LETTER Communicated by Cuntai Guan

Adaptive Classification on Brain-Computer Interfaces Using Reinforcement Signals

A. Llera
a.llera@donders.ru.nl
V. Gómez
v.gomez@science.ru.nl
H. J. Kappen
b.kappen@science.ru.nl
Radboud University Nijmegen, the Netherlands, and Donders Institute for Brain, Cognition and Behaviour, Nijmegen, Gelderland 6525 EZ, Netherlands

We introduce a probabilistic model that combines a classifier with an extra reinforcement signal (RS) encoding the probability of an erroneous feedback being delivered by the classifier. This representation computes the class probabilities given the task related features and the reinforcement signal. Using expectation maximization (EM) to estimate the parameter values under such a model shows that some existing adaptive classifiers are particular cases of such an EM algorithm. Further, we present a new algorithm for adaptive classification, which we call *constrained means adaptive classifier*, and show using EEG data and simulated RS that this classifier is able to significantly outperform state-of-the-art adaptive classifiers.

1 Introduction

The final goal of a brain-computer interface (BCI) is to provide human subjects with control over some device or computer application through only their measured brain activity by, for example, electroencephalogram (EEG) (Wolpaw, Birbaumer, McFarland, Pfurtscheller, & Vaughan, 2002). To gain control over the device, users usually participate in an offline training/calibration session, where they are instructed to perform some mental task according to visual stimuli while the generated brain activity is being recorded (Ramoser, Müller-Gerking, & Pfurtscheller, 1998; van Gerven & Jensen, 2009). From the recorded data, discriminative features associated with the different intentions of the user are extracted and used to train a classifier, which will predict the intention of the user during the test/feedback session.

However, due to the poor signal-to-noise ratio and the nonstationary character of the EEG data (Krauledat, 2008), the patterns extracted during the training of the BCI may differ for the feedback session, leading to poor

performance (Shenoy, Krauledat, Blankertz, Rao, & Müller, 2006). It has been shown that to keep performance acceptable, EEG-based BCIs require a robust feature space (Tomioka, Hill, Blankertz, & Aihara, 2006; Blankertz et al., 2008; von Bünau, Meinecke, Scholler, & Müller, 2010; Reuderink, Farquhar, Poel, & Nijholt, 2011; Samek, Vidaurre, Müller, & Kawanabe, 2012), and/or online adaptation of the classifier parameters (Millán, 2004; Shenoy et al., 2006). Since in practical BCI scenarios, the user intention is unknown, online supervised adaptation is not possible. Therefore, the design of unsupervised adaptive classifiers that are robust to changes due to the nonstationary character of the data has become the focus of intense research within the BCI community (Sykacek, Roberts, & Stokes, 2004; Gan, 2006; Kawanabe, Krauledat, & Blankertz, 2006; Vidaurre, Schlöogl, Cabeza, Scherer, & Pfurtscheller, 2006; Hasan & Gan, 2009a).

A common approach considers the class-conditional features as normally distributed variables and performs unsupervised adaptation of a linear discriminant analysis (LDA) classifier. Adaptive LDA pooled-mean (Pmean) and adaptive LDA pooled-mean + global covariance (PmeanGcov) are two representative examples of this kind of method (Vidaurre, Kawanabe, von Bünau, Blankertz, & Müller, 2010). Recently, interesting online experiments have shown the practical application of these techniques, not only for improving the performance with respect to static classifiers but also for reducing the training time or increasing the number of possible BCI users (Vidaurre & Blankertz, 2010; Vidaurre, Sannelli, Müller, & Blankertz, 2010). Other methods model the user intention as a latent variable for which its posterior probabilities (responsibilities) are computed and subsequently used to update a classifier. This idea has been introduced using expectation-maximization (EM) in gaussian mixture models (GMM) in Blumberg et al. (2007), and extended to sequential EM in Hasan and Gan (2009a) and Liu, Huang, Meng, Zhang, and Zhu (2010). Other extension for joint adaptive feature extraction and classification was considered in Li and Guan (2006).

A possible way to improve unsupervised adaptive methods consists of the use of a reinforcement signal (RS), which acts as a feedback provided by the user to the machine. Examples can range from button presses delivered by the user (supervised), to measured muscular activity in a hybrid BCI (Leeb, Sagha, Chavarriaga, & Millan, 2010). Another relevant example of RS is the error-related potential (ErrP), a stereotyped pattern elicited following an unexpected response from the BCI (Ferrez, 2007; Ferrez & Millán, 2008). There is evidence that this signal can be detected with high accuracy (Chavarriaga, Ferrez, & Millán, 2007; Blankertz et al., 2004; Llera, van Gerven, Gómez, Jensen, & Kappen, 2011). The inclusion of such an RS into the adaptive BCI cycle was introduced in Blumberg et al. (2007) in which, the authors extend the latent variable approach with an additional binary ErrP classifier. Similarly, in Llera et al. (2011), we proposed a discriminant-based approach that also uses a binary ErrP classifier and provided a detailed

analysis of the negative effect due to false positives or negatives on the ErrP misclassification.

In this work, we introduce a unifying framework that accommodates existing approaches in two families according to whether a latent variable is explicitly modeled. Our framework is derived from a graphical model that includes a probabilistic RS instead of a binary RS. This is a way to include the reliability over the measured RS that implements soft updates when the uncertainty is high and recovers unsupervised and supervised learning as particular cases. Further, we develop a novel algorithm for adaptive classification and present an overview of the relations between existing methods.

In section 2.1 we introduce a probabilistic graphical model, describe the EM algorithm for estimating the parameters in this model, and derive its sequential version (CSEM). In section 2.2 we develop a new sequential algorithm (CMAC) for classification and parameter estimation. In section 3.1, we provide a description of the simulated RS, which will be considered to evaluate the proposed methods. In sections 3.2 and 3.3, we present the results obtained using synthetic data. In section 4 we compare the proposed methods with other state-of-the-art classifiers using EEG data and simulated RS. In section 5, we give a brief description of methods related to our work and describe the relationships and differences between the proposed and the previously existing methods. Finally, in section 6 we discuss the presented results and consider directions for future work.

2 Methods

In section 2.1, we introduce a probabilistic graphical model that includes task-related features as well as a reinforcement signal (RS) encoding the discrepancy between user intention and the output given by the device. This formulation estimates the posterior probabilities of each class (responsibilities) after observing not only the task-related features but also the RS. We describe the EM algorithm for this model and derive a sequential version of it (CSEM). In section 2.2 we derive a new algorithm for online parameter estimation and classification (CMAC), which includes the RS in the computation of the responsibilities.

2.1 The Model. We consider a binary random variable $I \in \{1, 2\}$ representing the hidden intention of the user. Given the intention, a vector $\mathbf{x} \in \mathbb{R}^n$ represents the features extracted from the recorded brain activity of the user while having intention I. Based on these features, a probabilistic task classifier is used to compute the output of the BCI, $Z \in \{1, 2\}$, which is used for control. The output Z can be interpreted as feedback from the BCI to the user. Further, once Z is observed by the user, we consider a probabilistic RS, which provides the probability of an erroneous feedback, $E \in [0, 1]$. Given Z and E as evidence, we can use the fact that I is a binary variable and write

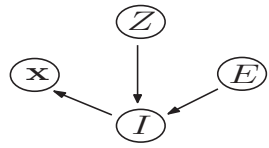

Figure 1: Probabilistic graphical model. $\mathbf{x} \in \mathbb{R}^n$ represents the task-related features extracted from the EEG data, $Z \in \{1, 2\}$ the BCI output computed using the task classifier, $E \in [0, 1]$ the RS encoding the probability that Z was an erroneous output, and $I \in \{1, 2\}$ the intention of the user.

its conditional probability as

$$p(I|Z, E) = \begin{cases} 1 - E & \text{if } I = Z \\ E & \text{if } I \neq Z \end{cases}. \tag{2.1}$$

For simplicity, we will not explicitly consider the dependence of Z on $\mathbf{x}$ through the task classifier or the dependence of E on Z through brain activity measured after observing Z and the RS. Instead, although the intention of the user is not actually caused by Z and E, we summarize the influence of Z through E as given by equation 2.1. Figure 1 shows a Bayesian network that captures the probabilities we have described.

