvalue decomposition, the proposed eigenvector ordering algorithm that aligns the principal axes of the covariance ellipsoids among alternate candidate solutions, and QR factorization using the Givens rotations method. We also showed that, given these parameters, a covariance matrix can be generated from the eigenvalue matrix $\mathbf{\Lambda}=\text{diag}(\lambda_1,\ldots,\lambda_d)$ and the eigenvector matrix $\mathbf{V}=\prod_{p=1}^{d-1}\prod_{q=p+1}^{d}\mathbf{G}(p,q,\phi^{pq})\mathbf{R}$ where $\mathbf{R}=\mathbf{I}$ as $\mathbf{\Sigma}=\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$.

5.3. Identifiability of Gaussian mixtures

Similar to the problem of ordering of the parameters within individual Gaussian components to obtain a unique set of parameters as discussed in the previous section, ordering of the Gaussian components within a candidate solution is important for obtaining a unique correspondence between two candidate solutions during their interactions for parameter updates throughout the stochastic search. The correspondence identifiability problem that arises from the equivalency of $K!$ possible orderings of individual components in a candidate solution for a mixture of K Gaussians affects the convergence of the search procedure. First of all, when the likelihood function has a mode under a particular ordering of the components, there exists $K!$ symmetric modes corresponding to all parameter sets that are in the same equivalence class formed by the permutation of these components. When these equivalencies are not known, a search algorithm may not cover the solution space effectively as equivalent configurations of components may be repeatedly explored. In a related problem, in the extreme case, a reproduction operation applied to two candidate solutions that are essentially equal may result in a new solution that is completely different from its

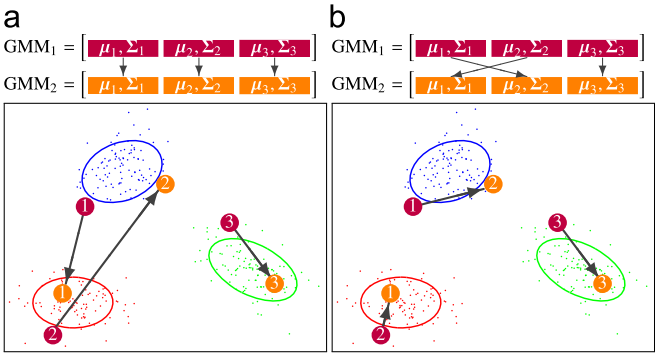

Fig. 4. Example correspondence relations for two GMMs with three components. The ellipses represent the true components corresponding to the colored sample points. The numbered blobs represent the locations of the components in the candidate solutions. When the parameter updates are performed according to the component pairs in the default order, some of the components may be updated based on interactions with components in different parts of the data space. However, using the reference matching procedure, a more desirable correspondence relation can be found enabling faster convergence. (a) Default correspondence relation and (b) Desired correspondence relation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

parents. Secondly, the knowledge of the correspondences helps performing the update operations as intended. For example, even for two candidate solutions that are not in the same equivalence class, matching of their components enables effective use of both direct interactions and cross interactions. For instance, cross interactions may be useful to increase diversity; on the other hand, direct interactions may be more helpful to find local minima. Without such matching of the components, these interactions cannot be controlled as desired, and the iterations proceed with arbitrary exploration of the search space. Fig. 4 shows examples for default and desired correspondence relations for two GMMs with three components.

We propose a matching algorithm for finding the correct correspondence relation between the components of two GMMs to enable interactions between the corresponding components in different solution candidates. In the following, the correspondence identification problem is formulated as a minimum cost network flow optimization problem. Although there are other alternative distance measures that can be used for this purpose, the objective is set to find the correspondence relation that minimizes the sum of Kullback–Leibler (KL) divergences between pairs of Gaussian components. For two Gaussians $g_1(\mathbf{x}|\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1)$ and $g_2(\mathbf{x}|\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2)$, the KL divergence has the closed form expression

$$D(g_1\|g_2)=\frac{1}{2}\left(\log\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|}+\mathrm{tr}(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1)-d+(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^T\boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)\right). \tag{14}$$

Consequently, given a target GMM with parameters $\{\{\boldsymbol{\mu}_1^{\mathrm{tar}},\boldsymbol{\Sigma}_1^{\mathrm{tar}}\},\ldots,\{\boldsymbol{\mu}_K^{\mathrm{tar}},\boldsymbol{\Sigma}_K^{\mathrm{tar}}\}\}$ and a reference GMM with parameters $\{\{\boldsymbol{\mu}_1^{\mathrm{ref}},\boldsymbol{\Sigma}_1^{\mathrm{ref}}\},\ldots,\{\boldsymbol{\mu}_K^{\mathrm{ref}},\boldsymbol{\Sigma}_K^{\mathrm{ref}}\}\}$, the cost of matching the ith component of the first GMM to the jth component of the second GMM is computed as

$$c_{ij}=\log\frac{|\boldsymbol{\Sigma}_j^{\mathrm{ref}}|}{|\boldsymbol{\Sigma}_i^{\mathrm{tar}}|}+\mathrm{tr}((\boldsymbol{\Sigma}_j^{\mathrm{ref}})^{-1}\boldsymbol{\Sigma}_i^{\mathrm{tar}})+(\boldsymbol{\mu}_i^{\mathrm{tar}}-\boldsymbol{\mu}_j^{\mathrm{ref}})^T(\boldsymbol{\Sigma}_j^{\mathrm{ref}})^{-1}(\boldsymbol{\mu}_i^{\mathrm{tar}}-\boldsymbol{\mu}_j^{\mathrm{ref}}) \tag{15}$$

and the correspondences are found by solving the following optimization problem:

$$\underset{I_{11},\ldots,I_{KK}}{\text{minimize}} \quad \sum_{i=1}^{K}\sum_{j=1}^{K}c_{ij}I_{ij}$$

$$\text{subject to} \quad \sum_{i=1}^{K}I_{ij}=1, \quad \forall j\in\{1,\ldots,K\}$$

$$\sum_{j=1}^{K}I_{ij}=1, \quad \forall i\in\{1,\ldots,K\}$$

$$I_{ij}=\begin{cases}1 & \text{correspondence between } i\text{th and } j\text{th components}\\ 0 & \text{otherwise}\end{cases} \tag{16}$$

