# Efficient Similarity Search for Covariance Matrices via the Jensen-Bregman LogDet Divergence

Anoop Cherian
Dept. of CS & Engg.
University of Minnesota
cherian@cs.umn.edu

Suvrit Sra
MPI for Intelligent Systems
Tübingen, Germany
suvrit@tuebingen.mpg.de

Arindam Banerjee
Dept. of CS & Engg.
University of Minnesota
banerjee@cs.umn.edu

Nikolaos Papanikolopoulos
Dept. of CS & Engg.
University of Minnesota
npapas@cs.umn.edu

## Abstract

*Covariance matrices provide compact, informative feature descriptors for use in several computer vision applications, such as people-appearance tracking, diffusion-tensor imaging, activity recognition, among others. A key task in many of these applications is to compare different covariance matrices using a (dis)similarity function. A natural choice here is the Riemannian metric corresponding to the manifold inhabited by covariance matrices. But computations involving this metric are expensive, especially for large matrices and even more so, in gradient-based algorithms. To alleviate these difficulties, we advocate a novel dissimilarity measure for covariance matrices: the* Jensen-Bregman LogDet Divergence*. This divergence enjoys several useful theoretical properties, but its greatest benefits are: (i) lower computational costs (compared to standard approaches); and (ii) amenability for use in nearest-neighbor retrieval. We show numerous experiments to substantiate these claims.*

## 1. Introduction

An ever increasing number of computer vision and machine learning tasks depend on structured data where instead of vectors, one uses richer representations of data such as graphs, strings, or matrices. The focus of this paper is on a particular class of such structured data, namely the positive-definite matrices. These matrices arise naturally as covariance matrices and because they offer compact, informative, and empirically beneficial feature descriptors, they are by now popular in a variety of applications. Some recent applications are mentioned below.

In multi-camera tracking [12,25] covariance matrices derived from appearance silhouettes are known to be robust to noise as well as to affine transformations, to illumination changes, and to variations in the camera parameters. In [26] it was shown how covariance matrices can be computed in real-time using integral images, thus making them an ideal platform for video surveillance applications. In the medical arena Diffusion Tensor Imaging (DTI) is an established technique that depends on $3 \times 3$ positive-definite matrices which are used to track the diffusion of water molecules in the human brain, with applications such as diagnosis of neuro-psychiatric illnesses [1, 34]. Further, it has also been shown in computational anatomy that positive-definite deformation tensors generated from Magnetic Resonance Imaging (MRI) can better model changes in the brain structures [17]. Covariances computed from a bank of Gabor filters are used for robust face recognition in [27], while SIFT based covariances are proposed for emotion recognition in [33]. Other recent applications include covariances of optical flow for human action recognition [32], and covariances from acoustic features for speech recognition [31].

However, these successful applications suffer from a common drawback: whenever distance or similarity computations with covariances are required, the corresponding algorithms tend to be very slow. This slowness arises from the fact that covariances do not conform to Euclidean geometry, but rather form a Riemannian manifold, which makes defining similarity measures between covariance matrices 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 other 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.

Driven by these concerns, we take a closer look at similarity computation for covariance matrices, for which we introduce the Jensen-Bregman LogDet (JBLD) divergence. We discuss some 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.

To put our work into perspective, let us recall some standard similarity measures for covariance matrices. The simplest but naive approach is to view $dxd$ 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., $\ell_p$-distance functions, etc.). But these vectorial measures ignore the manifold structure of covariance matrices and can therefore be inferior choices. A more suitable choice is to consider the manifold structure of positive-definite matrices and use the corresponding geodesic distance: the *Affine Invariant Riemannian Metric (AIRM)* [5, 28] defined for $X$, $Y$ in $\mathcal{S}^d_{++}$, the set of $d \times d$ positive-definite matrices, by:

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

where $\log(\cdot)$ is the matrix logarithm. This metric enjoys several useful theoretical properties [5], and is perhaps the most widely used similarity measure for covariance matrices. But it can be unattractive as it requires eigenvalue computations or sometimes even matrix logarithms, which for larger matrices causes significant slowdowns.