The joint probability distribution of the proposed model is

$$p(I, \mathbf{x}, Z, E) = p(\mathbf{x}|I)p(I|Z, E)p(Z)p(E). \tag{2.2}$$

We consider $p(I|Z, E)$ as given by equation 2.1 and assume normally distributed features given the intention, that is,

$$p(\mathbf{x}|I) = \frac{1}{(2\pi)^{n/2}|\Sigma_I|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_I)^\intercal \Sigma_I^{-1}(\mathbf{x} - \boldsymbol{\mu}_I)\right) \equiv \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_I, \boldsymbol{\Sigma}_I),$$

with the intention I replaced with a mean vector $\boldsymbol{\mu}_I \in \mathbb{R}^n$ and covariance matrix $\boldsymbol{\Sigma}_I \in \mathbb{M}_{n\times n}$ as sufficient statistics. Further, we will assume a flat prior over E and Z, so $p(E) = 1, \forall E \in [0, 1]$; and $p(Z) = \frac{1}{2}, \forall Z \in \{1, 2\}$. We compactly represent the set of parameters of the model using the vector $\boldsymbol{\theta} := (\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$.

Suppose that we have a data set of observations $\mathcal{S}$ from which we can estimate the unknown model parameters $\boldsymbol{\theta}$. In our case, the observations correspond to T trials that are composed of an observed component $\mathcal{S} = \{\langle \mathbf{x}^t, Z^t, E^t \rangle\}_{t\in\{1,\dots,T\}}$ and a latent variable I^t, the intention at trial t. The log

likelihood of the data given the model parameters can be written as

$$\log p(\mathcal{S}|\boldsymbol{\theta}) = \sum_{t=1}^{T} \log \sum_{I^t=1}^{2} p(I^t, \mathbf{x}^t, Z^t, E^t|\boldsymbol{\theta})$$
$$= \sum_{t=1}^{T} \log \sum_{I^t=1}^{2} p(\mathbf{x}^t|I^t)p(I^t|Z^t, E^t)p(Z^t)p(E^t). \tag{2.3}$$

Maximizing equation 2.3 is not straightforward, since the summation over the latent variables I^t occurs inside the logarithm. Typically, the expectation maximization (EM) algorithm is used to solve this problem (Bishop, 2007). The EM algorithm is a procedure that iterates two steps until convergence.

In the E-step, one assumes current parameter values $\boldsymbol{\theta}^{\text{old}}$ and computes the posterior distribution of each of the intentions $I^t \in \{1, 2\}$, the so-called responsibilities:

$$p(I^t|\mathbf{x}^t, Z^t, E^t, \boldsymbol{\theta}^{\text{old}}) = \frac{p(\mathbf{x}^t|I^t, \boldsymbol{\theta}^{\text{old}})p(I^t|Z^t, E^t)}{\sum_{I^t=1}^{2} p(\mathbf{x}^t|I^t, \boldsymbol{\theta}^{\text{old}})p(I^t|Z^t, E^t)} \equiv \gamma_I^t. \tag{2.4}$$

In the M-step, we replace the old parameter values with the ones that result from maximizing the expected log likelihood:

$$\boldsymbol{\theta}^{\text{new}} = \arg\max_{\boldsymbol{\theta}} \sum_{t=1}^{T} \sum_{I^t=1}^{2} p(I^t|\mathbf{x}^t, Z^t, E^t, \boldsymbol{\theta}^{\text{old}}) \log p(I^t, \mathbf{x}^t, Z^t, E^t|\boldsymbol{\theta}). \tag{2.5}$$

Taking derivatives of equation 2.5 with respect to the elements of $\boldsymbol{\theta}$ and setting them to zero results in the updates

$$\boldsymbol{\mu}_I = \frac{1}{N_I} \sum_{t=1}^{T} \gamma_I^t \mathbf{x}^t, \tag{2.6}$$

$$\boldsymbol{\Sigma}_I = \frac{1}{N_I} \sum_{t=1}^{T} \gamma_I^t (\mathbf{x}^t - \boldsymbol{\mu}_I)(\mathbf{x}^t - \boldsymbol{\mu}_I)^{\intercal}, \tag{2.7}$$

where $N_I = \sum_{t=1}^{T} \gamma_I^t$.

Note that this representation (see Figure 1) computes the posterior probability of each of the intentions given the task-related features and the RS. As a consequence, the difference between the proposed methodology and the standard EM for GMM relies on the use of $p(I^t|\mathbf{x}^t, Z^t, E^t, \boldsymbol{\theta}^{\text{old}})$, equation 2.4, an RS-dependent quantity, instead of simply

$$p(I^t|\mathbf{x}^t, \boldsymbol{\theta}^{\text{old}}) = \frac{p(\mathbf{x}^t|I^t, \boldsymbol{\theta}^{\text{old}})}{\sum_{I^t=1}^{2} p(\mathbf{x}^t|I^t, \boldsymbol{\theta}^{\text{old}})}. \tag{2.8}$$

It is interesting to see that the model recovers the two following well-known cases:

- *Unsupervised case*: If the RS is noninformative, $E^t = 1/2$, the updates become the ones of the EM for GMM.
- *Supervised case*: If the RS is always correct and returns only a binary answer $E^t \in \{0, 1\}$, the responsibility of the incorrect intention is zero, whereas the responsibility of the correct intention is one.

Using the responsibilities as defined in equation 2.4, we can interpolate between unsupervised and supervised learning using the RS. This suggests an improvement over the unsupervised method given that the RS is informative. We show evidence for this in section 3.2.

If we have an incoming stream of data, as is the case for online BCI, the previous optimization might not be efficient since it uses a batch of data and an iterative procedure in order to optimize the model parameters. For online BCI, an incremental approach is necessary (Hasan & Gan, 2009b). A sequential version of the previously described EM algorithm is defined by the updates

$$\boldsymbol{\mu}_I' = (1 - \beta_\mu \gamma_I)\boldsymbol{\mu}_I + \beta_\mu \gamma_I \mathbf{x}, \tag{2.9}$$

$$\boldsymbol{\Sigma}_I' = (1 - \beta_\Sigma \gamma_I)\boldsymbol{\Sigma}_I + \beta_\Sigma \gamma_I (\mathbf{x} - \boldsymbol{\mu}_I)(\mathbf{x} - \boldsymbol{\mu}_I)', \tag{2.10}$$

where $\mathbf{x}$ is the observed task-related feature vector, γ_I are the responsibilities computed using equation 2.4 and β_μ and β_Σ are the learning rates. We denote this algorithm as corrected sequential EM (CSEM).

2.2 Constrained Means Adaptive Classifier. In this section we develop the constrained means adaptive classifier (CMAC) algorithm for online classification and parameter estimation, a novel sequential update for the model parameters $\boldsymbol{\theta}$ that will allow different rates of adaptation for shifts and rotations.

When no labels are available (unsupervised case), one can update a global mean of the data $(\mu_1 + \mu_2)/2$ by means of the learning rule

$$\frac{\boldsymbol{\mu}_1' + \boldsymbol{\mu}_2'}{2} = (1 - \beta)\frac{\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2}{2} + \beta \mathbf{x}, \tag{2.11}$$

where $\boldsymbol{\mu}_I'$ represents the updated $\boldsymbol{\mu}_I$, $\beta \in [0, 1]$ is the learning rate controlling the adaptation, and $\mathbf{x}$ is the observed task-related feature vector. This update rule, which can be seen as a constraint over the sum of the means, was introduced in Vidaurre et al. (2010). In terms of a discriminant function, this learning rule updates the bias term. Despite its simplicity, it has been shown to be able to reliably keep track of the bias, significantly improving the classification accuracy with regard to a static classifier. We generalize

this rule and obtain an update for each of the means independently, which in terms of an LDA discriminant function will allow updating both the bias and the weights of the discriminant function. Consider an update rule for the means of the form

$$\boldsymbol{\mu}_I' = (1-\beta)\boldsymbol{\mu}_I + 2\beta\gamma_I\mathbf{x}, \tag{2.12}$$

where $\boldsymbol{\mu}_I'$ represents the updated $\boldsymbol{\mu}_I$, $\beta \in [0, 1]$ is the learning rate controlling the adaptation, $\mathbf{x}$ is the observed task-related feature vector, and γ_I are the responsibilities computed using equation 2.4. A well-known fact from online learning in changing environments (Heskes, Gielen, & Kappen, 1991; Heskes & Kappen, 1992) is that the optimal learning rate depends on the noise as well as on the rate of change on the data. Under assumption 2.12, the difference of the means depends on the responsibilities, while the sum does not. Therefore, the sum can be adapted more reliably than the difference, that is, using larger learning rates. Extending equation 2.12 for the case of different learning rates ($\beta_+ > \beta_-$) results in the following constraints on the sum/difference of the means

$$\boldsymbol{\mu}_1' + \boldsymbol{\mu}_2' = (1-\beta_+)(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2) + 2\beta_+\mathbf{x}, \tag{2.13}$$

$$\boldsymbol{\mu}_2' - \boldsymbol{\mu}_1' = (1-\beta_-)(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1) + 2\beta_-(\gamma_2 - \gamma_1)\mathbf{x}. \tag{2.14}$$

Solving for $\boldsymbol{\mu}_1'$ and $\boldsymbol{\mu}_2'$ gives the final updates for the means, which are shown in equations 3.1 and 3.2 in algorithm 1. To update the covariances, we use the updated means and the learning rule, equation 2.10, where the learning rate $\beta_\Sigma \in [0, 1]$ controls the adaptation of the covariances. The full CMAC algorithm is described in algorithm 1. The initial parameters required by this algorithm can be obtained from a previous training session.

