Jensen-Bregman LogDet Divergence with Application to Efficient Similarity Search for Covariance Matrices

Anoop Cherian, *Student Member, IEEE,* Suvrit Sra, Arindam Banerjee and Nikolaos Papanikolopoulos, *Fellow, IEEE*

Covariance matrices have found success in several computer vision applications, including activity recognition, visual surveillance, and diffusion tensor imaging. This is because they provide an easy platform for fusing multiple features compactly. An important task in all of these applications is to compare two covariance matrices using a (dis)similarity function, for which the common choice is the Riemannian metric on the manifold inhabited by these matrices. As this Riemannian manifold is not flat, the dissimilarities should take into account the curvature of the manifold. As a result such distance computations tend to slow down, especially when the matrix dimensions are large or gradients are required. Further, suitability of the metric to enable efficient nearest neighbor retrieval is an important requirement in the contemporary times of big data analytics. To alleviate these difficulties, this paper proposes a novel dissimilarity measure for covariances, the *Jensen-Bregman LogDet Divergence* (JBLD). This divergence enjoys several desirable theoretical properties, at the same time is computationally less demanding (compared to standard measures). Utilizing the fact that the square-root of JBLD is a metric, we address the problem of efficient nearest neighbor retrieval on large covariance datasets via a metric tree data structure. To this end, we propose a K-Means clustering algorithm on JBLD. We demonstrate the superior performance of JBLD on covariance datasets from several computer vision applications.

Index Terms—Region Covariance Descriptors, Bregman Divergence, Image Search, Nearest Neighbor Search, LogDet Divergence, Video Surveillance, Activity Recognition

1 INTRODUCTION

Recent times have witnessed a steep increase in the utilization of structured data in several computer vision and machine learning applications, where instead of vectors, one uses richer representations of data such as graphs, strings, or matrices. A class of such structured data that has been gaining importance in computer vision is the class of Symmetric Positive Definite (SPD) matrices, specifically as covariance matrices. These matrices which offer a compact fusion of multiple features and they are by now preferred data representations in several applications.

To bring out the importance of covariance matrices in computer vision, we concisely review a few applications in which these data descriptors have found immense success. SPD matrices are fundamental objects in Diffusion Tensor Imaging for mapping biological tissue structures, with applications to the diagnosis of neuro-psychiatric illnesses including Alzheimer's disease, brain atrophy, and dementia [1]–[3]. Covariances provide a convenient platform for fusing multiple features, are robust to static noise, and can be easily made invariant to image affine transformations, illumination changes or changes in camera parameters. As a result they are used extensively in multi-camera object tracking applications [4], [5]. Other important applications of covariances include but are not limited to human detection [6], image segmentation [7], texture segmentation [8], robust face recognition [9], emotion recognition [10], and human action recognition [11]. Application of covariances as data descriptors is not limited to computer vision; an example is speech recognition [12].

These successful applications are however burdened by a common problem: whenever distance or similarity computations with covariances are required, the corresponding algorithms tend to slow down. This is because, covariances do not conform to the Euclidean geometry, but rather form a Riemannian manifold. Data points on this manifold are no more connected by straight lines, but rather geodesics along the curvature of the manifold. As a result, computing similarity between covariance matrices is non-trivial. But the choice

• *A. Cherian, A. Banerjee and N. Papanikolopoulos are with the Dept. of Computer Science and Engineering at the University of Minnesota, Minneapolis, MN-55455. Their emailids are: {cherian,banerjee,npapas}@cs.umn.edu respectively.*
• *S. Sra is with the Max Planck Institute for Intelligent Systems, Tüebingen, Germany. His emailid is suvrit.sra@tuebingen.mpg.de.*

of similarity measure is crucial, especially for a key task such as Nearest Neighbor (NN) retrieval, which is a building block in countless applications. For example, for tracking the appearance of people in video surveillance, the number of database points can lie in the millions, and without efficient similarity computation both NN retrieval and the subsequent tracking are severely disadvantaged. Standard NN retrieval techniques such as locality sensitive hashing [13] cannot be directly applied to covariance datasets without ignoring the manifold structure, resulting in poor retrieval accuracy.

Driven by these concerns we take a closer look at similarity computation on covariances by introducing the Jensen-Bregman LogDet Divergence (JBLD). We discuss theoretical properties of JBLD and then apply it to the task of rapid NN retrieval on several image databases. Experiments against state-of-the-art techniques show the advantages afforded by JBLD. At this point, we would like to acknowledge that the Riemannian framework is still useful for computing various intrinsic statistics on covariances that are required in several applications [14].

Before proceeding, let us briefly describe our notation. At places where the dimensionality of the matrix is unimportant, an SPD matrix X might be introduced as $X > 0$. The set $\mathcal{S}^d$ denotes the space of $d \times d$ symmetric matrices and $\mathcal{S}^d_{++}$ denotes the set of Symmetric Positive Definite (SPD) matrices. We use $|\ |$ to denote matrix determinant, Tr denotes the trace and $\|\ \|_F$ the Frobenius norm. Also, $\mathcal{I}$ refers to a $d \times d$ identity matrix. For two SPD matrices X, Y, the inequality $X > Y$ means $X - Y$ is SPD; the notation $X \otimes Y$ denotes the usual Kronecker product of X and Y.

2 Related Work

We recall some standard similarity measures for covariance matrices. The simplest but naive approach is to view $d \times d$ covariance matrices as vectors in $\mathbb{R}^{d(d+1)/2}$, whereby the standard (dis)similarity measures of Euclidean space can be used (e.g., ℓ_p-distance functions, etc.). Recall that covariance matrices, due to their positive definiteness structure, belong to a special category of symmetric matrices and form a Riemannian manifold (which is a differentiable manifold associated with a suitable Riemannian metric). Euclidean distances on vectorized covariances ignore this manifold structure leading to poor accuracy [15], [16]. In addition, under this measure symmetric matrices with non-positive eigenvalues are at finite distances to positive definite covariances. This is unacceptable for a variety of applications, e.g. DTMRI [15].

A more suitable choice is to incorporate the curvature of the Riemannian manifold and use the corresponding geodesic length along the manifold surface as the distance metric. This leads to the *Affine Invariant Riemannian Metric (AIRM)* [14], [17] which is defined as follows: For two SPD matrices X and Y,

$$D_R(X,Y) := \|\log(X^{-1/2} Y X^{-1/2})\|_{\mathrm{F}}, \tag{1}$$

where $\log(\cdot)$ is the *principal* matrix logarithm. This metric enjoys several useful theoretical properties, and is perhaps the most widely used similarity measure for covariance matrices. As is clear from (1), symmetric matrices with nonpositive eigenvalues are at infinite distances. The metric is invariant to inversion and similarity transforms. Other properties of this metric can be found in [17]. Computationally, this metric can be unattractive as it requires eigenvalue computations or sometimes even matrix logarithms, which for larger matrices cause significant slowdowns. A few examples of such applications using large covariances are: face recognition [9] (40×40), and emotion recognition [10] (30×30).

Amongst the many measures that have been proposed to replace AIRM, a closely related one is the *Log-Euclidean Riemannian Metric* (LERM). Considering the log-Euclidean mapping $\log : \mathcal{S}^d_{++} \to \mathcal{S}^d$, Arsigny et al. [15] introduce the LERM as:

$$D_{le}(X,Y) := \|\log(X) - \log(Y)\|_{\mathrm{F}}. \tag{2}$$

This metric maps SPD matrices to a flat Riemannian space (of zero curvature) so that the ordinary Euclidean distances can be used. The metric is easy to compute, and preserves some of the important properties of the AIRM (such as invariance to inversion and similarity transforms). In addition, from a practical point of view, since this metric untangles the two constituent matrices from their generalized eigenvalues, the logarithms on each of these matrices can be evaluated *offline*, gaining a computational edge over AIRM. As a result, LERM has found many applications in visual tracking [18], stereo matching [19], etc. On the negative side, computing matrix logarithms can dramatically increase the computational costs. The flattening of the manifold as in LERM often leads to less accurate distance computations, affecting application performance. A more important problem that one encounters when using LERM is that its moments (such as Hessian, etc.) do not have closed forms. Moreover, it is computationally difficult even to approximate these moments due to the need to find derivatives of matrix logarithms. The following proposition shows that LERM is a lower bound to AIRM. This result will come useful later in this paper.

Proposition 1. *For $X, Y \in \mathcal{S}^d_{++}$, we have: $D_{le}(X,Y) \leq D_R(X,Y)$. Further, the equality holds only when X and Y commute.*

Proof: See Theorem 6.1.4 in [17]. □

Similar to our approach, there have been previous attempts to use symmetrized *f-divergences* from information theory for developing distances on SPD matrices. One such idea is to view the SPD matrices as being the covariances associated with zero-mean Gaussian distributions [16], and then use the symmetrized KL-Divergence Metric (KLDM) (popularly known as Jeffrey's Kullback-Leibler divergence) as the distance between the distributions. This leads to the following definition of KLDM[1]:

$$D^2_{jkl}(X,Y) := \frac{1}{2} \mathrm{Tr}\left(X^{-1}Y + Y^{-1}X - 2\mathcal{I}\right). \tag{3}$$

This measure does not require matrix eigenvalue computations, or logarithms, and at the same time enjoys many of the

1. We will use the squared form of KLDM for consistency with other metrics

properties of AIRM. On the negative side, the measure requires inversion of the constituent covariances, which can be slow (or can even lead to instability when the data matrices are poorly conditioned). A bigger concern is that KLDM can in fact *overestimate* the Riemannian metric as the following proposition shows and thus can lead to poor accuracy.

