Voxel Selection in fMRI Data Analysis Based on Sparse Representation

Yuanqing Li, Praneeth Namburi[†], Zhuliang Yu, Cuntai Guan[♯], Jianfeng Feng[‡], Zhenghui Gu

School of Automation Science and Engineering, Southchina University of Technology, Guangzhou, 510640, China

[†]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639689

[♯]Institute for Infocomm Research, Singapore 119613

[‡]Computer Science and Mathematics Department, Warwick University, UK

Correspondence to: Dr. Yuanqing Li, E-mail: auyqli@scut.edu.cn

Abstract

This paper proposes an iterative sparse representation-based detection algorithm for voxel selection in fMRI data. In each iteration of this algorithm, we solve a linear programming problem and obtain a sparse weight vector. The final weight vector is the mean of those obtained in all iterations. The characteristics of our algorithm are: (i) The weight vector (output) is sparse; (ii) The magnitude of each entry of the weight vector represents the significance of its corresponding variable or feature in classification or regression problem; (iii) Due to the convergence of this algorithm, a stable weight vector can be obtained. To demonstrate the validity of our algorithm and illustrate its application, we apply this algorithm to the Pittsburgh Brain Activity Interpretation Competition (PBAIC) 2007 functional magnetic resonance imaging (fMRI) data set for selecting the voxels which are the most relevant to the *tasks* of the subjects, computed during the experiment. The above mentioned characteristics of our algorithm are analyzed for this data set. Furthermore, compared with the baseline method, general linear model (GLM)-based statistical parametric mapping (SPM), our method shows significantly better performance for voxel selection for this data set.

Keywords: Functional magnetic resonance imaging (fMRI), voxel selection, sparse representation, statistical parametric mapping (SPM), prediction.

I. INTRODUCTION

In functional magnetic resonance imaging (fMRI), an fMRI scanner measures the blood-oxygenation-level dependent (BOLD) signal at all points in a three dimensional grid, or image of the brain. The cells within this three-dimensional image are known as voxels. A typical fMRI data set is composed of the time series (BOLD signals) of tens of thousand voxels. High dimensionality is a characteristic of fMRI data. Therefore, voxel selection plays an important role in fMRI data analysis because of: (i) heavy computation burden; (ii) un-correlation (or redundancy) of a large number of voxel time series with respect to the stimulus/task presented to the subject. Much of current fMRI research such as identifying brain regions activated in response to some task or stimulus is related to voxel selection.

Voxel selection can be performed according to the characteristics of the stimulus and the brain functional areas. For example, if image stimulus is presented, then voxels can be selected from the areas in visual cortex [1]. One class of voxel selection methods are based on statistical test/statistics that find brain regions with statistically significant response. A typical example is statistical parametric mapping (SPM) based on general linear model (GLM). SPM is a powerful tool for the analysis of fMRI data including voxel selection [2]-[4]. The second class of voxel selection methods are based on the correlation between the voxel time series and the time series of the task or stimulus [5]. Correlation method, classifier-based method [6], multiple regressor model [7], as well as least square regression with L_2 (ridge) and L_1 (Lasso) regularization [8] can be categorized in the second class. In [31], an elastic net regression technique, which achieves both sparsity and the grouping effect by using a weighted combination of $1-$norm and $2-$norm penalties on top of the least-squares problem, was applied to the analysis of the fMRI data set of Pittsburgh Brain Activity Interpretation Competitions (PBAIC) 2007. Through simultaneously considering sparsity and the grouping effect, the authors demonstrated the distributed nature of neural function and the importance of localized clusters of activity.

In this paper, we present a sparse representation based method for voxel selection in fMRI data.

The sparse representation of signals can be modeled by

$$\mathbf{y} = \mathbf{A}\mathbf{w}, \tag{1}$$

where $\mathbf{y} \in R^N$ is a given signal vector, $\mathbf{A} \in R^{N \times M}$ $(N < M)$ is a basis matrix. When the model (1) is used for fMRI data analysis, $\mathbf{A}$ is a data matrix of which each column is a time series of a voxel, and $\mathbf{y}$ is a transformed stimulus/task function which is obtained by convolving a stimulus/task function with a hemodynamical response function.

The task of sparse representation is to find a solution $\mathbf{w} \in R^M$ of (1) such that this solution is as sparse as possible. In many references such as [10], a basis pursuit (BP) algorithm was presented, in which a sparse solution (i.e. 1-norm solution) can be found by solving the following optimization problem.

$$\min ||\mathbf{w}||_1, \;\; s.\ t.\ \ \mathbf{A}\mathbf{w} = \mathbf{y}, \tag{2}$$

where 1-norm $||\mathbf{w}||_1$ is defined as $\sum\limits_{i=1}^{M} |w_i|$.

Setting $\mathbf{w} = \mathbf{u} - \mathbf{v}$, where $\mathbf{u}$, $\mathbf{v} \in R^M$ are nonnegative, (2) can be converted to the following equivalent linear programming problem,

$$\min \sum_{i=1}^{M} (u_i + v_i), \text{ subject to } [\mathbf{A}, -\mathbf{A}][\mathbf{u}^T, \mathbf{v}^T]^T = \mathbf{y},\ \mathbf{u} \geq 0,\ \mathbf{v} \geq 0. \tag{3}$$

The solution of a linear programming problem is generally unique [18], which can be obtained by standard softwares. In this paper, all linear programming problems are solved using Matlab function “linprog”.

Sparse representation of signals has received a great deal of attention in recent years (e.g. [10]-[15]). For instance, Donoho & Elad discussed optimal sparse representation in general (non-orthogonal) dictionaries via l_1 minimization [16]. In practical applications, sparse representation can be used in underdetermined blind source separation (BSS), which is difficult to deal with using a standard independent component analysis (ICA) method [17]-[22]. Basis pursuit is also an important application of sparse representation [10], [16]. Recently, it has been found that Model (2) has applications in feature selection and detection. In [21], (2) was successfully used for cross-modal localization of sound-related region in the video where $\mathbf{A}$ was constructed from the video and $\mathbf{y}$ was constructed from the accompanying audio.

A related method is 1-norm support vector machine (SVM). Similar to the BP algorithm, 1-norm SVM solves a linear programming problem to obtain a sparse solution. Thus it is also called sparse SVM [23]-[28]. 1-norm SVM has potential applications in feature selection including dimension reduction [23], detection of region of interest of images [24], detection of machine damage or highlighting abnormal features (localization) in medical data [28]. There exist differences between the models (2) and 1-norm SVM. For instance, when 1-norm SVM and model (2) are used for the same data set, there are more variables and constraints for 1-norm SVM than for (2); this implies a heavier computational burden. Another related method is Lasso regularization, which also appears in potential SVM [30], [29]. Compared with Lasso method or potential SVM with quadratic objective functions, (2) can be converted into a standard linear programming problem and has computational advantage especially when the number of the variables is extremely large.

In the following, we compare the model (1) with the GLM model, and analyze the difference between the two models. GLM model is represented by

$$\mathbf{x}_i = \mathbf{G}\beta_i + \mathbf{e}_i, \tag{4}$$

where $\mathbf{x}_i \in R^N$ is a time series of the i-th voxel, $\mathbf{G} \in R^{N\times K}$ is called a design matrix, $\beta_i \in R^K$ is an unknown parameter vector to be estimated for each voxel, $\mathbf{e}_i \in R^N$ is an error (noise) vector, $i = 1, \cdots, M$. Each column of $\mathbf{G}$ corresponds to an explanatory variable related to the specific experimental conditions under which the data were collected, β_i represents the weights of the explanatory variables (columns) of $\mathbf{G}$.

Considering all the voxels, the matrix form of (4) becomes

$$\mathbf{X} = \mathbf{G}\beta + \mathbf{E}, \tag{5}$$

where $\mathbf{X} \in R^{N\times M}$ is the data matrix, which is the same as $\mathbf{A}$ in (1), $\beta \in R^{K\times M}$, $\mathbf{E} \in R^{N\times M}$.

Multiplying both sides of (5) by the Moor-Penrose inverse β^{+} of β, we have

$$\mathbf{G} = \mathbf{X}\beta^{+} - \mathbf{E}\beta^{+}. \tag{6}$$

Furthermore, considering each column $\mathbf{g}_j$ of $\mathbf{G}$ and letting the noise vector be included implicitly in the coefficient vector, (6) can be rewritten as

$$\mathbf{g}_j = \mathbf{X}(\beta_j^{+} + \bar{\mathbf{e}}_j), \tag{7}$$

where $\bar{\mathbf{e}}_j = -\mathbf{X}^{+}\mathbf{E}\beta_j^{+}$.

Since $\mathbf{g}_j$ representing a specific experimental condition is the convolution of a stimulus/task function and a hemodynamic response function (HRF), it is the same as $\mathbf{y}$ in (1). Furthermore, in view of $\mathbf{X}$ in (7) and $\mathbf{A}$ in (1) representing the same data matrix, (7) is equivalent to the model in (1). The above analysis shows the connection of the model in (1) and the GLM model in (4). Now we point out the main differences between the two models: (i) In model (1), a transformed stimulus/task function is linearly represented by the time series of a set of voxels. The assumption of sparse representation implies that the number of voxels used in this representation is small. Note that although only small number of voxels are needed in sparse representation, they are generally representative voxels distributed in different activated brain areas. Conversely, in (7), the time series of each voxel is linearly represented by the columns of a design matrix, of which each column is a transformed stimulus/task function, or a function related to noise etc. (ii) A lot of voxels are considered simultaneously in model (1), i.e. the connection of different voxels instead of the connection of different stimulus/task functions is emphasized in (1). Conversely, all the stimulus/task functions are considered simultaneously in the GLM model (4). Thus it is the connection of different stimulus/task functions other than the connection of different voxels that is emphasized in (4). The connection of different voxels is generally considered ((e.g. by random field method)) in later analysis of SPM.

In this paper, we design a sparse representation algorithm based on the linear programming problem (2) for voxel selection in fMRI data analysis. The task of sparse representation is to find a coefficient vector $\mathbf{w}$ of model (1) such that $\mathbf{w}$ is as sparse as possible. The motivations that we use sparse representation here are: (i) Considering a huge number of voxels of the brain, only a small number of voxels are useful for representing a stimulus/task function $\mathbf{y}$ in (1). This is reflected by the sparsity of $\mathbf{w}$; (ii) Through sparse representation, we obtain a combination of voxels. This combination of voxels can represent the stimulus/task function $\mathbf{y}$ with high efficiency since it contains a small number of voxels ($\mathbf{w}$ is sparse). Thus the connections between those voxels in the combination are emphasized through a effective/sparse representation to $\mathbf{y}$; (iii) The combination mainly contains two classes of voxels. The fMRI time series of the first class of voxels are significantly correlated to $\mathbf{y}$, while the fMRI time series of the second class of voxels are not significantly correlated to $\mathbf{y}$ but important for representing $\mathbf{y}$. The first class of voxels can be identified using statistic parametric methods e.g. GLM-SPM, correlation method, however the second class of voxels are difficult to be identified using statistic parametric methods (see Fig. 7). (iv) From the latter discussion, the effectiveness of the combination of voxels selected by sparse representation can be evaluated using decoding approach. Furthermore, considering the similarity between the model (1) and decoding model [6], the voxel selection based on model (1) could be more suitable for decoding tasks than GLM model, as shown in this paper.