3 Results: Synthetic Data

In this section we first explain the way in which the RS will be simulated in the rest of this work. Then we show that using the responsibilities obtained by including the RS can improve the quality of the parameter estimation with regard to an unsupervised GMM estimation and perform a simulated online scenario to identify the kind of nonstationarities CMAC is able to deal with while considering different RSs.

3.1 The Simulated Reinforcement Signal. By definition, the RS encodes the probability of presence of an error, so $E \in [0, 1]$. Thus, an informative RS should provide at each trial a value E, such that $E \approx 0$ for correctly classified trials and $E \approx 1$ for erroneously classified trials. Obviously no RS is totally reliable, and violations of this condition are due to false positives

Algorithm 1: Constrained Means Adaptive Classifier.

Require: Current model parameters $\{\boldsymbol{\mu}_I, \boldsymbol{\Sigma}_I\}_{I\in\{1,2\}}$.

Currently observed feature vector at present trial t, $\mathbf{x}^t$.

Reinforcement signal E^t (after observing the output of the task classifier Z^t).

Parameters controlling the adaptation β_+, β_-, β_Σ.

1: $\Sigma := \frac{\Sigma_1+\Sigma_2}{2}$.

2: Compute the output of the task classifier at trial t as:

$$Z^t = \arg\max_I p(I|\mathbf{x}^t, \boldsymbol{\mu}_I, \boldsymbol{\Sigma}) := \arg\max_I p(\mathbf{x}^t|\boldsymbol{\mu}_I, \boldsymbol{\Sigma}).$$

3: Observe E^t and evaluate the responsibilities using equations 2.4 and 2.1:

$$\gamma^t_{Z^t} = \frac{\mathcal{N}(\mathbf{x}^t|\boldsymbol{\mu}_{Z^t}, \boldsymbol{\Sigma})(1-E^t)}{\mathcal{N}(\mathbf{x}^t|\boldsymbol{\mu}_{Z^t}, \boldsymbol{\Sigma})(1-E^t) + \mathcal{N}(\mathbf{x}^t|\boldsymbol{\mu}_{\neg Z^t}, \boldsymbol{\Sigma})E^t},$$

$$\gamma^t_{\neg Z^t} = 1 - \gamma^t_{Z^t}.$$

4: Update the model parameters:

$$\boldsymbol{\mu}'_1 = (1 - \frac{\beta_+ + \beta_-}{2})\boldsymbol{\mu}_1 + \frac{\beta_- - \beta_+}{2}\boldsymbol{\mu}_2 + \left(\beta_+ - \beta_-(\gamma^t_2 - \gamma^t_1)\right)\mathbf{x}^t, \quad (2.15)$$

$$\boldsymbol{\mu}'_2 = (1 - \frac{\beta_+ + \beta_-}{2})\boldsymbol{\mu}_2 + \frac{\beta_- - \beta_+}{2}\boldsymbol{\mu}_1 + \left(\beta_+ + \beta_-(\gamma^t_2 - \gamma^t_1)\right)\mathbf{x}^t, \quad (2.16)$$

$$\boldsymbol{\Sigma}'_I = \left(1 - \beta_\Sigma\gamma^t_I\right)\boldsymbol{\Sigma}_I + \beta_\Sigma\gamma^t_I(\mathbf{x}^t - \boldsymbol{\mu}_I)(\mathbf{x}^t - \boldsymbol{\mu}_I)^\intercal, \forall I \in \{1,2\}. \quad (2.17)$$

5: **return** Updated parameters: $\{\boldsymbol{\mu}'_I, \boldsymbol{\Sigma}'_I\}_{I\in\{1,2\}}$.

and negatives of the RS. In order to illustrate such an scenario, we model $p(E)$ as a symmetric mixture of beta distributions:

$$p(E) = \frac{1}{2}\left(\beta(E|w_1, w_2) + \beta(E|w_2, w_1)\right), \quad (3.1)$$

where

$$\beta(E|w_1, w_2) = \frac{E^{w_1-1}(1-E)^{w_2-1}}{\int_0^1 z^{w_1-1}(1-z)^{w_2-1}dz} \quad (3.2)$$

for some $(w_1, w_2) \in \mathbb{R}^+ \times \mathbb{R}^+$.

Making use of the output delivered by the task classifier Z and the real intention of the user I, we define $p(E|I = Z) := \beta(E|w_1, w_2)$ and $p(E|I \neq Z) := \beta(E|w_2, w_1)$ with $w_1 < w_2$. In this way, at each trial, E is generated by

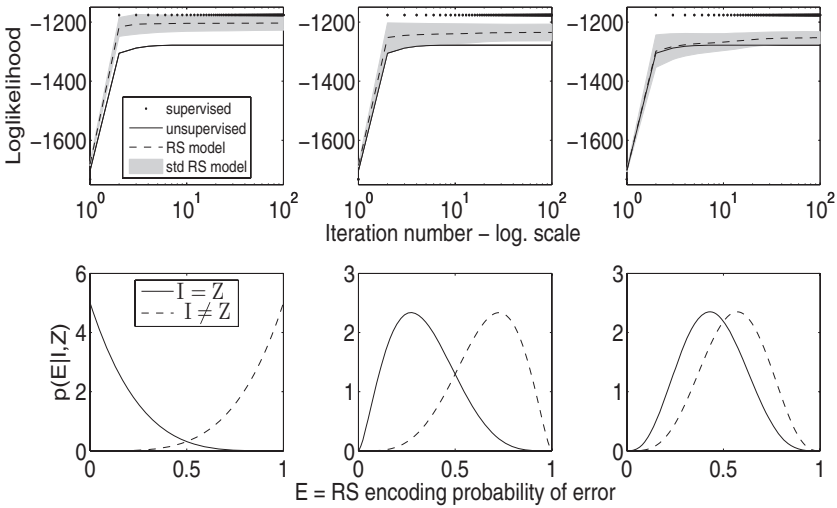

Figure 2: (Top) Log likelihood of the solutions obtained at each iteration for the unsupervised case (continuous line), supervised case (dotted line), and proposed model (dashed line). The three horizontal plots show the different results obtained considering three levels of accuracy of the RS corresponding to the parameterizations of $p(E|I, Z)$ shown in the second row. (Bottom) $p(E|I, Z)$ is represented for three parameterizations of mixtures of β distributions. Continuous lines show $p(E|I = Z)$, and discontinuous lines $p(E|I \neq Z)$. From left to right, the parameters for $(w1, w2)$ were set to $\{(1, 5), (2.5, 5), (4, 5)\}$, respectively.

drawing a sample from $p(E|I = Z)$ if the trial was correctly classified and from $p(E|I \neq Z)$ otherwise.

As an illustration, in the second row of Figure 2, we show three examples of the resulting density functions for $(w_1, w_2) \in \{(1, 5), (2.5, 5), (4, 5)\}$, respectively. This parameterization results in a Bayes classification error of the RS of approximately 5%, 20%, and 40%. respectively.

Summarizing, the parameterization allows the simulation of a probabilistic RS whose accuracy can be controlled by means of the values of (w_1, w_2). Note that the supervised case is an extreme scenario in which the beta distributions became delta peaks at zero or one.

