DOI: https://doi.org/10.15446/recolma.v57n2.115858

Revista Colombiana de Matemáticas
Volumen 57(2023)2, páginas 231-255

A geometric mean algorithm of symmetric positive definite matrices

Cálculo de una media geométrica en el cono de las matrices simétricas definidas positivas

Juan Olmos[1,✉], Juan Galvis[2], Fabio Martínez[1]

[1]Biomedical Imaging, Vision and Learning Laboratory (BIVL[2]ab). Universidad Industrial de Santander, Bucaramanga, Colombia

[2]Departamento de Matemáticas. Universidad Nacional de Colombia, Bogotá, Colombia

Abstract. This work introduces a geometric mean algorithm for positive definite matrices using generalized eigenvalue problems and Cholesky factorization. The geometric mean of a finite set of positive definite matrices minimizes the sum of square distances to all the matrices where the distance is an *affine-invariant* Riemannian metric in the manifold of the symmetric positive definite matrices S^N_{++}. In order to compute numerical approximations of the geometric mean several algorithms have been proposed. Some of these algorithms require the computation of several diagonalizations in each iteration. We show that by rewriting the iterations in terms of generalized eigenvalue problems, it is possible to omit some of the diagonalizations at the cost of introducing much less generalized eigenvalue problems that can be solved using Cholesky factorizations. We numerically compare the performance of classical methods and the modified algorithms that use generalized eigenvalue problems. The resulting method is applied to video analysis using the mean of covariance matrices as a compact descriptor for actions classification. The proposed mean descriptor with just 105 scalar values achieved an average accuracy of 75% over a public action video dataset.

Key words and phrases. covariance mean, action recognition, symmetric positive-definite matrices, generalized eigenvalues, Cholesky decomposition.

2020 Mathematics Subject Classification. 68W25, 68T45.

Resumen. Este trabajo presenta un algoritmo de media geométrica para matrices positivas definidas utilizando problemas de valores propios generalizados

y la factorización de Cholesky. La media geométrica de un conjunto finito de matrices positivas definidas minimiza la suma de los cuadrados de las distancias al conjunto de matrices, donde la distancia es una métrica de Riemann invariante afin en la variedad de matrices definidas positivas simétricas S_{++}^N. Para calcular aproximaciones numéricas de la media geométrica se proponen varios algoritmos. Algunos de estos algoritmos requieren el cálculo de varias diagonalizaciones en cada paso. Mostramos que al reescribir las iteraciones de estos pasos en términos de un problema de valores propios generalizados, es posible omitir algunas de las diagonalizaciones a costa de introducir problemas de valores propios que pueden resolverse utilizando factorizaciones de Cholesky. Comparamos numéricamente el rendimiento de los métodos clásicos y los algoritmos modificados que utilizan problemas de valores propios generalizados. El método resultante se aplica al análisis de vídeo utilizando la media de matrices de covarianza como un descriptor compacto para la clasificación de acciones. El descriptor medio propuesto con solo 105 valores escalares logró una precisión promedio del 75% en un conjunto de datos de video.

Palabras y frases clave. Media de covarianza, Reconocimiento de acciones, Matrices simétricas definidas positivas, Autovalores generalizados, Descomposión de Cholesky.

1. Introduction

The identification of patterns in image and video sequences that correspond with semantic labels is a common practice in the field of computer vision, particularly in the context of automatic classification and action recognition. Algorithms for performing these tasks are complex due to the high-variability of movements, camera variations, lighting changes, and subjectivity in the description of actions. Therefore, one important goal in this field is to design and develop image/video descriptors to address a specific application. The developed descriptors and methods must be efficient in order to be able to use them in online applications or on-the-fly action recognition.

The dimension of the descriptor is an important parameter to approach special problems that need fast computations. Most approaches produce overall high-dimensional descriptors, and are not usually used in basic devices or computers. There are also far too complex and require too much computer time when we want to make predictions in a short lapse of time. An approach that has been used to deal with these issues is the use of covariance matrices to compact spatio-temporal features that can be used to describe the video [11, 18].

If the video has K frames and each one is described by N features of dimension $W \times H$, then, a usual descriptor would have a total dimension of $K \times N \times W \times H$. In contrast, to have a compact descriptor we can compute the frame covariance matrices that describe how selected features are related in each frame. These covariance matrices have dimension $N \times N$. After a check for a possible regularization, these frame covariance matrices are elements of a

Riemannian manifold: the space of symmetric positive-definite matrices (S_{++}^N). This manifold has a special geometry structure where we can compute distances and unique geodesics. Hence we can compute statistics along the manifold to capture important information in a compact way.

Following several authors and previous articles in this area [19, 10, 3], this work uses the geometric mean as a statistic and compact descriptor of the set of frame covariance matrices. As mentioned in [5], the geometric mean of two symmetric positive definite matrices A and B, has been defined in several ways. A well-established definition is

$$A^{\frac{1}{2}}(A^{-\frac{1}{2}}B^{\frac{1}{2}}A^{-\frac{1}{2}})A^{\frac{1}{2}}. \tag{1}$$

Geometrically, this is the midpoint of the geodesic joining A and B. This observation motivated Moakher [19] to generalize the definition of geometric mean for more than two symmetric positive definite matrices based on a Riemannian metric. It is defined as the unique solution X of the matrix equation

$$\sum_{i=1}^{k} \log(C_i^{-1}X) = 0. \tag{2}$$

This definition is similar to the characterization of the geometric mean of real numbers and in the case $k = 2$ is the usual geometric mean definition (1). Therefore the geometric mean of the covariance matrices, $\{C_i\}_{i=1}^k \subset S_{++}^N$ is the solution of this equation. This can be approximated using the iteration

$$\mu_{t+1} = \exp_{\mu_t}\left(\frac{1}{k}\sum_{i=1}^{k}\log_{\mu_t}(C_i)\right), \tag{3}$$

where μ_0 is the initial guess and μ_{t+1} is the $(t+1)$ approximation of the geometric mean. This iteration solves, in the limit and under some conditions, the variational problem defining the geometric mean or FrÃ©chet mean (see [23, 10, 22]). Nevertheless, the computations required to implement (3) in each iterations involve the computation of the matrix function $\log_{\mu_t}$ and $\exp_{\mu_t}$. The computation of these matrix functions, even for symmetric matrices, requires a computational burden that has to be considered, and an efficient and stable algorithm has to be implemented. See [12] and references therein.

The main contribution of this work is the introduction of two modifications of the usual computational implementation of (3). The first modification is obtaining by using a generalized eigenvalues problem (instead of a classical eigenvalue problem) to compute the $\log_{\mu_t}$ and $\exp_{\mu_t}$ maps. The use of generalized eigenvalue problems will help us to reduce the total number of matrix diagonalization procedures. In the second modification, a Cholesky decomposition is applied to the generalized eigenvalue problems, thereby enabling more stable computations during the implementation of the generalized eigenvalue

decomposition. To stress the differences in our implementations we first review the algorithm in [23] and other existing approaches and then present the novel methodologies. For the mathematical analysis of the method in [23] we refer to [6, 27]. We mention that the connection between Cholesky decomposition and geometric mean of symmetric positive matrices has been used before. See for instance [17] where they use the Cholesky decomposition to introduce a new parametrization and metric on the set of positive definite matrices. We also mention [6] where it is also proposed as well as an iteration that takes advantage of the Cholesky decomposition. In the present work we further explore the usage of Cholesky factorization to write the iteration and also to compute the required eigendecompositions in order to compute functions of matrices, see Theorems 1 and 2 in Section 3.

