Classification of neural signals from sparse autoregressive features

Diego Vidaurre, Concha Bielza, Pedro Larrañaga

This paper introduces a signal classification framework that can be used for brain–computer interface design. The actual classification is performed on sparse autoregressive features. It can use any well-known classification algorithm, such as discriminant analysis, linear logistic regression and support vector machines. The autoregressive coefficients of all signals and channels are simultaneously estimated by the group lasso, and the estimation is guided by the classification performance. Thanks to the variable selection capability of the group lasso, the framework can drop individual autoregressive coefficients that are useless in the prediction stage. Also, the framework is relatively insensitive to the chosen autoregressive order. We devise an efficient algorithm to solve this problem. We test our approach on Keirn and Aunon's data, used for binary classification of electroencephalogram signals, achieving promising results.

C

1. Introduction

Brain–computer interfaces (BCIs) establish a direct communication between the brain and some external device, where brain activity is decoded to control the device. This is a very promising application of machine learning techniques that, for example, can help to greatly improve the quality of life of disabled people.

Of BCI technologies, non-invasive BCIs, which use neuroimaging inputs collected without entering the body, are of particular interest. Some of the possible data sources are electroencephalography (EEG) data, magnetoencephalography (MEG) data and functional magnetic resonance imaging (fMRI) data. In this paper, we focus on binary classification with EEG inputs. Therefore, data usually consist of a set of signals, each collected from a specific location on the scalp (called a channel). For a general review of EEG-based classification, see, for example, [1].

Usually, some feature extraction procedure is used to assemble the classifying predictors from the raw signals. To number a few possibilities, we have time–frequency features [2], power spectral density values [3] or the fitted autoregressive (AR) coefficients [4]. We focus on AR coefficients.

We assume that each instance of the data set is constituted by a signal or a simultaneously recorded set of signals (each corresponding to a different channel). Typically, a vector of AR linear coefficients is estimated separately for each single signal by, for example, least squares or Burg's algorithm [5]. These coefficients would be the inputs for the subsequent classifier. When more than one channel is available, the AR coefficients for all channels are concatenated to build a single instance. Hence, if a p-order AR model is fitted for each signal, we have p predictors per channel. Alternatively, the signal can be divided into various segments so that a p-order AR model is fitted for each segment. In this case, we would have a number of predictors equal to p multiplied by the number of segments, multiplied by the number of channels. This is done separately for each instance.

Huan and Palaniappan [6] compare these two methods, estimating the AR coefficients using both the least squares method and Burg's algorithm. All their estimates are of sixth-order AR coefficients, which have been reported in the literature to empirically produce good results. They use either linear discriminant analysis or a multilayer perceptron for the binary classification step, concluding that the best feature extraction approach is the simplest least squares method of fitting AR models for entire signals. No variable selection is performed. Unfortunately, this type of methods entails two major drawbacks. First, the order of the AR models is fixed beforehand, without considering the data. Also, the same order is used for all channels. Second, the AR coefficients are estimated by exclusively minimizing the AR prediction error, regardless of the classification performance.

This paper aims to overcome these pitfalls by choosing an initial arbitrarily high AR order and then looking for a sparse AR solution. This way, there is no need for specifying an exact AR order and the relevant predictors in the autoregressive model are automatically selected in a flexible way. The lasso [7] is a popular method for simultaneous regression and variable selection. Hsu et al. [8] apply the lasso to AR models.

The novelty of our approach is that, instead of making a separate AR estimate for each signal, we estimate the AR coefficients for all signals altogether. The objective is to discard (or select) the same variables for all signals. The selected variables can be different for each channel. We use the group lasso [9], which can discard or select entire groups by means of a block L_1-penalization, to generate a sparse solution.

There are efficient algorithms to solve the group lasso, either finding the whole regularization path [9] or computing the solution for a grid of different regularization parameter values [10]. In this paper, we devise an efficient LARS-type algorithm [9,11] based on multiresponse linear regression that provides computational advantages for this particular problem. Whatever algorithm we use, we select the group lasso solution that maximizes some classifier-related measure. Since the classifier somehow guides the AR coefficient estimation, it is with the second aforementioned issue that we are concerned here.

Our method can be deemed a wrapper method. Wrapper methods are often computationally expensive. In this case, though, the number of predictors (selected AR coefficients) is moderate, so the computational cost is affordable.

The rest of the paper is organized as follows. Section 2 introduces the spirit of the methodology. Section 3 details the simplest case, when there is only one channel. Section 4 extends the method to the multiple channel scenario. Section 5 introduces a LARS-type algorithm to efficiently approximate the solution of the proposed problem. Section 6 deals with some computational details. Section 7 describes the set of experiments used to test the algorithm. Finally, Section 8 sums up the paper.

2. Basic methodology

We consider N signals $\boldsymbol{z}_i = (z_{i1}, \dots, z_{iT})^t$, $i \in \{1, \dots, N\}$, each labeled as $c_i \in \{0,1\}$. Let us denote the class vector as $\boldsymbol{c} \in \{0,1\}^N$. We want to obtain a classifier that assigns any future signal $\boldsymbol{z}_i$, $i > N$, to a class in $\{0,1\}$.

