Introduction to OLINgui

Matthias E. Futschik
Institute for Theoretical Biology, Humboldt-University
URL: *http://itb.biologie.hu-berlin.de/~futschik/software/R/OLIN*

September 22, 2004

Contents

1 Overview

The package*OLINgui* provides a graphical user interface for visualisation, normalisation and quality testing of two-channel microarray data. using the functions of the OLIN package.

The function *OLINgui* launches a graphical user interface for the main computational functions of the OLIN package. The graphical interface is based on Tk widgets using the R-TclTk interface by Peter Dalgaard. It also employs some pre-made widgets from the tkWidgets Bioconductor-package by Jianhua Zhang for the selection of objects/files to be loaded.

OLINgui provides a convenient interface to most functionalities of the OLIN package without restriction of options. An exception is the visualization, where default arguments for plotting are used. To fine-tune plots, the underlying plotting functions in the OLIN package can be applied.

The usage of OLINgui assumes existing marray objects for the batch of arrays to be analysed and normalised. To construct marray objects using a graphical interface, refer to the *marrayInput* package. The underlying functions `mxy2.plot` and optionally `olin`) require an additional list of X- and Y-coordinates of spots. This has to be done *by hand* yet. However, most functions do not need this list.

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

2 Installation requirements

Following software is required to run the OLINgui-package:

- R (> 1.9.0). For installation of R, refer to *http://www.r-project.org*.
- R-packages: methods, stat, locfit, tcltk. For installation of these add-on packages, refer to *http://cran.r-project.org*.
- Bioconductor packages: Biobase, marray, OLIN. Refer to *http://www.bioconductor.org* for installation.

If all requirements are fulfilled, the OLINgui 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 OLINgui package is installed, you can load the package by

```
> library(OLINgui)
```

3 Functionality

To start the graphical interface, type:

```
> OLINgui()
```

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

1. Loading of data set
2. Visualisation for detection of artifacts
3. Application of statistical tests to identify experimental bias
4. Normalisation and scaling of data set

Note that only one data set at a time can be analysed or normalised. To visualise or test the normalized data set, it is necessary to save/export this data set and reload it.

The general structure of the GUI is straight-forward. Each button corresponds to one or two functions in the OLIN package. If additional arguments for the functions are needed, a window is launched and arguments can be selected. Note that, while some checks are implemented (e.g. checks if the loaded data set has the correct class),not everything is checked 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 OLIN package. In the following section, a brief introduction to the functionality of OLINgui is given. As an example, the datasets *sw* and *sw.xy* can be loaded in the global environment by

```
> data(sw)
> data(sw.xy)
```

and the loaded for analysis using the Browse objects buttons.