A first evaluation of the proposed compact method shows competitive advantages regarding existing previous algorithms concerning the good action classification due to the compactness and the low dimension of the descriptor. In Section 4 the proposed and existing algorithms are evaluated in terms of computing time and overall performance within a classification process. Additional contributions can be summarized as follows:

- An comprehensive analysis of geometric means [23, 3, 10, 1] is carried out to propose different computations variations.
- We design a covariance video representation to obtain a very compact descriptor that serves as input to a machine learning classifier.
- An exhaustive analysis of proposed geometric mean approaches by using simulated covariance matrices as well as with covariance matrices computed from some video datasets.
- A compact descriptor that can recognize activities in video sequences using the mean of frame-covariances matrices.

We review important aspects of Riemannian manifolds in Section 2, in particular, we review the geometry of the symmetric positive-definite matrices. We also re-visit existing algorithms to compute the geometric mean and some related variants. In Section 3 we present the proposed algorithms, one using generalized eigenvalues and another one Cholesky decomposition. In Section 4 we apply the methods in two computational experiments, one with simulated matrices to compare performance in terms of computation time and another one using a particular action recognition dataset to compare also the time and performance in action classification. Finally in Section 5 we include some concluding remarks.

2. Riemannian geometry and geometric mean approximation

The space of symmetric positive-definite matrices S_{++}^N is a Riemannian manifold, that is, a smooth manifold with a Riemannian metric. Pennec in [23] introduces the *affine-invariant* Riemannian metric for S_{++}^N. That is, for all A, B in the tangent space at P ($T_P S_{++}^N$), the inner product defined at P is given by

$$\langle A, B\rangle_P = \langle P^{-\frac{1}{2}} A P^{-\frac{1}{2}}, P^{-\frac{1}{2}} B P^{-\frac{1}{2}}\rangle.$$

Here $\langle C, D\rangle = \mathrm{tr}(C^T D)$ is the usual Frobenious matrix inner product. Note that the *affine-invariant* metric induces the norm $\|A\|_P = \left\|P^{-\frac{1}{2}} A P^{-\frac{1}{2}}\right\|$, where also $\|\cdot\|$ is the usual Frobenius matrix norm. With the *affine-invariant* Riemannian metric, S_{++}^N is a geodesically complete Riemannian manifold with non-positive curvature (see [18, 14]). The manifold S_{++}^N is also simply connected and therefore a Cartan-Hadamard manifold (See chapter XI Theorem 1.2 [14]). This implies that the exponential map $\exp_P : T_P S_{++}^N \to S_{++}^N$ is a diffeomorphism for all $P \in S_{++}^N$. The inverse of the exponential map is the logarithm $\log_P : S_{++}^N \to T_P S_{++}^N$. In this way, for each $Q, P \in S_{++}^N$, $\log_P(Q) = V$ where $V \in T_P S_{++}^N$ and $\exp_P(V) = Q$. Pennec in [23], Moakher in [20] and Fletcher in [10] introduce different ways to obtain the geodesics in S_{++}^N with the *affine-invariant* Riemannian metric. Given $\Sigma \in S_{++}^N$ and the tangent vector $V \in T_\Sigma S_{++}^N$ we have that the geodesic γ_V with direction vector V and passing through $\gamma_V(0) = \Sigma$ is given by $\gamma_V(t) = \Sigma^{\frac{1}{2}} \exp\left(t\Sigma^{-\frac{1}{2}} V \Sigma^{-\frac{1}{2}}\right) \Sigma^{\frac{1}{2}}$. Therefore, the exponential map based at Σ is given by

$$\exp_\Sigma(V) = \Sigma^{\frac{1}{2}} \exp\left(\Sigma^{-\frac{1}{2}} V \Sigma^{-\frac{1}{2}}\right) \Sigma^{\frac{1}{2}}. \tag{4}$$

For its inverse map, taking $V \in S_{++}^N$ the log map based at Σ is

$$\log_\Sigma(V) = \Sigma^{\frac{1}{2}} \log\left(\Sigma^{-\frac{1}{2}} V \Sigma^{-\frac{1}{2}}\right) \Sigma^{\frac{1}{2}}. \tag{5}$$

These functions map matrices between the manifold S_{++}^N and the tangent space $T_\Sigma S_{++}^N$, that corresponds to the space of symmetric matrices that we denote as S^N. In $\exp_\Sigma$ and $\log_\Sigma$, the exp and log are the corresponding functions of matrices that extend the real exponential and logarithmic functions, see [12]. Another important consequence for Cartan-Hadamard manifolds is that given two points P, Q in the manifold there exists a unique geodesic that joins P and Q and its length is the geodesic distance between P and Q, see [18, 14]. Then, in the case of the Cartan-Hadamard manifold S_{++}^N with the affine-invariant metric, the distance is given by,

$$\mathrm{dist}(P, Q) = \left\|\log_P(Q)\right\|_P. \tag{6}$$

Note that here $\log_P(Q) \in T_P S^N_{++}$ and it is such that $\exp_P(\log_P(Q)) = Q$. Then we can write,

$$\begin{aligned} \text{dist}(P,Q) &= ||\log_P(Q)||_P \\ &= ||P^{\frac{1}{2}} \log\left(P^{-\frac{1}{2}} Q P^{-\frac{1}{2}}\right) P^{\frac{1}{2}}||_P \\ &= ||\log\left(P^{-\frac{1}{2}} Q P^{-\frac{1}{2}}\right)|| \\ &= \sqrt{\text{tr}\left(\log\left(P^{-\frac{1}{2}} Q P^{-\frac{1}{2}}\right)^2\right)}. \end{aligned} \tag{7}$$

Now, with this distance, similar to the variational characterization of the mean in real numbers, the geometric mean of $\{C_i\}_{i=1}^k \subset S^N_{++}$ is defined as the solution of the variational formulation

$$\arg\min_{X \in S^N_{++}} \sum_{i=1}^{k} \text{dist}(X, C_i)^2 = \arg\min_{X \in S^N_{++}} \rho(X), \tag{8}$$

where $\rho(X) = \frac{1}{2k} \sum_{i=1}^{k} \text{dist}(X, C_i)^2$. Moakher in [19] dealing with the same distance, establishes an equivalent formulation by setting the non linear matrix formulation (2). Moakher also proves the unique solution of the equation. The unique solution have been referred as "Riemannian geometric mean" [19], "Fréchet mean" [23] or "geometric mean" [6]. Taking $\{C_i\}_{i=1}^k \subset S^N_{++}$, Pennec in [23] proposes a Newton gradient descent algorithm to calculate this geometric mean. We can see that

$$\nabla \rho(X) = -\frac{1}{k} \sum_{i=1}^{k} \log_X(C_i), \tag{9}$$

and then the gradient descent algorithm obtained in [23] is written as

$$\mu_{t+1} = \exp_{\mu_t}\left(\frac{1}{k} \sum_{i=1}^{k} \log_{\mu_t}(C_i)\right), \tag{10}$$

where μ_0 is the initial guess and μ_{t+1} is the $(t+1)$ approximation of the geometric mean. There exist different approximations to implement geometric mean. In next subsections we are going to review existing algorithms to compute the mean.