The autoregressive p-order model presumes, for each signal $\boldsymbol{z}_i$, that

$$z_{ij} = \beta_{i0} + \sum_{k=1}^{p} \beta_{ik} z_{i(j-k)} + \epsilon_j, \quad j \in \{p+1, \dots, T\}, \tag{1}$$

where ϵ_j is Gaussian white noise. Given some estimator $\hat{\boldsymbol{\beta}}_i = (\hat{\beta}_{i0}, \hat{\beta}_{i1}, \dots, \hat{\beta}_{ip})^t$, the squared sum of autoregressive errors is defined for $\boldsymbol{z}_i$ as

$$SSE(\hat{\boldsymbol{\beta}}_i, \boldsymbol{z}_i) = \sum_{j=p+1}^{T} \left(z_{ij} - \hat{\beta}_{i0} - \sum_{k=1}^{p} \hat{\beta}_{ik} z_{i(j-k)} \right)^2. \tag{2}$$

Now, let us consider a classifier ψ and a function $f_\psi(\cdot,\cdot)$, which returns some classifier-related fitness measure, and whose arguments are, respectively, a set of inputs and a set of responses. Let the $N \times p$ matrix $\boldsymbol{B} = [\hat{\boldsymbol{\beta}}_1^t, \dots, \hat{\boldsymbol{\beta}}_N^t]$ stack the N sparse vectors of autoregressive coefficients, and let $J(\cdot)$ be some function monotonic on the complexity of $\boldsymbol{B}$.

Given a sufficiently high order p, we can formulate a multicriterion problem to jointly estimate $\boldsymbol{B}$, whose criteria are the classifier accuracy, the autoregressive SSE and the complexity of $\boldsymbol{B}$. For some $\lambda \succcurlyeq \boldsymbol{0}$, which denotes a componentwise inequality, a scalarized version of this problem is given by

$$\hat{\boldsymbol{B}} = \underset{\boldsymbol{B}}{\operatorname{argmin}} - \lambda_1 f_\psi(\boldsymbol{B}, \boldsymbol{c}) + \lambda_2 \sum_{i=1}^{N} SSE(\hat{\boldsymbol{\beta}}_i, \boldsymbol{z}_i) + \lambda_3 J(\boldsymbol{B}). \tag{3}$$

By choosing $J(\cdot)$ to promote sparsity, we will expect to discard those coefficients that are useless for the estimation. The discarded coefficients should be the same for all vectors $\hat{\boldsymbol{\beta}}_i$, $i \in \{1, \dots, N\}$.

Solving for all $\lambda \succcurlyeq \boldsymbol{0}$ we obtain all the Pareto optimal solutions to the multicriterion problem (and also some non-optimal solutions). From the Pareto optimal set of solutions, that forms the optimal trade-off surface, we would select, for example, the solution that maximizes the classification accuracy over some separated (validation) data set. So, $\hat{\boldsymbol{B}}$ is the input of the classifier and is chosen to optimize the expected classification accuracy.

For $\lambda_1 > 0$, the scalarized problem (3) is in general non-convex. In addition, we need to evaluate it for a 2D grid of values of $\boldsymbol{\lambda}$ (note that the relative magnitudes of the components of $\boldsymbol{\lambda}$ are all that matters). This approach is computationally unaffordable. Instead, we resort to the more restricted problem

$$\hat{\boldsymbol{B}} = \underset{\boldsymbol{B}}{\operatorname{argmin}} \sum_{i=1}^{N} SSE(\hat{\boldsymbol{\beta}}_i, \boldsymbol{z}_i) + \lambda J(\boldsymbol{B}), \tag{4}$$

where we let $\lambda_1 = 0$ and $\lambda = \lambda_3/\lambda_2$. From the set of solutions corresponding to different values $\lambda \geq 0$, we will select the best solution according to $f_\psi(\boldsymbol{B}, \boldsymbol{c})$. With this approach, we only need to search in a one-dimensional grid of λ hyperparameters, and, also, the problem is convex for a proper choice of $J(\cdot)$.

Note that, even when the set of Pareto optimal solutions given by problem (4) is only a subset of the set of Pareto optimal solutions given by problem (3), this approach, unlike state-of-the-art methods, still considers $f_\psi(\cdot,\cdot)$ for the estimation of $\boldsymbol{B}$. Although in this paper we do not follow this road, a complete search on the optimal trade-off surface given by problem (3) could be conducted for moderate size problems and specific choices of the classifier ψ that preserve convexity. For further details about multicriterion optimization, see, e.g., [12].

In what follows, we give the specifics about the proposed method, including the choice of $J(\cdot)$.

3. Single channel classification

Let us define the following elements:

$$\boldsymbol{y} = (z_{1(p+1)}, \dots, z_{1T}, \dots, z_{N(p+1)}, \dots, z_{NT})^t,$$

$$\boldsymbol{X} = \begin{pmatrix}
1 & z_{11} & \dots & z_{1p} & 0 & 0 & \dots & 0 & \dots & 0 & 0 & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
1 & z_{1(T-p)} & \dots & z_{1(T-1)} & 0 & 0 & \dots & 0 & \dots & 0 & 0 & \dots & 0 \\
0 & 0 & \dots & 0 & 1 & z_{21} & \dots & z_{2p} & \dots & 0 & 0 & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & \dots & 0 & 1 & z_{2(T-p)} & \dots & z_{2(T-1)} & \dots & 0 & 0 & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & \dots & 0 & 0 & \dots & 0 & \dots & \dots & 1 & z_{N1} & \dots & z_{Np} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & \dots & 0 & 0 & \dots & 0 & \dots & \dots & 1 & z_{N(T-p)} & \dots & z_{N(T-1)}
\end{pmatrix},$$

