

# Introduction to Mfuzzgui

Matthias E. Futschik

Institute for Theoretical Biology, Humboldt-University

URL: <http://itb.biologie.hu-berlin.de/~futschik/software/R/Mfuzz>

and

Lokesh Kumar

Institute for Advanced Biosciences, Keio-University, Japan

April 3, 2007

## Contents

<b>1</b>	<b>Overview</b>	<b>1</b>
<b>2</b>	<b>Installation requirements</b>	<b>2</b>
<b>3</b>	<b>Functionality</b>	<b>2</b>
3.1	Loading data . . . . .	3
3.2	Pre-processing . . . . .	5
3.3	Clustering . . . . .	5
3.4	Cluster Analysis . . . . .	6
3.5	Visualisation . . . . .	6
<b>4</b>	<b>Help</b>	<b>7</b>

## 1 Overview

The package *Mfuzzgui* provides a graphical user interface for pre-processing, cluster analysis and visualization of gene expression microarray data using the functions of the *Mfuzz* package. The graphical interface is based on Tk widgets using the R-Tk interface developed by Peter Dalgaard. It also employs some pre-made widgets from the Bioconductor-package *tkWidgets* by Jianhua Zhang for the selection of objects or files to be loaded.

Mfuzzgui provides a convenient interface to most functions of the *Mfuzz* package without restriction of flexibility. An exception is the batch processes such as `partcoeff` and `cselection` routines which are used for parameter selection in fuzzy c-means clustering of microarray data. These routines are not included in *Mfuzzgui*. To select various parameters, the underlying *Mfuzz* routines may be applied.

Usage of Mfuzzgui assumes no existence of any `exprSet` objects and these objects can easily be constructed using the Mfuzzgui package itself from tab-delimited text files containing the gene expression data.

A brief introduction to the functionality of *Mfuzzgui* will be given in section 3. For more details about the computational approaches, please refer to the Mfuzz package documentation and the reference [1].

## 2 Installation requirements

Following software is required to run the Mfuzzgui-package:

- R ( $\geq 2.0.0$ ). For installation of R, refer to <http://www.r-project.org>.
- R-packages: Biobase, e1071 and tcltk, DynDoc and Tkwidgets. For installation of these add-on packages, refer to <http://cran.r-project.org>.
- Bioconductor packages: Biobase, Mfuzz ( $\geq 1.4.0$ ). Refer to <http://www.bioconductor.org> for installation.

If all requirements are fulfilled, the Mfuzzgui add-on R-package can be installed. To see how to install add-on R-packages on your computer system, start *R* and type in

```
>help(INSTALL)
```

Optionally, you may use the R-function *install.packages()*. Once the Mfuzzgui package is installed, you can load the package by

```
>library(Mfuzzgui)
```

## 3 Functionality

To start the graphical interface, type:

```
> Mfuzzgui()
```

A TclTk-Widget is launched (fig.1). The use of the interface is intuitive. The GUI is divided into five sections; the order of these sections reflects the order of a standard analysis and visualisation procedure for a microarray data set:

1. Loading of data set
2. Pre-processing of the data set

3. Clustering

4. Cluster Analysis

5. Visualisation of Results of Clustering and Cluster Analysis

Note that only one data set can be analysed at a time. To visualize the results of various clustering methods and cluster analysis, it is not necessary to save/export this data set and reload it.

The general structure of the GUI is straight-forward. Each button corresponds to or is very similar to the name of the corresponding function in the Mfuzz package. If additional arguments for the function are needed, an input window is launched. Note that not everything will be checked to be of correct type. While some checks are implemented (e.g. checks if the loaded data set has the correct class), most arguments remains unchecked before the underlying function is called (e.g. it will not be checked if the array index is a positive integer.) If errors are produced, the validity of the input arguments should be examined.

For details about the required types of arguments and corresponding functions, please refer to the help pages of the Mfuzz package. In the following section, a brief introduction to the functionality of Mfuzzgui is given. As an example, the dataset *yeast* can be loaded in the global environment by

```
> data(yeast)
```

or alternatively can be loaded for analysis using the Browse objects buttons.

### 3.1 Loading data

#### **exprSet object**

- Browse objects Import of an `exprSet` object of class `exprSet` from the global environment (`.GlobalEnv`). A window produced by the function `objectBrowser` of the `tkWidgets` package is generated. Note that, although several objects can be selected, only the first selected object will be loaded. A check is performed if the object to be loaded belongs to the correct classes.
- Browse files Loading of an R data set of class `exprSet` stored in a file. A window produced by the function `fileBrowser` of the `tkWidgets` package is generated. Note that, although several files can be selected, only the first selected file is loaded. A check is performed if the object to be loaded belongs to the correct classes.

