Recursive Estimation of the Stein Center of SPD Matrices & its Applications[*]

Hesamoddin Salehian
salehian@cise.ufl.edu

Guang Cheng
gcheng@cise.ufl.edu

Baba C. Vemuri[†]
vemuri@cise.ufl.edu

Jeffrey Ho
jho@cise.ufl.edu

Department of CISE, University of Florida
Gainesville, FL 32611, USA

Abstract

Symmetric positive-definite (SPD) matrices are ubiquitous in Computer Vision, Machine Learning and Medical Image Analysis. Finding the center/average of a population of such matrices is a common theme in many algorithms such as clustering, segmentation, principal geodesic analysis, etc. The center of a population of such matrices can be defined using a variety of distance/divergence measures as the minimizer of the sum of squared distances/divergences from the unknown center to the members of the population. It is well known that the computation of the Karcher mean for the space of SPD matrices which is a negatively-curved Riemannian manifold is computationally expensive. Recently, the LogDet divergence-based center was shown to be a computationally attractive alternative. However, the LogDet-based mean of more than two matrices can not be computed in closed form, which makes it computationally less attractive for large populations. In this paper we present a novel recursive estimator for center based on the Stein distance – which is the square root of the LogDet divergence – that is significantly faster than the batch mode computation of this center. The key theoretical contribution is a closed-form solution for the weighted Stein center of two SPD matrices, which is used in the recursive computation of the Stein center for a population of SPD matrices. Additionally, we show experimental evidence of the convergence of our recursive Stein center estimator to the batch mode Stein center. We present applications of our recursive estimator to K-means clustering and image indexing depicting significant time gains over corresponding algorithms that use the batch mode computations. For the latter application, we develop novel hashing functions using the Stein distance and apply it to publicly available data sets, and experimental results have shown favorable comparisons to other competing methods.

1. Introduction

Symmetric Positive-Definite (SPD) matrices are commonly encountered in many fields of Science and Engineering. For instance, as covariance descriptors in Computer Vision, diffusion tensors in Medical Imaging, Cauchy-Green tensors in Mechanics, metric tensors in numerous fields of Science and Technology. Finding the mean of a population of such matrices as a representative of the population is also a commonly addressed problem in numerous fields. Over the past several years, there has been a flurry of activity in finding the means of a population of such matrices due to the abundant availability of matrix-valued data in various domains e.g., diffusion tensor imaging [1] and Elastography [16] in medical image analysis, covariance descriptors in computer vision [14, 4], dictionary learning on Riemannian manifolds [17, 7, 22] in machine learning, etc.

It is well known that the space of $n \times n$ SPD matrices equipped with the $GL(n)$-invariant metric is a Riemannian symmetric space [8] with negative sectional curvature [19], which we will henceforth denote by P_n. Finding the mean of data lying on P_n can be achieved through a minimization process. More formally, the mean of a set of N data $\mathbf{x}_i \in P_n$ is defined by $\mathbf{x}^* = argmin_{\mathbf{x}} \sum_{i=1}^{N} d^2(\mathbf{x}_i, \mathbf{x})$, where d is the chosen distance/divergence. Depending on the choice of d, different types of means are obtained. Many techniques have been published on computing the mean SPD matrix based on different kinds of similarity distances/divergences. In [20], symmetrized Kullback-Liebler divergence was used to measure the similarities between SPD matrices, and the mean was computed in closed-form and applied to texture and diffusion tensor image (DTI) segmentation. Karcher mean was obtained by using the GL-invariant (GL denotes the general linear group i.e., the group of (n, n) invertible matrices) Riemannian metric on

[*] This research was funded in part by the NIH grant NS066340 to BCV.

[†] Corresponding author

P_n and used for DTI segmentation in [11] and for interpolation in [13]. Another popular distance is the so called Log-Euclidean distance introduced in [6] and used for computing the mean. More recently, in [5] the $LogDet$ divergence was introduced and applied for tensor clustering and covariance tracking. Each one of these distances and divergences possesses their own properties with regards to invariance to group transformations/operations. For instance, the natural geodesic distance derived from the GL-invariant metric is GL-invariant. The LogEuclidean distance is invariant to the group of rigid motions and so on. Among these distances/divergences, the $LogDet$ divergence was shown to posses interesting bounding properties with regards to the natural Riemannian distance in [5] and much more computationally attractive for computing the mean. However, no closed-form expression exists for computing the mean using the $LogDet$ divergence, for more than two matrices. When the number of samples in the population is large and the size of SPD matrices is larger, it would be desirable to have a computationally more attractive algorithm for computing the mean using this divergence.

