Pattern Recognition of Brain Image Data

PROBID

PROBID is an academic toolbox for the analysis of MRI data using pattern recognition approaches

Development Team:

Andre Marquand: Pattern recognition algorithms, software and interface development

Jane Rondina: Software/interface development and documentation

Janaina Mourao-Miranda: Pattern recognition algorithms

Vanessa Rocha-Rego: Interface testing and documentation

Vincent Giampietro: Interface testing and documentation
## Contents

Pattern recognition analysis ........................................................................................................ 4

Feature extraction ......................................................................................................................... 5

Feature selection ......................................................................................................................... 5

Feature based classification ......................................................................................................... 6

Support Vector Machine (SVM) .................................................................................................... 6

Discrimination Maps .................................................................................................................... 8

Gaussian Process Classifier (GPC) .............................................................................................. 8

USER INSTRUCTIONS .................................................................................................................. 9

System Requirements ................................................................................................................ 9

Installation .................................................................................................................................. 9

General Pipeline .......................................................................................................................... 11

EXAMPLE FMRI EXPERIMENT .................................................................................................. 12

Experiment definition (Functional) .............................................................................................. 13

Preprocessing (Functional) .......................................................................................................... 21

Computing the Kernel Matrix ...................................................................................................... 24

Pattern Recognition .................................................................................................................... 27

Classification (Training and Test) ............................................................................................... 30

Permutation Test ........................................................................................................................ 32

Discrimination Map ..................................................................................................................... 33

t- Map ......................................................................................................................................... 34

EXAMPLE STRUCTURAL MRI EXPERIMENT ......................................................................... 36

Experiment Definition (Structural) .............................................................................................. 36

Preprocessing (Structural) .......................................................................................................... 42

Kernel Matrix Computation ....................................................................................................... 45

Pattern Recognition .................................................................................................................... 47
THEORETICAL INTRODUCTION

Univariate or voxel based analysis approaches have been traditionally used to analyze neuroimaging data (e.g. General Linear Model, GLM and Voxel Based Morphometry, VBM). These approaches are limited in their ability to characterize differences between groups because they are significantly biased toward detecting group differences that are highly localized in space and linear in nature. Therefore they are significantly less powerful and appropriate in cases for which the group differences are spatially distributed and subtle (Davatzikos, 2004). Structural and functional MRI data are inherently multivariate in nature, since each scan contains information about, for example, tissue structure or brain activation, at thousands of measured locations (voxels). Considering that most brain functions are distributed processes involving a network of brain regions, it would seem desirable to use the spatially distributed information contained in the data to obtain a better understanding of brain functions in normal and abnormal conditions. This spatially distributed information can be investigated using multivariate pattern recognition methods. Here we present a toolbox that performs multivariate pattern recognition analysis of neuroimaging data.

Pattern recognition analysis
Statistical pattern recognition is a field within the area of machine learning which is concerned with the automatic discovery of regularities in data through the use of computer algorithms, and with the use of these regularities to take actions such as classifying the data into different categories (Bishop, 2006). In the case of neuroimaging, brain scans are treated as spatial patterns and statistical learning methods are used to identify statistical properties of the data that discriminate between groups of subjects (e.g. task 1 vs. task 2 or patients vs. controls). The general idea of the pattern recognition analysis of fMRI data is illustrated in Box 1.
More specifically pattern recognition methods consist of three components: feature extraction, feature selection and feature based classification.

**Feature extraction**
Transforming the input data into a set of features is called feature extraction. In the context of neuroimaging this consists of transforming a 3 (or 4) dimensional brain scan into a long vector of features (voxels) within the brain. If the features are carefully chosen, it is then expected that the feature set will extract the relevant information from the input data in order to perform the desired classification task.

**Feature selection**
Feature selection is the technique, commonly used in machine learning, of selecting a subset of relevant features in order to build robust learning models. In the context of neuroimaging this technique could consist, for example, in selecting regions of interest or in using a mask to select a subset of voxels based on a previous analysis. By
removing most irrelevant and redundant features from the data, feature selection may improve the performance of learning models by:

* Alleviating the effect of the curse of dimensionality.
* Speeding up the learning process.