To demonstrate the effectiveness of our method, it was applied to the fMRI data set of PBAIC 2007. The data set of PBAIC 2007 was collected for a prediction task. The stimuli in the experiments performed to obtain these data sets are rich and non-repetitive. Therefore, it is difficult to perform voxel selection satisfactorily using typical methods such as Pearson correlation based methods. After voxel selection with our method, we perform the prediction of experience based cognitive tasks from the fMRI data set of PBAIC 2007 as in [8], [9]. The prediction results will be used in evaluation of our method. In our data analysis, we also compare our method with the baseline GLM-SPM method.

The remaining part of this paper is organized as follows. Our detection algorithm is presented in Section II. The analysis of convergence and effectiveness of this algorithm are also included. In Section III, we use our algorithm for voxel selection in fMRI data analysis. Additional discussions related to fMRI data analysis are included in Appendices 1 and 2. Finally, conclusions are presented in Section IV to review our method.

II. Materials and methods

A. Algorithm

In this section, we present our algorithm for voxel selection. We do not directly use (2) considering the following three aspects: (i) The number of nonzero entries of $\mathbf{w}$ generally equals to N [18]. This means that sparsity of $\mathbf{w}$ decreases with increasing N. This leads to a situation where increase in the amount of training data may not lead to any improvement in feature selection by (2). (ii) (2) is not suitable for the overdetermined case in which $N > M$; (iii) When N is not sufficiently large, $\mathbf{w}$ obtained by single optimization may not reflect the important features well. Even if N is sufficiently large, this problem still exists because of noise. In order to address these three problems, we will extend (2) and present an iterative detection algorithm in this paper.

Suppose that each row of $\mathbf{A}$ in (2) represents a data sample, $\mathbf{y}$ can be speech signal, stimulus, labels etc. In this paper, $\mathbf{A}$ is an fMRI data matrix of which each column is a time series of a voxel and each row contains the data of one volume (or part of one volume), $\mathbf{y}$ is the convolution of a stimulus/task function and a hemodynamical response function. The following algorithm is designed to detect the parts in the rows of $\mathbf{A}$ (e.g. pixels or voxels) relevant to $\mathbf{y}$.

Algorithm 1:

Step 1: For $k = 1, \cdots$, do the following Steps 1.1 to 1.4.

Step 1.1: Randomly choose L rows from $\{\mathbf{a}_1, \cdots, \mathbf{a}_N\}$ to construct a L by M matrix denoted as $\mathbf{A}_k$, the corresponding L entries of $\mathbf{y}$ form a column vector denoted as $\mathbf{y}_k \in R^L$.

Step 1.2: Solve the following optimization problem. Similar to (2), this optimization problem can be converted to a standard linear programming problem.

$$\min ||\mathbf{w}||_1, \ s.t., \ \mathbf{A}_k\mathbf{w} = \mathbf{y}_k. \tag{8}$$

The optimal solution of (8) is denoted by $\bar{\mathbf{w}}^{(k)}$.

Step 1.3: Let

$$\mathbf{w}^{(k)} = \frac{1}{k}\sum_{i=1}^{k} \bar{\mathbf{w}}^{(i)}. \tag{9}$$

Step 1.4: If $d(k) = ||\mathbf{w}^{(k)} - \mathbf{w}^{(k-1)}||_2 < \alpha$ or $k < K_0$, where α is a predefined small positive constant and K_0 is a predefined limiting upper bound for the number of iterations (e.g. $K_0 = 400$), set $\mathbf{w} = \mathbf{w}^{(k)}$ and go to Step 2. Otherwise go to Step 1.1.

Step 2: For a given positive θ, define $R = \{j| \, |w_j| > \theta, j = 1, \cdots, M\}$. Then R is our detected part of interest in all rows of $\mathbf{A}$.

Note that noise is not explicitly reflected in (8). However, the weight vector $\mathbf{w}$ generally is affected by noise (please see (7)). Through the average operation in (25), the effect of noise can be reduced (see Appendix 3: Robustness analysis). Furthermore, through randomly taking L rows from $\mathbf{A}$, $\bar{\mathbf{w}}^{(k)}$ obtained in (8) can be seen as a random sample of $\mathbf{w}$, while $\mathbf{w}^{(k)}$ in (25) can be seen as an approximation to the mean of $\mathbf{w}$. Suppose that Algorithm 1 terminates after K iterations. This implies that we obtain K random samples of $\mathbf{w}$ and then calculate their mean. The number of iterations (i.e. the number of random samples) can be easily determined because of the convergence of Algorithm 1 as shown later.

In the following, we discuss the setting of three parameters L, α and θ in Algorithm 1. Note that each $\bar{w}^{(k)}$ is obtained through solving a standard linear programming problem, thus Algorithm 1 is not involved in the setting of initial vector $\mathbf{w}^{(0)}$.

When the parameter L is set, two aspects need to be considered: (1) L is not very small such that the columns of $\mathbf{A}_k$ and $\mathbf{y}_k$ in (8) contain temporal evolution information; (2) L is not very large such that the computational burden for solving the optimization problem (8) is not heavy. It will be explained in Appendix 1 that the data analysis results are not sensitive to the value of L provided that L is not very small. In this paper, we generally set $L = 0.1N$. The other choices e.g. $L = 0.05N$, $0.15N$ and $0.2N$ are also acceptable.

As will be seen, Algorithm 1 is convergent, we can easily set a small α (e.g. $\alpha < 0.01$) to obtain a stable $\mathbf{w}$.

The parameter θ can be chosen in different ways depending on the applications. One way is cross-validation method, which can be seen in Appendix 2. Here we present a probability method. Considering the entries of $\mathbf{w}$ are sparse, we assume that the probability distribution of the entries of $\mathbf{w}$ is Laplacian. Using all entries of $\mathbf{w}$ as samples, we estimate the mean, the variance and the inverse cumulative distribution function F^{-1} of this Lapcian distribution. We then define $R = \{i| \, |w_i| > \theta, i = 1, \cdots, M\}$, where θ is chosen as $F^{-1}(p_0)$, p_0 is a given probability (e.g. 0.975 in this paper). As will be shown in section III, this method for determining θ is acceptable. The values of the parameter θ determined by the cross-validation method and the probability method are generally different. If sufficient training data are available and the effect of decoding is emphasized, we can use the cross-validation method to determine θ. If we emphasize the effect of localization, then we can use the probability method.

We now analyze the convergence and effectiveness of Algorithm 1.

Convergence: Suppose that in the kth iteration of Algorithm 1, we have output

$$\mathbf{w}^{(k)} = \frac{1}{k}\sum_{i=1}^{k}\bar{\mathbf{w}}^{(i)}. \tag{10}$$

Let

$$d(k+1) = ||\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)}||_2. \tag{11}$$

From (10),

$$\begin{aligned}\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)} &= \frac{1}{k+1}\sum_{i=1}^{k+1}\bar{\mathbf{w}}^{(i)} - \frac{1}{k}\sum_{i=1}^{k}\bar{\mathbf{w}}^{(i)} \\ &= \frac{k\sum\limits_{i=1}^{k+1}\bar{\mathbf{w}}^{(i)} - (k+1)\sum\limits_{i=1}^{k}\bar{\mathbf{w}}^{(i)}}{k(k+1)} \\ &= \frac{k\bar{\mathbf{w}}^{(k+1)} - \sum\limits_{i=1}^{k}\bar{\mathbf{w}}^{(i)}}{k(k+1)}.\end{aligned} \tag{12}$$

In Algorithm 1, for given data matrix $\mathbf{A}$ and parameter L, there are totally $C_N^L(=\frac{N(N-1)\cdots(N-L+1)}{L!})$ choices of the pairs ($\mathbf{A}_k$, $\mathbf{y}_k$) in (8). For each pair ($\mathbf{A}_k$, $\mathbf{y}_k$), there is a weight vector $\bar{\mathbf{w}}^{(k)}$ which is the solution of (8). Let $\gamma_0 = \max\{||\bar{\mathbf{w}}^{(k)}||_2, k = 1, \cdots, C_N^L\}$, then $||\bar{\mathbf{w}}^{(k)}||_2 < \gamma_0$. From (12), we have

$$||\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)}||_2 \le \frac{(2k)\gamma_0}{k(k+1)} = \frac{2\gamma_0}{k+1}. \tag{13}$$

Therefore, $\lim\limits_{k\to\infty} d(k) = 0$, i.e. Algorithm 1 is convergent.

In fact, the convergence of Algorithm 1 originates from the facts: (1) there are finite number of weight vectors $\bar{\mathbf{w}}^{(k)}$ of which each corresponds to a pair ($\mathbf{A}_k$, $\mathbf{y}_k$); (2) then they are bounded. Although the number of $\bar{\mathbf{w}}^{(k)}$ is huge generally (C_N^L), Algorithm 1 converges in several hundred of iterations (see Fig. 1 in our data analysis section).

Although $\mathbf{w}^{(k)}$ in (10) is generally not so sparse as $\bar{\mathbf{w}}^{(i)}$, our simulations and data analysis results show that a large fraction of entries of $\mathbf{w}^{(k)}$ are close to zero. Thus we say that $\mathbf{w}^{(k)}$ is still sparse.

Effectiveness: Regarding the effectiveness of Algorithm 1, we have the following explanation. Here, we only consider the underdetermined case where $N < M$. First we define a set of M dimensional vectors U such that $\forall \mathbf{w}(\tilde{\mathbf{A}}, \tilde{\mathbf{y}}) \in U$, $\mathbf{w}(\tilde{\mathbf{A}}, \tilde{\mathbf{y}})$ is the 1-norm solution of the equations

$\tilde{\mathbf{A}}\mathbf{w} = \tilde{\mathbf{y}}$, where $\tilde{\mathbf{A}}$ is composed by L rows randomly taken from $\mathbf{A}$, $\tilde{\mathbf{y}}$ is a vector composed by L corresponding entries of $\mathbf{y}$. Note that there are C_N^L vectors in U. Next, we define a M dimensional random vector $\mathbf{v} = [v_1, \cdots, v_M]^T$, where $\mathbf{v}$ randomly takes values in U.

For a sample $\mathbf{w}(\tilde{\mathbf{A}}, \tilde{\mathbf{y}})$ of $\mathbf{v}$ in U, it is sparse since it has at most L nonzeros. Generally, the magnitude of the ith entry of $\mathbf{w}(\tilde{\mathbf{A}}, \tilde{\mathbf{y}})$ reflects the significance of the ith column of $\tilde{\mathbf{A}}$ for the constraint $\tilde{\mathbf{A}}\mathbf{w} = \tilde{\mathbf{y}}$. Obviously, the output of Algorithm 1 satisfies $\mathbf{w} = \frac{1}{k}\sum\limits_{i=1}^{k} \bar{\mathbf{w}}^{(i)} \approx E(\mathbf{v})$, i.e. the expected value of $\mathbf{v}$. Now we show that $E(v_i)$ can reflect the significance of the ith column of $\mathbf{A}$ for the regression between $\mathbf{A}$ and $\mathbf{y}$.

For $\bar{\mathbf{w}}^{(i)}$ obtained in the ith iteration Algorithm 1, we have

$$\mathbf{A}\bar{\mathbf{w}}^{(i)} = \mathbf{y} + \mathbf{n}^{(i)}, \tag{14}$$

where $\mathbf{n}^{(i)} = [n_1^{(i)}, \cdots, n_N^{(i)}]^T$, $n_j^{(i)} = 0$ if $j \in Ind_i$ (Ind_i is the set of indices of the L rows of $\mathbf{A}_i$ in $\mathbf{A}$), otherwise $n_j^{(i)} = y_j - \mathbf{a}_j \cdot \bar{\mathbf{w}}^{(i)}$.

