Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 Keller Hall
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 12-013

Jensen-Bregman LogDet Divergence for Efficient Similarity Computations on Positive Definite Tensors

Anoop Cherian, Suvrit Sra, Arindam Banerjee, and Nikos Papanikolopoulos

May 02, 2012

Jensen-Bregman LogDet Divergence for Efficient Similarity Computations on Positive Definite Tensors

†Anoop Cherian ‡Suvrit Sra †Arindam Banerjee †Nikos Papanikolopoulos
†Dept. of Computer Science ‡MPI for Intelligent Systems
Minneapolis, MN-55455 Tuebingen, Germany
†{cherian, banerjee, npapas}@cs.umn.edu
‡suvrit.sra@tuebingen.mpg.de

May 1, 2012

Abstract

Covariance matrices provide an easy platform for fusing multiple features compactly and as a result have found immense success in several computer vision applications, including activity recognition, visual surveillance, and diffusion tensor imaging. An important task in all of these applications is to compute the distance between covariance matrices using a (dis)similarity function, for which the natural choice is the Riemannian metric corresponding to 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). To address the problem of efficient nearest neighbor retrieval on large covariance datasets, we propose a metric tree framework using kmeans clustering on JBLD. We demonstrate the superior performance of JBLD on covariance datasets from several computer vision applications.

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, they are by now preferred data representations in several applications.

A covariance descriptor is nothing but the covariance matrix of features from an image region. Mathematically,

Definition 1. *Let $F_i \in \mathbb{R}^p$, for $i = 1, 2, \cdots, N$, be the feature vectors from the region of interest of an image, then the Covariance Descriptor of this region $C \in \mathcal{S}^p_{++}$ is defined as:*

$$C = \frac{1}{N-1} \sum_{i=1}^{N} (F_i - \mu_F)(F_i - \mu_F)^T \tag{1}$$

where $\mu_F = \frac{1}{N} \sum_{i=1}^{N} F_i$, is the mean feature vector and $\mathcal{S}^p_{++}$ is the space of $p \times p$ Symmetric Positive Definite (SPD) matrices.

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 aplenty in multi-camera object tracking applications [4, 5]. Other important applications of covariances include but not limited to human detection [6], image segmentation [7], texture segmentation [8], robotics and autonomous vehicle navigation [9], robust face recognition [10], emotion recognition [11], structure tensor for background subtraction applications [12], and human action recognition [13]. Application of covariances as data descriptors is not limited to computer vision; examples are speech recognition [14], and acoustic compression [15].

However, these successful applications are 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 of similarity measure is crucial, especially for a fundamental task such as the Nearest Neighbor (NN) retrieval which forms the building block for many 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, NN retrieval and the subsequent tracking are severely disadvantaged. Standard NN retrieval techniques such as locality sensitive hashing [16] 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 for covariance matrices, for which we introduce 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.

This paper is organized as follows. We start with a review of several similarity metrics on covariance matrices in Section 2. This is followed by an introduction to the JBLD measure, and exposition of its properties in Section 3. Section 4 discusses the application of JBLD for nearest neighbor retrieval on covariances. Towards this end, we propose a kmeans clustering algorithm using JBLD in Section 4.1. Experiments and results are presented in Section 5 followed by conclusion in Section 6.

Before we proceed with the paper, we briefly describe our notation. We refer to the $d \times d$ space of Symmetric Positive Definite (SPD) matrices as $\mathcal{S}_{++}^d$. At places where the dimensionality of the matrix is unimportant, an SPD matrix X might be introduced as $X > 0$. The notation $\mathcal{S}^d$ represents the space of $d \times d$ symmetric matrices. We use $|\ |$ to denote matrix determinant, Tr denotes the trace and $\|\ \|_F$ for the matrix Frobenius norm. Also, $\mathcal{I}$ refers to a $d \times d$ identity matrix.

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 [17, 18]. 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. DT-MRI [17].