In this formulation, the first and third constraints force each component of the first GMM to be matched with only one component of the second GMM, and the second constraint makes sure that only one component of the first GMM is matched to each component of the second GMM. This optimization problem can be solved very efficiently using the Edmonds–Karp algorithm [25]. Note that the solution of the optimization problem in (16) does not change under any permutation of the component labels in the target and reference GMMs. Fig. 5 illustrates the optimization formulation for the example in Fig. 4. Once the correspondences are established, the parameter updates can be performed as intended.

We performed simulations to evaluate the effectiveness of correspondence identification using the proposed matching algorithm. We ran the stochastic search algorithms GA, DE, and PSO for maximum likelihood estimation of GMMs that were synthetically generated as follows. The mixture weights were sampled from a uniform distribution such that the ratio of the largest weight to the smallest weight was at most 1.3 and all weights summed up to 1. The mean vectors were sampled from the uniform distribution Uniform$[0,1]^d$ where d was the number of dimensions. The covariance matrices were generated by sampling the eigenvalues from the uniform distribution Uniform[1,1.6] and the Givens rotation angles from the uniform distribution Uniform $[-\pi/4,3\pi/4]$. The minimum separation between the components in the mixture was controlled with a parameter called c. Two Gaussians are defined to be c-separated if

$$\|\boldsymbol{\mu}_1-\boldsymbol{\mu}_2\|_2\geq c\sqrt{d\,\max\{\lambda_{\max}(\boldsymbol{\Sigma}_1),\lambda_{\max}(\boldsymbol{\Sigma}_2)\}} \tag{17}$$

where $\lambda_{\max}(\boldsymbol{\Sigma})$ is the largest eigenvalue of the given covariance matrix [26]. The randomly generated Gaussian components in a mixture were forced to satisfy the pairwise c-separation constraint. Distributions other than the uniform can be used to generate different types of synthetic data for different applications, but c-separation was the only criterion used to control the

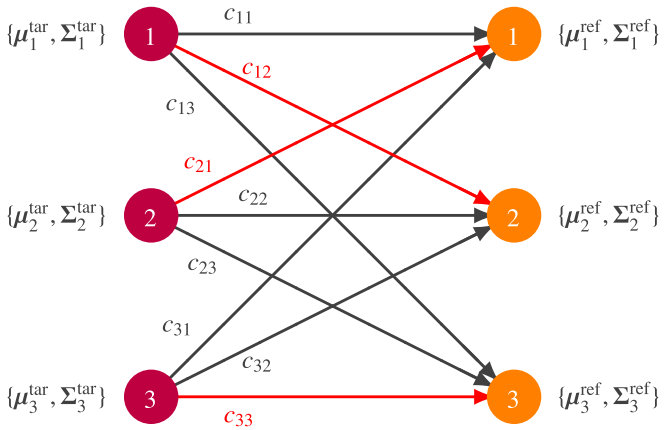

Fig. 5. Optimization formulation for two GMMs with three components shown in Fig. 4. The correspondences found are shown in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

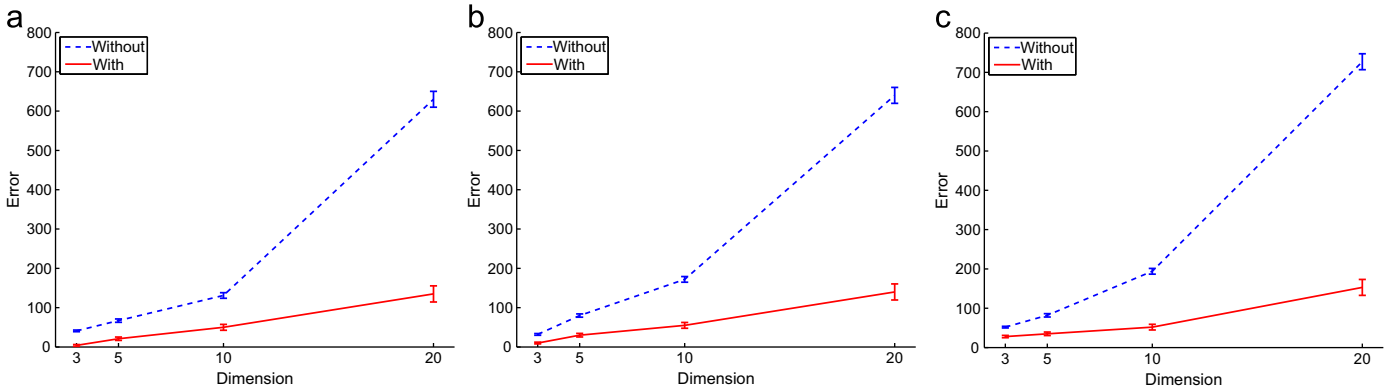

Fig. 6. Average error in log-likelihood and its standard deviation (shown as error bars at one standard deviation) in 1000 trials without and with the correspondence identification step in the estimation of GMMs using stochastic search. (a) GA, (b) DE and (c) PSO.

