A graphical tool for computational analysis of infrared spectroscopy datasets

This document explains how to use **bmtool**, *a graphical user interface (GUI) for identification of biomarkers in datasets generated from infrared spectroscopy.* **bmtool** *a frontend for a function library (toolkit) and also part of it.*

Contents

1. Installation

System requirement: the oldest MATLAB version tested was r2007b, although the toolkit may possibly work in older versions.

1) Download and extract the **bmtool.zip** file to a folder of choice. This will create five folders: **doc**, **lib**, **mat**, **txt** and **work**
2) Start MATLAB
3) Add the **lib** folder to your MATLAB path.

2. Supported dataset types

.mat files – **bmtool** loads MATLAB data files (**.mat**) that contain a structured variable called "DATA". Such files are usually generated by an utility that reads data from the MySQL database (see "External software" below).

A sample **.mat** file (**vc&mnng.mat)** is included in **mat** folder. Each spectrum is assigned one of the following classes: **{'VC|N', 'VC|T', 'MNNG|N', 'MNNG|T'}**, where VC stands for vehicle control, MNNG stands for *N*-Nitroso-*N*-methylnitroguanidine, N stands for non-transformed, and T stands for transformed. The class labelling can be seen as being hierarchic.

Text files – **bmtool** can read text files that obey the following convention: each line of the file contains one spectrum followed by an integer number indicating the class

the spectrum belongs to. Such files are typically edited in Excel or an equivalent program and generated as CSV (Comma-separated values) files.

3. Using `bmtool`

Start MATLAB and change the current directory to the work folder created before upon zip file extraction. Then at the MATLAB command prompt, type

```
>> bmtool
```

This will open the **`bmtool`** GUI (Figure 1).

The tool has a set of numbered panels that represent the usual order of the operational steps. These steps go from loading a dataset to visualizing analysis results.

All operations execute MATLAB code that is generated real-time. Once each piece of code is executed, it is added to the **Code generation** panel, making bmtool also a starting point for customized tasks and a guide for a series of functions of the library.

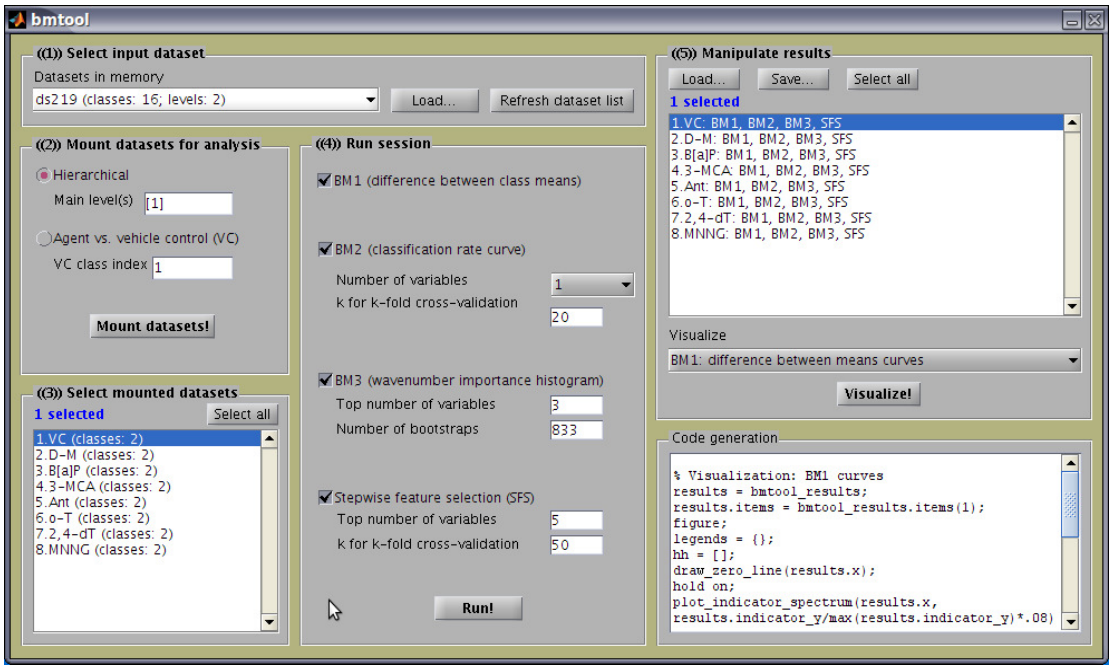

Figure 1 – the `bmtool` graphical user interface (GUI). The tool is normally operated sequentially in the order which the panels are numbered. All operations performed generate MATLAB code that appears in the Code generation box.