Furthermore, we have

$$\mathbf{A}\mathbf{w} = \mathbf{A}(\frac{1}{k}\sum_{i=1}^{k} \bar{\mathbf{w}}^{(i)}) = \mathbf{y} + \frac{1}{k}\sum_{i=1}^{k} \mathbf{n}^{(i)}. \tag{15}$$

Note that $n_j^{(i)}$ can be positive, negative or zero. In many cases especially when the parameter L in Algorithm 1 is not small, the expectation of $n_j^{(i)}$ can be assumed to be close to zero. That is, $\frac{1}{k}\sum\limits_{i=1}^{k} \mathbf{n}^{(i)} \approx \mathbf{0}$ for sufficiently large k. Thus

$$\mathbf{A}\mathbf{w} \approx \mathbf{A}E(\mathbf{v}) \approx \mathbf{y}. \tag{16}$$

Therefore, the correlation between $\mathbf{A}\mathbf{w}$ and $\mathbf{y}$ is close to 1 and $\mathbf{w}$ is close to a regression coefficient vector between training data matrix $\mathbf{A}$ and $\mathbf{y}$.

From the above analysis and the definition of $\mathbf{w}$ in Algorithm 1, we can see: (i) The magnitude of $\mathbf{w}$ (i.e. $E(v_i)$) reflects the significance of the ith column of $\mathbf{A}$ to the satisfaction of (16). (ii) If L in Algorithm 1 is fixed, we can obtain a consistent $\mathbf{w}$. This is due to the convergence of Algorithm 1. (iii) More importantly, $\mathbf{w}$ is still sparse. This will be demonstrated in our data analysis examples. Based on the sparsity of $\mathbf{w}$, the voxels which are the most correlated to the stimulus/task function can be selected.

B. Voxel selection in functional MRI data

In this section, we apply Algorithm 1 to the fMRI data of PBAIC 2007 [34] for voxel selection. The fMRI data was collected by Siemens 3T Allegra scanner with imaging parameters TR and TE being 1.75s and 25ms respectively. Three subjects' data were available in the competition. Each subject's data consists of three *runs*. Each run consists of 500 volumes of fMRI data of which each volume contain $64 \times 64 \times 34$ voxels. The size of a voxel is $3.2 \times 3.2 \times 3.5$ mm^3. The preprocessed data provided by the competition are used in this paper. The data preprocessing attempted to remove some standard artifacts that occur in fMRI data that may hinder data analysis. The functional and structural data were preprocessed with AFNI and NIS in the following steps: slice time correction, motion correction and detrending. The feature data was preprocessed by convolving the raw feature vectors with the double gamma hemodynamic response filter (HRF) produced by the SPM (see http://www.fil.ion.ucl.ac.uk/spm/). Through a mask preprocessing, the total number of voxels in the brain is ≈ 32000. Thus the fMRI data for each run is represented by a matrix consisting of around 32000 columns (voxels) and 500 rows (time points). Each column of the matrix is the time series of a voxel. When the scans were obtained, the subject was performing several tasks (e.g. listen to instructions, pick up fruits) in a virtual reality (VR) world. The *ratings* for these tasks were computed by considering the delay of hemodynamic responses and form the task functions. 13 tasks were considered in the competition. Only the tasks for the first two runs were distributed at www.braincompetition.org. Therefore, here we use only data from the first two runs for analysis. We present detailed results mainly for four tasks: (i) The *Hits* task, times when subject correctly picked up fruit or weapon or took picture of a pierced person; (ii) the *Instructions* task, which represents the task of listening to instructions from a cell phone in the virtual world; (iii) the *Faces* task, times when subject looked at faces of a pierced or unpierced person; (iv) the *Velocity* task, times when subject was moving but not interacting with an object. For more detailed description of the data, refer to [34]. The goal of the competition was to predict the task functions of the third run using the fMRI data. Our final submission based on Algorithm 1 to this competition was ranked 10th based on the average score of the features. As pointed out in [31], a fair comparison with other methods cannot be made, as post-processing had decisive effect on performance.

Since the number of voxels is huge, voxel selection plays an important role in fMRI data analysis. Using the Instructions task as an example, we now describe our data analysis method. The preprocessed fMRI data obtained from the competition website (www.braincompetition.org) is first filtered temporally and spatially. The temporal filter is $\frac{1}{4}[1, 2, 1]$, while the spatial filter is a cube with $[1, 2, 1; 2, 4, 2; 1, 2, 1]$, $[2, 4, 2; 4, 8, 4; 1, 2, 1]$, $[1, 2, 1; 2, 4, 2; 1, 2, 1]$. We then perform 2-fold cross-validation as follows. In the first fold, we use Run 1 data to calculate the Pearson correlation between the time series of each voxel and the transformed Instructions task function. The voxels with high absolute value of this correlation are chosen to form a set of voxels, $\mathcal{N}$. Then, our algorithm is used for a second selection of voxels to obtain $R \subset \mathcal{N}$. In Algorithm 1, $\mathbf{A} \in R^{500\times|\mathcal{N}|}$, of which each column is a time series of a voxel in $\mathcal{N}$, $\mathbf{y} \in R^{500}$ is a transformed task function. The parameters in Algorithm 1 are set as follows. The number of iterations is fixed to 600 to see the details of algorithm convergence, L is set to 25 and θ can be chosen as described in section II-A (or using a cross-validation method presented in Appendix 2). Ridge regression is used on the time series of voxels $\in R$ to predict the transformed Instructions task function of Run 2. Prediction accuracy is measured as the Pearson correlation between the actual transformed task and the predicted task. In the second fold, we use Run 2 data for training and predict the transformed Instructions task function for Run 1.

For the purpose of comparison, we use GLM-SPM method to replace our method for selection of voxels and perform the 2-fold cross-validation as described above. Note that when GLM-SPM method is used for voxel selection, all 13 transformed/convolved task functions provided by PBAIC 2007 are used to construct the design matrix. For each voxel and a task, a t-statistics is calculated as in [4]. For a task, those voxels with high absolute values of t-statistics are selected.

III. Results and discussions

In this section, we present our data analysis results to illustrate the convergence of Algorithm 1 and the sparsity of the weight vector $\mathbf{w}$. By comparing our method with GLM-SPM method, we demonstrate the validity of our algorithm for voxel selection.

Convergence of Algorithm 1. As an example, we show our convergence analysis result for Instructions task in run 1 of Subject 1. In the initial selection of voxels, we set $|\mathcal{N}|$ to 500. Three cases are considered, in which the parameter L (the number of constraints in (8)) of Algorithm 1 is set to be 25, 50 and 75 respectively. The three subplots in Fig. 1 show three iterative curves of the convergence index $d(k) = ||\mathbf{w}^{(k)} - \mathbf{w}^{(k-1)}||_2$ of Algorithm 1, which correspond to the three cases respectively. From Fig. 1, we can see that Algorithm 1 converges after ≈ 300 iterations in the three cases. However, the execution time in our PC computer (2.3GHz CPU, 3G RAM) for the three cases are 147s, 426s and 768s respectively. Thus when L increases, the computational burden increases rapidly.

Sparsity of weights. Fig. 2 shows two weight vectors $\mathbf{w}$ obtained by Algorithm 1 (top) and correlation coefficients (bottom) for the voxels in $\mathcal{N}$ obtained in the two-fold cross-validation for subject 1 and Instructions task. From this figure, we can see the sparsity in weights when compared to correlation coefficients. Therefore, we can say that the weight vector is more suitable for localization of voxels than Pearson correlation.

Effectiveness. First, we check the correlation between $A\mathbf{w}$ and $\mathbf{y}$, where $\mathbf{A}$ and $\mathbf{y}$ are fMRI data matrix and a transformed task vector respectively. As mentioned in Section II-A, the weight vector $\mathbf{w}$ can be seen as regression coefficients between A and $\mathbf{y}$. This means that the correlation between $A\mathbf{w}$ and $\mathbf{y}$ is big. Fig. 3 shows the iterative curves of this correlation for the Instructions task of three subjects for fold 1 (run 1 used for training). As the number of iterations increases, the three curves of correlation increase and tend to three limits which are larger than 0.9. This demonstrates our analysis. When sparse representation approach is used for voxel selection, a voxel whose fMRI time series is highly correlated to $\mathbf{y}$ can generally be selected. However, if there are a set of voxels e. g. belonging to the same brain area of which the fMRI time series are quite correlated to each other, then only small part of representative voxels are selected. The iterative Algorithm 1 of which each iteration uses different time points may alleviate the loss of these voxels which are highly correlated to $\mathbf{y}$.

Next, we analyze the prediction accuracy. We compare Algorithm 1 with GLM-SPM method for voxel selection. First, we compare the ability of each method in choosing the *most relevant* voxel. In the table I, we present the prediction accuracies (averaged over two folds) for $N_R = 1$ for three subjects, four transformed tasks. Hereafter, N_R denotes the number of voxels of R, the set of selected voxels. From table I, we can see that the voxels selected by Algorithm 1 is more correlated to the transformed task functions in most of cases than those selected by GLM-SPM method.

Furthermore, we test if Algorithm 1 is consistently better than GLM-SPM method for voxel selection. Let $\mathbf{b} = [1, 2, 4, \cdots, 300]$. For each i ($i = 1, \cdots, 151$), we set $N_R = b_i$, and predict the four transformed task functions for all subjects and average the results over two folds of cross validation. Fig. 4 shows the plots of average prediction accuracy with respect to $\mathbf{b}$ for the four methods and three subjects and four tasks. In several cases e.g. shown in the subplot in the first row and the second column of Fig. 4, the performance of Algorithm 1 is comparative to that of GLM-SPM method; while in the other cases e.g. shown in the subplot in the second row and the third column, the performance of Algorithm 1 is significantly better than that of GLM-SPM method.

We also analyze the effectiveness of choosing R using θ as described in Section II-A. Example, for instruction task, the number of voxels in R obtained using Algorithm 1, N_{th} are: *Subject 1,* 12 (fold 1), 29 (fold 2); *Subject 2,* 24 (fold 1), 21 (fold 2); *Subject 3,* 28 (fold 1), 20 (fold 2). The corresponding prediction accuracy (averaged over two folds) for the three subjects are 0.8151, 0.7469 and 0.8591 respectively. The prediction accuracy (averaged over two folds) for the four tasks are marked with a '*' in Fig. 4. Even though N_{th} does not correspond to the best prediction accuracy, it can lead to a satisfactory result. From this analysis, we conclude that the method described in Section II-A for selecting θ is acceptable (Except for task 3 of subject 1. The correlation between fMRI data and this task is always low).

Until now, we have presented our analysis results for 4 tasks. Considering 13 tasks are available for the 3 subjects in the data set, we have 39 cases (one case corresponds to one task and one subject). We analyzed each case as described above for the four tasks. After obtaining the prediction accuracy averaged over two folds of cross validation by Algorithm 1, we count the number of times, r_i that Algorithm 1 shows the best prediction accuracy among the four methods for each $N_R = b_i$, $\mathbf{b} = [1, 2, 4, \cdots, 300]$. Performing the similar counting for GLM-SPM method, we obtain $r_i^{(spm)}$. Next, we calculate two ratios (percentages) for the two methods: $\frac{r_i}{39} \cdot 100\%$, $\frac{r_i^{(spm)}}{39} \cdot 100\%$. Fig. 5 shows the two ratio curves, from which we can see Algorithm 1 has higher ratios.

Localization. Now we analyze the effectiveness of Algorithm 1 in selecting voxels from a biological perspective. Each of the four tasks evaluated here (hits, instructions, faces, velocity), can be related to activity in specific region(s) of the brain. For example, the hits and velocity events are expected to be correlated with activity in the motor cortex, especially the part for planning actions, which is the supplementary motor cortex. Instructions task is expected to be correlated with activity in the auditory cortex. Face events are expected to be correlated with activity in the fusiform face area (FFA), which is special part for face processing, located on the ventral surface of the temporal lobe.