difficulty of the experiments in this paper. The mixtures in the following simulations were generated for $c=4.0$, $K=5$, and dimensions $d \in \{3,5,10,20\}$. One hundred such mixtures were generated, and 1000 points were sampled from each mixture. The parameters in the candidate solutions in GA, DE, and PSO were randomly initialized as follows. The mean vectors were sampled from the uniform distribution Uniform$[0,1]^d$, the eigenvalues of the covariance matrices were sampled from the uniform distribution Uniform[0,10], and the Givens rotation angles were sampled from the uniform distribution Uniform$[-\pi/4, 3\pi/4]$. Ten different initializations were used for each mixture, resulting in 1000 trials. The true parameters were compared to the estimation results obtained without and with correspondence identification. Fig. 6 shows the plots of estimation errors resulting from the 1000 trials. The error was computed as the difference between the target log-likelihood computed from the true GMM parameters and the resulting log-likelihood computed from the estimated GMM parameters. Based on these results, we can conclude that using the proposed correspondence identification algorithm leads to significantly better results for all stochastic search algorithms used.

6. Particle swarm optimization

We illustrate the proposed solutions for the estimation of GMMs using stochastic search in a particle swarm optimization (PSO) framework. The following sections briefly describe the general PSO formulation by setting up the notation, and then present the details of the GMM estimation procedure using PSO.

6.1. General formulation

PSO is a population-based stochastic search algorithm that is inspired by the social interactions of swarm animals. In PSO, each member of the population is called a particle. Each particle $\mathbf{Z}^{(m)}$ is composed of two vectors, a position vector $\mathbf{Z}_u^{(m)}$ and a velocity vector $\mathbf{Z}_v^{(m)}$ where $m=1,\ldots,M$ indicates the particle index in a population of M particles. The position of each particle $\mathbf{Z}_u^{(m)} \in \mathbb{R}^n$ corresponds to a candidate solution for an n-dimensional optimization problem.

A fitness function defined for the optimization problem of interest is used to assign a goodness value to a particle based on its position. The particle having the best fitness value is called the *global best*, and this position is denoted as $\mathbf{Z}_u^{(\mathrm{GB})}$. Each particle also remembers its best position throughout the search history as its *personal best*, and this position is denoted as $\mathbf{Z}_u^{(m,\mathrm{PB})}$.

PSO begins by initializing the particles with random positions and small random velocities in the n-dimensional parameter space. In the subsequent iterations, each of the n velocity components in $\mathbf{Z}_v^{(m)}$ is computed independently using its previous value, the global best, and the particle's own personal best in a stochastic manner as

$$\mathbf{Z}_v^{(m)}(t+1) = \eta \mathbf{Z}_v^{(m)}(t) + c_1 U_1(t)(\mathbf{Z}_v^{(m,\mathrm{PB})}(t) - \mathbf{Z}_v^{(m)}(t)) + c_2 U_2(t)(\mathbf{Z}_v^{(\mathrm{GB})}(t) - \mathbf{Z}_v^{(m)}(t)) \quad (18)$$

where η is the inertia weight, U_1 and U_2 represent random numbers sampled from Uniform[0,1], c_1 and c_2 are acceleration weights, and t is the iteration number. The randomness of the velocity is obtained by the random numbers U_1 and U_2. These numbers can be sampled from any distribution depending on the application, but we chose the uniform distribution used in the standard PSO algorithm. Then, each particle moves from its old position to a new position using its new velocity vector as

$$\mathbf{Z}_u^{(m)}(t+1) = \mathbf{Z}_u^{(m)}(t) + \mathbf{Z}_v^{(m)}(t+1) \quad (19)$$

and its personal best is modified if necessary. Additionally, the global best of the population is updated based on the particles' new fitness values.

The main difference between PSO and other popular search algorithms like genetic algorithms and differential evolution is that PSO is not an evolutionary algorithm. In evolutionary algorithms, a newly created particle cannot be kept unless it has a better fitness value. However, in PSO, particles are allowed to move to worse locations and this mechanism allows the particles to escape from local optima gradually without the need of any long jump mechanism. In evolutionary algorithms, this can generally be achieved by mutation and crossover operations but these operations can be hard to design for different problems. In addition, PSO uses the global best to coordinate the movement of all particles and uses personal bests to keep track of all local optima found. These properties make it easier to incorporate problem specific ideas into PSO where the global best serves as the current state of the problem and the personal bests serve as the current states of the particles.

6.2. GMM estimation using PSO

The solutions proposed in this paper enable the formulation of a PSO framework for the estimation of GMMs with arbitrary covariance matrices. This formulation involves the definition of the particles, the initialization procedure, the fitness function, and the update procedure.

Particle definition: Each particle that corresponds to a candidate solution stores the parameters of the means and covariance matrices of a GMM. Assuming that the number of components in

the mixture is fixed as K, the position vector of the mth particle is defined as

$$\mathbf{Z}_u^{(m)} = ((\boldsymbol{\mu}_u^{(m,k)})^T, \lambda_{1,u}^{(m,k)}, \ldots, \lambda_{d,u}^{(m,k)}, \phi_u^{12,(m,k)}, \ldots, \phi_u^{(d-1)(d),(m,k)}, \quad \text{for } k=1,\ldots,K) \tag{20}$$

where $\boldsymbol{\mu}_u^{(m,k)} \in \mathbb{R}^d$ for $k=1,\ldots,K$ denote the mean vectors parametrized using d real numbers, $\lambda_{i,u}^{(m,k)} \in \mathbb{R}_{++}$ for $i=1,\ldots,d$ and $k=1,\ldots,K$ denote the eigenvalues of the covariance matrices, and $\phi_u^{pq,(m,k)} \in [-\pi/4, 3\pi/4]$ for $1 \le p < q \le d$ and $k=1,\ldots,K$ denote the Givens rotation angles as defined in Section 5.1. The velocity vector $\mathbf{Z}_v^{(m)}$ is defined similarly. The K mixture weights $\alpha_1,\ldots,\alpha_K$ are calculated from the probabilistic assignments of the data points to the components, and are not part of the PSO particles.