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)* [19, 20] which is defined as follows: For X, Y in $\mathcal{S}_{++}^d$,

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

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 (2), 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 [19]. 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 [10] (40×40), and emotion recognition [11] (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. [17] observed that under this

mapping, the Lie group of SPD matrices is *isomorphic* and *diffeomorphic* (smooth manifolds are mapped to smooth manifolds) to the space of symmetric matrices. That is, the $\log$ is a bijection. Using this mapping, the paper introduces LERM as:

$$D_{le}(X,Y) := \|\log(X) - \log(Y)\|_{\mathrm{F}}. \tag{3}$$

On the positive side, LERM maps SPD matrices to a flat Riemannian space (of null curvature) so that the ordinary Euclidean distances can be used. The metric is easy to compute, and preserves a few 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 [21], stereo matching [22], 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 (gradients, 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: $D_{le}(X,Y) \leq D_R$. Further, the equality holds only when X and Y commute.*

Proof. Since $\boldsymbol{X}, \boldsymbol{Y}$ are positive matrices, we can write them in the exponential form as $\boldsymbol{X} = e^X$ and $\boldsymbol{Y} = e^Y$ respectively, where X and Y are symmetric matrices. Now, recalling that the Riemannian metric D_R is affine invariant, we can rewrite (2) in the following equivalent form:

$$D_R^2 = \operatorname{Tr}\left(\left(\log^2(e^Y e^{-X})\right)\right) \tag{4}$$

Invoking the Golden-Thompson inequality [23] and the monotonicity of the $\log$ function, we have the following inequality from (4),

$$D_R^2 = \operatorname{Tr}\left(\log^2\left(e^Y e^{-X}\right)\right) \geq \operatorname{Tr}\left(\log^2\left(e^{Y-X}\right)\right) = D_{le}^2.$$

□

Similar to our approach, there have been previous attempts to use symmetrized *f-divergences* from information theory into developing distances on SPD matrices. One such idea is to view the SPD matrices as being the covariances associated with zero-mean Gaussian distributions [18], and then use the symmetrized KL-Divergence Metric (KLDM) as the distance between the distributions. This leads to the following definition of KLDM:

$$D_{kl}^2(X,Y) := \frac{1}{2}\operatorname{Tr}\left(X^{-1}Y + Y^{-1}X - 2\mathcal{I}\right) \tag{5}$$

This measure does not require matrix eigenvalue computations, or logarithms, and at the same time enjoys many of the 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 being 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 $D_{kl} > D_R$.*

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:

$$\begin{aligned} D_{kl}^2 &= \sum_{i=1}^{d} \left(\frac{e^{u_i} + e^{-u_i}}{2} \right) - 1 \\ &= \frac{D_R^2}{2} \sum_{i=1}^{d} \left(1 + 2\frac{u_i^2}{4!} + \cdots \right) - 1. \end{aligned}$$

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 [24]. 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:

$$D_C(X, Y) = \|L_1 - L_2\|_{\mathrm{F}}. \tag{6}$$

Other similarity measures on covariance matrices may be found in [25]. Albeit their easy formulations and properties close to those of AIRM, the above distances based on *f-divergences* have not been very popular in SPD matrix based applications due to their poor accuracy (as our experiments will later 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, as we will later see, 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. In [26], an attempt is made to adapt NN techniques from vector spaces to non-Euclidean spaces, while [27] 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)*. We remark that although this measure seems natural and simple, to our knowledge it has *not been* formally discussed in detail before. We alert the reader that JBLD should not be confused with its asymmetric cousin: the so-called LogDet divergence [28].

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

$$d_\phi(x, y) := \phi(x) - \phi(y) - \langle x - y, \nabla\phi(y)\rangle \,, \tag{7}$$

where $\phi : S \subseteq \mathbb{R}^d \to \mathbb{R}$ is a strictly convex function of Legendre-type on $\text{int}(\text{dom}\, S)$ [29]. From (7) 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 [29, 30], 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 [31–33], where assuming $s = (x + y)/2$,

$$J_\phi(x, y) := \frac{1}{2}\big(d_\phi(x, s) + d_\phi(s, y)\big), \tag{8}$$

or even variants that satisfy the triangle inequality [33, 34].

