NeuroImage 42 (2008) 1414–1429

Contents lists available at ScienceDirect

NeuroImage

journal homepage: www.elsevier.com/locate/ynimg

Sparse estimation automatically selects voxels relevant for the decoding of fMRI activity patterns

Okito Yamashita [a,*], Masa-aki Sato [a], Taku Yoshioka [a,b], Frank Tong [c], Yukiyasu Kamitani [a,b]

[a] ATR Computational Neuroscience Laboratories, Japan
[b] National Institute of Information Technology, Japan
[c] Vanderbilt University, Psychology Department, USA

ARTICLE INFO

Article history:
Received 10 January 2008
Revised 9 May 2008
Accepted 26 May 2008
Available online 6 June 2008

ABSTRACT

Recent studies have used pattern classification algorithms to predict or decode task parameters from individual fMRI activity patterns. For fMRI decoding, it is important to choose an appropriate set of voxels (or features) as inputs to the decoder, since the presence of many irrelevant voxels could lead to poor generalization performance, a problem known as overfitting. Although individual voxels could be chosen based on univariate statistics, the resulting set of voxels could be suboptimal if correlations among voxels carry important information. Here, we propose a novel linear classification algorithm, called sparse logistic regression (SLR), that automatically selects relevant voxels while estimating their weight parameters for classification. Using simulation data, we confirmed that SLR can automatically remove irrelevant voxels and thereby attain higher classification performance than other methods in the presence of many irrelevant voxels. SLR also proved effective with real fMRI data obtained from two visual experiments, successfully identifying voxels in corresponding locations of visual cortex. SLR-selected voxels often led to better performance than those selected based on univariate statistics, by exploiting correlated noise among voxels to allow for better pattern separation. We conclude that SLR provides a robust method for fMRI decoding and can also serve as a stand-alone tool for voxel selection.

Introduction

Conventional fMRI data analysis has primarily focused on voxel-by-voxel functional mapping using the general linear model, in which stimuli or behavioral parameters are used as regressors to account for the BOLD response (Friston et al., 1995; Worsely et. al., 2002). Recently, much attention has been paid to pattern classification, or decoding, as an alternative approach to conventional functional mapping. In this approach, fMRI activation patterns of many voxels can be used to characterize subtle differences between different stimuli or subjects' behavioral/mental states. The pioneering work by Haxby et al. (2001) has demonstrated that broadly distributed fMRI activity patterns can discriminate pictures of visual objects, which cannot be easily distinguished by the conventional functional mapping (see also Strother et al., 2002; Spiridon and Kanwisher, 2002; Cox and Savoy, 2003; Carlson et al., 2003; Mitchell et. al., 2004; Laconte et. al., 2005; O'Toole et al., 2005 for other examples). Furthermore, the decoding approach has proved useful in extracting information about fine-scale cortical representations, which has been thought to lie beyond the resolution of fMRI. Kamitani and Tong (2005, 2006) showed that low-level visual features, such as orientation and motion direction, can be reliably decoded by pooling weakly selective signals in individual voxels. Since cortical columns representing orientation or motion direction are thought to be much smaller than standard fMRI voxels, the signal in each voxel may arise from voxel sampling with biases due to variability in the distribution of cortical feature columns or their vascular supply. Decoding analysis can exploit such subtle information, available in individual voxels, to obtain robust selectivity from the ensemble activity pattern of many voxels ('ensemble feature selectivity'). For comprehensive reviews, see Haynes and Rees (2006) and Norman et al. (2006).

For fMRI decoding, selecting an appropriate set of voxels as the input for classification analysis is important for several reasons. First, voxel selection could improve decoding performance. fMRI decoding analysis takes a form of supervised learning (classification or regression), in which voxel values are the input variables or 'features', and a stimulus/task parameter is the output variable or 'labelled' category. In supervised learning, too many features can sometimes lead to poor generalization performance, a problem called overfitting.

* Corresponding author.
E-mail address: oyamashi@atr.jp (O. Yamashita).

1053-8119/$ – see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.neuroimage.2008.05.050

With many adjustable model parameters associated with the features, the learning model may fit to the noise present in the training data, and generalize poorly to novel test data. In a typical fMRI experiment, only tens or perhaps hundreds of samples (task blocks or volumes) are obtained, while the whole brain can contain as many as a hundred thousand voxels or features. Thus, fMRI decoding can easily lead to overfitting if all available voxels are used as input features. Support vector machines (SVM), one of the most popular classifiers in the fMRI decoding literature, avoids this problem by simultaneously minimizing the empirical classification error and maximizing the margin (Boser et al., 1992; Vapnik, 1998). However, generalization performance of SVM will still be degraded if too many irrelevant features are included.

Second, voxel selection is also useful for understanding neural information coding. Voxels can be selected based on separate anatomical or functional knowledge, so that decoding performance for one set of voxels can be compared with that of another. The higher the performance is, the more likely it is that the voxels represent information relevant to the task. Although careful examination is necessary to determine whether the voxels represent the decoded task parameter or some other correlated variable (Kamitani and Tong, 2005), comparisons of decoding performance for different brain areas can provide a powerful method for mapping the information available in local regions (see also Kriegeskorte et al., 2006).

In most previous studies, voxels have been selected based on anatomical landmarks, functional localizers (e.g., retinotopic mapping), or a voxel-by-voxel univariate statistical analysis obtained from training data or data from a separate experiment. The selected voxels were then used as input features for decoding analysis. An alternative to voxel selection for reducing dimensionality is to project the original feature space into a subspace of fewer dimensions using principal component analysis (PCA) (Carlson et al., 2003) or independent component analysis (ICA). The new dimensions can then be used as input features for decoding analysis. Such two-step methods for feature selection and decoding analysis have proven effective. But they could be suboptimal because the voxel/feature selection step does not take into consideration the discriminability of multi-voxel patterns.

In this paper, we introduce novel linear classification algorithms for binary and multi-class classification, which we will refer to as sparse logistic regression (SLR) and sparse multinomial logistic regression (SMLR), respectively. (Note that the term SLR will be used to refer to both binary and multi-class classifiers if the distinction is not critical). SLR is a Bayesian extension of logistic regression, which simultaneously performs feature (voxel) selection and training of the model parameters for classification. It utilizes automatic relevance determination (ARD) (MacKay, 1992; Neal, 1996) to determine the importance of each parameter while estimating the parameter values. This process selects only a few parameters as important and prunes away others. The resulting model has a sparse representation with a small number of estimated parameters. In fMRI decoding, this sparse estimation approach provides a method for voxel (feature) selection, which could improve decoding performance by avoiding overfitting. Furthermore, voxels selected by SLR may help reveal specific brain regions, within a large set of input voxels, that are relevant to a task.

We use SLR not only as an alternative to conventional classification methods such as Fisher's linear discriminant and SVM, but also as a feature selection or 'feature ranking' tool. To rank the relevance of voxels, we apply SLR repeatedly to sets of samples randomly selected from training data, and obtain an overall rank of each voxel based on its frequency of selection. After voxels are selected by this ranking procedure, any model or algorithm could be used for classification. We use an independent test data set, which is not used for feature selection, in evaluating classification performance. If test data were implicated in the feature selection process, performance would become erroneously better than chance even in the absence of discriminable patterns (Baker et al., 2007).

In this study, we first evaluate the performance of SLR using simulated data and then demonstrate characteristics of voxels selected by SLR using two real fMRI data sets. Using simple simulation data, we show that SLR can indeed select relevant features (voxels) among a large number of irrelevant ones. This allows SLR to maintain high classification performance in the presence of irrelevant features whereas other classification methods, such as SVM and regularized logistic regression (RLR), are less robust. Next, we apply SLR to fMRI data obtained while observers viewed a stimulus in one of four visual quadrants. We find that SLR selects voxels whose locations are consistent with known functional anatomy. Using fMRI data of orientation grating stimulus experiments, we also find that SLR-selected voxels may differ from those selected by univariate comparisons of different task conditions and can lead to superior performance. Further analyses suggest that SLR can exploit correlations among voxels, which cannot be detected by the conventional univariate statistics, and thereby attain superior decoding performance.

Methods

Classification algorithm

In this section, we first describe logistic regression (LR) and multinomial logistic regression (MLR), which provide probabilistic models for solving binary and multi-class classification problems, respectively. The parameters in the models are estimated by the maximum likelihood method. This method, however, can only be applied when the number of samples is larger than the number of features. Here, the logistic regression method is extended to a Bayesian framework by using a technique known as the automatic relevance determination (ARD) from the neural network literature (MacKay, 1992; Neal, 1996). By combining LR or MLR with the ARD, sparse logistic regression (SLR) or sparse multinomial logistic regression (SMLR) is obtained. ARD provides an effective method for pruning irrelevant features, such that their associated weights are automatically set to zero, leading to a sparse weight vector for classification. Throughout the paper, scalars and vectors are denoted by italic normal face letters (e.g. x, θ) and by bold-faced letters (e.g. $\mathbf{x}$, $\boldsymbol{\theta}$), respectively. The transpose of a vector $\mathbf{x}$ is denoted by $\mathbf{x}^{\mathrm{t}}$.

Logistic regression (LR)

The linear discriminant function that separates two classes S_1 and S_2 is represented by the weighted sum of each feature value;

$$f(\mathbf{x};\boldsymbol{\theta}) = \sum_{d=1}^{D} \theta_d x_d + \theta_0, \tag{1}$$

where $\mathbf{x}=(x_1, \ldots, x_D)^t \in \mathbb{R}^{\mathbb{D}}$ is an input feature vector in D dimensional space and $\theta=(\theta_0, \theta_1, \ldots, \theta_D)^t$ is a weight vector including a bias term (hereafter we may omit a bias term). The hyper-plane $f(\mathbf{x};\theta)=0$ determines the boundary between two classes. LR allows one to calculate the probability that an input feature vector belongs to category S_2, through a logistic function,

$$p = \frac{1}{1+\exp(-f(\mathbf{x};\boldsymbol{\theta}))} \equiv P(S_2|\mathbf{x}). \tag{2}$$

Note that p ranges from 0 to 1, and is equal to 0.5 when $f(\mathbf{x};\theta)=0$ (i.e. on the boundary) and approaches to 0 or 1 when $f(\mathbf{x};\theta)$ approaches minus infinity or infinity (i.e. far from the boundary). This probability p is interpreted as the probability that an input feature vector $\mathbf{x}$ belongs to class S_2 (conversely, $\mathbf{x}$ belongs to class S_1 with probability $1-p$).

For mathematical formulation, let us introduce a binary random variable y such that $y=0$ for S_1 and $y=1$ for S_2. Given N input–output data samples $\{(\mathbf{x}_1,y_1),\ldots, (\mathbf{x}_N, y_N)\}$, the likelihood function is expressed as

$$P(y_1,\ldots,y_N|\mathbf{x}_1,\ldots,\mathbf{x}_N;\boldsymbol{\theta}) = \prod_{n=1}^{N} P(y_n|\mathbf{x}_n;\boldsymbol{\theta}) = \prod_{n=1}^{N} p_n^{y_n}(1-p_n)^{1-y_n}, \tag{3}$$

$$p_n \equiv P(y_n=1|\mathbf{x}_n;\boldsymbol{\theta}) = \frac{1}{1+\exp(-f(\mathbf{x}_n;\boldsymbol{\theta}))}. \tag{4}$$

Since each term in the product of Eq. (3) represents the probability of observing the nth sample (p_n if $y_n=1$, $1-p_n$ if $y_n=0$), the product represents the probability observing the entire set of data samples. Thus we would like to find the parameter vector θ such that this likelihood function is maximized. This maximization is equivalent to maximizing the logarithm of the likelihood function,

$$l(\boldsymbol{\theta}) = \sum_{n=1}^{N} [y_n \log p_n + (1-y_n)\log(1-p_n)].$$

The function $l(\boldsymbol{\theta})$ is a rather complicated nonlinear function because p_n implicitly depends on a parameter vector $\boldsymbol{\theta}$. Since the gradient and Hessian of $l(\boldsymbol{\theta})$ can be obtained in the closed form, this maximization can be done by the Newton method efficiently (Bishop, 2006, pp. 205–208). It is noted that this optimization always converges to a unique maximum point because the Hessian matrix is positive definite everywhere in the parameter space. For a test sample $\mathbf{x}_{\text{test}}$ for which the class is unknown, the class S_2 is assigned if $f(\mathbf{x};\boldsymbol{\theta})>0$ (or equivalently if $p_{\text{test}}>0.5$), and the class S_1 if $f(\mathbf{x};\boldsymbol{\theta})<0$.