3.2 Batch Learning. We start by generating a data set $\mathcal{S} = \{\langle \mathbf{x}^t, Z^t, E^t \rangle\}_{t \in \{1,\ldots,T\}}$. We consider for simplicity a two-dimensional feature space and consequently generate the task-related features $\{\mathbf{x}^t\}_{t \in \{1,\ldots,T\}}$ by sampling with the same probability from two gaussian distributions with parameters $\bar{\boldsymbol{\mu}}_I \in \mathbb{R}^2$ and $\bar{\boldsymbol{\Sigma}}_I \in \mathbb{M}_{2\times 2}$, $I \in \{1, 2\}$. Each sample $\mathbf{x}^t$ is classified as $Z^t = \arg\max_I p(I|\mathbf{x}^t, \tilde{\boldsymbol{\mu}}_I, \tilde{\boldsymbol{\Sigma}}_I)$, where the values for $\tilde{\boldsymbol{\mu}}_I$ are chosen randomly

and $\tilde{\mathbf{\Sigma}}_I = \mathbb{I}_{2\times 2}$. Then we generate the RS outputs by drawing for each trial one sample E^t from $p(E|I = Z)$ if the trial was correctly classified and from $p(E|I \neq Z)$ otherwise, as explained in section 3.1. Once the data set $\mathcal{S}$ is defined, we iterate equations 2.4, 2.6, and 2.7 until convergence of the log likelihood, equation 2.5.

To evaluate the quality of the solutions obtained, we can compare their log likelihood with the one obtained applying an unsupervised EM algorithm for GMM (analytically equivalent to $E^t = \frac{1}{2}\ \forall t$ in our model) and with the supervised case (analytically equivalent to the proposed model using an RS that at each trial produces an output $E^t = 1 - \delta_{I^t,Z^t}$, with I^t the real intention at trial t).

In the first row of Figure 2, we plot the log likelihood of the parameters obtained at each iteration of the algorithm for the supervised case (dotted line), unsupervised case (continuous line), and proposed model, including the simulated RS (discontinuous line). In this case, to account for variations in learning due to different realizations of the RS, the presented results are the average over 100 realizations of the experiment, and the shaded area describes the standard deviation around the plotted mean solution. The three plots reflect the different behavior of the model while considering three different RS corresponding to the three parameterizations of $p(E|I, Z)$ plotted in the second row.

For this illustration we considered $T = 100$,

$$\bar{\boldsymbol{\mu}}_1 = (1, 1), \bar{\boldsymbol{\mu}}_2 = (2, -1), \bar{\mathbf{\Sigma}}_1 = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}, \bar{\mathbf{\Sigma}}_2 = \begin{pmatrix} 3 & 2 \\ 2 & 3 \end{pmatrix},$$

and the parameters were initialized as $\boldsymbol{\mu}_1 = \tilde{\boldsymbol{\mu}_1}$, $\boldsymbol{\mu}_2 = \tilde{\boldsymbol{\mu}_2}$, and $\mathbf{\Sigma}_I = \mathbb{I}_{2\times 2}$. However, the results obtained are not critically dependent on the choice of the parameters.

Note that the log likelihood in the supervised scenario is the highest, and the solutions obtained using the proposed RS model improve the ones obtained in the unsupervised case for all considered RS. Further, as the RS becomes more reliable, the log likelihood of the solutions is higher. This occurs because the RS is able to correct the responsibilities for the wrongly classified trials, either because they lie close to the decision boundary or would have been wrongly assigned a large responsibility in the standard unsupervised sense.

These results confirm that this model allows interpolating between the unsupervised and the supervised parameter estimation with the help of the RS and that the responsibilities computed using equation 2.4 provide a more accurate class measure than the responsibilities computed using equation 2.8.

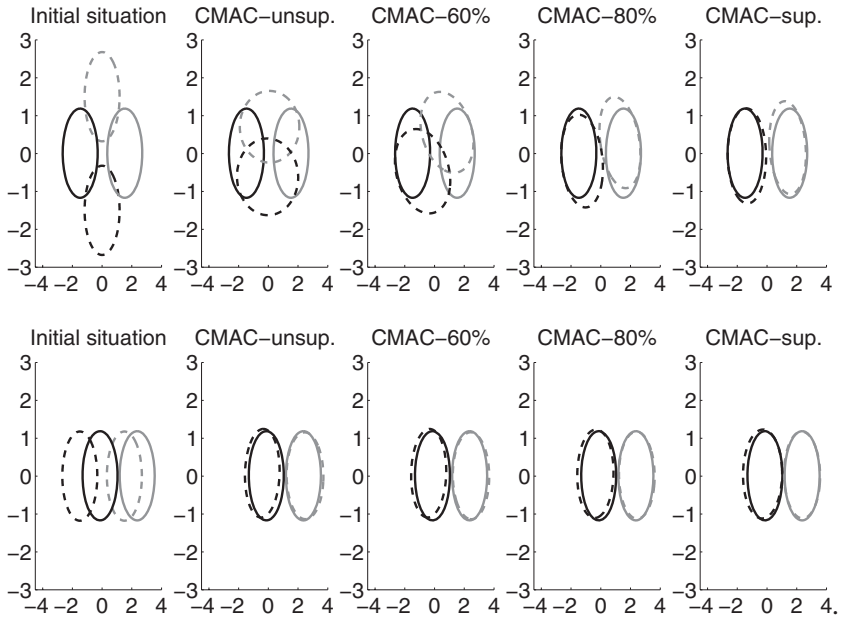

Figure 3: Black and gray curves represent the distributions of two classes. Each row represents a different change in the feature distributions—a rotation in the first row and a shift in the second. The left column represents the situation at the beginning of the test session, with dashed lines representing the train features distributions and solid lines the test features distributions. Columns 2 to 5 show the test features distributions (solid lines) and the learned distributions using CMAC (dashed lines) under the assumption of different RS after 50 samples were drawn from each of the test feature distributions.

3.3 CMAC: Online Learning Behavior. We consider now two simulated online scenarios in which the underlying feature distributions are modified from the train session to the test session by means of a rotation and a translation of the optimal decision boundaries. These simulations provide insight into which kinds of nonstationarities can be captured by the proposed algorithm CMAC.

In Figure 3, black and gray curves represent the distributions of two classes. Each row represents a different situation; the left-most column shows the situation at the beginning of the test session, with the continuous curves representing the distributions of the test features, while the discontinuous curves represent the distributions of the train features. In the first row, there is a rotation between train and test distributions, and in the second row, there is a shift of the distributions.

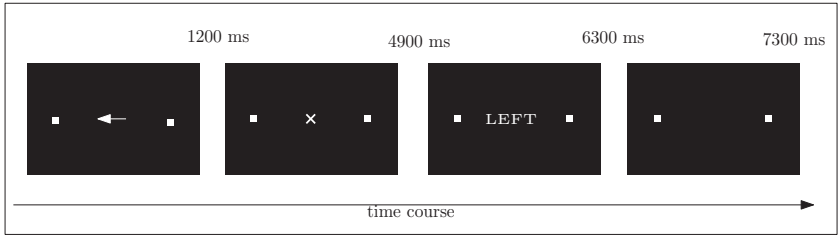

Figure 4: Experimental protocol during each trial of the test sessions. From time 0 to 1200 ms, an arrow indicates the side to which the task must be performed. After this period, a fixation cross is presented (1200–4900 ms) indicating the period to perform the task. At the end of this period, the device returns feedback to the user (4900–6300 ms), and it is followed by 1000 ms of no activity before the beginning of a new trial. During the calibration measurement, the protocol was identical with the exception that no feedback was returned.

Columns 2 to 5 show the test distributions (continuous lines) and the learned distributions using CMAC (dashed curves) under the assumption of different RS after generating 100 samples by sampling with the same probability from both test distributions. For the cases of 60% and 80% of accuracy of the RS, the results are the average over 100 realizations of the experiment. The standard deviation of these results was considered insignificant for visualization. In this example, the parameter values were fixed to the values $\beta_+ = 0.05$, $\beta_- = 0.005$, and $\beta_\Sigma = 0.01$. The choices of these parameters do not drastically change the results in terms of the obtained solution.

Note that in the case of a translation of the distributions (second row), the new distributions can be estimated without the use of any RS. If a rotation of the distributions occurred (first row), an RS is necessary, and the estimation improves with the quality of the RS. This result is due to the fact that in order to correct for a rotation in the optimal decision boundary, the weights of the discriminant function need to be updated, and that requires the class label information.