2.1. Gradient descent algorithm

Pennec in [23] proposed the algorithm showed in (10). In order to motivate this algorithm we first write,

$$\begin{aligned}\mu_{t+1} &= \exp_{\mu_t}\left(\frac{1}{k}\sum_{i=1}^{k}\log_{\mu_t}(C_i)\right)\\ &= {\mu_t}^{\frac{1}{2}}\exp\left(\frac{1}{k}\sum_{i=1}^{k}\log({\mu_t}^{-\frac{1}{2}}C_i{\mu_t}^{-\frac{1}{2}})\right){\mu_t}^{\frac{1}{2}}.\end{aligned} \tag{11}$$

We can then implement the computation of the last expression, as expressed in Algorithm 1.

Algorithm 1 Gradient descent algorithm proposed by Pennec in [23] to compute the geometric mean of the matrices $\{C_i\}_{i=1}^{k} \subset S_{++}^{N}$.

μ_0 {Any, e.g., $\mu_0 = \frac{1}{k}\sum_{i=1}^{k} C_i$}
δ {Maximum number of iterations}
$t = 0$
repeat
  $X_t = O_{n\times n}$ {Zero matrix}
  Q_{μ_t}, D_{μ_t} {Eigenvectors and eigenvalues of μ_t }
  ${\mu_t}^{\frac{1}{2}} = Q_{\mu_t} D_{\mu_t}^{\frac{1}{2}} Q_{\mu_t}^{T}$
  ${\mu_t}^{-\frac{1}{2}} = Q_{\mu_t} D_{\mu_t}^{-\frac{1}{2}} Q_{\mu_t}^{T}$
  for $X \in \{C_i\}_{i=1}^{k}$ **do**
    Q_i, D_i {Eigenvectors and eigenvalues of ${\mu_t}^{-\frac{1}{2}} X {\mu_t}^{-\frac{1}{2}}$ }
    $X_t = X_t + Q_i \log(D_i) Q_i^{T}$
  end for
  $X_t = \frac{1}{k} X_t$
  Q_{X_t}, D_{X_t} {Eigenvector and eigenvalues of X_t}
  ${\mu_{t+1}} = {\mu_t}^{\frac{1}{2}} Q_{X_t} \exp\left(D_{X_t}\right) {Q_{X_t}}^{T} {\mu_t}^{\frac{1}{2}}$
  $t = t + 1$
until $\|X_t\| > \epsilon$ and *number of iteration* $\leq \delta$

2.2. Gradient descent: step control

T. Fletcher in [10] based his algorithm in the one developed in (8) using the gradient (9) to control the step of the gradient descent. Fletcher compares the norm of the gradient between consecutive iterations and if it increases then the step is reduced in a half for the next iteration. See Algorithm 2.

Algorithm 2 Gradient descent with step control proposed by Fletcher in [10] to compute the geometric mean of $\{C_i\}_{i=1}^k \subset S_{++}^N$.

μ_0 {Any, e.g., $\mu_0 = \frac{1}{k}\sum_{i=1}^k C_i$}
δ {Maximum number of iterations}
$\tau = 1$ {Initial step }
$t = 0$
repeat
  $X_t = O_{n\times n}$ {Zero matrix}
  Q_{μ_t}, D_{μ_t} {Eigenvectors and eigenvalues of μ_t }
  ${\mu_t}^{\frac{1}{2}} = Q_{\mu_t} D_{\mu_t}^{\frac{1}{2}} Q_{\mu_t}^T$
  ${\mu_t}^{-\frac{1}{2}} = Q_{\mu_t} D_{\mu_t}^{-\frac{1}{2}} Q_{\mu_t}^T$
  for $X \in \{C_i\}_{i=1}^k$ **do**
    Q_i, D_i {Eigenvectors and eigenvalues of ${\mu_t}^{-\frac{1}{2}} X {\mu_t}^{-\frac{1}{2}}$ }
    $X_t = X_t + Q_i \log(D_i) Q_i^T$
  end for
  $X_t = \frac{1}{k} X_t$
  $G_t = {\mu_t}^{\frac{1}{2}} X_t {\mu_t}^{\frac{1}{2}}$ {Gradient}
  Q_{X_t}, D_{X_t} {Eigenvector and eigenvalues of τX_t}
  $\mu_{t+1} = {\mu_t}^{\frac{1}{2}} Q_{X_t} \exp\left(D_{X_t}\right) {Q_{X_t}}^T {\mu_t}^{\frac{1}{2}}$
  if $1 \leq t$ **and** $\|G_t\| > \|G_{t-1}\|$ **then**
    $\tau = \frac{\tau}{2}$, $G_t = G_{t-1}$
  end if
  $t = t + 1$
until $\|X_t\| > \epsilon$ and *number of iteration* $\leq \delta$

2.3. Log-Euclidean metric

Arsigny *et al.* [1] introduce a new family of metrics called *"Log-Euclidean"* that overcomes computational limitations of the affine-invariant metric and has been used as a fast compute alternative for different image applications [16]. This metrics comes from defining a Euclidean structure on S_{++}^N via its Lie algebra or tangent space S^N. With the norm associated to its metric the distance between $A, B \in S_{++}^N$ is

$$\operatorname{dist}_{le}(A,B) = \|\log(A) - \log(B)\|_{Id}\,. \tag{12}$$

With this structure the arithmetic operations between symmetric positive definite matrices are simple and faster to compute than corresponding operations with the affine-invariant metric proposed by Pennec in [23]. According to Theorem 3.13 of [1] the Fréchet mean (8) of $\{C_i\}_{i=1}^k$ using the distance defined in (12) is

$$\begin{aligned}\arg\min_{X\in S_{++}^N} \frac{1}{k}\sum_{i=1}^{k} \operatorname{dist}_{le}(X,C_i)^2 &= \arg\min_{X\in S_{++}^N} \frac{1}{k}\sum_{i=1}^{k} \|\log(X) - \log(C_i)\|_{Id}{}^2 \\ &= \exp\left(\frac{1}{k}\sum_{i=1}^{k}\log(C_i)\right).\end{aligned} \tag{13}$$

In the last step we used that the first order condition of this optimization problem is given by $\frac{1}{k}\sum_{i=1}^{k}(\log(C) - \log(C_i)) = 0$. See the method in Algorithm 3.

Algorithm 3 *Log-Euclidean* mean proposed by Arsigny *et. al* in [1] to compute the mean of the matrices $\{C_i\}_{i=1}^k \subset S_{++}^N$.

$X_0 = O_{n\times n}$ {Zero matrix}
for $X \in \{C_i\}_{i=1}^k$ **do**
  Q_i, D_i {Eigenvectors and eigenvalues of X }
  $X_0 = X_0 + Q_i \log(D_i) Q_i^T$
end for
$X_0 = \frac{1}{k} X_0$
Q_{X_0}, D_{X_0} {Eigenvector and eigenvalues of X_0}
$\mu = Q_{X_0} \exp(D_{X_0}) {Q_{X_0}}^T$

2.4. Barachant algorithm

Alexandre Barachant in [4] compute the geometric mean using τ to control the step of the gradient descent and as a stopping criteria. The step decreases progressively; however, as the magnitude of the gradient increases, the step size decreases at a more rapid pace, see [2, 3]. We present the method in Algorithm 4.

