

FSDA: A MATLAB toolbox for robust analysis and interactive data exploration

Marco Riani

Dipartimento di Economia, Università di Parma, Italy*,

Domenico Perrotta

European Commission, Joint Research Centre, Ispra, Italy[†],

and Francesca Torti

Facoltà di Statistica, Università di Milano Bicocca, Italy[‡]

October 31, 2011

Abstract

We present the FSDA (Forward Search for Data Analysis) toolbox, a new software library that extends MATLAB and its Statistics Toolbox to support a robust and efficient analysis of complex datasets, affected by different sources of heterogeneity.

As the name of the library indicates, the project was born around the Forward Search approach, but it has evolved to include the main traditional robust multivariate and regression techniques, including LMS, LTS, MCD, MVE, MM and S estimation.

To address problems where data deviate from typical model assumptions, tools are available for robust data transformation and robust model selection.

When different views of the data are available, e.g. a scatterplot of units and a plot of distances of such units from a fitted model, FSDA links such views and offers the possibility to interact with them. For example, selections of objects in a plot are highlighted in the other plots. This considerably simplifies the exploration of the data in view of extracting information and detecting patterns.

We show the potential of the FSDA in chemometrics using data from chemical and pharmaceutical problems, where the presence of outliers, multiple groups, deviations from normality and other complex structures are not exceptional circumstances.

Keywords: forward search; robust data analysis; robust transformations; exploratory data analysis; data visualization; graphics; statistical programming.

*e-mail: mriani@unipr.it

[†]e-mail: domenico.perrotta@ec.europa.eu

[‡]e-mail: francesca.torti@unimib.it

1 Introduction

With this paper we present to the chemometrics community easy to use software which drastically simplifies the conduct of rigorous multivariate statistical analyses, which are necessary to address properly datasets and problems that we introduce in Section 2.

The software is the FSDA (**F**orward **S**earch for **D**ata **A**nalysis) toolbox¹, that extends MATLAB² and its Statistics Toolbox to support a robust and efficient analysis of complex datasets, affected by different sources of heterogeneity. When data deviate from typical model assumptions, e.g. multivariate normality, tools are available to automatically estimate the most appropriate transformation type, in a way that is robust to the presence of outliers. For the same purpose, model selection tools are also available. As the name of the toolbox indicates, the project was born around the Forward Search approach, for which details are given in the books of Atkinson and Riani (2000) and Atkinson, Riani, and Cerioli (2004) and in the recent discussion paper of Atkinson, Riani, and Cerioli (2010). However, the project has evolved to include the main traditional robust multivariate and regression techniques, including LMS, LTS, MCD, MVE, MM and S estimation. There are several excellent introductions to robust statistics with applications of a chemometric nature e.g., just to cite a few, Rousseeuw (1991), Rousseeuw, Debruyne, Engelen, and Hubert (2006), Daszykowski, Kaczmarek, Heyden, and Walczak (2007) and Filzmoser, Serneels, Maronna, and Espen (2009). Therefore, this paper will describe the use of the main FSDA functions, but will not elaborate on the statistical properties of the implemented methods.

The choice of a commercial platform such as MATLAB is sometimes a source of criticism, especially from the part of the statistical community gravitating around R³. On the other hand, while it is widely recognized that users should have the possibility to interact with the data represented in the static or dynamic plots of traditional exploratory data analysis, to our knowledge MATLAB is currently the only environment with *built-in*⁴ interactive data exploration features. In the FSDA we fully exploit and even extend such features, by connecting graphs where the objects represented are logically connected but have no variable in common (e.g. a scatterplot of units and a plot of distances of such units from a fitted model).

Other robust statistics software proposed to the chemometrics community in recent years are LIBRA (Verboven and Hubert (2005) and Verboven and Hubert (2010)) and TOMCAT (Daszykowski, Serneels, Kaczmarek, Espen, Croux, and Walczak (2007)), both written in MATLAB. For the R environment we mention CHEMOMETRICS and RRCOV, discussed very recently by Filzmoser and Todorov (2011). TOMCAT addresses methodologies that are not covered by FSDA, being more focused on problems where the number of variables exceeds the number of observations. LIBRA and FSDA have a wider scope but, while the first is mainly centred on LTS and MCD estimation, the latter addresses a number of other common robust methods.

The focus of the paper is on the distinctive features of FSDA, such as (i) the attention given to the documentation system fully integrated with the MATLAB help, (ii) the care

¹FSDA is copyright of the European Commission and the University of Parma. It is protected under European Union Public Licence (EUPL), which is a free software license granting recipients rights to modify and redistribute the code. The logo, visible in Figure 2, has been trademarked.

²MATLAB is ©1984 - 2011 The MathWorks, Inc. See <http://www.mathworks.com>.

³The R project for statistical computing: <http://www.r-project.org/>.

⁴The R community has developed some interfaces to *external* interactive graphics systems, such as GGobi (<http://www.ggobi.org>) and the Java library iPlots (<http://www.iplots.org>).

given to computational performance, (iii) the tools for dynamic interaction with a number of exploratory plots, (iv) robust model selection tools and (v) automatic determination of transformations in regression and multivariate analysis. Above all, (vi) a comprehensive set of robust multivariate and regression methods written in a unified framework.

After introducing some common issues in chemometric applications (Section 2), we describe the main architectural features of FSDA, i.e. the platform requirements in Section 3 and some computational aspects in Section 4. These are complemented by Section 7, that addresses scalability issues that we plan to address in the near future in view of applications to large datasets. The core of the paper is Section 6, describing the use of the main statistical functions in FSDA. In particular, Section 6.4 introduces the general interactive paradigms that we are progressively extending to all FSDA graphical outputs, beyond the Forward Search plots. Since it is rare that to familiarise with a new software instrument and explore all its features one uses his/her own data, Section 5 describes how to use the datasets incorporated in FSDA. As is customary, a section of conclusions closes the paper.

2 Issues in bio-chemical data

Datasets coming from chemical and pharmaceutical problems can be very complex. Consider for example a chemical analysis of serum samples taken from a number of individuals following a standardized bioanalytical format. There are components in a chemical assay which are rather standard (e.g. reagents), but others necessarily depend on the specific experimental and environmental conditions of the assay. In addition, to reduce subjectivity, the analysis of each sample is often replicated by different operators over different days.

This gives rise to complex datasets possibly affected by outliers and observations from multiple populations. In addition, it is rather typical that one or more variables needs to be transformed in order to comply with the common model assumptions of a statistical analysis, e.g. multivariate normality.

Figure 1 shows two examples of such datasets coming from an international pharmaceutical company⁵, now included in the FSDA data repository. In both cases there are 50 units (chemical measurements on serum samples taken from multiple groups of 50 individuals). The first dataset has 17 variables and the second 12, one variable for each replicate of measurements on the individuals. We will call the two datasets DS17 and DS12. To make the two plots intelligible we report only 9 pairs of variables. Moreover, we highlight data from potentially different populations (due to different experimental settings of different groups of individuals) with different symbols. We observe that in DS17 the variables are very correlated, while in DS12 the correlation is not at all obvious. Moreover in DS12 the three groups, corresponding to three different plates where the serum samples were analyzed, are quite distinct and this has to be taken into account. On the other hand, in DS17 it is not clear whether the two overlapping groups, corresponding to samples from male and female individuals, have to be treated as a unique population.

