A new look at the Riemannian manifold of positive definite matrices

Anonymous authors
Paper under double-blind review

Abstract

The Riemannian manifold of positive definite matrices endowed with the affine-invariant metric is an important tool in machine learning. The distance between two points P and Q on such a manifold is known to be a function of their generalized eigenvalues. We show that geodesics from P to Q, including the geometric mean, the logarithmic and exponential maps of Q at base-point P as well as the parallel transport of any other point from P to Q, can be all expressed as a function of their generalized eigenvalues and eigenvectors. Thus, all important Riemannian manipulations on the manifold of positive definite matrices can be obtained by computing a joint diagonalizer. As compared to known expressions, those we derive for the geodesic, the logarithmic map, the exponential map and the parallel transport are simpler and lead to a unified conceptual and computational procedure, which is advantageous both in terms of speed and memory allocations. Backpropagation rules for applying this simplified framework in Riemannian neural networks are given.

1 Introduction

Over the past two decades, the manifold of symmetric positive definite matrices $\mathcal{S}_{++}$ has become a prominent framework for machine learning in a ever-growing number of applications. The motivation of the present study stems in particular from its use in the field of brain-computer interfaces (BCIs) based on electroencephalography, which has been among the first to adopt and develop it (Barachant et al., 2010; 2012). In these BCIs, segments of data – which are multivariate time-series — are encoded by means of covariance matrices; once regularized to ensure positivity, these matrices are treated and manipulated as points on the Riemannian manifold of positive definite matrices endowed with the affine-invariant metric (AIM). Discriminative models can then be based on the ensuing distance on the manifold or on standard machine learning tools acting on a tangent space thereof — see Congedo et al. (2017) and Lotte et al. (2018) for a review and Papillon et al. (2025) for an introduction to non-Euclidean machine learning. Nowadays, the approaches based on this geometry in BCIs are well established in terms of performance, robustness and generalizability. Recent large standardized benchmarks have shown their superior performance with respect to alternative standard approaches (Chevallier et al., 2024), achieving performance on par with convolutional neural networks (Eder et al., 2024), at a significantly lower computational cost.

Starting from the seminal work of Huang & Van Gool (2017), the interest of the community has progressively drifted from traditional Riemannnian machine learning models to Riemannian neural networks (Brooks et al., 2019; Pan et al., 2022; Kobler et al., 2022; Li & Gu, 2025; Gallet et al., 2026). Classical deep neural networks operate on vector-valued data, whereas Riemannian deep neural networks are designed to process symmetric (real) positive definite matrices. This paradigm aims at combining the advantageous geometric structure of Riemannian representations with the expressive power of deep learning models. The hope is to exceed the current discriminative performances, which in EEG-based BCI appears to be plateauing since the inception of Riemannian methods. The movement has recently prompted the development of a dedicated software library for developing and testing Riemannian neural networks (Aristimunha et al., 2026). In a recent prospective paper on the subject (Ju et al., 2025), the authors discuss both the potential of these networks and the key challenges they face, the most important of which being the computational complexity. In fact, all

operations on the Riemannian manifold of positive definite matrices require spectral decompositions, which computational cost grows cubically with the matrix size.

This paper is concerned with all fundamental operations involving two matrices on the Riemannian manifold of positive definite matrices endowed with the AIM: the geodesic, including the geometric mean, the logarithmic and exponential map, and the parallel transport. These operations are the backbone and fundamental ingredients of many complex operations that are useful in practice, both for traditional Riemannian machine learning and for Riemannian networks on the manifold of positive definite matrices. While significant advances have been made in the computational treatment of the underlying mathematical expressions—see Iannazzo (2016) for a review—we introduce a new perspective on the manifold that yields immediate computational benefits for all operations. In particular, we unveil a tight relationship between the joint diagonalization of two matrices and the fundamental operations on the manifold involving them. We show that a single generalized eigenvalue-eigevector decomposition of two positive definite matrices allows to compute these operations efficiently, more so than all known procedures. The resulting framework represents a unified view, both algebraically and computationally. Besides the simplified forward operations, we provide the corresponding backpropagation rules for their use in Riemannian neural networks.

The remainder of this paper introduces in a succinct fashion all mathematical operations of interest in the Riemannian manifold of positive definite matrices. In doing so, the attention of the reader is drawn toward their use in Riemannian neural networks. We continue by establishing the link between the joint diagonalization of two matrices and all Riemannian operations, which is an original contribution of this paper. As a result, a unified mathematical expression for the geodesic, the logarithmic map and the exponential map, as well as a new expression for the parallel transport, are provided in terms of the joint diagonalizer only. These yield advantageous computational procedures for the forward computations in a Riemannian neural network. The required backpropagation rules, which are now also unified, are given next, with the proof in Appendix A. This is another original contribution. Then, the simplicity of the proposed joint diagonalization approach is illustrated with example code in the Julia language (Bezanson et al., 2017) and its efficiency is evaluated with respect to the state-of-the art procedures by means of simulations. We continue by briefly commenting on the stability of the forward and backward computations that we have defined and discussing our contribution. Finally, in Appendix B we provide accessory results, found while carrying out this research, that we believe may turn useful in future research.

2 Preliminaries

2.1 The geodesic

Given P and Q two points on $\mathcal{S}_{++}$, the geodesic joining them with arc-length $t \in [0, 1]$ is — e.g., Eq. 1.3 in Iannazzo (2016) or Th. 6.1.6 and Prop. 4.1.9 in Bhatia (2007) —

$$P \#_t Q \coloneqq P(P^{-1}Q)^t = P^{1/2}(P^{-1/2}QP^{-1/2})^t P^{1/2}, \tag{1}$$

where $P^{1/2}$ is the principal square root of P. The geodesic is readily interpreted as the weighted Fréchet mean of P and Q, that is, the point minimizing the weighted dispersion according to the AIM. $P \#_{1/2} Q$, denoted simply as $P \# Q$, is the geometric mean, an object of paramount importance in mathematics (Bhatia, 2007), with countless applications in signal processing and machine learning on $\mathcal{S}_{++}$. The expression (1) is positive definite for $t \in (-\infty, \infty)$, i.e. the path extends indefinitely, allowing extrapolations.

The geodesic enjoys the congruence (or affine) invariance, that is to say, for any non-singular X it holds

$$(X^T P X) \#_t (X^T Q X) = X^T \big(P \#_t Q\big) X, \tag{2}$$

where X^T denotes the transpose of X, and the inversion invariance, reading

$$P^{-1} \#_t Q^{-1} = \big(P \#_t Q\big)^{-1}. \tag{3}$$

For the special case of geodesics intersecting the identity, which is the origin of manifold $\mathcal{S}_{++}$, we have

$$I \#_t Q = Q^t. \tag{4}$$

This is used to adjust the dispersion of points once the data has been recentered around the identity in domain adaptation (Rodrigues et al., 2018) and SPD learning (Kobler et al., 2022).

2.2 The Exponential and Logarithmic Map

Let ξ be a tangent vector in the tangent space at base-point P. The corresponding point on the manifold is given by the exponential map

$$Q = \mathrm{Exp}_P(\xi) \coloneqq P^{1/2}\mathrm{Exp}(P^{-1/2}\xi P^{-1/2})P^{1/2}. \tag{5}$$

The reverse map from the manifold to the tangent space at base-point P is the logaritmic map

$$\xi = \mathrm{Log}_P(Q) \coloneqq P^{1/2}\mathrm{Log}(P^{-1/2}QP^{-1/2})P^{1/2}. \tag{6}$$

In the above and hereafter the uppercase of the first letter differentiates matrix functions from their scalar counterparts, e.g., Exp() and Log() denote the matrix exponential and logarithmic function. On one hand the logarithmic map is needed to lift data points onto the tangent space, on the other hand the exponential map is extensively employed as a retraction in Riemannian optimization methods.