Now we choose two representative cases to show our results of localization. The first case is for task 2 (Instructions), run 1 and Subject 3, while the second case is for task 3 (Faces), run 1 and Subject 2. In this two cases, Algorithm 1 has better performance of prediction than GLM-SPM method. For the other cases in which our algorithm has better performance of prediction, we have similar conclusion presented in the following.

For Instructions task, Fig. 6 show 100 voxels, of which 50 voxels are selected by Algorithm 1, while the other 50 voxels are selected by GLM-SPM method. The brain slices are in neurological (L=L) space. We can see that most of these voxels selected by the two methods are in the *appropriate* areas of the brain. For instance, Voxels from the auditory cortex are shown in Slices 13-18 etc. It follows from this figure that there are several voxels which can be selected by both Algorithm 1 and GLM-SPM method. Furthermore, many voxels selected by the two methods are close in locations although they are not overlapped. However, the voxels selected by GLM-SPM method are mainly located in Slices 14 and 15, which form two clusters. The voxels selected by Algorithm 1, which do not form clusters, are distributed in more slices than those selected by GLM- SPM method.

There exist several voxels which can be selected by Algorithm 1 other than GLM-SPM method and useful for prediction/decoding. The two subplots in the first row of Fig. 7 show two voxels, in which the first one with the highest t value in SPM detection is selected by both Algorithm 1 and GLM-SPM method, the second one is selected only by Algorithm 1 other than GLM-SPM method. The second row show the corresponding fMRI signals of the two voxels. obviously, the fMRI signals of the two voxels contain useful information related to the task.

For Faces task, Fig. 8 shows the distribution of the 100 voxels, of which 50 voxels are selected by Algorithm 1, while the other 50 voxels are selected by GLM-SPM method. Similarly as in Fig. 6, we can see that several voxels (highlighted in yellow) selected by both Algorithm 1 and GLM-SPM method are common. Furthermore, most of these voxels selected by the two methods are in the *appropriate* areas of the brain. For instance, some selected voxels are in the occipital cortex (e.g. Slices 10-15). Furthermore, only these voxels selected by GLM-SPM method form several clusters (slices 11, 14 and 17).

Algorithm 1 based on sparse representation selects a combination of voxels which are generally distributed in wider brain areas than those selected by GLM-SPM method. Since this combination of voxels is used to represent the corresponding task function, Algorithm 1 is more suitable for prediction/decoding of tasks in many cases than GLM-SPM method. Conversely, since it is easily for GLM-SPM method to show cluster/group effect, this method is suitable for localization of active brain areas.

Parameters setting. In the following, we present our parameter settings of Algorithm 1 in this data analysis section. (i) From our analysis, the number of initially selected voxels does not make any significant difference to the results. This parameter is set to 500 voxels in this paper. We have tested our methods with 1000 and 2000 voxels for initial selection and confirmed the insensitivity of the results to this parameter. (ii) The number of iterations corresponding to the parameter α in Algorithm 1 is fixed to 600. From the fact that Algorithm 1 is convergent (section II-A) and that the changes is weights after 300 iterations is not significant (see figure 3), the results will be consistent as long as the number of iterations is sufficiently large. (iii) The number of constraints L in (8) is set to 50 except that it is especially pointed out. The results of Algorithm 1 are not very sensitive to L provided that L is not too small. In Appendix 1, we analyze the sensitivity of the results to L and provide support for the above statement. Another approach towards selecting L would be to choose its value randomly in each iteration. Although this is valid, we concluded from our analysis that the results are not significantly better. Therefore the value of L is chosen to be small, but big enough for Algorithm 1 to be valid so that the computational burden is minimized. (iv) The parameter θ in Algorithm 1 (corresponding to N_R) determines the number of voxels selected by Algorithm 1. If the objective is to localize important voxels, it can be set as in Section II-A. If the objective is to predict tasks as above, this number can be chosen from a wide range (see Fig. 4). Typically, it can be set to a number around 100. Another method for setting θ is cross-validation, which is described in Appendix 2.

IV. Conclusions

In this paper, we presented an iterative detection algorithm based on sparse representation. Then, we analyzed its convergence and effectiveness. This algorithm may be used for feature selection, localization, novelty detection, etc as 1-norm SVM. Here, we presented one application for voxel (feature) selection in fMRI data analysis. The results demonstrate that this method can be used for voxel selection in the cases of both repeated stimulus/tasks (e.g. Instructions task) and non-repeated stimulus/tasks (e.g. Faces task). The sparse representation model used in our algorithm can be seen as the opposite of the GLM Model widely used for fMRI analysis, however there exists significant difference between the two models. In the sparse representation model, a lot of voxels are considered simultaneously, but the task/stimulus conditions are considered separately. Using our algorithm, a combination of voxels are selected. This combination of voxels plays an important role in effective/sparse representation of a task/stimulus function. Conversely, voxels are considered separately, but the task/stimulus conditions are considered simultaneously in GLM model. The validity of our method was shown through the comparison with GLM-SPM method in our data analysis.

In addition to voxel selection as shown in this paper, the sparse representation approach also can be used for decoding a task based on fMRI activity. A simple implementation strategy can be: First, a sparse regression model is trained using model (2), where $\mathbf{A}$ is a known fMRI data matrix constructed from selected voxels and $\mathbf{y}$ is a known task function. Second, an unknown task function can be decoded using model (1), where $\mathbf{A}$ is a new fMRI data matrix of selected voxels, $\mathbf{w}$ is a sparse coefficient vector obtained in the training phase. Our initially analysis results, which are not included in this paper, has shown the effectiveness of this decoding method in several cases. However, further study is needed in our future work for improving the performance of this decoding method since the constraint equation in model (2) is underdetermined and noise is neglected.

Acknowledgement: The authors are grateful to the organizers of Pittsburgh Brain Activity Interpretation Competition 2007 for providing the fMRI data. The authors are grateful to Dr. Pei Sun (Brain Science Institute, RIKEN, Japan) for his contribution in the fMRI data analysis part of this paper. The authors are also grateful to the anonymous reviewers for their insightful and constructive suggestions.

REFERENCES

[1] Y. Kamitani, F. Tong, "Decoding the visual and subjective contents of the human brain," *Nature Neuroscience* 8(5), 679-685, 2005.

[2] K. J. Friston, P. Jezzard, R. Turner, "Analysis of Functional MRI time series," *Human Brain Mapping*, vol. 1, pp. 153-171, 1994.

[3] K. J. Friston, A. P. Holmes, J. B. Poline, et al, "Analysis of fMRI time-series revisited," *NeuroImage*, vol. 2, pp. 45-53, 1995.

[4] K. J. Friston, A. P. Holmes, K. Worsley, J. B. Poline, et al, "Statical parameter maps in functional imaging: a general linear approach," *Human Brain Mapping*, vol. 2, pp. 189-210, 1995.

[5] D. D. Cox, R. L. Savoy, “Functional magnetic resonance imaging (fMRI) brain reading: detecting and classifying distributed patterns of fMRI activity in human visual cortex,” *Neuroimage* 19, 261-270, 2003.

[6] T. M. Mitchell, R. Hutchinson, R. S. Niculescu, F. Pereira, X. Wang, M. Just, S. Newman, “Learning to decode cognitive states from brain images,” *Machine Learning* 57, 145-175, 2004.

[7] Haxby, J. V., Gobbini, M. I., Furey, M. L., Ishai, A., Schouten, J. L., Pietrini, P., “Distributed and overlapping representations of faces and objects in ventral temporal cortex,” *Science* 293, 2425-2430.

[8] A. Battle, G. Chechik, D. Koller, “Temporal and cross-subject probabilistic models for fMRI prediction task,” *Advances in Neural Information Processing Systems 19*, pp. 146-153. Cambridge, MA: MIT Press, 2006.

[9] F. Meyer, G. J. Stephens, “Locality and low-dimensions in the prediction of natural experience from fMRI,” *Proceedings of the Twenty First Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, 2007.

[10] S. Chen, D. L. Donoho & M. A. Saunders.“Atomic decomposition by basis pursuit,” *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33-61, 1998.

[11] B. A. Olshausen, P. Sallee, & M. S. Lewicki, “Learning sparse image codes using a wavelet pyramid architecture,” *Advances in Neural Information Processing Systems 13*, pp. 887-893. Cambridge, MA: MIT Press, 2001.

[12] B. A. Olshausen, & D. J. Field, “Sparse coding with an overcomplete basis set: a strategy employed by V1?” *Vision Research*, Vol. 37, pp. 3311-3325, 1997.

[13] M. S. Lewicki, & T. J. Sejnowski, “Learning overcomplete representations,” *Neural Computation* Vol. 12(2), pp. 337-365, 2000.

[14] R. Gribonval, M. Nielsen, “Sparse decompositions in unions of bases,” *IEEE Trans. Inf. Th.*, Vol. 49, No. 12, pp 3320-3325, 2003.

[15] J. A. Tropp, A. C. Gilbert, S. Muthukrishnan and M. J. Strauss, “Improved sparse approximation over quasi-incoherent dictionaries” *Proceedings of the 2003 IEEE International Conference on Image Processing*, Barcelona, September 2003.

[16] D. L. Donoho, & M. Elad, “Maximal sparsity representation via l_1 minimization,” *the Proc. Nat. Aca. Sci.* vol. 100, pp. 2197-2202, 2003.

[17] M. Girolami, “A variational method for learning sparse and overcomplete representations,” *Neural Computation*, Vol. 13(11), pp. 2517-1532, 2001.

[18] Y. Q. Li, A. Cichocki and S. Amari, “Analysis of Sparse representation and blind source separation,” *Neural Computation*, vol. 16, pp. 1193-1234, 2004.

[19] Y. Li, S. I. Amari, A. Cichocki, C. Guan, “ Probability estimation for recoverability analysis of blind source separation based on sparse representation,” *IEEE Trans. On Information Theory*, vol. 52, no. 7, 2006.

[20] Y. Li, S. I. Amari, A. Cichocki, D. W. C. Ho and S. Xie, “Underdetermined blind source separation based on sparse representation,” *IEEE Trans. on Signal Processing*, vol. 54, no.2, pp. 423-437, 2006.

[21] E. Kidron, Y.Y. Schechner, M. Elad, “Cross-modal localization via sparsity,” *IEEE Transactions on Signal Processing*, Vol. 55, no. 4, pp. 1390 - 1404, 2007.

[22] M. Zibulevsky, & B. A. Pearlmutter, “Blind Source Separation by Sparse Decomposition,” *Neural Computations*, vol. 13(4), pp.863-882, 2001.

[23] J. Bi, P. Bennett, M. Embrechts, C. M. Breneman, M. Song, “Dimensionality reduction via sparse support vector machines,” *Journal of Machine Learning Research*, vol. 3, pp. 1229-1243, 2003.

[24] J. Bi, Y. Chen and J. Wang, “A sparse support vector machine approach to region-based image categorization,” Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR’05), 2005.

[25] J. Zhu, S. Rosset, T. Hastie, R. Tibshirani, “1-norm support vector machines,” *Neural Information Processing Systems*, 2003.

[26] A. J. Smola, B. Scholkopf, and G. Gatsch, Linear Programs for Automatic Accuracy Control in Regression, *Proc. International Conference of Artificial Neural Networks*, Berlin, Springer, 1999.

[27] K. P. Bennett, Combining Support Vector and Mathematical Programming Methods for Classification, B. Scholkopf, C. Burges and A. Smola, editors, *Advances in Kernel Methods C Support Vector Machines*, pp. 307C326, 1999.