Algorithm 4 Method proposed by Alexandre Barachant in an applied Riemannian library *"PyRiemann"* [4] to compute the geometric mean of the matrices $\{C_i\}_{i=1}^k \subset S_{++}^N$.

μ_0 {Any, e.g., $\mu_0 = \frac{1}{k}\sum_{i=1}^k C_i$}
δ {Maximum number of iterations}
$\tau = 1$
$\rho >> 0$
$t = 0$
repeat
  $X_t = O_{n\times n}$ {Zero matrix}
  Q_{μ_t}, D_{μ_t} {Eigenvectors and eigenvalues of μ_t }
  ${\mu_t}^{\frac{1}{2}} = Q\mu_t D{\mu_t}^{\frac{1}{2}} Q{\mu_t}^T$
  ${\mu_t}^{-\frac{1}{2}} = Q\mu_t D{\mu_t}^{-\frac{1}{2}} Q{\mu_t}^T$
  for $X \in \{C_i\}_{i=1}^k$ **do**
    Q_i, D_i {Eigenvectors and eigenvalues of ${\mu_t}^{-\frac{1}{2}} X {\mu_t}^{-\frac{1}{2}}$ }
    $X_t = X_t + Q_i \log(D_i) Q_i^T$
  end for
  $X_t = \frac{1}{k} X_t$
  Q_{X_t}, D_{X_t} {Eigenvector and eigenvalues of τX_t}
  $h = \tau \|X_t\|$
  $\mu_{t+1} = {\mu_t}^{\frac{1}{2}} Q_{X_t} \exp\left(D_{X_t}\right) {Q_{X_t}}^T {\mu_t}^{\frac{1}{2}}$
  if $h < \rho$ **then**
    $\tau = 0.95\tau$
    $\rho = h$
  else
    $\tau = 0.5\tau$
  end if
  $t = t + 1$
until $\|X_t\| > \epsilon$ and $\tau > \epsilon$ and *number of iteration* $\leq \delta$

3. Proposed geometric mean

The previous algorithms made their modifications based on the gradient descent algorithm procedure. This work conducts an exhaustive algebraic analysis of the algorithm and shows various algebraic properties to applicable to the calculation of the mean. From this analysis we managed to make less and stable operations. The first proposed modification is using generalized eigenvalues and the second one is using Cholesky decomposition.

3.1. Generalized eigenvalues approach

We use some results from [12] where computation of functions of matrices is studied. Let f be a real function defined in the spectrum of A, so we can

consider $f(A)$. See [12] for the equivalent definitions of $f(A)$. If $A = ZDZ^{-1}$, where Z is a non-singular (square) matrix, we can compute $f(A)$

$$f(A) = Zf(D)Z^{-1} = Z\text{diag}(f(\lambda_i))Z^{-1}. \tag{14}$$

In particular, when A is symmetric, its eigenvalues are real and A is diagonalizable. More precisely, we can factor A as $A = QDQ^T$ where $D = \text{diag}(\lambda_i)$ is a diagonal matrix with the eigenvalues of A and the columns of the orthogonal matrix Q are the corresponding eigenvectors. Additionally for $A \in S^N_{++}$, the eigenvalues are positive real numbers. We have the following result.

Theorem 3.1 (Corollary 1.34 of [12])**.** *Let $A, B \in \mathbb{C}^{n\times n}$ and let f be defined in the spectrum of AB and BA. Then*

$$Af(BA) = f(AB)A.$$

We apply this result, and several others recalled below, to the definition of $\exp_\Sigma$ and $\log_\Sigma$ in (4) and (5), respectively. We present these results for general functions f and it can be used in particular for the exponential and logarithm Riemannian maps. In general, given $\Sigma \in S^N_{++}$, we define the matrix function $f_\Sigma : S^N \to S^N$ (or $f_\Sigma : S^N_{++} \to S^N$), named matrix function f based in Σ, by

$$f_\Sigma(V) = \Sigma^{\frac{1}{2}} f(\Sigma^{-\frac{1}{2}} V \Sigma^{-\frac{1}{2}})\Sigma^{\frac{1}{2}},$$

where $V \in S^N$ ($V \in S^N_{++}$ resp.).

The efficient computation of f_Σ is not straight forward. Using Theorem 1 we can find several equivalent ways to write $f_\Sigma(V)$ for a symmetric matrix V.

Corollary 3.2. *Assume that $\Sigma \in S^N_{++}$ and $V \in S^N$. We have*

$$f_\Sigma(V) = \Sigma f(\Sigma^{-1}V) = f(V\Sigma^{-1})\Sigma. \tag{15}$$

Proof. Taking $B = \Sigma^{-\frac{1}{2}}V$ and $A = \Sigma^{-\frac{1}{2}}$ in Theorem 3.1, we get

$$\Sigma^{-\frac{1}{2}} f(\Sigma^{-\frac{1}{2}} V \Sigma^{-\frac{1}{2}}) = f(\Sigma^{-\frac{1}{2}}\Sigma^{-\frac{1}{2}}V)\Sigma^{-\frac{1}{2}}$$

and therefore

$$\begin{aligned} f_\Sigma(V) &= \Sigma^{\frac{1}{2}} f(\Sigma^{-\frac{1}{2}} C \Sigma^{-\frac{1}{2}})\Sigma^{\frac{1}{2}} \\ &= \Sigma\left[f(\Sigma^{-\frac{1}{2}}\Sigma^{-\frac{1}{2}}V)\Sigma^{-\frac{1}{2}}\right]\Sigma^{\frac{1}{2}} = \Sigma f(\Sigma^{-1}V). \end{aligned}$$

Analogously we obtain the second equality of (15) by taking $A = \Sigma^{\frac{1}{2}}$ and $B = \Sigma^{-\frac{1}{2}} V \Sigma^{-1}$. ☑

Given $\Sigma \in S^N_{++}$ and $V \in S^N$, we can consider the generalized eigenvalue problem $V\phi = \lambda\Sigma\phi$ that in matrix form is written as

$$\Sigma^{-1}V = QDQ^{-1}, \tag{16}$$

where D is the diagonal matrix of generalized eigenvectors, that is, $D = \text{diag}(\lambda_i)_{i=1}^n$. It is known that the eigenvectors are orthogonal in the inner product $\langle u, v\rangle_\Sigma = v^T\Sigma u$ generated by Σ, that is, $Q^T\Sigma Q = I_d$. Note that we have

$$\Sigma^{-1}V = QDQ^T\Sigma, \tag{17}$$

or

$$\Sigma^{-\frac{1}{2}}V\Sigma^{-\frac{1}{2}} = \Sigma^{\frac{1}{2}}QDQ^T\Sigma^{\frac{1}{2}}. \tag{18}$$

Taking $\Psi = \Sigma^{\frac{1}{2}}Q$ we have

$$\Sigma^{-\frac{1}{2}}V\Sigma^{-\frac{1}{2}} = \Psi D\Psi^{-1}. \tag{19}$$

From (16) and (19) note that the eigenvalues of $\Sigma^{-1}V$ and $\Sigma^{-\frac{1}{2}}V\Sigma^{-\frac{1}{2}}$ are the same. Applying this on (19) we have

$$f\left(\Sigma^{-\frac{1}{2}}V\Sigma^{-\frac{1}{2}}\right) = \Psi f(D)\Psi^{-1}, \tag{20}$$

and therefore,

$$f_\Sigma(V) = \Sigma^{\frac{1}{2}} f\left(\Sigma^{-\frac{1}{2}}V\Sigma^{-\frac{1}{2}}\right)\Sigma^{\frac{1}{2}} = \Sigma Q f(D) Q^T\Sigma. \tag{21}$$

This shows that $f_\Sigma(V)$ can be computed by using the generalized eigenvalue problem $\Sigma^{-1}V$.