The regulatory pharmaceutical authorities recommend to statistically validate a given chemical assay, to be sure that it will reliably detect a given factor of interest in real clinical trials and tests. Typically, the statistical validation includes the determination of a *cut off point of the assay*, which is the multivariate (or univariate) threshold above which a unit

⁵Merck Serono Non-Clinical Development, Biomedical Research Institute RBM S.p.A., Collioretto Giacosa, Italy.

(or an individual measurement) is considered positive. Of course the determination of the cut off point must be based on data coming from samples of individuals not exposed to the agent or drug responsible for the factor to be detected (negative control samples).

Clearly, if a dataset includes multiple groups as in DS12, a single cut off point is inadequate. Thus, it is necessary to understand whether we are in the presence of a single population or there are multiple groups. Sometimes, as in DS12, these groups can be easily explained by the controlled experimental conditions. Other times multiple groups and/or outliers that may distort the estimation of the cut off point appear unexpectedly. Often, the presence of multiple distinct groups and outliers in supposedly negative control samples, may be an indication that the chemical assay should be re-designed, and the statistical analysis postponed to when better data become available.

The statistical recommendations of regulatory authorities for the validation phase often try to compromise between optimal and simple (in the sense of easy to implement and controllable by non specialists) statistics. For example Shankar and al. (2008), in discussing the validation of immunoassay used for anti-drug antibodies detection (DS17 and DS12 refer to such problem), propose to treat data separately, variable by variable, using a univariate perspective. Coherently with this choice, they propose the use of the traditional boxplots to detect anomalous values, possibly after logarithmic transformation in case of non normality of the data, which is checked using well known (but sensitive to outliers) tests such as the Shapiro-Wilks. As result of this approach, we have for each variable a different number of measurements declared as outliers. The supposedly normal clean data are successively combined into a single measurement using a simple arithmetic mean to determine the cut off point. Otherwise, if some of the variables are not normally distributed, an appropriate empirical percentile is proposed.

Clearly, in doing so one may lose relevant information, such as the high correlation between the variables of DS17 and the group structure of DS12, which require multivariate statistical tools. In addition, it is well known that in the presence of several outliers

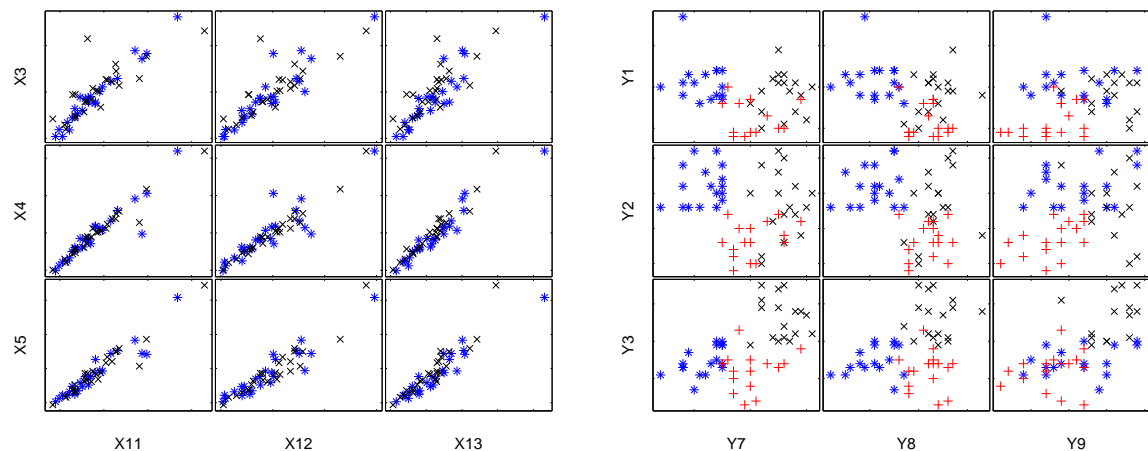


Figure 1: A partial view of two complex biochemical datasets, both with 50 units. DS17 (left panel) has 17 variables. The two symbols distinguish between male and female serum samples. DS12 (right panel) has 12 variables. In this case the serum samples were analyzed using three different plates identified by different symbols. The 9 pairs of variables chosen for the two plots show patterns which are also characteristic of the remaining undisplayed variable pairs.

the phenomenon of masking may completely invalidate the statistical estimates (e.g. the normality test itself) and the final cut off points. Shankar and al. (2008) seem to be well aware of these problems when they write (p. 1269, Section 2.2, first paragraph) “... These statistical computations can be applied *with the help of user-friendly commercial software*, without the need of a formal training in statistics. However, the assistance of a statistician for planning validation experiments and analyses of data can lead to the application of *more rigorous and elegant statistics* than suggested herein. ...”.

In the following sections we try to demonstrate that the FSDA toolbox meets the two general requirements mentioned by Shankar and al. (2008): on the one hand it offers the possibility to apply rigorous robust multivariate statistical tools and, on the other hand, it makes this task simple by means of user friendly and well documented tools.

3 Architecture

FSDA works from the release R2009b of MATLAB⁶ and uses the Statistics toolbox. Our software has been tested on Microsoft as well as Unix (Linux and MacOSX) platforms. Moreover all routines, with the exception of the interactive graphical tools, seem to work without major changes in SCILAB and OCTAVE.

We have made use of few third parties general utility functions. In addition, our implementation of the traditional robust estimators (S, MM, MVE and MCD) follows the lines of the R code developed during the years by many authors⁷.

On MS Windows, a setup program installs the code, updates the MATLAB search path variable and automatically opens inside the editor two files (`examples_multivariate.m` and `examples_regression.m`) containing a series of examples of analysis of regression and multivariate datasets organized in cells that can be executed one by one by the user for an overview of the FSDA functions and options. On Unix platforms the software has to be installed manually from a compressed tar-file. The setup software and the compressed tar-file can be downloaded from the FSDA web-site, <http://fsda.jrc.ec.europa.eu>, also accessible from <http://www.riani.it/MATLAB>.

Three graphical user interfaces (GUI) accessible from the standard ‘Start’ button of MATLAB, allow one to explore the main interactive features of FSDA on plots generated by running the different robust estimators available on regression and multivariate problems, including transformations (Figure 2). The GUIs are also accessible as m-files in the `\examples` subfolder. Other didactic material, also accessible from the ‘Start’ button as `FSDADemos`, is available in the form of movies (with audio) that can be run in a browser connected to the Internet.

The FSDA has a comprehensive documentation system. Of course the head of each m-function describes the function purpose, the input-output parameters, the bibliographic references, possible function dependencies, any third party acknowledgment and some self contained examples of use. In addition, even more extensive documentation on each FSDA function can be obtained directly from where the user is working using the standard ‘Func-

⁶To work with previous releases adjustments are necessary, for example on the syntax for ignoring unused function inputs and outputs (e.g. the tilde symbol in `[~, idx] = sort(A)` should be replaced with `[unusedvar, idx] = sort(A)`).