[28] C. Campbell, K. P. Bennett, “A linear programming a pproach to novelty detection,” *Neural Information Processing Systems*, vol. 13, pp. 395-401, 2000.

[29] R. Tibshirani, R., “Regression selection and shrinkage via the LASSO,” *Journal of the Royal Statistical Society, Series B (Methodological) 58(1), 267-288, 1996.*

[30] S. Hochreiter, K. Obermayer, “Support Vector Machines for Dyadic Data,” Neural Computation 18(6), 1472-1510, 2006.

[31] M. K. Carroll, G. A. Cecchi, I. Rish, R. Garg, A. R. Rao, “Prediction and interpretation of distributed neural activity with sparse models,” NeuroImage, vol. 44(1), 112-122, 2009.

[32] Y. Li, P. Namburi, C. Guan, J. Feng, “A sparse representation based algorithm for voxel selection in fMRI data analysis,” 14th Annual meeting of the Organization for Human Brain Mapping, 2008. 6, Melbourne, Australia.

[33] I. V. Tetko, A. E. P. Villa, “A comparative study of pattern detection algorithm and dynamical system approach using simulated spike trains,” Lecture Notes in Computer Science, vol. 1327, pp. 37-42, 1997.

[34] Schneider, W., Siegle, G. Pittsburgh Brain Activity Interpretation Competition 2007 Guide Book: Interpreting subject-driven actions and sensory experience in a rigorously characterized virtual world. http://www.braincompetition.org

Appendix 1: On setting the number of constraints in Algorithm 1

In this appendix, we first show our results obtained by Algorithm 1 with the numbers L of constraints in (8) set to be $L_1 = 25$, $L_2 = 50$ and $L_3 = 75$ respectively.

Let $\mathbf{b} = [1, 2, 4, \cdots, 300]$ as in Section III. For each pair of i ($i = 1, \cdots, 151$) and L ($L = 25, 50, 75$), we set $N_R = b_i$ and obtain average prediction accuracies by Algorithm 1 with parameter L (averaged over two folds of cross validation) for the 4 tasks, and 3 subjects. Each subplot of Fig. 9 shows three prediction accuracy curves (with respect to b_i) for $L = 25$, 50 and 75 respectively. Note that the prediction accuracy curves for $L = 50$ (black solid lines) are the same as those in Fig. 4. By comparing these curves in each subplot of Fig. 9, we can see that the prediction results are not sensitive to the parameter L.

Next, we compare the sets of voxels selected by Algorithm 1 with its parameter L set as $L_1 = 25$ and $L_2 = 50$ respectively. For the ith task ($i = 1, \cdots, 13$), the jth subject ($j = 1, 2, 3$), the kth run ($k = 1, 2$), we calculate weight vectors $\mathbf{w}_{L_1,i,j,k}$ and $\mathbf{w}_{L_2,i,j,k}$ using Algorithm 1. For a given number b_l ($b_l = 1, 2, 4, \cdots, 400$), we choose two sets of voxels $R_{L_1,b_l}(i, j, k)$ and $R_{L_2,b_l}(i, j, k)$ from $\mathcal{N}$ (the initially selected 500 voxels) using $\mathbf{w}_{L_1,i,j,k}$ and $\mathbf{w}_{L_2,i,j,k}$ respectively. Furthermore, we calculate the number $q(i, j, k, b_l)$ of voxels $\in R_{L_1,b_l}(i, j, k) \cap R_{L_2,b_l}(i, j, k)$ and the ratio $ratio(i, j, k, b_l) = \frac{q(i,j,k,b_l)}{b_l}$. Averaging $ratio(i, j, k, b_l)$ across i, j, k, we obtain the mean of $ratio(i, j, k, b_l)$ denoted as $r(b_l)$. Fig. 10 shows the curve of $r(b_l)$ with b_l (solid line).

For each b_l ($b_l = 1, 2, 4, \cdots, 400$), we also randomly take two subsets of $\mathcal{N}$. Denote the two subsets as $\bar{R}_{1,b_l}$ and $\bar{R}_{2,b_l}$, each of which contains b_l voxels. We also calculate the ratio $\bar{r}(b_l)$ with which $\bar{R}_{1,b_l}$ and $\bar{R}_{2,b_l}$ are overlapped. The curve of $\bar{r}(b_l)$ is shown as the dashed line in Fig. 10.

From Fig. 10, we can see that if $b_l >\approx 75$, $r(b_l) > 70\%$. Furthermore, the ratio $r(b_l) \gg \bar{r}(b_l)$. Therefore, the two voxel sets $R_{L_1,b_l}(i, j, k)$ and $R_{L_2,b_l}(i, j, k)$ determined by Algorithm 1 with two constraint parameters L_1 and L_2 respectively are overlapped to a high degree, i.e., most of the voxels selected by Algorithm 1 with different parameters L are the same. This is possibly why the results obtained by Algorithm 1 are not sensitive to L.

Appendix 2: Cross-validation method for setting the number of voxels for prediction of tasks in fMRI data

In this section, we show a cross-validation method for setting θ in Algorithm 1, which determines the number of voxels used to predict the tasks. First, for each subject and task, the data set (including fMRI data and task data) of run 1 and run 2 is equally divided into four parts, each consisting of 250 time points. The first three parts are used for 3-fold cross-validation, while the fourth part is used as an independent test data set. In one of the 3-fold cross-validation, we use two parts for training, and predict the task of the left part. For each value of N_R (1 to 500, the number of selected voxels), a prediction accuracy of the validation feature is obtained. After the 3-fold cross-validation is performed, three the prediction accuracy curves (with respect to N_R) are obtained. An average prediction accuracy curve is then obtained by taking the mean of the three ones. Suppose that this average prediction accuracy curve has maximum at N_0, i.e. if N_0 voxels are selected, the average prediction accuracy is the maximum.

Next, we use any two of the first three parts for training, and obtain a weight vector $\mathbf{w}$. If we rearrange the vector $|\mathbf{w}|$ (the absolute value vector) in descending order, and denote it as $[|w_{i_1}|, \cdots, |w_{i_{500}}|]$, then $\theta = |w_{i_{N_0}}|$. We now predict the task of the independent test set using the N_0 voxels with weights $\{w_{i_1}, \cdots, w_{i_{N_0}}\}$. Three prediction accuracies are obtained using different combinations of two parts for training, i.e. the three folds of cross validation. Similarly as above, we also obtain an average prediction accuracy curve with respect to N_R (1 to 500) for the task of this independent test set. Note that the value of N_0 is the point at which the average of the prediction curves of the three folds is maximum. In each subplot of Fig. 11, an average prediction accuracy curve for the independent test set is presented for one task of one subject and the “*” represents the average prediction accuracy determined by θ. From this figure, we can see that this cross-validation method for determining the parameter θ is acceptable.

Appendix 3: Robustness analysis

In this appendix, we analyze the robustness of Algorithm 1 to noise. Consider the following noisy model corresponding to (2),

$$\min ||\mathbf{w}||_1,\ s.t.,\ (\mathbf{A}+\mathbf{V})\mathbf{w}=\mathbf{y}, \tag{17}$$

where $\mathbf{V}\in R^{N\times M}$ is a noise matrix. The optimal solution of (17) is denoted by $\mathbf{w}_v$. In this paper, $\mathbf{A}$ and $\mathbf{y}$ are a data matrix and a transformed task function, thus we add noise to $\mathbf{A}$ other than $\mathbf{y}$.

Denote all $N\times N$ dimensional submatrices of $\mathbf{A}$ and $\mathbf{V}$ as $\mathbf{A}^{(j)}$ and $\mathbf{V}^{(j)}$ respectively, where $j=1,\cdots,C_M^N$. Since $\mathbf{A}$ is a real data matrix, we can assume that all submatrices $\mathbf{A}^{(j)}$ are full of rank. In this case, it follows from linear programming theory [18] that there is a submatrix say $\mathbf{A}^{(j_0)}$, such that the 1-norm solution $\mathbf{w}_1$ of (2) satisfies

$$\mathbf{w}_1=(\mathbf{A}^{(j_0)})^{-1}\mathbf{y}. \tag{18}$$

That is,

$$||(\mathbf{A}^{(j_0)})^{-1}\mathbf{y}||_1=\min\{||(\mathbf{A}^{(j)})^{-1}\mathbf{y}||_1, j=1,\cdots,C_M^N\}. \tag{19}$$

Note that

$$\lim_{||\mathbf{V}||\to 0}||(\mathbf{A}^{(j)}+\mathbf{V}^{(j)})^{-1}\mathbf{y}||_1=||(\mathbf{A}^{(j)})^{-1}\mathbf{y}||_1,\ \ j=1,\cdots,C_M^N. \tag{20}$$

It follows from (19) and (20) that when $||\mathbf{V}||$ is sufficiently small,

$$||(\mathbf{A}^{(j_0)}+\mathbf{V}^{(j_0)})^{-1}\mathbf{y}||_1=\min\{||(\mathbf{A}^{(j)}+\mathbf{V}^{(j)})^{-1}\mathbf{y}||_1, j=1,\cdots,C_M^N\}. \tag{21}$$

That is, $(\mathbf{A}^{(j_0)}+\mathbf{V}^{(j_0)})^{-1}\mathbf{y}$ is the 1-norm solution $\mathbf{w}_v$ of (17) with sufficiently small noise. Furthermore, since $\mathbf{w}_v$ is close to $\mathbf{w}_1$ in (18), $\mathbf{w}_v$ can be represented by

$$\mathbf{w}_v=\mathbf{w}_1+\triangle\mathbf{w}_v, \tag{22}$$

where $\triangle\mathbf{w}_v$ is a disturbance vector resulted by the noise matrix $\mathbf{V}$. It follows from (20) that $\triangle\mathbf{w}_v$ is small if $||\mathbf{V}||$ is sufficiently small.

In the following, we consider the following noisy model corresponding to the model (8) in Algorithm 1,

$$\min ||\mathbf{w}||_1,\ s.t.,\ (\mathbf{A}_k+\mathbf{V}_k)\mathbf{w}=\mathbf{y}_k, \tag{23}$$

where $\mathbf{V}_k \in R^{L\times M}$ is a noise matrix. The optimal solution of (23) is denoted by $\bar{\mathbf{w}}_v^{(k)}$. From the above analysis, $\bar{\mathbf{w}}_v^{(k)}$ can be represented by

$$\bar{\mathbf{w}}_v^{(k)} = \bar{\mathbf{w}}^{(k)} + \triangle\bar{\mathbf{w}}^{(k)}, \tag{24}$$

where $\bar{\mathbf{w}}^{(k)}$ is the solution of (8).

Thus the output of Algorithm 1 after K iterations in noise case is

$$\mathbf{w}_v^{(K)} = \frac{1}{K}\sum_{i=1}^{K}\bar{\mathbf{w}}^{(i)} + \frac{1}{K}\sum_{i=1}^{K}\triangle\bar{\mathbf{w}}_v^{(i)}. \tag{25}$$

When $||\mathbf{V}_k||$ ($k = 1, \cdots, K$) are sufficiently small, $\frac{1}{K}\sum\limits_{i=1}^{K}\triangle\bar{\mathbf{w}}_v^{(i)}$ is close to zero. This is because: (i) $||\triangle\bar{\mathbf{w}}_v^{(i)}||$ is small, their mean is still small; (ii) the mean of each entry of $\triangle\bar{\mathbf{w}}_v^{(i)}$ is generally zero. Thus we have

$$\mathbf{w}_v^{(K)} \approx \mathbf{w}^{(K)}, \tag{26}$$

$\mathbf{w}^{(K)}$ is the output of Algorithm 1 after K iterations in noiseless case. That is, the weight vector obtained by Algorithm 1 is robust to noise at least to some degree.