Amongst the many measures that have been proposed to replace AIRM, a closely related one is the *Log-Euclidean Riemannian Metric* (LERM) [2]

$$D_{LE}(X, Y) := \|\log(X) - \log(Y)\|_{\mathrm{F}}, \qquad (2)$$

which uses the logarithmic map of covariance matrices to turn them into symmetric matrices which can then be handled as objects in ordinary Euclidean space. Applications exist in visual tracking [18] and stereo matching [14]. However, using this metric requires pre-processing the dataset by computing matrix logarithms, which can dramatically increase the computational costs. Yet another alternative is the symmetrized KL-Divergence Measure (KLDM) for positive-definite matrices [20], though for our application its accuracy on NN is poor. Other similarity measures on covariance matrices may be found in [11].

In contrast to AIRM and LERM the similarity measure that we propose 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. These gains come at a price: our measure is not a metric. But this limitation is not that severe because we can still exploit convexity to build a fast NN retrieval procedure based on our similarity measure.

We note that NN retrieval for covariance matrices itself is still an emerging area, so literature on it is scarce. In [30], an attempt is made to adapt NN techniques from vector spaces to non-Euclidean spaces, while [9] 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.

## 2. Jensen-Bregman LogDet Divergence

We first recall some basic definitions and then present our new 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 [15].

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

$$d_\phi(x, y) := \phi(x) - \phi(y) - \langle x - y, \nabla\phi(y)\rangle\,, \qquad (3)$$

where $\phi : S \subseteq \mathbb{R}^d \to \mathbb{R}$ is a strictly convex function of Legendre-type on $\mathrm{int}(\mathrm{dom}\, S)$ [4, 6]. From (3) 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 [4, 8], 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 [3, 22, 23], where assuming $s = (x + y)/2$,

$$J_\phi(x, y) := \tfrac{1}{2}\big(d_\phi(x, s) + d_\phi(s, y)\big), \qquad (4)$$

or even variants that satisfy the triangle inequality [3, 10].

Both (3) and (4) can be naturally extended to matrix divergences (over Hermitian matrices) by composing the convex function $\phi$ with the eigenvalue map $\lambda$, and replacing the inner-product in (3) 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

$$J_{\ell d}(X, Y) := \log\left|\frac{X + Y}{2}\right| - \frac{1}{2}\log|XY|. \qquad (5)$$

where $|\cdot|$ denotes the determinant; this divergence is obtained from the matrix version of (4) by using $\phi(X) = -\log|X|$ as the seed function. It is easy to see that $J_{\ell d}$ is symmetric, nonnegative, and definite. Moreover, it is invariant under congruence transformations, ($J_{\ell d}(AXA^T, AYA^T) = J_{\ell d}(X, Y)$ for invertible $A$), and under inversion ($J_{\ell d}(X, Y) = J_{\ell d}(X^{-1}, Y^{-1})$).

Less trite is the connection to the Riemannian metric which we summarize below in Theorem 1. This connection also lends additional support to using $J_{\ell d}$ as a proxy for the AIRM $D_R^2$ (or $\sqrt{J_{\ell d}}$ as a proxy for $D_R$).

**Theorem 1** (Bounds)**.** *Let $X$, $Y \in \mathcal{S}^d_{++}$. Then,*

$$J_{\ell d}(X, Y) \le D_R^2(X, Y). \qquad (6)$$

*Additionally, if* $0 \prec mI \preceq X, Y \preceq MI$*, then*

$$D_R^2(X,Y) \le 2\log(M/m)(J_{\ell d}(X,Y)+\gamma), \tag{7}$$

*where* $\gamma = d\log 2$*.*

*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

$$J_{\ell d}(X,Y) = \sum\nolimits_{i=1}^{d} (\log(1+e^{u_i}) - u_i/2 - \log 2), \tag{8}$$

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 (6) 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, $f(u) \ge f(u^*) = 0$ for all $u \in \mathbb{R}$, which in turn implies that

$$\sum\nolimits_{i=1}^{d} f(u_i) = D_R^2(X,Y) - J_{\ell d}(X,Y) \ge 0. \tag{9}$$

To prove the next inequality (7), first observe that

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

which implies the bound

$$J_{\ell d}(X,Y) + d\log 2 \ge \tfrac{1}{2}\|u\|_1. \tag{10}$$

Since $u^T u \le \|u\|_\infty \|u\|_1$ (Hölder's inequality), using (10) we immediately obtain the bound

$$D_R^2(X,Y) = \|u\|_2^2 \le 2\|u\|_\infty (J_{\ell d} + \gamma), \tag{11}$$

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

**Computational Advantages.** The greatest advantage of this new measure 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 [13]. 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:

$$\nabla_X D_R^2(X,Y) = X^{-1}\log(XY^{-1}),$$
$$\nabla_X J_{\ell d}(X,Y) = (X+Y)^{-1} - \tfrac{1}{2}X^{-1}.$$