A recursive form can effectively address this problem. Recursive formulation leads to considerable efficiency in mean computation, because for each new sample, all one needs to do is to update the old. Consequently, the algorithm only needs to keep track of the most recently computed mean, while computing the mean in a batch mode requires one to store all previously given samples. This can prove to be quite memory intensive for large problems. Thus, by using a recursive formula we can significantly reduce the time and space complexity. Recently, in [3] recursive algorithms to estimate the mean SPD matrix based on the natural GL-invariant Riemannian metric and symmetrized KL-divergence were proposed and applied to DTI segmentation. Also in [21] a recursive form of *Log*-Euclidean based mean was introduced. In this paper we present a novel recursive algorithm for computing the mean of a set of SPD matrices, using the Stein metric.

The Jensen-Bregman LogDet (JBLD) divergence was recently introduced in [5] for $n \times n$ SPD matrices. Compared to the standard approaches, the JBLD has a much lower computational cost since the formula does not require any eigen decompositions of the SPD matrices. Moreover, it has been shown that it is useful for use in nearest neighbor retrieval [5]. However, JBLD is not a metric on P_n, since it does not satisfy the triangle inequality. In [17] the authors proved that the square root of JBLD is a metric, which is called *Stein* metric. Unfortunately, the mean of SPD matrices based on the Stein metric can not be computed in a closed form, for more than two matrices [2, 5]. Therefore, iterative optimization schemes are applied to find the mean for a given set of SPD matrices. The computational efficiency of these iterative schemes is effected considerably especially when the number of samples and size of matrices is large. This makes the Stein-based mean inefficient for computer vision applications that deal with huge amounts of data. In this paper, *we introduce an efficient recursive formula to compute the Stein mean.* To illustrate the effectiveness of the proposed algorithm, we first show that applying the recursive Stein mean estimator to the problem of K-means clustering leads to a significant gain in running time when compared to using the batch mode Stein center, as well as other recursive mean estimators based on aforementioned distances/divergences. Furthermore, we develop a novel hashing technique which is a generalization of the work in [9] to SPD matrices.

The key contributions of this paper are: (i) derivation of a closed form solution to the weighted Stein center of two matrices which is then used in the formulation of the recursive form for the Stein center estimation of more than two SPD matrices. (ii) Empirical evidence of convergence of the recursive estimator of Stein mean to the batch mode Stein mean is shown. (iii) A new hashing technique for image indexing and retrieval using covariance descriptors. (iv) Synthetic and real data experiments depicting significant gains in computation time for SPD matrix clustering and image retrieval (using covariance descriptor features), using our recursive Stein center estimator.

The rest of paper is organized as follows: in Section 2 we present the recursive algorithm to find the Stein distance-based mean of a set of SPD matrices. Section 3 presents the empirical evidences of the convergence of recursive Stein mean estimator to the Stein expectation. Furthermore, we present a set of synthetic and real data experiments showing the improvements in running time of SPD matrix clustering and hashing. Finally, we present the conclusions in Section 4.

2. Recursive Stein Mean Computation

The action of the general linear group of $n \times n$ invertible matrices (denoted by $GL(n)$) on P_n defines the natural group action and is defined as follows: $\forall g \in GL(n), \forall X \in P_n, X[g] = gXg^T$, where T denotes the matrix transpose operation. Let A and B be any two points in P_n. The geodesic distance on this manifold is defined by the following $GL(n)$-invariant Riemannian metric:

$$d_R(A, B)^2 = trace(Log(A^{-1}B)^2), \tag{1}$$

where Log is the matrix logarithm. The mean of a set of N SPD matrices based on the above Riemannian metric is called the Karcher mean, and is defined as

$$X^* = argmin_X \sum_{i=1}^{N} d_R^2(X, X_i), \tag{2}$$

where X^* is the Karcher mean, and X_i are the given matrix-valued data. However, computation of the distance using (1), requires eigen decomposition of the matrix, which for large matrices slows down the computation considerably. Furthermore, the minimization problem (2) does not have a closed form solution in general (for more than two matrices) and iterative schemes such as the gradient descent technique are employed to find the solution.

Recently in [5], the Jensen-Bregman LogDet (JBLD) divergence was introduced to measure similarity/dissimilarity between SPD matrices. It is defined as

$$D_{LD}(A, B) = logdet(\frac{A+B}{2}) - \frac{1}{2}logdet(AB), \quad (3)$$

where A and B are two given SPD matrices. It can be seen that JBLD is much more computationally efficient than the Riemannian metric, as no eigen decomposition is required. JBLD is however not a metric, because it does not satisfy the triangle inequality. However, in [17], it was shown that the square root of JBLD divergence is a metric, i.e., it is non-negative definite, symmetric and satisfies the triangle inequality. This new metric is called *Stein* metric and is defined by,

$$d_S(A, B) = \sqrt{D_{LD}(A, B)}, \quad (4)$$

where D_{LD} is defined in (3). Clearly, Stein metric can also be computed efficiently. Accordingly, the mean of a set of SPD tensors, based on Stein metric is defined by

$$X^* = argmin_X \sum_{i=1}^{N} d_S^2(X, X_i). \quad (5)$$

For a probability distribution $\mathbf{P}(x)$ on P_n, we can define its Stein expectation as