Initialization: Initialization of each particle at the beginning of the first iteration can be done using random numbers within the ranges defined for each parameter. The proposed parametrization makes this possible because the angles are in a fixed range while lower and upper bounds for the mean values and upper bounds for the eigenvalues can easily be selected with the knowledge of the data. As an alternative, one can first randomly select K data points as the means, and form the initial components by assigning each data point to the closest mean. Then, the covariance matrices can be computed from the assigned points, and the parameters of these matrices can be extracted using eigenvalue decomposition and QR factorization using the Givens rotations method as described in Section 5.1. Another alternative for selecting the initial components is the k-means initialization procedure described in [27].

Fitness function: The PSO iterations proceed to find the maximum likelihood estimates by maximizing the log-likelihood defined in (1).

Update equations: Before updating each particle as in (18) and (19), the correspondences between its components and the components of the global best particle are found. This is done by using the particle's personal best as the reference GMM and the global best particle as the target GMM in (15). The correspondence relation computed using (15) and (16) as $I_{ij}=1$ is denoted with a function $f(k)$ that maps the current particle's component index k to the global best particle's corresponding component index $f(k)$ according to $k=j$ and $f(k)=i$ for $I_{f(k)k}=1$. Using this correspondence relation, the mean parameters are updated as

$$\boldsymbol{\mu}_v^{(m,k)}(t+1) = \eta\boldsymbol{\mu}_v^{(m,k)}(t) + c_1(t)(\boldsymbol{\mu}_u^{(m,\mathrm{PB},k)}(t) - \boldsymbol{\mu}_u^{(m,k)}(t)) + c_2(t)(\boldsymbol{\mu}_u^{(\mathrm{GB},f(k))}(t) - \boldsymbol{\mu}_u^{(m,k)}(t)) \tag{21}$$

$$\boldsymbol{\mu}_u^{(m,k)}(t+1) = \boldsymbol{\mu}_u^{(m,k)}(t) + \boldsymbol{\mu}_v^{(m,k)}(t+1) \tag{22}$$

and the eigenvalues and angles as the covariance parameters are updated as

$$\lambda_{i,v}^{(m,k)}(t+1) = \eta\lambda_{i,v}^{(m,k)}(t) + c_1(t)(\lambda_{i,u}^{(m,\mathrm{PB},k)}(t) - \lambda_{i,u}^{(m,k)}(t)) + c_2(t)(\lambda_{i,u}^{(\mathrm{GB},f(k))}(t) - \lambda_{i,u}^{(m,k)}(t)) \tag{23}$$

$$\lambda_{i,u}^{(m,k)}(t+1) = \lambda_{i,u}^{(m,k)}(t) + \lambda_{i,v}^{(m,k)}(t+1) \tag{24}$$

$$\phi_v^{pq,(m,k)}(t+1) = \eta\phi_v^{pq,(m,k)}(t) + c_1(t)(\phi_u^{pq,(m,\mathrm{PB},k)}(t) - \phi_u^{pq,(m,k)}(t)) + c_2(t)(\phi_u^{pq,(\mathrm{GB},f(k))}(t) - \phi_u^{pq,(m,k)}(t)) \tag{25}$$

$$\phi_u^{pq,(m,k)}(t+1) = \phi_u^{pq,(m,k)}(t) + \phi_v^{pq,(m,k)}(t+1). \tag{26}$$

The uniform random numbers U_1 and U_2 are incorporated into c_1 and c_2. The rest of the notation is same as in Sections 5.1 and 6.1.

The convergence of the search procedure can also be improved by running a set of EM iterations for each particle at the end of each iteration. After the covariance parameters are updated as above, new covariance matrices are constructed from the parameters using $\boldsymbol{\Sigma} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$, the EM procedure is allowed to converge to a local maximum as described in Section 4, and new parameters are computed by performing another set of eigenvalue decomposition and QR factorization steps. These EM iterations help converging to local maxima effectively and efficiently, whereas the PSO iterations handle the search for the global maximum. The overall estimation procedure is summarized in Algorithm 2.

Algorithm 2. PSO algorithm for GMM estimation.

Input: d-dimensional data set with N samples, number of components (K), PSO parameters (η, c_1, and c_2)

```
Initialize population with M particles as in (20)
for t=1 to T1 do {T1: number of PSO iterations}
  for m=1 to M do
    Construct K eigenvalue matrices
    Construct K eigenvector matrices by multiplying Givens rotation angles
    Run EM for local convergence for T2 iterations {T2: number of EM iterations for each PSO iteration}
    Compute K eigenvalue and eigenvector matrices via singular value decomposition of new covariance matrices
    Reorder eigenvalues and eigenvectors of each covariance matrix according to personal best
    Extract Givens rotation angles using QR factorization
    Replace particle's means, eigenvalues, and angles
    Calculate log-likelihood
    Update personal best
  end for
  Update global best
  for m=1 to M do
    Reorder components of global best according to personal best
    Update particle's means, eigenvalues, and angles as in (21)–(26)
  end for
end for
```

7. Experiments

We evaluated the framework for GMM estimation (Sections 5 and 6) using both synthetic and real data sets. Comparative experiments were also done using the EM algorithm (Section 4). The procedure used for synthetic data generation and the results for both synthetic and real data sets are given below.