⁷See the R library `robustbase`, <http://robustbase.r-forge.r-project.org/> and the website <http://www.econ.kuleuven.be/public/NDBAE06/programs/>.

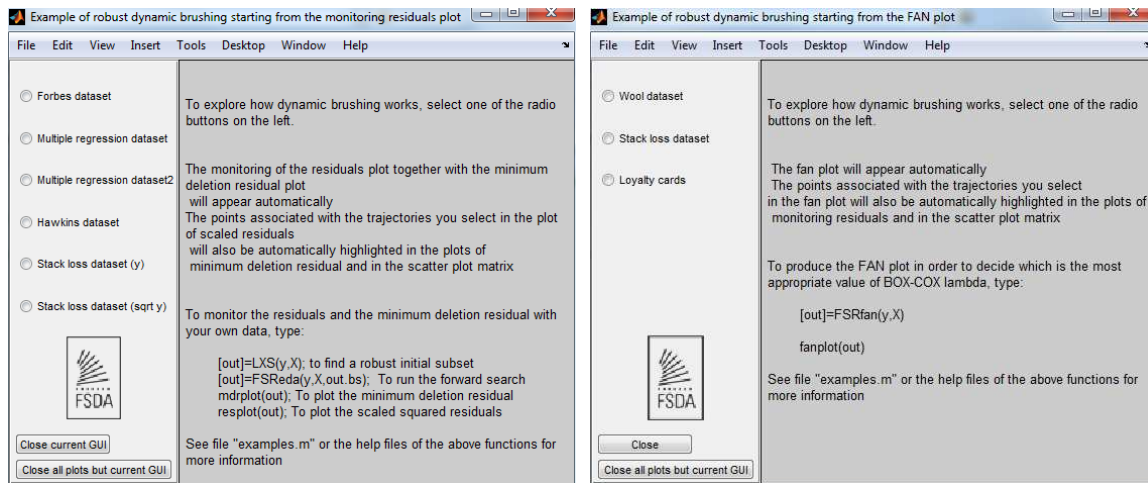


Figure 2: Graphical User Interfaces, accessible from the standard 'Start' button of MATLAB, which demonstrate how FSDA can be used to analyze regression problems (left) or to find the best transformation for a given dataset (right).

tion Browser' of MATLAB, which is activated by typing `shift+F1`. For example, Figure 3 shows what the user gets on typing "robust regression" from the help menu and selecting the entry "LXS (FSDA)". The same extensive documentation is also browsable and searchable in the standard help pages of MATLAB, accessible from the usual 'Help' menu. In the help pages we have also integrated both an informal and a technical introduction to robust statistics and the Forward Search (see left panel of Figure 4) and a detailed description of a collection of popular regression and multivariate datasets. The datasets are stored under subfolder `\datasets`.

4 Computational performance aspects

Computational performance is very important if there are big collections of datasets to analyse or extensive benchmarks to run for a standardized empirical assessment of different statistical methods. With this in mind, in developing FSDA, we carefully address different optimization aspects.

- As a standard practice, the execution times of the key functions are profiled to identify critical code segments. Loops are possibly replaced with matrix or array operations and convenient data types are chosen for large data structures. For some demanding combinatorial functions (e.g. the binomial coefficient), FSDA chooses automatically the algorithm to run on the basis of a trade-off between computation time and storage requirements.
- Almost all algorithms of robust statistics spend a lot of computation time on re-sampling and computing estimates on the subsamples. FSDA uses an efficient random sample generation strategy, implemented by function `subsets.m`, which dramatically drops the execution time of the robust algorithms. For example, in a dataset of size $n = 200$ with 5 explanatory variables ($p = 5$), to compute the estimate of the

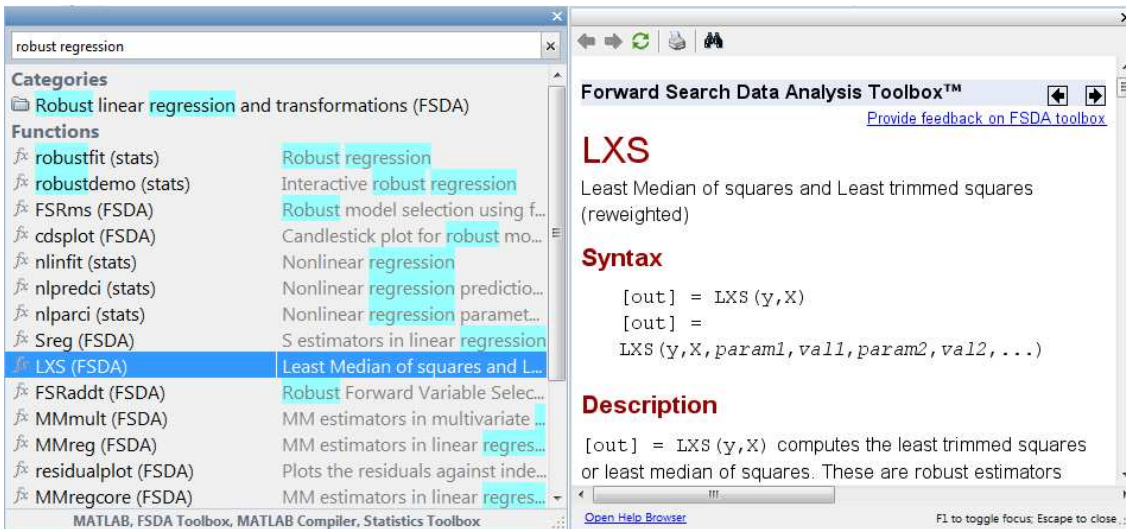


Figure 3: FSDA documentation is integrated in the standard Function Browser of MATLAB. Here, a search for “robust regression” has retrieved items from both the Statistics (stats) and FSDA toolboxes (left plot). Among such items, if the user selects the FSDA function “LXS”, its documentation is automatically displayed in a separate window (right plot).