Multinomial logistic regression (MLR)

In case of a multi-class classification problem, each class has its own linear discriminant function;

$$f_c\left(\mathbf{x};\boldsymbol{\theta}^{(c)}\right) = \sum_{d=1}^{D} \theta_d^{(c)} x_d + \theta_0^{(c)} \quad c=1,\ldots,C, \tag{5}$$

where C is the number of classes. Then the probability of observing one of classes S_c is calculated using the softmax function (Bishop, 2006, p. 356) as

$$P(S_c|\mathbf{x}) = \frac{\exp\left(f_c\left(\mathbf{x};\boldsymbol{\theta}^{(c)}\right)\right)}{\sum_{k=1}^{C}\exp\left(f_k\left(\mathbf{x};\boldsymbol{\theta}^{(k)}\right)\right)} \quad c=1,\ldots,C. \tag{6}$$

Note that the number of weight parameters to be estimated is $C\times D$ since each class has its own weight parameter vector (Fig. 1(a)). Using training data, these weight parameters are estimated by maximizing the following likelihood function,

$$P(\mathbf{y}_1,\ldots,\mathbf{y}_N|\mathbf{x}_1,\ldots,\mathbf{x}_N;\theta) = \prod_{n=1}^{N}\prod_{c=1}^{C} p_n^{(c)y_n^{(c)}} \tag{7}$$

$$p_n^{(c)} = P(S_c|\mathbf{x}_n) \equiv P\left(y_n^{(c)}=1|\mathbf{x}_n;\Theta\right) = \frac{\exp\left(\mathbf{x}_n^t\boldsymbol{\theta}^{(c)}\right)}{\sum_{k=1}^{C}\exp\left(\mathbf{x}_n^t\boldsymbol{\theta}^{(k)}\right)}. \tag{8}$$

This maximization is again attained by the Newton method since the gradient and Hessian can be written in a closed form (Bishop, 2006, pp. 209–210). In order to treat the multi-class output label in a similar way to Eq. (3), a binary vector $\mathbf{y}=[y^{(1)},\ldots,y^{(C)}]$ is introduced such that $y^{(c)}=1$ if $\mathbf{x}$ belongs to class S_c and $y^{(c)}=0$ otherwise. For a test sample $\mathbf{x}_{\text{test}}$ for which the class is unknown, the class that maximizes $P(S_1|\mathbf{x}_{\text{test}}),\ldots, P(S_C|\mathbf{x}_{\text{test}})$ is assigned.

Automatic relevance determination (ARD)

For neuroimaging data, it is often the case that the number of samples is fewer than the number of features (voxels). LR and MLR are not applicable to such data because of the ill-conditioned Hessian matrix. Therefore, some constraint must be imposed on the weight parameters. One method is to introduce a regularization term using L2 norm, or equivalently, to assume a Gaussian prior distribution with a zero mean vector and a spherical covariance matrix (regularized logistic regression, RLR). Automatic relevance determination (ARD) can also attain this end by assuming a Gaussian prior with a zero mean vector and a diagonal covariance matrix whose diagonal elements are adjustable hyper-parameters regulating possible values of corresponding weight parameters.

RLR assumes the prior distribution given by $P(\boldsymbol{\theta}|\alpha)=N(0, \alpha^{-1}I_D)$ where I_D is the identity matrix of size $D\times D$. SLR assumes the ARD prior given by

$$P(\theta_d|\alpha_d) = N\left(0,\alpha_d^{-1}\right) \quad d=1,\ldots,D, \tag{9}$$

where θ_d is the dth element of θ. The difference between RLR and SLR priors is that all of the weight parameters share one single variance parameter in RLR whereas every weight parameter has its own adjustable variance parameter in SLR. In the full-Bayesian formulation, SLR further assumes the non-informative prior distribution for hyper-parameters,

$$P_0(\alpha_d) = \alpha_d^{-1} \quad d=1,\ldots,D. \tag{10}$$

The hyper-parameter is referred to as the *relevance parameter*. This parameter controls the possible range of a corresponding weight parameter (see Fig. 1(b)).

The ARD prior can be applied to MLR in a similar way. The priors are assumed for each element in the weight parameter vectors of each class $\theta^{(1)}$, $\theta^{(2)}$,..., $\theta^{(c)}$,

$$P\left(\theta_d^{(c)}|\alpha_d^{(c)}\right) = N\left(0,\alpha_d^{(c)^{-1}}\right) \quad d=1,\ldots,D,\ c=1,\ldots,C \tag{11}$$

$$P_0\left(\alpha_d^{(c)}\right) = \alpha_d^{(c)^{-1}} \quad d=1,\ldots,D,\ c=1,\ldots,C. \tag{12}$$

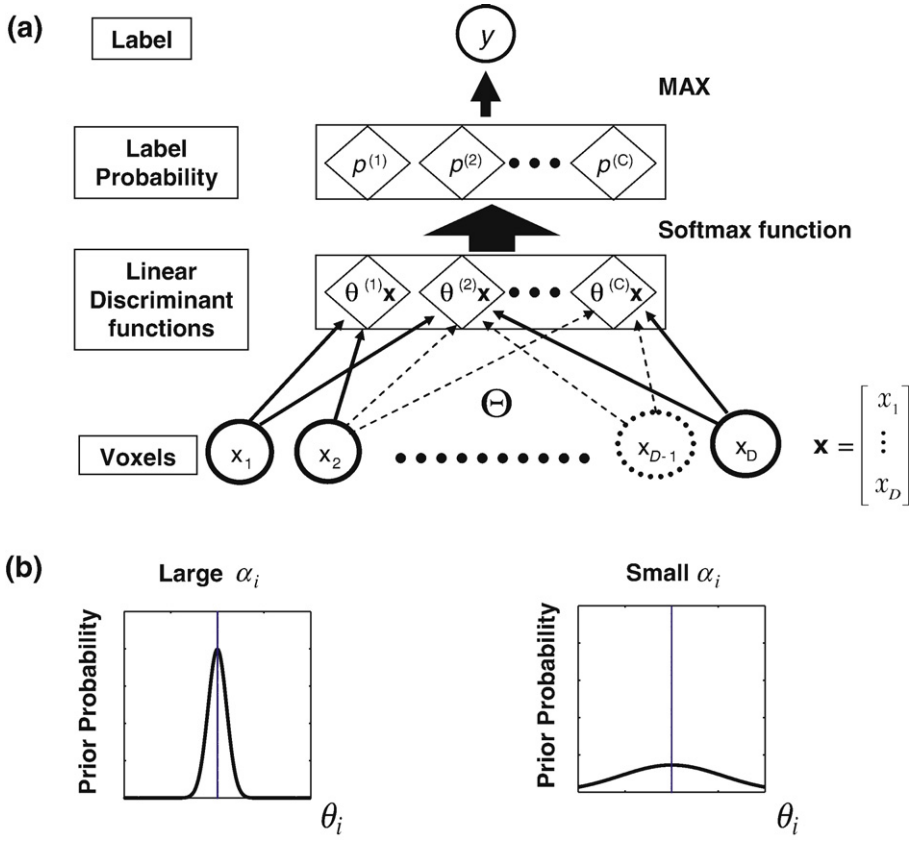

Fig. 1. Two elements of sparse logistic regression (SLR): the (multinomial) logistic regression model (a) and the automatic relevance determination (ARD) (b). (a) Each class or label has its own discriminant function, which calculates the inner product of the weight parameter vector of the label (θ) and an input feature vector ($\mathbf{x}$). The softmax function transforms the outputs of the discriminant functions to the probability of observing each label. The label with the maximum probability is chosen as the output label. Binary logistic regression is slightly different from multinomial logistic regression. The probability can be calculated by the logistic transformation of a single discriminant function that separates two classes (corresponding to $(\theta_1-\theta_2)^t\mathbf{x}$). SLR uses this conventional model for (multinomial) logistic regression, but the estimation of weight parameters involves a novel algorithm based on the automatic relevance determination. (b) SLR treats the weight parameters as random variables with prior distributions. The prior of each parameter θ_i is assumed to have a Gaussian distribution with mean 0. The precision (inverse variance) of the normal distribution is regarded as a hyper-parameter α_i, called a relevance parameter, with a hyper-prior distribution defined by a gamma distribution. The relevance parameter controls the range of the corresponding weight parameter. If the relevance parameter is large, the probability sharply peaks at zero as prior knowledge (left panel), and thus the estimated weight parameter tends to be biased toward zero even after observation. On the other hand, if the relevance parameter is small, the probability is broadly distributed (right panel), and thus the estimated weight parameter can take a large value after observation. While our iterative algorithm computes the posterior distributions of the model, most relevance parameters diverge to infinity. Thus, the corresponding weight parameters become effectively zeros, and can be pruned from the model. This process of determining the relevance of parameters is called the ARD. For the details of the algorithm, see Appendix A.

The probabilistic classification models, consisting of Eqs. (3), (4), (9), (10) and (7), (8), (11), (12), are called sparse logistic regression (SLR) and sparse multinomial logistic regression (SMLR), respectively. The weight parameters are estimated as the marginal posterior mean. Since the marginal posterior distributions cannot be derived in a closed form, we apply the variational Bayesian approximation and the Laplace approximation (see Appendix A for details). The algorithm to calculate the posterior mean becomes an iterative algorithm that alternately updates two equations: weight parameters are updated while fixing relevance parameters and relevance parameters are updated while fixing weight parameters. It turns out that most of the estimated α_d s diverge to infinity, and thus the corresponding weights (θ_d s) become zeros. As a result, the solution leads to a sparse model where many of the features x_d are effectively pruned. Faul and Tipping (2002) have analyzed the mechanism of sparsity from a mathematical perspective.

SLR-based feature selection procedure

Here, we describe the method for feature selection based on SLR. The decoding procedure consists of three steps; (1) feature selection, (2) training of a classifier and (3) evaluation of generalization performance. This section focuses on the feature selection method in the first step. Any classifier such as Fisher's linear discriminant, LR, SVM or SLR, could be used in the second and the third steps, once the features are selected.

Our method can be regarded as a kind of *variable ranking method*. The variable ranking method involves assigning a score to every feature, and then selecting a certain number of features according to their scores. For example, in the *T*-value ranking method, a *T*-value that quantifies the statistical difference between two conditions of interest is assigned to every feature. Features are then selected based on their *T*-values.

SLR can be used to assign scores to features based on the classification performance and selection frequency, which we refer to as selection counting value (SC-value). The basic idea is that features that are repeatedly selected with good classification performance among a variety of training data sets could be important, so high SC-values should be assigned. To implement the idea, we use a form of cross validation. The first step to computing SC-values is to divide the original training data set into two data sets according to some pre-

specified proportion (for example, 80%–20%). One data set is used for estimating weight parameters by SLR and the other used for evaluating classification performance. By generating a number of random divisions of the original data set and repeating the steps of parameter estimation and performance evaluation, we can obtain numerous estimates of the weight vectors with their corresponding measures of classification accuracy. Then the SC-value can be defined by the total frequency of each feature selected, weighted by classification accuracy (see Fig. 2 for schematics). More precisely speaking, let $\overline{\theta}(k)$ and $p(k)$ denote the estimated parameter vector and classification performance (percent) resulting from the kth division. Then the SC-value for the dth feature is defined by

$$\mathrm{SC}(d) = \sum_{\{k:p(k)>p_{\mathrm{chance}}\}}^{K} I\left(\overline{\theta}_d(k)\neq 0\right) \times p(k) \quad d = 1,\ldots,D, \tag{13}$$

where $I(\bullet)$ denotes an indicator function that takes the value of 1 if the condition inside the brackets is satisfied, 0 otherwise. K is the number of repetitions and $\overline{\theta}_d(k)$ is the estimate of the dth element of $\overline{\boldsymbol{\theta}}(k)$. In order to exclude the results of poor classification performance, only data sets with classification performance exceeding chance level p_{chance} are included in the summation. It should be noted that an additional data set that is not used for calculating SC-values is required to evaluate generalization performance of the selected features.

Shuffle measure

We developed a shuffle measure to quantify the effect of correlations between features on classification performance. The shuffle measure quantifies the extent to which classification performance is facilitated by the presence of correlated structures in the data. Computing the shuffle measure involves randomly permuting (shuffling) the order of samples within each class and each feature dimension and then comparing classification performance using the original input data with that using the shuffled input data. The rationale behind shuffling is that random permutation of the order of samples will remove any correlations between features while preserving the local set of values observed for each feature and condition.