$$\mu_S^* = \arg\min_{\mu \in P_n} \mathbf{E}_S(\mu),$$

where

$$\mathbf{E}_S(\mu) = \int_{P_n} {d_S}^2(x, \mu)\, \mathbf{P}(x) dx.$$

Before turning to the recursive algorithm for computing Stein expectation, we briefly remark on the metric geometry of P_n equipped with the Stein metric. Both the Stein metric d_S and the $GL(n)$-invariant Riemannian metric d_R are $GL(n)$-invariant. However, their similarity does not go beyond this $GL(n)$-invariance. In particular, the Stein metric is not a Riemannian metric, and more precisely, we have the following two important features of the Stein metric:

- P_n equipped with the Stein metric is **not** a length space, i.e., the distance $d_S(A, B)$ between two points is not given by the length of a shortest curve (path) joining A and B. Let M to the Stein mean of A and B, defined in Eq 5. Assuming there is a shortest curve γ on P_n connecting A and B that corresponds to the Stein distance. Then, based on the triangle inequality, since $d_S^2(A, P) + d_S^2(B, P) \leq 2(\frac{d_S(A,B)}{2})^2, \forall P \in P_n$, M should be the mid-point of γ or $d_S(A, M) = \frac{1}{2}d_S(A, B)$. However, the last equality is not satisfied for Stein distance in general. This implies Stein metric-based distance cannot be represented as the length of shortest curve on P_n.
- P_n equipped with the Stein metric satisfies the Reshetnyak inequality presented in [18]. The proof of this claim however is beyond the scope of this paper.

The two features together paint an interesting picture of the geometry of P_n endowed with the Stein metric: The first feature means that this new geometry defies easy characterization since it is not even a length space, arguably the most general type of spaces studied by geometers. However, the second feature shows that the Stein geometry has some characteristics of a negatively-curved metric space since Reshetnyak inequality is one of a few important properties satisfied by all metric (length) spaces with non-positive curvature (e.g., [18]). We remark that for metric spaces with non-positive curvature, there are existence and uniqueness results on the geometrically-defined expectations (similar to the Stein expectation above) [18]. Unfortunately, none of these known results are applicable to P_n endowed with the Stein metric because it is not a length space. Nevertheless, we are able to establish the following,

Theorem 1 *For any distribution $\mathbf{P}(x)$ on P_n with finite $\mathbf{L}^2$-Stein moment, its Stein expectation exists and is unique.*

The $\mathbf{L}^2$-Stein moment is defined as (for any $\mu \in P_n$)

$$\int_{P_n} {d_S}^2(x, \mu)\, \mathbf{P}(x) dx,$$

and it is easy to show that the finiteness condition on the distribution $\mathbf{P}(x)$ is independent of the chosen point μ. For the simplest case of P_1, the proof is straightforward and we present it here and defer the case for P_n to a later paper. Since $P_1 = \mathbb{R}_+$ and ${d_S}^2(x, y) = \log(\frac{x+y}{2}) - \frac{1}{2}\log(xy)$ for $x, y \in \mathbb{R}_+$, the first-order optimality condition for μ_S^* yields

$$\int_{\mathbb{R}_+} \frac{1}{\mu_S^* + x} \mathbf{P}(x) dx = \frac{1}{2\mu_S^*}.$$

Therefore, μ_S^* must be a zero of the function $\mathbf{F}(\mu)$:

$$\mathbf{F}(\mu) = \int_{\mathbb{R}_+} \frac{\mu - x}{\mu + x} \mathbf{P}(x) dx.$$

We show that $\mathbf{F}$ must have exactly one zero, and hence the existence and uniqueness of the Stein expectation on $\mathbb{R}_+$:

This follows from

$$\frac{d\mathbf{F}(\mu)}{d\mu} = \int_{\mathbb{R}_+} \frac{2x}{(\mu+x)^2}\mathbf{P}(x)dx > 0$$

for all $\mu \in \mathbb{R}_+$, and clearly, we have $\lim_{\mu\to 0}\mathbf{F}(\mu) = -1$ and $\lim_{\mu\to\infty}\mathbf{F}(\mu) = 1$.

2.1. Recursive Algorithm for Stein Expectation

Having established the existence and uniqueness of the Stein expectation, we now present a recursive algorithm for computing the same. Let $X_i \in P_n$ be i.i.d samples of a distribution $\mathbf{P}$. The recursive Stein mean can be defined as

$$M_1 = X_1 \qquad (6)$$

$$M_{k+1}(w_{k+1}) = argmin_M (1-w_{k+1})d_S^2(M_k, M) + w_{k+1}d_S^2(X_{k+1}, M) \qquad (7)$$

where $w_{k+1} = \frac{1}{k+1}$, M_k is the old mean of k SPD matrices, X_{k+1} is the new incoming sample and M_{k+1} is the updated mean for $k+1$ matrices. Note that (7) can be thought of as a weighted Stein mean between the old mean and the new sample point, with the weight being set to be the same as in Euclidean mean update.