$$\boldsymbol{\beta}^* = (\beta_{10}, \beta_{11}, \dots, \beta_{1p}, \dots, \beta_{N0}, \beta_{N1}, \dots, \beta_{Np})^t,$$

where vector $\boldsymbol{\beta}^*$ is the concatenation of the elements of $\boldsymbol{B}$. Therefore, if we include the intercept as an additional predictor for each signal, we have $\boldsymbol{y} \in \mathbb{R}^{Nq}$, $\boldsymbol{X} \in \mathbb{R}^{Nq \times N(p+1)}$ and $\boldsymbol{\beta}^* \in \mathbb{R}^{N(p+1)}$, where $q = T - p$. Otherwise, we have $\boldsymbol{y} \in \mathbb{R}^{Nq}$, $\boldsymbol{X} \in \mathbb{R}^{Nq \times Np}$ and $\boldsymbol{\beta}^* \in \mathbb{R}^{Np}$. In this paper, we choose to include the intercept.

Assuming that the N signals are independent, we can establish the linear relation

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}^* + \boldsymbol{\epsilon}, \tag{5}$$

where $\boldsymbol{\epsilon} = (\epsilon_{1(p+1)}, \dots, \epsilon_{1T}, \dots, \epsilon_{N(p+1)}, \dots, \epsilon_{NT})^t$ is Gaussian white noise.

We can impose an L_1-penalty that would drive some AR coefficients to zero (depending on some regularization parameter λ).

However, since we are interested in discarding the same coefficients for each signal, we use a group lasso penalty instead. The proposed formulation allows us to define p groups $\boldsymbol{\theta}_k = (\beta^*_{1k}, \ldots, \beta^*_{Nk})^t$, $k \in \{1, \ldots, p\}$, to estimate $\hat{\boldsymbol{\beta}}^*$ as the minimizer of

$$\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}^*\|_2^2 + \lambda \sum_{k=1}^{p} \|\boldsymbol{\theta}_k\|_2, \tag{6}$$

where $J(\cdot)$ is given by a sum of L_2-norms, and, thus, provides sparsity at a group level driving some groups $\boldsymbol{\theta}_k$ to exactly zero for all their components [9]. The intercepts $(\beta_{10}, \ldots, \beta_{N0})$ are not penalized.

This is a usual group lasso problem [9], and the complete exact regularization path can be obtained by group LARS [9,11] if $\boldsymbol{X}$ is orthogonal. Otherwise, the group LARS provides an approximation, which serves our purposes. From this approximated regularization path, we will select the solution $\hat{\boldsymbol{\beta}}^*$ that minimizes the estimated expected error with regard to ψ.

However, even though group LARS is a very efficient method, $\boldsymbol{X}$ can become a huge matrix and computation can be expensive unless we exploit its sparse structure. Similar arguments have been followed in the multiresponse regression literature. Below, we introduce a forward selection approach that considers this structure. This algorithm is based on group LARS and the *multiresponse sparse regression* algorithm [13].

Linear discriminant analysis (LDA) logistic regression (LR) or support vector machines (SVMs) are the options that we consider for the classifier ψ; see [14] for a general review. LDA and LR are linear in the most basic version, whereas SVM reaches nonlinearity by constructing a linear boundary in a transformed version of the feature space.

For LDA, $f_\psi(\hat{\boldsymbol{B}}, \boldsymbol{c})$ is naturally defined as the log-likelihood function

$$f_\psi(\hat{\boldsymbol{B}}, \boldsymbol{c}) = \sum_{i=1}^{N} \left(-\hat{\boldsymbol{\beta}}_i^t \hat{\boldsymbol{\Sigma}}^{-1} \hat{\boldsymbol{\mu}}_{c_i} + \frac{1}{2} \hat{\boldsymbol{\mu}}_{c_i}^t \hat{\boldsymbol{\Sigma}}^{-1} \hat{\boldsymbol{\mu}}_{c_i} + \log \hat{\pi}_{c_i} \right), \tag{7}$$

where $\hat{\boldsymbol{\mu}}_{c_i}$ is the mean of those vectors $\boldsymbol{\beta}_i$ whose class is c_i, $\hat{\boldsymbol{\Sigma}}$ is the common covariance matrix of $\hat{\boldsymbol{B}}$ and $\hat{\pi}_{c_i}$ is the estimated a priori probability of class c_i. We can estimate π_{c_i} as N_{c_i}/N, where N_{c_i} is the number of instances whose class is c_i. To compute (7), we remove the coefficients that correspond to dropped groups (since $\hat{\boldsymbol{\beta}}_i^*$ is sparse by groups). Note that this is necessary to compute $\hat{\boldsymbol{\Sigma}}$. Otherwise, matrix $\hat{\boldsymbol{B}}$ has columns with all elements equal to zero and is not full rank.

For LR, $f_\psi(\hat{\boldsymbol{B}}, \boldsymbol{c})$ can also be the log-likelihood function, defined as

$$f_\psi(\hat{\boldsymbol{B}}, \boldsymbol{c}) = \sum_{i=1}^{N} (c_i \hat{\boldsymbol{w}}^t \hat{\boldsymbol{\beta}}_i - \log(1 + e^{\hat{\boldsymbol{w}}^t \hat{\boldsymbol{\beta}}_i})), \tag{8}$$

where $\hat{\boldsymbol{w}} \in \mathcal{R}^p$ is the estimated vector of logistic regression coefficients, computed by the iteratively reweighted least squares (IRLS) algorithm.