Theorem 3.3. *The generalized eigenvalues of $f_\Sigma(V)$ with respect to Σ are of the form $f(\lambda_i)$ where λ_i is a generalized eigenvalue of V with respect to Σ.*

Proof. The eigenvalues of $f_\Sigma(V)$ with respect Σ are the eigenvalues of $\Sigma^{-1}f_\Sigma(V)$, but using (15) of Corollary 3.2 we get

$$\Sigma^{-1}f_\Sigma(V) = f(\Sigma^{-1}V) = f(QDQ^{-1}) = Qf(D)Q^{-1}.$$

Then, the eigenvalues of $\Sigma^{-1}f_\Sigma(V)$ are of the form form $f(D)$ where D is a diagonal matrix with the eigenvalues of V with respect Σ. ☑

We can apply these results to rewrite (10). Let $\mu, C \in S^N_{++}$. Note that if we put $f(r) = \log(r)$ we have

$$\log_\mu(C) = f_\mu(C) = \mu Q \log(D) Q^T\mu$$

where Q and D are given by $\mu^{-1}C = QDQ^T\mu$ or $\mu^{-\frac{1}{2}}C\mu^{-\frac{1}{2}} = \mu^{\frac{1}{2}}QDQ^T\mu^{\frac{1}{2}}$. Analogously

$$\exp_\mu(C) = \mu Q \exp(D) Q^T \mu.$$

We can rewrite (10) in the form,

$$\begin{aligned}\mu_{t+1} &= \exp_{\mu_t}\left(\frac{1}{k}\sum_{i=1}^{k}\log_{\mu_t}(C_i)\right)\\ &= \exp_{\mu_t}\left(\frac{1}{k}\sum_{i=1}^{k}\mu_t Q_i \log(D_i) Q_i^T \mu_t\right)\\ &= \exp_{\mu_t}\left(\mu_t\left[\frac{1}{k}\sum_{i=1}^{k}Q_i\log(D_i)Q_i^T\right]\mu_t\right).\end{aligned} \tag{22}$$

where the matrix of generalized eigenvectors Q_i and the diagonal matrix of generalized eigenvalues D_i, are given by $\mu_t^{-1}C_i = Q_iD_iQ_i^T\mu_t$. Let us denote $J = \mu_t\left[\frac{1}{k}\sum_{i=1}^{k}Q_i\log(D_i)Q_i^T\right]\mu_t$, then $\mu_t = \exp_{\mu_t}(J)$. To compute $\exp_{\mu_t}(J)$ we need the eigenvectors Q and eigenvalues D of ${\mu_t}^{-1}J$ and in this way (22) is

$$\begin{aligned}\mu_{t+1} &= \exp_{\mu_t}(J)\\ &= \mu_t Q \exp(D)\, Q^T \mu_t.\end{aligned} \tag{23}$$

Where ${\mu_t}^{-1}J = QDQ^T\mu_t$. Using the generalized eigenvalue expression (22) we can save several matrix operations and obtain the same results as in Pennec's method in Algorithm 1. Below is the method implemented in the experiments.

Algorithm 5 With the calculations made in (22) and (23) we present an algorithm using generalized eigenvalues to calculate the mean of the matrices $\{C_i\}_{i=1}^k \subset S_{++}^N$.

μ_0 {Any, e.g., $\mu_0 = \frac{1}{k}\sum_{i=1}^k C_i$}
δ {Maximum number of iterations}
$t = 0$
repeat
  $X_t = O_{n\times n}$ {Zero matrix}
  for $X \in \{C_i\}_{i=1}^k$ **do**
    Q_i, D_i {Generalized eigenvector and eigenvalues of ${\mu_t}^{-1}X$ }
    $X_t = X_t + Q_i \log(D_i) Q_i^T$
  end for
  $X_t = \frac{1}{k} X_t$
  $J = \mu_t X_t \mu_t$
  Q, D {Eigenvector and eigenvalues of $X_t\mu_t$}
  $\mu_{i+1} = \mu_t Q \exp(D)\, Q^T \mu_t$
  $t = t + 1$
until $\|J\| > \epsilon$ and *number of iteration* $\leq \delta$

3.2. Cholesky decomposition

Now we use Cholesky factorization applied to the generalized eigenvalue decomposition. We come back to the general case of a real function f. For a positive-definite matrix Σ recall that $f_\Sigma(V) = \Sigma Q f(D) Q^T \Sigma$ where Q and D correspond to the generalized eigenvectors and eigenvalues of V with respect to Σ. We use Cholesky factorization $\Sigma = R^T R$, where R is an upper triangular matrix. We have $VQ = \Sigma QD$, that is $VR^{-1}RQ = R^T RQD$. Taking $\psi = RQ$ we have $R^{-T}VR^{-1}\psi = \psi D$. We see that ψ solves the eigenvalue problem associated to the matrix $W = R^{-T}VR^{-1}$ (see [12]). Note that the eigenvalues of this problem are the same of initial problem $\Sigma^{-1}V$. From all this we have

$$f_\Sigma(V) = \Sigma Q f(D) Q^T \Sigma = R^T R R^{-1} \psi f(D) \psi^T R^{-T} R^T R = R^T \psi f(D) \psi^T R.$$

Applying this to (10) we have

$$\mu_{t+1} = \exp_{\mu_t}\left(\frac{1}{k}\sum_{i=1}^k R^T \psi_i \log(D_i) \psi_i^T R\right), \tag{24}$$

where ψ_i and D_i are the eigenvectors and eigenvalues of the eigenvalue problem associated to $V_i = R^{-T} C_i R^{-1}$. Denoting $J = \frac{1}{k}\sum_{i=1}^k \psi_i \log(D) \psi_i^T$, now (24) is

$$\mu_{t+1} = \exp_{\mu_t}(R^T J R) = R^T \psi' \exp(D') {\psi'}^T R, \tag{25}$$

where ψ' and D' are the eigenvectors and eigenvalues of the eigenvalue problem associated to $V' = R^{-T}(R^T J R)R^{-1} = J$. Note that with this Cholesky factorization we change the k generalized eigenvalue problems in the `for` buckle for k (classical) eigenvalue problems. We present the method in Algorithm 6.

Algorithm 6 With the calculations made in (24) and (25) we present an algorithm using the Cholesky decomposition to compute the mean of the matrices $\{C_i\}_{i=1}^k \subset S_{++}^N$.