Table 2 shows that computing $\nabla J_{\ell d}$ can be even more than 100 times faster than $\nabla D_R$. This speed proves critical for NN retrieval, or more generally when using any algorithm that depends on gradients of the similarity measure.

| $d$ | $D_R$ | $J_{\ell d}$ |
|---|---|---|
| 5 | $.025 \pm .012$ | $.035 \pm .007$ |
| 10 | $.038 \pm .005$ | $.042 \pm .009$ |
| 20 | $.085 \pm .006$ | $.064 \pm .009$ |
| 40 | $.334 \pm .332$ | $.127 \pm .012$ |
| 80 | $1.23 \pm .055$ | $.393 \pm .050$ |
| 200 | $8.198 \pm .129$ | $2.223 \pm .169$ |
| 500 | $77.311 \pm .568$ | $22.186 \pm 1.223$ |
| 1000 | $492.743 \pm 15.519$ | $119.709 \pm 1.416$ |

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 | $24.899 \pm 1.126$ | $.270 \pm .047$ |
| 80 | $99.486 \pm 5.181$ | $.921 \pm .028$ |
| 200 | $698.873 \pm 39.602$ | $8.767 \pm 2.137$ |
| 500 | $6377.742 \pm 379.173$ | $94.837 \pm 1.195$ |
| 1000 | $40443.059 \pm 2827.048$ | $622.289 \pm 37.728$ |

Table 2. Average times (millisecs/trial) to compute gradients; computed over 1000 trials (except for the last two experiments, where to save time only 100 trials were used) to reduce variance.

## 3. Fast Nearest Neighbor Retrieval for $J_{\ell d}$

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 to facilitate rapid NN retrieval. Typical NN data structures require the similarity measure used for computing "nearness" to be a metric. Thus, $J_{\ell d}$ raises a major complication since it (and even $\sqrt{J_{\ell d}}$) fails to satisfy the triangle inequality. Fortunately, this difficultly can be tackled by adapting the framework of Bregman-Ball Trees (BBT) [7]. But for an effective adaptation of the BBT, we must efficiently perform two subtasks: (i) partition the input data using clustering; and (ii) compute Jensen-Bregman projections onto appropriate convex (Bregman-Ball) sets for input query points. We discuss both subtasks below.

### 3.1. K-Means with $J_{\ell d}$

Suppose $S_1, \ldots, S_n$ are the input covariance matrices that we wish to partition into $K$ clusters. A standard K-means type approach proceeds by minimizing

$$\min_{\mathcal{C}_1,\ldots,\mathcal{C}_K} \quad \sum\nolimits_{k=1}^{K} \sum\nolimits_{S\in\mathcal{C}_k} J(X_k, S), \tag{12}$$

where $X_k$ represents the "centroid" of cluster $\mathcal{C}_k$. To solve (12) we alternate between cluster assignment and centroid-computation. The latter is the only non-trivial step, so we discuss it only and omit the rest for brevity.

It suffices to describe centroid computation for a single cluster. Let $\mathcal{C} \neq \emptyset$ be an arbitrary cluster—its centroid is computed by solving

$$\min_X \quad \Theta(X) := \sum\nolimits_{S\in\mathcal{C}} J_{\ell d}(X, S). \tag{13}$$

If Problem (13) has a solution, say $X^*$, then it must satisfy the first-order optimality conditions:

$$-2\nabla_X \Theta(X^*) = |\mathcal{C}|\,(X^*)^{-1} - \sum\nolimits_{S\in\mathcal{C}} \left(\frac{X^*+S}{2}\right)^{-1} = 0,$$

where $|\mathcal{C}|$ denotes size of $\mathcal{C}$. In other words, $X^*$ must satisfy

$$(X^*)^{-1} = \frac{1}{|\mathcal{C}|}\sum\nolimits_{S\in\mathcal{C}} \left(\frac{X^*+S}{2}\right)^{-1}. \tag{14}$$

Now we show that $X^*$ lies in a compact set, a fact that will be very useful. To that end, we first recall two basic lemmas (see e.g., [5] for proofs).

