Riemannian Procrustes Analysis : Transfer Learning for Brain-Computer Interfaces

Pedro L. C. Rodrigues, *Member, IEEE,* Christian Jutten, *Fellow, IEEE,* and Marco Congedo, *Member, IEEE*

***Abstract*—Objective: This paper presents a Transfer Learning approach for dealing with the statistical variability of EEG signals recorded on different sessions and/or from different subjects. This is a common problem faced by Brain-Computer Interfaces (BCI) and poses a challenge for systems that try to reuse data from previous recordings to avoid a calibration phase for new users or new sessions for the same user. Method: We propose a method based on Procrustes analysis for matching the statistical distributions of two datasets using simple geometrical transformations (translation, scaling and rotation) over the data points. We use symmetric positive definite matrices (SPD) as statistical features for describing the EEG signals, so the geometrical operations on the data points respect the intrinsic geometry of the SPD manifold. Because of its geometry-aware nature, we call our method the Riemannian Procrustes Analysis (RPA). We assess the improvement in Transfer Learning via RPA by performing classification tasks on simulated data and on eight publicly available BCI datasets covering three experimental paradigms (243 subjects in total). Results: Our results show that the classification accuracy with RPA is superior in comparison to other geometry-aware methods proposed in the literature. We also observe improvements in ensemble classification strategies when the statistics of the datasets are matched via RPA. Conclusion and significance: We present a simple yet powerful method for matching the statistical distributions of two datasets, thus paving the way to BCI systems capable of reusing data from previous sessions and avoid the need of a calibration procedure.**

***Index Terms*—Brain-Computer Interface, Riemannian geometry, Transfer Learning, Covariance Matrices, EEG.**

I. Introduction

A Brain-Computer Interface (BCI) is a system that allows a person to interact with a machine without any physical interaction. It works by extracting features from neuro-physiological signals (e.g., the power spectral densities on certain frequency bands) and assigning them to different classes. These classes may be associated to cognitive states, sensory responses, etc., and the features are chosen so that they are discriminative for each class. Among the many types of neural signals that have been used in BCI systems, electroencephalographic (EEG) recordings have received particular attention and are the focus of this paper. The success of EEG-based systems comes from several reasons, such as its low price as compared to other neuro-physiological modalities, its non-invasiveness, and its high temporal resolution. However, they also have to cope with low signal-to-noise ratio, low spatial resolution [1], and high cross-session and cross-subject variability [2].

While many signal processing techniques have been developed to ameliorate the quality of EEG signals and improve its spatial resolution [3], the cross-session and cross-subject variability problem have received much less attention. The standard approach in BCI consists of re-training a statistical classifier at the beginning of every experimental session with the help of a sequence of calibration trials, a procedure that can be time consuming and is clearly suboptimal, since it does not leverage any information from past experiments. Instead, "second-generation BCIs" initialize classifiers using Transfer Learning and update their parameters along sessions, avoiding the calibration step altogether [1].

We present in this paper a Transfer Learning approach based on geometrical transformations for matching the statistical distributions coming from two different experimental sessions, a *source* and a *target* session. For simplicity of exposition, hereafter we use the term "session" generically, however *source* and *target* may refer to different sessions as well as different subjects. We perform simple linear transformations, such as translation, rotation, and scaling, on the data points of both sessions with the goal of making the shape of their statistical distributions as similar as possible. Once the distributions are matched, one can expect that a classifier optimized for the data of the *source* session will work well enough with the data from the *target* session.

The branch of Machine Learning that studies the effects of mismatches between statistical distributions is called Transfer Learning [4]. It has been of great interest in several domains besides BCI, such as in computer vision, where the statistics of the data may vary due to changes in lighting conditions and acquisition devices, or in speech processing systems, where the changes in background noise and the differences in speaker genders and voice tonalities may affect the statistics of the signals. In [5], the phenomenon responsible for the drift in statistical distributions of two datasets was termed *covariate shift* and modelled by assuming that the distributions of the data points can be different for the *source* and *target* datasets, but the conditional distributions of the labels are the same. Ref. [6] presented real data examples on BCI experiments and showed that the *covariate shift* describes well the changes in statistics for this kind of application. A recent attempt in the literature on Transfer Learning has been to apply the theory of Optimal Transport to determine which geometrical transformations one should perform to match two statistical distributions, as proposed in [7] and applied to the BCI P300 paradigm in [8]. This approach begins by first defining a

P.L.C.Rodrigues, C.Jutten, and M.Congedo are with Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France. Manuscript received 25-May-2018 ; revised 09-Oct-2018; accepted 21-Dec-2018.

function that models the cost of transporting a point at location x to a location y. Then, it solves an optimization problem that minimizes the total cost of moving each point of the *source* dataset and make its statistical distribution as similar as possible to the distribution of the *target* dataset. Note that this is a completely unsupervised procedure and does not rely on any assumption regarding the statistical distributions of the datasets.

Another geometrical approach [9] matches the statistical distributions from the *source* and *target* sessions by means of a re-centering of their data points to the origin of the SPD space, the identity matrix. During the process of review of this paper, we came across the work in [10], which also proposes a Transfer Learning procedure for SPD matrices in the spirit of [9], but with the main difference that the data points are re-centered to the midpoint between the geometric means of the *source* and *target* datasets

Besides of distribution matching based on geometrical transformations, there has been mainly two other kinds of proposals for doing Transfer Learning in the BCI literature [2]. One is based on the concept of ensemble classifiers [3], [11]–[13], where the information from multiple *source* datasets are combined into a "global" classifier, which is then used to label the trials from any other *target* dataset. There has also been works using Bayesian models to describe the variability of the statistics on the *source* datasets and gather information from multiple datasets [14]. A recent approach that builds upon such Bayesian methods are the works in [15] and [16], which use a special form of the P300 experimental paradigm to do classification with no calibration. There are two main differences between these approaches and our proposal in this work. First, our approach is based on matching as much as possible the information from each *source-target* pair of datasets. Thus, it can be used in addition to an ensembling approach as in [12], which will combine the information from multiple matched-*source* subjects. Second, our approach is paradigm-agnostic and does not rely on any special modification of the experimental setup where the EEG signals are collected (as opposed to [16]), a feature that is appealing to a great number of practionners.

Our approach for distribution matching is based on the concept of Procrustes Analysis (PA) [17], a tool often employed in statistical shape analysis [18] with applications in text analysis [19], protein alignment [19], and many other fields. PA works by first selecting a set of pairs of landmark points from two different shapes and then performing geometrical transformations to get these landmarks as close as possible to each other. In our context, the shapes to be matched are actually point clouds consisting of high-dimensional statistical features describing EEG signals, and the landmarks are points that can be used to describe these statistical distributions (e.g. the mean, the farthest point from the mean, etc.). Because of its linear nature in the Euclidean case, and the fact that the operations are always global (the same rotation/translation/scaling is applied to all points each time), the space of transformations that one can cover using Procrustes Analysis does not include all possible transformations between the statistics of datasets. Nevertheless, the results that we have obtained on BCI applications indicate that the set of transformations that we apply are rich enough to model the *covariate shift* between experimental sessions.

A particularity of our work is that we use the spatial covariance matrices (SCM) of the EEG signals as statistical features to discriminate between classes. SCMs are symmetric positive definite matrices (SPD) and as such they are defined in a Riemannian manifold [20] whose geometry is taken into consideration during the classification procedure. Such an approach has proved sucessful in BCI applications in recent years [21]–[23] and has become part of the state of the art of the field [2]. The geometric transformations that we use for matching two statistical distributions are done taking into account the intrinsic geometry of the SPD manifold as well. Because of its geometry-aware nature, we call our method Riemannian Procrustes Analysis (RPA).

RPA can be seen as an evolution of the aforementioned procedures [9] and [10], with the re-centering step corresponding to the first of a series of geometrical transformations. Furthermore, the procedures in [9] and [10] are completely unsupervised, since they do not use any information from the labels of the data points, whereas RPA benefits from the labels in the *source* session (which are all known in advance) as well as from (at least part of) the labels that become sequentially available in the *target* session trial after trial.

We compared the performance of the RPA procedure for Transfer Learning to that of other distribution-matching methods proposed in [9], [10] and [7]. The results demonstrate that the RPA yields a superior classification accuracy in both simulated and real datasets. We also observe that, in general, RPA needs a very small amount of labeled trials from the *target* dataset to work well. Moreover, it is always superior (or at least equal) to the performance obtained with calibration, i.e., simply using the available trials on the *target* dataset and not do any transfer from the *source*.

The remainder of the paper goes as follows: Section II gives a mathematical formulation for the paradigm of Transfer Learning that we consider in this work. Section III introduces the tools for manipulating and classifying SPD matrices in their Riemannian manifold, and Section IV presents the method of Riemannian Procrustes Analysis. Sections V and VI discuss the results on simulated and real data, respectively. Lastly, Section VII concludes the paper.

II. Problem formulation

We consider two datasets, the *source* ($\mathcal{S}$) and the *target* ($\mathcal{T}$) datasets. They are comprised of couples

$$\begin{aligned} \mathcal{S} &= \left\{ (C_i, y_i) \text{ for } i = 1, \dots, K_S \right\} , \\ \mathcal{T} &= \left\{ (\widetilde{C}_i, \widetilde{y}_i) \text{ for } i = 1, \dots, K_T \right\} , \end{aligned} \tag{1}$$

with $C_i, \widetilde{C}_i \in \mathbb{R}^{n \times n}$ being data points, and $y_i, \tilde{y}_i \in \{1, \dots, L\}$ their corresponding class labels; K_S and K_T are the number of trials in the *source* and *target* sessions respectively. In this paper, the data points in $\mathcal{S}$ and $\mathcal{T}$ are always symmetric positive definite (SPD) matrices and are used to parametrize EEG multivariate time series [22]. As mentioned before,

SPD matrices are defined in a Riemannian manifold, so it is important that operations involving them respect the intrinsic geometry of this space (see Section III for more information).

