Learning Discriminative $\alpha\beta$-Divergences for Positive Definite Matrices

[1]A. Cherian* [2]P. Stanitsas* [3]M. Harandi [2]V. Morellas [2]N. Papanikolopoulos
[1]Australian Centre for Robotic Vision, [3]Data61/CSIRO, [1,3]The Australian National University
[2]Dept. of Computer Science, University of Minnesota, Minneapolis
{anoop.cherian, mehrtash.harandi}@anu.edu.au, {stani078, morellas, npapas}@umn.edu

Abstract

Symmetric positive definite (SPD) matrices are useful for capturing second-order statistics of visual data. To compare two SPD matrices, several measures are available, such as the affine-invariant Riemannian metric, Jeffreys divergence, Jensen-Bregman logdet divergence, etc.; however, their behaviors may be application dependent, raising the need of manual selection to achieve the best possible performance. Further and as a result of their overwhelming complexity for large-scale problems, computing pairwise similarities by clever embedding of SPD matrices is often preferred to direct use of the aforementioned measures. In this paper, we propose a discriminative metric learning framework, Information Divergence and Dictionary Learning *(IDDL), that not only learns application specific measures on SPD matrices* automatically*, but also embeds them as vectors using a learned dictionary. To learn the similarity measures (which could potentially be distinct for every dictionary atom), we use the recently introduced $\alpha\beta$-logdet divergence, which is known to unify the measures listed above. We propose a novel IDDL objective, that learns the parameters of the divergence and the dictionary atoms jointly in a discriminative setup and is solved efficiently using Riemannian optimization. We showcase extensive experiments on eight computer vision datasets, demonstrating state-of-the-art performances.*

1. Introduction

Symmetric Positive Definite (SPD) matrices arise naturally in several computer vision applications, such as covariances when modeling data using Gaussians, as kernel matrices for high-dimensional embedding, as points in diffusion MRI [37], and as structure tensors in image processing [5]. Furthermore, SPD matrices in the form of Region CoVariance Descriptors (RCoVDs) [44], offer an easy way to compute a representation that fuses multiple modalities (*e.g.*, color, gradients, filter responses, etc.) in a cohesive, and compact format. In various mainstream vision applications, including tracking, re-identification, object, texture, and activity recognition, the trail of SPD matrices to advance the state-of-the-art solutions can be seen [6, 45, 18]. SPD matrices are even used as second-order pooling operators for enhancing the performance of popular deep learning architectures [24, 22].

SPD matrices, due to their positive definiteness property, form a cone in the Euclidean space. However, analyzing these matrices through their Riemannian geometry (or the associated Lie algebra) helps avoiding unlikely/unrealistic solutions, thereby improving the outcomes. For example, in diffusion MRI [37, 3], it has been shown that the Riemannian structure (which comes with an affine invariant met-

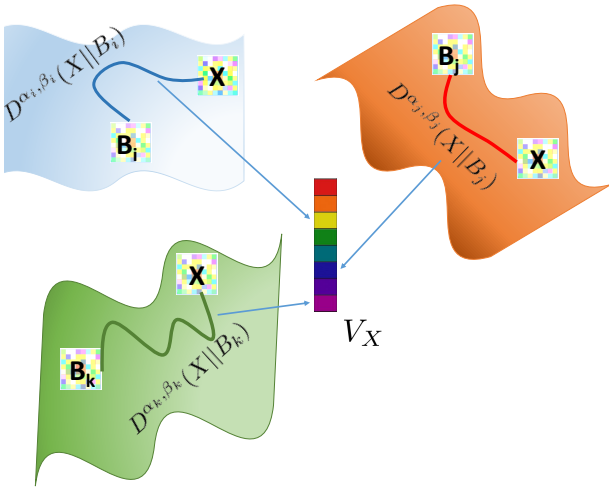


Figure 1. A schematic illustration of our IDDL scheme. From an infinite set of potential geometries, our goal is to learn multiple geometries (parameterized by (α, β)) and representative dictionary atoms for each geometry (represented by B's), such that a given SPD data matrix X can be embedded into a similarity vector V_X, each dimension of which captures the divergence of X to the Bs using the respective measure. We use V_X for classification.

*Equal contribution.

ric) is immensely useful for accurate modeling. A similar observation is made for RCoVDs [9, 20, 46]. This has resulted in the exploration of various geometries and similarity measures for SPD matrices, viewing them from disparate perspectives. A few notable such measures are: (i) the affine invariant Riemannian metric (AIRM) using the natural Riemannian geometry [37], (ii) the Jeffreys KL divergence (KLDM) using relative entropy [35], (iii) the Jensen-Bregman logdet divergence using information geometry [9], and (iv) Brug matrix divergence [28], among several others [12].

Each of the aforementioned measures has distinct mathematical properties and as such performs differently for a given problem. However and to some extent surprisingly, all of them can be obtained as functions acting on the generalized eigenvalues of their inputs. Recently, Cichocki et al. [12] show that all these measures can be interpreted in a unifying setup using $\alpha\beta$-logdet divergence (ABLD) and each measure can be derived as a distinct parametrization of this divergence. For example, one could get JBLD from ABLD[1] using $\alpha = \beta = \frac{1}{2}$, and AIRM as the limit of $\alpha, \beta \to 0$. With such an interesting discovery, it is natural to ask if the parameters α and β can be learned for a given task in a data-driven way. *This not only answers which measure is the right choice for a given problem, but also allows for deriving new measures that are not among the popular ones listed above.*

In this paper, we make the first attempt at learning an $\alpha\beta$-logdet divergence on SPD matrices for computer vision applications, dubbed *Information Divergence and Dictionary Learning* (IDDL). We cast the learning problem in a discriminative ridge regression setup where the goal is to learn α and β that maximize the classification accuracy for a given task.

Being vigilant to the computational complexity of the resulting solution, we propose to embed SPD matrices using a dictionary in our metric learning framework. Our proposal enables us to learn the embedding (or more accurately the dictionary that identifies the embedding), along with the proper choice of the metric (*i.e.*, parameters α and β of the ABLD) and a classifier jointly. The output of our IDDL is a vector, each entry of this vector computes a potentially distinct ABLD to a distinct dictionary atom.