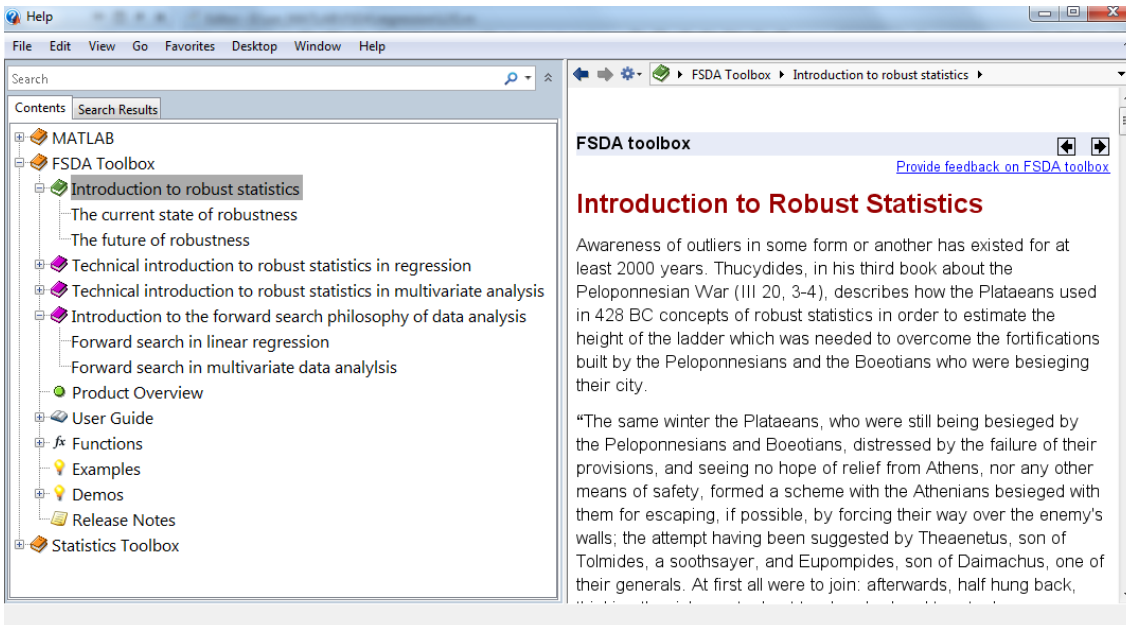


Figure 4: FSDA documentation is integrated in the standard Help Menu of MATLAB. Here, the user is browsing the introductory section on robust statistics.

vector of regression coefficients using Least Trimmed Squares ($\hat{\beta}_{LTS}$) after extracting 20,000 different subsets, it takes less than two seconds (on a 2.8 GHz Intel CPU).

Our re-sampling strategy critically depends on combinatorial functions such as the binomial coefficient and the lexicographic generation of all combinations. Traditionally combinatorial objects are built using recursive or iterative algorithms. We have re-designed these algorithms to exploit the way MATLAB stores and manipulates matrices, with a sometimes extraordinary computational gain. For example, our lexicographic generation of the matrix of all combinations (`combsFS.m`), which we populate by visiting portions of columns rather than sequentially one row after the other, is about 75 times faster than the native MATLAB implementation (`combs.m` contained in `nchoosek.m`). Moreover, given that our implementation uses the storage resources more parsimoniously, `combsFS.m` works well for combinations of n and p that break `nchoosek` due to memory fault.

A last combinatorial function intensively used by robust procedures is the random permutation of the elements of a vector⁸. Function `shuffling.m` in FSDA, uses the shuffling algorithm by Knuth (1997), p. 145 - 146, which is based on a much older method of Fisher and Yates (1963). This turned out to be more efficient than the native MATLAB solution, based on a complete sorting of the vector elements (function `randperm.m`).

In the R community it is quite common to accelerate code execution by compiling critical segments developed in C or C++ and linking them to an R program. MATLAB offers a similar instrument, consisting in compiling m-functions into dynamically linked binary mex-files. However, to distribute and maintain CPU dependent code is not practical and requires recompilation whenever the sources are modified. For these reasons, FSDA is exclusively based on MATLAB code. Nonetheless, users are free to compile specific portions of code to address specific demanding needs.

5 Datasets

FSDA includes a rich collection of popular datasets (currently about 50) that can be used for familiarize with the toolbox functions and replicate examples of the robust literature. The collection includes the multivariate and regression datasets in the books of Atkinson and Riani (2000) and Atkinson, Riani, and Cerioli (2004), several datasets of the book of Maronna, Martin, and Yohai (2006) and a series of other datasets used in the robust literature.

The datasets are provided in different formats.

- Tab separated data file (`.txt` file). In this case, to load a data set into the MATLAB workspace, the user has to type: `load datasetname.txt`. This will load the original txt file into a standard data matrix.
- Data structure, contained inside a `.mat` file, which is loaded in the workspace by typing `load datasetname.mat`. This complements the data matrix (stored in

⁸For example, if a subset is thought to be singular, it is increased taking one unit from a random permutation of the remaining units, until it becomes non singular.

datasetname . data) with the variable and observation names (datasetname . colnames and datasetname . rownames) and data specific notes (datasetname . notes).

- Cell and Dataset structures, which can be loaded by typing `load datasetnameC` and `load datasetnameD` respectively.

6 Functions

FSDA contains four main categories of functions: robust linear regression and transformations, robust multivariate analysis and transformations, robust model selection, and dynamic statistical visualization. Following the classical documentation style of MATLAB, all FSDA functions can be browsed alphabetically and by category in the FSDA ‘Function Reference’ help section, of which Figure 5 shows the categories links (left panel) and the part reached by clicking on the ‘Robust Model Selection’ link (right panel).

This section describes the implemented methods very briefly, given that a more formal introduction is available in the FSDA help pages titled ‘Technical introduction to . . .’ and, of course, in the specialized literature. In this section we give more details to the functions and methods which give insights on the biopharmaceutical problems discussed in Shankar and al. (2008). We use datasets DS17 and DS12 as examples to illustrate the use and the potential benefits of FSDA in the chemometrics world.

6.1 Input-output parameters

For many functions the set of input-output parameters is so rich that it is not convenient nor possible to treat them comprehensively here. Details on a specific option can be however retrieved from our FSDA documentation by typing `docsearch('option_name')` in the MATLAB Command Window. The order with which the optional input parameters are set does not matter.

Typically, even a well trained practitioner will make use of few of the optional parameters available. On the other hand, a researcher will have the possibility to experiment with

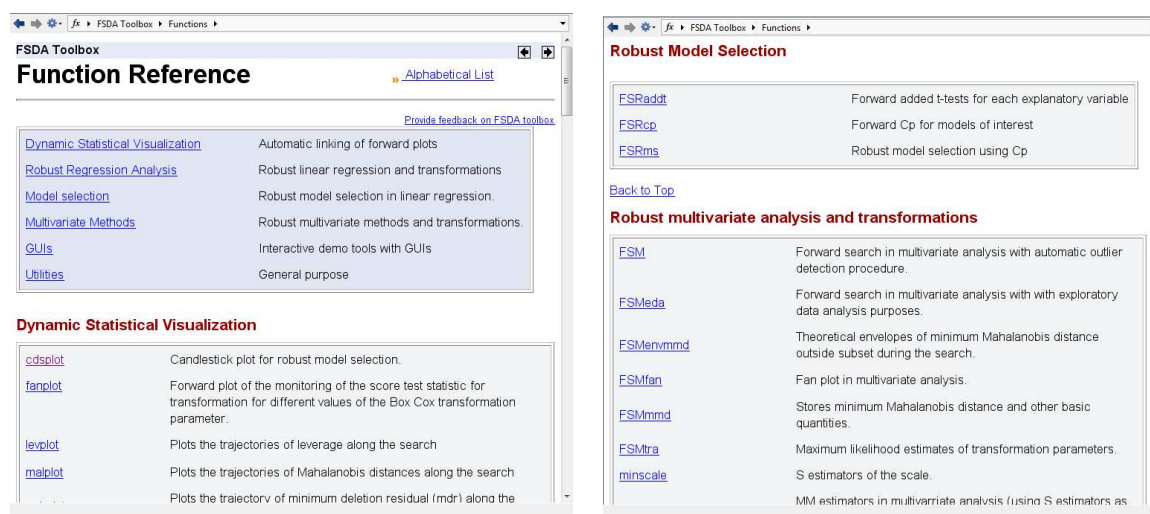


Figure 5: The FSDA functions are organized in thematic categories. The two snapshots in the figure show how these are documented.

many internal variables controlled by optional parameters, without being forced to touch the source codes. The use of very flexible and thus elaborated options is simplified by the adoption of data types of increasing complexity. For example, option `databrush` controls the interactive brushing features of our dynamic statistical visualization tools (see Section 6.4), but a user may simply want to plot their traditional default version. In this case, it will not be necessary to instantiate option `databrush`. To make a single selection, it will be set to a scalar (e.g. `databrush=1`). To make an indefinite number of selections, it will have to be a structure with a `persist` field set to 'on' (`databrush.persist='on'`). In general, when an option becomes a structure, the list of possible fields will be automatically set to default values and the user will only have to set what is of interest.