Feature based classification
Feature based classification is the process by which individual examples are separated into groups based on quantitative information from one or more features in the example and based on a training set of previously labeled items. In the context of neuroimaging the task of classifying the images into two classes (e.g. patients vs. controls) can be viewed as finding a separating hyperplane or decision boundary. The classification procedure consists of two phases: training and testing. During the training phase, the algorithm finds a hyperplane that separates the examples in the input space according to their class labels. The classifier is trained by providing examples of the form \{(x, c)\}, where \(x\) represents a spatial pattern (e.g. brain scan) and \(c\) is the class label (e.g. patient or control). Once the decision function is learned from the training data it can be used to predict the class of a new test example. A hypothetical example of classification in a 2D space is displayed in Box 2.

Depending on the machine learning method used, there could be many possible decision boundaries or hyperplanes (e.g. linear discriminant analysis, support vector machine, Gaussian processes, etc). However, some classifiers that correctly classify a training set may fail for unseen examples and therefore generalize badly. One can therefore choose between different learning methods or classifiers based on generalization performance.

Support Vector Machine (SVM)
The SVM algorithm (Boser et al., 1992) finds the largest margin hyperplane. Margin is the distance from the separating hyperplane to the closest training examples. It has been demonstrated that the optimal hyperplane is the one with maximal margin (i.e. more separation between the classes). A larger margin corresponds to a better generalization performance. An SVM classifier for a two dimensional problem is illustrated in Box 2.

In the PROBID implementation we use a linear kernel SVM to reduce the risk of overfitting the data and to allow direct extraction of the weight vector as an image (i.e. the SVM discrimination map). The linear kernel only has one parameter \(C\) that controls
the trade-off between having zero training errors and allowing misclassifications. This is fixed at $C = 1$ for all cases (default value).\(^1\)

---

**Box 2: Machine Learning Classification**

The figures below show an illustration of a machine learning classification problem between patients (red circles) and healthy controls (blue squares) for the simplified case of only two variables or voxels. Each axis represents the measurement in one voxel. Each symbol (circle or square) represents a brain scan of a different subject. In Figure A the dashed lines represent linear classifiers that correctly separate the groups. During the training, the machine learning approach finds the best classifiers according to a pre-determined criterion. Figure B illustrates the optimal classifier determined by a specific machine learning approach called the Support Vector Machine (SVM) (Boser et al., 1992). The optimal classifier (dashed line) is the one with a maximal margin of separation between the two groups. The training examples that lie on the margin are called support vectors (circled symbols). The green symbols represent new subjects that are classified as patients or controls depending on their positions in relation to the classifier. The vector $w$ is called classifier’s weight vector and carries the information about which variables or voxels are relevant for discriminating between the groups. The weight vector can be plotted as an image showing the relative importance or weight of each voxel in the brain for the classification (i.e. a discriminating map). Although, this example shows only linear classifiers, there are non-linear extensions of the SVM to deal with non-linearly separable cases.

---

\(^1\) In PROBID, SVM classification is provided by the LIBSVM library (http://www.csie.ntu.edu.tw/~cjlin/libsvm/).
Discrimination Maps
If the input space is the voxel space (one voxel per dimension) the weight vector normal to the hyperplane will be the direction along which the images of the two groups differ most. Hence, it can be used to generate a map of the most discriminating regions (i.e. a discrimination map). For example, given two groups, patients and controls, with the labels +1 and -1 respectively a positive value in the discrimination map means relatively higher values in patients than in controls and a negative value means relatively higher values in controls than in patients. Because the classifier is multivariate by nature, the combination of all voxels as a whole is identified as a global spatial pattern by which the groups differ (i.e. the discriminating pattern). This also means that you should avoid talking about the behaviour of each brain region separately from the rest of the pattern.