Both (7) and (8) 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 (7) 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

$$J_{\ell d}(X, Y) := \log\left|\frac{X + Y}{2}\right| - \frac{1}{2}\log|XY|. \tag{9}$$

where $|\cdot|$ denotes the determinant; this divergence is obtained from the matrix version of (8) 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 [35] 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 [35])
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)

We would like to remark that $J_{\ell d}$ can also be written as follows:

$$J_{\ell d}(X, Y) = \text{Tr}\left(\log\left(\frac{X + Y}{2}\right) - \frac{1}{2}\left(\log XY\right)\right) \tag{10}$$

where $\log$ is the matrix logarithm. Although this construction of $J_{\ell d}$ makes it slightly computationally expensive, such a formulation could be suitable for some applications.

Theorem 3 (Non-Convexity)**.** *Assuming $X, Y > 0$, 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

$$\nabla_X^2 J_{\ell d}(X, Y) = -(X+Y)^{-1} \otimes (X+Y)^{-1} + \frac{X^{-1} \otimes X^{-1}}{2}. \tag{11}$$

This expression is positive for $X \leq (1+\sqrt{2})Y$ and negative 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 are important mathematical objects in this field useful in the analysis of such diffusion patterns [1]. *Anisotropic index* is a useful quantity that is often used in this area [18], 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:

$$\min_{\alpha > 0} \mathcal{D}(\alpha\mathcal{I}, P) \tag{12}$$

There are closed form expressions for α when $\mathcal{D}$ is AIRM, LERM, or KLDM (see [18] for details). Unfortunately, for $J_{\ell d}$ there is no closed form for this. In the following, we investigate this front of our metric and propose a few theoretical properties.

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 polynomial equation:*

$$\begin{aligned} p(\alpha) :=& d\alpha^d + (d-2)\sum_i \lambda_i \alpha^{d-1} + (d-4)\sum_{i,j,i\neq j} \lambda_i\lambda_j \alpha^{d-2} \\ &+ \cdots + (2-d)\sum_i \prod_{i\neq j} \lambda_j \alpha - d\prod_i \lambda_i = 0, \end{aligned} \tag{13}$$

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

Proof. Using the definition of $J_{\ell d}$ in (12), 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 13. 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 the Riemannian distance.*

Since in DT-MRI, generally 3×3 SPD matrices are used, we show this case next.

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

$$\frac{1 + \operatorname{Tr}(P)/d}{1 + \operatorname{Tr}(P^{-1})/d} > |P|. \tag{14}$$

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:

$$\frac{d + \operatorname{Tr}(P)}{d\,|P| + \sum_i \prod_{j \neq i} \lambda_i \lambda_j} > 1 \tag{15}$$

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 (13), we have the following third degree polynomial equation:

$$p(\alpha) := 3\alpha^3 + \operatorname{Tr}(P)\alpha^2 - |P| \operatorname{Tr}(P^{-1})\alpha - 3\,|P| = 0 \tag{16}$$

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 of the interesting connections $J_{\ell d}$ has with the standard metrics on covariances.

Theorem 8 (Relations)**.**

$$\begin{array}{ll} (i) & J_{\ell d} \leq D_R^2 \\ (ii) & J_{\ell d} \leq D_{kl}^2 \end{array}$$

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

$$J_{\ell d}(X, Y) = \sum\nolimits_{i=1}^{d} (\log(1 + e^{u_i}) - u_i/2 - \log 2), \tag{17}$$

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

$$\sum\nolimits_{i=1}^{d} f(u_i) = D_R^2(X,Y) - J_{\ell d}(X,Y) \geq 0. \tag{18}$$

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

$$g(u) = \frac{1}{2}(e^u + \frac{1}{e^u}) - \log(1+e^u) + \frac{u}{2} + \log 2 - 1 \tag{19}$$

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 (upper bound)**.** *If* $0 \prec mI \preceq X, Y \preceq MI$*, then*

$$D_R^2(X,Y) \leq 2\log(M/m)(J_{\ell d}(X,Y) + \gamma), \tag{20}$$

where $\gamma = d\log 2$.