4 Results: EEG Data

In this section we use EEG data to perform a comparison between CMAC and other classifiers.

The EEG data were recorded from six subjects who participated in an experiment performing a binary motor imagery task (left-right hand) according to visual stimuli. Each subject participated in a calibration measurement consisting of 35 trials per class where no feedback was delivered and two test sessions of 70 trials each where the feedback was delivered to the user in the form of a binary response. (See Figure 4 for more detailed

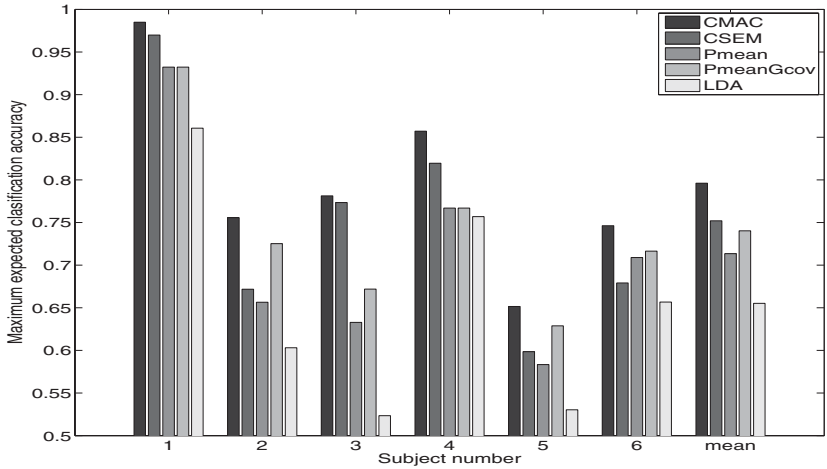

Figure 5: Maximum performance achievable using CMAC, CSEM, Pmean, and PmeanGcov, as well as the performance obtained using LDA.

information about the experiment design.) Between each two sessions, there was a pause of 5 minutes.

The brain activity was recorded using a multichannel EEG with 64 electrodes at 2048 Hz. The data were downsampled at 250 Hz and made into trials using at each trial the data from the imaginary movement period (1200–4900 ms). An automatic variance-based routine was applied to remove noisy trials and channels. The data were then linearly detrended and bandpass-filtered in the frequency band 8 to 30 Hz since these have been previously reported as the frequencies of main interest (Müller-Gerking, Pfurtscheller, & Flyvbjerg, 1998). Common spatial patterns (CSP) (Fukunaga, 1990; Lemm, Blankertz, Curio, & Müller, 2005) were computed using the data from the calibration session, and the number of selected filters was three from each side of the spectrum. After projecting each trial to the space generated by the six filters, the logarithm of the normalized variance of these projections were used as features, resulting in a feature space of dimension 6.

In the rest of this section, we use the previously described data set to perform a comparison between different classifiers, including a static classifier, *LDA* (Fukunaga, 1990), and the adaptive classifiers Pmean, PmeanGcov, CSEM, and CMAC. We first study the best possible performance obtained while using each of the considered methods.

Figure 5 shows the maximum classification accuracy (number of correctly classified trials divided by the total number of trials) obtained by each method while considering optimized learning rates and an optimal

Table 1: Mean Across Subjects and Standard Deviation of the Optimal Learning Rates for the Adaptive Classifiers.

	CMAC $\beta_+, \beta_-, \beta_\Sigma$	CSEM β_μ, β_Σ	PmeanGcov β_μ, β_Σ	Pmean β_μ
Mean	0.068, 0.019, 0.035	0.019, 0.069	0.019, 0.063	0.084
Standard deviation	0.079, 0.011, 0.062	0.014, 0.049	0.014, 0.041	0.073

RS. For each algorithm and each subject, the learning rates were optimized using grid search in parameter space.

First, note that all adaptive classifiers are able to outperform LDA. From the adaptive classifiers, CMAC is clearly the algorithm able to reach the highest accuracy, followed by CSEM and PmeanGcov, which reflect a similar performance. In the case of Pmean, we see that the model with fewer parameters (only one learning rate) is the one achieving the lowest accuracy, but it is still able to clearly outperform LDA. Globally, these results show that CMAC has the potential power to outperform all other considered methods. In Table 1, we show the mean (across subjects) and standard deviation of the optimal parameter values set for each of the considered adaptive classifiers.

In practice, we do not have prior access to the optimal learning rates and, furthermore, no RS is optimal. Clearly, different choices for the learning rates will affect the performance of all methods. Moreover, suboptimal RS will affect the performance of CMAC and CSEM. In Figure 6 we present the performance of each of the methods as a function of different RS simulated, as explained in section 3.1, while using for each subject a set of learning rates computed using leave-one(subject)-out cross-validation. For CMAC and CSEM, the reported results are averages over 100 realizations of the experiment to account for fluctuations due to different realizations of the RS. For CMAC, the standard deviation of the results is shown as error bars. In the case of CSEM, the standard deviation of the solutions was very similar to that of CMAC, and we decided to ignore them for visualization reasons.

We observe that in general, all adaptive classifiers are able to outperform the static LDA. Considering the supervised scenarios, note that CMAC (100%) outperforms CSEM (100%). Only for subject number 1 is CSEM (100%) the best algorithm. Ignoring the supervised methods, we see also that CMAC is less sensible to nonoptimal RS than CSEM. For subjects 3, 4, and 6 CMAC is the best algorithm for all considered RS, and for subjects 2 and 5, it requires an accurate RS in order to improve with regard to the unsupervised classifiers.

In Figure 7 we consider the cumulative classification accuracy against the trial number for PmeanGcov, Pmean, and CMAC with optimal RS.

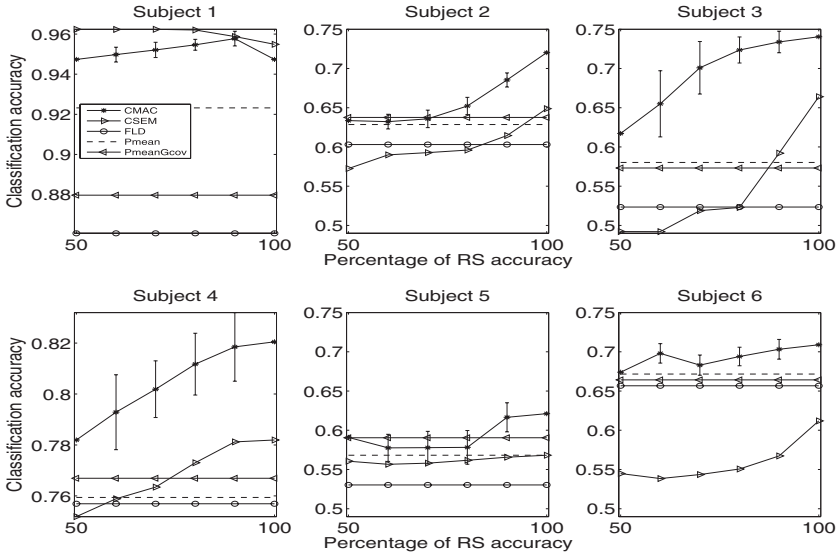

Figure 6: Classification accuracy (y-axis) as a function of the accuracy of the RS (x-axis) for all subjects. For CMAC, the standard deviation of the solutions obtained is presented as an error bar.

CMAC is able to outperform the other methods not only at the end of the experiment (trial 140) but in general also at the end of the first test session (trial number 70). While for subject number 6, all methods show a similar trend, note that for subjects numbers 1 to 4, the difference in performance between CMAC and the other methods is larger at trial 140 than at trial 70, showing that CMAC was able to adapt better after the pause between sessions. It is interesting to see that CMAC finishes the experiment with a general increasing trend, suggesting that the model continues a proper adaptation. Note in particular that for subject 5, the performance of CMAC was worse at trial 70 than one of the other methods, but due to the ability to keep adapting after the pause, CMAC is the best algorithm at the end of the experiment.

An interesting observation is that for some subjects, for example, subject 6, all methods perform similarly, while for other subjects, for example, number 3, the increase in performance obtained using CMAC is significant. To understand this behavior, in Figure 8 we show the distributions of the projected train data (discontinuous curves) and test data (continuous curves) onto the first and second CSP filters for subjects 6 and 3. Black and gray curves represent the two classes. The boundary learned using the training data as well as the optimal boundary for the test data are also shown.