In order to avoid overfitting, we select the solution of Eq. (6) that minimizes a penalized version of $f_\psi(\hat{\boldsymbol{B}}, \boldsymbol{c})$. In particular, since Eqs. (7) and (8) are loglikelihood functions, we can employ the *Akaike information criterion* [15]

$$AIC = -\frac{2}{N} f_\psi(\hat{\boldsymbol{B}}, \boldsymbol{c}) + 2\frac{df}{N}, \tag{9}$$

where df is the number of columns of $\hat{\boldsymbol{B}}$ with non-zero coefficients.

Finally, for SVM, $f_\psi(\hat{\boldsymbol{B}}, \boldsymbol{c})$ can be defined as some margin maximizing loss function. For convenience, we redefine the class to be in $\{-1, 1\}$. One possible formulation of the SVM estimates the separating hyperplane parameters $\boldsymbol{w} \in \mathbb{R}^{p^+}$ as the minimizer

$$\sum_{i=1} (1 - c_i h(\boldsymbol{\beta}_i)^t \boldsymbol{w})_+ + \alpha \|\boldsymbol{w}\|^2, \tag{10}$$

where $(\cdot)_+$ indicates the positive part, $h(\cdot)$ is some mapping function, p^+ is the dimension of the expanded feature space and $\alpha > 0$ is the regularization cost parameter. Hence, function $h(\cdot)$ gives the nonlinear power to the SVM. Typically, a kernel function that computes the distance between any two points in the expanded feature space defined by $h(\cdot)$ is all we need for an efficient computation. In this paper, we use a radial basis function kernel, whose corresponding feature space is a Hilbert space of infinite dimensions.

The entire regularization path for Eq. (10) can be computed with a small multiple of the computational cost of fitting an SVM model for a single α parameter [16]. From this regularization path, we select the model that minimizes the K-fold cross-validated error. Here, $f_\psi(\hat{\boldsymbol{B}}, \boldsymbol{c})$ can be the value of the left term of Eq. (10) that corresponds to the selected model. Since we use a cross-validated estimation of the loss function, we do not need to use a penalization such as AIC in Eq. (9).

Whereas the LDA formulation can be used for multiclass classification, the LR and SVM expressions are for binary classification but can be easily generalized. See [17], for example, for details about the multiclass SVM.

4. Multiple channel classification

An instance is usually defined as a group of signals instead of just one signal. For instance, an EEG instance is usually a set of signals recorded from different points on the scalp, that is, from different channels. Here, it makes sense to consider that the same AR coefficients are perhaps not appropriate for all channels.

Let us modify the above notation to accommodate this problem. We consider N sets of signals, each with M channels, and labeled as $\boldsymbol{c} \in \{0, 1\}^N$ as before. Now, each signal is denoted as $\boldsymbol{z}_{il} = (z_{il1}, \ldots, z_{ilT})$, $i \in \{1, \ldots, N\}$, $l \in \{1, \ldots, M\}$. We define

$$\boldsymbol{y} = (z_{11(p+1)}, \ldots, z_{11T}, \ldots, z_{1M(p+1)}, \ldots, z_{1MT}, \ldots, z_{NM(p+1)}, \ldots, z_{NMT})^t,$$

$$\boldsymbol{X}_{il} = \begin{pmatrix} 1 & z_{il1} & \cdots & z_{ilp} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & z_{il(T-p)} & \cdots & z_{il(T-1)} \end{pmatrix},$$

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{X}_{11} & \cdots & \boldsymbol{0} & \cdots & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{X}_{1M} & \cdots & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & \cdots & \boldsymbol{X}_{N1} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & \cdots & \boldsymbol{0} & \cdots & \boldsymbol{X}_{NM} \end{pmatrix},$$

$$\boldsymbol{\beta}^* = (\beta_{111}, \ldots, \beta_{11p}, \ldots, \beta_{1M1}, \ldots, \beta_{1Mp}, \ldots, \beta_{NM1}, \ldots, \beta_{NMp})^t,$$

where $\boldsymbol{y} \in \mathbb{R}^{NMq}$, $\boldsymbol{X}_{il} \in \mathbb{R}^{q \times p+1}$, $\boldsymbol{X} \in \mathbb{R}^{NMq \times NM(p+1)}$ and $\boldsymbol{\beta}^* \in \mathbb{R}^{MN(p+1)}$. As before, $q = T - p$. The same linear relation as Eq. (5) can be applied here.

Now, we have two choices to define the groups. We can either define p groups as

$$\boldsymbol{\theta}_k = (\beta_{11k}, \ldots, \beta_{1Mk}, \ldots, \beta_{N1k}, \ldots, \beta_{NMk})^t, \quad k \in \{1, \ldots, p\}, \tag{11}$$

or Mp groups as

$$\boldsymbol{\theta}_{lk} = (\beta_{1lk}, \ldots, \beta_{Nlk})^t, \quad l \in \{1, \ldots, M\}, \quad k \in \{1, \ldots, p\}. \tag{12}$$

Eq. (11) defines the same AR coefficients for all the channels, whereas Eq. (12) defines adaptive AR coefficients for each channel. The group lasso formulation in Eq. (6) is unchanged in the first case. In the second case, it becomes

$$\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}^*\|_2^2+\lambda\sum_{k=1}^{p}\sum_{l=1}^{M}\|\boldsymbol{\theta}_{lk}\|_2. \tag{13}$$

The solutions can be used as inputs for the classifier ψ as before, and the solution $\hat{\boldsymbol{\beta}}^*$ that maximizes $f_\psi(\hat{\boldsymbol{B}},\boldsymbol{c})$ is selected. $\hat{\boldsymbol{B}}$ is defined as $[\hat{\boldsymbol{\beta}}_1^T,\ldots,\hat{\boldsymbol{\beta}}_N^T]$, where $\hat{\boldsymbol{\beta}}_i=(\beta_{i10},\beta_{i11},\ldots,\beta_{i1p},\ldots,\beta_{iM0},\beta_{iM1},\ldots,\beta_{iMp})^t$, $i\in\{1,\ldots,N\}$.