Our distribution matching method considers the *semi*-supervised Transfer Learning paradigm [4], where one has knowledge of all the labels from the *source* dataset and access to a small subset of labels from the *target* dataset. Put in mathematical terms, we assume knowledge of all the labels from the elements in $\mathcal{S}$ and of a small subset $\mathcal{T}_\ell \subset \mathcal{T}$ with

$$\mathcal{T} = \mathcal{T}_\ell \cup \mathcal{T}_u \quad \text{and} \quad \mathcal{T}_\ell \cap \mathcal{T}_u = \emptyset, \tag{2}$$

where ℓ stands for *labeled* and u for *unlabeled*. We further assume that $\mathcal{T}_\ell$ has at least one example from each class. This setup describes well applications where a few calibration points from the *target* dataset can be used to guide the Transfer Learning procedure. Another relevant case is online algorithms, where labels are available sequentially and augment the $\mathcal{T}_\ell$ dataset after each time step.

Our goal is to train a classifier that leverages the available information from $\mathcal{S}$ and $\mathcal{T}_\ell$ and has good accuracy on the classification of data points from $\mathcal{T}_u$.

III. Background and notation

This section begins with a brief introduction to concepts of Riemannian geometry on SPD matrices. Then, we define the notion of statistical distributions of SPD matrices and review a simple way to do classification on datasets containing this type of matrix.

A. The Symmetric Positive Definite manifold

Let $\mathcal{P}(n)$ be the set of $n \times n$ symmetric positive definite (SPD) matrices, which is defined as

$$\mathcal{P}(n) = \left\{ C \in \mathbb{R}^{n\times n} \;\middle|\; C^T = C,\ \boldsymbol{x}^T C \boldsymbol{x} > 0,\ \forall \boldsymbol{x} \in \mathbb{R}^n \right\}. \tag{3}$$

Matrices in $\mathcal{P}(n)$ lie in a manifold [20], a set of points with the property that the neighborhood of each $C \in \mathcal{P}(n)$ can be bijectively mapped to an Euclidean space, also known as its tangent space $T_C\mathcal{P}(n)$. In particular, because $\mathcal{P}(n)$ is an open subspace of the set $\mathrm{Sym}(n)$ of symmetric matrices in $\mathbb{R}^{n\times n}$, we can identify its tangent space as simply being $\mathrm{Sym}(n)$ [24]. If we endow every tangent space of a manifold with a metric that changes smoothly along its elements, we say that we have a Riemannian manifold [24]. In this case, fundamental geometric notions are naturally defined, such as geodesic (shortest curve joining two points), distance between two points (length of the geodesic connecting them), the center of mass of a set of points, etc.

There are several possible choices of metric for $\mathcal{P}(n)$ and each one induces a different geometry that can be more or less adequate according to the applications we are interested in. A metric that is particularly relevant is the Affine-Invariant Riemannian metric (AIRM) [20], defined for $\eta, \xi \in T_C\mathcal{P}(n)$ as the inner product

$$\langle \eta, \xi \rangle_C = \mathrm{tr}\big(C^{-1}\eta C^{-1}\xi\big)\ , \tag{4}$$

where $C \in \mathcal{P}(n)$ and $\mathrm{tr}(\cdot)$ denotes the trace operator. It is well known [20] that the distance between matrices $C_i, C_j \in \mathcal{P}(n)$ induced by (4) is

$$\delta_R^2(C_i, C_j) = \sum_{k=1}^{n} \log^2(\lambda_k)\ , \tag{5}$$

where the λ_k's are the eigenvalues of $C_i^{-1}C_j$ or of similar matrix $C_i^{-1/2}C_jC_i^{-1/2}$. Because of the many interesting properties of (5), such as invariance to affine transformations by any invertible matrix $A \in \mathbb{R}^{n\times n}$,

$$\delta_R^2(C_i, C_j) = \delta_R^2(AC_iA^T, AC_jA^T)\ , \tag{6}$$

invariance under inversion, etc. [20], the AIRM has found great popularity in geometry-aware algorithms for processing SPD matrices [21] [25]. From now on, whenever we refer to $\mathcal{P}(n)$ we will be implicitly considering it has been equipped with the AIRM.

The center of mass according to distance (5) of a set of SPD matrices $\{C_1, \ldots, C_K\}$ is defined as [1]

$$\mathcal{G}\left(\{C_i\}_{i=1}^K\right) = \underset{X\in\mathcal{P}(n)}{\mathrm{argmin}} \sum_{i=1}^{K} \delta_R^2(X, C_i)\ . \tag{7}$$

In words, $\mathcal{G}\left(\{C_i\}_{i=1}^K\right)$ is the point in the manifold minimizing the dispersion (variance) of the set of matrices. Note that when the C_k's are actually strictly positive scalars, $\mathcal{G}\left(\{C_i\}_{i=1}^K\right)$ is their geometric mean. This explains why many researchers adopt the term "geometric mean" to refer to the center of mass of a set of SPD matrices. For three or more matrices, there is no closed form solution for (7), so one has to resort to iterative algorithms as the one given in [26]. The above definitions suffice for the intents of this paper, but the interested reader will find a thorough treatment of the subject in [20].

B. Statistical distributions in the SPD manifold

In this work, we assume that the data points come from statistical distributions that can be parametrized just by their geometric mean and the dispersion of points around it. More precisely, we assume that the statistical distribution generating the samples of the dataset is a mixture of Riemannian Gaussian distributions on the SPD manifold (one for each class) [27].

Under this assumption, we parametrize the statistics of each dataset defined in (1) using a set consisting of $L+2$ elements : the geometric mean M of the dataset, the geometric means M_k of each of the L classes, and the dispersion d around M. We have then

$$\Theta_{\mathcal{S}} = \left\{M, M_1, \ldots, M_L,\ d\right\}\ ,$$
$$\Theta_{\mathcal{T}} = \left\{\widetilde{M}, \widetilde{M}_1, \ldots, \widetilde{M}_L,\ \tilde{d}\right\}\ ,$$

where

$$M = \mathcal{G}(\{C_i \mid C_i \in \mathcal{S}\})\ ,$$
$$\widetilde{M} = \mathcal{G}(\{\widetilde{C}_i \mid \widetilde{C}_i \in \mathcal{T}_\ell\})$$

are the center of mass or geometric means of the datasets (all classes combined),

$$d = \sum_{C_i \in \mathcal{S}} \delta_R^2(M, C_i) \ ,$$
$$\tilde{d} = \sum_{C_i \in \mathcal{T}_\ell} \delta_R^2(\widetilde{M}, C_i)$$

are the dispersions around the geometric mean, and

$$M_k = \mathcal{G}(\{C_i \mid C_i \in \mathcal{S} \text{ and } y_i = k\}) \ ,$$
$$\widetilde{M}_k = \mathcal{G}(\{\widetilde{C}_i \mid \widetilde{C}_i \in \mathcal{T}_\ell \text{ and } \widetilde{y}_i = k\})$$

are the geometric mean of the trials belonging to each class. Note that the parameters for the *target* dataset are estimated using only the data points from $\mathcal{T}_\ell$.

The law of great numbers from statistics also applies to datasets containing SPD matrices [20]. In particular, if the elements of a dataset come from a statistical distribution with geometric mean M, the center of mass of a set of K matrices will converge to M as K grows. This implies that in experimental paradigms where the trials come in sequentially, it is reasonable to expect that with more trials one will obtain better and better estimates of the geometric mean.

C. Classification in the SPD manifold

There are many known classifiers for datasets whose elements are SPD matrices [22]. In this paper, we use the Minimum-Distance to Mean (MDM) classifier [21], which works as follows: in the *training* phase, calculate the geometric means for each class of a training dataset ($\mathcal{D}_{\text{train}}$). In the *testing* phase, each matrix C_i from the testing dataset ($\mathcal{D}_{\text{test}}$) is associated to the label y_i of the class mean that is the closest to C_i. Note that the statistical model described in Section III-B fits well the MDM classifier, since it assumes that the statistics of the dataset can be described by the geometric means of its classes.

IV. Transfer Learning via Procrustes Analysis

This section presents the Riemannian version of the Procrustes Analysis or RPA. We first introduce the concept of Procrustes Analysis on an Euclidean setting and then describe how to perform equivalent transformations on the SPD manifold. Then, we justify such operations with the help of a model relating the statistical distributions of the *source* and *target* datasets. Such model is the main theoretical contribution of this paper, since it gives a concrete justification for the geometric operations done in the RPA procedure and allows for a better comprehension of the assumptions that one has to make regarding the statistical distributions of the datasets.