μ_0 {Any, e.g., $\mu_0 = \frac{1}{k}\sum_{i=1}^k C_i$}
δ {Maximum number of iterations}
$t = 0$
repeat
  $X_t = O_{n\times n}$ {Zero matrix}
  R {Upper triangular Cholesky of μ_t}
  R^T {Transpose of R}
  R^{-1} {Solution of $XR = Id$}
  R^{-T} {Transpose of R^{-1}}
  for $X \in \{C_i\}_{i=1}^k$ **do**
    $X = R^{-T} X R^{-1}$
    Q_i, D_i {Eigenvalues and eigenvectors of X}
    $X_t = X_t + Q_i \log(D_i) Q_i^T$
  end for
  $X_t = \frac{1}{k} X_t$
  $J = R^T X_t R$
  Q, D {Eigenvalues and eigenvectors of X_t}
  $\mu_{t+1} = R^T Q \exp(D) Q^T R$
  $t = t + 1$
until $\|J\| > \epsilon$ and *number of iteration* $\leq \delta$

4. Evaluation and results

The proposed modifications of the classical algorithm was evaluated in terms of computation, efficiency and its respective performance as video descriptor into a task of computer action recognition. For that, we design two experiments: using simulated matrices and using human action videos dataset.

Experiment 1: Simulated matrices. Using simulated matrices we measure the time to compute the mean by each algorithm varying the size and the number of matrices, we also measure the number of iterations that each algorithm needs to compute the mean. We use a set of random symmetric positive definite matrices with different sizes to evaluate the proposed approach [21].

Experiment 2: Action recognition. In general, we consider a video D as sequence of K frames $\{I_t\}_{t=1}^K \subset \mathbb{R}^{W\times H}$, where each frame can be identified

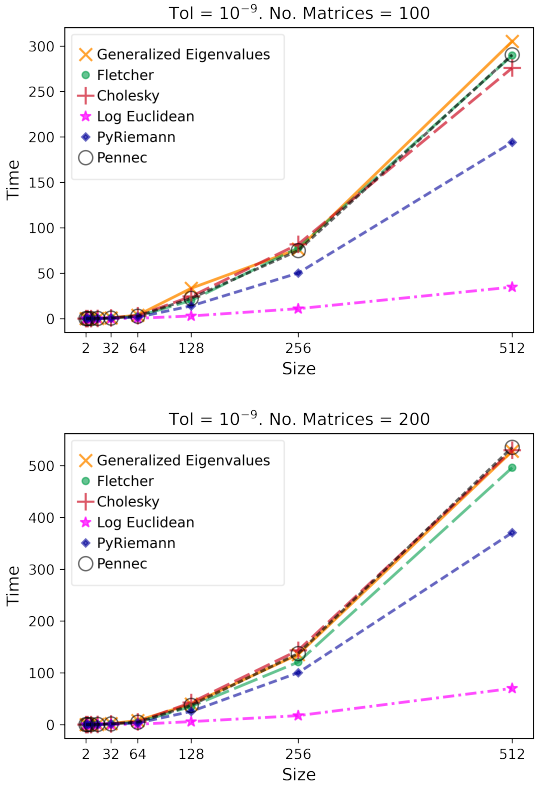

FIGURE 1. Comparison between the time that each algorithm need to calculate the mean changing the size of the matrices (from 2×2 to 512×512) with a tolerance of 10^{-9}.

by (or associated to) a set of N features $\left\{F^{(1)},\ldots,F^{(N)}\right\} \subset \mathbb{R}^{W\times H}$. For a compact description we calculate the mean covariance matrix μ_D of the set of frame covariance matrices $\{C_t\}_{t=1}^{K} \subset S_{++}^{N}$ where $C_t = [C_{ij}]_{i=1,j=1}^{N,N}$ and C_{ij} is the covariance between $F^{(i)}$ and $F^{(j)}$. Therefore each video is described with a mean covariance matrix. In this work, we utilized the optical flow method described in [7] to calculate features for each frame. This technique computes a vector field that represents the motion in each frame, and we termed these kinematic features as:

$$V_x, V_y, \|V\|, T_x, T_y, A_T, A_N, \|a\|, N_x, N_y, a_x, a_y, \|T'\|, \|N\|$$

With this features we represent each video with a covariance matrix of dimension 14×14. That can be rearranged in a vector of dimension $\frac{14\times 15}{2}$ to be input in an action classifier method. In this work we use the public academic action recognition dataset **UT-Interaction** [24]. This dataset have six different actions classes: "Hand shaking", "Hugging", "Kicking", "Pointing", "Punching" and "Pushing". It has two sets of videos, one recorded with a plane background and a static camera (Set 1) and another with some camera motions (Set 2). Each set has 60 videos so there are 10 videos of each class.

4.1. Evaluation over simulated matrices

We know that the size of the covariance matrices depends on the number of features so in Figure 1 we show two experiments, one with 100 matrices and other with 200, with a tolerance of 10^{-9}. The graphs present the time that each algorithm needs to calculate the mean while the size of the matrices changes from 2×2 to 512×512. In this case the generalized eigenvalues approach is a little bit slower because the computation cost that the generalized eigenvalues problems carry.

To evaluate the algorithms with regard the length of video sequences we show three experiments in Figure 2 with matrices of dimension 32×32, 128×128, and 512×512, and tolerance of 10^{-9}. The graphs present the time that each algorithm needs to calculate the mean while the number of matrices changes from 50 to 200. In this case the generalized eigenvalues approach is a little bit slower but improve w.r.t Cholesky approach while the size of the matrices increases, this is due to the stability of the computations involved in the generalized eigenvalues approach.

In Figure 3, for matrices of dimension 32×32, 64×64 and 128×128 we present the number of iterations that each algorithm needs to calculate the mean while the number of matrices change from 25 to 200 with a tolerance of 10^{-9}. It should be noted that the generalized eigenvalues approach has a considerable better performance on the number of iterations needed to compute the mean. In all simulations, the Log-Euclidean algorithm is faster and the one that requires fewer iterations.

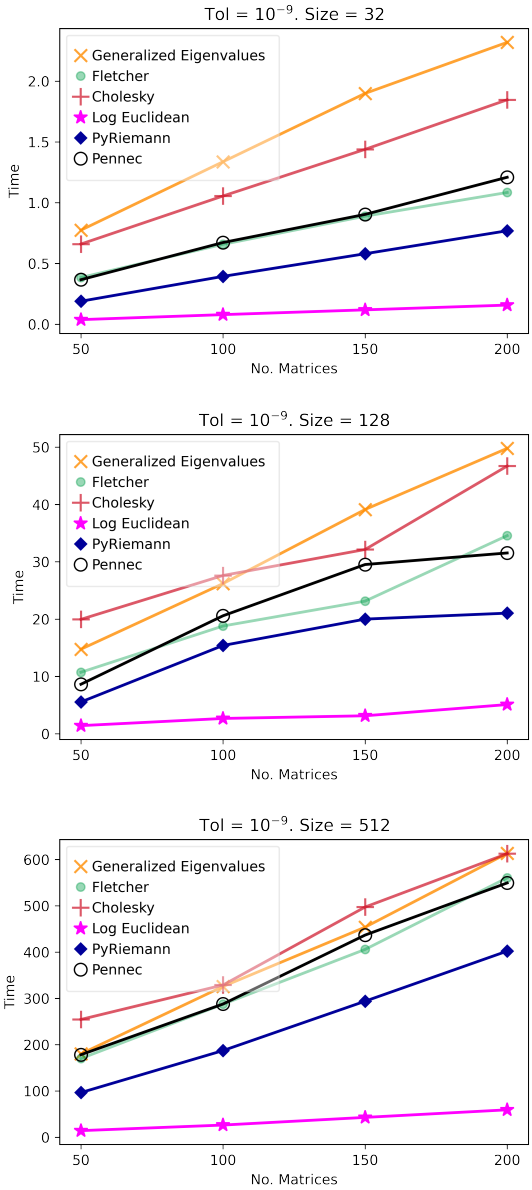

FIGURE 2. Comparison between the time that each algorithm needs to calculate the mean changing the number matrices (from 50 to 200) with a tolerance of 10^{-9}. In each subfigure varies the size of the matrices from 32×32 to 512×512.