To achieve our goal, we propose an efficient formulation that benefits from recent advances in optimization over Riemannian manifolds to minimize a non-convex and constrained objective. We provide extensive experiments using IDDL on a variety of computer vision applications, namely (i) action recognition, (ii) texture recognition, (iii) 3D shape recognition, and (iv) cancerous tissue recognition. We also provide insights into our learning scheme through extensive experiments on the parameters of the ABLD, and ablation studies under various performance settings. Our results demonstrate that our scheme achieves state-of-the-art accuracies against competing techniques, including the recent sparse coding, Riemannian metric learning, and kernel coding schemes.

[1] Up to a scaling factor.

2. Related Work

The $\alpha\beta$-logdet divergence is a matrix generalization of the well-known $\alpha\beta$-divergence [11] that computes the (a)symmetric (dis)similarity between two finite positive measures (data densities). As the name implies, $\alpha\beta$-divergence is a unification of the so-called α-family of divergences [2] (that includes popular measures such as the KL-divergence, Jensen-Shannon divergence, and the chi-square divergence) and the β-family [4] (including the squared Euclidean distance and the Itakura Saito distance). Against several standard measures for computing similarities, both α and β divergences are known to lead to solutions that are robust to outliers and additive noise [30], thereby improving application accuracy. They have been used in several statistical learning applications including non-negative matrix factorization [13, 26, 14], nearest neighbor embedding [21], and blind-source separation [34].

A class of methods with similarities to our formulation are metric learning schemes on SPD matrices. One popular technique is the manifold-manifold embedding of large SPD matrices into a tiny SPD space in a discriminative setting [20]. Log-Euclidean metric learning has also been proposed for this embedding in [23, 41]. While, we also learn a metric in a discriminative setup, ours is different in that we learn an information divergence. In Thiyam et al. [43], ABLD is proposed replacing symmetric KL divergence in better characterizing the learning of a decision hyperplane for BCI applications. In contrast[2], we propose to embed the data matrices as vectors, each dimension of these vectors learning a different ABLD, thus leading to a richer representation of the input matrix.

Vectorial embedding of SPD matrices has been investigated using disparate formulations for computer vision applications. As alluded to earlier, the log-Euclidean projection [3] is a common way to achieve this, where an SPD matrix is isomorphically mapped to the Euclidean space of symmetric matrices using the matrix logarithm. Popular sparse coding schemes have been extended to SPD matrices in [8, 40, 47] using SPD dictionaries, where the resulting sparse vector is assumed Euclidean. Another popular way to handle the non-linear geometry of SPD matrices is to resort to kernel schemes by embedding the matrices in an infinite dimensional Hilbert space which is assumed to

[2] Automatic selection of the parameters of $\alpha\beta$-divergence is investigated in [39, 15]. However, they deal with scalar density functions in a maximum-likelihood setup and do not consider the optimization of α and β jointly.

be linear [19, 32, 18]. In all these methods, the underlying similarity measure is fixed and is usually chosen to be one among the popular $\alpha\beta$-logdet divergences or the log-Euclidean metric.

In contrast to all these methods, to the best of our knowledge, it is for the first time that a joint dictionary learning and information divergence learning framework is proposed for SPD matrices in computer vision. In the sequel, we first introduce $\alpha\beta$-logdet divergence and explore its properties in the next section. This will precede exposition to our discriminative metric learning framework for learning the divergence and efficient ways of solving our formulation.

Notations: Following standard notations, we use upper case for matrices (such as X), lower-bold case for vectors $\mathbf{x}$, and lower case for scalars x. Further, $\mathcal{S}_{++}^d$ is used to denote the cone of $d \times d$ SPD matrices. We use $\mathbf{B}$ to denote a 3D tensor each slice of which is an SPD matrix of size $d \times d$. Further, we use $\mathbf{I}_d$ to denote the $d \times d$ identity matrix, Log for the matrix logarithm, and diag for the diagonalization operator.

3. Background

In this section, we will setup the mathematical preliminaries necessary to elucidate our contributions. We will visit the $\alpha\beta$-log-det divergence, its connections to other popular divergences, and its mathematical properties.

3.1. $\alpha\beta$-Log Determinant Divergence

Definition 1 (ABLD [12]) *For $X, Y \in \mathcal{S}_{++}^d$, the $\alpha\beta$-log-det divergence is defined as:*

$$D^{(\alpha,\beta)}(X\|Y) = \frac{1}{\alpha\beta}\log\det\left(\frac{\alpha(XY^{-1})^{\beta}+\beta(XY^{-1})^{-\alpha}}{\alpha+\beta}\right), \tag{1}$$

$$\alpha \neq 0,\ \beta \neq 0 \ \text{ and } \alpha+\beta \neq 0. \tag{2}$$

It can be shown that ABLD depends only on the generalized eigenvalues of X and Y [12]. Suppose λ_i denotes the i-th eigenvalue of XY^{-1}. Then under constraints defined in (2), we can rewrite (1) as:

$$D^{(\alpha,\beta)}(X\|Y) = \frac{1}{\alpha\beta}\sum_{i=1}^{d}\log\left(\alpha\lambda_i^{\beta}+\beta\lambda_i^{-\alpha}\right) - d\log(\alpha+\beta). \tag{3}$$

This formulation will come handy when deriving the gradient updates for α and β in the sequel. As alluded to earlier, a hallmark of the ABLD is that it unifies several popular distance measures on SPD matrices that one commonly encounters in computer vision applications. In Table 1, we list some of the popular measures in computer vision and the respective values of α and β.

3.2. ABLD Properties

Avoiding Degeneracy: An important observation regarding the design of optimization algorithms on ABLD is that the quantity inside the $\log\det$ term has to be positive definite; conditions on α and β for which are specified by the following theorem.

Theorem 1 ([12]) *For $X, Y \in \mathcal{S}_{++}^d$, if λ_i is the i-th eigenvalue of $X^{-1}Y$, then $D^{(\alpha,\beta)}(X \parallel Y) \geq 0$ only if*

$$\lambda_i > \left|\frac{\alpha}{\beta}\right|^{\frac{1}{\alpha+\beta}}, \text{ for } \alpha > 0 \text{ and } \beta < 0, \text{ or} \tag{4}$$

$$\lambda_i < \left|\frac{\beta}{\alpha}\right|^{\frac{1}{\alpha+\beta}}, \text{ for } \alpha < 0 \text{ and } \beta > 0, \forall i = 1, 2, \cdots, d. \tag{5}$$

Since λ_is depend on the input matrices, on which we have no control over, we constrain α and β to have the same sign, thereby avoiding the quantity inside $\log\det$ to be indefinite. We make this assumption in our formulations in Section 4.