2.3 The Parallel Transport

Given P, Q, and S three points on $\mathcal{S}_{++}$, and ζ the logarithmic map of S onto the tangent space at base-point P, the parallel transport of ζ from this tangent space to the tangent space at base-point Q is

$$\mathcal{P}_{P\to Q}(\zeta) = (QP^{-1})^{1/2}\zeta(QP^{-1})^{T/2}. \tag{7}$$

The exponential map of the transported tangent vector $\mathrm{Exp}_Q(\mathcal{P}_{P\to Q}(\zeta))$, is (Yair et al., 2019)

$$\mathcal{P}_{P\to Q}(S) = (QP^{-1})^{1/2}S(QP^{-1})^{T/2}. \tag{8}$$

Equation (8) shows that it is possible to transport points on the manifold without passing by tangent space lifting. This is of great practical utility. In particular, denoting by I the identity matrix,

$$\mathcal{P}_{P\to I}(S) = P^{-1/2}SP^{-1/2}, \tag{9}$$

which, when P is the geometric mean of a cloud of points, is the recentering (whitening) introduced by Zanini et al. (2018) in the context of domain adaptation and

$$\mathcal{P}_{I\to Q}(S) = Q^{1/2}SQ^{1/2}, \tag{10}$$

which, by analogy, is named coloration. Like the geodesic, the parallel transport enjoys the congruence and inversion invariance; for any non-singular X it holds

$$\mathcal{P}_{X^TPX\to X^TQX}(X^TSX) = X^T\mathcal{P}_{P\to Q}(S)X \tag{11}$$

and it holds

$$\mathcal{P}_{P^{-1}\to Q^{-1}}(S^{-1}) = \left(\mathcal{P}_{P\to Q}(S)\right)^{-1}. \tag{12}$$

The celebrated batch normalization involves the subtraction of the mean, the standardization of the variance and the introduction of a bias (Ioffe & Szegedy, 2015). It has been transposed to Riemannian neural networks by several authors (Brooks et al., 2019; Gallet et al., 2026; Kobler et al., 2022). Given a positive definite matrix C_i belonging to batch $\{C_i\}_{i=1}^k$, the general expression obtained by these authors to normalize each element of the batch is

$$C_i \leftarrow M^{1/2}(M_e^{-1/2}C_iM_e^{-1/2})^tM^{1/2}, \quad \forall i \in \{1,\dots,k\} \tag{13}$$

where

- M_e is an empirical mean (mean of the batch or a running mean),

- the expression within parentheses recenters the data at the identity via (9) (mean subtraction),
- t is an optional learnable parameter accomodating the dispersion via (4) (variance standardization),
- M is a learnable parameter to recolor (bias) the data via (10).

Now, since

$$(QP^{-1})^{1/2} = Q^{1/2}P^{-1/2} \iff PQ = QP, \tag{14}$$

we see that (13) is equivalent to the parallel transport from M_e to M only if they commute, which is not the case in general. This suggests a possible revision of the batch-normalization layers as they are currently defined, which will be possible thanks to the backpropagation rules we release in section 3.3.

3 Methods

3.1 Joint Diagonalization

Given two positive-definite symmetric matrices P and Q, there always exists a non-singular matrix B such that

$$\begin{cases} B^T P B = I \\ B^T Q B = \Lambda, \end{cases} \tag{15}$$

where Λ is diagonal and positive definite. Factorization (15) is the well-known *generalized eigenvalue-eigenvector decomposition (GEVD)* and B is named a joint (or simultaneous) diagonalizer. B is orthogonal if and only if P and Q commute (see Schott (1997), p. 160–165). For the remainder of the paper, let us denote $A \coloneqq B^{-1}$, the matrix such that $A(P^{-1}Q)B = \Lambda$. Also, let us remind that if B is a joint diagonalizer, so is any matrix BCD, with C a permutation matrix and D a diagonal (scaling) matrix. The permutation is arbitrary; it is often taken as the one sorting the diagonal elements of Λ in non-decreasing order. As per the scaling, in this work we always adopt convention (15), unless otherwise specified.

The preferred way to compute the *GEVD* is by means of a two-step procedure (Fukunaga (1990), p. 24–33), given in Algorithm 1:

Algorithm 1 (2-step *GEVD*): Let W be a matrix such that $W^T P W = I$ and U the orthogonal matrix holding in the columns the eigenvectors of $W^T Q W$. The joint diagonalizer as per (15) is $B = WU$ and its inverse is $A = U^T W^{-1}$.

The implementation in linear algebra libraries for programming languages such as Julia, Matlab and Python pick $W = L^{-T}$, where L is the (lower triangular) Cholesky factor of P. From (15), it follows immediately:

$$A \coloneqq B^{-1} = B^T P \tag{16}$$

$$P = A^T A \tag{17}$$

$$Q = A^T \Lambda A \tag{18}$$

$$QP^{-1} = A^T \Lambda B^T \tag{19}$$

A few remarks:

- The computation of A, which is essential in the following, does not necessitate the explicit inversion of B; it is therefore numerically stable.
- B diagonalizes any matrix $P \#_t Q$ for $t \in (-\infty, \infty)$.
- Diagonal matrix Λ holds the eigenvalues of $P^{-1}Q$, which are also the eigenvalues of $P^{-1/2}QP^{-1/2}$. Many divergences and distances on $\mathcal{S}_{++}$; for instance, all divergences/distances of the $\alpha\beta$-logdet family, which include the Stein, Jeffrey, Burg and AIM as special cases, are obtained as functions of Λ (Cichocki et al., 2011). The AIM distance is given by the sum of the squared logarithms, that is,

$$\delta^2(P, Q) = \mathrm{tr}(\mathrm{Log}^2(\Lambda)), \tag{20}$$

  where tr() denotes the trace of the argument.

We will now show that the computation of the *GEVD* is pretty much all that is needed to compute not only the distance, but also the geodesics, logarithmic and exponential maps, as well as the parallel transport on manifold $\mathcal{S}_{++}$ endowed with the AIM. This is a decisive advantage of our approach because, for computing the *GEVD*, we can rely on highly optimized LAPACK routines [1].

3.2 New Expressions

Comparing the expression of the geodesic (1), of the exponential map (5) and of the logarithmic map (6), it appears natural to generalize them by introducing the following

Definition 1 Let

$$\varphi(f, P, Q) := Pf(P^{-1}Q) = P^{1/2} f(P^{-1/2} Q P^{-1/2}) P^{1/2}, \tag{21}$$

where P and Q are two points on $\mathcal{S}_{++}$, and $f : \mathbb{R}_{++} \to \mathbb{R}$ is a function defined on the spectrum of $P^{-1}Q$. Then $\varphi(f, P, Q)$ is the expression of, among others:

- $P \#_t Q$, the geodesic from P to Q with arc-length t, for $f(\lambda) = \lambda^t$,
- $\mathrm{Log}_P(Q)$, the logaritmic map of Q at base-point P, for $f(\lambda) = \log(\lambda)$,
- $\mathrm{Exp}_P(Q)$, the exponential map of Q at base-point P, for $f(\lambda) = \exp(\lambda)$.