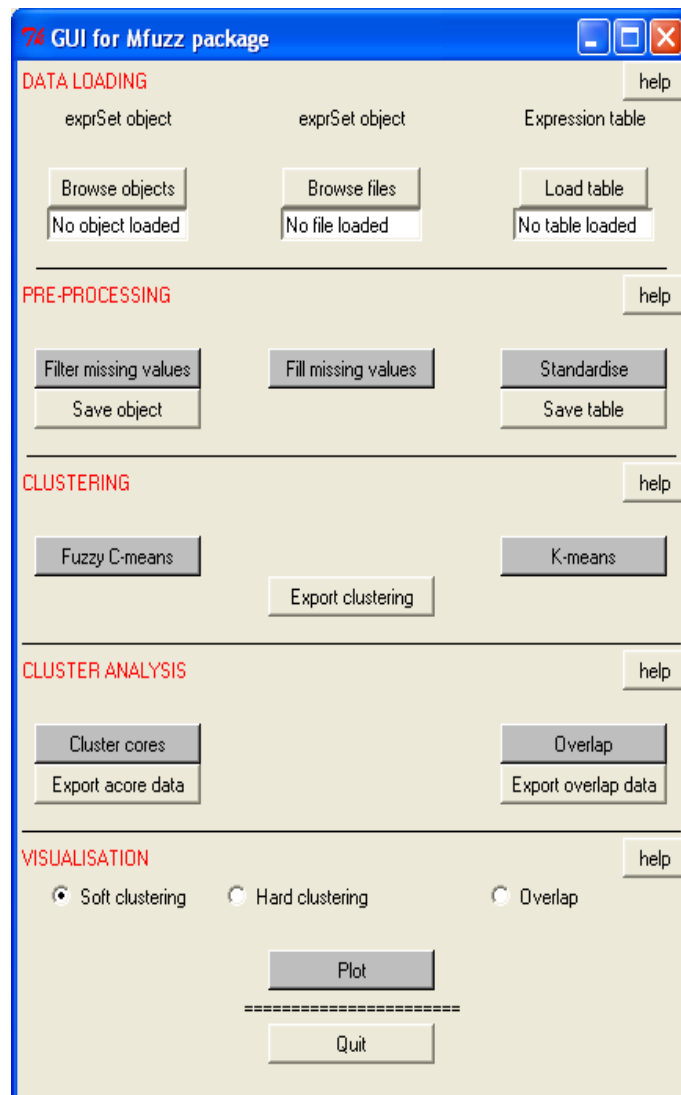


Figure 1: Screen-shot of Mfuzzgui

- **Load table** Construction and loading of an `exprSet` object from a tab-delimited data file. A window produced by the function `fileBrowser` of the `tkWidgets` package is generated. It is user's responsibility to use a correctly formatted file. The first row of the file contains sample labels and optionally, the second column can contain the time points. If the second row is used for the input the time, the first field in the second row must contain "Time". Similarly, the first column contains unique gene IDs and optionally second row can contain gene names. If the second row contains gene names, the second field in the first row must contain "Gene.Name". The rest of the file contains expression data. As example, two tables with expression data are provided. These examples can be view by inputing `data(yeast.table)` and `data(yeast.table2)` in the R console.

The name of the current `exprSet` object is shown in the text fields below the buttons once they are loaded into the current environment.

### 3.2 Pre-processing

- **Filter missing values:** This button performs two tasks. First it generates a pop-up window for the selection of threshold value for filtering the genes. Then after pressing the *OK* button of the pop-up window, it removes the genes which have more NAs (missing values) than the specified threshold value.
- **Fill missing values** Like before, this button again accomplishes two tasks. First it produces a pop-up window asking for the parameter values and method to be used for the replacement of missing values. After pressing the *OK* button, it replaces the missing values present in the partially processed data set using the user specified method and parameter values.
- **Standardise** Call of the function `standardise`. This step will standardise the gene expression values so that they have a mean value of zero and a standard deviation of one.
- **Save object** This button can be used to save the pre-processed object to some R data file. A window is generated to enable the user specify the file name where the object is to be stored.
- **Save table** This button can be used to save the pre-processed data in a tabular form in some disk file. A pop-up window is generated to facilitate user to specify the file name which can later be used to reload the data in the form of an `exprSet` object.

### 3.3 Clustering

- **Fuzzy C-means:** Call of function `mfuzz`. This button performs the fuzzy c-means clustering of the data. A new window is generated so that user can specify the parameter

values which are to be used in clustering the data.

- **K-means** Call of the function `kmeans2`. This button can be used for performing the standard k-means clustering. The parameter values can be specified in the pop-up window.
- **Export clustering** This button can be used to save the results of clustering into a text file. In the case of fuzzy C-means method, it stores the membership values of each gene for each cluster in a tabular format whereas for the k-means method, it stores the cluster vector containing the cluster numbers corresponding to each gene.

### 3.4 Cluster Analysis

- **Cluster cores** Call to the function `acore`. This function extracts genes forming the alpha cores of soft clusters. The minimum membership value can be specified in the pop-up window.
- **Overlap** Call to the function `overlap`. This function calculates the overlap of clusters produced by `mfuzz`.
- **Export acore data** This button exports the results produced by `Cluster cores` button in a text file.
- **Export overlap data** This button, like the previous one also saves the results generated by `Overlap` button into a text file. To see the details please refer to the documentation of `Mfuzz` package.

### 3.5 Visualisation

- **Soft clustering** Call to the function `mfuzz.plot`. This button can be used to visualise the results of soft clustering in the form of color-coded plots. Some parameters can be specified in the pop-up window.
- **Hard Clustering** Call to the function `kmeans2.plot`. This function visualises the clusters produced by `mfuzz`.
- **Overlap** Call to the function `overlap.plot`. This function visualises the cluster overlap produced by `Overlap`.

## 4 Help

Apart from the documentations provided with the *Mfuzzgui* package, there are *help* buttons in each section of the *Mfuzzgui* GUI. When one clicks on a help button, it gives a brief overview in a message box of what each button in the corresponding section does.

## References

- [1] M.E. Futschik and B. Charlisle, Noise robust clustering of gene expression time-course data, *Journal of Bioinformatics and Computational Biology*, Vol. 3, No. 4, 965-988, 2005
- [2] Cho RJ, Campbell MJ, Winzeler EA, Steinmetz L, Conway A, Wodicka L, Wolfsberg TG, Gabrielian AE, Landsman D, Lockhart DJ, Davis RW, A genome-wide transcriptional analysis of the mitotic cell cycle, *Mol Cell*, **2**:65–73, 1998
- [3] Bezdek JC, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981