The output parameters are dealt with by the same principle: when a function generates a lot of information, this is organized in an output structure so that the user can extract only fields of major interest. As an example see the left panel of Figure 6.

Finally, at the end of each function help page, as is customary in the MATLAB documentation system, we included a series of code segments which, once selected, can be immediately executed by typing F9. In this way the user can see in real time what he finds in the documentation. See for example right panel of Figure 6.

6.2 Robust multivariate analysis and transformations.

The normal distribution, perhaps following data transformation, has a central place in the analysis of multivariate data. Mahalanobis distances provide the standard test for outliers in such data. However, it is well known that the estimates of the mean and covariance matrix found by using all the data are extremely sensitive to the presence of outliers. When there are many outliers the parameter estimates may be so distorted that the outliers are 'masked' and the Mahalanobis distances fail to reveal any outliers, or indicate as outlying observations that are not in fact so. Accordingly, several researchers have suggested the use of robust parameter estimates for the mean and the covariance matrix in the calculation of the distances. For example, Rousseeuw and van Zomeren (1990) used minimum

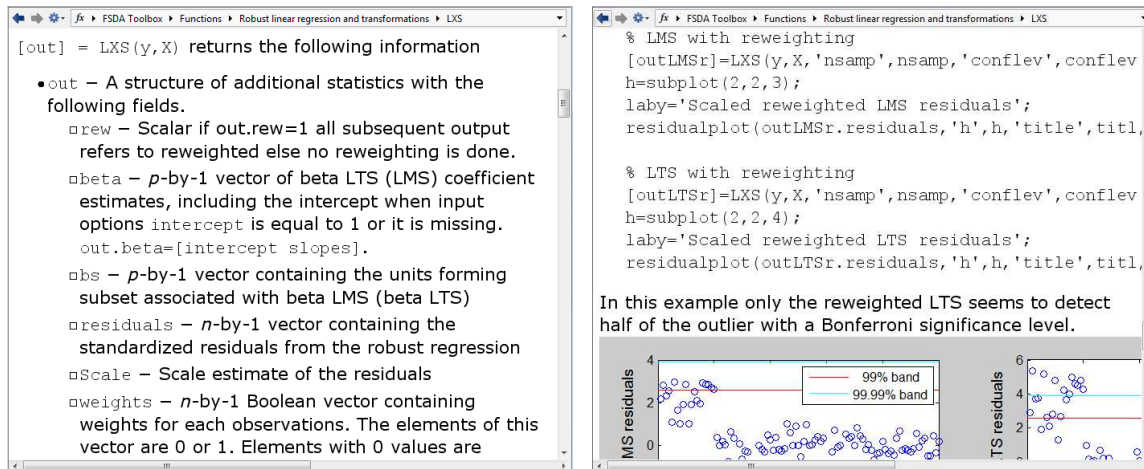


Figure 6: Example of a function documentation (`LXS`). The left panel shows some fields of the `out` structure returned by `LXS`. The right panel shows code segments that, once selected, can be executed by typing F9, to reproduce the type of output (graphical in this case) documented in the help page.

volume ellipsoid (MVE) estimators of both parameters in the calculation of Mahalanobis distances. More recent work such as Pison, Van Aelst, and Willems (2002) or Hardin and Rocke (2005) uses the minimum covariance determinant (MCD) estimator discussed in Rousseeuw and Van Driessen (1999).

In chemometrics, which has continuously to do with repeated applications of a statistical procedure to multiple samples, the remark of Cook and Hawkins (1990) on the fact that the procedure of Rousseeuw and van Zomeren (1990) may find “outliers everywhere” is of high relevance. The implication is that the size of the outlier test may be very much larger than the nominal 5% or 1%. In fact, many of these methods are designed to test whether individual observations are outlying. As do Becker and Gather (1999), we, however, stress the importance of multiple outlier testing and focus on simultaneous tests of outlyingness. Therefore, in FSDA we develop methods that are mainly intended, when the samples are multivariate normal, to find outliers in $\alpha\%$ of the datasets.

Given the above, in FSDA we have included the following bivariate/multivariate procedures.

6.2.1 unibiv

`unibiv` implements robust univariate and bivariate analysis. Robust bivariate ellipses (together with univariate boxplots) are constructed for each pair of variables and it is possible to analyze the units falling outside these robust bivariate contours. Figure 7 shows the application of the function to the variables X3 and X13 of dataset D17. The plot is generated by calling function

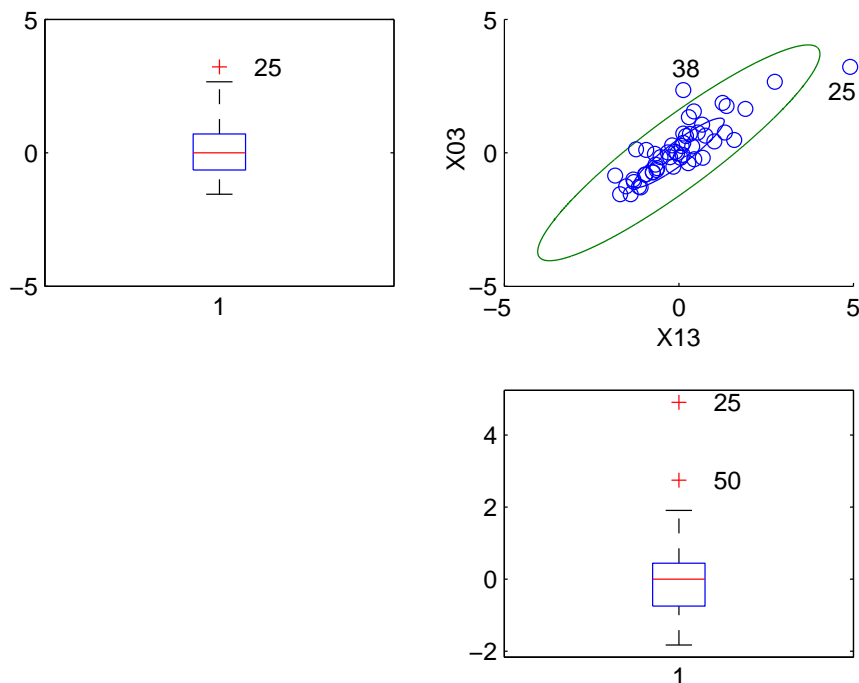


Figure 7: Robust univariate boxplots and bivariate ellipse for variables X3 and X13 of dataset D17, generated using function `unibiv.m`.

```
[out] = unibiv([X(:,3) X(:,13)], ...
               'rf',0.99,'plots',1,'textlab',1);
```

where the optional parameter `rf` specifies the confidence level of the robust bivariate ellipse (0.99 means that the ellipse leaves outside 1% of the values under normal conditions), while `plots` and `textlab` are set to 1 respectively to request the generation of the plot and to display labels associated with the units which are univariate outliers or which are outside the confidence levels of the contours. In the example, `unibiv` has used a robust estimate of correlation and dispersion of the two variables. Other possibilities are available by changing the optional parameter `rob scale`. Besides, `unibiv` has preliminarily standardized each variable, say x , using the median and the scaled MAD: $(x - \text{median}(x))/(1.4815 \cdot \text{mad}(x))$. With the optional parameter `madcoef`, it is possible to standardize the data using alternative measures of location and scale.