For simplicity, let us consider a binary classification problem and focus on one feature dimension. Samples of this feature dimension are denoted by a column vector $\mathbf{z}=[z_1^{(1)},\ldots, z_N^{(1)}, z_1^{(2)}, \ldots, z_N^{(2)}]^{\mathrm{t}}=[\mathbf{z}^{(1)},\mathbf{z}^{(2)}]^{\mathrm{t}}$, where $z_n^{(c)}$ is the nth sample of class c and $\mathbf{z}^{(c)}$ is collection of samples belonging to class c. By randomly permuting the order of samples within $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$separately, we obtain shuffled feature values $\mathbf{z}_{\mathrm{shuf}}=[\mathbf{z}_{\mathrm{shuf}}^{(1)},\mathbf{z}_{\mathrm{shuf}}^{(2)}]^{\mathrm{t}}$. After the shuffle operation above, any correlations between the shuffled feature dimension and the other feature dimensions are eliminated. Note that shuffling does not change the marginal distribution because only the order of samples is changed. By applying the shuffle operation to all feature dimensions, the correlations between all pairs of the dimensions can be removed. Using the shuffled feature values and the original feature values, we define the shuffle measure as follows;

- Train parameters of a classifier using the original feature values and then evaluate classification accuracy (percent correct) p_{original}.
- Train parameters of a classifier using the shuffled feature values (shuffling applied to all the dimensions) and then evaluate classification accuracy p_{shuffle}.
- The shuffle measure is defined by $p_{\mathrm{correlation}}=p_{\mathrm{orginal}}-p_{\mathrm{shuffle}}$.

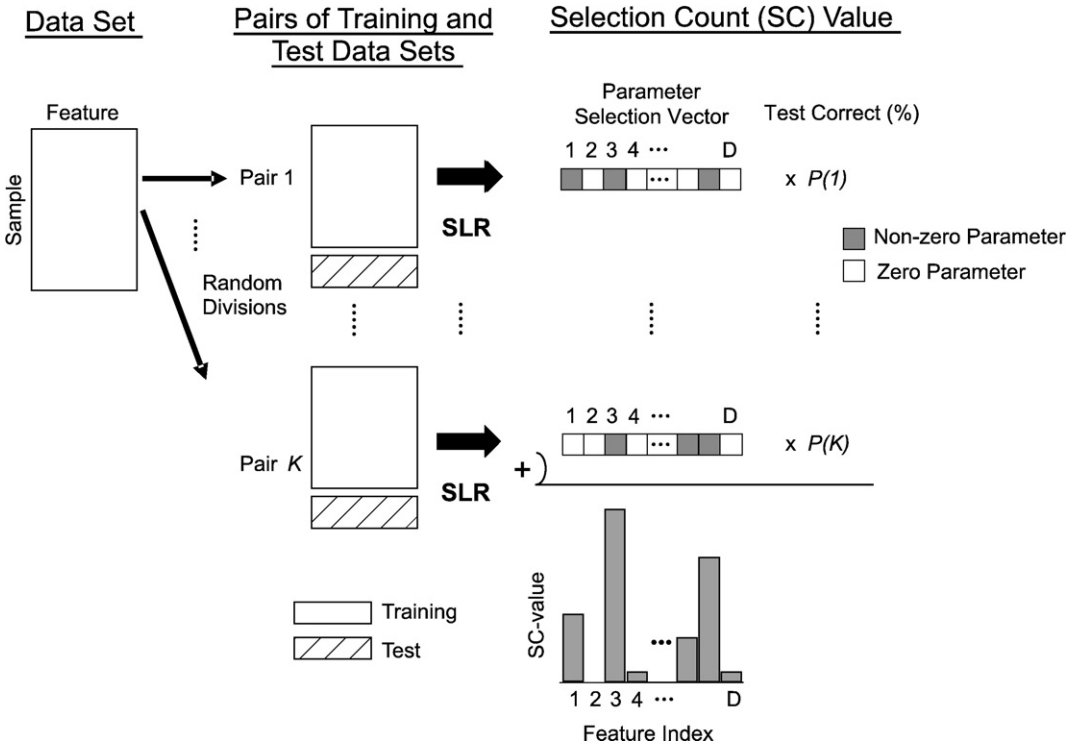

Fig. 2. SLR-based voxel ranking procedure. A whole data set is randomly separated into K pairs of training and test data sets. For each pair, SLR is applied to the training set to learn the weight parameters, which results in sparse parameter selection. Then, the classification performance is evaluated using the test set. The score of each parameter (SC-value) is defined by the count of selection in K-time SLR estimations, weighted by the corresponding test performance (percent correct).

Since the shuffle operation in step 2 uses random numbers, p_{shuffle} should be calculated as the average of many repetitions of step 2 in order to remove potential noise due to random numbers.

Simulation data

Data generation

The simulation data were generated from two D dimensional normal distributions with mean $\boldsymbol{\mu}_1$ and covariance Σ_1 for class 1, and with mean $\boldsymbol{\mu}_2$ and covariance Σ_2 for class 2, respectively. The means and covariances are given by

$$\mu_1 = [\underbrace{0.1, 0.2, \ldots, 0.9, 1.0}_{10}, \underbrace{0, \ldots, 0}_{D-10}],$$

$$\mu_2 = \underbrace{[0, 0, \ldots, 0]}_{D} \quad \Sigma_1 = \Sigma_2 = \begin{bmatrix} 1 & & & 0 \\ & 1 & & \\ & & \ddots & \\ 0 & & & 1 \end{bmatrix},$$

where D can equal 10, 100, 500, 1000, 1500, 2000 input features. Only the first 10 features were relevant for the two-class classification; the remaining $D-10$ features were irrelevant. The degree of relevance in the first 10 dimensions was manipulated by the mean values in class 1, which started from 0.1 and increased up to 1.0 by 0.1. Each dimension had a variance of 1, and there was no correlation between dimensions. For each D, 100 training samples and 100 test samples (50 for each class) were created.

Data analysis

We analyzed the simulation data using three classifiers; SLR, linear regularized logistic regression (RLR) and linear support vector machine (SVM). For estimating weight parameters of SLR and RLR, we used the algorithms in Appendix A. The numbers of iterations were set to 500 for SLR and 50 for RLR, respectively. For SVM, we used LIBSVM (http://www.csie.ntu.edu.tw/~cjlin/libsvm/) and applied all the default parameters (e.g. trade-off parameter $C=1$). Each feature was separately normalized to have mean 0 and variance 1 before applying SLR, RLR or SVM. We evaluated test performance by performing 200 Monte Carlo simulations; the results show average classification performance and its standard error.

Four quadrant stimuli experiment

Data acquisition

To evaluate the efficacy of the SLR classifier, we conducted a simple visual experiment in which stimuli were presented in one of the four visual quadrants on every stimulus block, and the location of the stimulus had to be decoded.

In the four quadrant stimuli experiment, four healthy subjects who gave written informed consent participated. The experiment relied on a conventional block design. Red and green (CIE coordinates 0.346, 0.299 and 0.268, 0.336, respectively) checkerboards appeared in one of four quadrants (upper right, lower right, lower left and upper left, abbreviated as 'UR', 'LR', 'LL' and 'UL' hereafter) for 15 s, followed by 15 s of a fixation period ('F'). One run consisted of 3 repetitions of UR-F–LR-F–LL-F–UL-F blocks. The order of blocks was not randomized. Each subject conducted 6 runs. During the experiment, echo-planar images of the whole brain were obtained (TR = 3 s, TE = 49 ms, FA 90 degrees, FOV 192×192 mm, 30 slices, voxel size of 3×3×5 mm; no gap; 64×64 matrix) using a 1.5 T MRI scanner (Shimadzu Marconi, MAGNEX ECLIPSE). A three dimensional anatomical scan was also acquired from each subject using a T1-weighted RF-FAST sequence (TR/TE/TI/NEX, 20 ms/2.26 ms/-/1 FA 40 degrees, FOV 256 mm×256 mm and 256×256 matrices), yielding sagittal slices with a slice thickness of 1 mm and an in-plane resolution of 1×1 mm. Furthermore a three dimensional anatomical scan with the same position as EPIs was acquired using a T2-weighted RF-FAST sequence (TR/TE/TI/NEX, 5468 ms/80 ms/-/2 FA 90 degrees, FOV 192 mm×192 mm and 256×256 matrices), yielding transverse slices with a slice thickness of 5 mm and an in-plane resolution of 0.75×0.75 mm.

Data analysis

The following fMRI preprocessing steps were used. For motion artifact removal, EPI images were realigned to the first EPI scan and coregistered to T2 anatomical image. No spatial smoothing was applied. The SPM2 toolbox (http://www.fil.ion.ucl.ac.uk/spm/) was used for image processing.

For classification analysis, the input feature vector consisted of the time-averaged BOLD response of each voxel for each stimulus block, using all voxels available in the occipital lobe. The occipital lobe (approximately 1500 voxels) was identified by converting the occipital lobe of the standard MNI brain (Maldjian et. al. 2003, 2004) to that of an individual brain using the SPM2 deformation toolbox. Average BOLD responses for each stimulus block were calculated based on the average signal level for volumes 2 to 5 after stimulus onset (i.e., 6–15 s post-stimulus), following baseline correction by subtracting the average response within each run. Finally, average BOLD response of each voxel for each block was concatenated across all runs to form a vector. Vectors from many selected voxels, with labels indicating the stimulus condition, served as input to the classifier.

We evaluated the performance of SMLR on the four quadrant data by using a leave-one-run-out cross-validation procedure. Five of the six runs were used as training data (60 samples) and the remaining run served as test data (12 samples); this process was repeated for all runs. Feature vectors were normalized such that each voxel had mean 0 and variance 1, using linear scaling factors computed from training data.

The same scaling factors were applied to test data. We performed the same analysis using a multi-class version of RLR called regularized multinomial logistic regression (RMLR, see Appendix A) to compare classification accuracy with and without voxel selection. Note that the number of initial parameters is four times the number of voxels (6000 parameters if 1500 voxels were used), because each of the four task conditions has its own linear discriminant function with a weight parameter for each voxel ($\Theta = (\boldsymbol{\theta}^{(\mathrm{UR})}, \boldsymbol{\theta}^{(\mathrm{LR})}, \boldsymbol{\theta}^{(\mathrm{LL})}, \boldsymbol{\theta}^{(\mathrm{UL})})$).

Orientation grating stimuli experiment

Data acquisition

We used data from Kamitani and Tong (2005), where a subject viewed one of the eight possible orientation stimuli while brain activity was monitored in retinotopic visual areas (V1–V4) using standard fMRI procedures. To investigate across-session generalization, we analyzed two experimental

data sets recorded about one month apart (Day 1 and Day 2) from the same subject. The data of Day 1 and Day 2 served as training and test data (24 blocks for each orientation in both sessions), respectively. See Kamitani and Tong (2005) for the details.

Data analysis

We performed binary classifications of all pairs of eight orientations (total 28 pairs) rather than eight-class classification. The results were then combined into four groups according to the orientation difference between the stimuli: 22.5 degrees (8 pairs), 45 degrees (8 pairs), 67.5 degrees (8 pairs) and 90 degrees (4 pairs), as decoding accuracy depends on the orientation difference.

Analyses were performed with voxels from V1–V4 (647 voxels in total) that were ranked by the *T*-value or by the SC-value using Day 1's data. *T*-values were calculated for each voxel based on the conventional *T*-statistics, which compared the responses to the two orientations to be classified. SC-values were calculated using the procedure described above (80% training, 20% test). The voxels with the *M* highest rank, either by the *T*-value or by the SC-value, were chosen as the elements of the feature vector for classification (*M* varied from 5 to 40). Then, the linear weights of the logistic regression model (without sparse estimation) were estimated using Day 1's data, and the classification performance was evaluated using Day 2's data. Before applying logistic regression, normalization was applied such that each chosen voxel has a mean 0 and variance 1. The above procedure was repeated for all of the 28 pairs. Average classification performance for *T*-value ranked voxels and SC-value ranked voxels were compared at four levels of orientation difference.

We then conducted the shuffle analysis on the *T*-value and SC-value ranked voxels to investigate the effect of voxel correlation on classification performance. The shuffle measure was calculated for each pair and voxel number, by repeating the shuffling of training data 300 times.

Results

Simulation study

We first tested the performance of SLR using a simulated data set in which we fixed the number of relevant features and varied the number of irrelevant features within the set. This data set, though much simplified, captures a potential problem that could occur with fMRI data: voxel patterns inside a small brain region show activity relevant to the given task, but the majority of voxels are irrelevant.