Proposition 2. *There exist X and $Y \in \mathcal{S}_{++}^d$ such that $D_{jkl}(X,Y) > D_R(X,Y)$.*

Proof: Let v_i be the ith eigenvalue of $X^{-1}Y$. Since v_i is always positive, we can write $v_i = e^{u_i}$ for $u_i \in \mathbb{R}$. Then from the definitions of KLDM and AIRM, we have:

$$D_{jkl}^2(X,Y) = \sum_{i=1}^{d} \left(\frac{e^{u_i} + e^{-u_i}}{2} \right) - 1$$
$$= \frac{D_R^2(X,Y)}{2} \sum_{i=1}^{d} \left(1 + 2\frac{u_i^2}{4!} + \cdots \right) - 1.$$

For a suitable choice of u_i, we have the desired result. □

A distance on the Cholesky factorization of the SPD matrices is presented in [20]. The idea is as follows: suppose $X = L_1 L_1^T$ and $Y = L_2 L_2^T$ represent the Cholesky decomposition of X and Y respectively, with lower triangular matrices L_1 and L_2, then the Cholesky distance is defined as:

$$D_C(X,Y) = \|L_1 - L_2\|_{\mathrm{F}}. \tag{4}$$

Other similarity measures on covariance matrices may be found in [21]. Despite their easy formulations and properties similar to those of AIRM, the above distances have not been very popular in SPD matrix based applications due to their poor accuracy (as our experiments also demonstrate).

In contrast to all these metrics, the similarity metric that we propose in this paper is much faster to compute, as it depends only on the determinant of the input matrices, and thus no eigenvalue computations are required. Moreover, it turns out to be empirically also very effective.

We note that NN retrieval for covariance matrices itself is still an emerging area, so literature on it is scarce (though for vector spaces, Bregman divergences have been used for several computer vision applications [22], [23]). In [24], an attempt is made to adapt NN techniques from vector spaces to non-Euclidean spaces, while [25] proposes an extension of the spectral hashing techniques to covariance matrices. However, both these techniques are based on a Euclidean embedding of the Riemannian manifold through the tangent spaces, and then using LERM as an approximation to the true similarity.

3 Jensen-Bregman LogDet Divergence

We first recall some basic definitions and then present our similarity measure: the *Jensen-Bregman LogDet Divergence (JBLD)*. Concurrent to our work, a one parameter generalization of this divergence has been recently discussed in [26]; our study arose independently of that paper. We alert the reader that JBLD should not be confused with its asymmetric cousin: the so-called LogDet divergence [27].

At the core of our discussion lies the *Bregman Divergence* $d_\phi : S \times \mathrm{relint}(S) \to [0,\infty)$, which is defined as

$$d_\phi(x,y) := \phi(x) - \phi(y) - \langle x - y, \nabla\phi(y) \rangle, \tag{5}$$

where $\phi : S \subseteq \mathbb{R}^d \to \mathbb{R}$ is a strictly convex function of Legendre-type on $\mathrm{int}(\mathrm{dom}\, S)$ [28]. From (5) the following properties of $d_\phi(x,y)$ are apparent: strict convexity in x; asymmetry; non-negativity; and definiteness (i.e., $d_\phi = 0$, iff $x = y$). Bregman divergences enjoy a host of useful properties [28], [29], but often their lack of symmetry and sometimes their lack of triangle-inequality can prove to be hindrances. Consequently, there has been substantial interest in considering symmetrized versions such as *Jensen-Bregman* divergences [30],

$$J_\phi(x,y) := \frac{1}{2}\big(d_\phi(x,s) + d_\phi(s,y)\big), \tag{6}$$

where $s = (x+y)/2$.

Both (5) and (6) can be naturally extended to matrix divergences (over Hermitian matrices) by composing the convex function ϕ with the eigenvalue map λ, and replacing the inner-product in (5) by the trace. We focus on a particular matrix divergence, namely the *Jensen-Bregman LogDet Divergence*, which is defined for X, Y in $\mathcal{S}_{++}^d$ by

$$J_{\ell d}(X,Y) := \log \left| \frac{X+Y}{2} \right| - \frac{1}{2}\log|XY|. \tag{7}$$

where $|\cdot|$ denotes the determinant; this divergence is obtained from the matrix version of (6) by using $\phi(X) = -\log|X|$ as the seed function.

3.1 Properties

For $X, Y, Z \in \mathcal{S}_{++}^d$ and invertible matrices A and B, we have the following properties (see [31] for details and proofs):

1) $J_{\ell d}(X,Y) \geq 0$ (nonnegativity)
2) $J_{\ell d}(X,Y) = 0$ iff $X = Y$ (definiteness)
3) $J_{\ell d}(X,Y) = J_{\ell d}(Y,X)$ (symmetry)
4) $\sqrt{J_{\ell d}(X,Y)} \leq \sqrt{J_{\ell d}(X,Z)} + \sqrt{J_{\ell d}(Z,Y)}$ (triangle inequality; see [31])
5) $J_{\ell d}(AXB, AYB) = J_{\ell d}(X,Y)$ (affine invariance)
6) $J_{\ell d}(X^{-1}, Y^{-1}) = J_{\ell d}(X,Y)$ (invariance to inversion).

Theorem 3 (Non-Convexity)**.** *Assuming $X, Y \in \mathcal{S}_{++}^d$, for a fixed Y, $J_{\ell d}(X,Y)$ is convex for $X \leq (1+\sqrt{2})Y$ and concave for $X \geq (1+\sqrt{2})Y$.*

Proof: Taking the second derivative of $J_{\ell d}(X,Y)$ with respect to X, we have

$$\nabla_X^2 J_{\ell d}(X,Y) = -(X+Y)^{-1} \otimes (X+Y)^{-1} + \frac{X^{-1} \otimes X^{-1}}{2}. \tag{8}$$

This expression is positive definite for $X \leq (1+\sqrt{2})Y$ and negative definite for $X \geq (1+\sqrt{2})Y$. □

3.2 Nearest Isotropic Matrix

As we alluded to earlier, diffusion tensor imaging is the process of mapping diffusion of water molecules in the brain tissues and helps in the diagnosis of neurological disorders *non-invasively*. When the tissues have an internal fibrous structure, water molecules in these tissues will diffuse rapidly in directions aligned with this structure. Symmetric positive definite matrices in this field- prove useful in the analysis of such diffusion patterns [1]. *Anisotropic index* is a useful quantity that is often used in this area [16], which is the distance of a given SPD matrix from its Nearest Isotropic Matrix (NIM). Mathematically, the NIM $\alpha\mathcal{I}$ ($\alpha > 0$) from a given tensor $P > 0$ with respect to a distance measure $\mathcal{D}(.,.)$ is defined as:

$$\min_{\alpha>0} \mathcal{D}(\alpha\mathcal{I}, P). \tag{9}$$

Below, we investigate this facet of JBLD (see [16] for details on how to compute α for other AIRM, LERM and KLDM).

Theorem 4. *Suppose $P \in \mathcal{S}_{++}^d$ and let $S = \alpha\mathcal{I}$ be such that $J_{\ell d}(P, S)$ is convex (see Theorem 3). Then the NIM to P is the minimum positive root of the following equation:*

$$\begin{aligned} p(\alpha) :=& d\alpha^d + (d-2)\sum_i \lambda_i \alpha^{d-1} + (d-4)\sum_{i,j,i\neq j} \lambda_i\lambda_j \alpha^{d-2} \\ &+ \cdots + (2-d)\sum_i \prod_{i\neq j} \lambda_j \alpha - d\prod_i \lambda_i = 0, \end{aligned} \tag{10}$$

where $\lambda_i, i = 1, 2, \cdots, d$ are the eigenvalues of P.

Proof: Using the definition of $J_{\ell d}$ in (9), and applying the assumption that $J_{\ell d}$ is convex, at optimality we have $\frac{\partial J_{\ell d}(\alpha\mathcal{I}, P)}{\partial \alpha} = 0$. This leads to:

$$\frac{1}{\alpha} = \frac{2}{d}\sum_{i=1}^{d} \frac{1}{\alpha + \lambda_i}.$$

Rearranging the terms, we have the polynomial equation in (10). Since the coefficient of α^{d-1} is always positive (for $d > 2$), there must always exist at least one positive root. □

Corollary 5. *When $d = 2$, we have $\alpha = \sqrt{|P|}$, which is the same as NIM for AIRM.*

DTMRI generally uses 3×3 SPD matrices. Since $p(\alpha)$ is a cubic in this case, one can obtain its roots in closed form. Nevertheless, we show below a useful property that helps bracket its positive root.

Lemma 6. *Let $P \in \mathcal{S}_{++}^d$ and suppose $\|P\|_2 < 1$, then*

$$\frac{1 + \operatorname{Tr}(P)/d}{1 + \operatorname{Tr}(P^{-1})/d} > |P|. \tag{11}$$

Proof: Suppose $P \in \mathcal{S}_{++}^d$ and $\|P\|_2 < 1$, then $\operatorname{Tr}(P) < d$. Suppose $\lambda_i, i = 1, 2, \cdots, d$ represents the eigenvalues of P, we have the following to prove from the lemma:

$$\frac{d + \operatorname{Tr}(P)}{d|P| + \sum_i \prod_{j\neq i} \lambda_i\lambda_j} > 1. \tag{12}$$

Since $|P| < \operatorname{Tr}(P)/d$ (due to AM-GM inequality) and since $\sum_i \prod_{j\neq i} \lambda_i\lambda_j < d$, we have the desired result. □

Theorem 7. *Let $P \in \mathcal{S}_{++}^3$, and if $S = \alpha\mathcal{I}, \alpha > 0$ is the NIM to P, then $\alpha \in (0, 1)$.*

Proof: Substituting $d = 3$ in (10), we have the following third degree polynomial equation:

$$p(\alpha) := 3\alpha^3 + \operatorname{Tr}(P)\alpha^2 - |P|\operatorname{Tr}(P^{-1})\alpha - 3|P| = 0. \tag{13}$$

Analyzing the coefficients of $p(\alpha)$ shows that only one root is positive. Now, we have $p(0) < 0$. Applying Lemma 6, we have $p(1) > 0$, which concludes that the smallest positive root lies in $(0, 1)$. □

3.3 Connections to Other Metrics

We summarize below some connections that $J_{\ell d}$ has with the standard metrics on covariances.