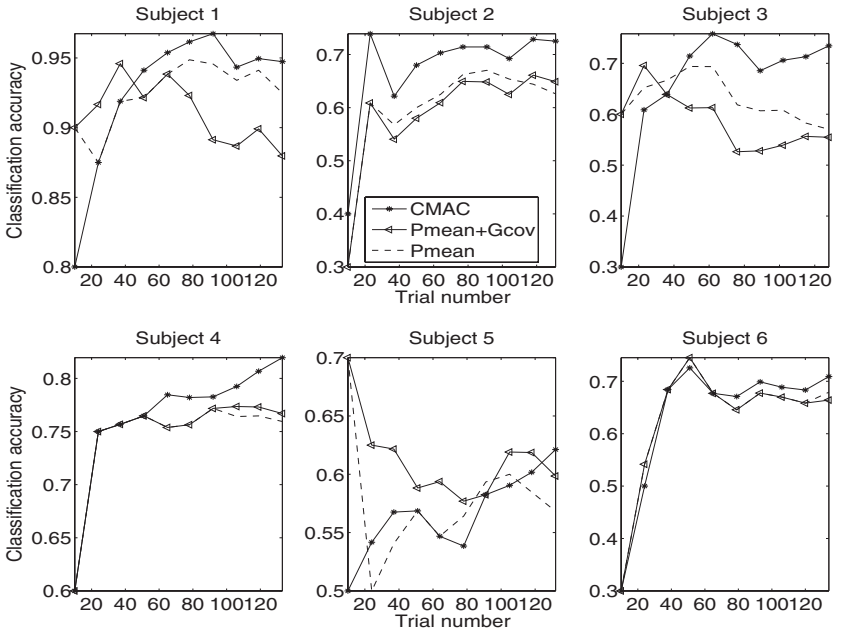

Figure 7: Cumulative classification accuracy against trial number for the PmeanGcov, Pmean, and CMAC.

For subject 6, we observe that the change in the distributions is very small, allowing every method to adapt to the changes. In contrast, the features of subject 3 change notably between train and test. In particular, there is a clear rotation of the optimal boundary, which prevents Pmean from adapting properly. In this subject, although PmeanGcov should theoretically be able to correct for the rotation since it updates the global covariance matrix, that is not the case. The reason that PmeanGcov performs worse than CMAC could then be explained by the fact that in addition to the rotation, the classes are swapped in the second filter (y-axis). Class information is therefore required to adapt to this type of changes, making it impossible for unsupervised methods such as PmeanGcov.

5 Related Work and Relations Between Methods

In this section we provide a short description of the existing adaptive classification methods commonly used for BCI purposes and show the relationships between them. We also give a brief description of the binary linear discriminant analysis (*LDA*) classifier.

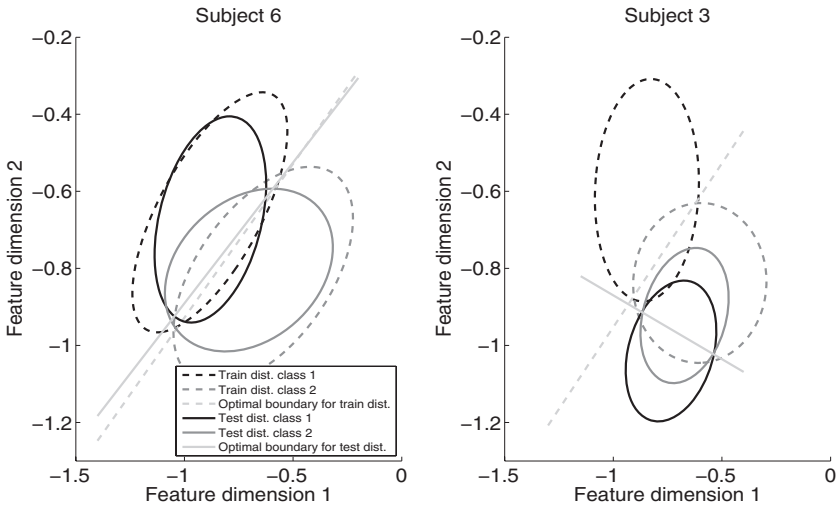

Figure 8: Projection of the data onto the first two CSP filters for subjects 6 and 3. Black and gray ellipses represent two different classes, and discontinuous and continuous curves represent the train data and the test data, respectively. Discontinuous and continuous straight lines show the optimal boundary for the train distributions and the test distributions, respectively. For subject 6 (left), feature distributions do not change significantly. In contrast, for subject 3 (right), the optimal boundary rotates and the class labels are swapped in the vertical axis.

5.1 Binary Linear Discriminant Analysis. The binary LDA (Fukunaga, 1990) is a classifier identified by a discriminant function $\mathcal{D}(\mathbf{x}) \in \mathbb{R}$ of the input feature vector $\mathbf{x} \in \mathbb{R}^n$. Denoting by $\boldsymbol{\mu}_k$ and Σ_k the means and covariance matrices of the train features of class $k \in \{1, 2\}$, respectively, and by $\Sigma = \frac{\Sigma_1 + \Sigma_2}{2}$ the mean of the covariance matrices, the discriminant function is defined by these equations:

$$\mathcal{D}(\mathbf{x}) = [b, \mathbf{w}^\mathsf{T}] \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix}, \tag{5.1}$$

$$\mathbf{w} = \Sigma^{-1} \left(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1 \right), \tag{5.2}$$

$$b = -\mathbf{w}^\mathsf{T} \boldsymbol{\mu}, \tag{5.3}$$

$$\boldsymbol{\mu} = \frac{1}{2} \left(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2 \right). \tag{5.4}$$

Using this notation, $\mathbf{x}$ is classified as class 2 if $\mathcal{D}(\mathbf{x}) > 0$ and as class 1 otherwise.

Note that $\mathbf{w} \in \mathbb{R}^n$ describes the vector of weights, and $b \in \mathbb{R}$ describes the bias term of the discriminant function. Consequently, $\mathbf{w}$ determines the direction of the separating hyperplane while b the shift of the hyperplane with regard to the origin.

5.2 Linear Discriminant Analysis with Pooled Mean Adaptation. Pmean (Vidaurre et al., 2010) is a discriminant-based unsupervised adaptive LDA algorithm that under the assumption of balanced classes identifies equation 5.4 with the global mean of the data. In this way, equation 5.4 can be updated without any label information using the learning rule

$$\boldsymbol{\mu}' = (1 - \beta)\boldsymbol{\mu} + \beta\mathbf{x}, \tag{5.5}$$

where $\boldsymbol{\mu}'$ is the updated $\boldsymbol{\mu}$, $\mathbf{x}$ is the new observed feature vector, and $\beta \in [0, 1]$ is the learning rate controlling the adaptation. In this way the bias from the discriminant function gets updated through equation 5.3.

This kind of adaptation tracks changes in the bias of the discriminant function; consequently, Pmean is able to adapt to shifts in feature space. Since the weights are not modified, this model cannot account for changes in the direction of the separating hyperplane, as, for example, rotations on feature space. Note that Pmean can be considered a particular case of CMAC where $\beta_{\_} = 0$ and $\beta_{\Sigma} = 0$.

5.3 Pmean with Global Covariance Adaptation. PmeanGcov (Vidaurre et al., 2010) is a discriminant-based unsupervised adaptive LDA algorithm that in addition to the Pmean adaptation performs a sequential adaptation of the inverse of the global covariance. Under the assumption of balanced classes, we have that

$$\mathbf{w} = \Sigma^{-1}\left(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1\right) \propto \tilde{\Sigma}^{-1}\left(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1\right) := \tilde{\mathbf{w}},$$

where $\tilde{\Sigma}$ is the global covariance matrix. Then, defining

$$\tilde{b} = -\tilde{\mathbf{w}}^{\intercal}\boldsymbol{\mu},$$

we have that $\mathcal{D}(\mathbf{x}) \propto \tilde{\mathcal{D}}(\mathbf{x}) := [\tilde{b}, \tilde{\mathbf{w}}^{\intercal}][\begin{smallmatrix} 1 \\ \mathbf{x} \end{smallmatrix}]$. This means that if the classes are balanced, we obtain the same separating hyperplane using $\tilde{\Sigma}$ rather than Σ. Using $\tilde{\Sigma}$ has the advantage that it can be updated online without using class information. Furthermore, making use of the Woodbury matrix identity

(matrix inversion lemma), we can directly update its inverse as

$$I = \tilde{\Sigma} - \frac{\mathbf{v}\mathbf{v}^{\mathsf{T}}}{\frac{1-\beta_{\Sigma}}{\beta_{\Sigma}} + \mathbf{x}^{\mathsf{T}}\mathbf{v}}, \tag{5.6}$$

$$\tilde{\Sigma}'^{-1} = \frac{I}{1-\beta_{\Sigma}}, \tag{5.7}$$

where $\tilde{\Sigma}$ is the global covariance, $\tilde{\Sigma}'^{-1}$ the updated inverse global covariance matrix, $\mathbf{v} = \tilde{\Sigma}^{-1}\mathbf{x}$, and $\beta_{\Sigma} \in [0, 1]$ is a learning rate controlling the adaptation of the covariance. This direct update of the inverse is recommended for online applications due to its efficiency.