Since there is no reason to assume that all the channels contribute equally, we prefer the second grouping approach.

5. An efficient LARS-type algorithm

In this section, we briefly describe the group LARS [11] and the multiresponse sparse regression algorithm [13]. We then introduce an efficient algorithm to find an approximate regularization path for Eqs. (6) and (13). Although we define the method for Eq. (13), it can be straightforwardly adapted to Eq. (6). We assume centered data (no intercepts) for notation simplicity.

The group LARS algorithm is an iterative procedure that adds a set of predictors, or group, to the model at each step. The group LARS starts with no groups. Firstly, it adds the group that is most correlated with the response to the active set of groups $\mathcal{A}$. The response is regressed on this group, moving the coefficients of this group towards the least squares solution until a new group reaches the same correlation with the vector of residuals as the active set. This new group is added to the active set $\mathcal{A}$. Now, the vector of residuals is regressed on the groups in $\mathcal{A}$, moving their coefficients towards the joint least squares solution until a new group reaches the same correlation with that vector of residuals as the active set. When the number of instances is greater than the number of predictors, this procedure is repeated until all predictors are in the model. With a small modification, the group LARS can find the entire regularization path of a group lasso problem if the design matrix is orthogonal. Otherwise, it provides an approximate solution.

The multiresponse sparse regression algorithm extends the (group) LARS algorithm to multiresponse linear regression by modifying the correlation criterion between the predictors and the current residual, which depends on multiple outputs.

In this paper, we derive an algorithm based on group LARS and multiresponse sparse regression for efficiently computing an approximate set of solutions of Eqs. (6) and (13). In the sequel, we will use the notation $\boldsymbol{X}_{il}$, $i\in\{1,\ldots,N\}$, $l\in\{1,\ldots,M\}$, defined in Section 4. Groups are defined in Eq. (12). The active set $\mathcal{A}$ is thus defined by a set of pairs (l,k), $l\in\{1,\ldots,M\}$, $k\in\{1,\ldots,p\}$. We also define $\boldsymbol{y}_{il}=(\boldsymbol{z}_{il(p+1)},\ldots,\boldsymbol{z}_{ilT})^t$ and $\boldsymbol{\beta}_{il}=(\beta_{il1},\ldots,\beta_{ilp})^t$.

The algorithm follows the group LARS steps described by Yuan and Lin [9], with some modifications. First, we devise a new correlation measure. We define the correlation of the lk-th group with the current residual as

$$\rho_{lk}=\sum_{i=1}^{N}(\boldsymbol{r}_{il}^t\boldsymbol{X}_{il}^{(k)})^2, \tag{14}$$

where $\boldsymbol{X}_{il}^{(k)}$ is the k-th column of $\boldsymbol{X}_{il}$ and $\boldsymbol{r}_{il}\in\mathbb{R}^q$ is the current residual for the i-th signal and the l-th channel, defined as

$$\boldsymbol{r}_{il}=\boldsymbol{y}_{il}-\boldsymbol{X}_{il}\hat{\boldsymbol{\beta}}_{il}. \tag{15}$$

Second, the joint least squares solution $\boldsymbol{d}_{il}\in\mathbb{R}^p$ is computed separately for the i-th signal and the l-th channel. Setting to zero the elements of $\boldsymbol{d}_{il}$ that are not in $\mathcal{A}$, we compute the remainder as

$$\boldsymbol{d}_{il}^{\mathcal{A}}=(\boldsymbol{X}_{il}^{\mathcal{A}^t}\boldsymbol{X}_{il}^{\mathcal{A}})^{-1}\boldsymbol{X}_{il}^{\mathcal{A}^t}\boldsymbol{r}_{il}, \tag{16}$$

where $\boldsymbol{X}_{il}^{\mathcal{A}}$ denotes the columns of $\boldsymbol{X}_{il}$ indexed by the active set $\mathcal{A}$. Hence, the regression coefficients for the i-th signal and the l-th channel are updated at each step as

$$\boldsymbol{\beta}_{il}=\boldsymbol{\beta}_{il}+\gamma\boldsymbol{d}_{il}, \tag{17}$$

where the $\gamma\in[0,1]$ constant is computed as

$$\gamma=\min_{\gamma_{lk}}\quad \text{s.t.}$$

$$\sum_{i=1}^{N}[\boldsymbol{X}_{il}^{(k)^t}(\boldsymbol{r}_{il}-\gamma_{lk}\boldsymbol{X}_{il}\boldsymbol{d}_{il})]^2=\sum_{i=1}^{N}[\boldsymbol{X}_{il'}^{(k')^t}(\boldsymbol{r}_{il'}-\gamma_{lk}\boldsymbol{X}_{il'}\boldsymbol{d}_{il'})]^2. \tag{18}$$