We compared test performance between SLR and linear regularized logistic regression (RLR), which uses the same logistic model as SLR but lacks the ARD procedure for reducing the number of dimensions. We also computed the test performance of support vector machines (SVM) for comparison. In Fig. 3(a), test performances of SLR, RLR and SVM are plotted as a function of the number of input features *D*. The performance of SLR was inferior and comparable to that of RLR and SVM, respectively, when the total number of features is small, such that most features are relevant (note that there are 10 relevant features). However, SLR begins to outperform RLR and SVM as the number of irrelevant features is increased. Although the performances of SLR, RLR and SVM drop off as the number of irrelevant features increases, the drop off is much slower for SLR, indicating that SLR is more robust to the presence of irrelevant features than the other two methods. In Fig. 3(b), the average number of features selected by SLR is plotted. The number of selected features was slightly larger than that of the relevant dimensions for a range of initial numbers of input dimensions (D=100–2000). Although several irrelevant features were selected in these cases, most of the irrelevant features were removed. As a result, the performance of SLR did not drop off so much as shown in Fig. 3 (a). In the case of D=10, where all features are relevant, the average number of selected features was fewer than 10. This 'overpruning' underlies the poor performance of SLR at D=10

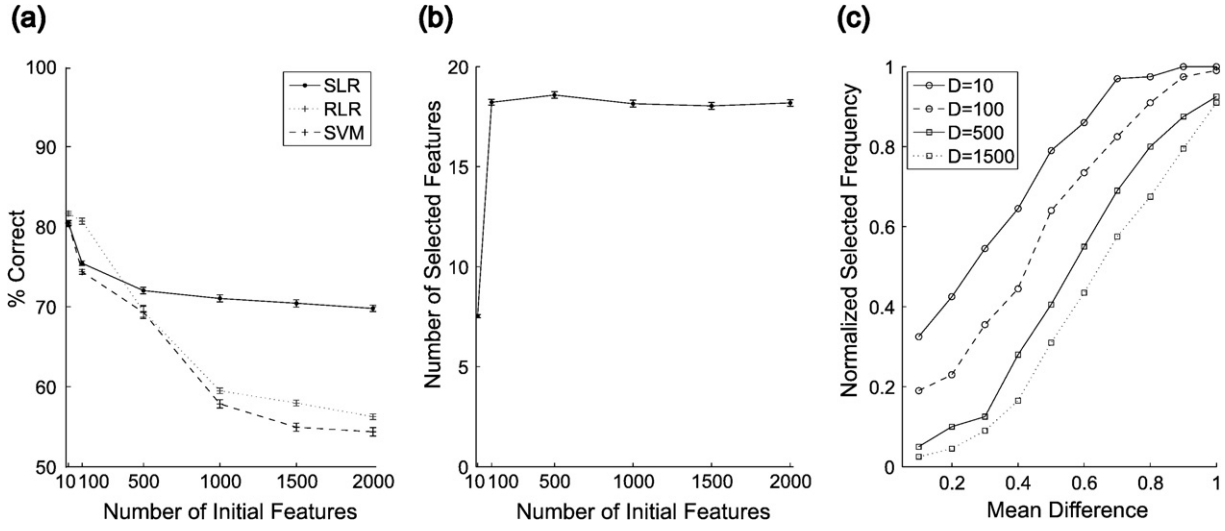

Fig. 3. Evaluation of SLR using simulation data. Data samples for binary classification were randomly generated from two Gaussian distributions. Only the first 10 dimensions were set to be informative with graded mean differences. Note that as the problem here is binary, the number of dimensions/features is identical to that of weight parameters. (a) Binary classification performance is plotted as a function of the number of initial input dimensions/features. Mean and standard errors computed from 200 Monte Carlo simulations are plotted. The solid, dotted and dashed lines indicate the results for SLR, regularized logistic regression (RLR) and support vector machine (SVM), respectively. RLR uses the same logistic regression model as SLR, but does not impose sparsity in estimating weight parameters. (b) The average number of selected dimensions by SLR is plotted against the number of initial dimensions. (c) The normalized frequency that each feature was selected by SLR in 200 repetitions of Monte Carlo simulation is plotted against the mean differences of the first 10 features. The lines indicate the results for different numbers of initial dimensions.

in Fig. 3(a). Fig. 3(c) shows the frequency that each of the relevant features was selected by SLR. We can observe that dimensions with higher relevance (large mean difference) tend to be selected more frequently, while the overall frequencies decrease with the number of initial features. These results demonstrate that although feature selection by SLR tends to overprune weakly relevant features, it automatically selects highly relevant features, removes most of the irrelevant features, and helps to improve classification performance in the presence of many irrelevant features.

Analysis of four quadrant data

Next, we applied SLR to experimental fMRI data obtained from a simple visual stimulation study. In each block of this experiment, a pie-shaped flickering checkerboard was presented in one of the four visual field quadrants (see Methods). Decoding analysis was performed to predict which visual quadrant received stimulation, using fMRI activity patterns from the visual cortex. This data set allowed us to test if the location of SLR-selected voxels is consistent with known functional anatomy of retinotopic visual cortex (Engel et al., 1994). As the classification problem here involves four classes, we used sparse multinomial logistic regression (SMLR), the multi-class version of SLR. Since each of the four task conditions has its own linear discriminant function, defined by a weight parameter for each voxel ($\Theta=(\boldsymbol{\theta}^{(UR)}, \boldsymbol{\theta}^{(LR)}, \boldsymbol{\theta}^{(LL)}, \boldsymbol{\theta}^{(UL)})$), SMLR can be used to identify relevant voxels for each of the four task conditions.

We evaluated the performance of SMLR and RMLR using a leave-one-run-out cross-validation procedure. Test performance and the number of parameters shown are averages over 6 cross-validation data sets. Table 1 summarizes classification accuracy and the number of features used with and without sparse estimation for four subjects. The number of parameters with sparse estimation is the total number of non-zero parameters in $\boldsymbol{\theta}^{(UR)}$, $\boldsymbol{\theta}^{(LR)}$, $\boldsymbol{\theta}^{(LL)}$, $\boldsymbol{\theta}^{(UL)}$. Note that the number of features used for RMLR is equal to the initial number of features for SMLR. Starting from approximately 6000 parameters, SMLR selected very few parameters (8.4±2.2 across subjects) yet still achieved high decoding accuracy (91.3±8.7% across subjects; chance level, 25%). Its performance was comparable to that of RMLR (89.9±9.2% across subjects) using all occipital voxels. This demonstrates that SMLR can select a small number of effective parameters without degrading classification performance. However, we did not find significant improvement in performance with sparse estimation by SMLR. This is presumably because the initial voxels had been pre-selected by the occipital mask, so that many of the voxels contained information useful for the classification task.

Table 1
Comparison of SMLR and RMLR in test performance and the number of parameters for the decoding of four quadrants

		Test performance (%)	Number of parameters
Subject 1	SMLR	97.2±6.8	5.7±0.5
	RMLR	95.8±4.6	6516
Subject 2	SMLR	87.5±7.0	9.2±1.2
	RMLR	87.5±11.5	6912
Subject 3	SMLR	94.4±4.3	8.5±1.0
	RMLR	86.1±10.9	5880
Subject 4	SMLR	86.1±11.4	10.3±2.2
	RMLR	90.3±8.2	6552

This table summarizes the results of leave-one-run-out cross validation for four subjects. The column of 'Test performance' shows the average correct percentages and the standard deviations calculated by SMLR and RMLR. The column of 'Number of parameters' shows the averaged numbers of parameters (and the standard deviations) selected by SMLR and those used by RMLR. Note that RMLR does not perform voxel/feature selection by itself. Thus, the number of parameters used by RMLR is four (i.e., the number of classes) times the number of initial input voxels.

We inspected the locations of voxels selected by SMLR. The images in the center of Fig. 4 show frequently selected voxels (identified more than 3 times in 6 cross-validation steps) with the occipital lobe mask overlayed on the T2-anatomical image for subject 1. Note that each of the four linear discriminant functions, which imply the presence of the stimulus in each quadrant, had its own set of selected voxels. The color indicates the corresponding quadrant for each selected voxel. In this subject, only one voxel was selected for three quadrants, and three for the remaining quadrant. The locations of these voxels nicely matched the known retinotopic organization of visual cortex: the voxels for the four quadrants were found in the corresponding region of the visual cortex (e.g., the voxel for the upper-left quadrant was found in the ventral bank of the right calcarine sulcus). The selected voxels also matched well with voxels that showed high *F*-values from the 1-way ANOVA analysis (data not shown). The average BOLD time courses (30 s from the stimulus onset) of the selected voxels for each stimulus location are also depicted in Fig. 4. Each voxel shows a very selective response to the stimulus presented in the corresponding quadrant. These results demonstrate that SMLR can automatically find voxels that are selectively activated by the individual task conditions.

It should be noted that we observed variability in the selected voxels, even though the training data sets used for the 6-fold cross-validation procedure involved considerable overlap. This indicates that the voxels selected by SLR are somewhat sensitive to the contents of the training data set. However, if we focus on the voxels consistently selected over multiple iterations, most of these were found near the calcarine sulcus in the primary visual cortex (by visual inspection of sagittal slices), and their relative positions matched well with the retinotopic organization of V1. These tendencies were found in most cases, except for conditions UR and UL of subject 3.

Analysis of orientation data

Finally, we applied SLR to fMRI data obtained while a subject viewed gratings of eight different orientations (Kamitani and Tong, 2005). Gratings of different orientations induce only subtle differences in activity in each voxel, unlike the stimuli presented in the four different quadrants. Thus, multiple voxels must be combined to achieve high levels of orientation-selective performance, which we call 'ensemble feature selectivity'. SLR could provide an effective means to find combinations of voxels for accurate decoding by removing irrelevant voxels.

For this analysis, we introduce the SC-value ranking method that sorts voxels according to the selection frequency by SLR weighted with the cross-validation accuracy. Then we compare it with the *T*-value ranking method based on the voxel-by-voxel univariate statistics that directly compare two conditions to be classified.

First we compared the difference between the two ranking methods, by plotting SC-values and *T*-values for voxels sorted

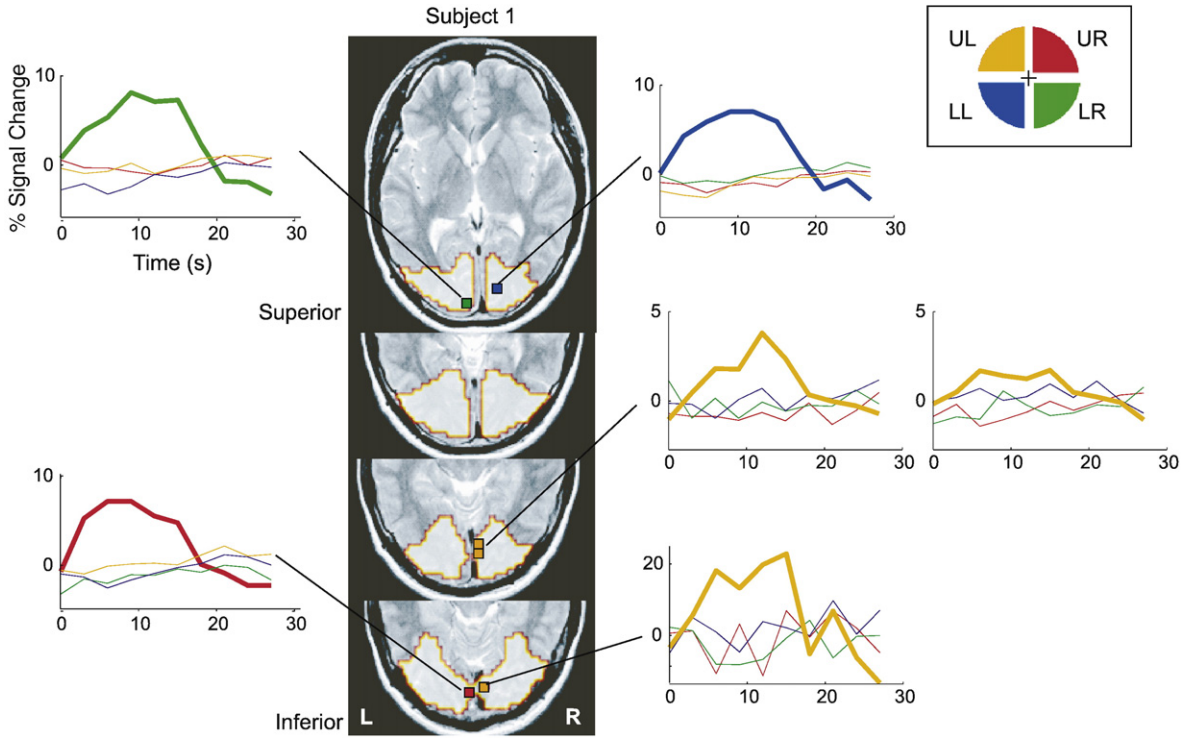

Fig. 4. Decoding of four quadrant stimuli. The locations of voxels selected by SLR are shown on the anatomical image. Filled squares indicate selected voxels for each of the four quadrants as in the legend. Note that in the multinomial logistic regression model, each class (quadrant) has its own weight parameters (see Fig. 1). The color indicates the class to which the selected weight parameter belongs. The lighter region shows the occipital mask, from which an initial set of voxels was identified. Only a few voxels were selected for this task (six voxels in total for this subject), and the selected voxels for each quadrant were found in the vertically and horizontally flipped locations, consistent with the visual field mapping in the early visual cortex. Trial-averaged BOLD time courses (percent signal change relative to the rest) are plotted for each of the selected voxels. Time 0 corresponds to the stimulus onset. The color here indicates the stimulus condition (one of the four quadrants) as in the legend.

by the SC ranking. Fig. 5 shows an example from the binary classification of 0 vs. 135 degrees. Although voxels with high SC-values tend to have high *T*-values, there is a substantial disagreement between them. Similar trends were observed in some other pairs. Thus, voxels selected by SLR are not consistent with those selected by univariate functional mapping.