The matrix $\varphi(f, P, Q)$ is positive definite if f maps from and to the positive real half-line. For a thorough study of $\varphi(f, P, Q)$, see Kubo & Ando (1980) and Bini et al. (2026). The key property for our purposes is the following:

Lemma 1 *Congruence invariance of the $\varphi(f, P, Q)$ function.* Let P and Q be two points on $\mathcal{S}_{++}$ and $\varphi(f, P, Q)$ as defined in (21), then for any non-singular matrix X it holds

$$\varphi(f, X^T P X, X^T Q X) = X^T \varphi(f, P, Q) X. \tag{22}$$

Proof.

$$\begin{aligned}
\varphi(f, X^T P X, X^T Q X) &= X^T P X \; f(X^{-1} P^{-1} \cancel{X^{-T} X^T} Q X) && \text{from definition (21)} \\
&= X^T P \cancel{X X^{-1}} f(P^{-1} Q) X && \text{(similarity invariance)} \\
&= X^T \varphi(f, P, Q) X && \text{again, from definition (21).}
\end{aligned}$$

□

The first of our two main results is found combining (21), (22) and (15):

Proposition 1 Let P and Q be two points on $\mathcal{S}_{++}$, B their joint diagonalizer as per (15) and $A := B^{-1}$, then

$$\varphi(f, P, Q) = A^T f(\Lambda) A. \tag{23}$$

Proof. From definition (21), we have $\varphi(f, I, \Lambda) = f(\Lambda)$. Therefore,

$$\begin{aligned}
A^T f(\Lambda) A &= A^T \varphi(f, I, \Lambda) A \\
&= \varphi\big(f, A^T A, A^T \Lambda A\big) && \text{by (22)} \\
&= \varphi(f, P, Q) && \text{by (17) and (18).}
\end{aligned}$$

□

Several results follow:

$$P \#_t Q = A^T \Lambda^t A \qquad \text{(geodesic)}, \tag{24}$$

$$P \# Q = A^T \Lambda^{1/2} A \qquad \text{(geometric mean)}, \tag{25}$$

$$\mathrm{Log}_P(Q) = A^T \mathrm{Log}(\Lambda) A \qquad \text{(logarithmic map)}, \tag{26}$$

$$\mathrm{Exp}_P(Q) = A^T \mathrm{Exp}(\Lambda) A \qquad \text{(exponential map)}. \tag{27}$$

[1] These LAPACK routines are called by ‘eigen‘ in Julia, ‘eig‘ in Matlab and ‘eigh‘ in Scipy.

The proof of (25) was given in Congedo et al. (2015). To the best of our knowledge, the generalization (23), therefore expressions (24), (26) and (27), are new. Note that the arithmetic mean and the harmonic mean can also be obtained by (23) with (Kubo & Ando, 1980)

$$\text{Arithmetic: } f(\lambda) = \frac{1+\lambda}{2}; \quad \text{Harmonic: } f(\lambda) = \frac{2\lambda}{1+\lambda}. \tag{28}$$

Let us now turn to the parallel transport. Its derivation as a function of the joint diagonalizer as per (15) is obtained substituting (19) in (7) and (8), yielding

Corollary 1 Given P, Q, and S three points on $\mathcal{S}_{++}$, ζ the logarithmic map of S onto the tangent space at base-point P, B the joint diagonalizer of P and Q as per (15), and $A \coloneqq B^{-1}$, the parallel transport of ζ from the tangent space at base-point P to the tangent space at base-point Q is

$$\mathcal{P}_{P\to Q}(\zeta) = A^T \Lambda^{1/2} B^T \zeta B \Lambda^{1/2} A. \tag{29}$$

The exponential map $\mathrm{Exp}_Q(\mathcal{P}_{P\to Q}(\zeta))$ from the tangent space at base-point Q onto the manifold is

$$\mathcal{P}_{P\to Q}(S) = A^T \Lambda^{1/2} B^T S B \Lambda^{1/2} A. \tag{30}$$

Taken together, the results of this section demonstrate that, given two matrices in $\mathcal{S}_{++}$, one can compute the distance between them, the geodesics joining them, the logaritmic map of one point in the tangent space having the other as base-point and the parallel transport of any other point from one to the other, among other functions, at a cost that basically amounts to a generalized eigenvalue-eigenvector decomposition. The exponential map is obtained exactly with the same computations starting from a point on the manifold and one on the tangent space. The new expressions we have found for the geodesic (24) — including the geometric mean (25), the logarithmic (26) and exponential (27) map, as well as the parallel transport (30) — are considerably simpler than all known counterparts. In Section 4.1 we show that this readily translates in a simpler computational procedure, hence to improved computational efficiency.

3.3 Backpropagation

In this section we provide the backpropagation rules for using in a neuronal network the new expressions we have found for the geodesic (24) — including the geometric mean (25) — the logarithmic (26) and exponential (27) map, as well as the parallel transport (30). All but the latter are special instances of function $\varphi(f, P, Q)$ in (23), for which we have

Proposition 2 Let P and Q be two points on $\mathcal{S}_{++}$, B their joint diagonalizer as per (15), $A \coloneqq B^{-1}$ and Λ their generalized eigenvalues. Given the error $\frac{\partial \ell}{\partial \varphi}$ of the loss with respect to the output $\varphi(f, P, Q)$ of (23), the errors $\frac{\partial \ell}{\partial P}$ and $\frac{\partial \ell}{\partial Q}$ with respect to the inputs P and Q are given by

$$\frac{\partial \ell}{\partial P} = B\,\mathrm{sym}\left(A \frac{\partial \ell}{\partial \varphi} A^T f(\Lambda) - A \frac{\partial \ell}{\partial Q} A^T \Lambda\right) B^T, \tag{31}$$

$$\frac{\partial \ell}{\partial Q} = B\left(K_{f,\Lambda} \odot \left(A \frac{\partial \ell}{\partial \varphi} A^T\right)\right) B^T, \tag{32}$$

where $\mathrm{sym}(\cdot)$ returns the symmetric part of its argument, $\odot$ is the Hadamard product and $K_{f,\Lambda}$ is the Löwner matrix with elements

$$(K_{f,\Lambda})_{ij} = \begin{cases} \frac{f(\lambda_i) - f(\lambda_j)}{\lambda_i - \lambda_j}, & \lambda_i \neq \lambda_j \\ f'(\lambda_i), & \lambda_i = \lambda_j. \end{cases} \tag{33}$$

The proof is given in appendix A.

We now turn to the parallel transport. Let

$$\psi(f, P, Q) = B f(\Lambda) A. \tag{34}$$

With $f(\lambda) = \sqrt{\lambda}$, this function allows to compute the parallel transport via the expression we have provided in (30). The general backpropagation rule is given by

Proposition 3 Let P and Q be two points on $\mathcal{S}_{++}$, B their joint diagonalizer as per (15), $A \coloneqq B^{-1}$ and Λ their generalized eigenvalues. Given the error $\frac{\partial \ell}{\partial \psi}$ of the loss with respect to the output $\psi(f, P, Q)$ of (34), the errors $\frac{\partial \ell}{\partial P}$ and $\frac{\partial \ell}{\partial Q}$ with respect to the inputs P and Q are given by

$$\frac{\partial \ell}{\partial P} = B \operatorname{sym}\left(\operatorname{skew}\left(A \frac{\partial \ell}{\partial \psi}^T B\right) f(\Lambda) - A \frac{\partial \ell}{\partial Q} A^T \Lambda\right) B^T, \tag{35}$$

$$\frac{\partial \ell}{\partial Q} = B \left(K_{f,\Lambda} \odot \operatorname{sym}\left(A \frac{\partial \ell}{\partial \psi}^T B\right)\right) B^T, \tag{36}$$

where $\operatorname{skew}(\cdot)$ returns the skew-symmetric part of its argument and $K_{f,\Lambda}$ is the Löwner matrix defined in (33). The proof is given in appendix A.