Here, (l',k') are a pair of indexes arbitrarily chosen from $\mathcal{A}$. Basic algebraic manipulations lead to

$$\gamma_{il}=\frac{-v+v'\pm\sqrt{v^2+v'^2-2vv'-4ub+4u'b'}}{2(u-u')}, \tag{19}$$

where we define

$$u=\sum_{i=1}^{N}(\boldsymbol{X}_{il}^{(k)^t}\boldsymbol{X}_{il}\boldsymbol{d}_{il})^2,\quad u'=\sum_{i=1}^{N}(\boldsymbol{X}_{il'}^{(k')^t}\boldsymbol{X}_{il'}\boldsymbol{d}_{il'})^2,$$

$$v=2\sum_{i=1}^{N}(\boldsymbol{X}_{il}^{(k)^t}\boldsymbol{r}_{il})(\boldsymbol{X}_{il}^{(k)^t}\boldsymbol{X}_{il}\boldsymbol{d}_{il}),\quad v'=2\sum_{i=1}^{N}(\boldsymbol{X}_{il'}^{(k')^t}\boldsymbol{r}_{il'})(\boldsymbol{X}_{il'}^{(k')^t}\boldsymbol{X}_{il'}\boldsymbol{d}_{il'}),$$

$$b=\sum_{i=1}^{N}(\boldsymbol{X}_{il}^{(k)^t}\boldsymbol{r}_{il})^2,\quad b'=\sum_{i=1}^{N}(\boldsymbol{X}_{il'}^{(k')^t}\boldsymbol{r}_{il'})^2.$$

Thus, the indexes $(l,k)\notin\mathcal{A}$ that minimize Eq. (18) correspond to the group that is added to the active set in the next iteration. As with group LARS, we now update the residual

$$\boldsymbol{r}_{il}=\boldsymbol{y}_{il}-\gamma\boldsymbol{X}_{il}\boldsymbol{\beta}_{il}\quad\forall il. \tag{20}$$

This procedure is repeated until $\gamma=1$.

It turns out that the group LARS solution is the same yielded for the multiresponse sparse regression algorithm. The same connection holds for the algorithm that we propose. Unlike group LARS, however, we do not need to store an $NMq\times NM(p+1)$ matrix in memory (in the multiple channel case), speeding up the computations. Since $\boldsymbol{X}$ is not an orthogonal matrix, this solution is only approximated. At the cost of storing the entire matrix $\boldsymbol{X}$ in memory, an exact group lasso regularization path can be computed following the algorithm described in recent work by Friedman et al. [18]. In this paper, however, we do not follow this approach.

6. Further computational issues

Sometimes a fast algorithm is needed to run in a real-time (online) environment with very limited computational resources available. Some BCI applications are of this type. Such an approach can be hard to apply in the high-dimensional setting. However, offline (previously stored) data can be used to simplify the online task. Offline data are usually available in the BCI field, where the device has to be trained for each subject prior to its real use. For instance, we can run the algorithm on the offline data set so as to select which channels and AR coefficient indexes are relevant. Afterwards, in the online phase, we can use least squares to estimate the AR coefficients that correspond to the previously selected channels and AR coefficient indexes.

This two-step procedure follows the spirit of the relaxed lasso [19]. The relaxed lasso firstly discovers the sparsity pattern by lasso. Then, either least squares or the lasso, with a small penalty (i.e., with no variable selection), is used just on the selected variables. Among other nice theoretical properties, the relaxed lasso is less biased than the original lasso.

7. Experiments

To test the proposed framework, we use the EEG data recorded by Zak Keirn at Purdue University [20]. The data is a collection of experiments on seven different subjects. Each subject is told to perform five different mental activities, say: stay relaxed, solve a mathematics problem, write a letter, mentally rotate a geometric figure and count a series of made-up numbers. Each subject repeats each task for a number of trials. Specifically, subjects 1, 3, 4 and 6 perform ten trials, subjects 2 and 7 perform five trials and subject 5 performs fifteen trials. Each trial lasts ten seconds with a sampling frequency of 250 Hz, that is, it has a total of 2500 sample points per channel. EEG signals were recorded from seven channels, from positions C3, C4, P3, P4, O1, O2 and EOG. The positions are defined by the 10–20 system of electrode placement. Recordings were made with reference to electrically linked mastoids A1 and A2. Keirn and Aunon [20] describe the collection procedure in more detail.

We use these data to test our signal classifier on the 10 possible pairwise activity combinations, that is, on ten binary classification problems. The ten pairwise combinations are listed in the first column of Table 1. Binary classification was performed on these data for example by Keirn and Aunon [20] and Huan and Palaniappan [6]. In both papers, the trials are divided into subperiods that make up the instances for classification. Keirn and Aunon [20] use two-second subperiods and train a unique model for all subjects. Huan and Palaniappan [6], instead, use half-second subperiods, and have 20 available instances for each subject, trial and activity, with $2500/20=125$ sample points per channel. Huan and Palaniappan [6] estimate individual models for each subject, using ten trials. Subjects performing only five trials are ignored, as are the last five trials of the subject who performs fifteen trials.

In this paper, like [6], we train individual models for each subject. To be able to use all available data in equilibrated conditions, we consider each group of five trials as a different data set. Thus, there are thirteen five-trials sets: two for subjects 1, 3, 4 and 6, one for subjects 2 and 7 and three for subject 5. We omitted one set (from subject 4) due to missing data. Therefore, for each pair of activities (each binary classification problem), we obtain twelve different, individual models.