Theorem 8. *Let $X, Y > 0$. Then,*

$$\begin{aligned} &(i) \quad J_{\ell d}(X, Y) \leq D_R^2(X, Y) \\ &(ii) \quad J_{\ell d}(X, Y) \leq D_{jkl}^2(X, Y) \end{aligned}$$

Proof: Let $v_i = \lambda_i(XY^{-1})$. Since X, $Y \in \mathcal{S}_{++}^d$, the eigenvalues v_i are also positive, whereby we can write each $v_i = e^{u_i}$ for some $u_i \in \mathbb{R}$. Using this notation, the AIRM may be rewritten as $D_R(X, Y) = \|u\|_2$, and the JBLD as

$$J_{\ell d}(X, Y) = \sum\nolimits_{i=1}^{d} (\log(1 + e^{u_i}) - u_i/2 - \log 2), \tag{14}$$

where the equation follows by observing that $J_{\ell d}(X, Y) = \log|I + XY^{-1}| - \frac{1}{2}\log|XY^{-1}| - \log 2^d$.

To prove inequality (i), consider the function $f(u) = u^2 - \log(1 + e^u) + u/2 + \log 2$. This function is convex since its second derivative

$$f''(u) = 2 - \frac{e^u}{(1 + e^u)^2},$$

is clearly nonnegative. Moreover, f attains its minimum at $u^* = 0$, as is immediately seen by solving the optimality condition $f'(u) = 2u - e^u/(1 + e^u) + 1/2 = 0$. Thus, $f(u) \geq f(u^*) = 0$ for all $u \in \mathbb{R}$, which in turn implies that

$$\sum\nolimits_{i=1}^{d} f(u_i) = D_R^2(X, Y) - J_{\ell d}(X, Y) \geq 0. \tag{15}$$

Similarly to prove inequality (ii), consider the function $g(u) = D_{jkl}^2 - J_{\ell d}$, which expands to:

$$g(u) = \frac{1}{2}(e^u + \frac{1}{e^u}) - \log(1 + e^u) + \frac{u}{2} + \log 2 - 1. \tag{16}$$

Going by the same steps as before, it is straight-forward to show that $g(u)$ is convex and attains its minimum when $u = 0$, proving the inequality. □

Theorem 9. *If $0 < mI \leq X, Y \leq MI$, then*

$$D_R^2(X, Y) \leq 2\log(M/m)(J_{\ell d}(X, Y) + \gamma), \tag{17}$$

where $\gamma = d\log 2$.

Proof: Observe that

$$\sum_{i=1}^{d}(\log(1+e^{u_i})-u_i/2-\log 2)\geq\sum_{i=1}^{d}(|u_i|/2-\log 2),$$

which implies the bound

$$J_{\ell d}(X,Y)+d\log 2\geq\frac{1}{2}\|u\|_1. \tag{18}$$

Since $u^Tu\leq\|u\|_\infty\|u\|_1$ (Hölder's inequality), using (18) we immediately obtain the bound

$$D_R^2(X,Y)=\|u\|_2^2\leq 2\|u\|_\infty(J_{\ell d}+\gamma), \tag{19}$$

where $\gamma=d\log 2$. But $mI\preceq X,Y\preceq MI$ implies that $\|u\|_\infty\leq\log(M/m)$, which concludes the proof. □

Our next result touches upon a condition when $J_{\ell d}<D_{le}^2$. A more general treatment of this relationship is outside the scope of this paper, mainly because the Hessian of D_{le} do not have closed forms[2].

Theorem 10. *If $X,Y\in\mathcal{S}_{++}^d$ commute, then $J_{\ell d}(X,Y)\leq D_{le}^2(X,Y)$.*

Proof: We use the fact that when X,Y commute, $D_{le}(X,Y)=D_R(X,Y)$ (See Proposition 1). Now, using the connection between AIRM and JBLD (refer Theorem 8), we have the result. □

3.4 JBLD Geometry

In Figure 1, we plot the three dimensional balls (isosurfaces) associated with JBLD for various radii (0.1, 0.5 and 1) and centered at the identity tensor. We also compare the JBLD ball with the isosurfaces of Frobenius distance, AIRM, and KLDM. As expected, the Frobenius distance is isotropic, while AIRM and KLDM induce non-spherical convex balls. Against these plots, and as was pointed by Theorem 3, the isosurfaces of JBLD are convex in some range and become concave as the radius increases.

3.5 Computational Advantages

The greatest advantage of $J_{\ell d}$ against the Riemannian metric is its computational speed: $J_{\ell d}$ requires only computation of determinants, which can be done rapidly via three Cholesky factorizations (for $X+Y$, X and Y), each at a cost of $(1/3)d^3$ flops [32]. Computing D_R on the other hand requires generalized eigenvalues, which can be done for positive-definite matrices in approximately $4d^3$ flops. Thus, in general $J_{\ell d}$ is much faster (see also Table 1). The computational advantages of $J_{\ell d}$ are much more impressive when comparing evaluation of gradients[3]. Table 2 shows that computing $\nabla J_{\ell d}$ can even be more than 100 times faster than ∇D_R. This difference ca be critical for NN retrieval, or more generally when using any algorithm that depends on gradients of the similarity measure, e.g., see [33] and the references therein. Table 3 provides a summary of the various metrics, their gradients and computational complexities.

2. Due to the need to take derivatives of matrix logarithms

3. The computation of $J_{\ell d}$ for matrices over $d=13$ was seen to be faster when the determinants were computed using the Cholesky decomposition.

d	D_R	$J_{\ell d}$
5	$.025\pm.012$	$.030\pm.007$
10	$.036\pm.005$	$.040\pm.009$
15	$.061\pm.002$	$.050\pm.004$
20	$.085\pm.006$	$.061\pm.009$
40	$.270\pm.332$	$.123\pm.012$
80	$1.23\pm.055$	$.393\pm.050$
200	$8.198\pm.129$	$2.223\pm.169$
500	$77.311\pm.568$	22.186 ± 1.223
1000	492.743 ± 15.519	119.709 ± 1.416

TABLE 1
Average times (millisecs/trial) to compute function values; computed over 10,000 trials to reduce variance.

d	$\nabla_X D_R^2(X,Y)$	$\nabla_X J_{\ell d}(X,Y)$
5	$0.798\pm.093$	$.036\pm.009$
10	$2.383\pm.209$	$.058\pm.021$
20	$7.493\pm.595$	$.110\pm.013$
40	24.899 ± 1.126	$.270\pm.047$
80	99.486 ± 5.181	$.921\pm.028$
200	698.873 ± 39.602	8.767 ± 2.137
500	6377.742 ± 379.173	94.837 ± 1.195
1000	40443.059 ± 2827.048	622.289 ± 37.728

TABLE 2
Average times (millisecs/trial) to compute gradients; computed over 1000 trials to reduce variance.

4 Fast Nearest Neighbor Retrieval

Now we turn to the key application that motivated us to investigate $J_{\ell d}$: Nearest Neighbor (NN) retrieval for covariance matrices. Here, we have a dataset $\{S_1,\ldots,S_n\}$ of $d\times d$ covariance matrices that we must organize into a data structure to facilitate rapid NN retrieval. Towards this end, we choose to use the metric tree data structure as we this allows us to show the performance on an exact NN algorithm for covariances; and for this which approximations can be easily effected for faster searches. A key component of the metric tree is a procedure to partition the data space into mutually exclusive clusters, so that heuristics such as branch and bound can be applied to prune clusters that are guaranteed not to occupy candidate neighbors to a query. To this end, we derive below a K-Means algorithm on $J_{\ell d}$ which will later be used to build the metric tree on covariances.

4.1 K-Means with $J_{\ell d}$

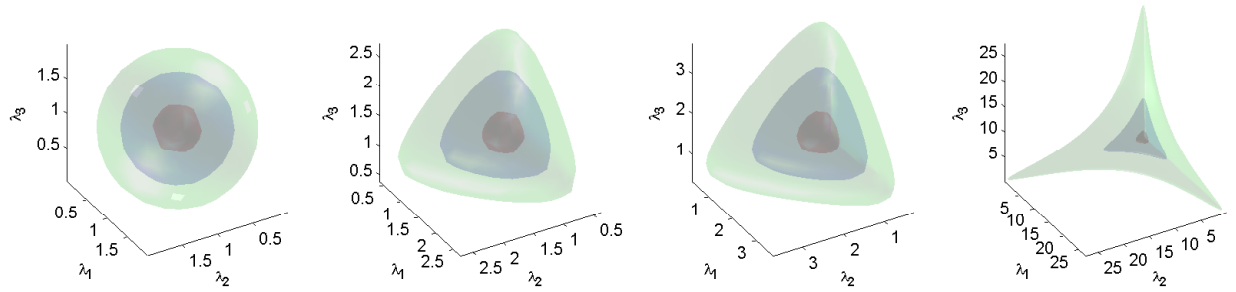
Let $S_1,S_2,\cdots,S_n$ be the input covariances that must be clustered. A standard K-Means algorithm gives rise to the following optimization problem:

$$\min_{C_1,C_2,\cdots,C_K}\sum_{k=1}^{K}\sum_{S\in C_k}J_{\ell d}(X_k,S), \tag{20}$$

where X_k is the *centroid* of cluster C_k. As usual, we can alternate between the centroid computation and the clustering stages to minimize (20). The only significant step is the computation of the centroid for the kth cluster; this can be written as:

$$F:=\min_{X_k}\sum_{S\in C_k}J_{\ell d}(X_k,S) \tag{21}$$

$$:=\min_{X_k}\sum_{S\in C_k}\log|\frac{X_k+S}{2}|-\frac{1}{2}\log|X_kS|. \tag{22}$$



(a) (b) (c) (d)

Fig. 1. Isosurface plots for various distance measures. First, distances for arbitrary three dimensional covariances from the identity matrix are computed, and later isosurfaces corresponding to fixed distances of 0.1, 0.5 and 1 are plotted. The plots show the surfaces for: (a) Frobenius distance, (b) AIRM, (c) KLDM, and (d) JBLD respectively.