**Lemma 2.** *The function $f(X) = X^{-1}$ is matrix convex on positive-definite matrices, i.e., if $X, Y \succ 0$, then*

$$f(tX + (1-t)Y) \preceq tf(X) + (1-t)f(Y), \quad t \in [0,1].$$

**Lemma 3.** *If $X \succeq Y \succ 0$. Then, $Y^{-1} \succeq X^{-1}$.*

Applying Lemma 2 to Equation (14) we obtain

$$(X^*)^{-1} \preceq \frac{1}{|\mathcal{C}|}\sum\nolimits_{S\in\mathcal{C}} \left(\frac{(X^*)^{-1}}{2} + \frac{S^{-1}}{2}\right), \tag{15}$$

while inverting (14) and then invoking Lemma 2 we get

$$X^* \preceq \frac{1}{|\mathcal{C}|}\sum\nolimits_{S\in\mathcal{C}} \left(\frac{X^*+S}{2}\right). \tag{16}$$

Simplifying (15) and (16), and using Lemma 3 we have

$$\left(\tfrac{1}{|\mathcal{C}|}\sum\nolimits_{S\in\mathcal{C}} S^{-1}\right)^{-1} \preceq X^* \preceq \left(\tfrac{1}{|\mathcal{C}|}\sum\nolimits_{S\in\mathcal{C}} S\right). \tag{17}$$

The bounds (17) imply that $X^*$ lives within a compact set; moreover this set is convex. Thus, the effective domain of $\Theta$ in (13) can be restricted to be this compact set, which guarantees existence of a minimum [29].

Since $\Theta$ is strictly convex, this minimum is unique. In light of this fact, one can use numerous convex optimization procedures, the simplest of which includes Gradient-Projection (GP). But taking cue from the literature on nonlinear matrix equations (e.g., [16]), we prefer iterating the following nonlinear map

$$\mathcal{G}Y = \frac{1}{|\mathcal{C}|}\sum\nolimits_{S\in\mathcal{C}} \left(\frac{Y^{-1}+S}{2}\right)^{-1}, \tag{18}$$

using the iteration:

$$Y_{k+1} = \mathcal{G}Y_k, \qquad k = 0, 1, \ldots. \tag{19}$$

If $Y_0$ is chosen properly, then similar to (17) we can show that each of the iterates $Y_k$ produced by satisfies

$$\left(\tfrac{1}{|\mathcal{C}|}\sum\nolimits_{S\in\mathcal{C}} S\right)^{-1} \preceq Y_k \preceq \left(\tfrac{1}{|\mathcal{C}|}\sum\nolimits_{S\in\mathcal{C}} S^{-1}\right). \tag{20}$$

Assuming $Y_0$ is chosen to satisfy (20), we can further show (inductively) $Y_{k+1} \succeq Y_k$. Thus, the sequence $\{Y_k\}_{k\geq 0}$ is monotonic, and since it lies in a compact set, it must converge to a unique limit point. This limit must be the minimum point because it satisfies the optimality condition (14). Thus, iteration (19) computes the centroid by solving (13).

### 3.2. NN Using Bregman Ball Trees

Bregman Ball Trees (BBT) were introduced in [7] as an alternative to metric trees for enabling fast NN retrieval when the underlying similarity measure is a Bregman divergence. Even though JBLD is not a Bregman divergence, we can still adapt the BBT framework, as shown below.

**Building BBT.** As suggested in [7], to build the ball tree, we perform top-down bi-partitioning of the input space by recursively applying the JBLD-K-Means algorithm (introduced above). Each partition of the BBT is identified by a centroid and the ball radius. For $n$ data points, the total build time of the tree is $O(n \log n)$. 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 againsts doing k-means on it.