A. Matching statistical distributions

A common approach to match two shapes in Euclidean space is the Procrustes Analysis [17], which works as follows : suppose we have two sets of landmark points

$$\mathcal{X} = \left\{\boldsymbol{x}_i \in \mathbb{R}^n\right\}_{i=1}^m \text{ and } \widetilde{\mathcal{X}} = \left\{\widetilde{\boldsymbol{x}}_i \in \mathbb{R}^n\right\}_{i=1}^m \ , \qquad (8)$$

and assume there is a linear relationship relating the m pairs of landmark points as in

$$\widetilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{m}} = s\ U\big(\boldsymbol{x}_i - \boldsymbol{m}\big) \ , \qquad (9)$$

where $s \in \mathbb{R}$, $\boldsymbol{m}, \tilde{\boldsymbol{m}} \in \mathbb{R}^n$, and $U \in \mathbb{R}^{n \times n}$ is an orthogonal matrix. The goal of the procedure is to determine the values of $\{s, \boldsymbol{m}, \tilde{\boldsymbol{m}}, U\}$ so to obtain a new set $\widetilde{\mathcal{X}}^{(\text{PA})}$ containing points $\tilde{\boldsymbol{x}}_i^{(\text{PA})}$ that matches exactly with $\boldsymbol{x}_i$, where

$$\tilde{\boldsymbol{x}}_i^{(\text{PA})} - \boldsymbol{m} = \frac{1}{s} U^T \big(\widetilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{m}}\big) \ . \qquad (10)$$

Note that the operations transforming $\widetilde{\boldsymbol{x}}_i$ can be interpreted as a re-centering to zero (subtracting $\tilde{\boldsymbol{m}}$) followed by a stretching or compression (division by s), and a rotation (multiplication by U^T); the final re-centering to $\boldsymbol{m}$ is optional, since it is often more interesting to re-center $\mathcal{X}$ to the origin and consider only the zero-mean matched shapes.

B. Riemannian Procrustes analysis (RPA)

In order to do Procrustes Analysis on SPD matrices, we have to adapt the steps of re-centering, stretching and rotation according to the intrinsic geometry of $\mathcal{P}(n)$. We call such procedure *Riemannian Procrustes Analysis* (RPA) and describe its steps here below :

1) Re-center matrices to identity: In $\mathcal{P}(n)$ the Identity matrix plays the role of the origin of the space. Therefore, the first step of RPA is to transform the matrices in $\mathcal{S}$ and $\mathcal{T}$ so they are both centered around I_n (see Fig. 1B). This amounts to the transformation proposed in [9] if the covariance matrices used to describe the resting activity of each session were chosen to be the geometric mean of the trials of each dataset.

Due to the affine-invariance of (5) and (7), the geometric mean of the set of re-centered matrices

$$C_i^{(\text{rct})} = M^{-1/2} C_i M^{-1/2} \qquad (11)$$

is I_n. Moreover, since $\widetilde{M}$ is estimated from a subset of points in $\mathcal{T}_\ell \subset \mathcal{T}$, the geometric mean of the set of matrices

$$\widetilde{C}_i^{(\text{rct})} = \widetilde{M}^{-1/2} \widetilde{C}_i \widetilde{M}^{-1/2} \qquad (12)$$

is approximately the identity matrix (it tends to the identity as the number of elements in $\mathcal{T}_\ell$ grows).

We have then two new datasets consisting of re-centered matrices

$$\begin{aligned} \mathcal{S}^{(\text{rct})} &= \left\{(C_i^{(\text{rct})}, y_i) \text{ for } i = 1, \ldots, K_S\right\} \ , \\ \mathcal{T}^{(\text{rct})} &= \left\{(\widetilde{C}_i^{(\text{rct})}, \widetilde{y}_i) \text{ for } i = 1, \ldots, K_T\right\} \ , \end{aligned} \qquad (13)$$

with the indices of the partition $\mathcal{T}^{(\text{rct})} = \mathcal{T}_\ell^{(\text{rct})} \cup \mathcal{T}_u^{(\text{rct})}$ being the same as in (1).

2) Equalize the dispersions on each dataset: The next step of RPA consists of rescaling the distributions on both datasets so that their dispersions around the mean are the same (see Fig. 1C). To do so, we can see from (5) that

$$\delta_R^2\left(\left(\widetilde{C}_i^{(\text{rct})}\right)^s, I_n\right) = s^2\ \delta_R^2\left(\widetilde{C}_i^{(\text{rct})}, I_n\right) \ , \qquad (14)$$

which implies that one can modulate the dispersion of $\mathcal{T}^{(\text{rct})}$ by simply moving each of its matrices along the geodesic that links it to the identity matrix. Note that the parameter s plays the same role of the scaling factor in (9). We match the dispersions from *source* and *target* by building a new dataset $\mathcal{T}^{(\text{str})}$ containing the stretched matrices

$$\widetilde{C}_i^{(\text{str})} = \left(\widetilde{C}_i^{(\text{rct})}\right)^s \ , \tag{15}$$

where we require $s \in \mathbb{R}$ to verify

$$s^2 = d/\widetilde{d} \ . \tag{16}$$

Note that the re-centering of matrices in Step 1 does not alter the dispersion of the matrices around their geometric mean, which means that the stretching step could have been done before re-centering the matrices in $\mathcal{T}$. However, in this case, the geodesic move in (15) would have to be done with respect to $\widetilde{M}$, that is, we would have to use a more involved relation

$$\widetilde{C}_i^{(\text{str})} = \widetilde{M}^{1/2}\left(\widetilde{M}^{-1/2}\widetilde{C}_i\widetilde{M}^{-1/2}\right)^s\widetilde{M}^{1/2} \ . \tag{17}$$

Note that up to this point no information from the trials' classes has been used. We say then that the *recentering* and *stretching* operations form the *unsupervised* part of the RPA method.

3) Rotate around the geometric mean: The last step consists of rotating the matrices from $\mathcal{T}^{(\text{str})}$ around the origin and matching the orientation of its point cloud with that of $\mathcal{S}^{(\text{rct})}$ (see Fig. 1D). To do so, we note that if U is an orthogonal matrix, then

$$\delta_R^2(U^T\widetilde{C}_i^{(\text{str})}U,\ U^TU) \ = \ \delta_R^2(\widetilde{C}_i^{(\text{str})},\ I_n) \ , \tag{18}$$

which indicates that the effect of an orthogonal matrix over a set of matrices centered at the identity is that of a rotation around their mean. We form a new dataset $\mathcal{T}^{(\text{rot})}$ containing rotated matrices with

$$\widetilde{C}_i^{(\text{rot})} = U^T\ \widetilde{C}_i^{(\text{str})}\ U \ , \tag{19}$$

where U is an orthogonal matrix to be determined from the data. As we will see in the next sub-section, matrix U is determined using the labels from the trials, so it corresponds to the *supervised* part of the procedure.

C. The orthogonal matrix U

The procedure to determine matrix U comes up naturally once the assumptions of the RPA method are written in mathematical form. For simplicity, we will first assume that $\mathcal{T}_\ell = \mathcal{T}$. We use the geometric means of the *source* and *target* datasets as landmarks to be matched, so one can write

$$\widetilde{M} = A\ M\ A^T \ , \tag{20}$$

and

$$\widetilde{M}_k = A\ M_k\ A^T \quad k \in \{1,\dots,L\} \ , \tag{21}$$

where $M, \widetilde{M}, M_k, \widetilde{M}_k$ are all defined in Section III-B, and $A \in \mathbb{R}^{n\times n}$ is an unknown invertible matrix that models the discrepancies between the statistics of the *source* and *target* datasets. We can rewrite the relation in (20) as

$$\left(\widetilde{M}^{1/2}\widetilde{M}^{1/2}\right) = A\ \left(M^{1/2}M^{1/2}\right)\ A^T \ , \tag{22}$$

$$I_n = \widetilde{M}^{-1/2}A\ \left(M^{1/2}M^{1/2}\right)\ A^T\widetilde{M}^{-1/2} \ , \tag{23}$$

$$I_n = \left(\widetilde{M}^{-1/2}AM^{1/2}\right)\left(\widetilde{M}^{-1/2}AM^{1/2}\right)^T \ , \tag{24}$$

$$UU^T = \left(\widetilde{M}^{-1/2}AM^{1/2}\right)\left(\widetilde{M}^{-1/2}AM^{1/2}\right)^T \ , \tag{25}$$

where U is the $n \times n$ orthogonal matrix that we want to determine. Matrix U can then be simply written as

$$U = \widetilde{M}^{-1/2}AM^{1/2} \ , \tag{26}$$

where M and $\widetilde{M}$ are directly estimated from the data points as explained in Section III-A, and A remains unknown. To determine an expression for U only in terms of variables that can be estimated from the data, we use (26) in (21) to get

$$\widetilde{M}_k = \left(\widetilde{M}^{1/2}UM^{-1/2}\right)\ M_k\ \left(\widetilde{M}^{1/2}UM^{-1/2}\right)^T, \tag{27}$$

$$\widetilde{M}^{-1/2}\widetilde{M}_k\widetilde{M}^{-1/2} = U\ M^{-1/2}M_kM^{-1/2}\ U^T \ . \tag{28}$$

Defining the matrices

$$G_k = M^{-1/2}M_kM^{-1/2} \tag{29}$$

and

$$\widetilde{G}_k = \widetilde{M}^{-1/2}\widetilde{M}_k\widetilde{M}^{-1/2} \ , \tag{30}$$

and using the expression in (20) for $\widetilde{M}$, one can rewrite (28) as

$$\widetilde{G}_k = \tilde{M}^{1/2}A^{-T}M^{-1/2}G_kM^{1/2}A^T\tilde{M}^{-1/2} \tag{31}$$

and conclude that G_k and $\widetilde{G}_k$ are related via a similarity transform. Similar matrices having the same eigenvalues, one can write the eigendecompositions

$$G_k = Q_k\Lambda Q_k^T \quad \text{and} \quad \widetilde{G}_k = \widetilde{Q}_k\Lambda\widetilde{Q}_k^T$$

so that (28) becomes

$$\widetilde{Q}_k\Lambda\widetilde{Q}_k^T = U\ Q_k\Lambda Q_k^T\ U^T. \tag{32}$$

Solving (32) for U we obtain

$$U = \widetilde{Q}_kQ_k^T \ , \tag{33}$$

which is ultimately an expression in terms of variables estimated from the dataset. Note that (33) is also the solution to the following optimization problem

$$\underset{U^TU=I_n}{\text{minimize}} \quad \delta_R^2\left(\widetilde{G}_k,\ UG_kU^T\right) \ , \tag{34}$$

and so it can be interpreted as the orthogonal matrix that acts to minimize the distance between a modified version of the class means of the *source* and *target* datasets.