4 Results

4.1 Efficiency

In this section we report the result of a benchmark to assess the numerical efficiency of the proposed general expression (23) for computing the φ function with respect to the state-of-the-art alternative. This benchmark serves for comparing the efficieny of geodesics and exponential maps, since the computations are in all cases identical, with the exception of the f function applied to the generalized eigenvalues. We also report the result of a benchmark to assess the numerical efficiency of expression (30) for computing the parallel transport as compared to the standard expression (8). All benchmarks have been carried out using the *BenchmarkTools.jl* package for the Julia language (Bezanson et al., 2017)(v1.12.4), on a laptop equipped with a Intel® Core™ Ultra 7 258V @2.2GHz and 32GB of RAM. BLAS, used by Julia for linear algebra operations and by LAPACK, was instructed to run on a single thread; this way the execution time is roughly proportional to flops count. For each of several matrix sizes (n), we report the number of samples used, which was fixed to the maximum of 1000 and the nearest integer of $10^6/n \log_2(n)$. The reported execution time is the median across the samples. This has been preferred to the minimum execution time in order to provide realistic estimations of execution time. In addition, we report the size of memory allocation (*mem*) and memory allocation counts (*alloc*), which are key indicators of efficiency in some applications.

Julia code for computing (23) is almost as straightforward as pseudo-code:

```
using LinearAlgebra, PosDefManifold

function phi_naive(f::Function, P, Q::Hermitian{T}) where T<:Union{Real, Complex}
    λ, B = eigen(Q, P)
    A = B' * P
    return Hermitian(A' * f(Diagonal(λ)) * A)
end

# Example usage on two randomly generated SPD matrices P and Q
P = randP(4)
Q = randP(4)
logarithmic_map = phi(P, Q, log)
geodesic_t = phi(P, Q, x->x^t) # anonymous functions allow to design any f
```

Note that the above function works in the real and complex case alike. When the function f maps to the strictly positive half real line, about 15 % memory allocations can be saved by forming the matrix $F = A^T(f(\Lambda))^{1/2}$ and multiplying it by its adjoint using BLAS routine *syrk* (for real matrices) or *herk* (for complex matrices), so as to compute only the upper triangle of the product. This yields

```
function phi(f::Function, P, Q::Hermitian{T}) where T<:Union{Real, Complex}
    λ, B = eigen(Q, P)
    mul!(B, P, B*Diagonal(map(x -> sqrt(f(x)), ))) # F = P*B*sqrt(f(Λ))
    return T<:Real ? Hermitian(BLAS.syrk('U', 'N', 1.0, B)) :
                     Hermitian(BLAS.herk('U', 'N', 1.0, B)) # (F * F')
end
```

We compare the 'phi' implementation here above to Algorithm 3 in Iannazzo (2016), which uses the "Cholesky-Schur" method. This is the most efficient currently known in the literature (Bini et al., 2026). The φ function is computed by this method as $LVT^tV^TL^T$, where L is the Cholesky factor of P and VTV^T is the eigenvalue-eigenvector decomposition (EVD) of $L^{-1}QL^{-T}$. Table 1 reports the results. The execution time is always inferior for the $GEVD$ method, although the advantage is small for matrix size ≥ 16. Both the memory engagement and allocation counts are consistently reduced by a factor ≈ 2 using the $GEVD$ method. We conclude that the $GEVD$ method is more efficient for all matrix sizes, with an interesting reduction in memory engagement and memory allocations.

Table 1: Performance comparison of the proposed $GEVD$ method via (24) and the "Cholesky-Schur" method for computing the geodesic with arc-length 0.5 (geometric mean).

n	samples	GEVD (a)			Cholesky-Schur (b)			Ratio ($\frac{a}{b}$)		
		time	mem	alloc	time	mem	alloc	time	mem	alloc
4	125000	1.9*	1.89†	20	3.0*	4.30†	36	0.63	0.44	0.56
8	41667	5.4*	4.58†	20	7.40*	10.26†	37	0.73	0.45	0.54
16	15625	16.7*	14.50†	25	20.9*	31.46†	49	0.8	0.46	0.51
32	6250	63.0*	52.11†	25	70.6*	109.38†	50	0.9	0.48	0.50
64	2604	279.8*	199.4†	26	285.45*	409.17†	50	0.98	0.49	0.52
128	1116	1.16**	781.93†	26	1.36**	1587.2†	51	0.85	0.49	0.51
256	1000	6.17**	3.03‡	28	6.6**	6.09‡	53	0.93	0.5	0.53
512	1000	38.5**	12.05‡	28	43.35**	24.19‡	53	0.88	0.5	0.53

* microseconds (μs); [$**$] milliseconds (ms); [†] kibibyte (KiB); [‡] mebibyte (MiB)

Next, we compare the proposed method to compute the parallel transport via $GEVD$ by (30) to the standard expression (8). For computing (8), QP^{-1} is obtained by Cholesky decomposition and back-substitution to avoid the explicit inversion of P. In total, the standard expression involves one inversion, one square root of a non-symmetric matrix and three multiplications. A naive implementation of the proposed $GEVD$ method involves computing one $GEVD$ and four matrix multiplications [2]. The code is

```
function pt_naive(S, P, T::Hermitian{T}) where T<:RealOrComplex
    λ, B = eigen(Q, P)
    C = P*B*sqrt(Diagonal(λ))*B'
    return Hermitian(C*S*C')
end
```

Table 2 reports the results. The execution time is about two times lower for the proposed method, except for the smallest matrices ($n = 8$) where it is about 15% lower. The memory engagement is always lower for the proposed method, although not by much in general. The allocation count is about 20% lower for the proposed method. We conclude the $GEVD$ method is always more efficient in all considered aspects.

Regrding the efficiency of the backpropagation rules given in Section 3.3, all of them are based on a single GEVD, wherease the backpropagation rules corresponding to the expressions known before the present paper are based on two eigenvalue-eigenvector decompositions. Our approach is therefore more efficient also for what it concerns the backpropagation.

4.2 Stability

Here stability refers to the numerical precision of the computations. This depends on rounding-off numerical errors, which increases as the involved number of algebraic operations increases, and on the conditioning of input matrices with respect to inversion (Iannazzo, 2016).

[2]For simplicity, we do not mention operations requiring n or n^2 operations, such as multiplication with a diagonal matrix, as those are dominated by terms proportional to n^3

Table 2: Performance comparison of the proposed *GEVD* method via (30) and expression (8) for computing the parallel transport.

n	samples	evals	GEVD (a)			Standard (8) (b)			Ratio ($\frac{a}{b}$)		
			time	mem	alloc	time	mem	alloc	time	mem	alloc
8	4167	2083	8.94*	6.34†	27	10.66*	8.51†	34	0.84	0.74	0.79
16	1562	781	28.26*	20.96†	35	49.90*	55.32†	44	0.57	0.38	0.8
32	625	312	84.06*	76.57†	35	166.02*	116.32†	44	0.5	0.66	0.8
64	260	130	601.93*	295.87†	36	1078.0*	358.88†	44	0.55	0.82	0.82
128	112	56	2.55**	1.14‡	36	6.25**	1.29‡	44	0.41	0.88	0.82
256	49	24	8.42**	4.53‡	38	21.09**	5.07‡	47	0.4	0.89	0.81
512	22	11	85.69**	18.05‡	38	198.58**	20.15‡	47	0.43	0.9	0.81

* microseconds (μs); [$**$] milliseconds (ms); [†] kibibyte (KiB); [‡] mebibyte (MiB)

Regarding the φ function, involved in the forward computations for the geodesic, the logarithmic map and the exponential map, we note that the "Cholesky-Schur" method is algebraically equivalent to the *GEVD* method we have proposed. In practice, they both involve the Cholesky and *EVD* decomposition, with some matrix multiplications. Therefore, the numerical stability of the two methods with respect to the conditioning of the input matrices is expected to be roughly equal. Yet, our method, which is simpler algebraically, involves less matrix multiplications, thus it is expected to feature less rounding error (Bini et al., 2026).