7.1. Experiments on synthetic data

Data sets of various dimensions $d \in \{5, 10, 15, 20, 30, 40\}$ and number of components $K \in \{5, 10, 15, 20\}$ were generated. For dimensions $d \in \{5, 10, 15\}$, $d=20$, and $d \in \{30, 40\}$, the sample size N was set to 1000, 2000, and 4000, respectively. The d and N values were chosen based on real data sets used for the experiments described in the next section. For a particular d and K combination, a GMM was generated as follows. The mixture weights were sampled from a uniform distribution such that the ratio of the largest weight to the smallest weight was at most 2 and all weights summed up to 1. The mean vectors were sampled from the uniform distribution Uniform$[0,100]^d$. The covariance matrices were generated using the eigenvalue/eigenvector parametrization described in Section 5.1. The eigenvalues were sampled from the uniform distribution Uniform[1,16], and the Givens rotation angles were sampled from the uniform

distribution Uniform$[-\pi/4, 3\pi/4]$. Furthermore, the proximity of the components were controlled using c-separation defined in (17). Different values of $c \in \{2.0, 4.0, 8.0\}$ were used to control the difficulty of the estimation problem. The selection of c value was based on visual observations in two-dimensional data. We observed that the minimum value of c where K individual Gaussian components were distinguishable by visual inspection was close to 2.0, and $c=8.0$ corresponded to the case where the components were well separated. Consequently, we divided the relative difficulties of the data sets into three. The *easy* settings corresponded to $d \in \{5, 10\}$ and $c=8.0$, the *medium* settings corresponded to $d \in \{10, 15, 20\}$ and $c=4.0$, and the *hard* settings corresponded to $d \in \{20, 30, 40\}$ and $c=2.0$. Ten different mixtures with N samples each were generated for each setting.

The PSO and EM parameters were initialized similarly for a fair evaluation. We assumed that the number of components was known a priori for each data set. Following the common practice in the literature, the initial mean vector for each component was set to a randomly selected data point. The initial covariance matrices and the initial mixture weights were calculated from the probabilistic assignment of the data points to the components with the initial mean vectors and identity covariance matrices. The initial mixture weights were used only in the EM procedure as the proposed algorithm does not include the weights as parameters. After initialization, the search procedure constrained the components of the mean vectors in each particle defined in (20) to stay in the data region defined by the minimum and maximum values of each component in the data used for estimation. Similarly, the eigenvalues were constrained to stay in $[\lambda_{\min}, \lambda_{\max}]$ where $\lambda_{\min} = 10^{-5}$ and $\lambda_{\max}$ was the maximum eigenvalue of the covariance matrix of the whole data, and the Givens rotation angles were constrained to lie in $[-\pi/4, 3\pi/4]$. The PSO parameters η, c_1, and c_2 in (18) were fixed at $\eta = 0.728$, $c_1 = c_2 = 1.494$ following the common practice in the PSO literature [24]. Thus, no parameter tuning was done during both initialization and search stages.

For each test mixture, each PSO run consisted of M particles that were updated for T_1 iterations where each iteration also consisted of at most T_2 EM iterations as described at the end of Section 6.2. Each primary EM run consisted of a group of M individual secondary runs where the initial parameters of each secondary run was the same as the parameters of one of the M particles in the corresponding PSO run. Each secondary run was allowed to iterate for at most $T_1 \times T_2$ iterations or until the relative change in the log-likelihood in two consecutive iterations was less than 10^{-6}. The number of iterations were adjusted such that each PSO run (M particles with T_1 PSO iterations and T_2 EM iterations for each PSO iteration) and the corresponding primary EM run (M secondary EM runs with $T_1 \times T_2$ iterations each) were compatible.

Table 3 shows the details of the synthetic data sets generated using these settings. For each setting, 10 different mixtures with N samples each were generated as described above. For each mixture, the target log-likelihood was computed from the true GMM parameters. Then, for each mixture, 10 different initializations were obtained as described above, and both the PSO and the EM procedures were run for each initial configuration. The parameters of the global best particle were selected as the final result of each PSO run at the end of the iterations. The final result of each primary EM run was selected as the parameters corresponding to the best secondary run having the highest log-likelihood among the M secondary runs. The estimation error was computed as the difference between the target log-likelihood and the resulting log-likelihood computed from the estimated GMM parameters.

Table 4 and Fig. 7 present the error statistics computed from the 100 runs (10 different mixtures and 10 different initializations for each mixture) for each setting. When all settings were considered, it could be seen that the proposed PSO algorithm resulted in better estimates compared to those by the EM algorithm for all settings. In particular, the PSO algorithm converged to the true GMM parameters in more than half of the runs for 11 out of 18 settings (all of the 10 *easy* and *medium* settings and one *hard* setting) with a median error of zero, whereas the EM algorithm could do the same for only five settings. For all settings, the average error obtained by the PSO algorithm was significantly lower than the error by the EM algorithm. For the settings with a small number of components, both EM and PSO had no problem in finding the optimal solution. This was mainly due to good initial conditions where it was relatively easier to find a small number of good initial data points that behaved as good initial means. Note that a good initialization for only one of the M secondary runs for each primary EM run was sufficient to report a perfect performance because the best out of M was used.

Table 3
Details of the synthetic data sets used for performance evaluation. The three groups of rows correspond to the settings categorized as *easy*, *medium*, and *hard* with respect to their relative difficulties. The parameters are described in the text.