If the function is run on the full set made up of 17 variables, the plot would be hard to interpret, but the output `out` would provide a synthesis of the application of univariate and bivariate analyses in a matrix containing the unit indexes, the number of times each unit has been declared a univariate or bivariate outlier. For example, using the full dataset unit 25 is always declared a univariate outlier or a bivariate outlier in 90.44% of the cases, while unit 50 is declared as a univariate outlier 15 times and 57.35% times as a bivariate outlier.

6.2.2 FSM

It is clear that univariate or pair-wise approaches to complex and correlated data may lead to partial and perhaps erroneous conclusions. The bivariate view of DS17 in Figure 7 is just one out of 136 possible ones. A proper multivariate analysis of the dataset can be addressed with MVE, MCD, S, MM estimation or with the Forward Search approach using functions `FSM` and `FSMeda`.

`FSM` implements an automatic outlier detection procedure, described in Riani, Atkinson, and Cerioli (2009), which has a simultaneous size close to its nominal 1% and a great power. In this method, a series of theoretical simultaneous confidence bands (envelopes) associated with the quantiles of the distribution of the minimum Mahalanobis distance, provide an objective basis for decisions about the number of outliers in a sample. This detection method can be applied to the full set of DS17 variables using

```
out = FSM(X).
```

Then, `out.outliers` will contain the list of the units declared as outliers, 12, 21, 25, 38, 46 and 50. Recalling that in this biopharmaceutical problem units are individuals carefully selected to produce negative control samples, this fraction of outliers seems too high. An excessive number of outliers suggests that the distribution of the data is likely to be non normal. In this case, the user can exploit option `bonflev` to force the procedure to stop when the trajectory exceeds for the first time the very conservative 99.9% Bonferroni bound,

```
out = FSM(X, 'bonflev', 0.999),
```

`out.outliers` still contains five outliers (units 12, 21, 25, 38, and 50), which is an indication of strong non normality. In such a situation one may try improving the analysis by using a transformation of the data.

6.2.3 FSMtra and FSMfan

In the extension of the well known Box and Cox (1964) family to multivariate responses there is a vector λ of v transformation parameters, one for each of the v response variables. `out=FSMtra(X)` implements the monitoring of maximum likelihood estimates of transformation parameters and the likelihood ratio test statistic for transformation. The main fields of the output structure are `out.MLEtra`, with the MLE of transformation parameters along the search, and `out.LIKrat`, with the associated likelihood ratio tests. The specific nature of DS17 suggests to estimate a common transformation value for all variables that can be done by setting the optional parameter `'onelambda'` to 1. Figure 8 shows the likelihood ratio test plots obtained using function `FSMtra` as follows

```
[out] = FSMtra(X, 'onelambda', 1, 'la0', lambda, 'plotslrt', 1)
```

where $\lambda = [1 \ 0 \ -0.5 \ -0.25]$. Option `'plotslrt'`, when set to 1, provides the graphical output. Option `'la0'` controls which values of the transformation parameters have to be tested. For example `la0 = 1` tests the hypothesis of no transformation, `la0 = 0` tests the logarithmic transformation, `la0 = -0.5` tests the reciprocal of square root and `la0 = -0.25` tests the reciprocal fourth root. From the plots it is clear that DS17 must be transformed ($\lambda = 1$ is totally inappropriate) and that the most promising transformation parameter is $\lambda = -0.25$.

FSDA also includes a confirmatory test for the suggested transformations (Riani and

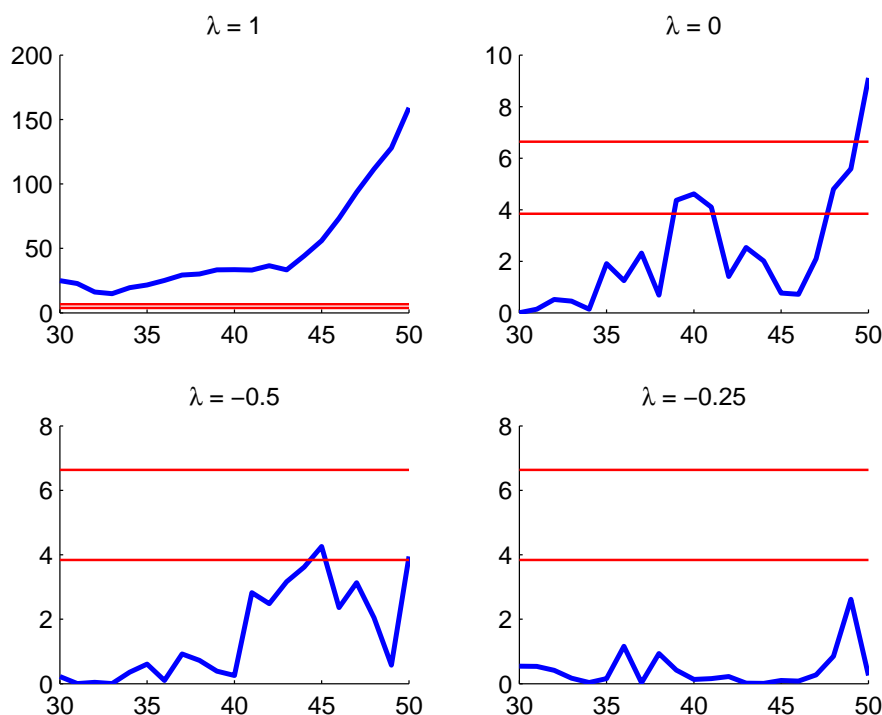


Figure 8: Likelihood ratio test plots obtained using `FSMtra` for testing the transformation parameters $\lambda = 1, 0, -0.5, -0.25$ on DS17. A common value of λ is tested for all variables. Throughout the search the test should be smaller than a χ^2 of 1 degree of freedom at a desired confidence level. The two horizontal lines refer to the 95% and 99% levels of the χ^2_1 distribution.

```

>> out = FSM(X.^(-0.25));
-----
Signal detection loop
dmin(45,50)>99.9% and dmin(46,50)>99.9% and rmin(44,50)>99%
-----
Signal validation
Validated signal
-----
Start resuperimposing envelopes from step m=44
Superimposition stopped because d_{min}(45,48)>99% envelope
$d_{min}(45,48)>99$\% envelope
Subsample of 47 units is homogeneous
-----
Final output
Number of units declared as outliers=3
Summary of the exceedances
      1          99          999          9999          99999
      2           8           6           5           1

```

Figure 9: The FSM output displayed in the MATLAB command window.