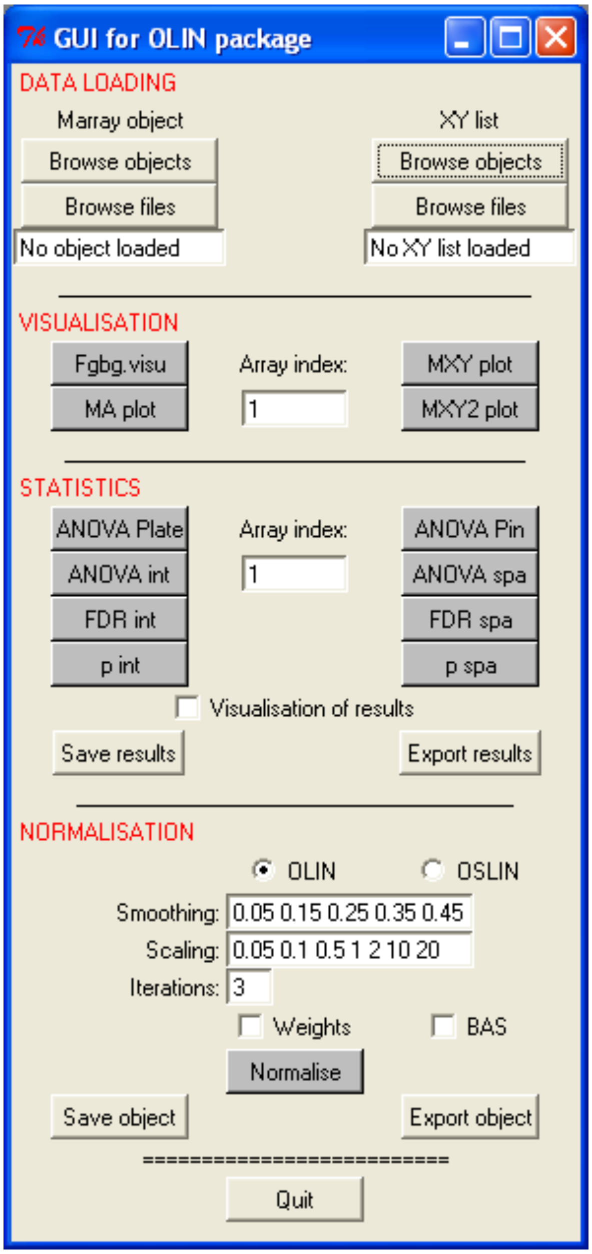

Figure 1: Screen-shot of OLINgui

3.1 Loading data

Marray object

- $\boxed{\text{Browse objects}}$: Loading of an Marray object of class *marrayRaw* or *marrayNorm* 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 is loaded. A check is performed if the object to be loaded belongs to the correct classes.

- $\boxed{\text{Browse file}}$: Loading of an R data set of class *marrayRaw* or *marrayNorm* 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.

XY list

- $\boxed{\text{Browse objects}}$: Loading of an R object of type *list* 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 is loaded. A check is performed if the object is of type *list* with name attributes "X" and "Y". No check is performed if the dimensions of the list elements are correct.

- $\boxed{\text{Browse file}}$: Loading of an R object of type *list* stored in a file. A window produced by the function `fileBrowser` of the tkWidgets package is generated. Note that, although several objects can be selected, only the first selected object is loaded. A check is performed if the object is of type *list* with name attributes "X" and "Y". No check is performed if the dimensions of the list elements are correct.

The name of the marray object and XY list is shown in the text fields below the buttons.

3.2 Visualisation

- *Array index:* Selection of the arrays to be visualised. If several arrays to be visualised, the indeces are to be space delimited. If no array index is given, all arrays of the data set are to be visualised.

- $\boxed{\text{Fgbg.visu:}}$ Call of the function `fgbg.visu`. Visualisation of spatial distribution of foreground and background fluorescence intensities in both channels.

- $\boxed{\text{MA plot}}$: Produces MA-plots that display the logged fold change ($M = log_2(Ch2) - log_2 Ch1$) with respect to the average logged spot intensity ($A = 0.5(log_2(Ch1) + log_2 Ch2)$).

- $\boxed{\text{MXY plot}}$: Call of the function `mxy.plot`. Visualisation of spatial distribution of logged fold change (M) across the array. The row and column indeces are used as proxies for the spatial spot coordinates.

- $\boxed{\text{MXY2 plot}}$: Call of the function `myx2.plot`. Visualisation of spatial distribution of logged fold change (M) across the array. In contrast to `mxy.plot`, function `mxy2.plot` produces MXY-plots based on spatial spot coordinates as stored in the XY-list. Therefore, this function requires that a XY-list has been loaded.

3.3 Statistics

- *Array index:* Selection of the arrays to be analysed. If several arrays to be analysed, the indeces are to be space delimited. If no array index is given, all arrays of the data set are to be analysed.

- $\boxed{\text{ANOVA Plate}}$: Call of the function `anovaplate`. An one-factorial ANOVA assessing plate-dependent bias is performed. If *Visualisation or results* is ticked, a window with the results is shown.

- $\boxed{\text{ANOVA Pin}}$: Call of the function `anovapin`. An one-factorial ANOVA assessing pin-dependent bias is performed. If *Visualisation or results* is ticked, a window with the results is shown.

- $\boxed{\text{ANOVA int}}$: Call of the function `anovaint`. An one-factorial ANOVA assessing intensity-dependent bias is performed. The predictor variable is A; the response variable is M. As function argument an integer has to be chosen. This determines the number of intervals along the A-axis and, thus, the number of factors for the ANOVA. If *Visualisation or results* is ticked, a window with the results is shown.

- $\boxed{\text{ANOVA spa}}$: Call of the function `anovaspatial`. An one-factorial ANOVA assessing spatial bias is performed. The array is divided in a number of blocks determining the factors for the ANOVA. The predictor variable is the index of the spatial blocks; the response variable is M. The number of blocks can be selected by the input arguments determining the number of intervals in the X and Y direction. If *Visualisation or results* is ticked, a window with the results is shown. Additionally, a plot is produced showing the significance of blocks based on the ANOVA model.