Now, we show that (7) has a closed form solution for SPD matrices. Let A and B be two matrices in P_n. The weighted mean of A and B, denoted by C, with the weights being w_a and w_b such that $w_a + w_b = 1$, should minimize (7). Therefore, one can compute the gradient of this objective function and set it to zero to find the minimizer C

$$w_a[(\frac{C+A}{2})^{-1} - C^{-1}] + w_b[(\frac{C+B}{2})^{-1} - C^{-1}] = 0 \quad (8)$$

Multiplying both sides of (8) by matrices C, $C+A$ and $C+B$ in a right order yields:

$$CA^{-1}C + (w_b - w_a)C(I - A^{-1}B) - B = 0 \qquad (9)$$

It can be verified that for any matrices A, B and C in P_n, satisfying (9), the matrices $A^{-\frac{1}{2}}CA^{-\frac{1}{2}}$ and $A^{-\frac{1}{2}}BA^{-\frac{1}{2}}$ commute. In other words

$$A^{-1}CA^{-1}B = A^{-1}BA^{-1}C \qquad (10)$$

Left multiplication of (9) by A^{-1} yields

$$A^{-1}CA^{-1}C + (w_b - w_a)A^{-1}C(I - A^{-1}B) = A^{-1}B \qquad (11)$$

The equation above can be rewritten in a matrix quadratic form as the following, by using the equality in (10)

$$(A^{-1}C + \frac{(w_b - w_a)}{2}(I - A^{-1}B))^2 = A^{-1}B + \frac{(w_b - w_a)^2}{4}(I - A^{-1}B)^2 \qquad (12)$$

Taking the square root of both sides and rearranging yields

$$A^{-1}C = \sqrt{A^{-1}B + \frac{(w_b - w_a)^2}{4}(I - A^{-1}B)^2} - \frac{(w_b - w_a)}{2}(I - A^{-1}B) \qquad (13)$$

Therefore, the solution of (9) for C can be written in the following closed form

$$C = A[\sqrt{A^{-1}B + \frac{(w_b - w_a)^2}{4}(I - A^{-1}B)^2} - \frac{w_b - w_a}{2}(I - A^{-1}B)] \qquad (14)$$

It can be verified that the solution in (14) satisfies Eq. (10). Therefore, Eq. (7) for recursive Stein mean estimation can be rewritten as

$$M_{k+1} = M_k[\sqrt{M_k^{-1}X_{k+1} + \frac{(2w_{k+1}-1)^2}{4}(I - M_k^{-1}X_{k+1})^2} - \frac{2w_{k+1}-1}{2}(I - M_k^{-1}X_{k+1})] \quad (15)$$

with w_{k+1}, M_k, M_{k+1} and X_{k+1} being the same as in (7).

If P_n equipped with the Stein metric were a *global Non-Positive Curvature (NPC) space* [18], Sturm shows that M_k converges to the unique Stein expectation as $k \to \infty$ [18]. Unfortunately, as shown earlier, it is not even a length space, let alone being a global NPC space. Therefore, a proof of convergence for the recursive estimator for Stein metric-based center would be considerably more delicate and involved. However, we present empirical evidence for 100 SPD matrices randomly drawn from a log-Normal distribution to indicate that the recursive estimates of the Stein mean converge to the batch mode Stein mean (see Fig. 1).

3. Experiments

In this section, we present several synthetic and real data experiments. All running times reported in this section are for experiments performed on a machine with a single 2.67GHz Intel-7 CPU with 8GB RAM.

3.1. Performance of the Recursive Stein Center

To illustrate the performance of the proposed recursive algorithm, we generate 100 i.i.d samples form a Log-normal distribution [15] on P_3 with the variance and expectation set to 0.25 and the identity matrix, respectively. Then, we input these random samples to the recursive Stein mean estimator (RSM) and its non-recursive counterpart (SM). To compare the accuracy of RSM and SM we compute the Stein distance between the ground truth and the computed

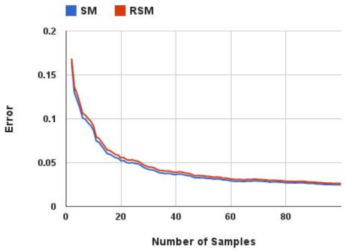

Figure 1. Error comparison for the recursive (red) versus non-recursive (blue) Stein mean computation for data on P_3. (**Image best viewed in color**)