**Querying using BBT.** 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 $X_c$ (see Fig. 1). 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 $q$

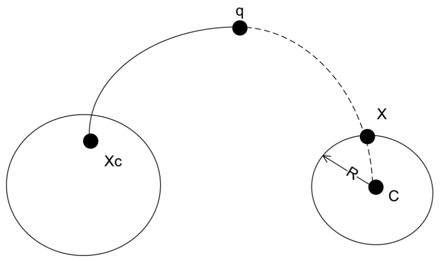


Figure 1. Bregman projection: Point $X$ is the projection of $q$ onto the ball with centroid $C$ and radius $R$. The curve segments show the geodesics connecting the points in the space of covariances.

than $X_c$. To this end, we solve the following optimization problem on each of the sibling centroids $C$:

$$d(X_c, q) > \min_{X; d(X,C)=R} d(X, q) \tag{21}$$

where $X$ is called the projection of $q$ onto the ball with centroid $C$, radius $R$ and $d$ is some distance function. If (21) is satisfied, then the sibling node should be explored, otherwise it can be pruned. When $d$ is a metric, (21) has a simple solution utilizing the triangle inequality. If $d$ happens to be a Bregman divergence, then the segment connecting $C$ and $q$ is a straight line and thus bisection can be used to compute the projection point $X$ [7].

This approach cannot be directly applied to our framework because our choice is $d = J_{\ell d}$, which is a symmetrized divergence, *not* a Bregman divergence. In [22, 24], this situation is investigated in detail; the bisection line search being replaced by a dichotomic search on the geodesic linking $q$ and $C$ to compute the unique projection. To this end, we iteratively bisect the geodesic between $q$ and $C$ using our efficient JBLD k-means algorithm until the projection point falls on the ball of radius R. Algorithm 1 details the various steps involved in the process.

**Algorithm 1** Projection Algorithm.

```
Require: $C$, $R$, $q$;
  Initialize $Y_{\min} \Leftarrow C$, $Y_{\max} \Leftarrow q$;
  repeat
    $X \Leftarrow$compute centroid of $Y_{\min}$ and $Y_{\max}$ using Eq. (19)
    $d \Leftarrow R - J_{\ell d}(X, C)$;
    if $d > 0$ then
      $Y_{\min} \Leftarrow X$
    else $\{d \leq 0\}$
      $Y_{\max} \Leftarrow X$
    end if
  until $|d|$ is less than a threshold
  return $X$
```

## 4. Experiments

We are now ready to describe our experimental setup and results to substantiate the effectiveness of the new similarity measure. We first discuss the performance metric on which our experiments are based, later providing simulation results, followed by the results on three real-world datasets. All algorithms were implemented in MATLAB and tested on a machine with 3GHz CPU and 3GB RAM.

**Accuracy:** Since many 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. For NN experiments, we create a database and a query set of $q$ items. For each query item $i$, $k$ ground truth neighbors ($G_i^k$) are found using linear scan, followed by $k$ nearest neighbors ($A_i^k$) retrieved using the respective algorithm. We define

$$Accuracy = \frac{1}{q} \sum_i \frac{|G_i^k \cap A_i^k|}{|G_i^k|}. \tag{22}$$

Note that *Accuracy* is technically equivalent to both the standard measures of *precision* and *recall* in our case.

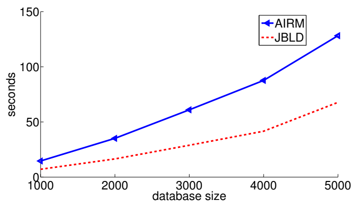
### 4.1. Simulations

**Clustering Performance:** In this section, we evaluate in a controlled setting a key component of our proposed method: the JBLD K-means algorithm. To this end, we created a base set of covariance matrices from a set of simulated feature vectors. Subsequently, noise of varying mean was added to these feature vectors to obtain a set of noisy covariances that formed a cluster. 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. Using this setup, we evaluate three different timing scenarios: (i) as database size increases; (ii) as matrix size increases; and (iii) as the number of true clusters grow. Each experiment varied only one of these three parameters. The results are shown in Fig. 2, and as is evident, across all three timing scenarios the clustering method based on the AIRM is significantly outperformed by our algorithm.

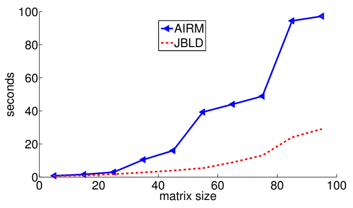
### 4.2. Real Data Experiments

**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 [21] was used, while the vehicle tracking video was taken from the ViSOR repository[1]. The images from the video were resized to $244 \times 320$ for speed and integral images computed on each frame. An input tracking

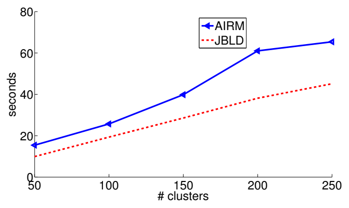
[1]http://www.openvisor.org



(a)