Fig. 12 shows three pairs of $\mathbf{w}$ (black) and $\mathbf{w}_v$ obtained by Algorithm 1 in noiseless case and 40dB noise case respectively. When the noise is sufficiently small, we can see that $\mathbf{w}_v$ is close to $\mathbf{w}$.

Now we enlarge the additive noise in model (23) to 25dB, data analysis results (see Figs. 13 and 14) show that the weight vector obtained by Algorithm 1 is still effective for voxel selection.

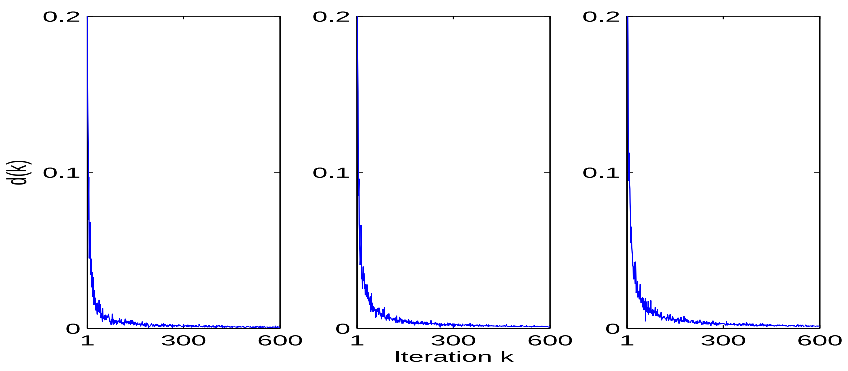

Fig. 1. Three iterative curves demonstrating the convergence of Algorithm 1 with parameter L set to be 25, 50 and 75 respectively, where $d(k)$ is the convergence index of Algorithm 1. The execution time for the three cases are 147s, 426s and 768s respectively.

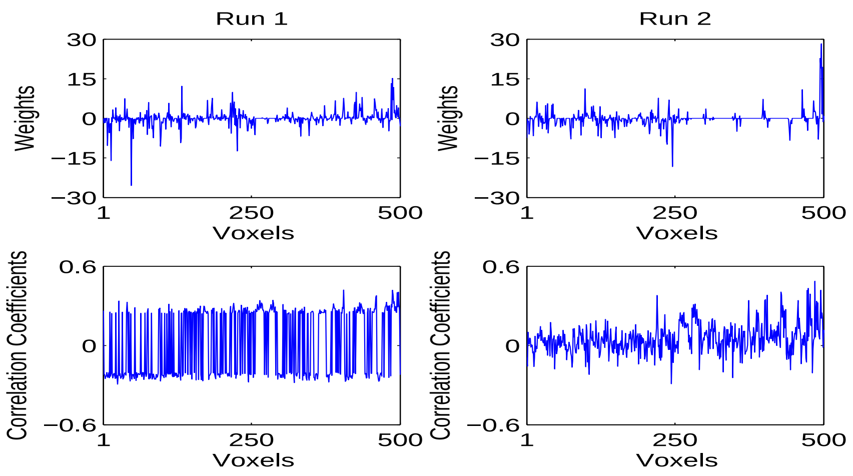

Fig. 2. First row: Weights of voxels in $\mathcal{N}$ obtained by Algorithm 1 for subject 1 and Instructions task; Second row: Correlation coefficients for the same subject and task, each is calculated using the time series of a voxel in $\mathcal{N}$ and the transformed task function. The two columns correspond to two runs (i.e. two folds in cross-validation) respectively.

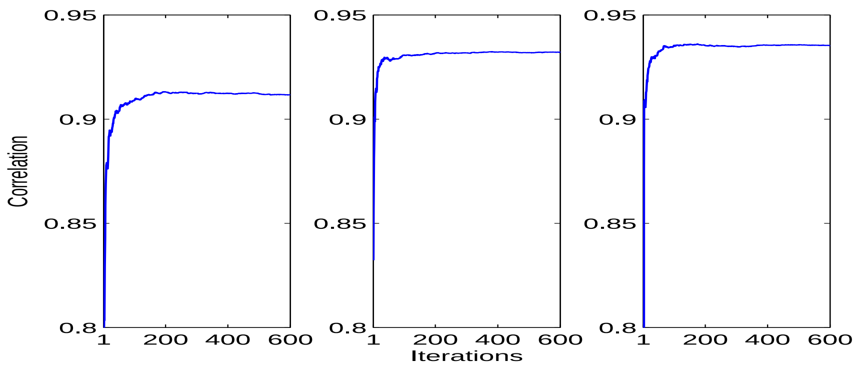

Fig. 3. The iterative curves of correlation between $A\mathbf{w}$ and $\mathbf{y}$ (Instructions task) for three subjects and run 1, each subplot corresponds to one subject.

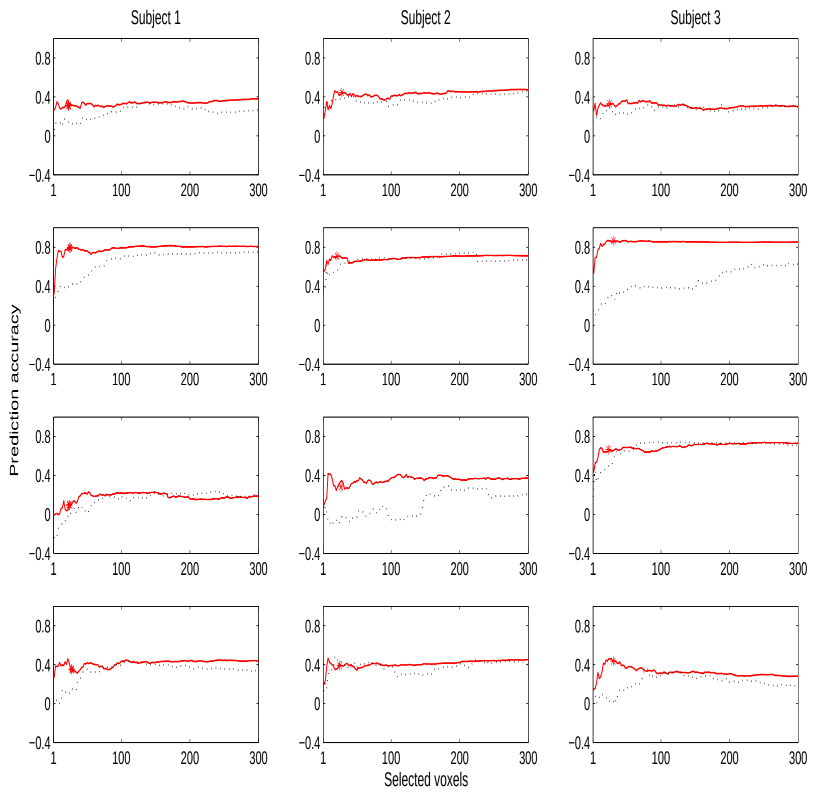

Fig. 4. Prediction accuracy curves obtained by two methods. In each subplot, red solid line: Alg. 1; black dotted line: GLM-SPM method. The four rows correspond to four tasks (Hits, Instructions, Faces, Velocity) respectively. In each subplot, the average prediction accuracy marked by “*” is determined by θ as in Section II-A.

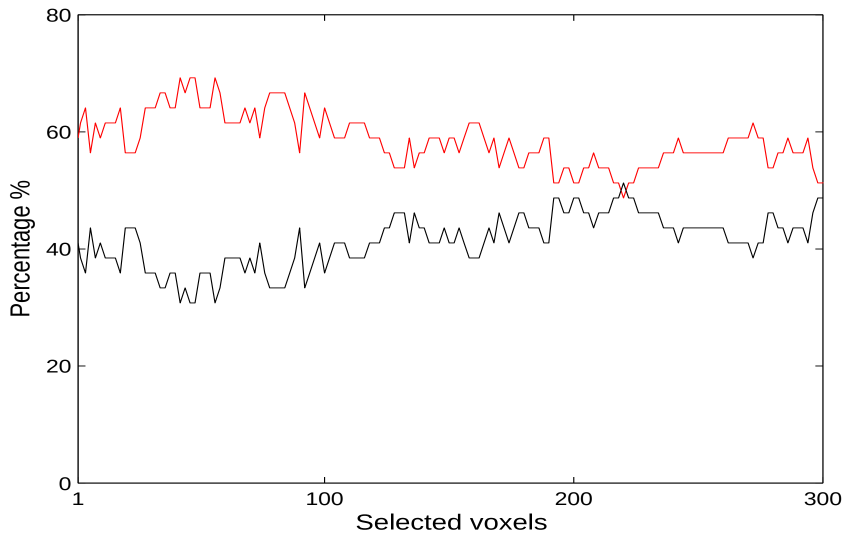

Fig. 5. Two percentage curves (%) showing the performance of two methods in 39 cases (3 subjects and 13 tasks). Red solid line with stars (top): Algorithm 1; dash-dotted line with x-marks: GLM-SPM.

TABLE I

AVERAGE PREDICTION ACCURACY RATES OVER TWO FOLDS OBTAINED WITH ONE VOXEL FOR THREE SUBJECTS, FOUR TASKS AND TWO METHODS. THE ACCURACY RATES OBTAINED BY GLM-SPM METHOD ARE IN BRACKETS

	Sub 1	Sub 2	Sub 3
Hits	0.2656 (0.1308)	0.1819 (0.2143)	0.2595 (0.1451)
Instructions	0.3227 (0.2233)	0.5518 (0.3620)	0.5337 (0.1417)
Faces	-0.0064 (-0.3193)	0.0989 (0.1466)	0.4368 (0.1033)
Velocity	0.2711 (0.0300)	0.1975 (0.0517)	0.1489 (0.1432)