Atkinson 2000) based on the monitoring of (signed square root) likelihood ratio for testing $H_0 : \lambda_j = \lambda_C$, where the default value of λ_C is $[-1, -0.5, 0, 0.5, 1]$) when all the other variables are transformed as required. For example to produce a confirmatory plot (fan plot) for each of the 17 variables using an expansion based on the five most common values of λ , while transforming all the other variables using the reciprocal fourth root, we can use the following syntax:

```
[out] = FSMfan(X, -0.25*ones(1,17))
```

If we now again apply FSM on the data transformed using the best parameter value found with FSMtra and FSMfan, i.e. if we run

```
[out] = FSM(X.^(-0.25))
```

we find that only three units, namely 12, 32 and 38, are declared as outliers. During the execution of the code, information on the status of the Forward Search run is displayed in the MATLAB command window, shown in Figure 9. In the listing, the “Signal detection loop” segment refers to the detection rule based on consecutive exceedences, while the “Superimposition of the envelopes” refers to a validation phase of a potential signal that exceeds the envelopes.

6.2.4 FSMeda

FSMeda has exploratory purposes. It enables to storage of a series of quantities along the Forward Search (Mahalanobis distances (MD), minimum MD outside subset, maximum MD among the units belonging to subset in each step and other tests). Through the joint analysis of the plots which monitor the progression of the statistics along the Forward Search, it is possible to detect the observations that differ from the bulk of the data. These

may be individual observations that do not belong to the general model, that is outliers. Alternatively, there may be a subset of data that is systematically different from the majority. The monitoring of the progression of the statistics along the search not only enables the identification of such observations, but also lets us appraise the effect that these observations exert on parameter estimates and on inferences about models and their suitability.

FSMeda can, for example, be used to explore the complex structure of DS12. As an example, we can try running the Forward Search from an initial subset `bs1` chosen within the first group of DS12 which, based on the very partial view of Figure 1 (right scatterplot, ‘*’ symbols), seems to deviate from the other two. This is done with

```
[out] = FSMeda(Y,bs1,'init',13,'plots',1,'scaled',1)
```

where `init` specifies the point where to initialize the search and start monitoring the diagnostics. `init` is set to 13 because DS12 has 12 variables and `bs1` must be formed mainly by units in the ‘*’ group. Figure 10 shows the graphical output of the run, a plot of minimum Mahalanobis distances (scaled, using option ‘scaled’ set to 1) where the first big jump above the 99% envelope occurs precisely at step 17, that is in the step prior to the inclusion of the first unit from the two other groups. The sudden decrease of the distance, even below the lowest 1% confidence envelope, is due to the big change in the position of the centroid and gives a strong signal that the minimum Mahalanobis distance is much smaller than it should be. The exceedance of the lower envelope frequently happens when the data in the subset contain observations from more than one population. This corroborates the conclusion that DS12 cannot be treated with the standard models based on just

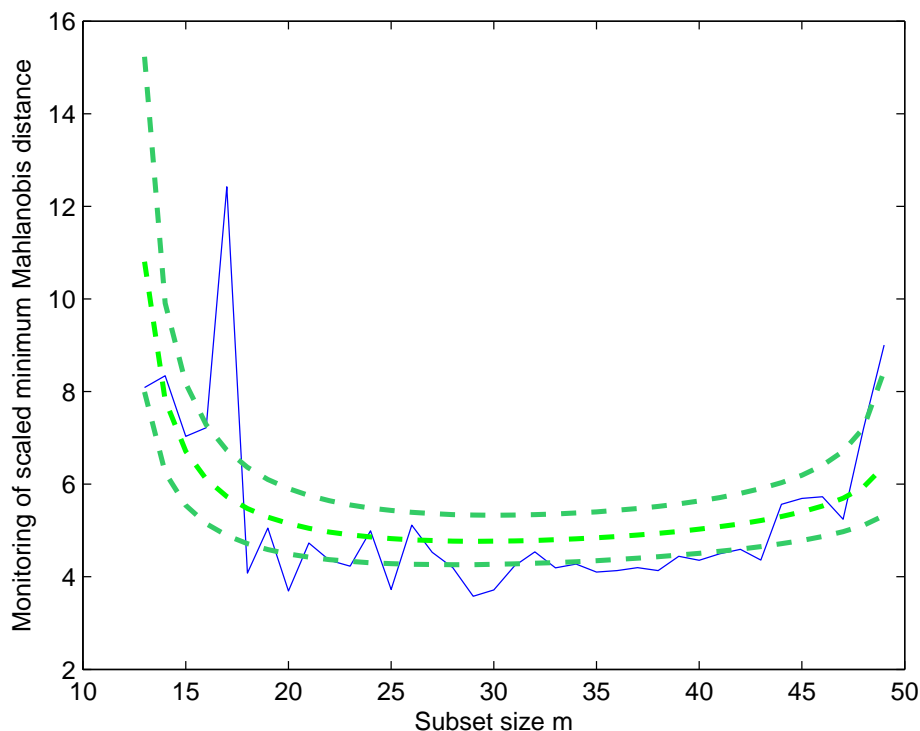


Figure 10: Progression of scaled minimum Mahalanobis distance monitored with `FSMeda` for dataset DS12, starting from units chosen within the group which seems rather distinct from the other two. The three dashed lines respectively refer to 1%, 50% and 99% confidence levels.

one population and perhaps suggests reconsidering the experimental setting giving rise to such dis-homogeneous populations.

The output structure `out` obtained by function `FSMeda` contains many other statistics, which can be plotted separately with minimum effort. These include the monitoring of Mahalanobis distances for all units (`out.MAL`), the units belonging to the subset (`out.BB`), the minimum Mahalanobis distance of units outside the subset (`out.mmd`), the maximum Mahalanobis distance of units inside the subset (`out.msr`), the elements of the covariance matrix (`out.S2cov`), and others.

6.2.5 Smult and MMmult

`Smult` and `MMmult` compute S and MM estimators in multivariate analysis. S estimators are robust estimators which have the highest possible break-down point (0.5). Unfortunately S -estimates with a smooth ρ function cannot simultaneously have a high breakdown point and high efficiency (Maronna, Martin, and Yohai (2006), p. 131). In particular, it was shown that an S -estimate with breakdown point equal to 0.5 has an asymptotic efficiency under normally distributed errors that is not larger than 0.33 (see Hössjer 1992). So these estimators are mainly used as starting point in the MM procedure Yohai (1987). In the FSDA implementation, the fast S algorithm for multivariate location estimation is for the moment only based on Tukey's biweight function.