D. Determining U from data

In the previous section, we assumed $\mathcal{T}_\ell = \mathcal{T}$. In practice, however, we have $\mathcal{T}_\ell \subset \mathcal{T}$, so the estimation of the class means of the *target* dataset are only approximations of the real class means of the statistical distribution. Because of this, instead of giving preference to a particular noisy estimate of a class mean to determine U via (33), we obtain it as a solution to the following optimization problem on the manifold of orthogonal matrices :

$$\underset{U^TU=I_n}{\text{minimize}} \quad \sum_{k=1}^{L} w_k \delta_R^2 \left(\widetilde{G}_k,\ U G_k U^T \right) \ , \tag{35}$$

where the $w_k \in [0,1]$ are coefficients allowing to balance the optimization according to the quality of the estimators for the mean of each class. Note that problem (35) is a generalization of the Procrustes problem in the SPD manifold proposed in [28]. We solve (35) using a special form of the steepest-descent algorithm adapted for optimization procedures on manifolds, as described in [24]. To do so, we first rewrite each term of the cost function in (35) as

$$f_k(U) = \delta_R^2 \left(\widetilde{G}_k,\ U G_k U^T \right) \tag{36}$$

and express its Jacobian as

$$D_U \mathcal{L}(U) = \sum_{k=1}^{L} w_k \ D_U f_k(U) \ , \tag{37}$$

with

$$D_U f_k(U) = 4 \log \left(\widetilde{G}_k U G_k U^T \right) U \ , \tag{38}$$

where the derivative of the AIRM distance was obtained from [29]. On each iteration of the gradient descent procedure, the vector $D_U f_k(U)$ is projected onto the tangent space of the manifold of orthogonal matrices (see [24] for details). We used the `pymanopt` package [30] for carrying out the optimization procedure.

E. Classification on the transformed datasets

Once the datasets have been transformed with RPA, we use the MDM classifier (see Section III-C) to infer the unknown labels from the elements in $\mathcal{T}_u$. In the *training* phase we have

$$\mathcal{D}_{\text{train}} = \mathcal{S}^{(\text{rct})} \cup \left\{ (\widetilde{C}_i^{(\text{rot})}, \widetilde{y}_i) \ \middle| \ (\widetilde{C}_i, \widetilde{y}_i) \in \mathcal{T}_\ell \right\} \ , \tag{39}$$

and in the *testing* phase we infer the labels from

$$\mathcal{D}_{\text{test}} = \left\{ (\widetilde{C}_i^{(\text{rot})}, \tilde{y}_i) \ \middle| \ (\widetilde{C}_i, \tilde{y}_i) \in \mathcal{T}_u \right\} \ . \tag{40}$$

F. Summary of the RPA method

Algorithm 1 recapitulates the steps of a classification task using RPA for matching the statistical distributions of the *source* and *target* datasets.

Algorithm 1: Transfer Learning via RPA

Input: $\mathcal{S}$, $\mathcal{T}_\ell$ and $\mathcal{T}_u$ as defined in (1) and (2)
Output: accuracy of classification using MDM on $\mathcal{T}_u$

1. Estimate M and $\widetilde{M}$ from the data in $\mathcal{S}$ and $\mathcal{T}_\ell$
2. Re-center the matrices in $\mathcal{S}$ and $\mathcal{T}$ using (11) and (12), and form new datasets
$$\mathcal{S}^{(\text{rct})} \text{ and } \mathcal{T}^{(\text{rct})} = \mathcal{T}_\ell^{(\text{rct})} \cup \mathcal{T}_u^{(\text{rct})}$$
3. Calculate the ratio of dispersions in $\mathcal{S}^{(\text{rct})}$ and $\mathcal{T}_\ell^{(\text{rct})}$ as in (16) and use it to form the new dataset
$$\mathcal{T}^{(\text{str})} = \mathcal{T}_\ell^{(\text{str})} \cup \mathcal{T}_u^{(\text{str})}$$
with matrices as described in (15)
4. Estimate matrices M_k and $\widetilde{M}_k$ for $k \in \{1, \ldots, L\}$ and obtain the orthogonal matrix U as a solution from (35)
5. Rotate the matrices from $\mathcal{T}^{(\text{str})}$ as in (19) and obtain
$$\mathcal{T}^{(\text{rot})} = \mathcal{T}_\ell^{(\text{rot})} \cup \mathcal{T}_u^{(\text{rot})}$$
6. Form the training dataset for the MDM classifier with
$$\mathcal{D}_{\text{train}} = \mathcal{S}^{(\text{rct})} \cup \mathcal{T}_\ell^{(\text{rot})}$$
and get the accuracy of classification on the data points from the test dataset
$$\mathcal{D}_{\text{test}} = \mathcal{T}_u^{(\text{rot})}$$

G. An interpretation of the steps in RPA

We give now an interpretation of the steps of RPA in terms of the statistical distributions of the datasets. Without loss of generality, we will consider that $d = \widetilde{d}$, since the dispersions can always be made equal prior to the transformations. We will also assume $\mathcal{T}_\ell = \mathcal{T}$ for simplicity.

The relations in (20) and (21) define which landmark data points we should match in the RPA procedure, an approach that is justified from the fact that we parametrize the statistics of $\mathcal{S}$ and $\mathcal{T}$ using their geometric means, as described in Section III-B. From this observation, one can also conclude that a simple approach for matching the statistical distributions of $\mathcal{S}$ and $\mathcal{T}$ would be to estimate matrix A from the available data points and apply A^{-1} to all the elements of $\mathcal{T}$, as in

$$\widetilde{C}_i \mapsto A^{-1} \widetilde{C}_i A^{-T} \ . \tag{41}$$

From (26) we can write

$$A = \widetilde{M}^{1/2} U M^{-1/2} \ , \tag{42}$$

where M and $\widetilde{M}$ are estimated directly from the dataset, and the orthogonal matrix U is determined as discussed in Section IV-C. Applying A^{-1} to the matrices in $\mathcal{T}$, we get

$$A^{-1} \widetilde{C}_i A^{-T} = M^{1/2} \left[U^T \left(\widetilde{M}^{-1/2} \widetilde{C}_i \widetilde{M}^{-1/2} \right) U \right] M^{1/2} \ , \tag{43}$$

which describes the same steps of RPA: re-center to identity, stretch (with $s = 1$) and rotate, followed by a translation of the mean back to M. From the expressions above, we see that the sequence of operations in RPA are nicely justified by the assumptions of our statistical model for the data points.

V. Numerical Illustrations : Simulated data

A. The dataset

Our example on simulated data consists of a *source* and a *target* datasets containing 2×2 SPD matrices and belonging to two classes. Data points from the *source* dataset are generated as follows:

1) Generate a random SPD matrix $M_1 \in \mathcal{P}(2)$ and define it to be the geometric mean of class 1
2) Generate $N_t = 100$ random SPD matrices around M_1 by mapping small random tangent vectors (norm fixed to $\varepsilon = 10$) from $T_{M_1}\mathcal{P}(2)$ back to the SPD manifold. We associate to each of these matrices the label $y_i = 1$
3) Generate a random SPD matrix M_2 whose distance to M_1 is $s = 5$. This is the geometric mean for class 2
4) Generate $N_t = 100$ random SPD matrices around M_2 by mapping small random tangent vectors (norm fixed to $\varepsilon = 10$) from $T_{M_2}\mathcal{P}(2)$ back to the SPD manifold. These matrices have a label $y_i = 2$ associated to them

We generate the data points from the *target* dataset ($\mathcal{T}$) exactly as for the *source* dataset, but add an extra translation step that ensures that the geometric mean of all the matrices from $\mathcal{S}$ will be at a distance $d = 8$ from the geometric mean of $\mathcal{T}$.

B. Visualization of the steps of RPA

We use the algorithm of Diffusion maps [31] to obtain new representations of our data points using only two axis. This nonlinear dimensionality reduction algorithm works by first building a matrix with all pairwise distances between data points. Then, it calculates a new matrix called Laplacian which contains important information on the geometry of the low-dimensional manifold where the data points are assumed to live. The axis for the new data representations are then obtained from the spectral decomposition of the Laplacian matrix (see [32] for more details on the algorithm and [33] for an application on multivariate time series).

Figure 1 illustrates the distribution of data points after each step of RPA applied to the *source* and *target* datasets. In this example, we consider that we know the labels of all matrices from the *target* dataset, i.e., $\mathcal{T} = \mathcal{T}_\ell$. Figure 1A shows the point clouds of each dataset, which are clearly unmatched. After recentering (Fig. 1B), stretching (Fig. 1C) and rotating (Fig. 1D), the statistical distributions get matched and the same classifier can be used on both datasets.