Gaussian Process Classifier (GPC)
Gaussian processes (Rasmussen and Williams, 2006) are Bayesian methods for high-dimensional regression or classification, and inference is performed according to the rules of probability. Gaussian process classification can most easily be understood as an extension of logistic regression where a Gaussian process prior is placed over a latent function which models relationships between the input data. One of the attractions to GPC inference is that it produces probabilistic class predictions. In practice, these are obtained by computing the posterior expectation of the latent function evaluated at the test data points. Exact inference is not analytically tractable for classification, but the expectation propagation algorithm is known to have good performance (see Rasmussen and Williams, 2006 for details). In the PROBID implementation, hyperparameters controlling regularization and a bias are set by an empirical Bayesian approach. PROBID also provides two mapping methods for neuroimaging data (described in detail in Marquand et al., in press).²

² The Gaussian Processes for Machine Learning toolbox (http://www.gaussianprocess.org/gpml/) provides GPC inference in PROBID
USER INSTRUCTIONS

System Requirements

PROBID has been tested on Matlab versions 7.1, 7.2, 7.4, 7.8 and 7.9 in Windows XP and vista, Mac OS X, Solaris Unix, and CentOS/Ubuntu Linux. PROBID may work on earlier versions of Matlab, but Matlab 7.2 or higher is recommended.

Installation

Installing PROBID is a simple process:

1. Unzip the distribution in an empty directory.
2. If you are running a version of Matlab prior to 7.2, you will need to delete the SVM precompiled binaries and replace them with the copies in the matlab_7.1/ subdirectory of the PROBID distribution. On windows this can be achieved by opening a command window and typing:

   cd <path-to-probid_installation>
   del svm*.mexw32
   copy matlab_7.1/svm*.dll .

   This step is not necessary for Matlab 7.2 or greater.
3. Start Matlab
4. Add the PROBID installation directory to the Matlab search path by typing:
addpath <path-to-probid-installation>

from within the Matlab command window.
Alternatively, if you are using the full Matlab graphical interface, you can add this folder to your path by choosing 'File > Set Path...' followed by the 'Add Folder...' button

5. Start the application by typing probid at the Matlab command prompt.
6. (Optional) Compile the GP classification libraries by running the
   compile_gpml.m script (found in the utils/ subdirectory of the PROBID software distribution).\(^3\)

---

**Mode Selection**

PROBID currently supports pattern recognition of functional MRI (time series, GLM coefficients or spatiotemporal analysis), structural MRI (e.g. gray matter images), arterial spin labeling (ASL) data and a text processing module which can be used for any modality that can be specified in ASCII plain text format (e.g. behavioural data).

Please note that the ‘Structural Images’ processing module is suitable for any modality for which there is only a single image per subject (e.g. diffusion-weighted MRI).

For didactic purposes, we provide an example of a simple fMRI experiment along with the parameter settings necessary to run a straightforward SVM pattern recognition

---

\(^3\) Note that GPC inference will still run without performing this step, but computation time can be reduced using the compiled version of the software.
The data used in this demonstration can be downloaded from http://www.brainmap.co.uk/probid/

The first step in PROBID is to choose your analysis modality (Fig. 1).

In this tutorial we will be analyzing full fMRI time series, so please choose the ‘BOLD Timeseries’ option and click on ‘Select’.

![Figure 1: Modality Selection](image)

**General Pipeline**

In general, there are four distinct steps in a pattern recognition analysis of MRI data:
1. Data/Design Specification
2. Preprocessing
3. Computing Kernel Matrix
4. Pattern Recognition

Each of these phases can be initiated by clicking the appropriate button on the main application window. Note that the preprocessing mentioned in step 2 is distinct from the SPM/FSL pre-processing, which must be done before data can be analyzed with PROBID. It is beyond the scope of this document to discuss this process in detail.

**EXAMPLE FMRI EXPERIMENT**

In this tutorial experiment, stimuli were presented in an event-related fashion. There were three different active conditions: viewing unpleasant (dermatological diseases), neutral (people) and pleasant (pretty girls in swimsuits) images, and a control condition (fixation). The stimuli were presented in a random order according to a randomized design. There were 10 image presentations (events) of each condition. For the purposes of this tutorial, we will use data from 5 subjects (subject 03, subject 04, subject 05, subject 06 and subject 07).
**Experiment definition (Functional)**

The first stage in a pattern recognition analysis is to define the experimental design. All design information is stored in a file called **Expt_def.mat**, which is stored in a user-defined location. Click on the ‘Specify Design/Data’ button to start defining your experiment.