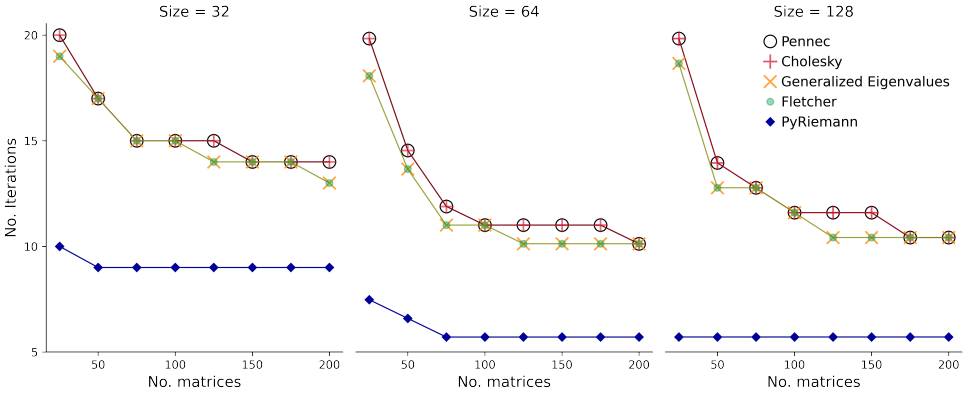

FIGURE 3. Comparison between the number of iterations that each algorithm needs to calculate the mean changing the size of the matrices (32×32, 64×64 and 128×128), with tolerance of 10^{-9}.

In Figure 3 we can see that when the number of matrices increases the number of iterations needed for the algorithms is the same or decreases. The *"PyRiemann"* algorithm is the one that needs fewer iterations to converge after the Log-Euclidean algorithm. Both new algorithms have very close performance being Cholesky factorization faster but generalized eigenvalue take relative less iterations. It is possible that this is due to the well known good stability properties of the Cholesky factorization.

4.2. Action recognition: evaluation as video descriptor

In this part, we use the videos of the dataset *UT-Interaction* and the kinematic features to represent each frame. Once we describe a video with the covariance mean calculated with a specific algorithm, we can measure with the accuracy how successful it is to identify the right action class. The accuracy of the algorithms on the Set 1 and Set 2 is presented in Table **??**.

Accuracies of the algorithms in the video sequences of the Set 1 and Set 2 of the Dataset *UT-Iteraction*.

	Log Euclidean	Pennec	Fletcher	PyRiemann	**Generalized eigenvalues**	**Cholesky**
Accuracy Set 1	68.33 %	75 %	73.33 %	73.33 %	75 %	75 %
Accuracy Set 2	49.99%	60%	54.99%	54.99%	60 %	60%

Since there are algorithms whose accuracy is very similar we calculated other classification measure, the recall, the ratio of well-predicted action videos of a specific action class. Note that recall is a measure for each class so we

can take the mean to measure our classifier. The recall of each algorithm is presented in Table **??**

Average recall of the algorithms in the video sequences of the Set 1 and Set 2 of the Dataset *UT-Iteraction*.

	Log Euclidean	Pennec	Fletcher	PyRiemann	**Generalized eigenvalues**	**Cholesky**
Recall Set 1	68.33 %	63.33 %	66.66 %	69.99 %	73.33 %	68.33
Recall Set 2	61.66 %	63.33 %	64.99%	68.33%	65 %	63.33 %

To see how compact is the covariance mean as descriptor in Table **??** we compare the proposed algorithms (that have the same accuracy for both Sets) and other descriptors of the state of the art.

Average accuracies in Set 1 and 2 of UT-Interaction by different descriptors in the state of the art and proposed algorithms that have the same accuracy: Generalized eigenvalues and Cholesky. Some approaches depend on *np*, a number proportional to the dimension of the frames and is in order of thousands.

Approaches	Descriptor Size	Set 1	Set 2
Propagative Houg Voting[25]	162×np	93.3 %	91.7%
Daysy 3D [8]	192×np	71.67 %	56.67%
Slimani [9]	22500	40 %	66%
Laptev [15]	41800×np	68 %	65%
Proposed algorithms	105	**75%**	**60%**

To compare how the algebraic modifications proposed in our algorithms affect the time in Table **??** we measure the average time of computing the mean by the different algorithms.

Average time needed by the different algorithms to calculate the mean for video from the set 1 and set 2 of the Dataset *UT-Iteraction*.

	Log Euclidean	Pennec	Fletcher	PyRiemann	**Generalized eigenvalues**	**Cholesky**
Set 1	0.012 s	0.523 s	0.481 s	0.173 s	0.788 s	0.531 s
Set 2	0.012 s	0.568 s	0.499 s	0.199 s	0.691 s	0.562 s

The *log-Euclidean* metric was adopted as a baseline alternative because of its faster computational time performance. However, from the accuracy performance of Table **??**, it should be noted that this metric approximate the covariance average more than a geometric mean representation. For both sets, the proposed algorithms and the Pennec's algorithm achieved the best accuracy: 75% for the set 1 and 60% for the set 2. Calculating the recall we see that

the generalized eigenvalues approach has the best performance in the set 1 and in the set 2 the *PyRiemann* algorithm has the best performance. In real scenarios, the numerical composition of features in online action recognition may vary, and the treatment of covariance may result in numerical inaccuracies. The use of generalized eigenvalues and Cholesky decomposition in the proposed algorithms generates more stability in the calculation of the mean and therefore a lower effects of numerical errors. Therefore, this fact can cause the algorithms to perform well in classification tasks, since it is closer to the only geometric mean of the set of covariances.

5. Discussion and Conclusions

Using the study of the mean in the manifold of symmetric positive definite matrices we reviewed and evaluated different algorithms like the one proposed by Pennec [23], Fletcher *et. al* [10] and Alexandre Barachant *et. al* [4]. From there we proposed two algorithms. The proposal starts with the study of the calculus of the mean of covariance matrices in this manifold understanding the mathematical and geometrical background of the problem. The next goal was to propose a method to do fewer calculations and compare it with others in an application problem as a compact descriptor for action recognition.

In this work the use of the mean of covariance matrices which is the result of an optimization problem to find the matrix that is closer to all frame covariance matrices that lie in a Riemannian manifold give us a compact descriptor of only 105 scalar values to describe all the video. Regarding some other high dimensional descriptors, the covariance mean get the same or better performance in terms of accuracy. Once the optimization problem and mathematical foundations of the geometry of the manifold were understood, we introduced two algorithms: the generalized eigenvalues algorithm to do fewer calculations and then the Cholesky decomposition algorithm to do less and simple calculations.

The studied and proposed algorithms were first evaluated with simulated matrices and then with a set of public videos. In these implementations, the proposed algorithms take the longest time but present the best accuracy in both sets: 75% for set 1 and 60% for set 2, also especially the generalized eigenvalues approach achieved the best recall only in Set 1 this because camera movement in Set 2 resulted in more complex and sparse matrices. This fact is associated to complexity of representation on input features that form frame covariance matrices. These results show the stability of calculations involved in both proposed algorithms. This can affect the computation time but reach a numerically better approach to the geometric mean, which is reflected in a better performance. According to our study the generalized eigenvalues bring stability and allows a better representation of covariance mean. Nonetheless, a detailed complexity analysis of the proposed work should be directed on future works to establish mathematical properties that clarify the advantages of the approach we proposed.