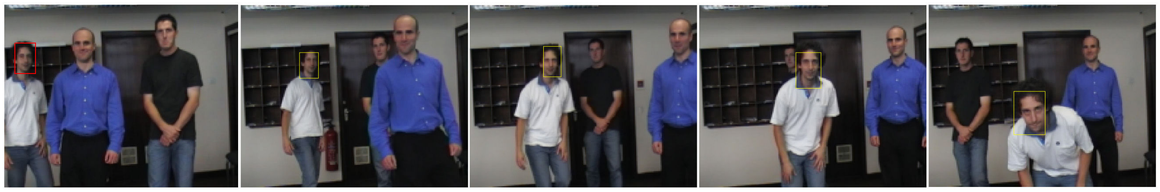


(b)

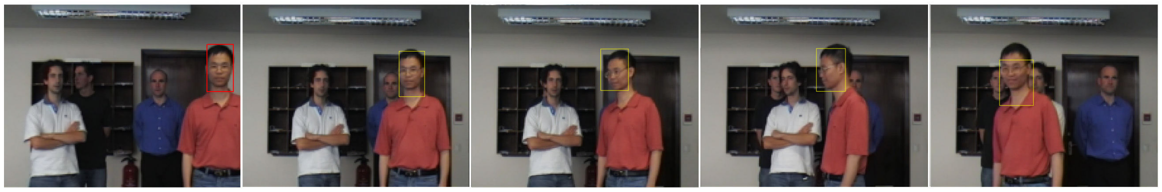


(c)

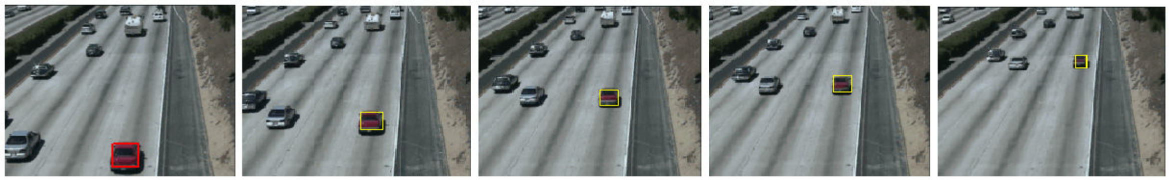
Figure 2. Clustering time for (a) increasing dataset size, (b) increasing matrix size, (c) increasing number true clusters.



(a)



(b)



(c)

Figure 3. Tracking using JBLD on covariances computed from integral images: (a) affine face tracking, (b) tracking face with pose variations, and (c) vehicle tracking. 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.

region was given at the beginning of the video, which is then tracked in subsequent images using the integral transform and covariances thus computed. We used the color and the first order gradient features for the covariances. Fig. 3 shows a few qualitative results of this experiment. We compared the window of tracking for both AIRM and JBLD, and found that they always fall at the same location in the video.

### 4.3. NN Performance via Exhaustive Search

Next, we evaluate the performance of the different metrics against the accuracy of search and speed of retrieval on real data. This is important, as most of the real-time applications (e.g., tracking) cannot afford to spend time in building a metric tree. We use three different covariance datasets: (i) Brodatz textures; (ii) People appearance tracking[2]; and (iii) FERET face[3]. See Fig. 4 for sample images from each dataset. A summary of these datasets follows.

**Brodatz Texture:** This dataset has approximately 160 true texture images of size $512 \times 512$. To create the covariances, we followed the suggestions in [26] and used the first and second order gradient based features. The final dataset contained 5K covariances.

**People Appearances:** For this database we used videos of people appearances tracked using multiple cameras. The background was first learned using a mixture of Gaussians, then the silhouettes of people were extracted. The first and second order image gradients along with the color information were used to obtain approximately 10K covariances of size $8 \times 8$.

**FERET Faces:** This dataset consisted of approximately 4K

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

[3]http://www.itl.nist.gov/iad/humanid/feret/

face images. We first applied a face detection algorithm[4] to extract the frontal features of the face, and then applied Gabor filters to these features based on the suggestions in [19]. From the resulting feature vectors, covariances of size $40 \times 40$ were then generated.