C. Classification accuracy after RPA

We compare the classification accuracy results on the simulated dataset for six different methods of Transfer Learning. In each of them, the training and testing datasets are different but the classifier is always the MDM :

- **direct (DCT)**: direct use of the points from the *source* dataset to do classification on the *target* dataset (no transformation whatsoever).
- **recentering (RCT)**: transfer learning considering only the data points of each dataset recentered to I_n. This corresponds to step (1) in the RPA procedure and is similar to what has been done in [9].
- **parallel transport (PRL)**: transfer learning using the method proposed in [10]. The procedure is analogous to **RCT**, but with the points being re-centered to the halfway point along the geodesic path linking the geometric means of each dataset instead of the Identity matrix.
- **optimal transport (OPT)**: transfer learning using the optimal transport approach proposed in [7] and adapted to take into account the fact that we have data points defined in the SPD manifold instead of Euclidean vectors.
- **RPA**: transfer learning with matrices transformed using RPA, as described in Section IV-B.
- **calibration**: classification using only the labeled trials available in the *target* dataset, with no help from the data in the *source* dataset.

We assess the performance of each method via a randomized cross-validation procedure consisting of:

1) Select $2n$ random elements from $\mathcal{T}$ (n from each class). These data points define $\mathcal{T}_\ell$
2) Define the test dataset $\mathcal{T}_u$ containing the other $200 - 2n$ elements of $\mathcal{T}$
3) Obtain the accuracy of the classification via MDM for this particular partition of $\mathcal{T}$
4) Repeat the three preceeding steps 10 times and get the mean accuracy for each method.

The results in Fig. 2 show that the **DCT** pipeline gives classification results at chance level (0.5) independently of the number of matrices available in $\mathcal{T}_\ell$. We also observe that simply using **RCT** already greatly improves classification accuracy, as reported in [9]. Our **RPA** method further improves the results. We also observe that **PRL** has virtually the same performance as **RCT**, which is not surprising, since they are both unsupervised methods based on the idea of re-centering the datasets to a common point in the SPD manifold. The results with **OPT** are equivalent to **RCT** and **PRL** as well. The accuracy with **calibration** improves when the number of available labels in the *target* dataset increases, eventually converging to the same performance as **RPA**. This result is not surprising, because with a sufficient amount of data in $\mathcal{T}_\ell$ it is already possible to train a good classifier without the need of doing transfer learning.

Our observations in this session are in accordance with the theoretical results of [34], which says that "if there is enough target data, then no source data are needed (...). This is because the possible reduction in error due to additional source data is always less than the increase in error caused by the source data being too far from the target data". Such a result points to the existence of a certain "saturation" effect in the quality of transfer learning when too many trials are available in the *target* session, a behavior that could be exploited to decide when to stop transferring information from previous experimental sessions.

VI. Numerical Illustrations : Real data

The analysis on real data were all done using EEG recordings from experiments with Brain-Computer Interfaces (BCI)

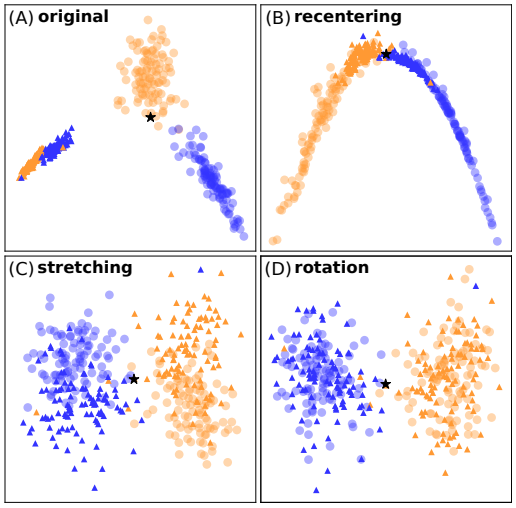

Fig. 1. Representation of the sequence of operations of RPA applied to a dataset simulated as described in Section V-A (better visualized with colors). Each point on the scatter plot represents a SPD matrix and the axes for the figures were obtained using Diffusion Maps [32]. The filled dots (degree of transparency set to $\alpha = 0.30$) represent the *target* dataset whereas the circles are the *source* dataset. Each color represents a class and the black star is the Identity matrix. (A) Distribution of the SPD matrices in the *source* and *target* datasets as they are originally available and (B) after re-centering their geometric means to the Identity. In (C) the distribution after the stretching operation and (D) after the rotation.

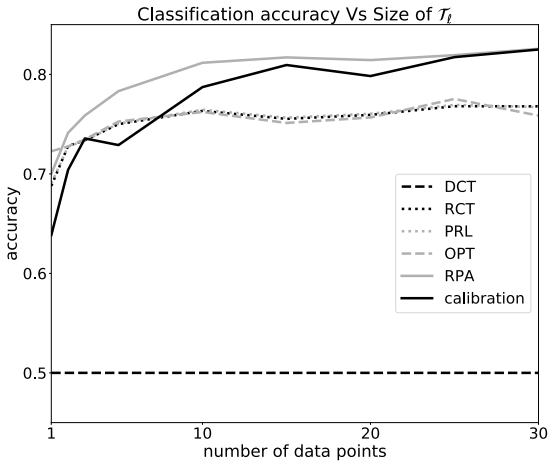

Fig. 2. Accuracy of the classification of unlabeled data points from the *target* dataset for different methods of transfer learning. The curve shows how the accuracy for each method evolves when the number of data points in $\mathcal{T}_\ell$ increases. The generation of the data points is explained in Section V-A.

experiments. As usual, the EEG signal is a n-dimensional multivariate time-series and is denoted by $\boldsymbol{x}(t)$ where each dimension represents an electrode. Each experimental trial i lasts a few seconds and is associated to a matrix $X_i \in \mathbb{R}^{n \times T}$, where T is the number of time samples defining the trial. To every trial we associate a SPD matrix C_i describing its multivariate statistics and a label y_i indicating what was the task performed during the trial. The dataset for each subject is composed of a set of couples (C_i, y_i). Our investigation focuses on the classification accuracy of a MDM classifier that is trained with the data from a *source* subject and is used to classify the signals from a *target* subject. We compare

TABLE I
MAIN FEATURES DESCRIBING EACH DATASET USED IN THIS WORK.

dataset	paradigm	subjects	classes	trials per class	reference
PhysionetMI	MI	109	2	22	[38]
Cho2017	MI	50	2	100	[39]
SSVEP	SSVEP	12	3	8	[37]
P300	P300	24	2	72 and 360	[36]
BNCI2014001	MI	9	4	72	[40]
BNCI2014002	MI	15	2	80	[41]
BNCI2015001	MI	13	2	100	[42]
MunichMI	MI	11	2	150	[43]

the performance of such classifier using the different Transfer Learning strategies described in Section V-C.

A. The datasets

The investigations were carried out on eight datasets covering three different BCI paradigms. All Motor Imagery datasets are publicly available and were downloaded and pre-processed using the `MOABB` framework [35]. The P300 dataset comes from experiments performed in our laboratory on the P300-based game Brain Invaders [36]. The SSVEP dataset was the same as the one presented in [37]. See Table I for a brief overview of each dataset's features.

We estimated the SPD matrices for each BCI paradigm differently. For the MI datasets, the SPD matrices were the spatial covariance matrices of the multivariate EEG recordings. The signals of each trial in the SSVEP paradigm were filtered using bandpass filters around certain frequencies of interest and its SPD matrices were diagonal blocks concatenating the spatial covariance matrices of the filtered signals [1]. For the P300, the SPD matrix of each trial was obtained using the approach from [44], where one estimates a special form of covariance matrix that captures the influence of event-related potentials in each trial.

B. Comparing cross-subject classification accuracies

In this section, we compare the accuracy of the classification of trials for each pair of *target* and *source* subject. The classification is done using the MDM classifier and the values of the accuracies are assessed using the same cross-validation scheme explained in Section V-C.

We begin with a qualitative analysis of the cross-subject classification accuracy using a tool from combinatorial data analysis called *seriation* [45]. This procedure sorts the lines and columns of a data matrix in order to make relevant patterns appear. In our case, the matrix S to be re-ordered contains at its (i, j) coordinate the accuracy of the classification using subject i as *target* and subject j as *source*. The rows of S are then sorted in decreasing order of columns sum, and the columns of this row-sorted-matrix S are rearranged in decreasing order of their rows sum. The output of this procedure is a new representation where the pairs of *source-target* subjects with the best accuracy are located at the top-left region of the matrix, while the worst pairs are at the bottom-right region. Figure 3 shows the results of this seriation procedure on the PhysionetMI dataset for two sizes of $\mathcal{T}_\ell$ (the number of labeled

target trials) and three different pipelines: **DCT**, **RCT**, and **RPA**. We observe that with **RCT** and **RPA** there are more pairs of subjects with high values of cross-subject classification than with **DCT**. In particular, for **RCT** and **RPA** we note that there are a few *target* subjects that have very good accuracy on classification for almost all possible *source* subjects, a feature that is possibly related to the performance of each *target* subject to classify its own trials (intra-subject accuracy). This can been interpreted as : subjects that are "good" for classifying their own data should be "good" for receiving information from other *source* subjects. We also observe a clear improvement in the average value of the cross-subject classification accuracies when more points are available in $\mathcal{T}_\ell$.