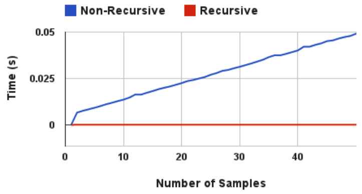

Figure 2. Running time comparison for the recursive (red) versus non-recursive (blue) Stein mean computation for data on P_3. (**Image best viewed in color**)

estimate. Further, the computation time for each newly acquired sample is recorded. We repeat this experiment 20 times and plot the average error and the average computation time at each step. Fig. 1 shows the accuracies of RSM and SM in the same plot. It can be seen that for the given 100 samples, as desired, the accuracy of the recursive and non-recursive algorithms are almost the same. Further, Fig. 2 shows that RSM takes the same computation time for all given samples, while the time taken by SM increases almost linearly with respect to the number of matrices. It should be noted that RSM computes the new mean by a simple matrix operations, e.g., summations and multiplications, which makes it very fast for any number of samples. This means that the recursive Stein-based mean is computationally far more efficient, especially when the number of samples is very large and the samples are input incrementally, for example as in clustering and some segmentation algorithms.

3.2. Application to K-means Clustering

In this section we evaluate the performance of the proposed recursive algorithm applied to K-means clustering. The two fundamental components of the K-means algorithm at each step are: (i) distance computation and (ii) the mean update. Due to the computational efficiency involved in evaluating the Stein metric, the distances can be efficiently computed. However, due to the lack of a closed form formula for computing the Stein mean, the cluster center update is more time consuming, and to tackle this problem, we employ our recursive Stein mean estimator.

More specifically, at the end of each K-means iteration, only the matrices that change their cluster memberships in previous iteration are considered. Then, each cluster center is updated only by applying the changes incurred by the matrices that most recently changed cluster memberships. For instance, let C_1^i and C_2^i be the centers of the first and second clusters, at the end of the i-th iteration. Also, let X be a matrix which has moved from the first cluster to the second one. Therefore, we can directly update C_1^i by removing X from it to get C_1^{i+1}, and adding X to C_2^i in its update, to get C_2^{i+1}. This will significantly decrease the computation time of the K-means algorithm, especially for huge datasets.

To illustrate the gained efficiency resulting from using our proposed recursive Stein mean (RSM) update, we compared its performance to the non-recursive Stein mean (SM), as well as the following three widely used mean computation methods: Karcher mean (KM), symmetric Kullback-Leibler mean (KLsM) and Log-Euclidean (LEM) mean. Furthermore, to show the effectiveness of the Stein metric in K-means distance computation, we included comparisons to the following recursive mean estimators recently introduced in literature: Recursive Log-Euclidean mean (RLEM) [21], Recursive Karcher Expectation Estimator (RKEE) and Recursive KLs-mean (RKLsM) in [3]. We should emphasize that for each of these mean estimators, we used its corresponding distance/divergence in the K-means algorithm.

The efficiency of the proposed K-means algorithm is investigated in the following set of experiments. We tested our algorithm in three different scenarios, with increasing (i) number of samples, (ii) matrix size, and (iii) number of clusters. For each scenario, we generated samples from a mixture of Log-normal distributions, where the expectation of each component is assumed to be the true cluster center. To measure the error in clustering, we compute the geodesic distance between each estimated cluster center and its true value, and take the summation of error values over all clusters.

Fig. 3 shows the time comparison between the aforementioned K-means clustering techniques. It is clearly evident that the proposed method (RSM) is significantly faster than other competing methods, in all the aforementioned settings of the experiment. There are two reasons that support the time efficiency of RSM: (i) recursive update of the Stein mean, which is achieved via the closed form expression in Eq. 15, (ii) fast distance computation, by exploiting the Stein metric, as the Stein distance is computed using a simple matrix determinant followed by a scalar logarithm, while the Log-Euclidean, GL-invariant Riemannian distances and the KLs divergence, require more complicated matrix operations, e.g., matrix logarithm, inverse and square root. Consequently, it can be seen in Fig. 3 that for large datasets, the recursive Log-Euclidean, Karcher and KLs-mean methods are as slow as their non-recursive counterparts, since a substantial portion of the running time is consumed in the distance computation involved in the algorithm.

Furthermore, Fig. 4 shows the errors defined earlier, for each experiment. It can be seen that, in all the cases, the accuracy of the RSM estimator is very close to the other competing methods, and in particular to the non-recursive Stein mean (SM) and Karcher mean (KM). Thus, in terms of accuracy, the proposed RSM estimator is as good as the best in the class but far more computationally efficient. These experiments verify that the proposed recursive method is a computationally attractive candidate for the task of K-means clustering in the space of SPD matrices.

3.3. Application to Image Retrieval

In this section, we present results of applying our recursive Stein mean estimator to the image hashing and retrieval problem. To this end, *we present a novel hashing function* which is a generalization of the spherical hashing applied to SPD matrices. The spherical hashing was introduced in [9] for binary encoding of large scale image databases. However, it can not be applied as is (without modifications) to SPD matrices, since it has been developed for inputs in a vector space. In this section we describe our extension of the spherical hashing technique in order to deal with SPD matrices (which are elements of a Riemannian manifold with negative sectional curvature).