Proof. Observe that

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

which implies the bound

$$J_{\ell d}(X,Y) + d\log 2 \geq \frac{1}{2}\|u\|_1. \tag{21}$$

Since $u^T u \leq \|u\|_\infty \|u\|_1$ (Hölder's inequality), using (21) we immediately obtain the bound

$$D_R^2(X,Y) = \|u\|_2^2 \leq 2\|u\|_\infty (J_{\ell d} + \gamma), \tag{22}$$

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 gradient and the Hessian of D_{le} do not have closed forms.

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

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 is expected Frobenius distance is isotropic and thus its balls are spherical, while AIRM and KLDM induce convex balls. Against these plots, and as was pointed by Theorem 3, the isosurfaces of JBLD are convex in some range while become concave as the radius goes large.

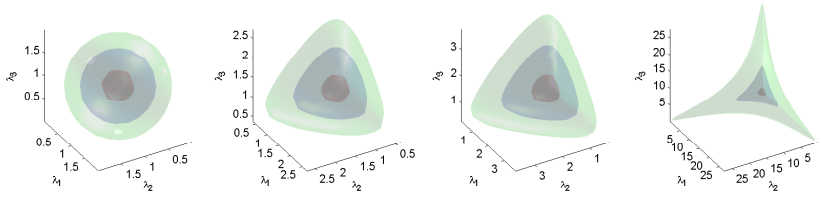


Figure 1: Isosurface plots for various distance measures. First, distances for arbitrary three dimensional covariances from the identity matrix are computed, and later isosurfaces corresponding to fixed distances of 0.1, 0.5 and 1 are plotted. The plots show the surfaces for: (from left) Frobenius distance,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

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 3 Cholesky factorizations (for $X+Y$, X and Y), each at a cost of $(1/3)d^3$ flops [36]. 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[1]. Table 2 shows that computing $\nabla J_{\ell d}$ can be even more than 100 times faster than ∇D_R. This speed proves critical for NN retrieval, or more generally when using any algorithm that depends on gradients of the similarity measure, e.g., see [37] and the references therein. Table 3 provides a summary of the various metrics, their gradients and computational complexities.

4 Fast Nearest Neighbor Retrieval using JBLD

Now we turn to the key application that originally 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

[1]From a technical point, $J_{\ell d}$ computation for matrices over $d = 13$ was seen 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.

to facilitate rapid NN retrieval. Towards this end, we chose to use the metric tree data structure as we wanted to show the performance on an exact NN algorithm for covariances and for 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 unlikely to occupy candidate neighbors to a query. To this end, we derive below a kmeans algorithm on $J_{\ell d}$ which will later be used to build the metric tree on covariances.

$metric$	$D^2(X,Y)$	$FLOPS$	$Gradient(\nabla_{\mathrm{x}})$
AIRM	$\|\log(X^{-1/2}YX^{-1/2})\|_{\mathrm{F}}^2$	$4d^3$	$2X^{-1}\log(XY^{-1})$
LERM	$\|\log(X) - \log(Y)\|_{\mathrm{F}}^2$	$\frac{8}{3}d^3$	$2X^{-1}(\log X - \log Y)$
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	$\log\left\lvert\frac{X+Y}{2}\right\rvert - \frac{1}{2}\log\lvert XY\rvert$	d^3	$(X+Y)^{-1} - \frac{1}{2}X^{-1}$

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

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

In this section, we derive a K-Means clustering algorithm based on $J_{\ell d}$. Let $S_1, S_2, \cdots, S_n$ be the input covariances that we need to be clustered. A standard K-Means algorithm gives rise to the following optimization problem:

$$\min_{C_1, C_2, \cdots, C_K} \sum_{k=1}^{K} \sum_{S \in C_k} J_{\ell d}(X_k, S), \tag{23}$$

where X_k is the *centroid* of cluster C_k. Following the traditional K-Means algorithm, we can alternate between the centroid computation and the clustering stages to minimize (23). The only significant step then amounting to the computation of the centroid for the kth cluster, which can be written as:

$$F := \min_{X_k} \sum_{S \in C_k} J_{\ell d}(X_k, S) \tag{24}$$

$$:= \min_{X_k} \sum_{S \in C_k} \log |\frac{X_k + S}{2}| - \frac{1}{2} \log |X_k S| \tag{25}$$