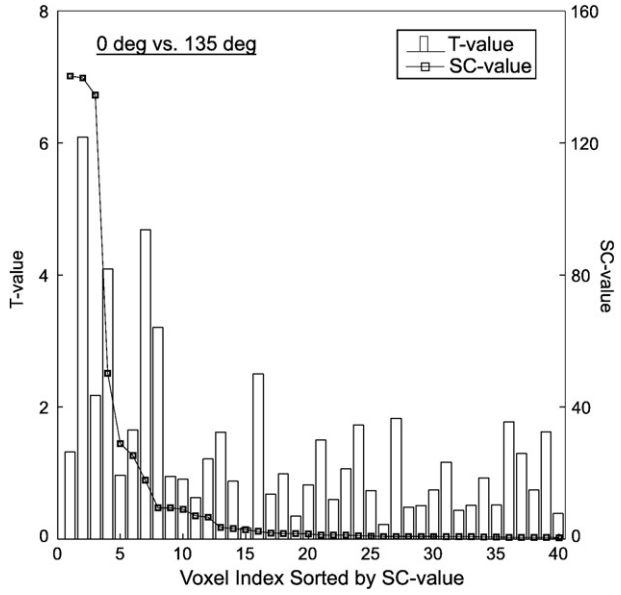

Fig. 5. Difference between SC-values and *T*-values. SC-values (solid line) and *T*-values (bars) are plotted for voxels sorted by the SC-values. These values were obtained for the classification of 0 vs. 135 degrees of orientation.

Second, we investigated if SC-ranked voxels lead to better performance than *T*-value ranked voxels. Fig. 6 shows test performance for the SC-ranked voxels and the *T*-value ranked voxels. Percent correct classification is plotted against the number of voxels selected from the top of the ranks. The results of 28 binary classifications are grouped according to the orientation differences (22.5, 45, 67.5 and 90 degrees). SC-ranked voxels generally outperformed *T*-value ranked voxels. The performance differences using the top-ranked 40 voxels are 6.5%, 6.3%, 10.4% and 4.7% for orientation differences of 22.5, 45, 67.5 and 90 degrees, respectively. Significant differences in the overall performance profiles were observed for all four orientation comparisons (two-way ANOVA, repeated measurement, [ranking method]×[voxel number], significant [ranking method] effect $P<0.05$, no significant [voxel number] effect except 90 degree difference group, and no significant interaction).

It may seem puzzling that voxels with low *T*-values, which do not produce distinctive responses to different task conditions, can lead to higher classification accuracy. However, it is known that non-distinctive features, which in this case have low *T*-values, can make the multivariate patterns of two (or more) classes more discriminable if they are correlated with distinctive features (e.g., Averveck et al., 2006). Fig. 7(a) shows such an example, where the values of the first two voxels in Fig. 5 are displayed in a scattered plot. The red diamond and blue cross denote samples labeled as 0 degrees and 135 degrees, respectively. The gray line is the linear boundary estimated by logistic regression using the

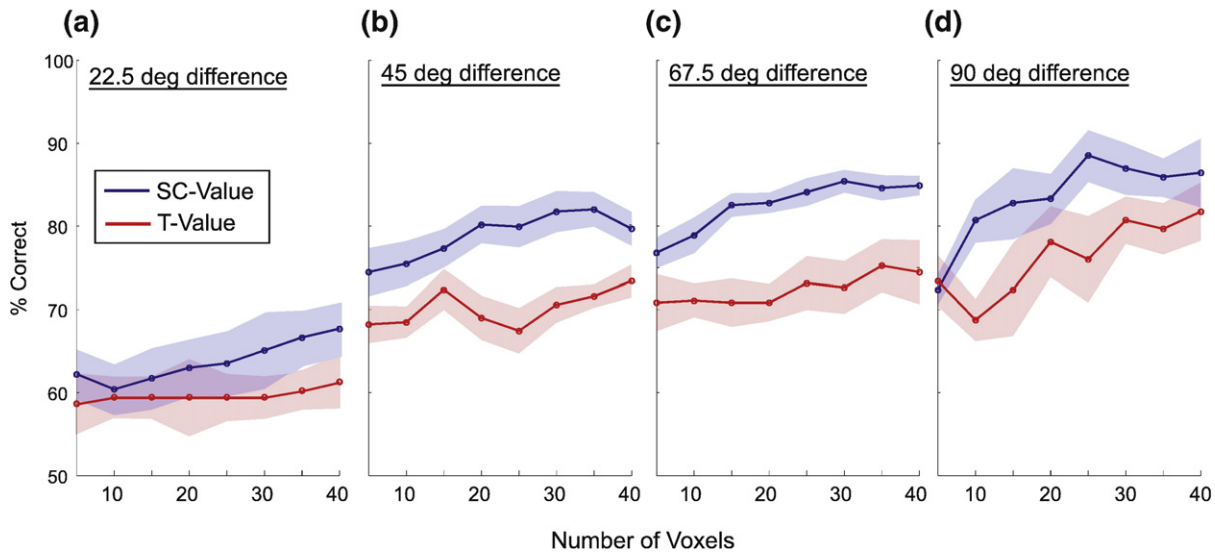

Fig. 6. Comparison of classification performance between the SC-value and the *T*-value rankings. The test performance for the classification of two orientations, chosen from eight orientations (0, 22.5, 45, … degrees), is plotted against the number of voxels. Voxels were sorted either by the SC-values or by the *T*-values, and those with highest ranks were used. The results of all orientation pairs were grouped by the orientation differences. Panels (a–d) summarize the results of 22.5 degree (8 pairs of orientations), 45 degree (8 pairs), 67.5 degree (8 pairs), and 90 degree (4 pairs) differences, respectively. Voxel ranking was computed for each pair of orientations. The blue and red lines indicate test performance for the SC-value ranking and the *T*-value ranking, respectively. The shaded areas represent the standard errors.

first five voxels (corresponding to the left-most point of *x* axis in Fig. 6(b)) as the feature vectors. Thus, the boundary is a projection from the five dimensional feature space. Histograms of the values of the first and the second voxel are shown along the horizontal and the vertical axes, respectively. These histograms indicate that the first dimension poorly discriminates between the two classes, when compared to the second dimension. However, it can also be seen that the presence of the first dimension makes the two dimensional patterns more discriminable. This occurs because the two voxels are negatively correlated in terms of their mean response to the two classes ('signal correlation') while they are positively correlated for the samples within each class ('noise correlation') (Averveck et al., 2006). Thus, SLR seems to be able to exploit noise correlation for achieving high decoding accuracy.

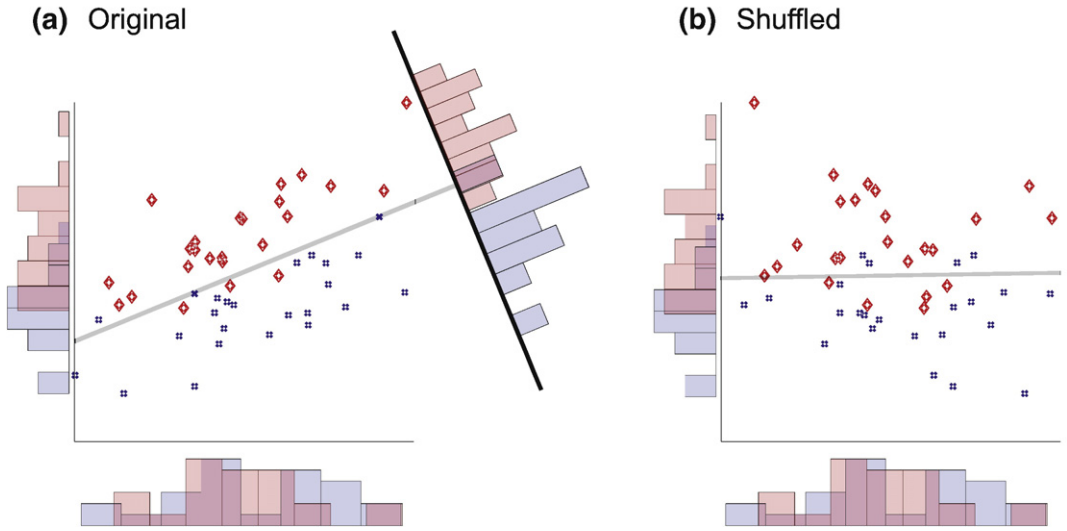

Fig. 7. Contribution of voxel correlation to classification. (a) The values of the top two voxels in the SC-value ranking (Fig. 5) are shown in a scatter plot and histograms. The red diamonds and the blue crosses represent 0 degree and 135 degree samples in the training data set, respectively. The gray line is the discriminant boundary estimated by logistic regression. Histograms show the distributions of the samples along the axes of the first and the second voxels, and along the axis orthogonal to the discriminant boundary. The first voxel (*x* axis) is poorly discriminative (as indicated by the low *T*-value in Fig. 5), while the second voxel (*y* axis) is more discriminative. When these voxels are combined (the axis orthogonal to the discriminant boundary), the distributions of two classes become even more discriminative. Note that the discriminant boundary provides better discrimination than the second voxel alone. The first voxel could contribute to the discrimination via its correlation with the second voxel, even though it has a low *T*-value and is poorly discriminative itself. (b) The values in the original data (a) were shuffled within each voxel and class so that the correlation between voxels was removed from the distribution of each class. The histograms of two individual voxels are identical to those of the original data (a). But the discriminant boundary is different: the discrimination is almost solely dependent on the second voxel.

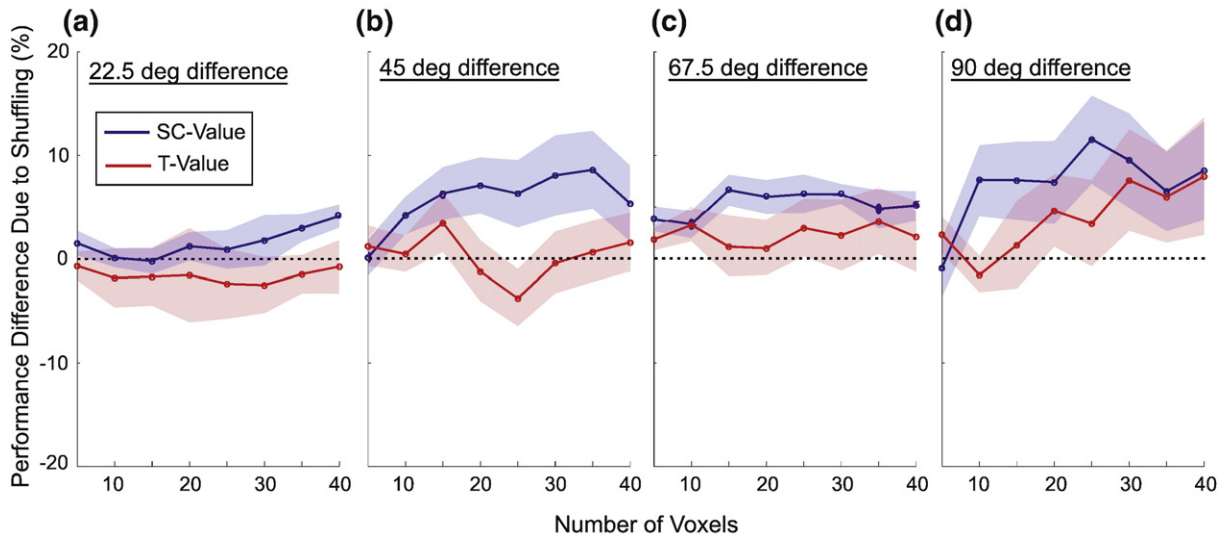

Fig. 8. Effect of shuffling on the performance of SC-ranked voxels and *T*-ranked voxels. The same analysis as in Fig. 6 was performed with shuffled training data. The difference in test performance between the original and the shuffled training data was calculated (shuffle measure). The average shuffle measure (over 300 times shufflings) is plotted as a function of the number of voxels, for SC-ranked voxels (blue) and *T*-ranked voxels (red) and for four orientation differences.