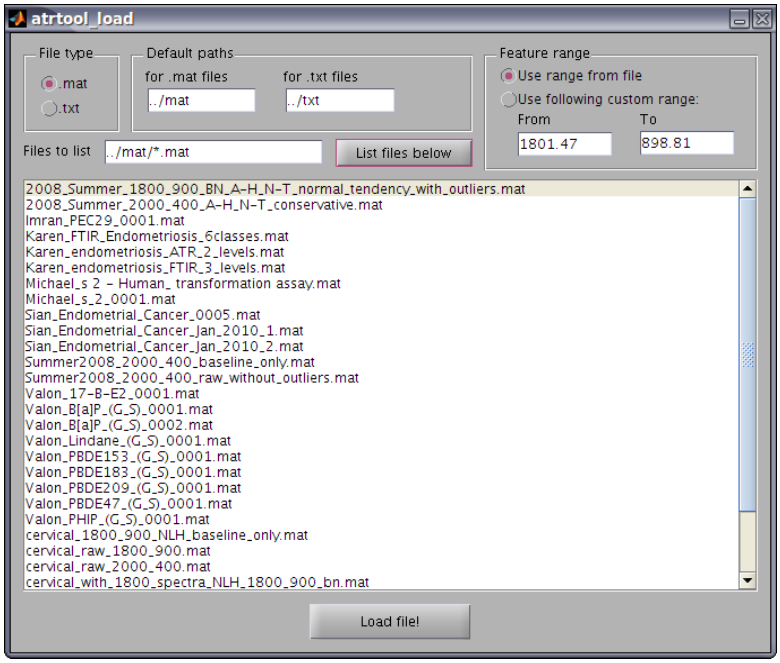

Figure 2 – file load dialog box.

((1)) Select input dataset

All the variables in MATLAB memory that start with "**ds**" are considered to be valid datasets and are shown in the Datasets in memory drop-down menu. Datasets can be loaded pressing the **Load…** button. This will open a file load dialog box (Figure 2). By convention, mat files are stored in the **mat** folder and text files are stored in the **txt** folder.

Alternatively to using the file load dialog box, two scripts (**load_4_classes.m** and **load_2_classes.m**) exist in the **work** folder, intended to load the provided **.mat** file for respectively comparing non-transformed *versus* transformed and comparing MNNG *versus* VC.

Comparing non-transformed *versus* transformed

At the MATLAB prompt, type

```
>> load_4_classes
>> bmtool
```

This script only loads the **vc&mnng.mat** file, which can also be done using the file load dialog. You should see a four-class dataset in the **Datasets in memory** drop-down menu. The dataset loaded in this way is suitable for the **Hierarchical** way of mounting datasets for analysis (see below).

Comparing MNNG *versus* VC

When comparing VC against MNNG, only the first hierarchic level is desired. A script that loads the dataset and makes it two-class (*i.e.*, **{'VC', 'MNNG'}**) is provided. In the MATLAB prompt, type

```
>> load_2_classes
>> bmtool
```

You should see a two-class dataset in the **Datasets in memory** drop-down menu. The dataset loaded in this way is suitable for the **Agent vs. vehicle control (VC)** way of mounting datasets for analysis (see below).

((2)) Mount datasets for analysis

This step generated sub-datasets of the full dataset, usually each containing two classes, for biomarker analysis. There are two ways to generate such datasets:

Hierarchical – this way works only when the input dataset has hierarchical levels (as is the case in the provided mat file). Being hierarchical means that the class labels have a "|" dividing levels of hierarchy (*e.g.*, **{'VC|N', 'VC|T', 'MNNG|N', 'MNNG|T'}**). By specifying which is the **Main level(s)**, the full dataset will be split the dataset (when you click Mount datasets!) using the **Main level(s)** and each generated sub-dataset will have only the classes contained in the not-main hierarchical level(s) (*e.g.*, **{'N', 'T'}**).