The update of the inverse of the covariance allows for an update of equation 5.2, so PmeanGcov allows for the update of the weights and also the bias of the classifier. Consequently, this model has the potential power to adapt to shifts as well as changes in the direction of the separating hyperplane.

Note that the update of the covariance in PmeanGcov is different from the class-dependent covariance updates in CMAC. However, if the classes are balanced, using the pooled covariance in place of the mean of the class-wise covariance matrices results in the same separating hyperplane, so in that case, PmeanGcov is equivalent to CMAC, with $\beta_{\_} = 0$.

5.4 Adaptive Linear Discriminant Analysis. ALDA (Blumberg et al., 2007) is a latent-variable-based unsupervised adaptive LDA algorithm in the sense that considers the user intention as a hidden variable, models the feature space as a GMM, and constructs an LDA classifier with the parameters estimated using EM. The batch optimization is performed using a sliding window of the last $\mathcal{M} \in \mathbb{N}$ trials. In this way, only the most recent trials take part in the optimization, allowing the modeling of nonstationary environments.

Since ALDA updates means and covariances, it performs an adaptation of all parameters included in the discriminant function, allowing it to adapt for shifts as well as for rotation on feature space.

Since CSEM in the unsupervised scenario ($E^t = \frac{1}{2}\ \forall t$) is a sequential EM algorithm for GMM, we can identify ALDA as the batch version of CSEM and can consider them as equivalent.

5.5 Adaptive Linear Discriminant Analysis with Error Correction. Similar to our algorithm CSEM, Blumberg et al. (2007) introduce a probabilistic model of a binary error signal, which corresponds to our RS term $p(I_k|Z, E)$. ALDEC performs implicit modeling of the decoding power DP (accuracy of the task classifier) and the reliability R of the RS. In contrast, in CSEM, we explicitly include this information in the model and provide

update rules for the means and covariances (see equations 2.6 and 2.7) where the RS is included through the novel responsibilities, equation 2.4. This allows us to recover both supervised and unsupervised cases. Further, this allows us to include more realistic, single-trial realizations of the RS. Notice that in Blumberg et al. (2007), the RS is modeled as two delta peaks at R and $1 - R$ (see section 3.1).

5.6 Sequential EM. SEM (Hasan & Gan, 2009a) is a sequential version of EM for GMM. In this algorithm, the means and covariance matrices of the task-related feature distributions are sequentially updated using

$$\boldsymbol{\mu}_I' = (1 - \beta_\mu \gamma_I)\boldsymbol{\mu}_I + \beta_\mu \gamma_I \mathbf{x}, \tag{5.8}$$

$$\boldsymbol{\Sigma}_I' = (1 - \beta_\Sigma \gamma_I)\boldsymbol{\Sigma}_I + \beta_\Sigma \gamma_I (\mathbf{x} - \boldsymbol{\mu}_I)(\mathbf{x} - \boldsymbol{\mu}_I)', \tag{5.9}$$

where $\boldsymbol{\mu}_I'$ and $\boldsymbol{\Sigma}_I'$ represent the updated means and covariance matrices, respectively, $\mathbf{x}$ is the observed task related feature vector, γ_I the responsibilities computed using equation 2.8, and β_μ and β_Σ are learning rates that value decreases in size with the iteration number in order to reach convergence to the optimal static set of parameters. If our goal is to dynamically model a nonstationary environment, we can relax the dependence of the learning rates value on the trial number, so $\beta_\mu \in [0, 1]$ and $\beta_\Sigma \in [0, 1]$.

Note that SEM is analytically equal to CSEM in the unsupervised scenario ($E^t = \frac{1}{2}\ \forall t$). Consequently, SEM is equivalent to the unsupervised CSEM and consequently to ALDA.

5.7 CMAC and CSEM. CMAC and CSEM can not be reduced to the same class. If we compare the CMAC equations for the mean update, equations 2.15 and 2.16, with the CSEM equations, 2.9, we see that CMAC contains additional terms involving the mean of the opposite class that are not in the CSEM update rule. These terms appear because CMAC constrains the sum and the differences of the class means.

6 Discussion

Our contribution in this letter is twofold. First, we introduced the probabilistic methodology to include a reinforcement signal (RS) in the adaptive BCI cycle, and we show that under the assumption of an informative RS, this model allows a better estimation of the class probabilities (responsibilities) than the one obtained using only the task-related features. We also develop a novel update scheme for adaptive classification in BCI, CMAC, which can make use of the responsibilities computed using the proposed model and is able to outperform state of the art adaptive classifiers.

In the supervised scenario, CMAC is able to improve a standard supervised adaptation (supervised CSEM); even in the unsupervised scenario,

CMAC has the potential to outperform other methods. However, the ability to get a big improvement with regard to other methods clearly depends on the quality of the RS. Such an RS can range from button presses delivered by the user (supervised), to measured muscular activity in an hybrid BCI (Leeb et al., 2010) or an ErrP classifier if we consider a pure BCI setting. Clearly, using muscular activity to detect erroneous performance can provide an accurate RS. It is important to note that in previous work (Llera et al., 2011), we reported the possibility of relatively high ErrP classification rates (80% of mean accuracy across eight subjects), which agrees with the results presented previously by other researchers (Blankertz et al., 2004; Ferrez et al., 2008). Furthermore, Ferrez et al. (2008) also showed the high stability in ErrP detection across sessions. This facts make this kind of RS an optimal candidate to include in online BCI experiments.

The improvement reported using the CMAC algorithm is due not only to the accuracy of the RS but also to the introduction of an extra parameter β_- controlling the change in the difference between the means of the feature distributions. The optimal value for this parameter is clearly dependent on the accuracy of the estimated responsibilities. However, the simulations performed showed that the choice of this parameter value is not critical. In fact, we observed that setting $\beta_\Sigma = 0$, $\beta_+ = \beta_\mu \in [00.1]$, $\beta_- \in [0, 0.005]$, CMAC showed no significant difference with regard to Pmean independent of the accuracy of the estimated responsibilities. For values $\beta_- \in [0.005, 0.03]$, the improvement with regard to Pmean was proportional to the accuracy of the estimated responsibilities, and values $\beta_- > 0.03$ produced a decrease on performance with regard to Pmean. The improvement with regard to Pmean was obtained independent of β_+, which confirms the robustness of the method.

It is clear that the optimal choice of the learning rates is important for obtaining good results for all subjects and all methods. There exist methods that can automatically adapt the learning rate to an optimal value and make a trade-off between accuracy for stationary data (low learning rate) and adaptivity to change (large learning rates) (Heskes & Kappen, 1991, 1992). We believe that such methods should be integrated in the adaptive BCI methodology.

An open question of considerable importance is how to generalize the proposed adaptive BCI methodology to nonbinary tasks. Such learning tasks are more complex, since the error signal will indicate that an error has occurred but will not provide information on what the correct output should have been. This type of learning paradigm is called reinforcement learning (Rescorla, 1967; Sutton & Barto, 1998; Dayan & Abbott, 2001). An important future research direction is to integrate these reinforcement learning methods in online adaptive BCI.

Finally, due to its generality, the proposed methodology has a broader application not restricted to BCI, for instance, in the construction of adaptive spam filters (Zhou, Mulekar, & Nerellapalli, 2005). In this environment, the

RS could be represented by the user getting a file out of or in the spam folder.

Acknowledgments

We gratefully acknowledge the support of the Brain-Gain Smart Mix Programme of the Netherlands Ministry of Economic Affairs and the Netherlands Ministry of Education, Culture and Science.

References

Bishop, C. M. (2007). *Pattern recognition and machine learning*. New York: Springer.