Measure	$D^2(X,Y)$	$FLOPS$	Gradient (∇_X)
AIRM	$\|\log(X^{-1/2}YX^{-1/2})\|_F^2$	$4d^3$	$2X^{-1}\log(XY^{-1})$
LERM	$\|\log(X)-\log(Y)\|_F^2$	$\frac{8}{3}d^3$	$2X^{-1}\log(X)-2\sum_{j=1}^{\infty}\frac{1}{j}\sum_{k=0}^{j-1}(\mathcal{I}-X)^{(j-k-1)}\log(Y)(\mathcal{I}-X)^k$
KLDM	$\frac{1}{2}\operatorname{Tr}\left(X^{-1}Y+Y^{-1}X-2\mathcal{I}\right)$	$\frac{8}{3}d^3$	$Y^{-1}-X^{-1}YX^{-1}$
JBLD	$\log\left\|\frac{X+Y}{2}\right\|-\frac{1}{2}\log\|XY\|$	d^3	$(X+Y)^{-1}-\frac{1}{2}X^{-1}$

TABLE 3
A comparison of various metrics on covariances and their computational complexities against $J_{\ell d}$.

Unfortunately, $J_{\ell d}$ is not convex, which makes centroid computation tricky. The good news is that, we can write (22) as the sum of a convex function $F_{vex}(X_k,S) = -\sum_{S\in C_k}\frac{|C_k|}{2}\log|X_k|$ and a concave term $F_{cave}(X_k,S) = \sum_{S\in C_k}\log\left|\frac{X_k+S}{2}\right|$. Such a combination of convex and concave objectives can be efficiently solved using Majorization-Minimization using the Convex-ConCave Procedure (CCCP) [34]. The main idea of this procedure is to approximate the concave part of the objective by its first order Taylor approximation around the current best estimate X_k^t; that is, for the $(t+1)$st step:

$$X_k^{t+1} = \underset{X_k}{\operatorname{argmin}}\, F_{vex}(X_k,S) - X_k^T\nabla_{X_k}F_{cave}(X_k^t,S). \quad (23)$$

Substituting (23) in (22), later taking the gradient of (22) with respect to X_k and setting it to zero (recall that now we have a convex approximation to (22)), we have:

$$\sum_{S\in C_k}\nabla_{X_k}F_{vex}(X_k^{t+1},S) = -\sum_{S\in C_k}\nabla_{X_k}F_{cave}(X_k^t,S). \quad (24)$$

Expanding the gradient terms for $J_{\ell d}$, we have the following *fixed-point* iteration:

$$X_k^{t+1} = \left[\frac{1}{|C_k|}\sum_{S\in C_k}\left(\frac{S+X_k^t}{2}\right)^{-1}\right]^{-1}. \quad (25)$$

We now derive conditions guaranteeing the convergence of the fixed point iteration in (25). The uniqueness of a centroid thus found is reported in [35].

Lemma 11. *The function $f(X) = X^{-1}$ for $X \in \mathcal{S}_{++}^d$ is matrix convex, i.e., for $X, Y \in \mathcal{S}_{++}^d$ and for $t \in [0,1]$,*

$$f(tX+(1-t)Y) \le tf(X)+(1-t)f(Y). \quad (26)$$

Proof: See Exercise V.1.15 [36] for details. □

Lemma 12. *If $X, Y \in \mathcal{S}_{++}^d$ and suppose $X \ge Y$, then $X^{-1} \le Y^{-1}$.*

Proof: See Corollary 7.7.4 [37]. □

Theorem 13. *Let $S_1, S_2, \cdots, S_n$ be the input covariances and let X^* be the centroid found by* (25)*. Then X^* lies in the* compact *interval*

$$\left(\frac{1}{n}\sum_{i=1}^{n}S_i^{-1}\right)^{-1} \le X^* \le \frac{1}{n}\sum_{i=1}^{n}S_i. \quad (27)$$

Proof: The left inequality: Applying Lemma 11 to (25), we have:

$$X^{-1} \le \frac{1}{n}\sum_{i=1}^{n}\left(\frac{S_i^{-1}+X^{-1}}{2}\right) \quad (28)$$

$$\le \frac{1}{n}\sum_{i=1}^{n}\frac{S_i^{-1}}{2}+\frac{1}{2}X^{-1}. \quad (29)$$

Now, applying Lemma 12, the result follows.

The right inequality: As one can see, the right side of (25) is essentially the harmonic mean of $\frac{X+S_i}{2}$ for $i = 1, 2, \cdots, n$. Since the harmonic mean is always less than or equal to the arithmetic mean [38], we have the result. □

Theorem 14. *Let $\{X^t\}$ (for $t \ge 1$) be the sequence of successive iterates generated as per* (25)*. Then, $X^t \to X^*$, where X^* is a stationary point of* (22)*.*

Proof: It is clear that F_{vex} and $-F_{cave}$ are strictly convex functions and $-\nabla F_{cave}$ is continuous. Further, from Theorem 13 it is clear that the solution lies in a compact interval inside $\mathcal{S}_{++}^d$. Thus, following the conditions of convergence stipulated in [39] (CCCP-II, Theorem 8), the iterations in (25) converges for a suitable initialization inside the compact set. □

4.2 NN Using Metric Tree

Metric Trees (MT) [40] are fundamental for fast NN retrieval when the underlying objects lie in a metric space (recall that square-root of JBLD is a metric). NN using the MT involves two steps: (i) Building the tree, and (ii) Querying the tree. We discuss each of these steps below.

4.2.1 Building MT

To build the MT, we perform top-down partitioning of the input space by recursively applying the JBLD K-Means algorithm (introduced above). Each partition of the MT is identified by a centroid and the ball radius. For n data points, and assuming we bi-partition each cluster recursively, the total build time of the tree is $O(n \log n)$ (ignoring the cost for K-Means itself). To save time, we stop partitioning a cluster when the number of points in it goes below a certain threshold; this threshold is selected as a balance between the computational time to do exhaustive search on the cluster elements against doing K-Means on it.

4.2.2 Querying Using MT

Given a query point q, one first performs a greedy binary search for the NN along the most proximal centroids at each level. Once a leaf partition is reached, exhaustive search is used to localize to the candidate centroid X_c. Then one backtracks to check if any of the sibling nodes (that were temporarily ignored in the greedy search) contain a data point that is closer to q than X_c. To this end, we solve the following optimization problem on each of the sibling centroids C:

$$d(X_c, q) > \min_{X; d(X,C)=R} d(X, q), \tag{30}$$

where X is called the projection of q onto the ball with centroid C, radius R and d is some distance function. If (30) is satisfied, then the sibling node should be explored, otherwise it can be pruned. When d is a metric, (30) has a simple solution utilizing the triangle inequality as is described in [41]. The mechanism can be extended to retrieve k-NN by repeating the search ignoring the (k-1) NNs already retrieved. This can be efficiently implemented by maintaining a priority queue of potential subtree centroids and worst case distances of the query to any candidate node in this subtree, as described in [40].

5 Experiments

We are now ready to describe our experimental setup and results to substantiate the effectiveness of $J_{\ell d}$. We first discuss the performance metric on which our experiments are based, then provide simulation results exposing various aspects of our metric; these are followed by the results on four real-world datasets. All algorithms were implemented in MATLAB and tested on a machine with 3GHz single core CPU and 4GB RAM.

5.1 Performance Metric

Accuracy@K: Suppose we have a covariance dataset $\mathcal{D}$ and a query set $\mathcal{Q}$. Accuracy@K describes the average accuracy when retrieving K nearest covariances from $\mathcal{D}$ for each item in $\mathcal{Q}$. Suppose G_q^K stands for the ground truth label subset associated with the qth query, and if M_q^K denotes the label subset associated with the K nearest covariances found using a metric M for the query q, then we formally define:

$$Accuracy@K = \frac{1}{|Q|} \sum_{q \in Q} \frac{|G_q^K \cap M_q^K|}{|G_q^K|}. \tag{31}$$

Note that *Accuracy@K* as defined in (31) subsumes the standard performance metrics: *precision* and *recall*. Most often we work with $K = 1$, in which case we will drop the suffix and will refer as *Accuracy*. Since some of the datasets used in our experiments do not have ground truth data available, the baselines for comparison were decided via a linear scan using the AIRM metric as this metric is deemed the state-of-the-art on covariance data.

5.2 Simulations

Before we delve into the details of our experiments, we highlight here the base experimental configurations that we used for all the simulation experiments. Since there are a variety of code optimizations and offline computations possible for the various metrics, we decided to test all the algorithms with the base implementation as provided by MATLAB. An exception here are the experiments using LERM. It was found that for computing LERM, projecting the input matrices into the log-Euclidean space (through matrix logarithms) resulted in expensive computations, as a result of which the performances were incomparable with the setup used for other metrics. Thus, before using this metric, we took the logarithm of all the covariances offline.

For the NN experiments, we used a metric tree with four branches and allowed a maximum of 100 data points at the leaf nodes. With regard to computing the cluster centroids (for k-means), LERM and FROB metrics used the ordinary Euclidean sample mean, while AIRM used the Fréchet mean using the iterative approximation algorithm described in [42]. The centroid for KLDM boils down to computing the solution of a Riccati equation as described in [43]. For the simulation experiments, we used the results produced by AIRM as the ground truth.

Now we are ready to describe our base configuration for the various simulation experiments. We used 1K covariances of $\mathcal{S}_{++}^{10}$ with 50 true number of clusters as the dataset and a collection of 100 covariances as the query set. The plots that we are about to show resulted from average performances by repeating the experiments 100 times using different database and query sets.

5.2.1 Accuracy Against Noise

Given that the distances on covariances are nonlinear, the primary goal of this experiment is to validate the robustness of JBLD against noise in the covariance descriptors for the task of NN retrieval. This is especially useful when considering that our data can be poorly conditioned so that small perturbations of poorly conditioned data matrices can lead to large divergences, which for some applications might be uncalled for. Towards this end, we created a base set of 1K covariances from a set of simulated feature vectors. Subsequently, Gaussian noise of varying magnitude (relative to the signal strength) was added to the feature vectors to obtain a set of 100 noisy covariances. The base covariances were used as queries while the noisy ones as the database. A linear scan through the data using the Riemannian metric to measure nearness *defined* the ground truth. Figure 2 shows the average accuracies for decreasing SNR for three different covariance dimensions (10D, 20D and 40D). It is clear that JBLD is more robust than LERM and KLDM, at the same time yields accuracy almost close to the baseline Riemannian metric, irrespective of the dimension of the matrix. It is to be noted that a retrieval using the Frobenius distance (FROB) is clearly seen to perform poorly. We would also like to highlight that we noticed a small drop in the accuracy of KLDM (as seen in Figure 2(c)) as the data dimensionality increases, which we suspect is due to the poor conditioning of the data matrices as the dimensionality grows, impacting the matrix inversions.