Figure 4. Sample images from the appearance tracking dataset (top), FERET face appearances (middle), and Brodatz texture database (bottom).

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 the Riemannian metric. The results are shown in Table 3: clearly the JBLD measure outperforms all the other metrics in speed, without compromising accuracy.

| Dataset( size) | AIRM | JBLD | LERM | KLDM |
|---|---|---|---|---|
| Texture (4428) | | | | |
| Avg. Accuracy(%) | 88.0 | **88.0** | 75.2 | 79.6 |
| Avg. Time (s) | 120.88 | **110.25** | 61.54 | 263.02 |
| Appearance (8596) | | | | |
| Avg. Accuracy(%) | – | **100** | 83.3 | 70.0 |
| Avg. Time (s) | 303.43 | **274.2** | 150.17 | 592.26 |
| Face (3010) | | | | |
| Avg. Accuracy(%) | 60.0 | **60.5** | 43.5 | 56.5 |
| Avg. Time (s) | 836.47 | **356.68** | 106.55 | 830.15 |

Table 3. Performance of the similarity measure on different datasets for one 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.

## 4.4. NN Performance Using BBT

**Building the Tree:** The time required to build the NN data structure plays a critical role in the deployment of the measure. Thus, in Table 4 we show a comparison of the initialization time for the BBT (against a metric tree using AIRM) as the database size grows. Even though the build time seems quite similar for small datasets, AIRM takes dramatically more time than JBLD with increasing size. The reason being the Karcher mean algorithm [25] used for the computation of the cluster centroids under the AIRM metric which requires the gradient computations, that we know is computationally inferior to JBLD. The iterations for JBLD K-means were found to converge under a threshold of 1e–3 in 8–12 iterations.

| Dataset Size | AIRM(s) | JBLD(s) |
|---|---|---|
| 3K | 16.52 | **15.66** |
| 4K | 45.86 | **22.49** |
| 5K | 321.5 | **30.56** |

Table 4. Comparison of initialization times (seconds) for the BBT.

**NN Retrieval:** Finally, we compare accuracy and the speed of retrieval of JBLD against AIRM. Table 5 shows the results. As is evident, without any noticeable drop in the accuracy, JBLD achieves superior performance in NN retrieval. We assume the AIRM to be the baseline, as we already know from Table 3 that they have the same base accuracy.

| Dataset | Metric | AIRM | JBLD |
|---|---|---|---|
| Texture | Avg. Accuracy(%) | – | **99.6** |
| | Avg. Time (ms) | 203.27 | **10.02** |
| Appearance | Avg. Accuracy(%) | – | **99.0** |
| | Avg. Time (ms) | 383.06 | **11.54** |
| Face | Avg. Accuracy(%) | – | **100** |
| | Avg. Time (ms) | 620.20 | **114.11** |

Table 5. Average performance per query on NN using BBT and AIRM as baseline. The NN accuracy for LERM was found to be less than 5% and thus not shown. Refer Table 3 for *accuracy* comparison with other methods.

## 5. Conclusion

We introduced a novel similarity measure based on the Jensen-Bregman LogDet Divergence defined over the set of positive-definite (i.e., covariance) matrices. The measure has several desirable theoretical properties including inequalities relating it to the Riemannian metric for covariances. More importantly, it was shown to outperform the Riemannian metric in speed, without any drop in application performance. Although this similarity measure does not satisfy the triangle inequality, we showed that this problem can be easily circumvented using the convexity of the measure by invoking the Bregman-ball tree framework. Experiments substantiated the effectiveness of the measure. Going forward, we would like to investigate the applicability of this similarity measure in classification and regression settings, and to study its theoretical properties further.

[4]http://staff.science.uva.nl/ zivkovic/download.html

## 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 Tran. Med. Imaging*, 20(11):1131–1139, 2002. 1

[2] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Log-Euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic Resonance in Medicine*, 56(2):411–421, 2006. 2

[3] A. Banerjee, D. Boley, and S. Acharyya. Symmetrized Bregman Divergences and Metrics. *The Learning Workshop*, 2009. 2

[4] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *JMLR*, 6:1705–1749, 2005. 2

[5] R. Bhatia. *Positive definite matrices*. Princeton Univ Press, 2007. 2, 4

[6] L. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217, 1967. 2

[7] L. Cayton. Fast nearest neighbor retrieval for bregman divergences. In *ICML*, pages 112–119, 2008. 3, 4, 5