Given a population of SPD matrices, our hashing function is based on the distances to a set of fixed *pivot* points. Let $P_1, P_2, ..., P_k$ be the set of such pivot points for the given population. We denote the hashing function by $H(X) = (h_1(X), ..., h_k(X))$, with X being the given SPD matrix, and each h_i is defined by

$$h_i(X) = \begin{cases} 0 & \text{if } dist(P_i, X) > r_i \\ 1 & \text{if } dist(P_i, X) \leq r_i \end{cases} \tag{16}$$

where $dist(.,.)$ denotes any distance defined on the manifold of SPD matrices. The value of $h_i(X)$ illustrates whether the given matrix X is inside the geodesic ball formed around P_i, with the radius r_i. In our experiments we used the Stein distance defined in Equation (4), because it is more computationally appealing for large datasets.

An appropriate choice of pivot points as well as radii is crucial to guarantee the accuracy of the hashing. In order to locate the pivot points we have employed the K-means clustering based on the Stein mean as discussed in Section 3.2. Furthermore, the radius r_i is determined so that the hashing function, h_i satisfies $Pr[h_i(X) = 1] = \frac{1}{2}$, which guarantees that each geodesic ball contains half of the samples. Based on this framework, each member of a set of $(n \times n)$ SPD matrices is mapped to a binary code with length k. To measure similarity/dissimilarity between binary codes, the spherical Hamming distance described in [9] is used.

In order to evaluate the performance of the proposed recursive Stein mean algorithm in this image hashing context, we compare the performance for locating the pivot points by four of the K-means clustering techniques discussed in Section 3.2: RSM, SM, RKEE and RLEM. Using the found pivot points, the retrieval precision for each method is experimentally evaluated and compared.

Experiments were performed on the COREL image database [12], which contains 10K images categorized into 80 classes. For each image a set of feature vectors were computed of the form $f = [I_r, I_g, I_b, I_L, I_A, I_B, I_x, I_y, I_{xx}, I_{yy}, |G_{0,0}(x,y)|, ..., |G_{2,1}(x,y)|]$, where the first three components represent the RGB color channels, the second three encode the Lab color dimensions, and the next four specify the first and second order gradients at each pixel. Further, as in [7], the $G_{u,v}(x,y)$ is the response of a 2D Gabor wavelet, centered at (x,y) with scale v and orientation u. Finally, for the set of N feature vectors extracted from each image, $f_1, f_2, ..., f_N$, a covariance matrix was computed using $Cov = \frac{1}{N}\sum_1^N (f_i - \bar{f})(f_i - \bar{f})^T$, where $\bar{f}$ is the mean vector. Therefore, from this dataset, ten thousand 16×16 covariance matrices were extracted.

To compare the time efficiency, we record the total time taken to compute the pivots and find the radii, for each aforementioned technique. Furthermore, a set of 1000 random queries were selected from the dataset, and for each query its 10 nearest neighbors were retrieved based on the spherical Hamming distance. The retrieval precision for each query was measured by the number of correct matches to the total number of retrieved images, namely 10. Total precision is then computed by averaging these (precision) values.

Fig. 5 shows the time taken by each method. As expected, it can be observed that the recursive Stein mean estimator significantly outperforms other methods, especially for longer binary codes. The recursive framework provides an efficient way to update the mean covariance matrix. Further, $RKEE$ which is based on the GL-invariant Riemannian metric is much more computationally expensive than the recursive Stein method. Fig. 6 shows the accuracy for each technique. It can be seen that the recursive Stein mean estimator provides almost the same accuracy as the non-recursive Stein as well as the $RKEE$. Therefore, the accuracy and computational efficiency of the proposed method makes it an appealing choice for image indexing and retrieval on huge datasets. Fig. 7 shows the outputs of the proposed system for four sample queries. Note that all of the retrieved images shown in Fig. 7 belong to the same class in the provided ground truth.