Setting #	d	K	c	N	M	T_1	T_2	$T_1 \times T_2$
1	5	5	8.0	1000	20	30	20	600
2	5	10	8.0	1000	20	30	20	600
3	10	5	8.0	1000	20	30	20	600
4	10	5	4.0	1000	20	30	20	600
5	10	10	4.0	1000	20	30	20	600
6	10	15	4.0	1000	20	30	20	600
7	15	5	4.0	1000	30	30	20	600
8	15	10	4.0	1000	30	30	20	600
9	15	15	4.0	1000	30	30	20	600
10	20	5	4.0	2000	30	50	20	1000
11	20	10	2.0	2000	30	50	20	1000
12	20	15	2.0	2000	30	50	20	1000
13	20	20	2.0	2000	30	50	20	1000
14	30	10	2.0	4000	40	100	20	2000
15	30	15	2.0	4000	40	100	20	2000
16	30	20	2.0	4000	40	100	20	2000
17	40	15	2.0	4000	40	100	20	2000
18	40	20	2.0	4000	40	100	20	2000

Table 4
Statistics of the estimation error for the synthetic data sets using the GMM parameters estimated via the EM and PSO procedures. The mean, standard deviation (std), median, and median absolute deviation (mad) are computed from 100 different runs for each setting.

Setting #	EM				PSO			
	Mean	Std	Median	Mad	Mean	Std	Median	Mad
1	6.18	61.46	0.00	0.00	0.00	0.00	0.00	0.00
2	304.99	183.36	362.71	71.94	41.30	112.55	0.00	0.00
3	66.59	335.93	0.00	0.00	17.42	122.22	0.00	0.00
4	20.32	115.54	0.00	0.00	0.00	0.00	0.00	0.00
5	283.29	135.85	331.03	37.41	27.15	81.98	0.00	0.00
6	500.68	110.17	480.89	78.46	69.80	83.05	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	300.83	174.13	367.08	68.42	11.28	55.66	0.00	0.00
9	654.48	145.67	654.23	163.56	51.39	100.70	0.00	0.00
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
11	490.14	307.53	615.89	126.93	112.75	227.90	0.00	0.00
12	842.94	242.63	880.06	192.40	224.89	231.03	97.21	75.14
13	975.60	152.44	912.21	113.53	261.34	98.73	120.66	45.12
14	1171.30	592.29	1105.42	205.61	236.63	315.23	102.31	102.70
15	1651.47	518.35	1576.24	124.21	309.21	232.49	272.18	58.23
16	2098.39	460.39	1971.43	384.08	523.84	183.92	375.28	114.02
17	2328.13	676.15	2093.80	403.16	609.92	281.59	412.54	93.84
18	2946.89	760.48	2882.77	425.04	697.02	292.17	468.27	100.57

The above argument could be extended for PSO to all settings relatively independent of the number of dimensions and the number of components. We could conclude that the proposed algorithm was less sensitive to initializations because in every iteration the particles took small number of steps toward one of the local maxima using the local EM iterations, and then due to their interaction with the global best, they could move away from that local maximum. We could argue that the common characteristic of the small number of wrong convergences of PSO was the initialization of most of the particles including the global best near the same local maximum. In that case, both the local EM iterations and the global best particle attracted all particles toward the same region. This problem could be eliminated by a more sophisticated initialization procedure that increased the diversity of the particles. However, we used the same initialization procedure that used the same random points for both EM and PSO algorithms to do a fair comparison.

In this paper, we only investigated the advantages of correspondence identification with regard to finding better global maxima of the log-likelihood. We showed that stochastic search algorithms performed better in finding global optima. However, correspondence identification can also be useful in increasing the population diversity. For instance, once we find the correspondence relations via the proposed matching algorithm, we can force the parameters to be updated with the distant (not matching) ones in the global best in some random way to increase the diversity. Another approach may be to temporarily modify the update equations so that the particles move away from the global best if the KL divergence between their personal best and the global best becomes too small in early iterations to overcome premature convergence to a local maximum.

We did not try to tune the parameters of PSO such as η, c_1, and c_2. For different settings, parameter tuning might be useful in terms of increased convergence speed and increased estimation accuracy. However, such tuning could have led to an unfair advantage of PSO over the EM algorithm. We also did not tune the number of particles and the number of iterations except increasing them linearly with increasing dimension. Increasing the number of iterations will not improve the performance of EM after its convergence but larger number of iterations will allow PSO to explore a larger portion of the parameter space. However, the number of iterations were fixed to the same number for EM and PSO to allow a fair comparison.

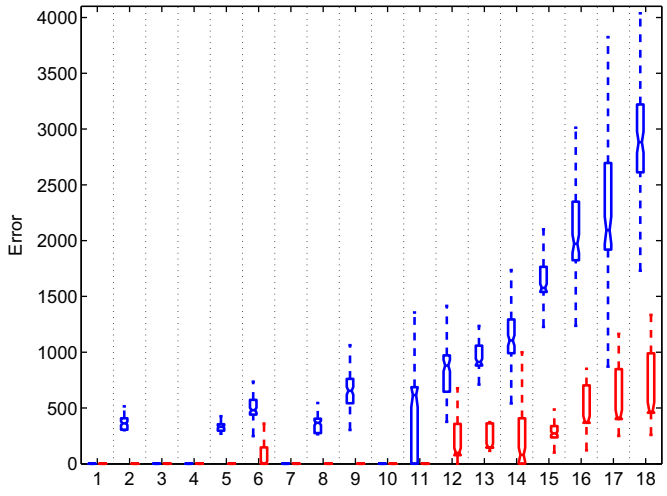

Fig. 7. Statistics of the estimation error for the synthetic data sets using the GMM parameters estimated via the EM (blue) and PSO (red) procedures. The boxes show the lower quartile, median, and upper quartile of the error. The whiskers drawn as dashed lines extend out to the extreme values. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

7.2. Experiments on real data

We also used four data sets from the UCI Machine Learning Repository [28] for real data experiments. These data sets are referred to as *Glass* (glass identification), *Wine*, *ImgSeg* (Statlog image segmentation), and *Landsat* (Statlog Landsat satellite). Table 5 summarizes the characteristics of these data sets and the corresponding experimental settings. For each data set and for each K value, both PSO and EM were run using 10 different initial configurations that were generated as described in the previous section. The resulting log-likelihood values for each setting for each data set are shown in Fig. 8. The results show that the proposed PSO algorithm performed better than the EM algorithm for all settings.