![Figure 2: Design/Data specification](image)

Before any subject, group or task information can be input, the data directory has to be created using the ‘Analysis Dir’ button, all information in the ‘Globals’ panel must be specified and the groups and tasks should be named accordingly using the dedicated panels.
Figure 3: Experiment Definition
A description of the global parameters and of the necessary values for the tutorial dataset is provided below in Table 1:

Table 1

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis Dir</td>
<td>The location where Expt_def.mat will be stored along with the preprocessed data (as *.mat files).</td>
<td>&lt;user specified&gt;</td>
</tr>
<tr>
<td>N groups</td>
<td>Number of groups for which fMRI data are available.</td>
<td>1</td>
</tr>
<tr>
<td>N subjects / group</td>
<td>Total number of subjects in the fMRI experiment. Note that all classes must have the same number of subjects</td>
<td>5</td>
</tr>
<tr>
<td>N tasks</td>
<td>Number of tasks in each fMRI run.</td>
<td>3</td>
</tr>
<tr>
<td>N repetitions / task</td>
<td>Number of repetitions of each task. This corresponds to the number of blocks for a block design experiment or the number of events for and event-related design.</td>
<td>10</td>
</tr>
<tr>
<td>TR</td>
<td>Repetition Time <strong>in milliseconds</strong></td>
<td>3000</td>
</tr>
<tr>
<td>Group Name</td>
<td>Name of each group for which fMRI data are available.</td>
<td>e.g. Healthy Controls</td>
</tr>
<tr>
<td>Task Name</td>
<td>Name of each task in each fMRI run.</td>
<td>e.g. Ple, Unp and Neu</td>
</tr>
</tbody>
</table>

Please note that the name you assign to each task will be appended to each data file, so short names are desirable.

Everytime you have finished entering information in a panel or subpanel, click the ‘Apply’ button to save your work.
PROBID accepts different onsets for different subjects. However the number of repetitions of each task (i.e. number of blocks for a block design experiment or number of events for an event-related design) has to be the same for all subjects. The length parameter specifies the duration of the task (i.e. duration of the block for a block design or duration of the event for an event-related design). In practical terms, this denotes how many temporally consecutive volumes will be averaged to construct a sample for the classifier. The onsets should be specified in units of TR and not in seconds.

For each subject and for each task the user needs to enter the task onset and the length parameters (both should be entered in units of TR and not in seconds). If the design is the same for all subjects the user can select the ‘Use the same design for all subjects’ option before specifying the tasks. In this case, once the tasks are specified for the first subject (by clicking on ‘Apply’) onsets and lengths will be copied to all subjects.

To accommodate for an haemodynamic delay, the actual onset applied to each subject’s time series will be automatically shifted by a number of volumes determined by:

\[
delay = 3 / (TR/1000)
\]

This is rounded down to the nearest integer. For TRs of 2 or 3 seconds, this corresponds to delaying the onsets by one TR. It is important to ensure that the haemodynamic delay does not cause the duration of an event to exceed the maximum length of the time series. For example, if the time series length is 100 images, and the last event has an onset of 99 with a duration of 2 TRs, it will not be possible to include
this event in the paradigm, as adding the haemodynamic delay causes it to extend beyond the length of the time series. Possible solutions to this problem include excluding the offending event, advancing its onset or shortening the duration to fit within the time series length.