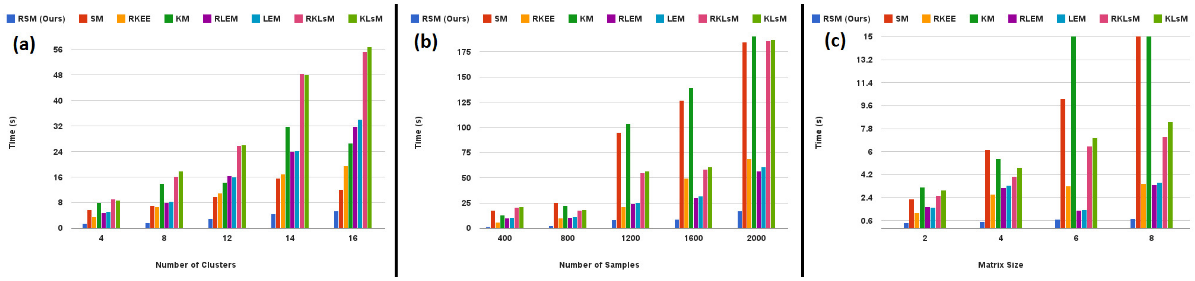

Figure 3. Running time comparison for the K-means clustering using non-recursive Stein, Karcher, Log-Euclidean, KLs-mean denoted by SM, KM, LEM and KLsM, respectively, as well as their recursive counterparts denoted by RSM, RKEE, RLEM and RKLsM . (**a**) Running time comparison for different numbers of clusters with the number of samples and matrix dimension fixed at 1000 and 2, respectively. (**b**) Running time comparison for different database sizes, from 400 to 2000, with 5 clusters, on P_2. (**c**) Running time comparison for different matrix dimensions In this experiment, 1000 samples and 3 clusters were used. The times taken by KM for (6×6) and (8×8) matrices were much larger than other methods (211 and 332 seconds, respectively) and well beyond the range used in the plot.

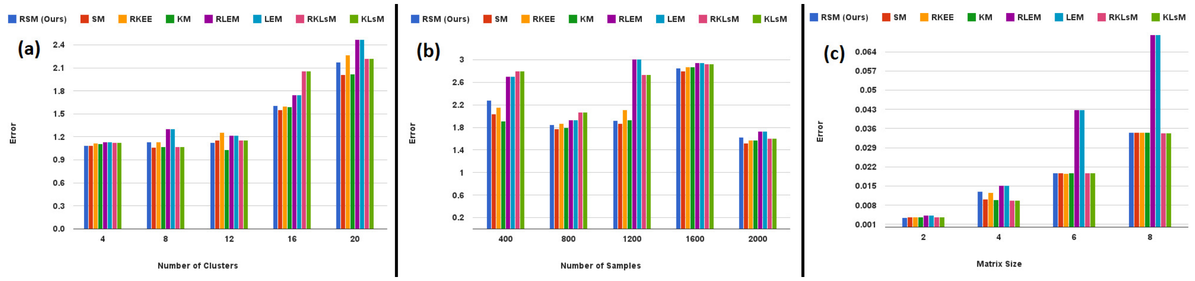

Figure 4. Error comparison for the K-means clustering using methods specified in Fig. 3. (a), (b) and (c) show the error comparison between the methods with varying number of clusters, number of samples and matrix dimensions, respectively.

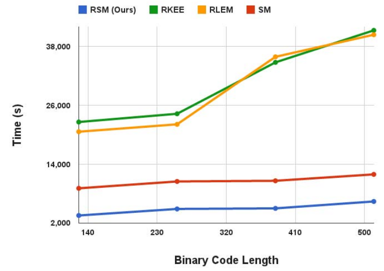

Figure 5. Running time comparison for the initialization of hashing functions, for recursive Stein mean (RSM), non-recursive Stein mean (SM), recursive LogEuclidean mean ($RLEM$) and recursive Karcher expectation estimator ($RKEE$), over increasing binary code lengths.

3.4. Application to Shape Retrieval

In this section, the image hashing technique presented in Section 3.3 is evaluated in a shape retrieval experiment, using the MPEG-7 database [10], which consists of 70 different objects with 20 shapes per object, for a total of 1400 shapes. To extract the covariance features from each shape, we first partition the image into four regions of equal area and compute a 2×2 covariance matrix from the (x, y) coordinates of the edge points in each region. Finally, we combined these matrices into a single block diagonal matrix, resulting in an 8×8 covariance descriptor.

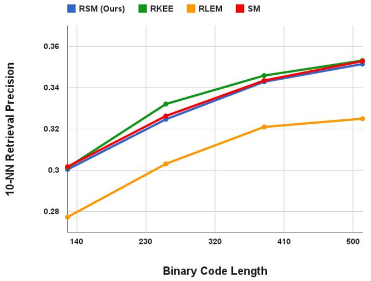

Figure 6. Retrieval accuracy comparison for the same collection of methods specified in Fig. 5

We used the same methods as in Section 3.3 to compare the shape retrieval speed and precision. Table 1 contains the retrieval precision comparison, and it can be seen that the RSM provides roughly the same retrieval accuracy as RKEE, while table 2 shows that RSM is significantly faster

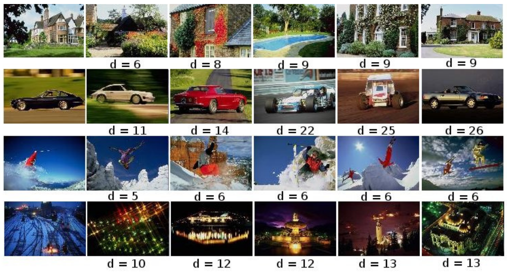

Figure 7. Sample results returned by the proposed retrieval system based on the recursive Stein mean using 640-bits binary codes. Query images are shown in the leftmost column and the remaining columns display the top five images returned by the retrieval system. The retrieved images are sorted in the increasing order of the Hamming distance to the query image, with Hamming distance specified under each returned image.