List of Figures

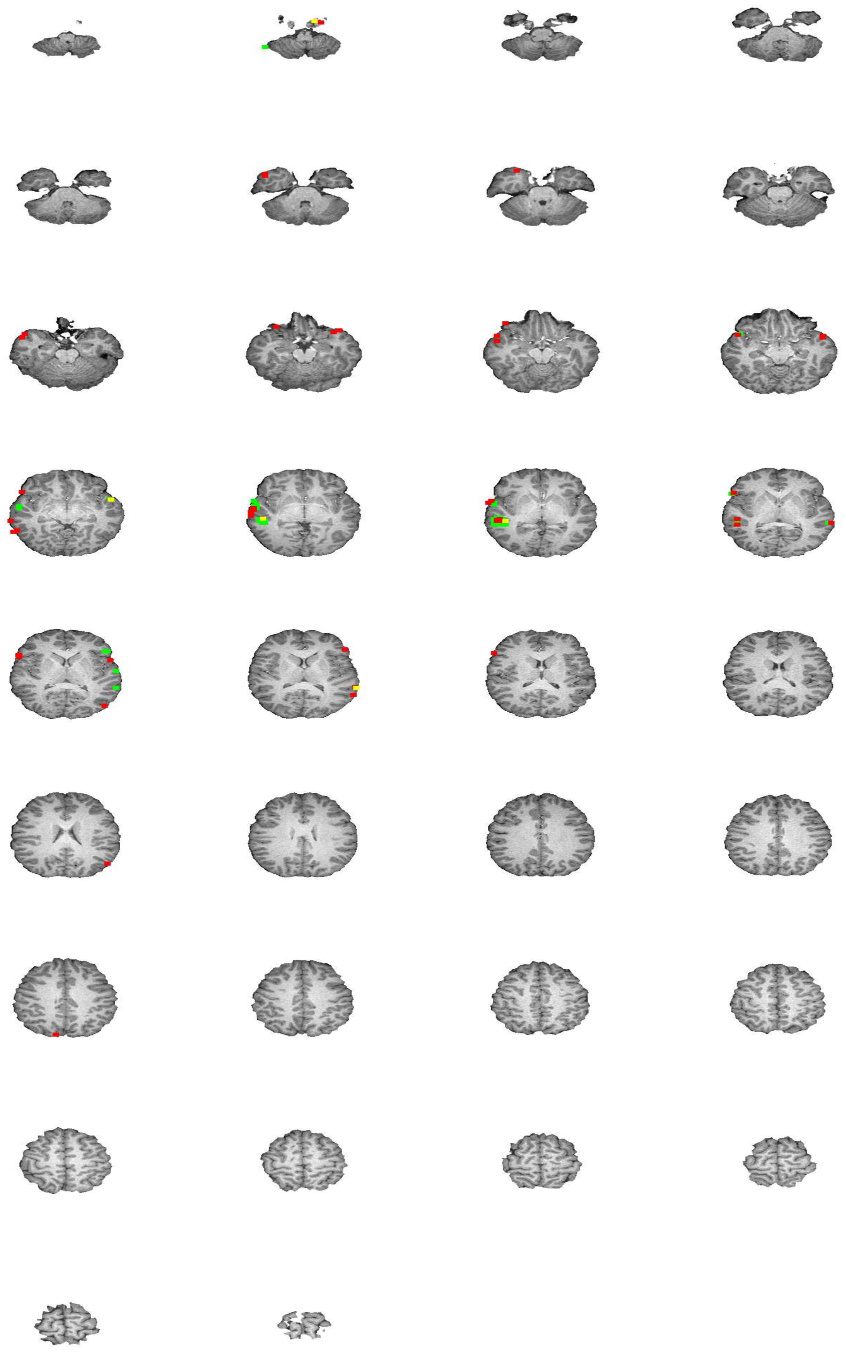

Fig. 6. Distribution of 100 selected Voxels, of which 50 voxels (highlighted in red) correspond to the first 50 highest weights calculated by Algorithm 1, while the other 50 voxels (highlighted in green) correspond to the first highest values of t-statistics calculated by GLM-SPM method for task 2 (Instructions), Run 1 and Subject 3. If two voxels obtained by two methods are the same, then it is highlighted in yellow. Slices are numbered from inferior to the superior parts of the brain.

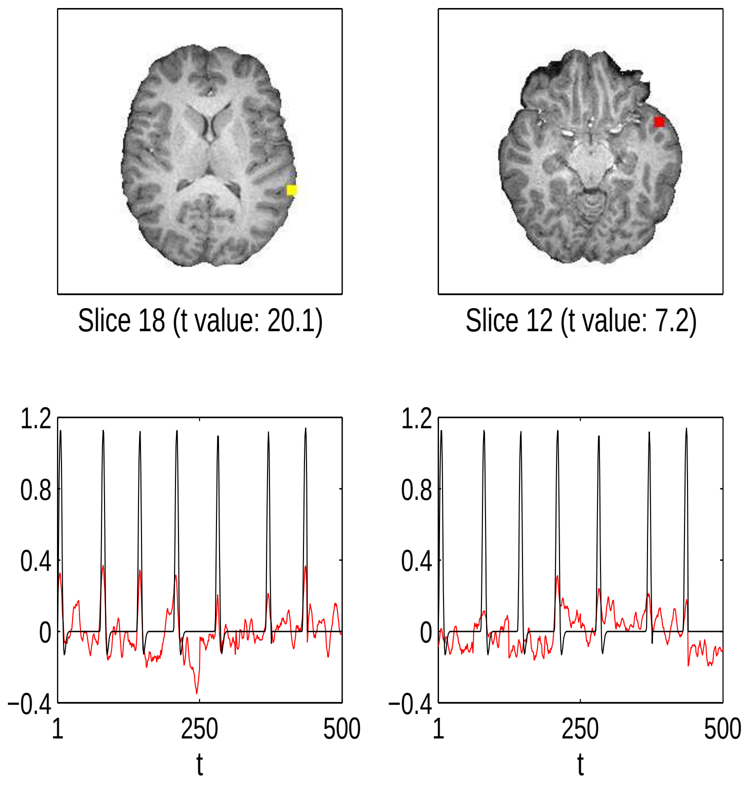

Fig. 7. The first row: Two voxels of which the first one is selected by both Algorithm 1 and GLM-SPM method and the second one is selected only by Algorithm 1. The t values in the brackets are obtained in the SPM detection. The second row: fMRI signals corresponding to the two voxels respectively.

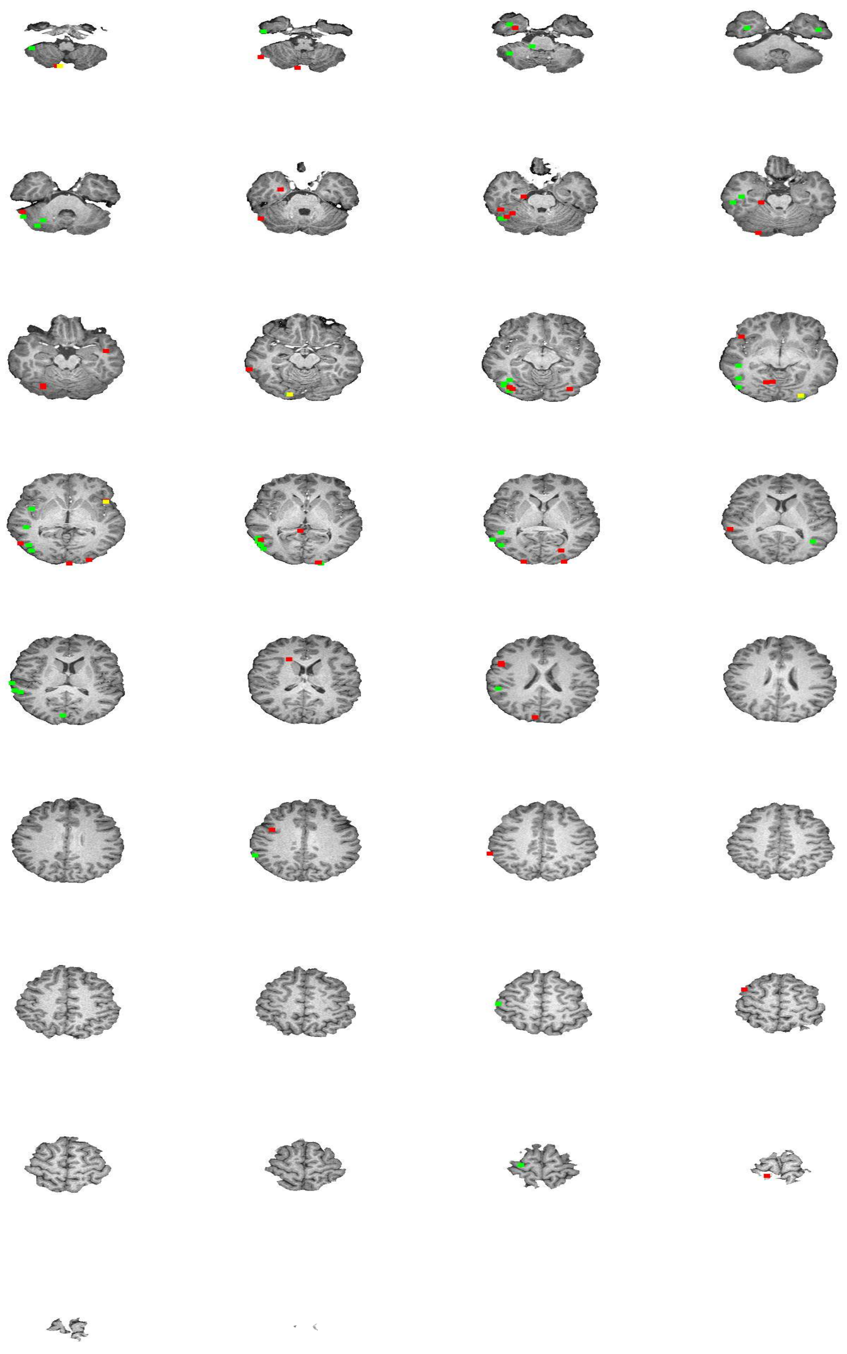

Fig. 8. Distribution of 100 selected Voxels, of which 50 voxels (highlighted in red) correspond to the first 50 highest weights calculated by Algorithm 1, while the other 50 voxels (highlighted in green) correspond to the first highest values of t-statistics calculated by GLM-SPM method for task 3 (Faces), Run 1 and Subject 2. If two voxels obtained by two methods are the same, then it is highlighted in yellow. Slices are numbered from inferior to the superior parts of the brain.

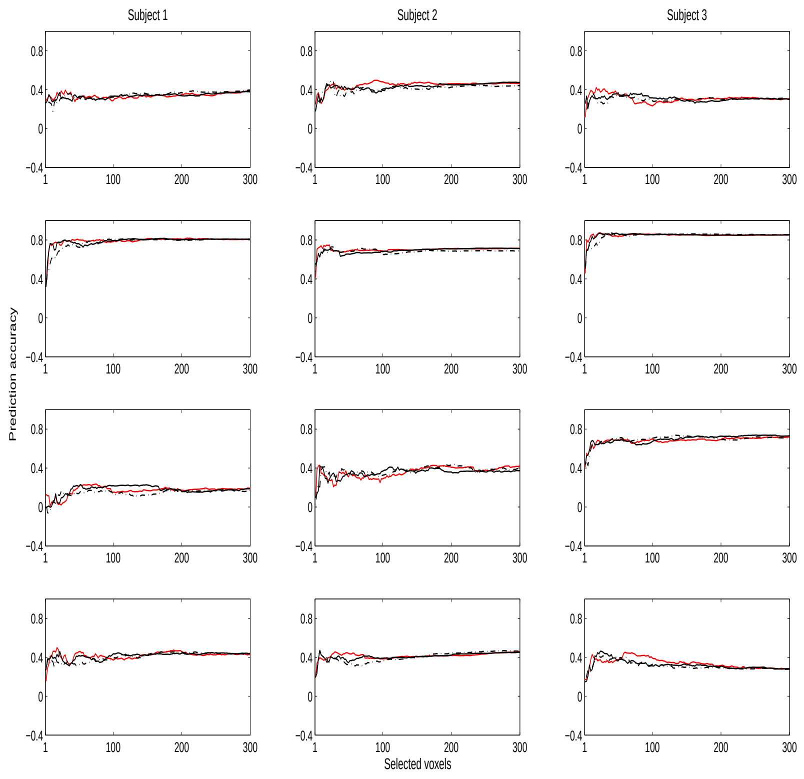

Fig. 9. Three prediction accuracy curves obtained by Algorithm 1 with the numbers of constraints in (8) set to be 25 (red solid line), 50 (black solid line) and 75 (black dashed line) respectively in Appendix 1. The four rows correspond to four tasks (Hits, Instructions, Faces, Velocity) respectively.