Smoothness of α, β: Assuming α, β have the same sign, except at origin ($\alpha = \beta = 0$), ABLD is smooth everywhere with respect to α and β, thus allowing us to develop Newton-type algorithms on them. Due to the discontinuity at the origin, we ought to design algorithms specifically addressing this particular case.

Affine Invariance: It can be easily shown that

$$D^{(\alpha,\beta)}(X \parallel Y) = D^{(\alpha,\beta)}(AXA^T \parallel AYA^T), \tag{6}$$

for any invertible matrix A. This is an important property that makes this divergence useful in a variety of applications, such as diffusion MRI [37].

Dual Symmetry: This property allows us to extend results derived for the case of α to the one on β later.

$$D^{(\alpha,\beta)}(X \parallel Y) = D^{(\beta,\alpha)}(Y \parallel X). \tag{7}$$

Before concluding this part, we briefly introduce the concept of optimization on Riemannian manifolds and in particular the method of Riemmanian Conjugate Gradient descent (RCG).

3.3. Optimization on Riemannian Manifolds

As will be shown in § 4, we need to solve a non-convex constrained optimization problem in the form

$$\begin{aligned} &\text{minimize } \mathcal{L}(B) \\ &\text{s.t. } \ B \in \mathcal{S}_{++}^d\ . \end{aligned} \tag{8}$$

Classical optimization methods generally turn a constrained problem into a sequence of unconstrained problems for which unconstrained techniques can be applied.

(α, β)	ABLD	Divergence
$(\alpha, \beta) \to 0$	$\left\|\operatorname{Log} X^{-\frac{1}{2}} Y X^{-\frac{1}{2}}\right\|_F^2$	Squared Affine Invariant Riemannian Metric [37]
$\alpha = \beta = \pm\frac{1}{2}$	$4\left(\log\det\frac{X+Y}{2} - \frac{1}{2}\log\det XY\right)$	Jensen-Bregman Logdet Divergence [9]
$\alpha = \pm 1, \beta \to 0$	$\frac{1}{2}Tr\left(XY^{-1} + YX^{-1}\right) - d$	Jeffreys KL Divergence[3] [35]
$\alpha = 1, \beta = 1$	$Tr\left(XY^{-1}\right) - \log\det XY^{-1} - d$	Burg Matrix Divergence [28]

Table 1. ABLD and its connections to popular divergences used in computer vision applications.

In contrast, in this paper we make use of the optimization on Riemannian manifolds to minimize (8). This is motivated by recent advances in Riemannian optimization techniques where benefits of exploiting geometry over standard constrained optimization are shown [1]. As a consequence, these techniques have become increasingly popular in diverse application domains [8, 18].

A detailed discussion of Riemannian optimization goes beyond the scope of this paper, and we refer the interested reader to [1]. However, the knowledge of some basic concepts will be useful in the remainder of this paper. As such, here, we briefly consider the case of Riemannian Conjugate Gradient method (RCG), our choice when the empirical study of this work is considered. First we formally define the SPD manifold.

Definition 2 (The SPD Manifold) *The set of $(d \times d)$ dimensional real, SPD matrices endowed with the Affine Invariant Riemannian Metric (AIRM) [37] forms the SPD manifold $\mathcal{S}_{++}^d$.*

$$\mathcal{S}_{++}^p \triangleq \left\{X \in \mathbb{R}^{d\times d} : v^T X v > 0,\ \forall v \in \mathbb{R}^d - \{0_d\}\right\}. \quad (9)$$

To minimize (8), RCG starts from an initial solution $B^{(0)}$ and improves its solution using the update rule

$$B^{(t+1)} = \tau_{B^{(t)}}\left(P^{(t)}\right), \quad (10)$$

where $P^{(t)}$ identifies a search direction and $\tau_B(\cdot) : T_B\mathcal{S}_{++}^d \to \mathcal{S}_{++}^d$ is a *retraction*. The retraction serves to identify the new solution along the geodesic defined by the search direction $P^{(t)}$. In RCG, it is guaranteed that the new solution obtained by Eq. (10) is on $\mathcal{S}_{++}^d$ and has a lower objective. The search direction $P^{(t)} \in T_{B^{(t)}}\mathcal{S}_{++}^d$ is obtained by

$$P^{(t)} = -\operatorname{grad}\mathcal{L}(B^{(t)}) + \eta^{(t)}\pi(P^{(t-1)}, B^{(t-1)}, B^{(t)}). \quad (11)$$

Here, $\eta^{(t)}$ can be thought of as a variable learning rate, obtained via techniques such as Fletcher-Reeves [1]. Furthermore, $\operatorname{grad}\mathcal{L}(B)$ is the Riemannian gradient of the objective function at B and $\pi(P, X, Y)$ denotes the parallel transport of P from T_X to T_Y. In Table 2, we define the mathematical entities required to perform RCG on the SPD manifold. Note that computing the standard Euclidean gradient of the function $\mathcal{L}$, denoted by $\nabla_*(\mathcal{L})$, is the only requirement to perform RCG on $\mathcal{S}_{++}^d$.

	$\mathcal{S}_{++}^d$
Riemannian gradient	$\operatorname{grad}\mathcal{L}(B) = B\operatorname{sym}\left(\nabla_B(\mathcal{L})\right)B$
Retraction.	$\tau_B(\xi) = B^{\frac{1}{2}}\operatorname{Exp}(B^{-\frac{1}{2}}\xi B^{-\frac{1}{2}})B^{\frac{1}{2}}$
Parallel Transport.	$\pi(P, X, Y) = ZPZ^T$

Table 2. Riemannian tools to perform RCG on $\mathcal{S}_{++}^d$. Here, $\operatorname{sym}(X) = \frac{1}{2}(X + X^T)$, $\operatorname{Exp}(\cdot)$ denotes the matrix exponential and $Z = (YX^{-1})^{\frac{1}{2}}$.

4. Proposed Method

In this section, we first introduce the most general form of our joint IDDL formulation and follow it up by providing simplifications and derivations for specific cases (such as for $\alpha = \beta = 0$).