There are three tasks in the tutorial dataset, which correspond to subjects viewing pleasant, unpleasant and neutral pictures. In this case we will use a task length of 2 TRs and specify the onsets as follows (same onsets for all the subjects):

<table>
<thead>
<tr>
<th>Task Number</th>
<th>Task Name</th>
<th>Onset Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PLE</td>
<td>1, 11, 29, 42, 54, 67, 86, 105, 108, 119</td>
</tr>
<tr>
<td>2</td>
<td>UNP</td>
<td>8, 15, 25, 34, 45, 49, 72, 81, 90, 103</td>
</tr>
<tr>
<td>3</td>
<td>NEU</td>
<td>3, 37, 39, 47, 58, 61, 76, 79, 96, 114</td>
</tr>
</tbody>
</table>

After clicking ‘Apply’, the task details will be saved to the Expt_def.mat file. A sample configuration screen for the tutorial dataset is presented below:
Once all the design-related information has been entered, you need to specify which data files to use for each subject, in each group and task.

For functional data, PROBID accepts Analyze or NifTI *.hdr/*.img pairs, NifTI .nii files and 4D volumes (.gz files will be uncompressed if needed) By default the program also...
assumes the default SPM image dimensions of 79x95x69 with an isotropic voxel resolution of 2mm, but also includes a mask for image dimensions 91x109x91 (FSL default). If your image dimensions don’t match these, it will be necessary to use a custom binary brain mask, which should have the same dimensions and orientation as your image data. Such a brain mask can be easily constructed using most common fMRI analysis packages (e.g. SPM/FSL).

To specify the fMRI data for each subject, click the ‘Files’ button and add the appropriate header files (*.hdr). Note that the first image specified will correspond to time point “1” when defining the onsets.

![Select Analyze files dialog box](image)

**Figure 5: Data selection**
In the case of this tutorial, please include for each subject all the MRI files matching “swsub*.hdr”, i.e.:

Figure 6: Experiment Definition
Do not forget to click on ‘Apply’ after selecting the data files for each subject in order to add these to Expt_def.mat. Repeat for all 5 subjects.

Once you have completed the specification of the subjects, groups and tasks parameters, the dataset is ready to be pre-processed. Click on ‘Close’ to close the ‘Experiment Definition – BOLD Timeseries Processing’ window.

A previously specified Expt_def.mat file can be loaded by clicking the ‘Load data previously defined’ button. This may be useful to check if the specifications are correct or to modify the configuration.

**Preprocessing (Functional)**

To start the preprocessing module, click on the ‘Preprocess’ button in the main application window.
You first need to specify the Expt_def.mat file containing the fMRI paradigm information (Fig. 8) by clicking the ‘Expt. Def.’ button.

The second step is to specify a mask with the same dimensions as the data. The default mask is the SPM 79x95x69, but 91x109x91 masks are also provided. If this does not correspond to your images you can select an ROI mask created by an external program (e.g. MRIcro, MARSbar) with the relevant dimensions.

It is recommended to check the model prior to preprocessing. A simple set of tests is provided with the package. To run these click the ‘Check model’ button. If any errors are detected, go back to the ‘Specify Design/Data’ module to correct them before running the preprocessing.
To preprocess the fMRI paradigm, click ‘Preprocess’. All preprocessed data will be stored in the same directory as the Expt_def.mat file.

Figure 8: BOLD time series processing
Computing the Kernel Matrix

With the exception of the class labels, the Kernel matrix contains all the information necessary to perform pattern recognition. A separate Kernel matrix should be computed for each binary contrast. To compute the Kernel matrix, click the appropriate button in the main application window.

![Figure 9: Configure Kernel Matrix](image)

Firstly, specify the results directory with the ‘Results Dir’ button. Note that a separate directory should be used for each binary contrast and ideally this directory should be distinct from the one used to store the Expt_def.mat file and preprocessed data.

Secondly, specify the location of the Expt_def.mat file with the ‘Expt. Def.’ button.
Next, specify the group and task which constitutes each class. Either the group or the class can be the same for class 1 and class 2 but not both.

Figure 10: Compute Kernel Matrix
After specifying the classes, the Kernel matrix can be computed by clicking the appropriate button. Depending on the size of your dataset, this might take some time.
Pattern Recognition

The pattern recognition module can be started from the main application window.