Unfortunately, as we saw earlier, $J_{\ell d}$ is neither a Bregman divergence, nor is it convex and thus we cannot use the traditional centroid computation. The good news is that, we can write (25) 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 $F_{cave}(X_k, S) = \sum_{S \in C_k} \log |\frac{X_k + S}{2}|$. Such a combination of convex and concave objectives can be efficiently solved using Majorization-Minimization through the Convex-ConCave Procedure (CCCP) [38]. 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:

$$X_k^{t+1} = \operatorname*{argmin}_{X_k} F_{vex}(X_k, S) - X_k^T \nabla_{X_k} F_{cave}(X_k^t, S). \tag{26}$$

Substituting (26) in (25), later taking the gradient of (25) with respect to X_k and setting it to zero (recall that now we have a convex approximation to (25)), we have:

$$\sum_{S \in C_k} \nabla_{X_k} F_{vex}(X_k^{t+1}, S) = -\sum_{S \in C_k} \nabla_{X_k} F_{cave}(X_k^t, S). \tag{27}$$

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

$$X_k^{t+1} = \left[\frac{1}{|C_k|} \sum_{S \in C_k} \left(\frac{S + X_k^t}{2} \right)^{-1} \right]^{-1}. \tag{28}$$

Convergence of the CCCP procedure is tied to the compactness of the solution space. Unfortunately, the space of SPD matrices is of *non-compact* type [39] with a non-positive sectional curvature; the latter property implying that the barycenter of a set of covariances in the respective Riemannian manifold need not be unique [40]. Thus, in the following we investigate the convergence of the fixed point iteration in (28).

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 $t \in [0, 1]$,*

$$f(tX + (1-t)Y) \leq tf(X) + (1-t)f(Y). \tag{29}$$

Proof. See Exercise V.1.15 [23] for details. □

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

Proof. See Corollary 7.7.4 [41]. □

Theorem 13. *Let $S_1, S_2, \cdots, S_n$ be the input covariances and let X^* be the centroid returned found by* (28)*. Then X^* lies in the* compact *interval*

$$\left(\frac{1}{n}\sum_{i=1}^{n} S_i^{-1}\right)^{-1} \leq X^* \leq \frac{1}{n}\sum_{i=1}^{n} S_i \tag{30}$$

Proof. Proving the left inequality: Applying Lemma 11 to (28), we have:

$$X^{-1} \leq \frac{1}{n}\sum_{i=1}^{n}\left(\frac{S_i^{-1} + X^{-1}}{2}\right) \tag{31}$$

$$\leq \frac{1}{n}\sum_{i=1}^{n}\frac{S_i^{-1}}{2} + \frac{1}{2}X^{-1}. \tag{32}$$

Now, applying Lemma 12, the result follows.

Proving the right inequality: As one can see, the right side of (28) is essentially the harmonic mean of $\frac{X+S_i}{2}$ for $i = 1, 2, \cdots, n$. Using the fundamental inequality that harmonic mean is always less than or equal to the arithmetic mean, we have the result. □

Theorem 14. *Let $\{X^t\}$ (for $t \geq 1$) be the sequence of successive iterates generated as per* (28)*. Then, $X^t \to X^*$, where X^* is a stationary point of* (25)*.*

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 [42] (CCCP-II, Theorem 8), the iterations in (28) converges for a suitable initialization inside the compact set. □

4.2 NN Using Metric Tree

As we mentioned earlier, we decided to use a metric tree for the task of efficient NN retrieval on covariance datasets. Metric Trees (MT) [43] are one of the fundamental tree based algorithms for fast NN retrieval useful when the underlying similarity measure 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 kmeans 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:

$$\mathcal{D}(X_c, q) > \min_{X; d(X,C)=R} \mathcal{D}(X, q) \tag{33}$$

where X is called the projection of q onto the ball with centroid C, radius R and $\mathcal{D}$ is some distance function. If (33) is satisfied, then the sibling node should be explored, otherwise it can be pruned. When $\mathcal{D}$ is a metric, (33) has a simple solution utilizing the triangle inequality as is described in [44]. 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 sub-trees centroids and worst case distances of the query to any candidate node in this sub-tree, as described in [43].