4.1. Information Divergence & Dictionary Learning

Suppose we are given a set of SPD matrices $\mathcal{X} = \{X_1, X_2, \cdots, X_N\}$, $X_i \in \mathcal{S}_{++}^d$ along their associated labels $y_i \in \mathcal{L} = \{1, 2, \cdots, L\}$. Our goal is three-fold: (i) learn a dictionary $\mathbf{B} \in \mathcal{S}_{++}^d{}^{\times_n}$, a product of n SPD manifolds, (ii) learn an ABLD on each dictionary atom to best represent the given data for the task of classification, and (iii) learn a discriminative objective function on the encoded SPD matrices (in terms of $\mathbf{B}$ and the respective ABLDs) for the purpose of classification. These goals are formally captured in the IDDL objective proposed below. Let the k-th dictionary atom in $\mathbf{B}$ be B_k, then,

$$\begin{aligned}\text{IDDL} := \min_{\mathbf{B}>0, \boldsymbol{\alpha}>0, \boldsymbol{\beta}>0, W} & \sum_{i=1}^{N} f(\mathbf{v}_i, y_i; W) \quad (12)\\ \text{subject to } \mathbf{v}_i^k &= D^{(\boldsymbol{\alpha}_k, \boldsymbol{\beta}_k)}(X_i \parallel B_k),\end{aligned}$$

where the vector $\mathbf{v}_i \in \mathbb{R}^n$ denotes the encoding of X_i in terms of the dictionary, and $\mathbf{v}_i^k$ is the k-th dimension of this encoding. The function f parameterized by W learns a classifier on $\mathbf{v}_i$ according to the provided class labels y_i. While, there are several choices for f (*e.g.*, max-margin hinge-loss), we resort to a simple ridge regression objective in this paper. Thus, our f is defined as follows: suppose $\mathbf{h}_i \in \{0, 1\}^n$ is a one-off encoding of class labels (i.e., $\mathbf{h}_i^{y_i} = 1$, everywhere else zero), then

$$f(\mathbf{v}_i, y_i; W) = \frac{1}{2}\left\|\mathbf{h}_i - W\mathbf{v}_i\right\|^2 + \gamma\left\|W\right\|_F^2, \quad (13)$$

where $W \in \mathbb{R}^{L \times n}$ and γ is a regularization parameter. Note that a separate α_k, β_k for each dictionary atom is the most general form of our formulation. In our experiments, we explore simplified cases when these parameters are shared across the atoms.

4.2. Efficient Optimization

In this section, we propose efficient ways to solve the IDDL objective in (12). We propose to use a block-coordinate descent (BCD) scheme for optimization, in which each variable is updated alternately while fixing others. Going by the recent trends in Riemannian optimization for SPD matrices [8, 18], we use the Riemannian conjugate gradient (RCG) algorithm [1] for optimizing over each variable. As our objective is non-convex in its variables (except for W), convergence of BCD iterations to a global minima is not guaranteed. In Alg. 1, we detail out the meta-steps in our optimization scheme. We initialize the dictionary atoms and the divergence parameters as described in Section 6.3. Following that, we update the atoms, the divergence parameters, and classifier parameters in an alternating manner manner – that is, updating one variable whie fixing all others.

Recall from Section 3.3 that an essential ingredient in RCG is efficient computations of the Euclidean gradients of the objective with respect to the variables. In the following, we derive expressions for these gradients. Note that we assume that the dictionary atoms (*i.e.*, B_i) to be on an SPD manifold. Also w.l.o.g, we assume $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ belong to the non-negative orthant of the Euclidean space (for reasons in Section 3).

Input: $\mathcal{X}$, H, n
$\mathbf{B} \leftarrow \text{kmeans}(\mathcal{X}, n)$, $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \leftarrow \text{GridSearch}$;
repeat
  for $k = 1$ **to** n **do**
    $B_k \leftarrow \text{update\_B}(\mathcal{X}, W, \boldsymbol{\alpha}, \boldsymbol{\beta}, B_k)$; // use (18)
  end
  $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \leftarrow \text{update\_}\alpha\beta(\mathcal{X}, W, \mathbf{B}, \boldsymbol{\alpha}, \boldsymbol{\beta})$; // use (20)
  $W \leftarrow \text{update\_}W$; // using (21)
until *until convergence*;
return $\mathbf{B}, \boldsymbol{\alpha}, \boldsymbol{\beta}$

Algorithm 1: Block-Coordinate Descent for IDDL.

4.2.1 Gradients wrt B

As is clear from our formulation, only the k-th dimension of $\mathbf{v}_i$ involves B_k. To simplify the notations, let us assume

$$\zeta = -(\mathbf{h}_i - W\mathbf{v}_i)^T W, \tag{14}$$

and let ζ^k be its k-th dimension. Then we have (see the supplementary material for the details),

$$\nabla_{B_k} f := \zeta_i^k \nabla_{B_k} \left(D^{(\boldsymbol{\alpha}^k, \boldsymbol{\beta}^k)}(X_i \parallel B_k) \right). \tag{15}$$

Substituting for ABLD in (15) and rearranging the terms, we have:

$$\nabla_{B_k} f = \frac{1}{\boldsymbol{\alpha}_k \boldsymbol{\beta}_k} \nabla_{B_k} \log\det \left[\frac{\boldsymbol{\alpha}_k}{\boldsymbol{\beta}_k} \left(X_i^{-1} B_k \right)^{\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k} + \mathbf{I}_d \right] - \frac{1}{\boldsymbol{\beta}_k} B_k^{-1}. \tag{16}$$

Let $\theta_k = \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k$ and $\mathbf{r}_k = \frac{\boldsymbol{\alpha}_k}{\boldsymbol{\beta}_k}$. Further, let $Z_i = X_i^{-1}$. Then, the term inside the gradient in (16) simplifies to:

$$g(B_k; Z, \mathbf{r}_k, \theta_k) = \log\det \left[\mathbf{r}_k \left(Z B_k \right)^{\theta_k} + \mathbf{I}_d \right]. \tag{17}$$

Theorem 2 *Let $A, B \in \mathcal{S}_{++}^d$. Furthermore assume $p, q \geq 0$. We have*

$$\nabla_B \log\det \left[p \left(AB \right)^q + \mathbf{I}_d \right] = pqB^{-1}A^{-\frac{1}{2}} \left(A^{\frac{1}{2}} B A^{\frac{1}{2}} \right)^q \left(\mathbf{I}_d + p \left(A^{\frac{1}{2}} B A^{\frac{1}{2}} \right)^q \right)^{-1} A^{\frac{1}{2}}.$$

Proof *See the extended version of this paper [10].* ■

As such, the gradient $\nabla_{B_k} g$ is:

$$\nabla_{B_k} g = \mathbf{r}_k \theta_k B_k^{-1} Z_i^{-\frac{1}{2}} \left(Z_i^{\frac{1}{2}} B_k Z_i^{\frac{1}{2}} \right)^{\theta_k} \times \left(\mathbf{I}_d + \mathbf{r}_k \left(Z_i^{\frac{1}{2}} B_k Z_i^{\frac{1}{2}} \right)^{\theta_k} \right)^{-1} Z_i^{\frac{1}{2}}. \tag{18}$$

Combining (18) with (16), we have the expression for the gradient with respect to B_k.