Regarding the forward computations for the parallel transport, our method involves the same operations as per the φ function, with the addition of two matrix multiplications. The standard method (8) involves a matrix inversion, the square root of a non-symmetric matrix and two matrix multiplications. The square root of a non-symmetric matrix makes the standard method unfavorable — see Bini et al. (2026).

Regarding the backpropagation rules we have given in Section 3.3, they are based on the computation of a single *GEVD* and some matrix multiplications. Therefore, their stability is expected comparable to the existing backpropagation rules for the standard expressions, which are based on *EVDs* and some matrix multiplications.

5 Concusions and Discussion

In this paper we have presented new expressions for the geodesic, logarithmic maps, exponential maps and parallel transport involving two matrices in the manifold of positive definite matrices endowed with the AIM. These expressions are simpler and more efficient as compared to existing ones, with equal if not better stability. We have provided forward computation procedures and backpropagation rules for applying the proposed expressions to develop more efficient version of existing and future Riemannian neural networks. Whereas we have focused on Riemannian neural networks, these expressions for the forward computations are fundamental and can be applied in any context.

While the φ function is well known to matrix analysts, the connection with the joint diagonalization — i.e., (23) — has been neglected so far. Only for the geometric mean, it was exploited in Congedo et al. (2015). We argue that the *GEVD* method for computing the φ function cannot be surpassed from a computational point of view. This is because φ is a function of the *GEVD* of positive definite matrices P and Q and computing it directly is very efficient using specialized LAPACK routines; other methods are workarounds that do not involve the direct computation of the *GEVD* and cannot be as fast — of course, unless new theoretical results are found allowing to bypass the *GEVD* as well.

Given two positive definite matrices P and Q, the matrix $P^{-1}Q$ assumes an important role in data science. We have already mentioned that a real function of its eigenvalues Λ yields the distance between P and Q according to a large family of metrics. Its eigenvectors are the columns of B, the joint diagonalizer. The present paper has shown that geodesics, the exponential and logarithmic map, as well as the parallel transport are all functions of Λ and B. In the literature on signal processing, the joint diagonalizer yields a

myriad of optimal filters depending on the choice of P and Q; for example, when P and Q are some means of covariance matrices in two distinct classes, it yields the spatial filter known in the BCI field as "common spatial pattern" (Ramoser et al., 2000); when P and Q are the covariance and lagged covariance matrix of a multivariate time-series, it provides a solution to the blind source separation problem (Molgedey & Schuster, 1994), etc. The ubiquitous role of the joint diagonalizer makes even more intriguing its connection with the Riemannian geometry of positive definite matrices that we have clarified in this paper.

While here we have focused on expressions involving two positive definite matrices, further research is needed to facilitate the computation of operations involving more than two at once. Above all, the computation of the barycenter of a batch of data points. As we have seen, this is instrumental in designing batch-norm layers. Adopting the AIM, no closed-form solution is known. The standard gradient descent algorithm features high complexity per iteration and linear convergence. The need to repeat the estimation for each data batch makes the use of iterative algorithms prohibitive (Ju et al., 2025). As a matter of fact, none of the existing networks adopts the AIM for computing the barycenter. Instead, only metrics yielding a closed-form solution for the barycenter have been adopted. Following Huang & Van Gool (2017), the log-Euclidean metric is almost always chosen. The use of the Jeffrey metric has been proposed in Gallet et al. (2026). As compared to the former, the latter has the advantage of allowing a barycenter enjoying the congruence and inversion invariance, like for the AIM. Besided the consideration of aspects related to the computational efficiency, further work is needed to understand how the chosen metric for computing a barycenter affects the performance of the network.

References

Bruno Aristimunha, Cheng Ju, Antoine Collas, Frédéric Bouchard, Adil Mian, Bertrand Thirion, Sylvain Chevallier, et al. Spd learn: A geometric deep learning python library for neural decoding through trivialization. *arXiv preprint arXiv:2602.22895*, 2026. URL `https://arxiv.org/abs/2602.22895`. arXiv:2602.22895.

Alexandre Barachant, Stéphane Bonnet, Marco Congedo, and Christian Jutten. Riemannian geometry applied to bci classification. In *International Conference on Latent Variable Analysis and Signal Separation*, Lecture Notes in Computer Science, pp. 629–636. Springer Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-15995-4_79.

Alexandre Barachant, Stéphane Bonnet, Marco Congedo, and Christian Jutten. Multi-class brain computer interface classification by riemannian geometry. *IEEE Transactions on Biomedical Engineering*, 59(4): 920–928, April 2012. doi: 10.1109/TBME.2011.2172210.

Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017. doi: 10.1137/141000671.

Rajendra Bhatia. *Positive Definite Matrices*. Princeton University Press, 2007.

Dario A. Bini, Massimiliano Fasi, and Bruno Iannazzo. Computing matrix functions associated with a hermitian–definite pencil. *arXiv preprint arXiv:2510.16473*, 2026. doi: 10.48550/arXiv.2510.16473. URL `https://arxiv.org/abs/2510.16473`.

Daniel Brooks, Olivier Schwander, Frederic Barbaresco, Jean-Yves Schneider, and Matthieu Cord. Riemannian batch normalization for SPD neural networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Sylvain Chevallier, Igor Carrara, Bruno Aristimunha, Pierre Guetschel, Sara Sedlar, Bruna Junqueira Lopes, Sébastien et al.Velut, Salim Khazem, and Thomas Moreau. The largest EEG-based BCI reproducibility study for open science: the MOABB benchmark. April 2024. URL `https://universite-paris-saclay.hal.science/hal-04537061`. working paper or preprint.

Andrzej Cichocki, Sergio Cruces, and Shun-ichi Amari. Generalized alpha-beta divergences and their application to robust nonnegative matrix factorization. *Entropy*, 13(1):134–170, 2011. doi: 10.3390/e13010134.

Marco Congedo, Bijan Afsari, Alexandre Barachant, and Maher Moakher. Approximate joint diagonalization and geometric mean of symmetric positive definite matrices. *PLoS ONE*, 10(4):e0121423, 2015.

Marco Congedo, Alexandre Barachant, and Rajendra Bhatia. Riemannian geometry for eeg-based brain-computer interfaces: a primer and a review. *Brain-Computer Interfaces*, 4(3):155–174, September 2017. doi: 10.1080/2326263X.2017.1297192.

Manuel Eder, Jiachen Xu, and Moritz Grosse-Wentrup. Benchmarking brain-computer interface algorithms: Riemannian approaches vs convolutional neural networks. *Journal of Neural Engineering*, 21(4), 2024. doi: 10.1088/1741-2552/ad6793.

Miroslav Fiedler and Vlastimil Pták. A new positive definite geometric mean of two positive definite matrices. *Linear Algebra and its Applications*, 251:1–20, January 1997. doi: 10.1016/S0024-3795(96)00269-4.

Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, London, 2 edition, 1990.

Matthieu Gallet, Florent Bouchard, Ammar Mian, and Guillaume Ginolhac. ARMAGNAC: A new parametric batch normalization layer for SPDNet architecture, 2026. hal-05529546.

Zhiwu Huang and Luc Van Gool. A riemannian network for spd matrix learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, pp. 2036–2042, 2017.