7.3. Computational complexity

The overall worst case time complexity of the EM algorithm in terms of the overall number of iterations T, the number of components K, and the number of data dimensions d is $O(TKd^3N)$. It involves a singular value decomposition that takes $O(d^3)$ for each of the K covariance matrices in each of the T iterations, and the multiplication of K eigenvalues (d), eigenvector matrices ($d \times d$), and mean subtracted data matrices ($d \times N$). The former has $O(TKd^3)$

Table 5
Details of the real data sets used for performance evaluation. K_{true} corresponds to the number of classes in each data set. K corresponds to the number of Gaussian components used in the experiments. The rest of the parameters are described in the text.

Data set	d	K_{true}	K	N	M	T_1	T_2	$T_1 \times T_2$
Glass	9	6	{6, 7, 8, 9, 10}	214	20	30	20	600
Wine	13	3	{3, 4, 5, 6, 7}	178	30	30	20	600
ImgSeg	19	7	{7, 8, 9, 10, 11}	2310	30	50	20	1000
Landsat	36	7	{7, 8, 9, 10, 11}	4435	40	100	20	2000

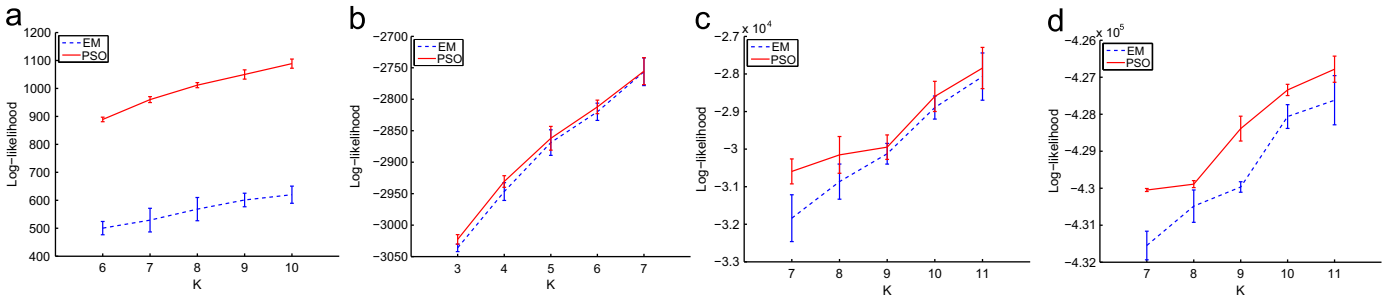

Fig. 8. Average log-likelihood and its standard deviation (shown as error bars at one standard deviation) computed from 10 different runs of EM and PSO procedures for the real data sets. (a) *Glass*, (b) *Wine*, (c) *ImgSeg* and (d) *Landsat*.

complexity and the latter has $O(TKd^3N)$ complexity, leading to the overall complexity given above. The PSO algorithm has additional QR factorizations to extract the Givens rotation angles and the multiplication of the resulting angles that both take $O(d^3)$ time, but these operations do not change the overall complexity. We can conclude that both the EM algorithm and the proposed PSO-based algorithm have the same worst case time complexities.

8. Conclusions

We presented a framework for effective utilization of stochastic search algorithms for the maximum likelihood estimation of unrestricted Gaussian mixture models. One of the contributions of this paper was a covariance parametrization that enabled the use of arbitrary covariance matrices in the search process. The parametrization used eigenvalue decomposition, and modeled each covariance matrix in terms of its eigenvalues and Givens rotation angles extracted from the eigenvector matrices. This parametrization allowed the individual parameters to be independently modifiable so that the resulting matrices remained valid covariance matrices after the stochastic updates. Furthermore, the parameters had finite lower and upper bounds so that they could be searched within a bounded solution space. We also described an algorithm for ordering the eigenvectors so that the parameters of individual Gaussian components were uniquely identifiable.

Another contribution of this paper was an optimization formulation for resolving the identifiability problem for the mixtures. The proposed solution allowed a unique correspondence between two candidate solutions so that desirable interactions became possible for parameter updates throughout the stochastic search.

We showed that the proposed methods can be used effectively with different stochastic search algorithms such as genetic algorithms, differential evolution, and particle swarm optimization. The final set of experiments using particle swarm optimization with synthetic and real data sets showed that the proposed algorithm could achieve significantly higher likelihood values compared to those obtained by the conventional EM algorithm under the same initial conditions.

Acknowledgment

This work was supported in part by the TUBITAK Grants 104E074 and 109E193.