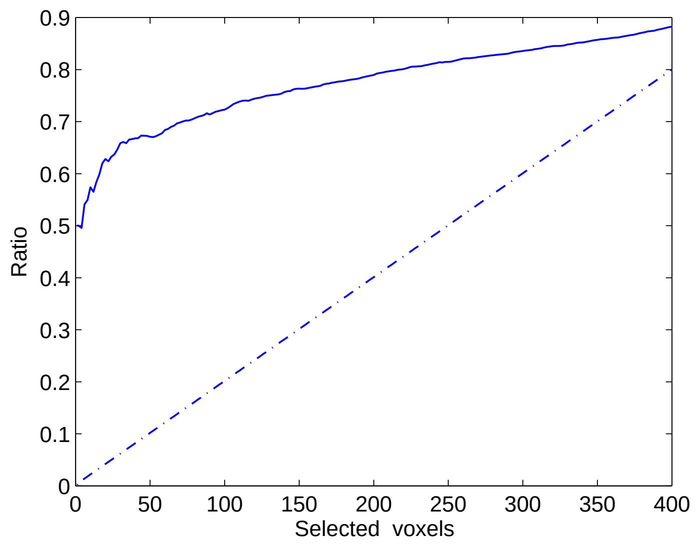

Fig. 10. Ratio curves showing the degree that the two selected voxel sets are overlapped in Appendix 1. The sold line: $r(b_l)$ obtained by Alg. 1; the dash-dotted line: $\bar{r}(b_l)$ obtained by randomly taking two sets of voxels.

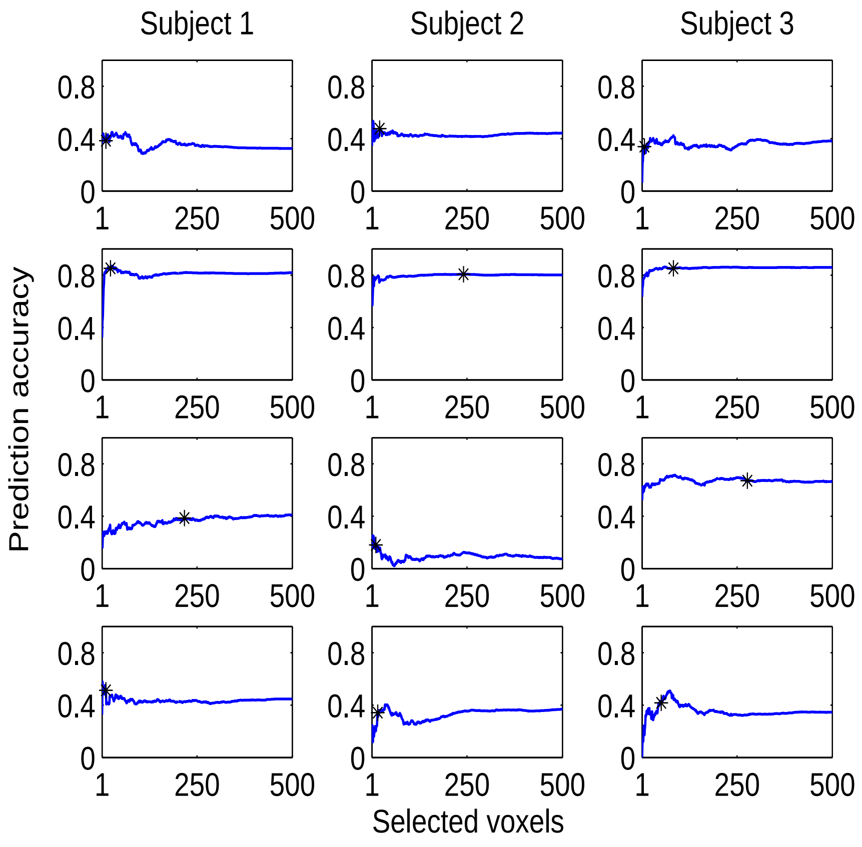

Fig. 11. Average prediction accuracy curves for independent test set for three subjects and four tasks in Appendix 2, where the four rows correspond to the four tasks (Hits, Instructions, Faces, Velocity) respectively. In each subplot, "*": average prediction accuracy determined by θ.

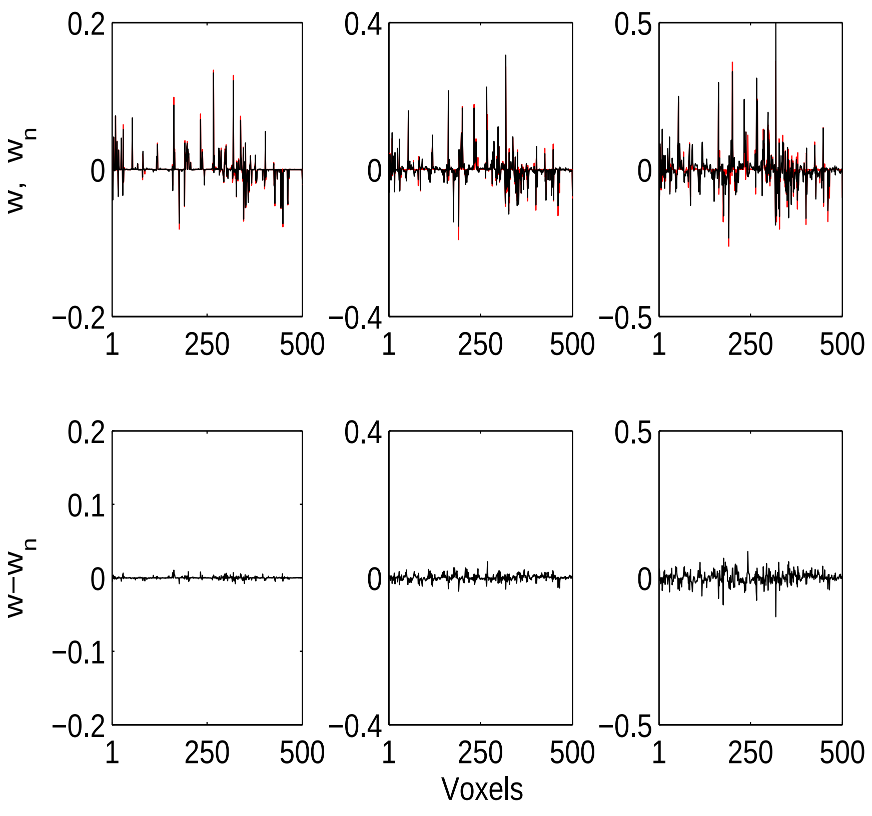

Fig. 12. Each of the three subplots of the first row shows the weight vectors $\mathbf{w}$ (black) and $\mathbf{w}_v$ (red) obtained by Algorithm 1 after 600 iterations in noiseless case and noise case respectively in Appendix 3, where $\mathbf{A}$ and $\mathbf{y}$ are the same fMRI data matrix and transformed task function as in Fig. 1, $\mathbf{w}_v$ corresponds to $40dB$ simulated zero mean Gaussian noise. Each of the three subplots of the second row shows the error $\mathbf{w} - \mathbf{w}_v$. The three columns of this figure correspond to three parameter settings of Algorithm 1: $L = 25, 50, 75$ respectively.

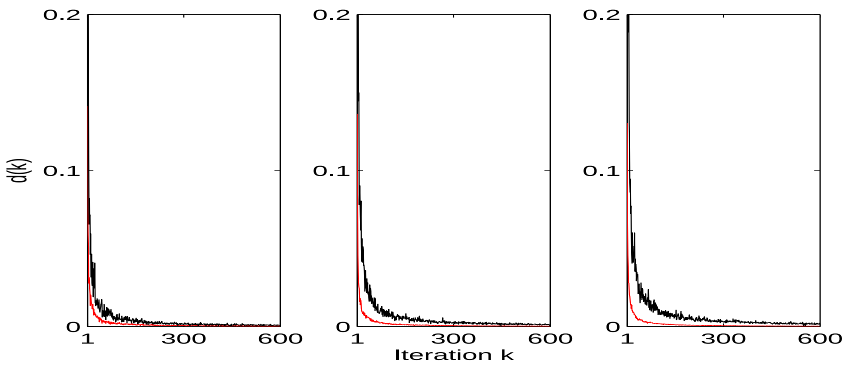

Fig. 13. Each subplot shows two iterative curves of Algorithm 1 obtained in noiseless case (black curve) and 25 dB nose case (red curve) in Appendix 3, where $\mathbf{A}$ and $\mathbf{y}$ are the same fMRI data matrix and transformed task function as in Fig. 1. The three subplots correspond to the three settings $25, 50$ and 75 of the parameter L in Algorithm 1 respectively.

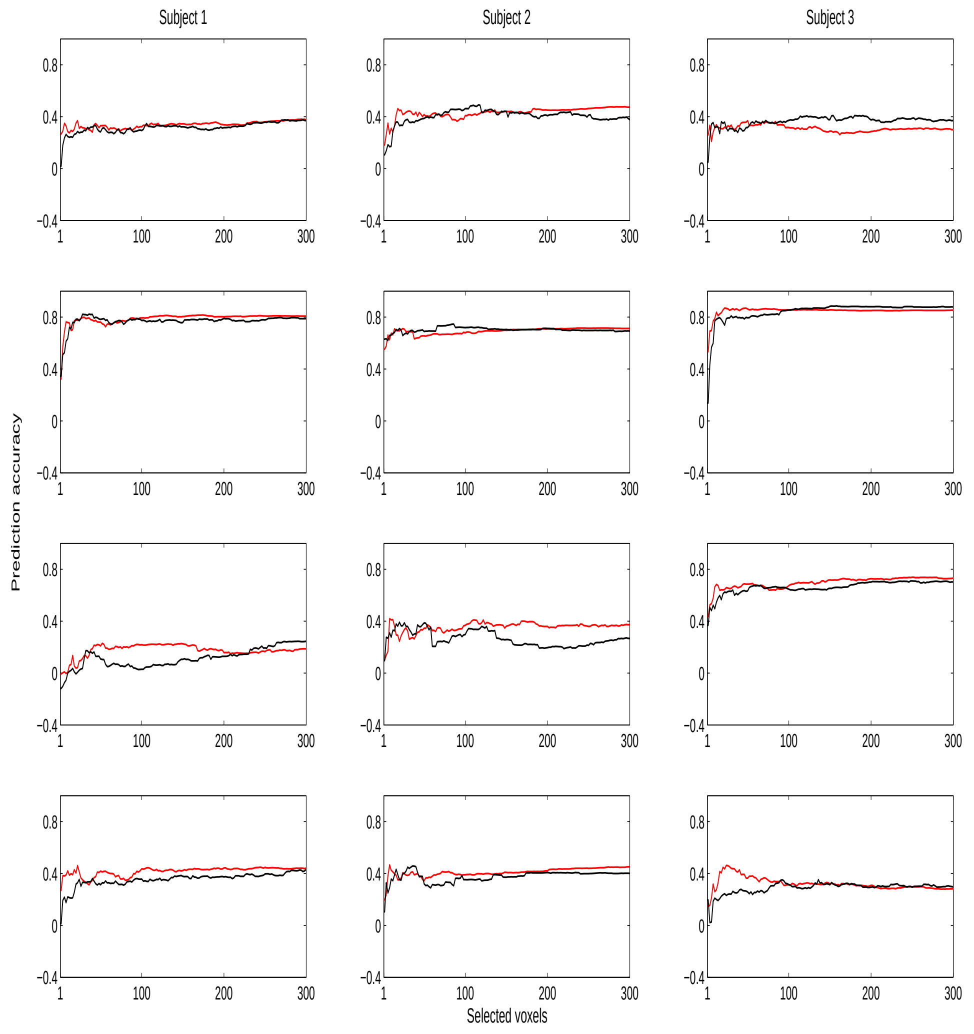

Fig. 14. Each subplot shows two prediction accuracy curves obtained by Algorithm 1 in noiseless case (red curve) and 25dB noise case (black curve) respectively in Appendix 3. The four rows correspond to four tasks (Hits, Instructions, Faces, Velocity) respectively.