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, later providing simulation results exposing various aspects of our metric, 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:

$$Accuracy@K = \frac{1}{|Q|} \sum_{q \in Q} \frac{|G_q^K \cap M_q^K|}{|G_q^K|}. \tag{34}$$

Note that *Accuracy@K* as defined in (34) 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 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 Frechet mean using the iterative approximation algorithm described in [45]. The centroid for KLDM boils down to computing the solution of a Riccati equation as described in [46]. 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 10D 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. Next, we consider the various experiments and present the results.

5.2.1 Accuracy Against Noise

Given that the metrics 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 such that small perturbations of a poorly conditioned data matrices can lead to large metric distances, 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. Fig. 2 shows the average accuracy values 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 $J_{\ell d}$-kmeans algorithm against the kmeans 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 kmeans 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 perfectly 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.

5.3 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.

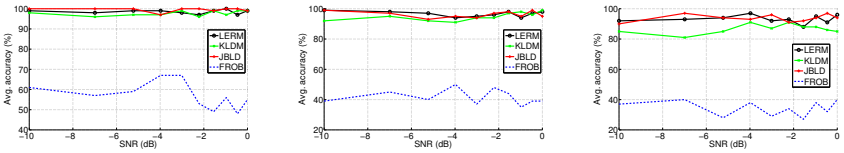
5.3.1 Tracking using Integral Images

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 [47] was used, while the vehicle tracking video was taken from the ViSOR repository[2]. 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. We used the color and the first order gradient features for the covariances. Figures 6(a),6(b), and 6(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

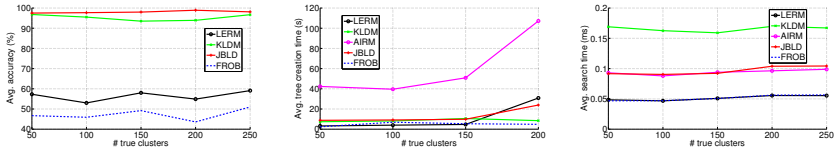
Another important application of covariances has been in texture segmentation [4] which has further application in DT-MRI, background subtraction [12], etc. In Figure 6(e), we present a few qualitative results from segmentation on the Brodatz texture dataset. Each of the images were a combination of 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 kmeans algorithm for the texture mixture, later segregating the patches using their cluster labels.

[2]http://www.openvisor.org



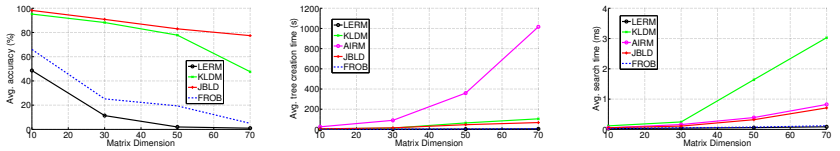
(a) n=10x10 (b) n=20x20 (c) n=40x40

Figure 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. MFD stands for the Matrix Frobenius Distance.



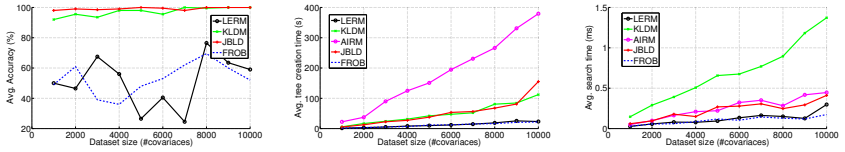
(a) (b) (c)

Figure 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)

Figure 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)

Figure 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.