5.2.2 Effect of Cluster Size

This section analyzes the scalability of $J_{\ell d}$ to an increasing number of true data clusters (given fixed database size). The basic goal of this experiment is to expose the clustering performance of our $J_{\ell d}$-K-Means algorithm against the K-Means based on other metrics. The performance comparison is analyzed on three aspects: (i) the average accuracy of NN retrieval, (ii) average metric tree creation time (which includes K-Means clustering for each internal node of the metric tree), and (iii) average search time using a metric tree. Figure 3 shows results from this experiment. There are a few important properties of the metrics that are revealed by these plots: (i) the accuracy of $J_{\ell d}$ matches closely with that of AIRM (note that AIRM is used as the ground truth), (ii) assuming the metric tree is constructed optimally, the search time for AIRM and $J_{\ell d}$ are comparable, and (iii) (which is the most important) the metric tree construction for AIRM almost increases quadratically with increasing number of true clusters, while that for other metrics is more favorable. Together, the three plots substantiate the superior performance of $J_{\ell d}$. Later in this paper, we will get back to illustrating these claims on real-data.

5.2.3 Effect of Matrix Dimension

One of the major motivations for proposing $J_{\ell d}$ as a replacement for existing metrics on covariances is its scalability to increasing matrix dimensions. Figure 4 shows the results of accuracy, metric tree creation time and search time using a metric tree. As is clear from the plots, the metric tree creation time increases at many orders of magnitude worse with AIRM than with other metrics, while $J_{\ell d}$ performs better at accuracy and retrieval time against other metrics. Similar to what we noticed in Figure 2, the accuracy of KLDM worsens as the matrix dimension increases.

5.2.4 Effect of Increasing Database Size

This experiment shows the performance of $J_{\ell d}$ against searching in larger datasets. Towards this end, we kept the number of true clusters constant and same as in other experiments, but increased the number of data points (covariances) associated with each cluster. The results of this experiment in terms of accuracy, tree buildup time and retrieval performance is shown in Figure 5. Similar to the previous plots, it is clear that $J_{\ell d}$ provides promising results in all the three properties, while maintaining nearly perfect retrieval accuracy, showing that it does not get distracted from the nearest neighbor even when the datasize increases.

5.2.5 Anisotropic Index

As we alluded to briefly in Subsection 3.2, Anisotropic Index (AI) of a matrix over a divergence (or metric) is a useful quantity in several DTMRI applications [16]. Figure 6 plots the average Fractional Anisotropic Index[4] ($\mathrm{FAI}(X) = \mathrm{AI}(X)/(1+\mathrm{AI}(X))$) for the various distances on 3×3 SPD tensors for increasing levels of tensor anisotropy (increasing condition number). As is clear from the plots, JBLD was found to have lower anisotropy when compared to other metrics on covariances.

5.3 Real Data Experiments

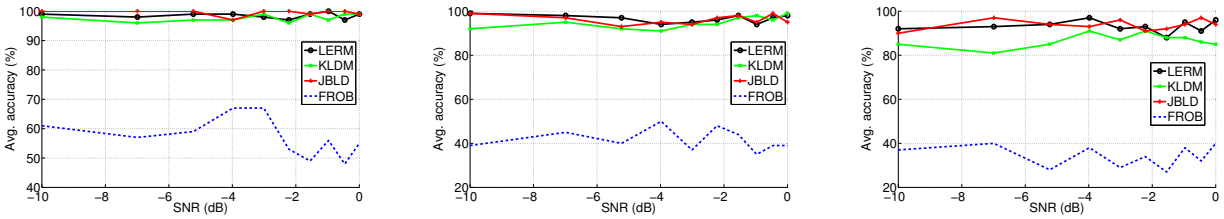
Continuing upon the simulated performance figures of $J_{\ell d}$ against other metrics, this subsection provides results on real-data. First, we will showcase a few qualitative results from some important applications of covariances from literature. We will demonstrate that JBLD outperforms other metrics in accuracy *not only* when AIRM is assumed to be the ground truth, but also in situations when we know the correct ground truth of data as provided by an external agency or human labeling.

5.3.1 Tracking using Integral Images

People appearance tracking has been one of the most successful applications using covariances. We chose to experiment with some of the popular tracking scenarios: (i) face tracking under affine transformations, (ii) face tracking under changes in pose, and (iii) vehicle tracking. For (i) and (ii), the tracking dataset described in [44] was used, while the vehicle tracking video was taken from the ViSOR repository[5]. The images from the video were resized to 244×320 for speed and integral images computed on each frame. An input tracking region was given at the beginning of the video, which is then tracked in subsequent images using the integral transform, later computing covariances from the features in this region.

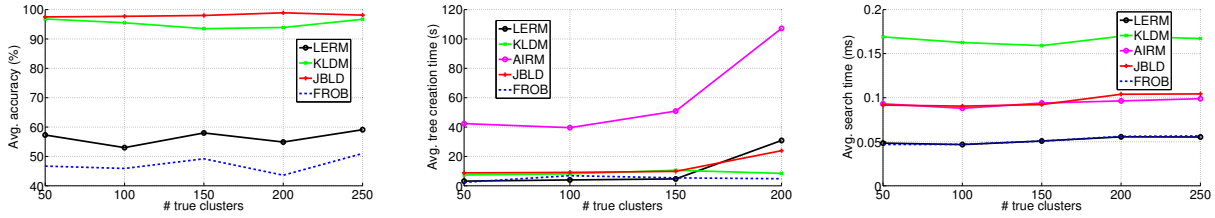
4. FAI is in the range of $[0, 1]$ irrespective of the underlying divergence and thus provides a fair comparison instead of just using AI.

5. http://www.openvisor.org



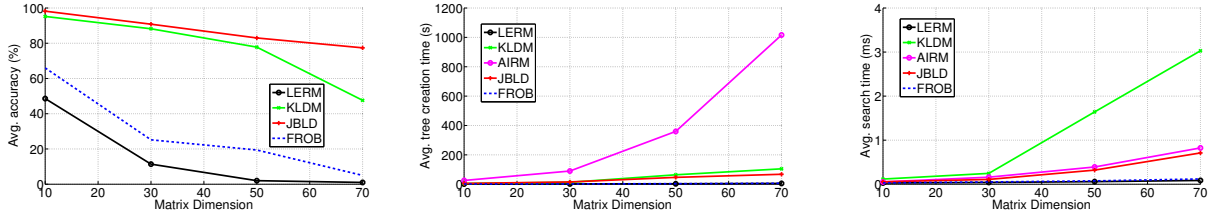
(a) n=10x10 (b) n=20x20 (c) n=40x40

Fig. 2. Accuracy against increasing noise for various matrix dimensions n; (a) $n = 10 \times 10$, (b) $n = 20 \times 20$, (c) $n = 40 \times 40$. It is assumed that the AIRM is the ground truth. FROB stands for the Matrix Frobenius Distance.



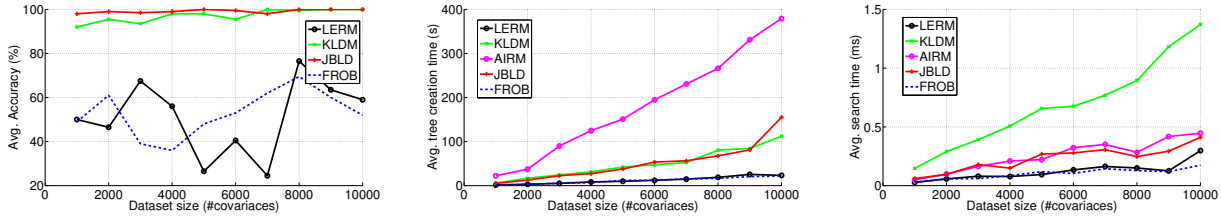
(a) (b) (c)

Fig. 3. Fixed dataset size of 1K, query size of 100 and for increasing number of true clusters: 3(a) accuracy of search, 3(b) time to create the metric tree, and 3(c) speed of retrieval using the metric tree. The average is computed over 100 trials.



(a) (b) (c)

Fig. 4. Fixed dataset size of 1K, query size of 100 and for increasing covariance matrix dimensions: 4(a) accuracy of search, 4(b) time to create the metric tree, and 4(c) speed of retrieval using the metric tree. The average is computed over 100 trials.



(a) (b) (c)

Fig. 5. Fixed number of true number clusters, query size of 100 and but increasing the covariance dataset size: 5(a) accuracy of search, 5(b) time to create the metric tree, and 5(c) speed of retrieval using the metric tree. The average is computed over 100 trials.

We used the color and the first order gradient features for the covariances. Figures 7(a),7(b), and 7(c) show qualitative results from these experiments. We compared the window of tracking for both AIRM and JBLD, and found that they always fall at the same location in the video (and hence not shown).

5.3.2 *Texture Segmentation*

Another important application of covariances has been in texture segmentation [4], which has further application in DTMRI, background subtraction [45], etc. In Figure 7(d), we present a few qualitative results from segmentation on the Brodatz texture dataset. Each of the images were a combination of

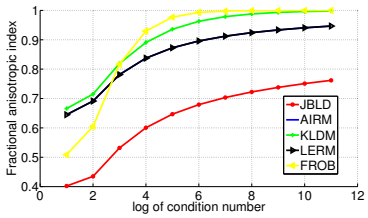


Fig. 6. Plots of fractional anisotropic indices of various distance functions on 3×3 SPD tensors for increasing condition numbers of the constituent tensors. The FAI of AIRM and LERM were found to be very close.

two different textures, the objective being to find the boundary and separate the classes. We first transformed the given texture image into a tensor image, where each pixel was replaced by a covariance matrix computed using all the pixels in a $p \times p$ patch around the given pixel. The 5×5 covariances were computed using features such as image coordinates of the pixels in this patch, image intensity at each pixel, and first order moments. Next, we applied the JBLD K-Means algorithm for the texture mixture, later segregating the patches using their cluster labels.