Finally, the effect of voxel correlations on test classification performance was evaluated by using a shuffle procedure that removes correlations by randomly permuting the order of samples within each class and each dimension (see Methods for details). Fig. 7(b) shows the distribution of the samples after the shuffling of the samples in Fig. 7(a). It can be seen that shuffling removes the correlation between the first and the second voxels while the unidimensional histograms are unaffected. Next, we calculated the difference in test performance between the original data and the shuffled data, in which deviations above zero indicate improved classification due to correlations between voxels. The shuffle procedure was applied to binary classifications using all the voxel numbers in Fig. 6 of all 28 pairs of eight orientations, and results are plotted by the four orientation differences (Fig. 8). Shuffle measures for the top 40 SC-ranked voxels, respectively, reached 4.1%, 5.3%, 5.2% and 8.5% for the groups of 22.5, 45, 67.5 and 90 degree difference, while those for *T*-value ranked voxels were much smaller (−0.7%, 1.6%, 2.1% and 8.0.%). Furthermore, if we examine the difference in test performance between the curves resulting from SC-value and *T*-value ranking in Fig. 6 and those in Fig. 8, the shapes look very similar. The correlation value between the difference curves in Figs. 6 and 8 was 0.73 on average across the 28 pairs. This suggests that the difference in test performance between SC-value ranking and univariate *T*-value ranking is partially explained by the benefit of selecting voxels with noise correlation when using the SC-value ranking method. It should be noted that the shuffle measure (or any measure that evaluates higher moment) may not work reliably in a high-dimensional feature space because samples are distributed only sparsely (well-known as 'the curse of dimension'). Therefore, some caution should be taken to interpret the results of shuffling when the number of voxels is large.

Discussion

We have introduced a novel linear classifier for fMRI decoding, sparse logistic regression (SLR). SLR is a binary/multi-class classifier with the ability to automatically select voxels relevant for a given classification problem. Using a set of simulation data and two sets of real experimental data, we have demonstrated the following: (1) SLR can automatically select relevant features, and thereby prevent overfitting to some extent; (2) the locations of SLR-selected voxels are consistent with known functional anatomy; (3) SLR-selected voxels were different from those selected by the conventional voxel-wise univariate statistics, and the former outperformed the latter in classification; (4) this difference in classification performance can be accounted for in part by the correlation structure among the selected voxels.

The simulation study demonstrated that SLR can outperform other classification methods for data sets with a large number of irrelevant features, by automatically removing them. The performance of other classifiers, such as regularized logistic regression (RLR) and support vector machine (SVM), was degraded remarkably with increase of the number of irrelevant features. As shown in the case where all the features are relevant ($D=10$), SLR did not always select all the relevant features, but captured many of the highly relevant features. Thus, the effect of omitted relevant voxels on classification performance is expected to be small. Even highly relevant features were selected less frequently with more irrelevant features. As a result, the performance of SLR dropped gradually with the number of irrelevant features, but the slope was less steep than those of the other two methods.

Results from the quadrant visual stimulation experiment suggested the possibility of interpreting sparsely estimated parameters from a physiological point of view. In decoding analysis, classification performance is often used as an index for the functional selectivity of an area, or a set of voxels. Here, we were able to identify relevant brain regions by the non-zero weight parameters selected by SLR. In SMLR (sparse multinomial logistic regression, the multinomial version of SLR), each class has its own set of parameters. Thus, the distribution of selected voxels for each class provides a class-specific cortical map. It should be noted, however, that since this mapping indicates voxels that most efficiently classify the data from different experimental conditions, they are not

the complete set of voxels that may be involved in a given experimental condition. This method can be regarded as a thresholded version of the SVM-based mapping method proposed by LaConte et al. (2005), in which weight parameters estimated by linear SVM are used for mapping. Our method may be more robust, as it takes the variability of estimation into consideration (see the alpha-step of the SLR algorithm in Appendix A). For further improvement of mapping, it may be preferable to map voxels common to several training data sets using a cross-validation technique, which is the basis of the SLR-based voxel selection method (or the selection count (SC) ranking method).

The analysis of the orientation data showed that SC-value ranked voxels were different from *T*-value ranked voxels, and that SC-ranked voxels outperformed *T*-value ranked voxels in the classification of orientation. Furthermore, we found that this difference in performance can be explained in part by the correlation structure among voxels. Although the classification performance shown in Fig. 6 was calculated using logistic regression, qualitatively similar results were obtained when linear SVM, linear RLR, or Gaussian mixture classifier (MATLAB, classify.m) was used. There are a few remarks to be made about this analysis. First, we only considered up to 40 voxels when comparing the performance of SC-value ranked voxels and that of *T*-value ranked voxels. This is because SC-values of rank below 40 became almost zeros, thus it was impossible to rank those voxels reliably by SC-values. Second, the shuffle measure indicates the benefit of voxel correlation under the assumption that the distributions of training and test data sets are stationary. Values of this shuffle measure can be affected by non-stationarity between the distributions of Day 1 and Day 2. However, as the SC- and *T*-value rankings are both calculated from the same training data set, they should not have specific biases in terms of (non-)stationarity. Thus, even in the presence of non-stationarity, the difference in shuffle measure between the SC- and *T*-value rankings should reflect the difference in the benefit of voxel correlation. Third, we observed significantly non-zero shuffle measures even for *T*-value ranked voxels. Although voxel selection based on *T*-value ranking ignores correlations between voxels, voxels selected for their high *T*-value could nonetheless be correlated with each other. Fourth, the comparison between *T*-value ranked voxels and SC-value ranked voxels actually confounds two factors: the difference in the algorithm (univariate *T*-value vs. multivariate SLR) and the difference in the data sampling method. In order to control the latter factor, we have computed a bootstrap distribution of *T*-value using a resampling procedure analogous to that used for the SC-value ranking. Then, voxels were ranked by the average or the normalized average (divided by the standard deviation) of the distribution. In both cases, the test percent correct did not change from that with the original *T*-value ranking, in which each *T*-value was calculated only once using the whole training data, thus indicating that the difference in the algorithm was the main factor for the comparison.

SLR is a convenient classifier in several respects. It can work even when the number of training data samples is less than the number of voxels (features). It can minimize overfitting by automatically removing irrelevant voxels, and can be used for voxel selection. It does not require adjusting parameters manually. Thus, the application of SLR could prove as useful as SVM, which has been gaining popularity in several recent fMRI studies (Cox and Savoy, 2003; Mitchell et al., 2004; LaConte et al., 2005; Kamitani and Tong, 2005, 2006; Haynes et al., 2007 and so on). It is interesting to compare the characteristics of SLR and SVM. Both SLR and linear SVM are defined by a linear discriminant function, and involve sparse estimation of parameters. However, the parameters in SLR are associated with features, while the parameters in SVM are associated with samples. The linear discriminant function of SLR is $f(\mathbf{x};\boldsymbol{\theta}) = \boldsymbol{\theta}^{\mathrm{t}}\mathbf{x} = \sum_{d=1}^{D} \theta_d x_d$ while that of SVM is the kernel representation, $g(\mathbf{x};\boldsymbol{\theta}) = \sum_{i=1}^{N} (\mathbf{x}_i^{\mathrm{t}}\mathbf{x})\theta_i = \left(\sum_{i=1}^{N} \theta_i \mathbf{x}_i^{\mathrm{t}}\right)\mathbf{x}$. Here, d and i are the indices for features and samples, respectively, and D and N are the total numbers of features and samples, respectively. When the parameter vector θ is sparsely estimated in the former representation, only features associated with non-zero θ_i s decide the discriminant function. Thus, the discriminant function for SLR lies in the space of lower dimension than the original dimension D. On the other hand, when the parameter vector θ is sparsely estimated in the latter representation, only samples associated with non-zero θ_i s (called support vectors) decide the discriminant function. Thus, the discriminant function for SVM lies in the space of the original dimension D. Therefore, only SLR is equipped with the feature selection property. Because of this capability of feature selection, SLR seems to work better than SVM if a feature vector consists of many irrelevant features, as indicated in our simulation results.

To obtain the optimal voxel subset from an initially large voxel set, an exhaustive combinatorial search is generally required. This search, however, becomes intractable with even a modest number of voxels. An alternative approach is to handle each voxel independently, as the *T*-value ranked voxel selection procedure does. But this method neglects the dependency among voxels. A good compromise may be the searchlight method suggested by Kriegeskorte et al. (2006), where the correlation structure among a local set of voxels (voxels within a sphere of 4 mm radius) in a 'searchlight' is utilized. This method, however, does not take into account potential correlations between spatially remote voxels. In contrast, SLR-based voxel selection can exploit the correlation structure among all the voxels in the initial voxel set, without requiring an exhaustive combinatorial search, although the result may be suboptimal. For regression problems, Jo-Anne Ting et al. (2008 to appear) compared matching between features selected by the ARD method and those selected by the brute-force method. They showed that most of features (neurons) selected by the ARD methods matched with those selected by the brute-force method (over 90%). Their result might also support the validity of feature selection by SLR.

A drawback of SLR is the computational cost (time and memory). As mentioned in Appendix A, the estimation algorithm is an iterative algorithm with the Hessian matrix inversion of size $D \times D$, where D is the number of feature dimensions or voxels. In the case of 10,000 dimensions, it is intractable to keep a matrix of size 10,000 × 10,000 in memory. Even if a matrix can be kept in memory, a matrix inversion of size $D \times D$ also requires computation time proportional to D^3. In our example of the four quadrant experiment, it takes about 1 h to analyze the data of one subject performing the whole leave-one-run-out procedure with a Linux machine with a

2.66 GHz CPU and a 4 GB memory. One approach to overcome this computational problem is direct approximation of the logistic function using a variational parameter (Jaakkola and Jordan, 2000; Bishop and Tipping, 2000). This approach does not require the computation of the Hessian matrix explicitly. Recently, we have implemented this estimation algorithm. Although this approximation is only valid for binary classification, we can now conduct a whole-brain classification analysis. Another approach is the component-wise sequential update procedure. Two different algorithms based on this approach have been proposed for the logistic regression model with the ARD model (Tipping and Faul, 2003) and for the multinomial logistic regression with the Laplace prior (Krishnapuram et al., 2005). These algorithms require less memory and less computation than our method. In particular, the latter algorithm only requires memory and computation time proportional to D, which are much smaller than those required by our algorithm. Comparison of computational time, classification performance and survived voxels between our algorithm and these sequential algorithms could be an interesting future work.

Our current method is limited to linear classification. It is possible to extend the framework to nonlinear discriminant functions by combining a kernel function with the automatic relevance determination (ARD) technique. But this approach could cost much more computational resources and suffer from the problem of local minima more severely.

Although we focused on 'voxel' selection for fMRI decoding, SLR can be applied to the pattern classification of other neuroimaging signals such as EEG, MEG and NIRS. In particular, its application to brain–computer interface (BCI) (Wolpaw et al., 2002), is of great interest. SLR could be used to determine relevant channels (Lal et al., 2004) or frequency bands in advance, and thus may provide a tool to customize the input features to BCI for individual subjects.

Acknowledgments

The authors would like to thank Dr. M. Kawato from ATR computational neuroscience laboratories for his helpful suggestions. This research was supported in part by the NICT, Honda Research Institute, the SCOPE, SOUMU, the Nissan Science Foundation, and the National Eye Institute (R01 EY017082 and R01 EY14202).

Appendix A

The parameter estimation algorithm of sparse multinomial logistic regression (SMLR) and its rough derivation are presented. The algorithm is identical to that of the relevance vector machine except that we treat the multi-class problem with the full-Bayesian approach rather than the binary problem with the marginal likelihood approach. For the relevance vector machine, see Tipping (2001).