Our next analysis consisted in calculating the mean over all cross-subject AUC (Area Under the ROC Curve) for each Transfer Learning pipeline on each dataset. We used these values as quantitative measures for assessing whether one pipeline is better than the other on cross-subject classification. The scores are shown in Table II. We should mention that only the subjects whose intra-score AUC (i.e., classification of its own data) was above chance level were used in these calculations.

Figure 4 shows the results of statistical tests performed on each pair of methods, allowing for a more substantiated assessment of the performance of the methods. The statistical tests comparing method **A** versus method **B** were carried out in the following way: (1) For each *source* subject j, we perform a signed paired t-test comparing the scores of method **A** to method **B** along all *target* subjects. Each of these tests yield a statistic T_j and a p-value p_j is obtained via permutations tests [46]. (2) We combine the p-values of all the *source* subjects using Stouffer's Z-score method [47]. This yields a single p-value for the comparison between methods as well as the direction to which the null hypothesis has been rejected (i.e., whether method **A** is better than **B** or vice-versa). (3) We adjust p-values of each pairwise comparison using Holm's step-down procedure [48] to account for the multiple comparison problem.

The results in Figure 4 indicate that when there are enough points in $\mathcal{T}_\ell$ ("enough" depending on each dataset), transforming the data points with **RCT**, **PRL** or **RPA** is always better than not doing any distribution matching (**DCT**). We also observe that most of the time there is no statistical significance between the results with **PRL** and **RCT**, as expected, since they both amount to re-centering the datasets to a new point in the SPD manifold. For increasing values of N (the number of labeled trials in the *target* dataset), **RPA** gets better in comparison to almost all other methods, as expected and observed in Figure 2 for simulated data. Interestingly, **OPT** has very poor results in comparison to all other methods, probably because it does not use any prior hypothesis on the statistical distributions of the datasets and has to solve a difficult optimization problem to determine its transportation plan. Lastly, during our statistical analysis of the results, we have observed that better results on Transfer Learning via RPA are often associated to good intra-subject accuracy, since in this case the estimation of the class means is more stable and thus the rotation matrix U is better estimated. This explains why for some databases (e.g. PhysionetMI) the RPA is not necessarily the best method for Transfer Learning and an unsupervised approach like **RCT** has better results.

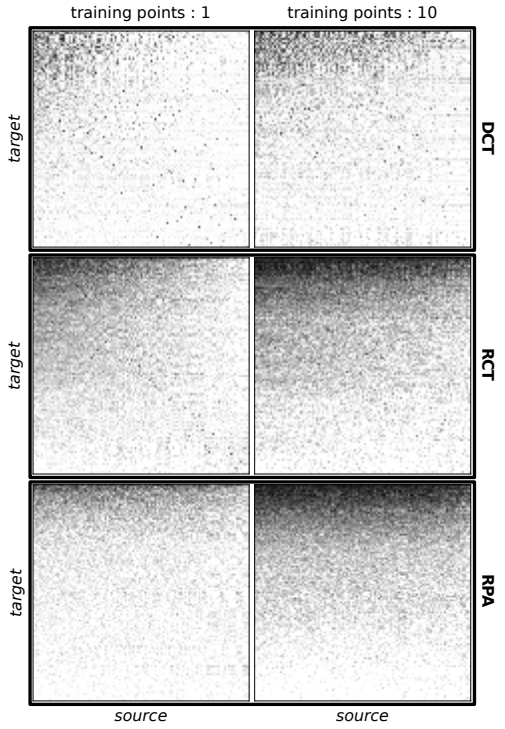

Fig. 3. Accuracies of the cross-subject classification for three different Transfer Learning procedures on the PhysionetMI database. The rows and columns of each subplot were reordered using the seriation procedure explained in the text. The colormap shows white for accuracies of 0.5 or less and black when it is 1.0. The compared methods are described in Section V-C and we consider the cases when there are one and ten labeled matrices in $\mathcal{T}_\ell$.

C. The role of the size of $\mathcal{T}_\ell$

As pointed out in the simulation results from Section V-C, when the size of $\mathcal{T}_\ell$ increases, using Transfer Learning is no longer relevant, since one may already have enough data to build a good classifier for the *target* subject. To investigate this behavior on our real datasets, we compared the cross-subject classification accuracy of **RPA** to that of the **calibration** method (see Section V-C for details).

Figure 5 shows a scatter plot comparing the classification accuracies on the MunichMI dataset. We see that when $\mathcal{T}_\ell$ grows, there are more pairs of subjects for which the classification using the **calibration** method on the *target* subject is better than doing transfer learning via **RPA** (28% to 34% of all the pairs of subjects). However, the location of the cloud of points in the figure indicates that the Transfer Learning with RPA is still superior to the Calibration method for most pairs of subjects. We used a one-sided paired *t*-test with random data permutation [46] to compare the accuracies of **RPA** and **calibration** on each dataset for different sizes of the $\mathcal{T}_\ell$. The null hypothesis of equivalency between the two methods was rejected ($p < 0.01$) on almost all tests, the only exception being for those on the BNCI2014001 dataset. Moreover, for

TABLE II
MEAN VALUES OF THE CROSS-SUBJECT AUC (AREA UNDER THE ROC CURVE) FOR FIVE PIPELINES (ALL DESCRIBED IN SECTION V-C) ON EIGHT DIFFERENT DATASETS. PARAMETER N IS THE NUMBER OF TRAINING POINTS AVAILABLE ON THE *target* DATASET ON EACH SITUATION. THE BEST METHOD IN EACH INSTANCE IS WRITTEN IN **BOLD**.

PhysionetMI

MEAN AUC					
N	DCT	RCT	PRL	OPT	RPA
1	0.54	**0.61**	**0.61**	0.59	0.56
5	0.55	**0.65**	**0.65**	0.60	0.63
10	0.56	**0.67**	**0.67**	0.60	0.66
15	0.57	**0.68**	**0.68**	0.60	0.67

Cho2017

MEAN AUC					
N	DCT	RCT	PRL	OPT	RPA
1	0.54	**0.59**	0.58	0.57	0.54
5	0.55	**0.61**	**0.61**	0.57	0.59
10	0.55	**0.62**	**0.62**	0.57	**0.62**
25	0.57	0.64	0.64	0.58	**0.66**

BNCI2014001

MEAN AUC					
N	DCT	RCT	PRL	OPT	RPA
1	0.58	**0.69**	**0.69**	0.65	0.62
6	0.59	**0.73**	**0.73**	0.65	0.71
18	0.61	**0.76**	**0.76**	0.65	**0.76**
36	0.64	0.78	0.77	0.66	**0.79**

BNCI2014002

MEAN AUC					
N	DCT	RCT	PRL	OPT	RPA
1	0.55	**0.68**	0.67	0.64	0.63
5	0.56	**0.71**	0.70	0.64	0.70
10	0.57	**0.72**	0.71	0.65	**0.72**
20	0.58	**0.73**	0.72	0.65	**0.73**

SSVEP

MEAN AUC					
N	DCT	RCT	PRL	OPT	RPA
1	0.64	0.67	0.66	0.59	**0.70**
2	0.67	0.71	0.71	0.59	**0.75**
4	0.72	0.76	0.76	0.59	**0.80**
6	0.74	0.78	0.78	0.57	**0.82**

P300

MEAN AUC					
N	DCT	RCT	PRL	OPT	RPA
6	0.57	0.56	0.56	**0.58**	0.55
12	0.62	**0.64**	**0.64**	0.61	0.63
32	0.71	**0.74**	**0.74**	0.67	0.73
48	0.74	**0.76**	**0.76**	0.69	0.75

BNCI2015001

MEAN AUC					
N	DCT	RCT	PRL	OPT	RPA
1	0.51	0.56	0.55	0.54	**0.57**
5	0.51	0.57	0.57	0.55	**0.60**
10	0.52	0.59	0.58	0.56	**0.63**
25	0.54	0.62	0.62	0.56	**0.65**

MunichMI

MEAN AUC					
N	DCT	RCT	PRL	OPT	RPA
1	0.55	**0.63**	**0.63**	0.61	0.55
25	0.58	**0.69**	**0.69**	0.62	0.68
50	0.60	0.71	0.71	0.62	**0.72**
75	0.62	0.72	0.72	0.62	**0.73**