5.4 Real Data NN Experiments

Now we are ready to present quantitative results on real-world datasets. For real-world experiments that are described in the subsequent sections, we use four different vision applications for which covariance descriptors have shown to produce promising results: (i) texture recognition, (ii) action recognition, (iii) face recognition, and (iv) people appearance tracking. We briefly review below each of these datasets and how covariances were computed for each application.

Texture Dataset: Texture recognition has been one of the oldest applications of covariances spanning a variety of domains, e.g., DTMRI, satellite imaging, etc. The texture dataset for our experiments was created by combining the 160 texture images in the Brodatz dataset and the 60 texture classes in the CURET dataset [46]. Each texture category in the Brodatz dataset consisted of one 512×512 image. To create the covariances from these images, we followed the suggestions in [4]. First patches of size 20×20 were sampled from random locations in each image, later using the image coordinate of each pixel in a patch, together with the image intensity, and the first order gradients to build 5D features. Covariances computed from such feature vectors on all the pixels inside the patch constituted one such data matrix. We generated approximately 5K such covariances from all the texture images in all the categories from the Brodatz dataset. To build a larger dataset for textures, we combined this dataset with texture covariances from the CURET dataset [46] which consists of 60 texture categories, with each texture having varying degrees of illumination and pose variations. Using the RGB color information, together with the 5 features described before, we created approximately 27K covariances each of size 8×8. To have covariances of the same dimensionality across the two datasets, we appended a unit matrix of small diagonal for the RGB to the covariances computed from the Brodatz dataset.

Action Recognition Dataset: Activity recognition via optical flow covariances is a recent addition to the family of applications with covariance descriptors, and shows great promise. For every pair of frames in a given video, the optical flow is initially computed; the flow is then threshold and 12D feature vectors were extracted from each non-zero flow location (refer [11] for details on this feature vector). It is proposed that the covariance computed from the optical flow features captures the profile of that activity uniquely. To build the optical flow covariance dataset, we used a combination of activity videos from the Weizmann activity dataset [47], the KTH dataset[6] and the UT tower dataset [48]. This resulted in a large dataset of approximately 63.5K covariances each of dimension 12×12.

Face recognition: Face recognition is still an active area of research in computer vision and there has been many effective ideas suggested. In [9], the idea of covariance descriptors was extended for recognizing faces, where each face image was convolved with 40 Gabor filters, the outputs of which were then collated to form 40×40 covariances. Although the covariance descriptors are not the state-of-the-art in face recognition, our choice of this application for this paper is to analyze the performance of our metric for real-data of large dimensions. Towards this end, we used the images from the *Faces in the Wild* dataset [49], which consists of approximately 31K face images mainly collected from newspapers. We used the same approach as in [9] for computing the covariances, along with incorporating the RGB color information of each pixel and the first and second order intensity gradients to form 48×48 covariances.

People Appearances: An important real-time application of covariances is people tracking from surveillance cameras [4]. To analyze the suitability of our metric for such applications, we illustrate empirical results on tracking data. For this experiment, we used videos of people appearances tracked using multiple cameras[7]. The background was first learned using a mixture of Gaussians, then the silhouettes of people in the scene were extracted. The first and second order image gradients along with the color information were used to obtain approximately 10K covariances of size 8×8.

Ground Truth: Note that the texture dataset, the action dataset and the faces dataset have ground truth labels associated with each data point and thus for accuracy comparisons, we directly use this class label of the query set against the class label associated with the NN found by a metric. Unfortunately, the people appearances dataset does not have a ground truth and thus we use the label of the NN found by AIRM as the ground truth.

5.5 NN via Exhaustive Search

Here we present our experiments and results for NN via exhaustive search using the various metrics. Exhaustive search is important from a practical point of view as most real-time applications (such as tracking) cannot spend time in building a metric tree. In this section, we analyze the performance of JBLD in terms of accuracy and retrieval speed on each of the datasets we described in the previous section.

5.5.1 Accuracy

We divided each of the datasets into database and query sets, and then computed accuracy against either the available ground truth or the baseline computed using AIRM. The query set typically consisted of 1K covariances. The results are shown in Table 4. Clearly, JBLD outperforms all the other metrics in accuracy, without compromising much on the speed of retrieval. In the case of LERM, we had to vectorize the covariances using the log-Euclidean projections for tractability

6. http://www.nada.kth.se/cvap/actions/

7. http://cvlab.epfl.ch/research/body/surv/#data

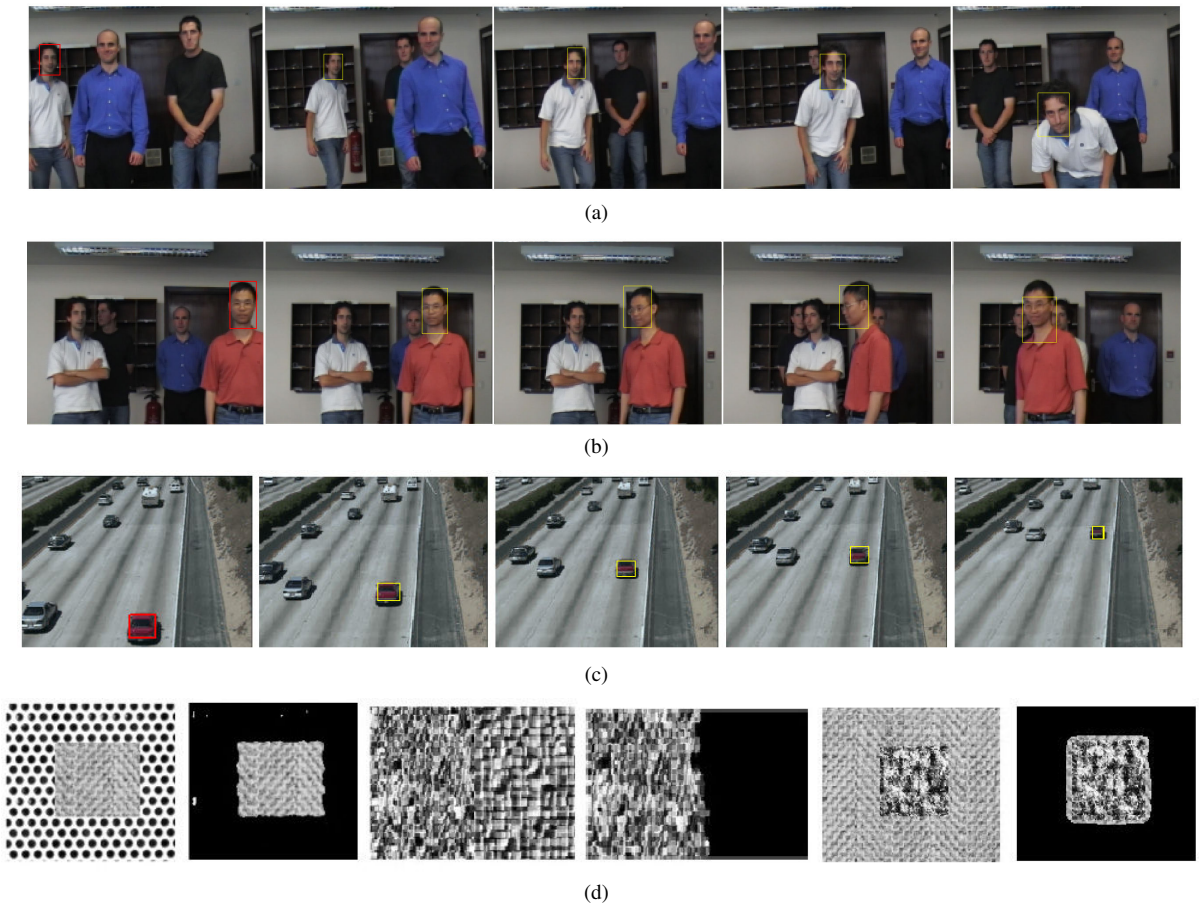


Fig. 7. Tracking using JBLD on covariances computed from integral images: (a) affine face tracking, (b) tracking face with pose variations, (c) vehicle tracking, (d) shows results from texture segmentation. The red rectangle in the first image in each row shows the object being tracked. The yellow rectangles in the subsequent images are the nearest objects returned by JBLD. (d) shows sample results from three texture segmentation experiments. The left image in each pair shows the original mixed texture image and the right image in each pair shows the output of segmentation, with one texture masked out.

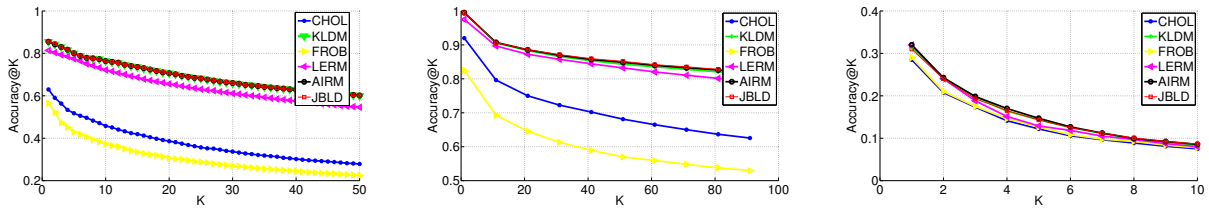
Dataset (size)	AIRM	JBLD	LERM	KLDM	CHOL	FROB
Texture (25852)						
Accuracy(%)	85.5	**85.5**	82.0	85.5	63.0	56.5
Time (s)	1.63	**1.50**	1.16 (4.21)	1.71	1.81	1.21
Activity (62425)						
Accuracy(%)	99.5	**99.5**	96.5	99.5	92.0	82.5
Time (s)	4.04	**3.71**	2.42 (10.24)	4.34	4.98	2.53
Faces (29700)						
Accuracy(%)	32.5	**33.0**	30.5	31.5	29.5	26.5
Time (s)	10.26	**4.68**	2.44 (24.54)	10.33	12.13	2.13
Tracking (8596)						
Accuracy(%)	–	**100**	83.3	70.0	91.0	52.1
Time (s)	0.44	**0.40**	0.17 (1.7)	0.42	0.28	0.15

TABLE 4
Performance of JBLD on different datasets and against various other metrics for 1-NN query using exhaustive search averaged over 1K queries. Note that for the appearance tracking dataset, we used AIRM as the baseline (and thus the accuracy not shown). Avg. time is in seconds for going over the entire dataset once to find the NN. The time taken for the offline log-Euclidean projections is shown in brackets under LERM.

of the application. The time taken for this operation for each of the datasets is also shown in the table. Since this embedding uses the eigen decomposition of the matrices, this operation is seen to be computationally expensive, deterring the suitability of LERM for real-time applications. We also compare the performance of JBLD against other distances such as the Cholesky (CHOL) distance and the Frobenius (FROB) distance. Frobenius distance was seen to perform poorly in all our experiments, although as expected, it is the fastest. The numerical results are averaged over 10 trials, each time using a different database and a query set.