BC Length	RSM	SM	$RKEE$	$RLEM$
64	**60.67**	62.10	61.46	61.15
128	**63.59**	64.65	64.69	63.23
192	**69.69**	69.63	70.10	68.19
256	**73.13**	73.13	73.84	70.14

Table 1. Average shape retrieval precision (%) for the MPEG7 database using four different Binary Code (BC) lengths.

BC Length	RSM	SM	$RKEE$	$RLEM$
64	**48.76**	104.61	381.14	397.66
128	**53.44**	185.80	366.60	415.62
192	**89.04**	189.89	380.41	397.66
256	**105.33**	196.61	368.63	398.23

Table 2. Running time (in seconds) comparison for shape retrieval.

than all the competing methods.

4. Conclusions

In this paper, we have presented a novel recursive estimator for computing the Stein center/mean for a population of SPD matrices. The key contribution here is the derivation of a closed form solution for the computation of a weighted Stein mean for two SPD matrices which is then used in developing a recursive algorithm for computing the Stein mean of a population of SPD matrices. In the absence of a proven convergence, we presented compelling empirical results demonstrating the convergence of the recursive Stein mean estimator to the batch-mode Stein mean. Several experiments were presented showing superior performance of the recursive Stein estimator over the non-recursive counterpart as well as the recursive Karcher expectation estimator in K-means clustering and image retrieval. Another key contribution of this work is the design of hashing functions for the image retrieval application using covariance descriptors as features. Our future work will be focused on several new theoretical and practical aspects of the recursive estimator presented here.

References

[1] P. Basser, J. Mattiello, and D. LeBihan. MR diffusion tensor spectroscopy and imaging. *Biophysical Journal*, 66, 1994.

[2] Z. Chebb and M. Moakher. Means of hermitian positive-definite matrices based on the log-determinant divergence function. *Linear Algebra Appl.*, 40, 2012.

[3] G. Cheng, H. Salehian, and B. C. Vemuri. Efficient recursive algorithms for computing the mean diffusion tensor and applications to DTI segmentation. In *ECCV*, 2012.

[4] G. Cheng and B. C. Vemuri. A novel dynamic system in the space of SPD matrices with applications to appearance tracking. *SIAM J. Imaging Sciences*, 6(1):592–615, 2013.

[5] A. Cherian et al. Efficient similarity search for covariance matrices via the JB LogDet divergence. In *ICCV*, 2011.

[6] P. Fillard et al. Extrapolation of sparse tensor fields: application to the modeling of brain variability. In *IPMI*, 2005.

[7] M. Harandi et al. Sparse coding and dictionary learning for SPD matrices: A kernel approach. In *ECCV*, 2012.

[8] S. Helgason. *Differantial Geometry, Lie Groups, and Symmetric Spaces*. American Mathematical Society, 2001.

[9] J.-P. Heo et al. Spherical hashing. In *CVPR*, 2012.

[10] L. J. Latecki et al. Shape descriptors for non-rigid shapes with a single closed contour. In *CVPR*, 2000.

[11] C. Lenglet, M. Rousson, and R. Deriche. DTI segmentation by statistical surface evolution. *TMI*, 2006.

[12] J. Li and J. Z. Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *PAMI*, 2003.

[13] M. Moakher and P. G. Batchelor. *SPD Matrices: From Geometry to Applications and Visualization*. Visual. and Proc. of Tensor Fields, 2006.

[14] F. Porikli, O. Tuzel, and P. Meer. Covariance tracking using model update based on Lie algebra. In *CVPR*, 2006.

[15] A. Schwartzman. *Random ellipsoids and false discovery rates: Statistics for diffusion tensor imaging data*. PhD thesis, Stanford University, 2006.

[16] D. Sosa Cabrera et al. A tensor approach to elastography analysis and visualization. Visual. and Proc. of Tensor Fields, 2009.

[17] S. Sra. Positive definite matrices and the symmetric Stein divergence. *http://people.kyb.tuebingen.mpg.de/suvrit/*, 2011.

[18] K. T. Sturm. Probability measures on metric spaces of nonpositive curvature. In *Heat Kerels and Analysis on Manifolds, Graphs, and Metric Spaces*, 2003.

[19] A. Terras. *Harmonic Analysis on Symmetric Spaces and Applications*. Springer-Verlag, 1985.

[20] Z. Wang and B. C. Vemuri. An affine invariant tensor dissimilarity measure and its applications to tensor-valued image segmentation. In *CVPR*, 2004.

[21] Y. Wu, J. Wang, and H. Lu. Real-time visual tracking via incremental covariance model update on Log-Euclidean Riemannian manifold. In *CCPR*, 2009.

[22] Y. Xie, B. C. Vemuri, and J. Ho. Dictionary learning on Riemannian manifolds. In *MICCAI workshop on STMI*, 2012.