Bruno Iannazzo. The geometric mean of two matrices from a computational viewpoint. *Numerical Linear Algebra with Applications*, 23(2):208–229, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, volume 37 of *Proceedings of Machine Learning Research*, pp. 448–456, Lille, France, 2015. PMLR.

Cheng Ju, Reinmar Kobler, Antoine Collas, Motoaki Kawanabe, Cuntai Guan, and Bertrand Thirion. Spd matrix learning for neuroimaging analysis: Perspectives, methods, and challenges. *arXiv preprint arXiv:2504.18882*, 2025. URL `https://arxiv.org/abs/2504.18882`. arXiv:2504.18882.

Reinmar Kobler, Jun ichiro Hirayama, Qibin Zhao, and Motoaki Kawanabe. Spd domain-specific batch normalization to crack interpretable unsupervised domain adaptation in EEG. In *Advances in Neural Information Processing Systems*, volume 35, 2022. Main Conference Track (NeurIPS 2022).

Fumio Kubo and Tsuyoshi Ando. Means of positive linear operators. *Mathematische Annalen*, 246(3): 205–224, 1980. doi: 10.1007/BF01371042.

Changchun Li and Zhenghui Gu. A riemannian convolutional neural network for eeg-based motor imagery decoding. *Neurocomputing*, 639:130206, April 2025. doi: 10.1016/j.neucom.2024.130206.

Fabien Lotte, Laurent Bougrain, Andrzej Cichocki, Maureen Clerc, Marco Congedo, Alain Rakotomamonjy, et al. A review of classification algorithms for eeg-based brain–computer interfaces: a 10 year update. *Journal of Neural Engineering*, 15(3):031005, 2018. doi: 10.1088/1741-2552/aab2f2.

Lutz Molgedey and Heinz G. Schuster. Separation of a mixture of independent signals using time delayed correlations. *Phys. Rev. Lett.*, 72:3634–3637, Jun 1994. doi: 10.1103/PhysRevLett.72.3634. URL `https://link.aps.org/doi/10.1103/PhysRevLett.72.3634`.

Yu-Ting Pan, Jui-Lung Chou, and Chia-Shiang Wei. Matt: a manifold attention network for eeg decoding. In *Advances in Neural Information Processing Systems*, volume 35, pp. 31116–31129, 2022.

Mathilde Papillon, Sophia Sanborn, Johan Mathe, Louisa Cornelis, Abby Bertics, Domas Buracas, Hansen J. Lillemark, Christian Shewmake, Fatih Dinc, Xavier Pennec, and Nina Miolane. Beyond euclid: an illustrated guide to modern machine learning with geometric, topological, and algebraic structures. *Machine Learning: Science and Technology*, 6(3):031002, 2025. doi: 10.1088/2632-2153/adf375.

Herbert Ramoser, Johannes Müller-Gerking, and Gert Pfurtscheller. Optimal spatial filtering of single trial eeg during imagined hand movement. *IEEE Transactions on Rehabilitation Engineering*, 8(4):441–446, December 2000. doi: 10.1109/86.895946.

Pedro L.C. Rodrigues, Marco Congedo, and Christian Jutten. Multivariate time-series analysis via manifold learning. In *Proc. of the IEEE Statistical Signal Processing Workshop (SSP 2018)*, Fribourg-en-Brisgau, Germany, 2018.

James R. Schott. *Matrix Analysis for Statistics*. John Wiley & Sons, New York, 1997.

Or Yair, Mirela Ben-Chen, and Ronen Talmo. Parallel transport on the cone manifold of SPD matrices for domain adaptation. *IEEE Transactions on Signal Processing*, 67(7):1797–1811, 2019.

Paolo Zanini, Marco Congedo, Christian Jutten, Salem Said, and Yannick Berthoumieu. Transfer learning: A riemannian geometry framework with applications to brain-computer interfaces. *IEEE Transactions on Biomedical Engineering*, 65(5):1107–1116, May 2018. doi: 10.1109/TBME.2017.2742541.

A Appendix

A.1 Proof of Proposition 2

Proof. By conservation of the error, we have

$$\operatorname{tr}\left(\frac{\partial \ell}{\partial \varphi}\mathrm{d}\varphi(f,P,Q)\right) = \operatorname{tr}\left(\frac{\partial \ell}{\partial P}\mathrm{d}P\right) + \operatorname{tr}\left(\frac{\partial \ell}{\partial Q}\mathrm{d}Q\right).$$

Moreover,

$$\begin{aligned} \mathrm{d}P = A^T\mathrm{d}A + \mathrm{d}A^T A, \quad \mathrm{d}Q = A^T\mathrm{d}\Lambda A + A^T\Lambda\mathrm{d}A + \mathrm{d}A^T\Lambda A \\ \text{and} \quad \mathrm{d}\varphi(f,P,Q) = A^T f'(\Lambda)\mathrm{d}\Lambda A + A^T f(\Lambda)\mathrm{d}A + \mathrm{d}A^T f(\Lambda)A. \end{aligned}$$

It follows that

$$\begin{aligned} \operatorname{tr}\left(\frac{\partial \ell}{\partial \varphi}\mathrm{d}\varphi(f,P,Q)\right) &= \operatorname{tr}\left(A\frac{\partial \ell}{\partial \varphi}A^T f'(\Lambda)\mathrm{d}\Lambda + \frac{\partial \ell}{\partial \varphi}A^T f(\Lambda)\mathrm{d}A + f(\Lambda)A\frac{\partial \ell}{\partial \varphi}\mathrm{d}A^T\right) \\ &= \operatorname{tr}(G_\varphi f'(\Lambda)\mathrm{d}\Lambda + G_\varphi f(\Lambda)\mathrm{d}AA^{-1} + f(\Lambda)G_\varphi A^{-T}\mathrm{d}A^T), \end{aligned}$$

where $G_\varphi = A\frac{\partial \ell}{\partial \varphi}A^T$. Denoting $G_P = A\frac{\partial \ell}{\partial P}A^T$ and $G_Q = A\frac{\partial \ell}{\partial Q}A^T$, we also get

$$\begin{aligned} \operatorname{tr}\left(\frac{\partial \ell}{\partial P}\mathrm{d}P\right) + \operatorname{tr}\left(\frac{\partial \ell}{\partial Q}\mathrm{d}Q\right) = \operatorname{tr}\left(G_P[\mathrm{d}AA^{-1} + (\mathrm{d}AA^{-1})^T]\right) \\ + \operatorname{tr}(G_Q\mathrm{d}\Lambda + G_Q\Lambda\mathrm{d}AA^{-1} + \Lambda G_Q(\mathrm{d}AA^{-1})^T). \end{aligned}$$

Let $\mathrm{d}AA^{-1} = \Sigma + \Omega$, the decomposition of $\mathrm{d}AA^{-1}$ into its symmetric part Σ and skew-symmetric part Ω. It follows that

$$\begin{aligned} \operatorname{tr}([G_\varphi f(\Lambda) - f(\Lambda)G_\varphi]\Omega) + \operatorname{tr}([G_\varphi f(\Lambda) + f(\Lambda)G_\varphi]S) + \operatorname{tr}(G_\varphi f'(\Lambda)\mathrm{d}\Lambda) \\ = \operatorname{tr}([G_Q\Lambda - \Lambda G_Q]\Omega) + \operatorname{tr}([G_Q\Lambda + \Lambda G_Q + 2G_P]S) + \operatorname{tr}(G_Q\mathrm{d}\Lambda). \end{aligned}$$