Blankertz, B., Dornhege, G., Schäfer, C., Krepki, R., Kohlmorgen, J., Müller, K. R., et al. (2004). Boosting bit rates and error detection for the classification of fast-paced motor commands based on single-trial EEG analysis. *IEEE Transactions on Biomedical Engineering*, *51*, 993–1002.

Blankertz, B., Kawanabe, M., Tomioka, R., Hohlefeld, F., Nikulin, V., & Müller, K. R. (2008). Invariant common spatial patterns: Alleviating nonstationarities in brain-computer interfacing. In J. C. Platt, D. Koller, Y. Singer, & S. Roweis (Eds.), *Advances in neural information processing systems*, *20* (pp. 113–120). Cambridge, MA: MIT Press.

Blumberg, J., Rickert, J., Waldert, S., Schulze-Bonhage, A., Aertsen, A., & Mehring, C. (2007). Adaptive classification for brain computer interfaces. In *Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (pp. 2536–2539). Piscataway, NJ: IEEE Press.

Chavarriaga, R., Ferrez, P. W., & Millán, J. del R. (2007). To err is human: Learning from error potentials in brain-computer interfaces. In *Proceedings of the International Conference on Cognitive Neurodynamics* (pp. 777–782). New York: Springer.

Dayan, P., & Abbott, L. F. (2001). *Theoretical neuroscience: Computational and mathematical modeling of neural systems*. Cambridge, MA: MIT Press.

Ferrez, P. W. (2007). *Error-related EEG potentials in brain-computer interfaces*. Unpublished doctoral dissertation, Ecole polytechnique fédérale de Lausanne EPFL.

Ferrez, P. W., & Millán, J. del R. (2008). Error-related EEG potentials generated during simulated brain-computer interaction. *IEEE Transactions on Biomedical Engineerings*, *55*, 923–929.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition*. Orlando, Fl: Academic Press.

Gan, J. G. (2006). Self-adapting BCI based on unsupervised learning. In *Proceedings of the 3rd International Workshop on Brain-Computer Interfaces* (pp. 50–51). New York: Springer.

Hasan, B.A.S., & Gan, J. Q. (2009a). Unsupervised adaptive GMM for BCI. In *Proceedings of the International IEEE EMBS Conf. on Neural Engineering* (pp. 295–298). New York: Springer.

Hasan, B.A.S., & Gan, J. Q. (2009b). *Sequential EM for unsupervised adaptive gaussian mixture model based classifier*. New York: Springer.

Heskes, T., Gielen, S., & Kappen, H. J. (1991). Neural networks learning in a changing environment. In T. Kohonen, K. Makisara, O. Simula, & J. Kangas (Eds.), *Artificial neural networks* (vol. 1, pp. 15–20). Amsterdam: North-Holland.

Heskes, T., & Kappen, H. J. (1991). Learning processes in neural networks. *Physical Review A*, *44*, 2718–2726.

Heskes, T., & Kappen, H. J. (1992). Learning-parameter adjustment in neural networks. *Physical Review A*, *45*, 8885–8893.

Kawanabe, M., Krauledat, M., & Blankertz, B. (2006). A Bayesian approach for adaptive BCI classification. *In Proceedings of the 3rd International Brain-Computer Interface Workshop and Training Course 2006* (pp. 54–55). Graz: Verlag der Technischen Universität Graz.

Krauledat, M. (2008). *Analysis of nonstationarities in EEG signals for improving brain-computer interfaces performance*. Unpublished doctoral dissertations, Technische Universität Berlin.

Leeb, R., Sagha, H., & Chavarriaga, R. and Millan, J. del R. (2010). Multimodal fusion of muscle and brain signals for a hybrid-BCI. In *Proceedings of the 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (pp. 4343–4346). Piscataway, NJ: IEEE.

Lemm, S., Blankertz, B., Curio, G., & Müller, K. R. (2005). Spatio-spectral filters for improved classification of single trial EEG. *IEEE Trans. Biomed. Eng*, *52*, 1541–1548.

Li, Y., & Guan, C. (2006). An extended EM algorithm for joint feature extraction and classification in brain-computer interfaces. *Neural Computation, 18*, 2730–2761.

Liu, G., Huang, G., Meng, J., Zhang, D., & Zhu, X. (2010). Improved GMM with parameter initialization for unsupervised adaptation of brain-computer interface. *International Journal for Numerical Methods in Biomedical Engineering*, *26*, 681–691.

Llera, A., van Gerven, M.A.J., Gómez, V., Jensen, O., & Kappen, H. J. (2011). On the use of interaction error potentials for adaptive brain computer interfaces. *Neural Networks*, *24*, 1120–1127.

Millán, J. del R. (2004). On the need for on-line learning in brain-computer interfaces. *IEEE Proc. of the Int. Joint Conf. on Neural Networks*, *4*, 2877–2882.

Müller-Gerking, J., Pfurtscheller, G., & Flyvbjerg, H. (1998). Designing optimal spatial filters for single-trial EEG classification in a movement task. *Clinical Neurophysiology*, *110*, 787–798.

Ramoser, H., Müller-Gerking, J., & Pfurtscheller, G. (1998). Optimal spatial filtering of single trial EEG during imagined hand movement. *IEEE Transactions on Rehabilitation and Engineering*, *8*, 441–446.

Rescorla, R. A. (1967). Pavlovian conditioning and its proper control procedures. *Psychological Review*, *74*, 71–80.

Reuderink, B., Farquhar, J., Poel, M., & Nijholt, A. (2011). A subject-independent brain-computer interface based on smoothed, second-order baselining. In *Proceedings of the 33st Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (pp. 4600–4604). Piscataway, NJ: IEEE.

Samek, W., Vidaurre, C., Müller, K. R., & Kawanabe, M. (2012). Stationary common spatial patterns for brain-computer interfacing. *Journal of Neural Engineering*, *9*, 0260134.

Shenoy, P., Krauledat, M., Blankertz, B., Rao, R. P., & Müller, K. R. (2006). Towards adaptive classification for BCI. *Journal of Neural Engineering*, *3*, R13–R23.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.

Sykacek, P., Roberts, S. J., & Stokes, M. (2004). Adaptive BCI based on variational Kalman filtering: An empirical evaluation. *IEEE Transactions on Biomedical Engineering*, *51*, 719–727.

Tomioka, R., Hill, J. N., Blankertz, B., & Aihara, K. (2006). Adapting spatial filter methods for nonstationary BCIs. In *Proceedings of 2006 Workshop on Information-Based Induction Sciences* (pp. 65–70). Munich: Max Planck Society.

van Gerven, M., & Jensen, O. (2009). Attention modulations of posterior alpha as a control signal for two-dimensional brain-computer interfaces. *Journal of Neuroscience Methods*, *179*, 78–84.

Vidaurre, C., & Blankertz, B. (2010). Towards a cure for BCI illiteracy. *Brain Topogr.*, *23*, 194–198.

Vidaurre, C., Kawanabe, M., von Bünau, P., Blankertz, B., & Müller, K. R. (2010). Toward an unsupervised adaptation of LDA for brain-computer interfaces. *IEEE Trans. Biomed. Eng.*, *58*, 587–597.

Vidaurre, C., Sannelli, C., Müller, K. R., & Blankertz, B. (2010). Machine-learning based co-adaptive calibration. *Neural Computation*, *23*(3), 791–816.

Vidaurre, C., Schlöogl, A., Cabeza, R., Scherer, R., & Pfurtscheller, G. (2006). A fully on-line adaptive BCI. *IEEE Transactions on Biomedical Engineering*, *6*, 1214–1219.

von Bünau, P., Meinecke, F. C., Scholler, S., & Müller, K. R. (2010). Finding stationary brain sources in EEG data. In *Proceedings of the 32nd Annual Conference of the IEEE EMBS*, 2010 (pp. 2810–2813). Piscataway, NJ: IEEE.

Wolpaw, J. R., Birbaumer, N., McFarland, D. J., Pfurtscheller, G., & Vaughan, T. M. (2002). Brain-computer interfaces for communication and control. *Clinical Neurophysiology*, *6*, 767–791.

Zhou, Y., Mulekar, M. S., & Nerellapalli, P. (2005). Adaptive spam filtering using dynamic feature space. In *Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence* (pp. 302–309). Piscataway, NJ: IEEE.

Received December 10, 2011; accepted May 9, 2012.