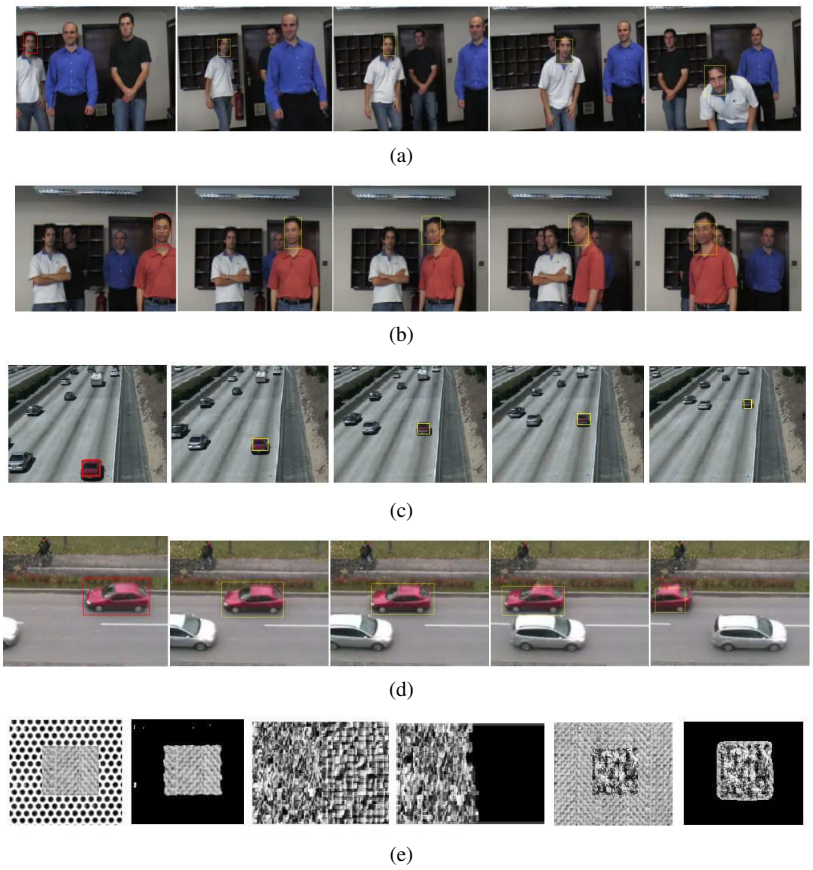


Figure 6: Tracking using JBLD on covariances computed from integral images: (a) affine face tracking, (b) tracking face with pose variations, (c), (d) vehicle tracking, and (e) shows results from texture segmentation. The red rectangle in the first image in each row shows the object being tracked. The yellow rectangles in the subsequent images are the nearest objects returned by JBLD. (e) 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.

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 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. See Figure 7 for sample images from each dataset.

Texture Dataset: Texture recognition has been one of the oldest applications of covariances spanning a variety of domains, e.g., DT-MRI, 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 [48]. 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. The covariance matrices computed such feature vectors on all the pixels inside the patch constituted one such data matrix and approximately 5K 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 [48] 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 [13] 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 [49], the KTH dataset[3] and the UT tower dataset [50]. 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 [10], 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 [51], which consists of approximately 31K face images

[3] http://www.nada.kth.se/cvap/actions/

mainly collected from newspapers. We used the same approach as in [10] 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[4]. 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.

(a)

(b)

(c)

Figure 7: Sample images from the various datasets used in our real world data experiments: 7(a) texture images from the Brodatz dataset, 7(b) Faces in the Wild dataset, and 7(c) people appearance tracking dataset.

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

[4] http://cvlab.epfl.ch/research/body/surv/#data

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

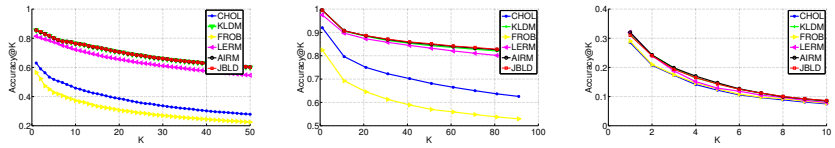
real-time applications (such as tracking) cannot spend time in building a metric tree. In this section, we analyze the performance of JBLD in terms of accuracy and retrieval speed on each of the datasets we described in the previous section.