Agent vs. vehicle control (VC) – this way generates two-class sub-datasets of the full dataset combining a reference class (specified in VC class index) with each of the remaining classes in turn. Hierarchy in classes is disregarded.

Sub-datasets are generating by clicking the **Mount datasets!** button.

((3)) Select mounted datasets

This simple step allows for choosing which of the mounted datasets will undergo analysis. Mounted datasets are analysed individually and each dataset generates one results item.

((4)) Run session

The basic time unit to measure the complexity of the methods below will be the train-test time (*tt*), which represents the time for training and testing a classifier in the specific context where formulas are applied.

The number of variables in the dataset is represented by *nv*.

The computing time is represented by *t*.

BM1: difference-between-means

BM1 has no parameters and takes only a fraction of a second to generate output.

BM2: classification rate curve/surface

Number of variables can be 1 or 2. By choosing 1, results will be curves, whereas for 2 variables, the output result will be a 2-D surface.

The number of cross-validation folds (**k for k-fold cross-validation**) will determine the stability of the curves/surfaces across different runs, as these outputs are statistics. Is is generally accepted that 10-20-fold cross-validation is suitable in almost all cases [2].

Time considerations for each mounted dataset:

For one Number of variables = 1: $t = nv \times tt$

For Number of variables = 2: $t \cong \frac{nv^2}{2} \times tt$

BM3: wavenumber importance histogram

This method can be very time consuming, potentially taking hours to generate results, because many train-test sessions need to be performed to generate the histograms.

The following formula gives an idea about the time taken to generate the result for each dataset: $t \cong \frac{TopVars!}{(nv - TopVars)!} \times NoBootstraps \times tt$

((5)) Manipulate results

This step provides simple result loading and saving. Results are usually stored in .mat files.

One can select which results to be used in visualization.

In order to specific visualizations to work, corresponding methods outputs need to be stored in the results. The results listbox shows the names of the datasets followed by a ":" and the result outputs stored.

- **BM1: difference between means curves**: standard visualization for BM1 [1].
- **BM2: classification rate curves**: standard visualization for BM2 [1].
- **BM2: classification rate surface**: Visualization for BM2 when this method is run using two variables at a time instead of one. This visualization allows one to inspect the classification rates corresponding to wavenumbers used in pairs. This visualization uses only the first selected result.
- **BM3: best wavenumbers histogram**: Standard visualization for BM3.
- **SFS: [#wn]x[best average classification rate]**: Shows the effect of adding variables to a set of wavenumbers in the classification rate
- **SFS: evolution**: This visualization uses only the first selected result.
- **Mixed: Biomarker-localization plots**: Compact visualization that shows markers (*i.e.*, squares, circles, triangles) to denote peaks present in the distinguishing potential (DP) curves of BM1, BM2 and BM3. This visualization allows for all biomarker identification methods results for all datasets be visualized in a single figure.
- **Mixed: BM1, BM2 and BM3 curves**: Single-result visualization that shows the DP curves for BM1, BM2 and BM3 in stacked panels. This visualization uses only the first selected result.
- **Biomarker localization report:** Prints biomarkers common to at least two biomarker identification methods for the selected results.

4. The toolkit

`bmtool` is a frontend for a function library developed for computational analysis of spectroscopy data. Actually, **bmtool** uses only partially the resources of this library. A complete list of functions can be found in the **`doc`** folder (use an Internet browser to open the **`index.html`** file).

5. External software

The provided dataset was generated *via* an import utility that can access the database where the dataset was originally stored. The database is managed by a database program that can import Pirouette .dat (Informetrix Inc.) files and automatically organize these spectra. This database software will be made available for download in the future.

References

[1] Trevisan, J. *et. al.* A mathematical framework for spectroscopy data analysis towards robust identification of biomarkers associated with chemical-induced alterations in cells (Submitted 19/Feb/2010).

[2] Hastie, T., Tibshirani, R. & Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, New York, USA (2001).