We divide each trial into ten one-second subperiods of 250 sample points. This way, we have ten instances per trial. Since we have five trials for each of two activities, we have 100 instances for training and testing for each experiment. We use a maximum AR order of 15. Using the notation from previous sections, we have $T=250$, $N=100$, $M=7$, $p=15$ and $q=T-p=235$. In this setting, we test the basic methodology explained in Section 4, using Eq. (13), and the algorithm devised in Section 5. We compare our approach (using also this setting) with the best feature extraction method reported by [6], that is, the sixth-order AR coefficients computed by least squares. For the classification methods, we use LDA, LR and SVM.

On the one hand, Table 1 shows the mean classification accuracy for each method and each combination of activities, averaging across all thirteen five-trials sets. These results give an idea of the global performance of each method for each binary classification problem. On the other hand, Table 2 gives the best accuracy for each five-trials set, reporting which pairwise combination of activities and classification algorithm produced this result. The left columns show the best of our methods, and the right columns show the best of Huan and Palaniappan's methods. In a practical scenario, the pairwise combination of activities and the algorithm that best discriminate for a given subject would be chosen to implement the customized BCI device for this subject. All results are obtained by 5-fold cross-validation.

Table 1
Summary of the classification accuracy for each combination of activities and methods.

Activities	Accuracy $\pm$ std. deviation					
	sLDA	sLR	sSVM	LDA	LR	SVM
Relax,maths	0.70 ± 0.07	$\mathbf{0.73}\pm0.06$	$\mathbf{0.73}\pm0.10$	0.70 ± 0.06	0.69 ± 0.05	$\mathbf{0.73}\pm0.08$
Relax,letter	0.62 ± 0.08	0.63 ± 0.09	$\mathbf{0.68}\pm0.10$	0.63 ± 0.07	0.59 ± 0.07	$\mathbf{0.68}\pm0.08$
Relax,rotate	0.71 ± 0.10	0.74 ± 0.06	$\mathbf{0.78}\pm0.10^*$	0.70 ± 0.09	0.67 ± 0.07	0.73 ± 0.10
Relax,count	0.64 ± 0.12	0.65 ± 0.05	$\mathbf{0.71}\pm0.10$	0.66 ± 0.09	0.65 ± 0.07	0.70 ± 0.08
Maths,letter	0.65 ± 0.10	0.66 ± 0.07	$\mathbf{0.73}\pm0.10$	0.71 ± 0.06	0.67 ± 0.05	$\mathbf{0.73}\pm0.07$
Maths,rotate	0.69 ± 0.07	0.69 ± 0.09	$\mathbf{0.73}\pm0.08$	0.70 ± 0.06	0.65 ± 0.04	$\mathbf{0.73}\pm0.10$
Maths,count	0.67 ± 0.08	0.68 ± 0.09	$\mathbf{0.70}\pm0.10$	0.66 ± 0.09	0.64 ± 0.10	0.69 ± 0.08
Letter,rotate	0.75 ± 0.12	0.74 ± 0.10	$\mathbf{0.83}\pm0.12^*$	0.74 ± 0.10	0.70 ± 0.10	0.77 ± 0.14
Letter,count	0.66 ± 0.08	0.70 ± 0.10	$\mathbf{0.73}\pm0.12$	0.65 ± 0.07	0.61 ± 0.07	0.71 ± 0.08
Rotate,count	0.65 ± 0.10	0.65 ± 0.10	0.71 ± 0.10	0.67 ± 0.11	0.64 ± 0.08	$\mathbf{0.72}\pm0.10$

Table 2
Accuracy of the best combination of activities for each five-trials set.

Subject	Accuracy	Method	Activities	Accuracy	Method	Activities
1a	$\mathbf{0.93}\pm0.05$	sSVM	Maths,rotate	0.85 ± 0.06	LDA	Relax,count
1b	0.85 ± 0.06	sSVM	Relax,rotate	$\mathbf{0.89}\pm0.05$	LR	Relax,maths
2a	$\mathbf{0.92}\pm0.08$	sSVM	Relax,rotate	$\mathbf{0.92}\pm0.06$	SVM	Letter,rotate
3a	$\mathbf{0.86}\pm0.05^*$	sSVM	Letter,rotate	0.71 ± 0.11	LR	Relax,maths
3b	$\mathbf{0.93}\pm0.12^*$	sSVM	Letter,rotate	0.82 ± 0.08	SVM	Letter,rotate
4a	$\mathbf{0.94}\pm0.20^*$	sLR	Relax,maths	0.87 ± 0.09	SVM	Relax,rotate
5a	$\mathbf{0.99}\pm0.02^*$	sSVM	Letter,rotate	0.86 ± 0.10	SVM	Letter,rotate
5b	$\mathbf{0.81}\pm0.06^*$	sLDA	Letter,rotate	0.73 ± 0.09	LDA	Maths,letter
5c	0.75 ± 0.60	sLR	Maths,count	$\mathbf{0.77}\pm0.08$	SVM	Relax,rotate
6a	0.88 ± 0.10	sSVM	Relax,rotate	$\mathbf{0.95}\pm0.04^*$	SVM	Letter,rotate
6b	$\mathbf{0.96}\pm0.20$	sLDA	Relax,rotate	0.9 ± 0.04	SVM	Relax,rotate
7a	$\mathbf{0.92}\pm0.02$	sSVM	Relax,rotate	0.9 ± 0.08	SVM	Letter,rotate

In both tables, the methods related to our approach are referred to as sparse LDA (sLDA), sparse logistic regression (sLR) and sparse SVM (sSVM). The methods related to the AR coefficients feature extraction approach are referred to as LDA, LR and SVM. Best results are highlighted. Statistical significance is checked by means of the t-test, so that the symbol $*$ is added when the difference between the best and the second best method is statistically significant with a significance level of 0.05. In Table 2, each five-trials set is identified by a number indicating the subject (1,…,7) and a letter (a, b) indicating the five-trials set within this subject.