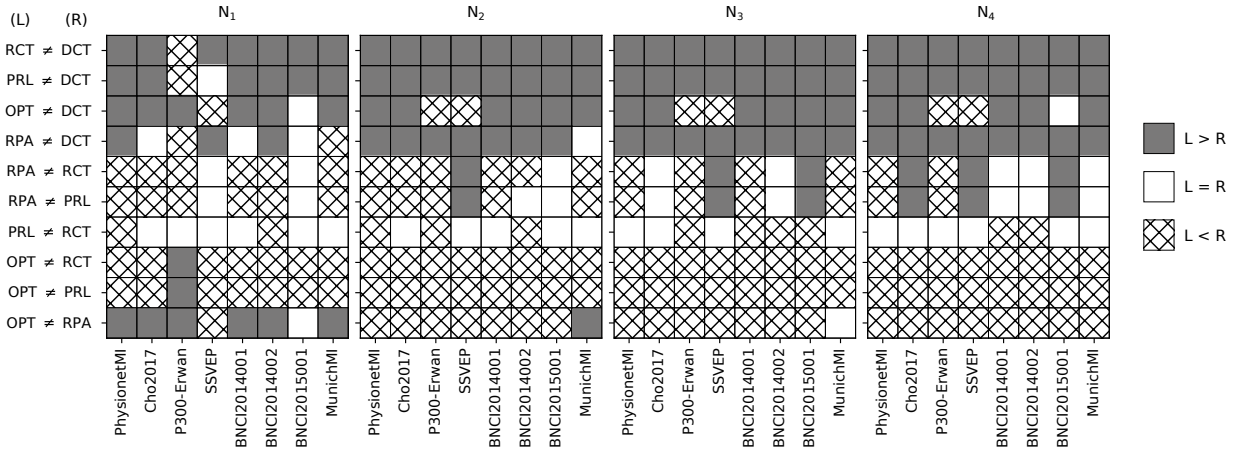

Fig. 4. Results of the statistical tests on each pair of pipelines for all possible values of N on each dataset as indicated in Table II (for instance, on *Cho2017* we have $N_1 = 1, N_2 = 5, N_3 = 10$, and $N_4 = 25$). The color/pattern of the squares indicate whether there's no statistical difference between two methods (white squares), if the Left method is superior to the Right one (L and R in the legend) (dark gray squares) or the contrary (squares with crossed patterns). All conclusions are with $p < 0.05$ corrected via Holm's adjustement [48]. For instance, we see that for dataset *Cho2017*, the method **RPA** is inferior to **RCT** when $N = 1$, but **RPA** becomes superior when $N = 25$. Furthermore, for this same dataset, there's no statistical difference in the comparison between **PRL** and **RCT** for any N.

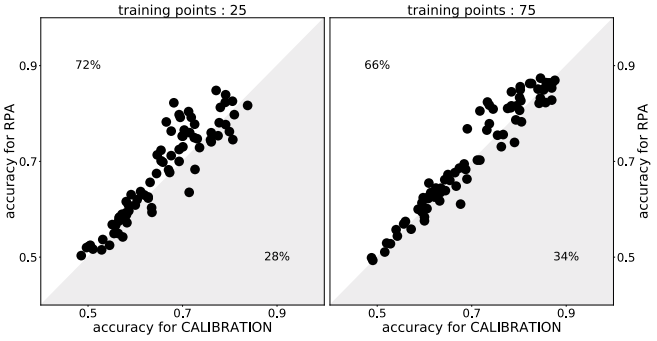

Fig. 5. Scatter plots comparing the accuracies of the cross-subject classification on the MunichMI dataset for the RPA and Calibration methods. We consider two sizes for $\mathcal{T}_\ell$. The percentage numbers indicate the proportion of dots above or below the diagonal line.

the tests where $\mathcal{H}_0$ was rejected, we observed a superiority of **RPA** in comparison to **calibration**.

D. Combining information from multiple subjects

We investigated how the matching of statistical distributions via RPA affects the performance of two baseline methods for gathering information from the data of multiple subjects: pooling and ensembling. The classification for each *target* subject is done using information coming from all other *source* subjects available in the database. Following the same approach as in previous sections, we only considered *source* subjects featuring an intra-subject accuracy above chance level, i.e., subjects in which it is meaningful to use transfer learning. The experiments were done on the *PhysionetMI* dataset with $|\mathcal{T}_\ell| = 15$ labeled trials available for each *target* subject and the *Cho2017* dataset with $|\mathcal{T}_\ell| = 25$.

The pooling strategy consists of gathering for each *target* subject the data from all other *source* subjects into one big dataset. Then, a classifier is trained on the pooled dataset and used to infer the trials from the *target* subject. We compared the performance of a MDM classifier when the *source* subjects were pooled with no transformation (**DCT**) to when the statistical distributions of each *source* subject were matched to that from the *target* subject using **RCT** or **RPA** (**PRL** is not fit for *pooling*, since the matrices are not all recentered to the same place in the SPD manifold). The boxplots in Figure 6 show the distributions of the classification scores of each of the *target* subjects. Using pairwise one-sided paired t-test with random permutations, the null hypothesis of equivalency between the scores of **DCT**, **RCT**, and **RPA** were all rejected

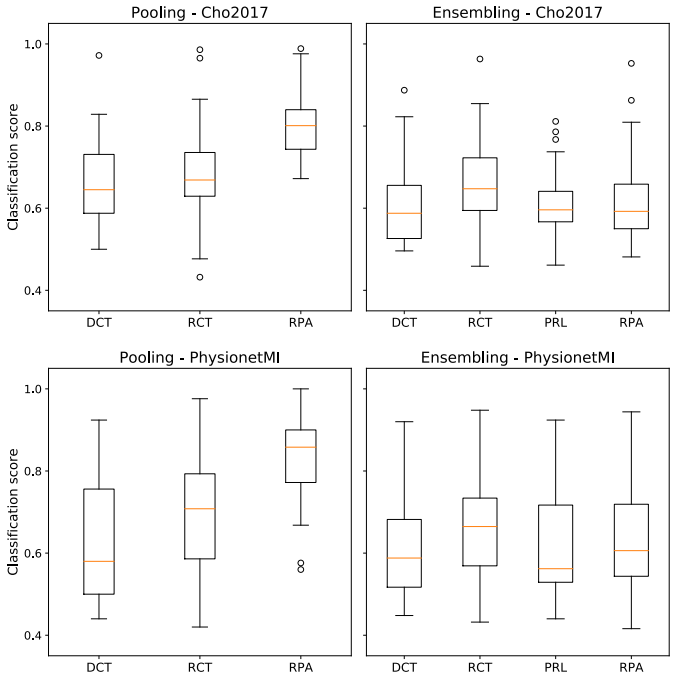

Fig. 6. Box plots with the distribution of the classification scores for the ensembling and pooling strategies for different methods of statistical matching between datasets. For the *Cho2017* dataset we had 25 labeled trials in the *target* dataset and for the *PhysionetMI* there were 15 labeled trials in it.

with $p < 10^{-6}$ (adjusted for multiple comparisons). The results show a clear improvement in the average score for the *pooling* strategy when using a method for matching the statistics of the *source* and *target* datasets, with differences of at least 15% between **RPA** and **DCT** for both datasets.

Our second analysis considered an ensembling strategy, where the trials of each *target* subject were classified using a majority voting scheme. These votes came from MDM classifiers trained on all other *source* subjects and were weighted equally. The results in Figure 6 show the scores with each method (including the **PRL** approach this time). To compare the scores of each method, we used pairwise one-sided paired t-tests with random permutations (corrected for multiple comparisons). The results of the statistical tests indicate that the ensembling strategy with **RPA** is superior as compared to **DCT** in the PhysionetMI dataset ($p < 0.05$) but they are equivalent for the Cho2017 dataset ($p = 0.23$). The **RCT** method is superior to **DCT** for both datasets ($p < 0.01$) whereas the scores with **PRL** are equivalent to **DCT** for both datasets. We see then that the ensembling strategy can also be improved when adding an extra step for matching the statistics of the datasets of each pair of *source-target* subjects.

VII. Conclusion and Future works

In this paper, we have presented a new method for overcoming the negative effects of statistical distribution mismatch between subjects in BCI classification. Our proposal consists of a sequence of geometrical transformations on the data points from two sessions with the intention of making the shapes of their point clouds in a high-dimensional space as similar as possible. The inspiration for this method comes from Procrustes Analysis, however, here the method has been adapted to the case where the data points are SPD matrices and live in a Riemannian manifold, which is here proposed for the first time. Another relevant theoretical contribution is the mathematical framework proposed in Section IV-B, which includes the methods in [9] and [10] and extends them, leading to our RPA method. Such formalism allows for a better understanding of the intrinsic assumptions regarding the statistics of the data points during the distribution matching procedure.

An important aspect of our proposal is that we exploit the availability of supervised information in the *source* session as well as the sequential nature of the trials in the *target* session. It should be noted, however, that when no labels are available for the *target* session, a re-centering of data points based solely on the geometric means of each dataset (which does not rely on any supervised information) already greatly improves the cross-session and cross-subject classification, as first noted in [9]. This would be the case, for instance, in BCI applications for people with extreme motor disability, where the labeling of classes is very challenging. In this kind of situation, one may still perform the re-centering and stretching steps of the RPA method for matching the statistical distributions, turning the transfer learning procedure into a completely unsupervised one.

We have assessed the superiority of the RPA method on several publicly available datasets and have used a heterogeneous panel of statistical tools to analyze the results. Also, we have included in our study other recent contributions from the literature, leading to a comprehensive comparison of the performance of state-of-the-art methods. We hope that the breath of the analysis performed here will be useful as a reference for future works related to Transfer Learning on the SPD manifold. In order to foster reproducible research, complete Python code for the results in this paper is available at `https://github.com/plcrodrigues/RPA`.

Future perspectives for this work shall include an online implementation of our method, where usual drifts in statistics from signals on the same session would be corrected via distribution matching. An important challenge for such procedure would be to detect when changes in the statistics occur as well as when the number of new trials is already large enough so that no information from data points drawn from previous statistical distributions are needed. Another interesting line of work would be to go further in the analysis of Section VI-D by extending the methods proposed in [12], [14], and [13] with a statistical matching step based on the RPA. Finally, another interesting topic to investigate would be to consider a Transfer Learning approach for matching data from subjects having different numbers of electrodes.

VIII. Acknowledgement

This work is partly supported by the ERC Grant CHESS 2012-ERC-AdG-320684. We thank Florent Bouchard for his help on the optimization procedure in Equation (35).