Remark 1 *Computing $\nabla_{B_k} g$ for large datasets may become overwhelming. Let (U_i, Δ_i) be the Schur decomposition $Z_i^{\frac{1}{2}} B_k Z_i^{\frac{1}{2}}$ (which is faster than the eigenvalue decomposition [17]). With $\boldsymbol{\delta}_i = \text{diag}(\Delta_i)$, the gradient in (18) can be rewritten as:*

$$\nabla_{B_k} g = \mathbf{r}_k \theta_k B_k^{-1} \left(Z_i^{-\frac{1}{2}} U_i \right) \left[\text{diag} \left(\frac{\boldsymbol{\delta}_i^\theta}{1 + \mathbf{r}_k \boldsymbol{\delta}_i^{\theta_k}} \right) \right] \times \left(Z_i^{-\frac{1}{2}} U_i \right)^{-1}. \tag{19}$$

Compared to (18), this simplification reduces the number of matrix multiplications from 5 to 3 and matrix inversions from 2 to 1.

4.2.2 Gradients wrt $\boldsymbol{\alpha}_k$ and $\boldsymbol{\beta}_k$

For gradients with respect to $\boldsymbol{\alpha}_k$, we will use the form of ABLD given in (3), where λ_{ijk} is assumed to be the j-th generalized eigenvalue of X_i and dictionary atom B_k. Using the notations defined in (14), the gradient has the form:

$$\begin{aligned}\nabla_{\boldsymbol{\alpha}_k} f &= \zeta_i^k \sum_{j=1}^{d} \nabla_{\boldsymbol{\alpha}_k} \left[\frac{1}{\boldsymbol{\alpha}_k \boldsymbol{\beta}_k} \log \frac{\boldsymbol{\alpha}_k \lambda_{ijk}^{\boldsymbol{\beta}_k} + \boldsymbol{\beta}_k \lambda_{ijk}^{-\boldsymbol{\alpha}_k}}{\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k} \right] \\ &= \frac{\zeta_i^k}{\boldsymbol{\alpha}_k^2 \boldsymbol{\beta}_k} \sum_{j=1}^{d} \left\{ \frac{\boldsymbol{\alpha}_k \lambda_{ijk}^{\boldsymbol{\beta}^k} - \boldsymbol{\alpha}_k \boldsymbol{\beta}_k \lambda_{ijk}^{-\boldsymbol{\alpha}_k} \log \lambda_{ijk}}{\boldsymbol{\alpha}_k \lambda_{ijk}^{\boldsymbol{\beta}_k} + \boldsymbol{\beta}_k \lambda_{ijk}^{-\boldsymbol{\alpha}_k}} \right. \\ &\quad \left. - \frac{\boldsymbol{\alpha}_k}{\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k} - \log \frac{\boldsymbol{\alpha}_k \lambda_{ijk}^{\boldsymbol{\beta}_k} + \boldsymbol{\beta}_k \lambda_{ijk}^{-\boldsymbol{\alpha}_k}}{\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k} \right\}. \end{aligned} \tag{20}$$

The gradients wrt $\boldsymbol{\beta}_k$ from (20) can be derived using the dual symmetry property described in (7).

4.3. Closed Form for W

When fixing $\mathbf{B}, \boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, the objective reduces to the standard ridge regression formulation in W, which can be solved in closed form as:

$$W^* = HV^T(VV^T + \gamma \mathbf{I}_d)^{-1}, \tag{21}$$

where matrices V and H have $\mathbf{v}_i$ and $\mathbf{h}_i$ along their i-th column, for $i = 1, 2, \cdots, N$.

4.4. The Solution When $\boldsymbol{\alpha}, \boldsymbol{\beta} \to 0$

As alluded to earlier, ABLD is non-smooth at the origin and we need to resort to the limit of the divergence, which happens to be the natural Riemannian metric (AIRM). That is,

$$D^{(0,0)}(X_i \parallel B_k) = \left\| \text{Log}\left(X_i^{-\frac{1}{2}} B_k X_i^{-\frac{1}{2}} \right) \right\|_F^2. \tag{22}$$

Using the same ridge regression cost for f defined in (13), and using ζ_i^k defined in (14), we have the gradient using B_k as:

$$\nabla_{B_k} f = 2\zeta_i^k X_i^{-\frac{1}{2}} \text{Log}\,[P_{ik}] P_{ik}^{-1} X_i^{-\frac{1}{2}}, \tag{23}$$

where $P_{ik} = X_i^{-\frac{1}{2}} B_k X_i^{-\frac{1}{2}}$. Note that a simplification similar to (19) is also possible for (23).

5. Computational Complexity

We note that some of the terms in the gradients derived above could be computed offline (such as X_i^{-1}), and thus we omit those terms from our analysis. Using the simplifications depicted in (19) and Schur decomposition, gradient computation for each B_k takes $\mathcal{O}(Nd^3)$ flops. Using the gradient formulation in (20) for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, we need $\mathcal{O}(Ndn + Nd^3)$ flops. Computations of the closed form for W in (21) takes $\mathcal{O}(n^2(L+N) + n^3 + nLN)$. At test time, given that we have learned the dictionary and the parameters of the divergence, encoding a data matrix requires $\mathcal{O}(nd^3)$ flops, which is similar in complexity to the recent sparse coding schemes such as [8].

6. Experiments

In this section, we evaluate the performance of the IDDL scheme on eight computer vision datasets, which are known to benefit from SPD-based descriptors. Below, we provide details about all these datasets and the way SPD descriptors are obtained on them. We use the standard evaluation schemes reported previously on these datasets. In some cases, we use our own implementations of popular methods but strictly following the recommended settings.

6.1. Datasets

HMDB [27] and JHMDB [25] datasets: These are two popular action recognition benchmarks. The HMDB dataset consists of 51 action classes associated with 6766 video sequences, while JHMDB is a subset of HMDB with 955 sequences in 21 action classes. To generate SPD matrices on these datasets, we use the scheme proposed in [7], where we compute RBF kernel descriptors on the output of per-frame CNN class predictions (fc8) for each stream (RBF and optical flow) separately, and fusing these two SPD matrices into a single block-diagonal matrix per sequence. For the two-stream model, we use a VGG16 model trained on optical flow and RGB frames separately as described in [38]. Thus, our descriptors are of size 102×102 for HMDB and 42×42 for JHMDB.