- $\boxed{\text{FDR int}}$: Call of the function `fdr.int`. This function assesses the intensity-dependent dye bias using a permutation test. The test statistics is the average value of $\bar{M}$ in a spot intensity neighourhood of chosen size. The number of permutations determines how often the intensity order of spots is permutated for the calculation of the empirical background distribution. For averaging, *mean* or *median* can be chosen. If *Visualisation or results* is ticked, a plot is produced showing the false discovery rates for biased intensity neighbourhoods. Green (red) dots indicate the false discovery rate for negative (positive) derivations of the test statistics. Low false discovery rates indicate intensity-dependent bias. Note, however, that the scale of the y-axis is $-log_{10}(\text{fdr})$.

- $\boxed{\text{FDR spa}}$: Call of the function `fdr.spatial`.The function assesses location-dependent dye bias using a permutation test. The number of permutations determines how often the spatial location of spots on an array is permutated for the calculation of the empirical background distribution. If *Visualisation or results* is ticked, a MXY plot is generated showing the false discovery rate for $\bar{M}$ of spatial spot neighbourhoods.

- $\boxed{\text{p int}}$: Call of the function `p.int`. This function assesses intensity-dependent dye bias using a randomisation test. Similarly to *fdr.int*, the average logged fold change ($\bar{M}$ in a spot intensity neighourhood is compared to a empirical background distribution based on randomised intensity order of spots. The number of samples determines how many random samples (with replacement) are generated for the construction of an empirical background distribution. The significance of $\bar{M}$ is based on Fisher randomisation. Several methods for p-value adjustment can be chosen. If *Visualisation or results* is ticked, the significance is plotted along the spot intensity axis. Green (red) dots indicate the false discovery rate for negative (positive) derivations of the test statistics. Low false discovery rates indicate intensity-dependent bias. Note, however, that the scale of the y-axis is $-log_{10}$(fdr).

- $\boxed{\text{p spa}}$: Call of the function `p.spatial`. This function assesses location-dependent dye bias using a randomisation test.The number of samples determines how many random samples (with replacement) are generated for the construction of an empirical background distribution. The significance of $\bar{M}$ is based on Fisher randomisation. Several methods for p-value adjustment can be chosen. If *Visualisation or results* is ticked, a MXY plot is generated showing the significance of $\bar{M}$ for spatial spot neighbourhoods.

- *Visualisation or results*: Results are visualised.

- $\boxed{\text{Save results}}$: The results are stored in a list and saved in a file. Each list elements corresponds to the output of the underlying functions that has been used for the statistical analysis.

- $\boxed{\text{Export results}}$: The results are stored in a list and exported to the global environment. Each list elements corresponds to the output of the underlying functions that has been used for the statistical analysis.

3.4 Normalisation

- *OLIN - OSLIN radiobuttons*: Selection which normalisation scheme should be applied

- *Smooting*: Vector of smoothing parameter to be tested for optimised normalisation.

- *Scaling*: Vector of scaling parameter to be tested for optimised normalisation. The scaling parameter determine the amount of smoothing in Y-direction compared to smoothing in X-direction.

- *Iterations*: number of iterations in the OLIN/OSLIN procedure. Three are usually sufficient.

- *Weights*: Weighting of spots for local regression. If checkbutton is ticked, spots are weighted by the weights stored in maW slot of the marray object.

- *BAS*: Subsequent between-array normalisation (scaling). If checkbutton is ticked, a between-array normalisation is applied. Three methods can be chosen from: *var*- arrays are scaled to have the same variance of M, *mad* - arrays are scaled to have the same median absolute deviation of M and *qq* - arrays are scaled using quantile normalisation.

- $\boxed{\text{Normalise}}$: Call of the function `olin`.

- $\boxed{\text{Save object}}$: The normalised batch of arrays will be stored as maNorm object and saved in a file.
- $\boxed{\text{Export object}}$: The normalised batch of arrays will be exported to the global environment.

To visually inspect and stastically assess the results of the normalisation, the data has to be exported (or saved) and reloaded.

References

[1] M.Futschik and T. Crompton (2004) Model selection and efficiency testing for normalisation of cDNA microarray data, *Genome Biology*, 5:R60