REFERENCES

[1] M. Congedo, "EEG Source Analysis," 2013. [Online]. Available: https://tel.archives-ouvertes.fr/tel-00880483/document

[2] F. Lotte, L. Bougrain, A. Cichocki, M. Clerc, M. Congedo, A. Rakotomamonjy, and F. Yger, "A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update," *Journal of Neural Engineering*, vol. 15, no. 3, p. 031005, apr 2018.

[3] M. Congedo, A. Barachant, and A. Andreev, "A new generation of brain-computer interface based on riemannian geometry," 2013.

[4] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, oct 2010.

[5] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of Statistical Planning and Inference*, vol. 90, no. 2, pp. 227–244, oct 2000.

[6] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bnau, and M. Kawanabe, "Direct importance estimation for covariate shift adaptation," *Annals of the Institute of Statistical Mathematics*, vol. 60, no. 4, pp. 699–746, aug 2008.

[7] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy, "Optimal transport for domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 9, pp. 1853–1865, sep 2017.

[8] N. T. H. Gayraud, A. Rakotomamonjy, and M. Clerc, "Optimal Transport Applied to Transfer Learning For P300 Detection," in *7th Graz Conference*, Graz, Austria, Sep. 2017, p. 6.

[9] P. Zanini, M. Congedo, C. Jutten, S. Said, and Y. Berthoumieu, "Transfer learning: a Riemannian geometry framework with applications to brain-computer interfaces," *IEEE Transactions on Biomedical Engineering*, pp. 1–1, 2017.

[10] O. Yair, M. Ben-Chen, and R. Talmon, "Parallel transport on the cone manifold of SPD matrices for domain adaptation," *ArXiv e-prints*, Jul. 2018.

[11] B. Blankertz, M. Kawanabe, R. Tomioka, F. Hohlefeld, K.-R. Müller, and V. V. Nikulin, "Invariant common spatial patterns: Alleviating nonstationarities in brain-computer interfacing," in *Advances in Neural Information Processing Systems 20*, 2008, pp. 113–120.

[12] S. Fazli, F. Popescu, M. Danczy, B. Blankertz, K.-R. Müller, and C. Grozea, "Subject-independent mental state classification in single trials," *Neural Networks*, vol. 22, no. 9, pp. 1305 – 1312, 2009.

[13] N. R. Waytowich, V. J. Lawhern, A. W. Bohannon, K. R. Ball, and B. J. Lance, "Spectral transfer learning using information geometry for a user-independent brain-computer interface," *Frontiers in Neuroscience*, vol. 10, sep 2016.

[14] V. Jayaram, M. Alamgir, Y. Altun, B. Schölkopf, and M. Grosse-Wentrup, "Transfer Learning in Brain-Computer Interfaces," *IEEE Computational Intelligence Magazine*, no. February, pp. 20–31, 2015.

[15] P.-J. Kindermans, M. Schreuder, B. Schrauwen, K.-R. Muller, and M. Tangermann, "True zero-training brain-computer interfacing ? an online study," *PLOS ONE*, vol. 9, no. 7, pp. 1–13, 07 2014.

[16] D. Hübner, T. Verhoeven, K. Müller, P. Kindermans, and M. Tangermann, "Unsupervised learning for brain-computer interfaces based on event-related potentials: Review and online comparison [research frontier]," *IEEE Computational Intelligence Magazine*, vol. 13, no. 2, pp. 66–77, May 2018.

[17] J. C. Gower and G. B. Dijksterhuis, *Procrustes Problems*. Oxford University Press, jan 2004.

[18] D. G. Kendall, "A survey of the statistical theory of shape," *Statistical Science*, vol. 4, no. 2, pp. 87–99, may 1989.

[19] C. Wang and S. Mahadevan, "Manifold alignment using procrustes analysis," in *Proceedings of the 25th international conference on Machine learning*. ACM Press, 2008.

[20] R. Bhatia, *Positive definite matrices*. Princeton university press, 2009.

[21] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Multiclass brain–computer interface classification by Riemannian geometry," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 4, pp. 920–928, apr 2012.

[22] M. Congedo, A. Barachant, and R. Bhatia, "Riemannian geometry for EEG-based brain-computer interfaces; a primer and a review," *Brain-Computer Interfaces*, pp. 1–20, 2017.

[23] F. Yger, M. Berar, and F. Lotte, "Riemannian approaches in brain-computer interfaces: A review," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 10, pp. 1753–1762, oct 2017.

[24] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[25] M. Harandi, M. Salzmann, and R. Hartley, "Dimensionality reduction on SPD manifolds: The emergence of geometry-aware methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 1, pp. 48–62, jan 2018.

[26] M. Congedo, A. Barachant, and E. K. Koopaei, "Fixed point algorithms for estimating power means of positive definite matrices," *IEEE Transactions on Signal Processing*, vol. 65, no. 9, pp. 2211–2220, may 2017.

[27] S. Said, L. Bombrun, Y. Berthoumieu, and J. H. Manton, "Riemannian Gaussian distributions on the space of symmetric positive definite matrices," *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 2153–2170, apr 2017.

[28] R. Bhatia and M. Congedo, "Procrustes problems in riemannian manifolds of positive definite matrices," *Linear Algebra and its Applications*, vol. 563, pp. 440–445, feb 2019.

[29] M. Moakher, "A differential geometric approach to the geometric mean of symmetric positive-definite matrices," *SIAM Journal on Matrix Analysis and Applications*, vol. 26, no. 3, pp. 735–747, jan 2005.

[30] J. Townsend, N. Koep, and S. Weichwald, "Pymanopt: A python toolbox for optimization on manifolds using automatic differentiation," *Journal of Machine Learning Research*, vol. 17, no. 137, pp. 1–5, 2016.

[31] R. R. Coifman and S. Lafon, "Diffusion maps," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 5–30, jul 2006.

[32] S. Lafon and A. Lee, "Diffusion maps and coarse-graining: a unified framework for dimensionality reduction, graph partitioning, and data set parameterization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1393–1403, sep 2006.

[33] P. L. C. Rodrigues, M. Congedo, and C. Jutten, "Multivariate time-series analysis via manifold learning," in *2018 IEEE Statistical Signal Processing Workshop (SSP)*, 2018.

[34] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine Learning*, vol. 79, no. 1-2, pp. 151–175, oct 2009.

[35] V. Jayaram and A. Barachant, "Moabb: Trustworthy algorithm benchmarking for bcis," *Journal of Neural Engineering*, 2018.

[36] M. Congedo, M. Goyat, N. Tarrin, L. Varnet., B. Rivet, G. Ionescu, N. Jrad, R. Phlypo, M. Acquadro, and C. Jutten, "Brain invaders: a prototype of an open-source p300-based video game working with the openvibe platform," *5th International BCI Conference, Graz, Austria, 280-283*, vol. 2011, no. Bci, pp. 1–6, 2011.

[37] E. K. Kalunga, S. Chevallier, Q. Barthélemy, K. Djouani, E. Monacelli, and Y. Hamam, "Online SSVEP-based BCI using Riemannian geometry," *Neurocomputing*, vol. 191, pp. 55–68, may 2016.

[38] G. Schalk, D. McFarland, T. Hinterberger, N. Birbaumer, and J. Wolpaw, "BCI2000: A general-purpose brain-computer interface (BCI) system," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 1034–1043, jun 2004.

[39] H. Cho, M. Ahn, S. Ahn, M. Kwon, and S. C. Jun, "EEG datasets for motor imagery brain–computer interface," *GigaScience*, vol. 6, no. 7, pp. 1–8, may 2017.

[40] M. Tangermann, K.-R. Mller, A. Aertsen, N. Birbaumer, C. Braun, C. Brunner, R. Leeb, C. Mehring, K. J. Miller, G. R. Mller-Putz, G. Nolte, G. Pfurtscheller, H. Preissl, G. Schalk, A. Schlgl, C. Vidaurre, S. Waldert, and B. Blankertz, "Review of the BCI competition IV," *Frontiers in Neuroscience*, vol. 6, 2012.

[41] D. Steyrl, R. Scherer, J. Faller, and G. R. Mller-Putz, "Random forests in non-invasive sensorimotor rhythm brain-computer interfaces: a practical and convenient non-linear classifier," *Biomedical Engineering / Biomedizinische Technik*, vol. 61, no. 1, pp. 77–86, feb 2016.

[42] J. Faller, C. Vidaurre, T. Solis-Escalante, C. Neuper, and R. Scherer, "Autocalibration and recurrent adaptation: Towards a plug and play online ERD-BCI," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 20, no. 3, pp. 313–319, may 2012.

[43] M. Grosse-Wentrup, C. Liefhold, K. Gramann, and M. Buss, "Beamforming in noninvasive brain–computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 4, pp. 1209–1219, apr 2009.

[44] A. Barachant and M. Congedo, "A plug & play P300 BCI using information geometry," vol. 1104, no. 1, pp. 1–9, 2014.

[45] I. Liiv, "Seriation and matrix reordering methods: An historical overview," *Statistical Analysis and Data Mining*, pp. n/a–n/a, 2010.

[46] E. Edgington and P. Onghena, *Randomization Tests*. Chapman and Hall CRC, 2007.

[47] D. V. Zaykin, "Optimally weighted z-test is a powerful method for combining probabilities in meta-analysis," *Journal of Evolutionary Biology*, vol. 24, no. 8, pp. 1836–1841, may 2011.

[48] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, 1979.