5.5.2 *Accuracy@K*

We take the previous experiments of 1-NN a step further and present results on K-NN retrieval for an increasing K. The idea is to generalize the power of 1-NN to a K-NN application. We plot in Figure 8, the results of Accuracy@K, where the maximum value of K is determined by the cardinality of a ground truth class. The plots clearly show that JBLD performs well against almost all other metrics in terms of accuracy for increasing K.



(a) (b) (c)

Fig. 8. Accuracy@K plots for (a) texture dataset, (b) activity dataset, (c) faces dataset.

5.6 NN Performance Using Metric Tree

Building the Tree: The time required to build the NN data structure plays a critical role in the deployment of a measure. In Table 5, we show a comparison of the build time of the metric tree for each of the datasets, with comparisons of JBLD against AIRM. As is clear from the table, the performance of AIRM is poor and worsens with the increase in the matrix dimensions (see the face dataset). JBLD, on the other hand, takes far lesser time to initialize and shows consistent performance even against increasing dataset size and matrix dimensions.

Dataset (size)	AIRM	JBLD
Texture (25852)	769.96	**131.31**
Activity (62425)	2985.62	**746.67**
Faces (29700)	13776.30	**854.33**
People (8596)	213.41	**53.165**

TABLE 5
Comparison of metric tree buildup times (in seconds) for the various datasets.

5.7 NN Retrieval

5.7.1 Exact NN via Metric Tree

Next, we compare the accuracy and the speed of retrieval of JBLD against the other metrics using the metric tree. For this experiment, we used a metric tree with four branches at each internal node and 1K leaf nodes, for all the datasets. Since K-Means using AIRM was found to take too much time until it converged (it was found that with the face dataset with 48x48 covariances took more than 3 hours with approximately 26K covariances), we decided to stop the clustering process when there was less than 10% of data movements in the underlying Loyd's algorithm. This configuration was forced on K-Means using other metrics as well for fairness of comparison of the results. We show in Table 6 the average results of 1-NN using the metric tree with 500 queries, and with averages computed over 10 trials, each time using a different sample set for the database and the query. As is clear from the table, JBLD provides accuracy equal to AIRM with at least 1.5 times speedup with the matrices of small size, while more than 7 times speedup for the face dataset. The retrieval speed of LERM and FROB is high, while the accuracy is low. KLDM was seen to provide accuracy similar to JBLD, but with low retrieval speed. In short, JBLD seems to provide the best mix of accuracy and computational expense.

5.7.2 Approximate NN via Metric Tree

It is well-known that the worst case computational complexity of metric tree is linear. Thus in Table 7, we also evaluate the performance of an approximate variant of metric tree based retrieval in which we limit the search for NNs while backtracking the metric tree to at most n items, where in our experiments we used $n = 5$. This heuristic is in fact a variant of the well-known Best-Bin-First (BBF) [50] method, the idea is to sacrifice the accuracy a little bit for a large speedup in retrieval. As is clear from the table, such a simple heuristic can provide a speedup of approximately 100 times than that of the exact NN, while not much of a lose in the accuracy. Also, it is clear from the table that JBLD gives the best accuracy among other metrics with reasonable retrieval results.

5.8 Summary of Results

Here we summarize our findings about JBLD and the other metrics with regard to our experiments. As is clear from the above tables and plots, JBLD was seen to provide the best accuracy compared to other metrics, with accuracies sometimes even superseding that of the Riemannian metric. We found that both JBLD and the square-root of JBLD (which is a metric) provided similar performance results, except for the metric tree experiments for which we can use only the latter. It might seem from Table 7 that the speed of retrieval of JBLD is close to that of AIRM; this result needs to be seen together with the results in Table 5 which shows that building a metric tree for AIRM is extremely challenging, especially when the data is large dimensional. KLDM sometimes matches the accuracy of JBLD, and exhibits higher errors at other times. However, it always runs slower than JBLD, requiring up to more than twice as much computational time. LERM seemed superior in retrieval speed due to the capability of offline computations, while was seen to have lower accuracy. Finally, FROB was found to perform the best in speed as would be expected, but has the lowest accuracy. In summary, JBLD is seen to provide the most consistent results among all the experiments, with the best accuracy, scalability and moderate retrieval speeds.

6 Conclusion

We introduced a similarity measure based on the Jensen-Bregman LogDet Divergence (JBLD) for comparing covariance valued data. This measure has several desirable theoretical properties including inequalities relating it to other metrics

Dataset	AIRM	JBLD	LERM	KLDM	FROB
Texture					
Acc. (%)	83.00	**83.00**	78.40	83.00	52.00
Time (ms)	953.4	**522.3**	396.3	1199.6	522.0
Activity					
Acc. (%)	98.8	**99.00**	95.80	98.60	85.60
Time (ms)	3634.0	**3273.8**	1631.9	4266.6	1614.92
Faces					
Acc. (%)	26.6	**26.6**	22.8	26.1	20.6
Time (ms)	9756.1	**1585.1**	680.8	2617.7	658.6
People					
Acc. (%)	–	**100**	92.0	98.1	43.3
Time (ms)	354.3	**229.7**	214.2	701.1	163.7

TABLE 6
True NN using the metric tree. The results are averaged over 500 queries. Also refer to Table 5 for comparing the metric tree creation time.

Dataset	AIRM	JBLD	LERM	KLDM	FROB
Texture					
Acc. (%)	80.2	**81.40**	76.80	81.40	48.80
Time (ms)	34.28	**21.04**	18.18	52.98	17.73
Activity					
Acc. (%)	95.6	**96.20**	93.60	95.6	78.00
Time (ms)	38.1	**30.39**	20.3	85.9	12.2
Faces					
Acc. (%)	22.4	**24.2**	20.2	22.2	18.6
Time (ms)	26.16	**23.2**	20.6	55.7	16.6
People					
Acc.(%)	–	**91.3**	85.6	91.1	36.4
Time (ms)	4.81	**4.78**	3.31	8.12	3.07

TABLE 7
ANN performance using Best-Bin-First strategy using metric tree. The results are averaged over 500 queries. Also refer to Table 5 for comparing the metric tree creation time.

for covariances, including the Riemannian metric (which is probably the first choice among such metrics). We showed that JBLD outperforms the Riemannian metric in speed, without any drop in accuracy. Further, we showed results for computing the centroid of covariances under our metric, followed by an application of JBLD to nearest neighbor retrieval using a metric tree. Experiments validated the effectiveness of the measure. Going forward, we would like to apply JBLD to classification and regression problems (initial reports can be seen in [51]).

ACKNOWLEDGEMENTS

This material is based upon work supported in part by the U.S. Army Research Laboratory and the U.S. Army Research Office under contract #911NF-08-1-0463 (Proposal 55111-CI), and the National Science Foundation through grants #IIP-0443945, #CNS-0821474, #IIP- 0934327, #CNS-1039741, #IIS-1017344, #IIP-1032018, and #SMA-1028076. Arindam Banerjee is supported by NSF grants #IIS-0916750, #IIS-0812183, #IIS-1029711, #NetSE-1017647, and NSF CAREER award #IIS-0953274.

REFERENCES

[1] D. Alexander, C. Pierpaoli, P. Basser, and J. Gee, "Spatial transformations of diffusion tensor magnetic resonance images," *IEEE Trans. on Med. Imaging*, vol. 20, no. 11, pp. 1131–1139, 2002.

[2] H. Zhu, H. Zhang, J. Ibrahim, and B. Peterson, "Statistical analysis of diffusion tensors in diffusion-weighted magnetic resonance imaging data," *Journal of the American Statistical Association*, vol. 102, no. 480, pp. 1085–1102, 2007.

[3] M. Chiang, R. Dutton, K. Hayashi, O. Lopez, H. Aizenstein, A. Toga, J. Becker, and P. Thompson, "3D pattern of brain atrophy in HIV/AIDS visualized using tensor-based morphometry," *Neuroimage*, vol. 34, no. 1, pp. 44–60, 2007.

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

[5] F. Porikli, and O. Tuzel, "Covariance tracker," *CVPR*, 2006.

[6] O. Tuzel, F. Porikli, and P. Meer, "Human detection via classification on Riemannian manifolds," in *CVPR*, 2007, pp. 1–8.

[7] J. Malcolm, Y. Rathi, and A. Tannenbaum, "A graph cut approach to image segmentation in tensor space," in *CVPR*, 2007, pp. 1–8.

[8] T. Brox, M. Rousson, R. Deriche, and J. Weickert, "Unsupervised segmentation incorporating colour, texture, and motion," in *Computer Analysis of Images and Patterns*. Springer, 2003, pp. 353–360.

[9] Y. Pang, Y. Yuan, and X. Li, "Gabor-based region covariance matrices for face recognition," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 18, no. 7, pp. 989–993, 2008.

[10] W. Zheng, H. Tang, Z. Lin, and T. Huang, "Emotion recognition from arbitrary view facial images," *ECCV*, pp. 490–503, 2010.

[11] K. Guo, P. Ishwar, and J. Konrad, "Action recognition using sparse representation on covariance manifolds of optical flow," in *AVSS*. IEEE, 2010, pp. 188–195.