References

[1] R.A. Redner, H.F. Walker, Mixture densities, maximum likelihood and the EM algorithm, SIAM Review 26 (2) (1984) 195–239.
[2] G. McLachlan, D. Peel, Finite Mixture Models, John Wiley & Sons, Inc., 2000.
[3] P. Schroeter, J.-M. Vesin, T. Langenberger, R. Meuli, Robust parameter estimation of intensity distributions for brain magnetic resonance images, IEEE Transactions on Medical Imaging 17 (2) (1998) 172–186.
[4] A.M. Martinez, J. Vitria, Learning mixture models using a genetic version of the EM algorithm, Pattern Recognition Letters 21 (8) (2000) 759–769.
[5] F. Pernkopf, D. Bouchaffra, Genetic-based EM algorithm for learning Gaussian mixture models, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (8) (2005) 1344–1348.
[6] J. Tohka, E. Krestyannikov, I.D. Dinov, A.M. Graham, D.W. Shattuck, U. Routsalainen, A.W. Toga, Genetic algorithms for finite mixture model based voxel classification in neuroimaging, IEEE Transactions on Medical Imaging 26 (5) (2007) 696–711.
[7] D.-X. Chang, X.-D. Zhang, C.-W. Zheng, A genetic algorithm with gene rearrangement for k-means clustering, Pattern Recognition 42 (7) (2009) 1210–1222.
[8] U. Maulik, I. Saha, Modified differential evolution based fuzzy clustering for pixel classification in remote sensing imagery, Pattern Recognition 42 (9) (2009) 2135–2149.
[9] M. Omran, A.P. Engelbrecht, A. Salman, Particle swarm optimization method for image clustering, International Journal of Pattern Recognition and Artificial Intelligence 19 (3) (2005) 297–321.
[10] A. Abraham, S. Das, S. Roy, Swarm intelligence algorithms for data clustering, in: O. Maimon, L. Rokach (Eds.), Soft Computing for Knowledge Discovery and Data Mining, Springer, 2007, pp. 279–313.
[11] A. Paoli, F. Melgani, E. Pasolli, Clustering of hyperspectral images based on multiobjective particle swarm optimization, IEEE Transactions on Geoscience and Remote Sensing 47 (12) (2009) 4175–4188.
[12] S. Kiranyaz, T. Ince, A.Y.amdM. Gabbouj, Fractional particle swarm optimization in multidimensional search space, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 40 (2) (2010) 298–319.
[13] C. Ari, S. Aksoy, Maximum likelihood estimation of Gaussian mixture models using particle swarm optimization, in: Proceedings of 20th IAPR International Conference on Pattern Recognition, Istanbul, Turkey, 2010.
[14] G. Celeux, M. Hurn, C.P. Robert, Computational and inferential difficulties with mixture posterior distributions, Journal of the American Statistical Association 95 (451) (2000) 957–970.
[15] M. Stephens, Dealing with label switching in mixture models, Journal of the Royal Statistical Society: Series B 62 (4) (2000) 795–809.
[16] M. Sperrin, T. Jaki, E. Wit, Probabilistic relabelling strategies for the label switching problem in Bayesian mixture models, Statistics and Computing 20 (3) (2010) 357–366.
[17] S.J. Roberts, D. Husmeier, I. Rezek, W. Penny, Bayesian approaches to Gaussian mixture modeling, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (11) (1998) 1133–1142.
[18] M.A.F. Figueiredo, A.K. Jain, Unsupervised learning of finite mixture models, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (3) (2002) 381–396.
[19] N. Vasconcelos, A. Lippman, Learning mixture hierarchies, in: Proceedings of Neural Information Processing Systems, Denver, CO, 1998.
[20] P. Bruneau, M. Gelgon, F. Picarougne, Parsimonious reduction of Gaussian mixture models with a variational-Bayes approach, Pattern Recognition 43 (3) (2010) 850–858.
[21] G.J. McLachlan, T. Krishnan, The EM Algorithm and Extensions, 2nd ed., John Wiley & Sons, Inc., 2008.
[22] G.H. Golub, C.F. Van Loan, Matrix Computations, 3rd ed., Johns Hopkins University Press, 1996.
[23] R. Storn, K. Price, Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (4) (1997) 341–359.
[24] R.C. Eberhart, Y. Shi, Particle swarm optimization: developments, applications and resources. in: Proceedings of the Congress on Evolutionary Computation, vol. 1, 2001, pp. 81–86.
[25] J. Edmonds, R.M. Karp, Theoretical improvements in algorithmic efficiency for network flow problems, Journal of the ACM 19 (2) (1972) 248–264.
[26] S. Dasgupta, Learning mixtures of Gaussians, in: 40th Annual Symposium on Foundations of Computer Science, 1999, pp. 634–644.
[27] D. Arthur, S. Vassilvitskii, k-means++: the advantages of careful seeding, in: ACM-SIAM Symposium on Discrete Algorithms, New Orleans, LA, 2007, pp. 1027–1035.
[28] A. Asuncion, D.J. Newman, UCI machine learning repository 〈http://www.ics.uci.edu/∼mlearn/MLRepository.html〉, 2007.

Caglar Ari received the B.S. degree from Bilkent University, Ankara, Turkey, in 2004. He is currently a Ph.D. candidate at the Department of Electrical and Electronics Engineering at Bilkent. His research interests include computer vision and pattern recognition.

Selim Aksoy received the B.S. degree from the Middle East Technical University, Ankara, Turkey, in 1996 and the M.S. and Ph.D. degrees from the University of Washington, Seattle, in 1998 and 2001, respectively. Since 2004, he has been an Assistant Professor at the Department of Computer Engineering, Bilkent University, Ankara. His research interests include computer vision, statistical and structural pattern recognition with applications to remote sensing, medical imaging, and multimedia data analysis. He is an Associate Editor of Pattern Recognition Letters.

Orhan Arikan received the B.Sc. degree in Electrical and Electronics Engineering from the Middle East Technical University, Ankara, Turkey, in 1986, and both the M.S. and Ph.D. degrees in Electrical and Computer Engineering from the University of Illinois, Urbana-Champaign, in 1988 and 1990, respectively. Following his graduate studies, he worked for three years as a Research Scientist at Schlumberger-Doll Research, Ridgefield, CT. He joined Bilkent University in 1993, where he is presently Professor of Electrical Engineering. His current research interests are in statistical signal processing, time–frequency analysis, and radar signal processing.