![Pattern Recognition](image)

Figure 12: Pattern Recognition

Again the first steps are to specify the results directory and the location of the Expt_def.mat file (c.f. above).
The second step is to specify the pattern classification algorithm. The current version of PROBID supports two classifiers: Support Vector Machine (SVM) and Gaussian Process Classifier (GPC). SVM is a categorical classifier (i.e. it outputs +1/-1 for each test example). GPC is a probabilistic classifier (i.e. it gives predictive probabilities for each test example).
Note that the ‘Test mode’ radio buttons control whether sample averaging is used before the test phase. Averaging all samples before testing will generally lead to better generalization performance because it increases the signal-to-noise ratio in the data, but it may not be appropriate for all experimental designs.
Classification (Training and Test)

The classifier can be trained by clicking the ‘Train & Test Classifier’ button.

Figure 14: Train and Test Classifier
After training and testing the classifier, classification accuracy, specificity and sensitivity are reported in the results window, and a graphical representation of the test projections is displayed in the panel on the right hand side. The 'Save Figure' button allows this plot to be saved to a user specified location, as a Matlab figure or as a standard .tiff or .jpg picture.
Permutation Test

The permutation test is used to assess the statistical significance of the derived pattern and whether the predictions it provides are better than those that would be expected by chance (50%). The permutation test, while computationally demanding, is straightforward to run. Enter the desired number of permutations in the text box and click the 'Run Permutation Test' button. Results will be displayed in the 'Results' window.
A multivariate discrimination map can be generated by clicking the ‘Create Discrimination Map’ button. The map will be saved as an Analyze *.hdr/*,img image pair.
in the results directory. To visualize this map, we recommend using a software package such as MRICro. ([http://cnl.web.arizona.edu/mricro.htm](http://cnl.web.arizona.edu/mricro.htm)).

**t- Map**

A voxel-wise unpaired t-test can be applied to the input images according to the contrast specified. The resulting map will also be saved as an Nifti *.hdr/*,img image pair in the results directory and can be visualized in the same way as the discrimination map.

![Figure 17: Discrimination Map](image)
**IMPORTANT:** Since SVM and GPC are multivariate discriminative approaches, we cannot make inferences regarding individual regions on the discrimination maps. The discrimination map shows a spatially distributed pattern of discriminating regions. Further interpretations of individual regions will depend on a t test which can detect local univariate effects.
**EXAMPLE STRUCTURAL MRI EXPERIMENT**

The next section will illustrate how to run a pattern recognition analysis on a structural dataset with PROBID.

**Experiment Definition (Structural)**

In this experiment, anatomical MRI images were acquired from two groups of individuals: 21 healthy subjects and 21 patients with autism spectrum disorder (ASD). The experiment is described in more detail in (Ecker et al., in press). Here, for didactic purposes, we only describe the analysis performed on gray matter images.

To start, you need to select the ‘Structural Images’ mode:

![Figure 18: Modality selection](image)
Click on the 'Specify Design / Data' button in the main screen of the toolbox:

![Figure 19: Design/Data selection](image)

Similarly to the way it is done with functional images, all paradigm information is kept in a file called Expt_def.mat, which is stored in a user-specified location.

Start by creating the data directory using the ‘Analysis Dir’ button, fill in all the required information in the ‘Globals’ panel and then name the groups.
Figure 20: Experiment Definition
The parameters used for this tutorial are as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis Dir</td>
<td>The location where Expt_def.mat will be stored along with the preprocessed data (as *.mat files).</td>
<td>&lt;user specified&gt;</td>
</tr>
<tr>
<td>N groups</td>
<td>The number of groups considered in the study</td>
<td>2</td>
</tr>
<tr>
<td>N subjects / group</td>
<td>The number of subjects per group.</td>
<td>21</td>
</tr>
<tr>
<td>Group Name</td>
<td>Name of each group. Note: press the ‘Apply’ button after naming each group</td>
<td>e.g. patients, controls</td>
</tr>
</tbody>
</table>

Press the 'Apply' button to create/save the Expt_def.mat file after finishing to enter the global information.

Once you have specified the groups, move on to last panel ‘Subject Data’ to associate each subject with its data files.