Identification then yields the system of three equations

$$\begin{cases} G_\varphi f(\Lambda) + f(\Lambda)G_\varphi = G_Q\Lambda + \Lambda G_Q + 2G_P, \\ G_\varphi f(\Lambda) - f(\Lambda)G_\varphi = G_Q\Lambda - \Lambda G_Q, \\ \operatorname{diag}(G_\varphi)f'(\Lambda) = \operatorname{diag}(G_Q). \end{cases}$$

The result for $\frac{\partial \ell}{\partial P} = BG_PB^T$ directly follows from the first equation. For $\frac{\partial \ell}{\partial Q} = BG_QB^T$, it is more complicated. The result for the diagonal of G_Q is straightforward from the third equation. For off-diagonal elements, it is driven from the second equation. Let $i \neq j$, we have

$$(G_\varphi)_{ij}(f(\lambda_j) - f(\lambda_i)) = (G_Q)_{ij}(\lambda_j - \lambda_i)$$

For $\lambda_i \neq \lambda_j$, we immediately get

$$(G_Q)_{ij} = \frac{f(\lambda_i) - f(\lambda_j)}{\lambda_i - \lambda_j}(G_\varphi)_{ij}.$$

Finally, the result for $\lambda_i = \lambda_j$ is obtained by recalling that $\lim_{\lambda_j \to \lambda_i} \frac{f(\lambda_i)-f(\lambda_j)}{\lambda_i - \lambda_j} = f'(\lambda_i)$. □

A.2 Proof of Proposition 3

Proof. By definition,

$$\operatorname{tr}\left(\frac{\partial \ell}{\partial \psi}\mathrm{d}\psi(f,P,Q)\right) = \operatorname{tr}\left(\frac{\partial \ell}{\partial P}\mathrm{d}P\right) + \operatorname{tr}\left(\frac{\partial \ell}{\partial Q}\mathrm{d}Q\right).$$

We have

$$\begin{aligned} \mathrm{d}P = A^T\mathrm{d}A + \mathrm{d}A^TA, \quad \mathrm{d}Q = A^T\mathrm{d}\Lambda A + A^T\Lambda\mathrm{d}A + \mathrm{d}A^T\Lambda A \\ \text{and} \quad \mathrm{d}\psi(f,P,Q) = A^{-1}f'(\Lambda)\mathrm{d}\Lambda A + A^{-1}f(\Lambda)\mathrm{d}A - A^{-1}\mathrm{d}AA^{-1}f(\Lambda)A. \end{aligned}$$

It follows that

$$\begin{aligned} &\operatorname{tr}\left(\frac{\partial \ell}{\partial \psi}\mathrm{d}\psi(f,P,Q)\right) \\ &= \operatorname{tr}\left(A\frac{\partial \ell}{\partial \psi}A^{-1}f'(\Lambda)\mathrm{d}\Lambda + \frac{\partial \ell}{\partial \psi}A^{-1}f(\Lambda)\mathrm{d}A - f(\Lambda)A\frac{\partial \ell}{\partial \psi}A^{-1}\mathrm{d}AA^{-1}\right) \\ &= \operatorname{tr}(G_\psi f'(\Lambda)\mathrm{d}\Lambda + [G_\psi f(\Lambda) - f(\Lambda)G_\psi]\mathrm{d}AA^{-1}), \end{aligned}$$

where $G_\psi = A\frac{\partial \ell}{\partial \psi}A^{-1}$. As for the previous proposition, we also have

$$\begin{aligned} \operatorname{tr}\left(\frac{\partial \ell}{\partial P}\mathrm{d}P\right) + \operatorname{tr}\left(\frac{\partial \ell}{\partial Q}\mathrm{d}Q\right) = \operatorname{tr}\left(G_P[\mathrm{d}AA^{-1} + (\mathrm{d}AA^{-1})^T]\right) \\ + \operatorname{tr}(G_Q\mathrm{d}\Lambda + G_Q\Lambda\mathrm{d}AA^{-1} + \Lambda G_Q(\mathrm{d}AA^{-1})^T), \end{aligned}$$

with $G_P = A\frac{\partial \ell}{\partial P}A^T$ and $G_Q = A\frac{\partial \ell}{\partial Q}A^T$. Also exploiting the decomposition $\mathrm{d}AA^{-1} = \Sigma + \Omega$, with Σ symmetric and Ω skew-symmetric, we get

$$\begin{aligned} &\operatorname{tr}([G_\psi f(\Lambda) - f(\Lambda)G_\psi]\Omega) + \operatorname{tr}([G_\psi f(\Lambda) - f(\Lambda)G_\psi]S) + \operatorname{tr}(G_\psi f'(\Lambda)\mathrm{d}\Lambda) \\ &\qquad = \operatorname{tr}([G_Q\Lambda - \Lambda G_Q]\Omega) + \operatorname{tr}([G_Q\Lambda + \Lambda G_Q + 2G_P]S) + \operatorname{tr}(G_Q\mathrm{d}\Lambda). \end{aligned}$$

Identification then yields the system of three equations

$$\begin{cases} \operatorname{sym}(G_\psi f(\Lambda) - f(\Lambda)G_\psi) = G_Q\Lambda + \Lambda G_Q + 2G_P, \\ \operatorname{skew}(G_\psi f(\Lambda) - f(\Lambda)G_\psi) = G_Q\Lambda - \Lambda G_Q, \\ \operatorname{diag}(G_\psi)f'(\Lambda) = \operatorname{diag}(G_Q). \end{cases}$$

Basic manipulations with $\operatorname{sym}(\cdot)$ and $\operatorname{skew}(\cdot)$ operators give

$$\begin{cases} \operatorname{skew}(G_\psi)f(\Lambda) + f(\Lambda)\operatorname{skew}(G_\psi) = G_Q\Lambda + \Lambda G_Q + 2G_P, \\ \operatorname{sym}(G_\psi)f(\Lambda) - f(\Lambda)\operatorname{sym}(G_\psi) = G_Q\Lambda - \Lambda G_Q, \\ \operatorname{diag}(G_\psi)f'(\Lambda) = \operatorname{diag}(G_Q). \end{cases}$$

Finally, the result is obtained by following the same reasoning as for the previous proof. □

B Appendix

In this appendix we include a few additional results, as they are general and may turn useful for future research. A consequence of the *GEVD* view we have exposed is that it is easy to find compact expressions for functions involving two positive definite matrices P and Q and the associated map $\varphi(f, Q, P)$ (22). The reminder of this section provides, among other results, some examples in this sense.

B.1 Whitening matrices

Commenting on the parallel transport to the identity (9), we have mentioned that if G is the geometric mean of a cloud of k points $\{C_i\}_{i=1}^k$, then $G^{-1/2}C_iG^{-1/2}$ minimizes the dispersion (sum of squared distance) of the points to the identity, i.e, the sum of the squared norm associated with the AIM (recentering). $G^{-1/2}$ is the unique symmetric whitening matrix with respect to the geometric mean. In many situations, any whitening matrix, i.e., a non-symmetric one, suffices. In this case and when $k = 2$, there is no need to compute the geometric mean, left alone the inverse square root. Instead, we may use the following

Corollary B.1 Let P and Q two positive definite matrices, B their joint diagonalizer with generalized eigenvectors Λ as per (15) and define $Z \coloneqq B\Lambda^{-1/4}$. Then, using (17), (18) and (25), we see that it holds

$$\begin{cases} Z^T P Z = \Lambda^{-1/2} \\ Z^T Q Z = \Lambda^{1/2} \\ Z^T (Q \# P) Z = I. \end{cases} \tag{37}$$

This shows that Z is a whitening matrix with respect to the geometric mean, like its symmetric counterpart $G^{-1/2}$. Furthermore, by virtue of the congruence invariance of the AIM distance, it holds

$$\delta(Z^T P Z, Z^T Q Z) = \delta(G^{-1/2} P G^{-1/2}, G^{-1/2} Q G^{-1/2}). \tag{38}$$