Along with the proposed algorithms, we presented other algorithms to compute the mean. These algorithms are known in the state of the art and computational implementations are available to test in other application domains. Furthermore, these and the proposed algorithms can be used in online classification tasks due to their simple calculations. In [13, 26, 28] and the references therein, several methods and algorithms are presented to compute different means. These approaches, however, can introduce expensive computational procedures that limit their usability on specific scenarios. In this work, as initial exploration of the effects of introducing Cholesky factorization and generalized eigenvalue problems, we compared our method with selected implementations since our main focus is the practical application to action recognition. A more throughout comparison can still be done in future work. Our current work can be continued in several directions, for instance, other statistics and measures related to Riemannian manifold structure and covariance matrices can be incorporated into the classification procedure. Also, it would be interesting to inquire into the geometry and algebra of the manifold and to explore how the sparsity of the data can provide additional information and description to propose novel methods or improve existing ones.

References

[1] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, *Geometric Means in a Novel Vector Space Structure on Sysmetric Positive-Definite Matrices*, SIAM J. Matrix Analysis Applications **29** (2006), 328–347.

[2] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, *Multiclass Brain-Computer Interface Classification by Riemannian Geometry*, IEEE Transactions on Biomedical Engineering **59** (2012), no. 4, 920–928.

[3] ———, *Classification of Covariance Matrices Using a Riemannian-Based Kernel for BCI Applications*, Neurocomput. **112** (2013), 172?178.

[4] A. Barachant and J-R. King, *pyriemann v0.2.2*, June 2015.

[5] R. Bhatia and J. Holbrook, *Riemannian geometry and matrix geometric means*, Linear Algebra and its Applications **413** (2006), no. 2, 594–618, Special Issue on the 11th Conference of the International Linear Algebra Society, Coimbra, 2004.

[6] D. A. Bini and B. Iannazzo, *Computing the Karcher mean of symmetric positive definite matrices*, Linear Algebra and its Applications **438** (2013), no. 4, 1700–1710.

[7] T. Brox and J. Malik, *Large displacement optical flow: Descriptor matching in variational motion estimation*, IEEE transactions on pattern analysis and machine intelligence **33** (2011), 500–13.

[8] X. Cao, H. Zhang, C. Deng, Q. Liu, and H. Liu, *Action recognition using 3D DAISY descriptor*, Machine Vision and Applications **25** (2014), 159–171.

[9] K. N. e. H. Slimani, Y. Benezeth, and F. Souami, *Human interaction recognition based on the co-occurrence of visual words*, 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2014, pp. 461–466.

[10] P. T. Fletcher and S. Joshi, *Riemannian Geometry for the statistical analysis of diffusion tensor Data*, Signal Processing **87** (2007), 250–262.

[11] K. Guo, P. Ishwar, and J. Konrad, *Action Recognition From Video Using Feature Covariance Matrices*, IEEE Transactions on Image Processing **22** (2013), no. 6, 2479–2494.

[12] N. J. Higham, *Functions of Matrices: Theory and Computation (Other Titles in Applied Mathematics)*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008.

[13] B. Iannazzo and M. Porcelli, *The Riemannian Barzilai?Borwein method with nonmonotone line search and the matrix geometric mean computation*, IMA Journal of Numerical Analysis **38** (2018), no. 1, 495–517 (en).

[14] S. Lang, *Fundamentals of Differential Geometry*, vol. 191, Springer, 1999.

[15] I. Laptev, *On space-time interest points*, International journal of computer vision **64** (2005), 107–123.

[16] P. Li and Q. Wang, *Local log-euclidean covariance matrix (l2ecm) for image representation and its applications*, Computer Vision – ECCV 2012 (Berlin, Heidelberg) (Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, eds.), Springer Berlin Heidelberg, 2012, pp. 469–482.

[17] Z. Lin, *Riemannian geometry of symmetric positive definite matrices via cholesky decomposition*, SIAM Journal on Matrix Analysis and Applications **40** (2019), no. 4, 1353–1370.

[18] H. Q. Minh and V. Murino, *Covariances in Computer Vision and Machine Learning*, Morgan & Claypool Publishers, 2017.

[19] M. Moakher, *A Differential Geometric Approach to the Geometric Mean of Symmetric Positive-Definite Matrices*, SIAM Journal on Matrix Analysis and Applications **26** (2005), no. 3, 735–747, Publisher: Society for Industrial and Applied Mathematics.

[20] M. Moakher and M. Zerai, *The Riemannian Geometry of the Space of Positive-Definite Matrices and Its Application to the Regularization of Positive-Definite Matrix-Valued Data*, Journal of Mathematical Imaging and Vision **40** (2011), 171–187.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, B. Thirion V. Michel, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research **12** (2011), 2825–2830.

[22] X. Pennec, *Probabilities and Statistics on Riemannian Manifolds: Basic Tools for Geometric Measurements*, 01 1999, pp. 194–198.

[23] X. Pennec, Pierre Fillard, and Nicholas Ayache, *A riemannian framework for tensor computing*, International Journal of Computer Vision **66** (2006), no. 1, 41–66.

[24] M. S. Ryoo and J. K. Aggarwal, *UT-Interaction Dataset, ICPR contest on Semantic Description of Human Activities (SDHA)*, http://cvrc.ece.utexas.edu/SDHA2010/Human_Interaction.html, 2010.

[25] G. Yu, J. Yuan, and Z. Liu, *Propagative hough voting for human activity recognition*, Computer Vision – ECCV 2012 (Berlin, Heidelberg) (Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, eds.), Springer Berlin Heidelberg, 2012, pp. 693–706.

[26] X. Yuan, W. Huang, P. A. Absil, and K. A. Gallivan, *A Riemannian Limited-Memory BFGS Algorithm for Computing the Matrix Geometric Mean*, Procedia Computer Science **80** (2016), 2147–2157 (en).

[27] X. Yuan, W. Huang, P-A. Absil, and K. A. Gallivan, *Computing the matrix geometric mean: Riemannian versus Euclidean conditioning, implementation techniques, and a Riemannian BFGS method*, Numerical Linear Algebra with Applications **27** (2020), no. 5, e2321.

[28] T. Zhang, *A Majorization-Minimization Algorithm for the Karcher Mean of Positive Definite Matrices*, SIAM Journal on Matrix Analysis and Applications **38** (2013).

(Recibido en mayo de 2024. Aceptado en mayo de 2024)

Biomedical Imaging, Vision and Learning Laboratory (BIVL^{2}ab)
Universidad Industrial de Santander
Carrera 27, Calle 9
680002, Bucaramanga, Colombia
e-mail: jaolmosr@correo.uis.edu.co

Mathematics Department
Universidad Nacional de Colombia
Carrera 45, Calle 26 -85
Bogotá, Colombia
e-mail: jcgalvisa@unal.edu.co

Biomedical Imaging, Vision and Learning Laboratory (BIVL^{2}ab)
Universidad Industrial de Santander
Carrera 27, Calle 9
680002, Bucaramanga, Colombia
e-mail: famarcar@uis.edu.co