<table>
<thead>
<tr>
<th>Group</th>
<th>You can select the number corresponding to the group for which you want enter data (the range varies according to the number of groups defined in the ‘Globals’ panel).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject</td>
<td>Select the number corresponding to subject you are going to associate data files with.</td>
</tr>
<tr>
<td>Data files</td>
<td>When you click this button, a window will pop up (Fig. 21) enabling you to locate the header files corresponding to the subject you are including in the analysis. In this window (shown in the next figure), select the desired files, press the 'Add' button and then the 'Done' button. Before moving on to the next subject, remember to click on 'Apply' to save your selection to the Expt_def.mat file.</td>
</tr>
</tbody>
</table>

**Important:** Your subjects should be matched between the groups and they should be entered in the program accordingly: subject 1 in the first group should be matched with subject 1 in the second group, subject 2 in the first group should be matched to subject
2 in the second group… This is because during each iteration of the cross-validation procedure the classifier leaves out a pair of subjects for testing.
NOTE: After including a new subject, you must press the 'Apply' button.

You can reload the contents of a previously created Expt_def.mat file by clicking the 'Load data previously defined' button.
Preprocessing (Structural)

Once you have entered all the experimental data and selected all the subject data files, go back to the main probid window.

Click on ‘Preprocess’ to open the preprocessing window.
Start by selecting the relevant Expt_def.mat file by clicking on the ‘Expt. Def.’ button.
The 'Mask file' button enables you to select a mask that limits the voxels considered in the analysis. If you do not select one, a custom mask will be constructed from the data.

Once the Expt_def.mat and the mask have been selected, press the 'Check model' button to perform some basic tests on the integrity of your design (the results are shown underneath 'Results of checking').
If there are no errors, you can come back to the main screen and start to compute the kernel matrix.

**Kernel Matrix Computation**

Select ‘Compute Kernel’ in the main panel.

![Figure 25: Compute Kernel](image)

Once the kernel specification window has opened, press the 'Results Dir' button to select/create a directory to store the results into. You must also select the relevant Expt_def.mat file by pressing the 'Expt. Def.' button.
After that, you can specify the desired contrast. In this example, we want to contrast the ‘Patient’ and ‘Control’ groups.

NOTE: As this is a structural analysis, there are no tasks to select and ‘1: STRUCT’ is written in the task box.

Click on the 'Compute Kernel Matrix' button to build the kernel matrix. This process may take some time if you are using a large number of subjects.
Pattern Recognition

Once the ‘Kernel matrix complete' message appears in the box on the right of the 'Compute Kernel Mat…' button, close the kernel specification window and come back to the main screen.

The last step is ‘Pattern Recognition':

![Pattern Recognition](image)

Figure 27: Pattern Recognition

When you click this button, a new window pops up in which you first need to select the results directory and the file Expt_def.mat file.

After this is done, you can choose the classification method that you want to use. In
this version of the toolbox there are two classification options: SVM and GPC (c.f. functional tutorial for a description of these).

The classifier can then be trained by pressing 'Train & Test Classifier'.

A figure containing the classification results will eventually appear towards the middle of the window. Specificity (true negative rate), sensitivity (true positive rate) and accuracy (the average of the former two) are reported at the bottom of the graph.
In this example, the sensitivity (rate of autistic patients correctly classified) was 80.95% and the specificity (rate of healthy subjects correctly classified) was 80.95%, resulting in an overall accuracy of 80.95%.

Similarly to functional data (c.f. above) you can perform a permutation test, which calculates the statistical significance of the results obtained in relation to chance (i.e.
0.5). If required you can change the number of permutations which is 1000 by default.

The output of the permutation test are presented in the ‘Results’ window.

You can create a discrimination map as well as a t-map by clicking the corresponding buttons in the same way as in the functional analysis tutorial.

**SCRIPTING**

All the PROBID functionalities described above (and many more) can be scripted. In fact PROBID is comprised of a set of functions, which can be invoked either from the graphical interface or from the Matlab command line. It is beyond the scope of the present document to describe PROBID scripting in detail, but sample batch scripts implementing a comprehensive analysis pipeline are provided in the utils/ folder of the PROBID software installation.

**REFERENCES**