[12] C. Ye, J. Liu, C. Chen, M. Song, and J. Bu, "Speech emotion classification on a Riemannian manifold," *Adv. Multimedia Inf. Proc.*, pp. 61–69, 2008.

[13] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," *Annual Symposium on Computational Geometry*, pp. 253–262, 2004.

[14] X. Pennec, P. Fillard, and N. Ayache, "A Riemannian framework for tensor computing," *IJCV*, vol. 66, no. 1, pp. 41–66, 2006.

[15] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, "Log-Euclidean metrics for fast and simple calculus on diffusion tensors," *Magnetic Resonance in Medicine*, vol. 56, no. 2, pp. 411–421, 2006.

[16] M. Moakher and P. Batchelor, "Symmetric positive-definite matrices: from geometry to applications and visualization," *Visualization and Processing of Tensor Fields*, 2006.

[17] R. Bhatia, *Positive definite matrices*. Princeton Univ Press, 2007.

[18] X. Li, W. Hu, Z. Zhang, X. Zhang, M. Zhu, and J. Cheng, "Visual tracking via incremental Log-Euclidean Riemannian subspace learning," in *CVPR*, 2008.

[19] Q. Gu and J. Zhou, "A similarity measure under Log-Euclidean metric for stereo matching," in *CVPR*, 2009, pp. 1–4.

[20] Z. Wang, B. Vemuri, Y. Chen, and T. Mareci, "A constrained variational principle for direct estimation and smoothing of the diffusion tensor field from complex DWI," *IEEE Trans. on Med. Imaging*, vol. 23, no. 8, pp. 930–939, 2004.

[21] I. Dryden, A. Koloydenko, and D. Zhou, "Non-Euclidean statistics for covariance matrices, with applications to diffusion tensor imaging," *Annals of Applied Statistics*, vol. 3, no. 3, pp. 1102–1123, 2009.

[22] F. Nielsen, P. Piro, and M. Barlaud, "Bregman vantage point trees for efficient nearest neighbor queries," in *ICME*, 2009, pp. 878–881.

[23] L. Cayton, "Fast nearest neighbor retrieval for Bregman divergences," in *ICML*, 2008, pp. 112–119.

[24] P. Turaga and R. Chellappa, "Nearest-neighbor search algorithms on non-Euclidean manifolds for computer vision applications," in *CVGIP*, 2010, pp. 282–289.

[25] R. Chaudhry and Y. Ivanov, "Fast approximate nearest neighbor methods for non-Euclidean manifolds with applications to human activity analysis in videos," *ECCV*, pp. 735–748, 2010.

[26] Z. Chebbi and M. Moakher, "Means of hermitian positive-definite matrices based on the log-determinant alpha-divergence function," *LAA*, vol. 436, pp. 1872–1889, 2012.

[27] B. Kulis, M. Sustik, and I. Dhillon, "Low-rank kernel learning with Bregman matrix divergences," *JMLR*, vol. 10, pp. 341–376, 2009.

[28] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh, "Clustering with Bregman divergences," *JMLR*, vol. 6, pp. 1705–1749, 2005.

[29] Y. Censor and S. A. Zenios, *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1997.
[30] F. Nielsen and R. Nock, "On the centroids of symmetrized Bregman divergences," *Arxiv preprint arXiv:0711.3242*, 2007.
[31] S. Sra, "Positive definite matrices and the symmetric Stein divergence," *http://arxiv.org/abs/1110.1773*, 2011.
[32] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Johns Hopkins University Press, 1996.
[33] P. Fillard, V. Arsigny, N. Ayache, and X. Pennec, "A Riemannian framework for the processing of tensor-valued images," *Deep Structure, Singularities, and Computer Vision*, pp. 112–123, 2005.
[34] A. Yuille and A. Rangarajan, "The concave-convex procedure," *Neural Computation*, vol. 15, no. 4, pp. 915–936, 2003.
[35] F. Nielsen and S. Boltz, "The Burbea-Rao and Bhattacharyya Centroids," *IEEE Trans. on Info. Theory*, vol. 57, no. 8, pp. 5455–5466, 2011.
[36] R. Bhatia, *Matrix analysis*. Springer Verlag, 1997, vol. 169.
[37] R. Horn and C. Johnson, *Matrix analysis*. Cambridge University Press, 1990.
[38] J. D. Lawson and Y. Lim, "The geometric mean, matrices, metrics, and more," *The American Mathematical Monthly*, vol. 108, no. 9, pp. 797–812, 2001.
[39] B. Sriperumbudur and G. Lanckriet, "On the convergence of the concave-convex procedure," *NIPS*, vol. 22, pp. 1759–1767, 2009.
[40] P. Ciaccia, M. Patella, and P. Zezula, "M-tree: An efficient access method for similarity search in metric spaces," in *VLDB*. Morgan Kaufmann Publishers Inc., 1997, pp. 426–435.
[41] S. Brin, "Near neighbor search in large metric spaces," in *VLDB*, 1995.
[42] D. Bini and B. Iannazzo, "Computing the Karcher mean of symmetric positive definite matrices," *LAA*, 2011.
[43] T. Myrvoll and F. Soong, "On divergence based clustering of Normal distributions and its application to HMM adaptation," in *Euro. Conf. on Speech Comm. and Tech.*, 2003, pp. 1517–1520.
[44] E. Maggio, E. Piccardo, C. Regazzoni, and A. Cavallaro, "Particle PHD filtering for multi-target visual tracking," in *ICASSP*, vol. 1, 2007.
[45] R. Caseiro, J. Henriques, and J. Batista, "Foreground segmentation via background modeling on Riemannian manifolds," in *ICPR*, 2010, pp. 3570–3574.
[46] K. Dana, B. Van-Ginneken, S. Nayar, and J. Koenderink, "Reflectance and texture of real world surfaces," *TOG*, vol. 18, no. 1, pp. 1–34, 1999.
[47] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," *PAMI*, vol. 29, no. 12, pp. 2247–2253, 2007.
[48] C. Chen, M. Ryoo, and J. Aggarwal, "UT-Tower dataset: Aerial View Activity Classification Challenge," http://cvrc.ece.utexas.edu/SDHA2010/Aerial_View_Activity.html, 2010.
[49] V. Jain and E. Learned-Miller, "FDDB: a benchmark for face detection in unconstrained settings," University of Massachusetts, Amherst, Tech. Rep. UM-CS-2010-009, 2010.
[50] J. Beis and D. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," in *CVPR*, 1997, pp. 1000–1006.
[51] F. Nielsen, M. Liu, and B. C. Vemuri, "Jensen divergence-based means of SPD matrices," *Matrix Information Geometry*, pp. 111–122, 2012.

Anoop Cherian received his B.Tech (honours) degree in computer science and engineering from the National Institute of Technology, Calicut, India in 2002, and M.S. in computer science from the University of Minnesota, Minneapolis in 2010. He is currently working towards his Ph.D. degree in computer science at the University of Minnesota. From 2002–2007, he worked as a software design engineer at Microsoft. His research interests include machine learning, and computer vision. He is the recipient of the Best Student Paper award at the Intl. Conf. on Image Processing (ICIP) in 2012.

Suvrit Sra is a Senior Research Scientist at the Max Planck Institute for Intelligent Systems in Tbingen, Germany. He received a Ph.D. in Computer Science from the University of Texas at Austin in 2007. His research focuses on "large-scale data analysis and optimization". In particular, he designs, analyzes, and implements algorithms for large-scale (data intensive) problems in areas such as scientific computing, statistics, data mining, computer vision, and machine learning. Beyond optimization, he has interests in numerous subareas within mathematics; most notably in matrix algebra and analysis. His research has won awards at several international venues; the most recent being the "SIAM Outstanding Paper Prize (2011)" for his work on metric nearness. He regularly organizes the Neural Information Processing Systems (NIPS) workshops on "Optimization for Machine Learning" and has recently edited a book of the same title.

Arindam Banerjee is an associate professor at the Department of Computer and Engineering and a Resident Fellow at the Institute on the Environment at the University of Minnesota, Twin Cities. His research interests are in machine learning, data mining, convex analysis and optimization, and their applications in complex real-world problems including problems in Text and Web Mining, Climate Sciences, Finance, Social Network Analysis, and Bioinformatics. He has won several awards, including the NSF CAREER award in 2010, the McKnight Land-Grant Professorship at the University of Minnesota, Twin Cities (20092011), the J. T. Oden Faculty Research Fellowship from the Institute for Computational Engineering and Sciences (ICES), University of Texas at Austin (2006), the IBM PhD fellowship for the academic years 2003-2004 and 2004-2005, and four Best Paper awards.

Nikolaos Papanikolopoulos received the Diploma degree in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 1987, the M.S.E.E. in electrical engineering from Carnegie Mellon University (CMU), Pittsburgh, PA, in 1988, and the Ph.D. in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 1992. Currently, Dr. Papanikolopoulos is a Distinguished McKnight University Professor in the Department of Computer Science at the University of Minnesota and Director of the Center for Distributed Robotics and SECTTRA. His research interests include robotics, computer vision, sensors for transportation applications, and control. He has authored or coauthored more than 280 journal and conference papers in the above areas (sixty seven refereed journal papers). Dr. Papanikolopoulos was finalist for the Anton Philips Award for Best Student Paper in the 1991 IEEE Int. Conf. on Robotics and Automation and recipient of the best Video Award in the 2000 IEEE Int. Conf. on Robotics and Automation. Furthermore, he was recipient of the Kritski fellowship in 1986 and 1987. He was a McKnight Land-Grant Professor at the University of Minnesota for the period 1995-1997 and has received the NSF Research Initiation and Early Career Development Awards. He was also awarded the Faculty Creativity Award from the University of Minnesota. One of his papers (co-authored by O. Masoud) was awarded the IEEE VTS 2001 Best Land Transportation Paper Award. Finally, he has received grants from DARPA, DHS, U.S. Army, U.S. Air Force, Sandia National Laboratories, NSF, Lockheed Martin, Microsoft, INEEL, USDOT, MN/DOT, Honeywell, and 3M (more than $20M).