B.2 Congruence transformations

It is well known that the geometric mean $P\#Q$ is the unique positive definite solution to Riccati equation — see Bhatia (2007), p. 11 —

$$(P\#Q)P^{-1}(P\#Q) = Q. \tag{39}$$

Thus, the geometric mean of P and Q is the unique symmetric positive-definite congruence map transforming P^{-1} into Q. The definition of parallel transport (8) states that a congruence map transforming P into Q is $(QP^{-1})^{1/2}$. In fact,

$$\mathcal{P}_{P\to Q}(P) = (QP^{-1})^{1/2} P (QP^{-1})^{T/2} = Q. \tag{40}$$

This map is not unique. Replacing P^{-1} with P in (39) we see that the unique symmetric positive-definite congruence map is $P^{-1}\#Q$. Combining these arguments one can obtain several expressions relating, by affinity, P, Q and their inverse, such as, for example

$$\begin{aligned} Q &= (P\#Q)P^{-1}(P\#Q) = (QP)^{1/2}P^{-1}(QP)^{-T/2} \\ &= (P^{-1}\#Q)P(P^{-1}\#Q) = (QP^{-1})^{1/2}P(QP^{-1})^{-T/2}; \end{aligned} \tag{41}$$

$$\begin{aligned} Q^{-1} &= (P\#Q^{-1})P^{-1}(P\#Q^{-1}) = (Q^{-1}P)^{1/2}P^{-1}(Q^{-1}P)^{-T/2} \\ &= (P^{-1}\#Q^{-1})P(P^{-1}\#Q^{-1}) = (Q^{-1}P^{-1})^{1/2}P(Q^{-1}P^{-1})^{-T/2}. \end{aligned} \tag{42}$$

B.3 Parallel transport transformation matrix in coordinates

This appendix concentrates on the parallel transport matrix QP^{-1}.

Lemma B.3 Let P and Q be two points on $\mathcal{S}_{++}$, then

$$P^{1/2}(QP^{-1})^{T/2}Q^{-1/2} \quad \text{is orthogonal.} \tag{43}$$

Proof. Let us rewrite (40) as $Q^{-1/2}(QP^{-1})^{1/2}P^{1/2}P^{1/2}(QP^{-1})^{T/2}Q^{-1/2} = I$. This says that if $H^T = Q^{-1/2}(QP^{-1})^{1/2}P^{1/2}$, then $H^T H = I$, i.e., H is orthogonal. □

This result was given in Yair et al. (2019) (Appendix C) without proof and was used therein to prove (8). Bhatia (Bhatia (2007), p. 109) proved that if U is an orthogonal matrix such that $P^{1/2}UQ^{1/2}$ is positive, then $P^{1/2}UQ^{1/2} = P\#Q$, a fact already remarked in Fiedler & Pták (1997). Now, such a U is (43). This is seen as

$$\begin{aligned} P^{1/2}UQ^{1/2} &= P^{1/2}P^{1/2}(QP^{-1})^{T/2}\cancel{Q^{-1/2}Q^{1/2}} \\ &= P(QP^{-1})^{T/2} = P(P^{-1}Q)^{1/2} = P\#Q \qquad \text{from (1).} \end{aligned} \tag{44}$$

Remark B.3 Matrix (43) actually corresponds to the parallel transport transformation matrix in coordinates. Indeed, one can equip the tangent space at P with the orthogonal basis $\{P^{1/2}E_iP^{1/2}\}$, where $\{E_i\}$ is the canonical basis of the space of symmetric matrices $\mathcal{S}$. Then, given some tangent vector $\xi \in \mathbb{R}^{p(p+1)/2}$ at P in coordinates, it becomes a tangent vector at Q through parallel transport with the transformation $(H^T \otimes H^T)\xi$, where H corresponds to (43) and $\otimes$ denotes the Kronecker product. This yields an alternative proof to show that (43) is orthogonal. In fact, the Riemannian metric tensor needs to remain the identity matrix, thus, $(H^T \otimes H^T)(H^T \otimes H^T)^T = I$, which in turn yields $HH^T = I$.

B.4 Geodesic joining the arithmetic and harmonic mean

It is sometimes of interest the geodesic with arc-length parameter t joining the arithmetic ($t = 0$) and the harmonic mean ($t = 1$) of a cloud of k points on $\mathcal{S}_{++}$. For example, this is used in the batch-norm layer of Gallet et al. (2026) to learn t so as to find the mean M in (13) along such geodesic. When $k > 2$, the mid-point of the geodesic ($t = \frac{1}{2}$) is the Jeffrey mean, also known as the symmetrized Kullback-Leibler divergence, i.e., the $\alpha\beta$-logdet divergence with $\alpha \pm 1$ and $\beta \to 0$ (Cichocki et al., 2011). When $k = 2$, however, the mid-point is also the geometric mean $P\#Q$, i.e., the $\alpha\beta$-logdet divergence with $(\alpha, \beta) \to 0$, which is the AIM distance. Using the *GEVD* approach developed in this article, a simple expression for such a geodesic is given by the following

Corollary B.4 Let P and Q be two points on $\mathcal{S}_{++}$. Denoting as usual $P \nabla Q$ and $P \,!\, Q$ their arithmetic and harmonic mean, respectively — e.g., Kubo & Ando (1980) —,

$$(P \nabla Q)\#_t(P \,!\, Q) = \varphi(2^{2t-1}(1+\lambda)^{1-2t}\lambda^t,\, P,\, Q), \tag{45}$$

where the φ function has been defined in (21).

Proof. The f function in (21) applies element-wise to the eigenvalues of $P^{-1}Q$. For scalars, the mid-point of the geodesic joining the arithmetic (a) and harmonic (h) mean is the geometric mean; the geodesic from a to h is thus the weighted geometric mean and is given by $a\#_t b = a^{1-t}h^t$. The result is obtained substituting in this expression the functions in (28) and simplifying. □

The right-hand side of (45) is an efficient and stable way to compute the left-hand side, as it bypasses the computation of the harmonic mean — which involves at least one matrix inversion. An even simpler expression for the geodesic in (45) is obtained by finding a joint diagonalizer W with another scaling, such that $W^T\frac{1}{2}(P+Q)W = I$ and $W^T\frac{1}{2}PW = D$ (which implies $W^T\frac{1}{2}QW = I - D$, see Fukunaga (1990), p. 33-34). Then, computing $W^{-1} := W^T\frac{1}{2}(P+Q)$, the geodesic is

$$(P \nabla Q)\#_t(P \,!\, Q) = W^{-T}D^t(1-D)^tW^{-1}. \tag{46}$$

Notice that matrix W is a whitening matrix with respect to the arithmetic mean and diagonalizes all means along the geodesic — including the geometric and harmonic mean. When P and Q are barycenters of data belonging to distinct classes, the components extracted by W are sorted by the variance ratio between classes; keeping only the first and last $p < \frac{n}{2}$ vectors of W then yields the filter known as common spatial pattern (Fukunaga, 1990; Ramoser et al., 2000).

B.5 The dual of the geodesic

Thanks to the *GEVD* view we have presented, the computation of the dual of the geodesic can be obtained efficiently and without any inversion, i.e., in a stable fashion, thanks to

Corollary B.5 Let P and Q two positive definite matrices and B their joint diagonalizer with generalized eigenvectors Λ as per (15), it holds

$$P^{-1}\#_t Q^{-1} = (P\#_t Q)^{-1} = B\Lambda^{-t}B^T. \tag{47}$$

Proof. The first equality is due to (3). The second is obtained inverting (24). □