SHREC 3D Object Recognition Dataset [31]: It consists of 15000 RGBD covariance descriptors generated from the SHREC dataset [31] by following [16]. SHREC consists of 51 3D object classes. The descriptors are of size 18×18. Similar to [8], we randomly picked 80% of the dataset for training and used the remaining for testing.

KTH-TIPS2 dataset [33] and Brodatz Textures [36]: These are popular texture recognition datasets. The KTH-TIPS dataset consists of 4752 images from 11 material classes under varying conditions of illumination, pose, and scale. Covariance descriptors of size 23×23 are generated from this dataset following the procedure in [19]. We use the standard 4-split cross-validation for our evaluations on this dataset. As for the Brodatz dataset, we use the relative pixel coordinates, image intensity, and image gradients to form 5×5 region covariance descriptors from 100 texture classes. Our dataset consists of 31000 SPD matrices, and we follow the procedure in [8] for our evaluation using an 80:20 rule as used in the RGBD dataset above.

Virus Dataset [29]: It consists of 1500 images of 15 different virus types. Similar to the KTH-TIPS, we use the

procedure in [19] to generate 29×29 covariance descriptors from this dataset and follow their evaluation scheme using three-splits.

Cancer Datasets [42]. Apart from these standard SPD datasets, we also report performances on two cancer recognition datasets from [42]. We use images from two types of cancers, namely (i) Breast cancer, consisting of binary classes (tissue is either cancerous or not) consisting of about 3500 samples, and (ii) Myometrium cancer, consisting of 3320 samples; we use covariance-kernel descriptors as described in [42] which are of size 8×8. We follow the 80:20 rule for evaluation on this dataset as well.

6.2. Experimental Setup

Since we present experiments on a variety of datasets and under various configurations, we summarize our main experiments first. There are three sets of experiments we conduct, namely (i) comparison of IDDL against other popular measures on SPD matrices, (ii) comparisons among various configurations of IDDL, and (iii) comparisons against state of the art approaches on the above datasets. For those datasets that do not have prescribed cross-validation splits, we repeat the experiments at least 5 times and average the performance scores. For our SVM-based experiments, we use a linear SVM on the log-Euclidean mapped SPD matrices.

6.3. Parameter Initialization

In all the experiments, we initialized the parameters of IDDL (*e.g.*, the initial dictionary) in a principle-way. We initialized the dictionary atoms by applying log-Euclidean K-Means. To initialize α and β, we recommend grid-search by fixing the dictionary atoms as above, or using Burg divergence (i.e., $\alpha = \beta = 1$). The regularization parameter λ is chosen using cross-validation.

6.4. Performance for Varying α, β

In this section, we study the influence of each of the components in our algorithm. In Figure 2, we plot a heatmap of the classification accuracy against changing α and β on the KTH-TIPS2 and Virus datasets. We fixed the size of dictionaries to 22 for the KTH TIPS and 30 for the Virus datasets. The plots reveal that the performance varies for different parameter settings, thus (along with the results in Table 4) substantiates that learning these parameters is a way to improve performance.

6.5. Convergence Study

In Figure 3, we plot the convergence of our objective against iterations. We also depict the BCD objective as contributed by the dictionary learning updates and the parameter learning. As is clear, most part of the decrement in objective happens when the dictionary is learned, which is not surprising given that it has the most number of variables to learn. For most datasets, we observe that the RCG converges in about 200-300 iterations. For additional experiments, refer [10].

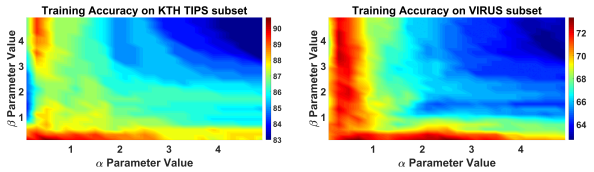


Figure 2. Parameter exploration for α and β on (left) KTH-TIPS2 and (right) VIRUS datasets fixing the number of dictionary atoms.

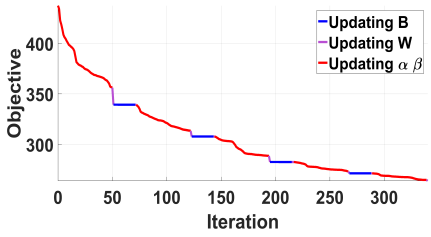


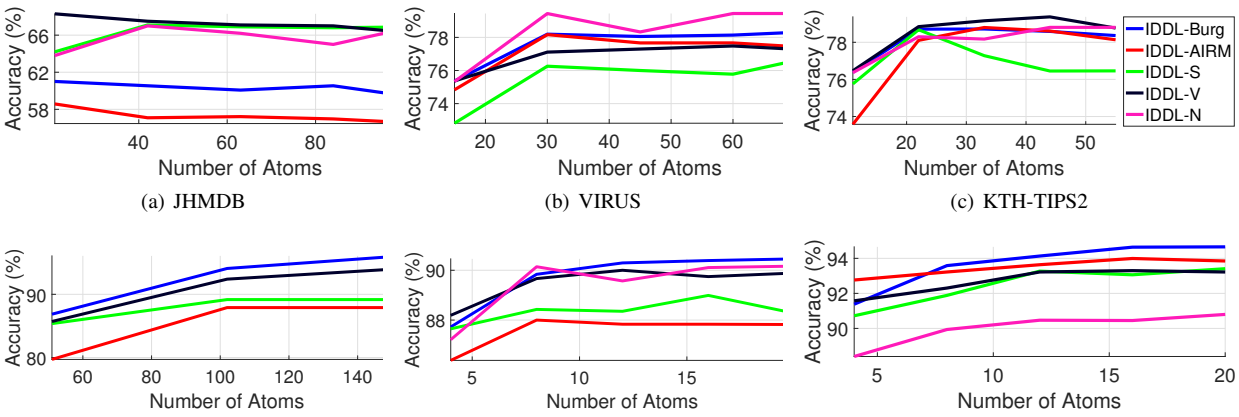
Figure 3. Convergence of the BCD optimization scheme for IDDL on the VIRUS dataset.