Let the input feature vector in D dimensional space denoted by $\mathbf{x}\in\mathbb{R}^{\mathbb{D}}$ and the output label by $\mathbf{y}=[y^{(1)},\ldots,y^{(C)}]$ such that $y^{(c)}=1$ if $\mathbf{x}$ belongs to class c and $y^{(c)}=0$ otherwise ("1-of-m encoding"). Given N training data $\{(\mathbf{x}_1,\mathbf{y}_1),\ldots,(\mathbf{x}_N,\mathbf{y}_N)\}$, the likelihood and prior distributions of SMLR is, respectively, expressed by the following multinomial distribution,

$$P(\mathbf{Y}|\mathbf{X};\Theta)=\prod_{n=1}^{N}\prod_{c=1}^{C}p_n^{(c)y_n^{(c)}} \tag{a1}$$

where

$$p_n^{(c)}=\frac{\exp\left(\mathbf{x}_n^{\mathrm{t}}\boldsymbol{\theta}^{(c)}\right)}{\sum_{k=1}^{C}\exp\left(\mathbf{x}_n^{\mathrm{t}}\boldsymbol{\theta}^{(k)}\right)} \tag{a2}$$

and $\mathbf{x}_n^{\mathrm{t}}$ denotes a transpose of a column vector $\mathbf{x}_n$. In addition the hierarchical automatic relevance determination (ARD) priors (MacKay, 1992; Neal, 1996) are assumed,

$$P\left(\theta_d^{(c)}|\alpha_d^{(c)}\right)=N\left(\theta_d^{(c)};0,\alpha_d^{(c)^{-1}}\right)\quad d=1,\ldots,D,\ c=1,\ldots,C \tag{a3}$$

$$P_0\left(\alpha_d^{(c)}\right)=\Gamma\left(\alpha_d^{(c)};\gamma_{d0}^{(c)},\overline{\alpha}_{d0}^{(c)}\right)\quad d=1,\ldots,D,\ c=1,\ldots,C \tag{a4}$$

where $N(\mathbf{x};\boldsymbol{\mu},S)$ denotes the Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance S, and $\Gamma(\alpha;\gamma_0,\overline{\alpha}_0)\propto\alpha^{\gamma_0-1}\exp\left(-\frac{\gamma_0}{\overline{\alpha}_0}\alpha\right)$ denotes the gamma distribution with mean $E(\alpha)=\bar{\alpha}_0$ and the degree of freedom γ_0. Note that we use a bar for the mean parameter of the Gamma distribution in order to discriminate a variable of the distribution and the expectation parameter of the distribution. Eq. (a1) is the likelihood function in which each term of the product is given by the multinomial distribution with probabilities $p_n^{(1)},\ldots,p_n^{(c)}$ calculated from the linear discriminant functions $\mathbf{x}_n^{\mathrm{t}}\boldsymbol{\theta}^{(1)},\ldots,\mathbf{x}_n^{\mathrm{t}}\boldsymbol{\theta}^{(c)}$ by Eq. (a2). Eq. (a3) is a prior distribution of weight parameters $\boldsymbol{\theta}^{(c)}=[\theta_1^{(c)},\ldots,\theta_D^{(c)}]^{\mathrm{t}}$ and Eq. (a4) is a hyper-prior distribution of hyper-parameters $\alpha^{(c)}=[\alpha_1^{(c)},\ldots,\alpha_D^{(c)}]^{\mathrm{t}}$. A vector $\Theta=[\boldsymbol{\theta}^{(1)\mathrm{t}},\ldots,\boldsymbol{\theta}^{(c)\mathrm{t}},\ldots\boldsymbol{\theta}^{(C)\mathrm{t}}]^{\mathrm{t}}$ is a collection of weight parameter vectors. The hyper-parameter $\alpha_d^{(c)}$ is referred to as the relevance parameter, and it controls the importance of the corresponding weight parameter by adjusting the variance of the normal distribution in Eq. (a4). The parameters $\overline{\alpha}_{d0}^{(c)}$ and $\gamma_{d0}^{(c)}$ in Eq. (a4) determine the expectation and the a-priori confidence of each relevance parameter, respectively. If we have prior knowledge on the importance of each parameter, it can be used to set the value of the confidence $\gamma_{d0}^{(c)}$. However, it is rare that such knowledge is available a-priori, and thus the non-informative prior (obtained by substituting $\gamma_{d0}^{(c)}=0$ into Eq. (a4)),

$$P_0\left(\alpha_d^{(c)}\right)=\alpha_d^{(c)^{-1}}\quad d=1,\ldots,D,\ c=1,\ldots,C \tag{a5}$$

is often used. We used this non-informative prior in all the analyses in this paper.

The estimation of weight and relevance parameters can be done by calculating the following posterior distributions,

$$P(\Theta|\mathbf{Y},\mathbf{X})=\int P(\Theta,\mathbf{A}|\mathbf{Y},\mathbf{X})d\mathbf{A} \tag{a6}$$

$$P(\mathbf{A}|\mathbf{Y},\mathbf{X})=\int P(\Theta,\mathbf{A}|\mathbf{Y},\mathbf{X})d\Theta. \tag{a7}$$

Because it is difficult to analytically integrate the right hand sides of Eqs. (a6) and (a7), the calculation requires either computational and stochastic approximations such as the Markov Chain Monte Carlo (MCMC) method or analytical and

deterministic approximations such as variational Bayesian (VB) method (Attias, 1999; Sato, 2001). The algorithm derived here is based on the VB method, since the MCMC method cannot be applied to high-dimensional problems because of its computational cost.

The VB method assumes the following conditional independence condition for the joint posterior distribution,

$$p(\Theta,\mathbf{A}|\mathbf{Y},\mathbf{X}) = Q(\Theta)Q(\mathbf{A}) \tag{a8}$$

where $Q(\Theta)$and $Q(\mathbf{A})$ denote the marginal posterior distributions given $\mathbf{Y}$ and $\mathbf{X}$ (for simplicity, $\mathbf{Y}$ and $\mathbf{X}$ are omitted from the expressions). Under the assumption (Eq. (a8)), the calculation of the posterior distributions can be reformulated by maximization of the variational free energy (see Attias, 1999 for details)

$$F(Q(\Theta),Q(\mathbf{A})) = \int Q(\Theta)Q(\mathbf{A})\log\frac{p(\mathbf{Y},\Theta,\mathbf{A}|\mathbf{X})}{Q(\Theta)Q(\mathbf{A})}d\Theta d\mathbf{A}.$$

This maximization is done by the iterative algorithm consisting of Eqs. (a9) and (a10),

$$[\theta\,\text{step}]\quad \log Q(\Theta) = \langle\log p(\mathbf{Y},\Theta,\mathbf{A}|\mathbf{X})\rangle_{Q(\mathbf{A})} + \text{const}, \tag{a9}$$

$$[\alpha\,\text{step}]\quad \log Q(\mathbf{A}) = \langle\log p(\mathbf{Y},\Theta,\mathbf{A}|\mathbf{X})\rangle_{Q(\Theta)} + \text{const}, \tag{a10}$$

where $\langle x\rangle_{Q(x)}$ is the expectation of x w.r.t the probability distribution $Q(x)$. By substituting the SMLR model Eqs. (a1)–(a4) into Eqs. (a9) and (a10), [θ step] and [α step] are, respectively, written as

$$\log Q(\Theta) = \sum_{n=1}^{N}\left[\sum_{c=1}^{C} y_n^{(c)}\mathbf{x}_n^{\mathrm{t}}\boldsymbol{\theta}^{(c)} - \log\left\{\sum_{c=1}^{C}\exp\left(\mathbf{x}_n^{\mathrm{t}}\boldsymbol{\theta}^{(c)}\right)\right\}\right] - \frac{1}{2}\sum_{c=1}^{C}\boldsymbol{\theta}^{(c)t}\langle A^{(c)}\rangle_{Q(A)}\boldsymbol{\theta}^{(c)} + \text{const} \tag{a11}$$

$$\log Q(\mathbf{A}) = \sum_{c=1}^{C}\sum_{d=1}^{D}\left[-\frac{1}{2}\langle\theta_2^{(c)2}\rangle_{Q(\Theta)}\alpha_d^{(c)} - \frac{1}{2}\log\alpha_d^{(c)}\right] + \text{const}. \tag{a12}$$

In [θ step], we further apply the Laplace approximation, a quadratic approximation around the maximum $\overline{\Theta}$, to the right hand side of Eq. (a11). Then [θ step] is rewritten;

$$\log Q(\Theta) \approx -\frac{1}{2}\left(\Theta-\overline{\Theta}\right)^{\mathrm{t}} H\left(\Theta-\overline{\Theta}\right) + \text{const} \tag{a13}$$

where H denotes the negative Hessian matrix at the maximum. The values $\overline{\theta}$ and H can be obtained by the Newton method using the gradient and the Hessian matrix respectively given by

$$\frac{\partial E}{\partial\Theta} = \left[\frac{\partial E}{\partial\boldsymbol{\theta}^{(1)t}},\ldots,\frac{\partial E}{\partial\boldsymbol{\theta}^{(c)t}}\right]^{\mathrm{t}}$$

$$\frac{\partial E}{\partial\boldsymbol{\theta}^{(c)}} = \sum_{n=1}^{N}\left\{y_n^{(c)} - p_n^{(c)}\right\}\mathbf{x}_i - \overline{A}^{(c)}\boldsymbol{\theta}^{(c)} \quad c = 1,\ldots,C,$$

and

$$\frac{\partial^2 E}{\partial\Theta\partial\Theta^{\mathrm{t}}} = -\sum_{n=1}^{N}\left(\begin{bmatrix} p_n^{(1)} & & \cdots & 0\\ & p_n^{(2)} & & \\ & & O & \\ 0 & \cdots & & p_n^{(C)}\end{bmatrix} - \begin{bmatrix} p_n^{(1)}p_n^{(1)} & p_n^{(1)}p_n^{(2)} & & \\ p_n^{(2)}p_n^{(1)} & p_n^{(2)}p_n^{(2)} & & \vdots \\ & & \ddots & \\ & \cdots & & p_n^{(C)}p_n^{(C)}\end{bmatrix}\right)\otimes\mathbf{x}_n\mathbf{x}_n^{t}.$$

Here we define $p_n^{(i)} = \exp\left(\mathbf{x}_n^{\mathrm{t}}\boldsymbol{\theta}^{(\mathrm{i})}\right)/\sum_{c=1}^{C}\exp\left(\mathbf{x}_n^{\mathrm{t}}\boldsymbol{\theta}^{(c)}\right)$,

$$E(\Theta) \equiv \sum_{n=1}^{N}\left[\sum_{c=1}^{C} y_n^{(c)}\boldsymbol{\theta}^{(c)t}\mathbf{x}_n - \log\left(\sum_{c=1}^{C}\exp\left(\boldsymbol{\theta}^{(c)}\mathbf{x}_n\right)\right)\right] - \frac{1}{2}\sum_{c=1}^{C}\boldsymbol{\theta}^{(c)}\overline{A}^{(c)}\boldsymbol{\theta}^{(c)}$$

and $\overline{A}^{(c)} = \mathrm{diag}\left(\langle\alpha^{(c)}\rangle_{Q(A)}\right)$. $\otimes$ denotes the Kronecker product. The maximum $\overline{\Theta}$ is the estimate of the weight vector, which is the approximation of the posterior mean of $P(\Theta|\mathbf{Y},\mathbf{X})$. The matrix H is the negative value of $\frac{\partial^2 E}{\partial\Theta\partial\Theta^{\mathrm{t}}}$ at the maximum $\overline{\Theta}$. From the functional form of Eq. (a13), $Q(\Theta)$ is the Gaussian distribution $N(\Theta;\overline{\Theta},S)$, where $S=H^{-1}$.

In [α step], given $Q(\Theta)\sim N(\Theta;\overline{\Theta},S)$, integrating the first term in Eq. (a12) with respect to $Q(\Theta)$ leads to

$$\log Q(\mathbf{A}) = \sum_{c=1}^{C}\sum_{d=1}^{D}\left[-\frac{1}{2}\left(\overline{\theta}_d^{(c)2} + S_{dd}^{(c,c)}\right)\alpha_d^{(c)} - \frac{1}{2}\log\alpha_d^{(c)}\right] + \text{const}. \tag{a14}$$

where $\overline{\theta}_d^{(c)}$ and $S_{dd}^{(c,c)}$ are the posterior mean and the posterior variance of $\theta_d^{(c)}$, respectively (elements of $\overline{\Theta}$ and S corresponding to $\theta_d^{(c)}$). From the functional form of Eq. (a14), $Q(\mathbf{A})$ is the product of the Gamma distributions $Q(\alpha_d^{(c)})\sim\Gamma(\alpha_d^{(c)};\gamma_d^{(c)},\overline{\alpha}_d^{(c)})$, of which degree of freedom and the mean parameter are respectively given by

$$\gamma_d^{(c)} = \frac{1}{2}\quad d = 1,\ldots,D,\ c = 1,\ldots,C$$

$$\overline{\alpha}_d^{(c)} = \frac{1}{\overline{\theta}_d^{(c)2} + S_{dd}^{(c,c)}}\quad d = 1,\ldots,D,\ c = 1,\ldots C.$$

To accelerate convergence, the following modified update rule motivated by the notion of the effective degree of freedom (MacKay, 1992) is employed,

$$\overline{\alpha}_d^{(c)} = \frac{1-\overline{\alpha}_d^{(c)}S_{dd}^{(c,c)}}{\left(\overline{\theta}_d^{(c)}\right)^2}\quad d = 1,\ldots,D,\ c = 1,\ldots,C.$$

The updated mean parameters $\overline{\alpha}_d^{(c)}$ are used in the next [θ step].