As observed, the devised method outperforms the best method reported by [6] in most experiments. The biggest differences can be observed in Table 2, where the best pair of activities and method is selected for each five-trials set. Interestingly, the SVM classifier offers the best results, possibly indicating that, in this scenario, the classification can be enhanced by appropriate non-linear modeling and variable (channel) interaction.

8. Discussion

In this paper, we have proposed a new feature extraction method based on sparse autoregressive features for multiple signal classification. We have applied the method, together with different classification algorithms, to an EEG signal classification problem, and compared its performance to a state-of-the-art AR feature extraction approach.

The performance benefits from the fact that model selection is guided by some classification-related measure. Moreover, we do not need to previously estimate the order of the AR models, because, starting from a high enough order, model selection is completely data driven. Finally, thanks to regularization, our approach is applicable to data sets with few instances, whereas other methods might suffer from overfitting in the same scenario.

Note that the proposed approach can be classed in the semi-supervised classification paradigm. In semisupervised classification, there is typically a considerable amount of data, but only a portion is labeled. To make the most of the data, it is beneficial to also use the information provided by the unlabeled data. In our particular case, if we have a collection of unlabeled EEG signals, they can be included in the group lasso estimation of the AR coefficients (Eqs. (6) and (13)), even though the posterior model selection relies on a fully supervised classification algorithm, that is, only on the labeled data.

Acknowledgments

This research was partially supported by the Spanish Ministry of Science and Innovation, projects TIN2010-20900-C04-04, Consolider Ingenio 2010-CSD2007-00018 and Cajal Blue Brain.

References

[1] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, B. Arnaldi, A review of classification algorithms for EEG-based brain–computer interfaces, J. Neural Eng. 4 (2007) 1–13.

[2] T. Wang, J. Deng, B. He, Classifying EEG-based motor imagery tasks by means of time-frequency synthesized spatial patterns, Clin. Neurophysiol. 115 (2004) 2744–2753.

[3] S. Chiappa, S. Bengio, HMM and IOHMM modeling of EEG rhythms for asynchronous BCI systems, in: European Symposium on Artificial Neural Networks, 2004, pp. 199–204.

[4] W.D. Penny, S.J. Roberts, E.A. Curran, M.J. Stokes, EEG-based communication: a pattern recognition approach, IEEE Trans. Rehabil. Eng. 8 (2000) 214–215.

[5] J.P. Burg, Maximum entropy spectral analysis, in: 37th Meeting of the Society of Exploration Geophysics, 1967.

[6] N.J. Huan, R. Palaniappan, Neural network classification of autoregressive features from electroencephalogram signals for brain–computer interface design, J. Neural Eng. 1 (2004) 142–150.

[7] R. Tibshirani, Regression shrinkage and selection via the lasso, J. R. Stat. Soc. Ser. B 58 (1996) 267–288.

[8] N.J. Hsu, H.L. Hung, Y.M. Chang, Subset selection for vector autoregressive processes using lasso, Comput. Stat. Data Anal. 52 (2008) 3645–3657.

[9] M. Yuan, Y. Lin, Model selection and estimation in regression with grouped variables, J. R. Stat. Soc. Ser. B 70 (2006) 53–71.

[10] L. Meier, S. van de Geer, P. Bühlmann, The group lasso for logistic regression, J. R. Stat. Soc. Ser. B 70 (2008) 53–71.

[11] B. Efron, I. Johnstone, T. Hastie, R. Tibshirani, Least angle regression, Ann. Stat. 32 (2) (2004) 407–499.

[12] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.

[13] T. Similä, J. Tikka, Common subset selection of inputs in multiresponse regression, in: International Joint Conference on Neural Networks, 2006, pp. 1908–1915.

[14] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Predictions, second edition, Springer Verlag, 2008.

[15] H. Akaike, A new look at the statistical model identification, IEEE Trans. Autom. Control 19 (6) (1974) 716–723.

[16] T. Hastie, S. Rosset, R. Tibshirani, J. Zhu, The entire regularization path for the support vector machine, J. Mach. Learning Res. 5 (2004) 1391–1415.

[17] Y. Lee, Y. Lin, G. Wahba, Multicategory support vector machines. Theory and application to the classification of microarray data and satellite radiance data, J. Am. Stat. Assoc. 99 (2004) 67–81.

[18] J. Friedman, T. Hastie, R. Tibshirani, A Note on the Group Lasso and a Sparse Group Lasso. Technical Report, Stanford University, 2010.

[19] N. Meinshausen, Lasso with relaxation, Comput. Stat. Data Anal. 52 (2007) 374–393.

[20] Z.A. Keirn, J.I. Aunon, A new mode of communication between man and his surroundings, IEEE Trans. Biomed. Eng. 37 (1990) 1209–1214.