6.6. Comparisons to Variants of IDDL

In this section, we analyze various aspects of the performance of IDDL. Generally speaking, IDDL formulation is generic and customizable. For example, even though we formulated the problem as using a separate ABLD on each dictionary atom, it does not hurt to learn the same divergence over all atoms in some applications. To this end, we test the performance of three scenarios, namely (i) using a scalar α and β that is shared across all the dictionary atoms (which we call IDDL-S), (ii) a vector α and β, where we assume $\alpha = \beta$, but each dictionary atom can potentially have a distinct parameter pair (we call this case IDDL-V), and (iii) the most generic case where we could have α, β as vectors and they may not be equal, which we refer as IDDL-N. In Figure 4, we compare all these configurations on six of the datasets. We also include specific cases such as the Burg divergence ($\alpha = \beta = 1$) and the AIRM case ($\alpha = \beta = 0$) for comparisons (using the dictionary learning scheme proposed in Section 4.2.1). Our experiments show that IDDL-N and IDDL-V consistently perform well on almost all datasets.

6.7. Comparisons to Standard Measures

In this experiment, we compare the IDDL (see Figure 4) to the standard similarity measures on SPD matrices including log-Euclidean Metric [3], AIRM [37], and JBLD [9]. We report 1-NN classification performance on



(a) JHMDB (b) VIRUS (c) KTH-TIPS2

(d) RGB-D Objects (e) Breast Cancer (f) Myometrium Cancer

Figure 4. Comparisons between variants of IDDL for increasing number of dictionary atoms.

these baselines. In Table 4, we report the performance of these schemes. As a rule of thumb (and also supported empirically by cross-validation studies on our datasets), for a C-class problem, we chose $5C$ atoms in the dictionary. Increasing the size of the dictionary seems not helping in most cases. We also report a discriminative baseline by training a linear SVM on the log-Euclidean mapped SPD matrices. The results reported in Table 4 demonstrates the advantage of IDDL against the baselines, where the benefits can go over more than 10% in some cases (such as the JHMDB and virus).

Dataset — Method	LEML	kSP$_{LE}$	kSP$_{JBLD}$	kLLC	IDDL	Variant
JHMDB	58.85%	55.97%	44.40%	57.46%	**68.3%**	**V**
HMDB	52.15%	44.9%	28.43%	40.20%	**55.5%**	**N**
VIRUS	74.60%	68.00%	57.84%	70.91%	**78.39%**	**N**
BRODATZ	47.15%	55.00%	65.19%	70.00%	**74.10%**	**N**
KTH TIPS	79.25%	77.18%	69.92%	73.96%	**79.37%**	**V**
3D Object	87.56%	59.26%	72.45%	87.40%	**96.08%**	**Burg**
Breast Cancer	83.18%	76.34%	71.67%	82.32%	**90.46%**	**Burg**
Myometrium Cancer	90.94%	88.69%	86.80%	88.74%	**94.66%**	**Burg**

Table 3. Comparisons against state of the art. Last column shows the variant of IDDL that performed the best.

6.8. Comparisons to the State of the Art

We compare IDDL to the following popular methods that share similarities to our scheme, namely (i) Log-Euclidean Metric learning (LEML) [23], (ii) kernelized Sparse Coding [19] that uses log-Euclidean metric for sparse coding SPD matrices (kSP_{LE}), (iii) kernelized sparse coding using JBLD (kSP_{JBLD}), and kernelized locality constrained coding [18]. For IDDL, we chose the variant from Figure 4 that performed the best on the respective dataset (refer to the last column for the IDDL-variant). Our results are reported in Table 3. Again we observe that IDDL performs the best amongst all the competitive schemes, clearly demonstrating the advantage of learning the divergence and the dictionary. Note that comparisons are established by considering the same number of atoms for all schemes and fine-tuning the parameters of each algorithm (*e.g.*, the bandwidth of the RBF kernel in kSP_{JBLD}) using a validation subset of the training set. As for LEML, we increased the number of pairwise constraints until the performance hit a plateau.

Dataset — Classifier	LE 1-NN	AIRM 1-NN	JBLD 1-NN	SVM-LE	IDDL	Variant
JHMDB	52.99%	51.87%	52.24%	54.48%	**68.3%**	**V**
HMDB	29.30%	43.3%	46.3%	41.7%	**55.50%**	**N**
VIRUS	66.67%	67.89%	68.11%	68.00%	**78.39%**	**N**
BRODATZ	80.10%	80.50%	80.50%	**86.80%**	74.10%	**N**
KTH TIPS	72.05%	72.83%	72.87%	75.59%	**79.37%**	**V**
3D Object	97.4%	98.2%	95.6%	**98.9%**	96.08%	**Burg**
Breast Cancer	87.42%	80.00%	84.00%	87.71%	**90.46%**	**Burg**
Myometrium Cancer	80.87%	84.18%	93.20%	93.22%	**94.66%**	**Burg**

Table 4. Comparisons against 1-NN and SVM classification. Last column shows the variant of IDDL that worked best.

7. Conclusions

In this paper, we proposed a novel framework unifying the problem of dictionary learning and information divergence learning on SPD matrices; two problems that have been investigated separately so far. We leveraged on the recent advances in information geometry for this purpose, namely using the $\alpha\beta$-logdet divergence. We formulated an objective for jointly learning the divergence and the dictionary and showed that it can be solved efficiently using optimization methods on Riemannian manifolds. Experiments on eight computer vision datasets demonstrate superior performance of our approach against alternatives.

Acknowledgments: This material is based upon work supported by the National Science Foundation through grants #CNS-0934327, #CNS-1039741, #SMA-1028076, #CNS-1338042, #CNS-1439728, #OISE-1551059, and #CNS-1514626. AC is funded by the Australian Research Council Centre of Excellence for Robotic Vision (#CE140100016).