[8] Y. Censor and S. A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1997. 2

[9] R. Chaudhry and Y. Ivanov. Fast approximate nearest neighbor methods for non-Euclidean manifolds with applications to human activity analysis in videos. *ECCV*, pages 735–748, 2010. 2

[10] P. Chen. *Bregman metrics and their applications*. PhD thesis, University of Florida, 2007. 2

[11] I. Dryden, A. Koloydenko, and D. Zhou. Non-Euclidean statistics for covariance matrices, with applications to diffusion tensor imaging. *Annals*, 3(3):1102–1123, 2009. 2

[12] F. Porikli, and O. Tuzel. Covariance tracker. *CVPR*, 2006. 1

[13] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, third edition, 1996. 3

[14] Q. Gu and J. Zhou. A similarity measure under Log-Euclidean metric for stereo matching. In *ICPR*, pages 1–4, 2009. 2

[15] B. Kulis, M. Sustik, and I. Dhillon. Low-rank Kernel Learning with Bregman Matrix Divergences. *Journal of Machine Learning Research*, 10:341–376, 2009. 2

[16] H. Lee and Y. Lim. Invariant metrics, contractions and nonlinear matrix equations. *Nonlinearity*, 21:857–878, 2008. 4

[17] N. Lepore, C. Brun, Y. Chou, M. Chiang, R. Dutton, K. Hayashi, E. Luders, O. Lopez, H. Aizenstein, A. Toga, et al. Generalized tensor-based morphometry of HIV/AIDS using multivariate statistics on deformation tensors. *IEEE Tran. Med. Imaging*, 27(1):129–141, 2007. 1

[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. 2

[19] C. Liu. Gabor-based kernel PCA with fractional power polynomial models for face recognition. *IEEE PAMI*, 26(5):572–581, 2004. 7

[20] M. Moakher, and P. Batchelor. Symmetric positive-definite matrices: From geometry to applications and visualization. *Springer Berlin Heidelberg, Chapter 17, isbn-978-3-540-31272-7*, 2006. 2

[21] E. Maggio, E. Piccardo, C. Regazzoni, and A. Cavallaro. Particle PHD filtering for multi-target visual tracking. In *ICASSP*, volume 1, 2007. 5

[22] F. Nielsen and R. Nock. On the centroids of symmetrized bregman divergences. *Arxiv preprint arXiv:0711.3242*, 2007. 2, 5

[23] F. Nielsen and R. Nock. Jensen-Bregman Voronoi diagrams and centroidal tessellations. In *Int. Symp. Voronoi Diagrams in Science and Engineering*, pages 56–65, 2010. 2

[24] F. Nielsen, P. Piro, and M. Barlaud. Tailored Bregman ball trees for effective nearest neighbors. In *Eur. Work. Comput. Geo.*, pages 16–18, 2009. 5

[25] O. Tuzel, F. Porikli, and P. Meer. Covariance Tracking using model update based on Lie Algebra. *CVPR*, 2006. 1, 7

[26] O. Tuzel, F.Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. *ECCV*, 2006. 1, 6

[27] Y. Pang, Y. Yuan, and X. Li. Gabor-based region covariance matrices for face recognition. *IEEE Tran. Circuits and Sys. for Video Tech.*, 18(7):989–993, 2008. 1

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

[29] R. T. Rockafellar and R. J.-B. Wets. *Variational Analysis*. Springer, 1997. 4

[30] P. Turaga and R. Chellappa. Nearest-neighbor search algorithms on non-Euclidean manifolds for computer vision applications. In *Indian Conf. Comp. Vis. Graph. and Img. Proc.*, pages 282–289, 2010. 2

[31] C. Ye, J. Liu, C. Chen, M. Song, and J. Bu. Speech emotion classification on a Riemannian manifold. *Adv. Multimedia Inf. Proc.*, pages 61–69, 2008. 1

[32] C. Yuan, W. Hu, X. Li, S. Maybank, and G. Luo. Human action recognition under log-Euclidean Riemannian metric. *ACCV*, pages 343–353, 2010. 1

[33] W. Zheng, H. Tang, Z. Lin, and T. Huang. Emotion recognition from arbitrary view facial images. *ECCV*, pages 490–503, 2010. 1

[34] 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*, 102(480):1085–1102, 2007. 1