The algorithm is summarized as follows:

1. [Initialization] Set the initial values for $\overline{\alpha}_d^{(c)}$

$$\overline{\alpha}_d^{(c)} = 1 \quad d = 1, \ldots D,\ c = 1, \ldots, C.$$

2. [θ step] Update $Q(\Theta)$, given $Q(\mathbf{A}) \sim \prod_{c,d} \Gamma\left(\alpha_d^{(c)}; \gamma_d^{(c)}, \overline{\alpha}_d^{(c)}\right)$.
$Q(\Theta)$ is the Gaussian distribution,
$Q(\Theta) \sim N(\Theta; \overline{\Theta}, S)$,
where its mean $\overline{\Theta}$ is given by the maximum of the function,

$$E(\Theta) \equiv \sum_{n=1}^{N} \left[\sum_{c=1}^{C} y_n^{(c)} \boldsymbol{\theta}^{(c)\mathrm{t}} \mathbf{x}_n - \log\left(\sum_{c=1}^{C} \exp\left(\boldsymbol{\theta}^{(c)\mathrm{t}} \mathbf{x}_n \right) \right) \right] - \frac{1}{2} \sum_{c=1}^{C} \boldsymbol{\theta}^{(c)\mathrm{t}} \overline{A}^{(c)} \boldsymbol{\theta}^{(c)}.$$

The maximization is done by the Newton method using the gradient vector and the Hessian matrix,

$$\frac{\partial E}{\partial \boldsymbol{\theta}^{(c)}} = \sum_{n=1}^{N} \left\{ y_n^{(c)} - p_n^{(c)} \right\} \mathbf{x}_i - \overline{A}^{(c)} \boldsymbol{\theta}^{(c)} \quad c = 1, \ldots, C$$

$$\frac{\partial^2 E}{\partial \Theta \partial \Theta^{\mathrm{t}}} = -\sum_{n=1}^{N} \left(\begin{bmatrix} p_n^{(1)} & & \cdots & 0 \\ & p_n^{(2)} & & \\ & & O & \\ 0 & \cdots & & p_n^{(C)} \end{bmatrix} - \begin{bmatrix} p_n^{(1)} p_n^{(1)} & p_n^{(1)} p_n^{(2)} & & \vdots \\ p_n^{(2)} p_n^{(1)} & p_n^{(2)} p_n^{(2)} & & \\ & & \ddots & \\ \cdots & & & p_n^{(C)} p_n^{(C)} \end{bmatrix} \right) \otimes \mathbf{x}_n \mathbf{x}_n^{\mathrm{t}}$$

where $\overline{A}^{(c)} = diag\left(\overline{\boldsymbol{\alpha}}^{(c)}\right)$ and $\overline{\boldsymbol{\alpha}}(c) = \langle \boldsymbol{\alpha}^{(c)} \rangle_{Q(\boldsymbol{\alpha}^{(c)})}$.
The covariance S is given by $\left(-\frac{\partial^2 E}{\partial \Theta \partial \Theta^{\mathrm{t}}} \right)^{-1}$ after convergence (i.e. evaluated at the maximum $\overline{\Theta}$).
Let $\overline{\theta}_d^{(c)}$ and $S_{dd}^{(c,c)}$ denote the posterior mean and variance of $\theta_d^{(c)}$, respectively.

3. [α step] Update $Q(\alpha_d^{(c)})$, given $Q(\theta_d^{(c)})$.
$Q(\alpha_d^{(c)})$ is the Gamma distribution,

$$Q\left(\alpha_d^{(c)}\right) \sim \Gamma\left(\alpha_d^{(c)}; \gamma_d^{(c)}, \overline{\alpha}_d^{(c)}\right) \quad d = 1, \ldots, D,\ c = 1, \ldots, C,$$

where its mean parameter $\overline{\alpha}_d^{(c)}$ and the degree of freedom $\gamma_d^{(c)}$ are, respectively, updated as,

$$\overline{\alpha}_d^{(c)} = \frac{1 - \overline{\alpha}_d^{(c)} S_{dd}^{(c,c)}}{\left(\overline{\theta}_d^{(c)}\right)^2} \quad d = 1, \ldots, D,\ c = 1, \ldots, C$$

$$\gamma_d^{(c)} = \frac{1}{2} \quad d = 1, \ldots, D,\ c = 1, \ldots, C$$

4. [Pruning] If the mean parameters $\overline{\alpha}_d^{(c)}$ (i.e. the relevance parameters) exceed some pre-specified big threshold value (10^8 in our code), the corresponding weight parameters are effectively regarded as 0. Thus the corresponding dimensions are removed from the later estimation algorithm.

5. [Judge convergence] Iterate [θ step] and [α step] alternatively until the amount of parameter changes becomes small enough or until the number of iterations exceeds a pre-specified number.

We find that most of the relevance parameters diverge to the infinity after iterations (about 200 iterations in case of the quadrant stimuli experiment). The weight parameters corresponding to large relevance parameters become effectively zeros, thus we can prune these weight parameters. In practice, we prune weight parameters whose relevance parameters exceed a pre-specified threshold (10^8 in our code) to avoid computational ill-conditioning. This accelerates the speed of the algorithm significantly because it decreases the number of parameters while iterations. As initial relevance parameters, we used a vector whose elements are all one. The algorithm may converge to different estimates when different initial parameters are used, but we observed that the algorithm worked quite robustly for a modest range of initial parameters (from 1 to 1000).

Regularized logistic regression is obtained by introducing a single hyper-parameter that controls the total L2-norm of a weight parameter vector. In contrast, SLR uses a number of hyper-parameters, each of which controls the L2-norm of a corresponding weight parameter. This is formulated by replacing the prior distribution (a2) with $P(\Theta|\alpha) = N(0, \alpha I_{D \times C})$, where α is a scalar hyper-parameter and $I_{D \times C}$ is an identity matrix of size $D \times C$. We obtain the algorithm to estimate parameters of RLR by slightly modifying the α step in the above algorithm to $\overline{\alpha} = \frac{DC - \overline{\alpha} \sum S_{dd}^{(c,c)}}{\sum \left(\theta_d^{(c)}\right)^2}$.

References

Attias, H., 1999. Inferring parameters and structure of latent variable models by variational Bayes. Proc. 15th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Pub. , pp. 21–30.

Averbeck, B.B., Latham, P.E., Pouget, A., 2006. Neural correlations, population coding and computation. Nat. Rev., Neurosci. 7, 358–366.

Baker, C.I., Hutchison, T.L., Kanwisher, N., 2007. Does the fusiform face area contain subregions highly selective for nonfaces? Nat. Neurosci. 10, 3–4.

Bishop, C., Tipping, M.E., 2000. Variational relevance vector machines. Proceedings of the 16th Conference in Uncertainty in Artificial Intelligence, pp. 46–53.

Bishop, C., 2006. Pattern Recognition and Machine Learning. Springer, New York.

Boser, B., Guyon, I., Vapnik, V., 1992. A training algorithm for optimal margin classifiers. In Fifth Annual Workshop on Computational Learning Theory, pp. 144–152.

Carlson, T.A., Schrater, P., He, S., 2003. Patterns of activity in the categorical representations of objects. J. Cogn. Neurosci. 15, 704–717.

Cox, D.D., Savoy, R.L., 2003. Functional magnetic resonance imaging (fMRI) "brain reading": detecting and classifying distributed patterns of fMRI activity in human visual cortex. NeuroImage 19, 261–270.

Engel, S.A., Rumelhart, D.E., Wandell, B.A., Lee, A.T., Glover, G.H., Chichilnisky, E.J., Shadlen, M.N., 1994. fMRI of human visual cortex. Nature 369, 525.

Faul, A.C., Tipping, M.E., 2002. Analysis of sparse Bayesian learning. Adv. Neural Inf. Process. Syst. 8, 383–389.

Friston, K.J., Holmes, A.P., Worsley, K.P., Poline, J.B., Frith, C.D., Frackowiak, R.S.J., 1995. Statistical parametric maps in functional imaging — a general linear approach. Hum. Brain Mapp. 2, 189–210.

Haxby, J.V., Gobbini, M.I., Furey, M.L., Ishai, A., Schouten, J.L., Pietrini, P., 2001. Distributed and overlapping representations of faces and objects in ventral temporal cortex. Science 293, 2425–2430.

Haynes, J.D., Rees, G., 2006. Decoding mental states from brain activity in humans. Nat. Rev., Neurosci. 7, 523–534.

Haynes, J.D., Sakai, K., Rees, G., Gilbert, S., Frith, C., Passingham, R.E., 2007. Reading hidden intentions in the human brain. Curr. Biol. 17, 323–328.

Jaakkola, T.S., Jordan, M.I., 2000. Bayesian parameter estimation via variational methods. Stat. Comput. 10, 25–37.

Kamitani, Y., Tong, F., 2005. Decoding the visual and subjective contents of the human brain. Nat. Neurosci. 8, 679–685.

Kamitani, Y., Tong, F., 2006. Decoding seen and attended motion directions from activity in the human visual cortex. Curr. Biol. 16, 1096–1102.

Kriegeskorte, N., Goebel, R., Bandettini, P., 2006. Information-based functional brain mapping. Proc. Natl. Acad. Sci. U. S. A. 103, 3863–3868.

Krishnapuram, B., Carin, L., Figueiredo, M.A.T., Hartemink, A.J., 2005. Sparse multinomial logistic regression fast algorithms and generalization bounds. IEEE Trans. Pattern Anal. Mach Intell. 27, 957–968.

LaConte, S., Strother, S., Cherkassky, V., Anderson, J., Hu, X., 2005. Support vector machines for temporal classification of block design fMRI data. NeuroImage 26, 317–329.

Lal, T.N., Schroder, M., Hinterberger, T., Weston, J., Bogdan, M., Birbaumer, N., Scholkopf, B., 2004. Support vector channel selection in BCI. IEEE Trans. Biomed. Eng. 51, 1003–1010.

MacKay, D., 1992. Bayesian interpolation. Neural Comput. 4, 415–447.

Maldjian, J.A., Laurienti, P.J., Burdette, J.H., 2004. Precentral gyrus discrepancy in electronic versions of the Talairach atlas. NeuroImage 21, 450–455.

Maldjian, J.A., Laurienti, P.J., Kraft, R.A., Burdette, J.H., 2003. An automated method for neuroanatomic and cytoarchitectonic atlas-based interrogation of fMRI data sets. NeuroImage 19, 1233–1239.

Mitchell, T.M., Hutchinson, R., Just, M.A., Niculescu, R.S., Pereira, F., Wang, X., 2004. Learning to decode cognitive states from brain images. Mach. Learn. 57, 145–175.

Norman, K.A., Polyn, S.M., Detre, G.J., Haxby, J.V., 2006. Beyond mind-reading: multi-voxel pattern analysis of fMRI data. Trends Cogn. Sci. 10, 424–430.

Neal, R.M., 1996. Bayesian learning for neural networks. Lect. Notes Stat. 118 Springer.

O'Toole, A.J., Jiang, F., Abdi, H., Haxby, J.V., 2005. Partially distributed representations of objects and faces in ventral temporal cortex. J. Cogn. Neurosci. 17, 580–590.

Sato, M., 2001. On-line model selection based on the variational Bayes. Neural Comput. 13, 1649–1681.

Spiridon, M., Kanwisher, N., 2002. How distributed is visual category information in human occipito-temporal cortex? An fMRI study. Neuron 35, 1157–1165.

Strother, S.C., Anderson, J., Hansen, L.K., Kjems, U., Kustra, R., Sidtis, J., Frutiger, S., Muley, S., LaConte, S., Rottenberg, D., 2002. The quantitative evaluation of functional neuroimaging experiments: the NPAIRS data analysis framework. NeuroImage 15, 747–771.

Tipping, M.E., 2001. Sparse Bayesian learning and the relevance vector machine. J. Mach. Learn. Res. 1, 211–244.

Tipping, M.E., Faul, A., 2003. Fast marginal likelihood maximisation for sparse Bayesian models. Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics.

Ting, J.A., D'Souza, A., Yamamoto, K., Yoshioka, T., Hoffman, D., Kakei, S., Sergio, L., Kalaska, J., Kawato, M., Strick, P., Schaal, S., 2008. In: Usui, S., Grillner, S., Bjaalie, J. (Eds.), Variational Bayesian Least Squares: An Application to Brain-Machine Interface Data, Neural Networks; special issue Neuroinformatics, Vol. 21. No.9, to appear.

Vapnik, V.N., 1998. Statistical Learning Theory. Wiley, New York.

Wolpaw, J.R., Birbaumer, N., McFarland, D.J., Pfurtscheller, G., Vaughan, T.M., 2002. Brain-computer interfaces for communication and control. Clin. Neurophysiol. 113, 767–791.

Worsley, K.J., Liao, C.H., Aston, J., Petre, V., Duncan, G.H., Morales, F., Evans, A.C., 2002. A general statistical analysis for fMRI data. NeuroImage 15, 1–15.