References

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009. 4, 5

[2] S.-i. Amari and H. Nagaoka. *Methods of information geometry*, volume 191. American Mathematical Soc., 2007. 2

[3] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Log-euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic resonance in medicine*, 56(2):411–421, 2006. 1, 2, 7

[4] A. Basu, I. R. Harris, N. L. Hjort, and M. Jones. Robust and efficient estimation by minimising a density power divergence. *Biometrika*, 85(3):549–559, 1998. 2

[5] T. Brox, J. Weickert, B. Burgeth, and P. Mrázek. Nonlinear structure tensors. *Image and Vision Computing*, 24(1):41–55, 2006. 1

[6] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *ECCV*, 2012. 1

[7] A. Cherian, P. Koniusz, and S. Gould. Higher-order pooling of CNN features via kernel linearization for action recognition. In *WACV*, 2017. 6

[8] A. Cherian and S. Sra. Riemannian dictionary learning and sparse coding for positive definite matrices. *IEEE Trans. on Neural Networks and Learning Systems*, 2016. 2, 4, 5, 6

[9] A. Cherian, S. Sra, A. Banerjee, and N. Papanikolopoulos. Jensen-bregman logdet divergence with application to efficient similarity search for covariance matrices. *PAMI*, 35(9):2161–2174, 2013. 2, 4, 7

[10] A. Cherian, P. Stanitsas, M. Harandi, V. Morellas, and N. Papanikolopoulos. Learning discriminative alpha-beta divergences for positive definite matrices (extended version). *CoRR*, –:–, 2017. 5, 7

[11] A. Cichocki and S.-i. Amari. Families of alpha-beta-and gamma-divergences: Flexible and robust measures of similarities. *Entropy*, 12(6):1532–1568, 2010. 2

[12] A. Cichocki, S. Cruces, and S.-i. Amari. Log-determinant divergences revisited: Alpha-beta and gamma log-det divergences. *Entropy*, 17(5):2988–3034, 2015. 2, 3

[13] A. Cichocki, R. Zdunek, A. H. Phan, and S.-i. Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009. 2

[14] I. S. Dhillon and S. Sra. Generalized nonnegative matrix approximations with bregman divergences. In *NIPS*, 2005. 2

[15] O. Dikmen, Z. Yang, and E. Oja. Learning the information divergence. *PAMI*, 37(7):1442–1454, 2015. 2

[16] D. Fehr. *Covariance based point cloud descriptors for object detection and classification*. PhD thesis, University Of Minnesota, 2013. 6

[17] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012. 5

[18] M. Harandi and M. Salzmann. Riemannian coding and dictionary learning: Kernels to the rescue. In *CVPR*, 2015. 1, 3, 4, 5, 8

[19] M. Harandi, M. Salzmann, and F. Porikli. Bregman divergences for infinite dimensional covariance matrices. In *CVPR*, 2014. 3, 6, 7, 8

[20] M. T. Harandi, M. Salzmann, and R. Hartley. From manifold to manifold: Geometry-aware dimensionality reduction for spd matrices. In *ECCV*, 2014. 2

[21] G. Hinton and S. Roweis. Stochastic neighbor embedding. In *NIPS*, 2002. 2

[22] Z. Huang and L. Van Gool. A Riemannian network for SPD matrix learning. *CoRR arXiv:1608.04233*, 2016. 1

[23] Z. Huang, R. Wang, S. Shan, X. Li, and X. Chen. Log-Euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *ICML*, 2015. 2, 8

[24] C. Ionescu, O. Vantzos, and C. Sminchisescu. Matrix backpropagation for deep networks with structured layers. In *ICCV*, 2015. 1

[25] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *ICCV*, 2013. 6

[26] R. Kompass. A generalized divergence measure for nonnegative matrix factorization. *Neural computation*, 19(3):780–791, 2007. 2

[27] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: a large video database for human motion recognition. In *ICCV*, 2011. 6

[28] B. Kulis, M. Sustik, and I. Dhillon. Learning low-rank kernel matrices. In *ICML*, 2006. 2, 4

[29] G. Kylberg, M. Uppström, K. Hedlund, G. Borgefors, and I. Sintorn. Segmentation of virus particle candidates in transmission electron microscopy images. *Journal of microscopy*, 245(2):140–147, 2012. 6

[30] J. Lafferty. Additive models, boosting, and inference for generalized divergences. In *Proc. conf. on Computational learning theory*, 1999. 2

[31] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view RGB-D object dataset. In *ICRA*, 2011. 6

[32] P. Li, Q. Wang, W. Zuo, and L. Zhang. Log-Euclidean kernels for sparse representation and dictionary learning. In *ICCV*, 2013. 3

[33] P. Mallikarjuna, A. T. Targhi, M. Fritz, E. Hayman, B. Caputo, and J.-O. Eklundh. The KTH-TIPS2 database, 2006. 6

[34] M. Mihoko and S. Eguchi. Robust blind source separation by beta divergence. *Neural computation*, 14(8):1859–1886, 2002. 2

[35] M. Moakher and P. G. Batchelor. Symmetric positive-definite matrices: From geometry to applications and visualization. In *Visualization and Processing of Tensor Fields*, pages 285–298. Springer, 2006. 2, 4

[36] T. Ojala, M. Pietikäinen, and D. Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59, 1996. 6

[37] X. Pennec, P. Fillard, and N. Ayache. A Riemannian framework for tensor computing. *IJCV*, 66(1):41–66, 2006. 1, 2, 3, 4, 7

[38] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014. 6

[39] U. Şimşekli, A. T. Cemgil, and B. Ermiş. Learning mixed divergences in coupled matrix and tensor factorization models. In *ICASSP*, 2015. 2

[40] R. Sivalingam, D. Boley, V. Morellas, and N. Papanikolopoulos. Tensor sparse coding for region covariances. In *ECCV*, 2010. 2

[41] R. Sivalingam, V. Morellas, D. Boley, and N. Papanikolopoulos. Metric learning for semi-supervised clustering of region covariance descriptors. In *ICDSC*, 2009. 2

[42] P. Stanitsas, A. Cherian, X. Li, A. Truskinovsky, V. Morellas, and N. Papanikolopoulos. Evaluation of feature descriptors for cancerous tissue recognition. In *ICPR*, 2016. 7

[43] D. B. Thiyam, S. Cruces, J. Olias, and A. Cichocki. Optimization of alpha-beta log-det divergences and their application in the spatial filtering of two class motor imagery movements. *Entropy*, 19(3):89, 2017. 2

[44] O. Tuzel, F. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. In *ECCV*, 2006. 1

[45] L. Wang, J. Zhang, L. Zhou, C. Tang, and W. Li. Beyond covariance: Feature representation with nonlinear kernel matrices. In *ICCV*, 2015. 1

[46] R. Wang, H. Guo, L. S. Davis, and Q. Dai. Covariance discriminative learning: A natural and efficient approach to image set classification. In *CVPR*, 2012. 2

[47] Y. Xie, J. Ho, and B. Vemuri. On a nonlinear generalization of sparse coding and dictionary learning. In *ICML*, 2013. 2