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 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)

Figure 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 kmeans 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 kmeans 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 over 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) [52] method, the idea being 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 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. 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) defined over the set of positive-definite (covariance) matrices. The measure has several desirable theoretical properties including inequalities relating it to other metrics for covariances. More importantly, it was shown to outperform the Riemannian metric in speed, without any drop in accuracy. Further, we showed results for computing the centroid of covariance matrices under our metric, followed by an application to nearest neighbor retrieval using a metric tree. Experiments validated the effectiveness of the measure. Going forward, we would like to investigate the applicability of JBLD in classification and regression settings.

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.

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 *Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[7] J. Malcolm, Y. Rathi, and A. Tannenbaum, "A graph cut approach to image segmentation in tensor space," in *CVPR*, june 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] H. Min, N. Papanikolopoulos, C. Smith, and V. Morellas, "Feature-based covariance matching for a moving target in multi-robot following," in *The 19th Mediterranean Conf. on Control and Automation*, 2011.

[10] 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.

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

[12] R. Caseiro, J. Henriques, and J. Batista, "Foreground segmentation via background modeling on riemannian manifolds," in *ICPR*, 2010, pp. 3570–3574.

[13] 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.

[14] 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.

[15] Y. Shinohara, T. Masuko, and M. Akamine, "Covariance clustering on Riemannian manifolds for acoustic model compression," in *ICASSP*, 2010, pp. 4326–4329.

[16] 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.

[17] 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.

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

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

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

[21] 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.

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

[23] R. Bhatia, *Matrix analysis*. Springer Verlag, 1997, vol. 169.

[24] 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.

[25] 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.

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

[27] 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.

[28] B. Kulis, M. Sustik, and I. Dhillon, "Low-rank Kernel Learning with Bregman Matrix Divergences," *JMLR*, vol. 10, pp. 341–376, 2009.

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

[30] Y. Censor and S. A. Zenios, *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1997.

[31] F. Nielsen and R. Nock, "On the centroids of symmetrized bregman divergences," *Arxiv preprint arXiv:0711.3242*, 2007.

[32] ——, "Jensen-Bregman Voronoi diagrams and centroidal tessellations," in *Intl. Symp. on Voronoi Diagrams in Science and Engg.*, 2010, pp. 56–65.

[33] A. Banerjee, D. Boley, and S. Acharyya, "Symmetrized Bregman Divergences and Metrics," *The Learning Workshop*, 2009.

[34] P. Chen, "Bregman metrics and their applications," Ph.D. dissertation, University of Florida, 2007.

[35] S. Sra, "Positive definite matrices and the symmetric stein divergence," *http://arxiv.org/abs/1110.1773*, 2011.

[36] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Johns Hopkins University Press, 1996.

[37] 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.

[38] A. Yuille and A. Rangarajan, "The concave-convex procedure," *Neural Computation*, vol. 15, no. 4, pp. 915–936, 2003.

[39] A. Terras, *Harmonic Analysis on Symmetric Spaces and Applications II*. Springer-Verlag, New York, 1988.

[40] H. Le, "Locating fréchet means with application to shape spaces," *Advances in Applied Probability*, vol. 33, no. 2, pp. 324–338, 2001.

[41] R. Horn and C. Johnson, *Matrix analysis*. Cambridge University Press, 1990.

[42] B. Sriperumbudur and G. Lanckriet, "On the convergence of the concave-convex procedure," *NIPS*, vol. 22, pp. 1759–1767, 2009.

[43] 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.

[44] S. Brin, "Near neighbor search in large metric spaces," in *VLDB*, 1995.

[45] D. Bini and B. Iannazzo, "Computing the karcher mean of symmetric positive definite matrices," *LAA*, 2011.

[46] T. Myrvoll and F. Soong, "On divergence based clustering of normal distributions and its application to HMM adaptation," in *European Conference on Speech Communication and Technology*, 2003, p. 1517.

[47] E. Maggio, E. Piccardo, C. Regazzoni, and A. Cavallaro, "Particle PHD filtering for multi-target visual tracking," in *ICASSP*, vol. 1, 2007.

[48] 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, Jan 1999.

[49] 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.

[50] 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.

[51] 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.

[52] J. Beis and D. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," in *CVPR*, 1997, pp. 1000–1006.