`MMmult(X)` receives an S -estimator as starting value and an M estimator with fixed scale and re-descending ψ Tukey biweight function is used from there. The default nominal efficiency which is used is 95%. The core of the algorithm is a function that can also be used as standalone, if the user supplies directly an estimate of location (`loc0`), shape (`shape0`) and scale (`auxscale`): `MMmultcore(X, loc0, shape0, auxscale)`. It does iterative reweighted least squares (IRWLS) steps from "initial location" (`loc0`) and shape matrix (`shape0`) keeping the estimate of the scale (`auxscale`) fixed. The graphical output of running the two estimators on DS17 (after transforming the data with negative fourth root) and using a simultaneous Bonferroni confidence level of 99% is shown in the left panel of Figure 11.

```
conflev = 1-0.01/(size(X,1));
[outMM]=MMmult(X.^(-0.25),'plots',1,'conflev',conflev);
[outS]=Smult(X.^(-0.25),'plots',1,'conflev',conflev);
```

In this case, the outliers declared on the basis of Mahalanobis distances from S or MM estimators would be too many. As usual, details on the content of `outMM` and `outS` are found in the FSDA help pages.

Huber and Ronchetti (2009) pointed out (Section 7.12, p. 195-198) an inherent instability of S -estimators which, in their opinion, even "disqualifies an estimate from being called robust". The problem has to do with the use of re-sampling methods for solving the S -estimation minimization problem (locally, unless all possible subsamples are extracted) and, therefore, goes beyond S estimation. Due to re-sampling, runs starting from different extracted samples are likely to produce different estimators, with a lack of reproducibility which is disturbing. To give more stability to resampling-based estimators, FSDA has adopted the efficient sampling strategy mentioned in Section 4, which allows to increase considerably the number of extracted subsets without incurring in critical degradation of computational performances.

6.2.6 MVE and MCD

Functions `mcd` and `mve` implement the Minimum Volume Ellipsoid (Rousseeuw (1985)) and Minimum Covariance Determinant (Rousseeuw and Van Driessen (1999)) estimators. MCD is given by the subset of h out of n data points with smallest covariance determinant. The location and scatter estimates are therefore the mean and a multiple of the covariance matrix computed on h such points. Similarly, MVE is built by looking at the smallest volume ellipsoid that covers h points. In FSDA the fraction of h points is chosen implicitly by setting the optional parameter for the breakdown point, `bdp`, which by default is 0.5. For example, `bdp=0.25` implies that approximately $h = 0.75n$ points (the exact formula is more complex) will be used by the estimators.

These methods assume that the number of observations is at least 5 times the number of variables (Rousseeuw and van Zomeren 1990), otherwise `bdp` should be decreased from the standard 0.5 to smaller fractions, e.g. 0.25 (Verboven and Hubert (2005)). Note that for DS17 this rule of thumb is not fulfilled and, thus, in this case the results should be taken with caution.

As for the S and MM estimators, graphical output includes the plot of the robust Mahalanobis distances, shown in the right panel of Figure 11, and the scatterplot matrix with the outliers highlighted with different symbol. As an illustration, we show the matrix of scatterplots of the first five variables of DS17 against each other (Figure 12). Note that the histograms along the diagonal account for the different elements that are present in each bin, grouping them by type (normal units and outliers).

These graphical outputs have been obtained using a simultaneous Bonferroni confidence level of 99% and 25% breakdown point with the following code:

```
conflev = 1-0.01/(size(X,1));
bdp = 0.25;
[RAWmcd,REWmcd] = ...
    mcd(X.^(-0.25),'plots',1,'conflev',conflev,'bdp',bdp);
[RAWmve,REWmve] = ...
```

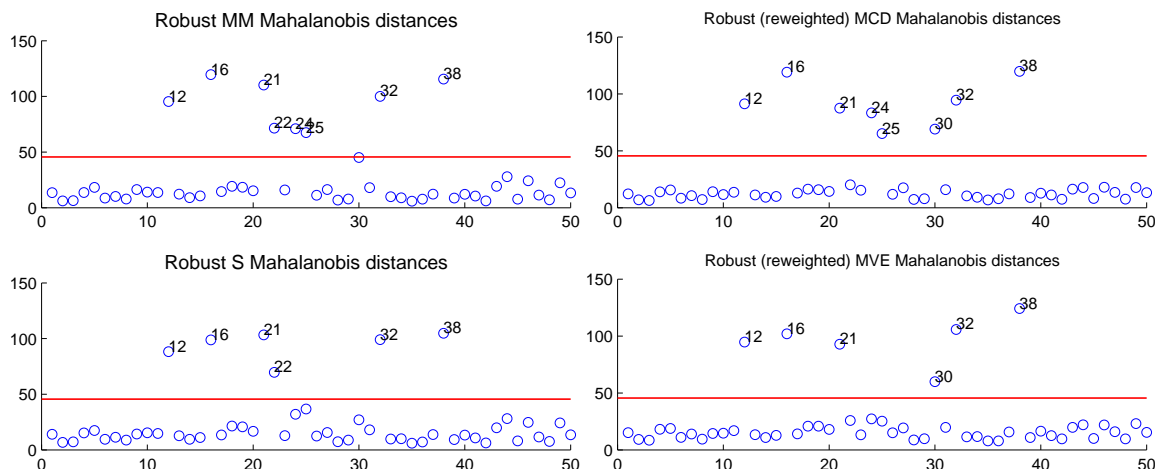


Figure 11: DS17, transformed with negative fourth root. Index plot of robust Mahalanobis distances found with functions `MMmult` and `Smult` (left panels) and `mcd` and `mve` (right panels).

```
mve(X.^(-0.25), 'plots', 1, 'conflev', conflev, 'bdp', bdp);
```

The output of the two estimators consist of two structures RAW and REW. RAW refers to the standard raw MCD or MVE; on the other hand, REW refers to the algorithms with re-weighting step. The structures contain several fields, including the location and covariance matrix (`loc` and `cov`), the correlation matrix (`cor`), the units forming the best subset (`bs`), the robust Mahalanobis distances (`md`), the list of the units declared as outliers (`outliers`) using confidence level specified in the optional input `conflev` (default is 0.975) and the number of subsets thought to be singular `singsub`. It is worth mentioning that the estimated output covariance matrix (`cov`, re-weighted or not) is based on consistency and small sample correction factors. The consistency factor is based on the χ^2 distribution and on the ratio h/n (see Tallis (1963) or Butler, Davies, and Jhun (1993)). The small sample correction factor is due to Pison, Van Aelst, and Willems (2002) to make the estimator unbiased.

A disadvantage of MCD and MVE, as we have seen in Figure 11, is that they tend to declare a larger number of outliers than predicted by the Bonferroni bound when the sample size is small. This is because they make substantial use of the asymptotic χ^2 approximation to the distribution of robust (re-weighted) distances, which may be far from accurate even for moderately large sample sizes. A much more accurate approximation, that can work well when $n/v \approx 5$ or even smaller, is due to Cerioli (2010) and for MCD is implemented in option `betathresh`.

6.2.7 Fast algorithms for S and MCD estimators

The MCD and S implementations exploit respectively the computationally efficient algorithms of Rousseeuw and Van Driessen (1999) and Salibian-Barrera and Yohai (2006). In FSDA the fast algorithm is fully controlled by optional parameters including:

- `nsamp`, the number of subsamples to be extracted (default is 500).

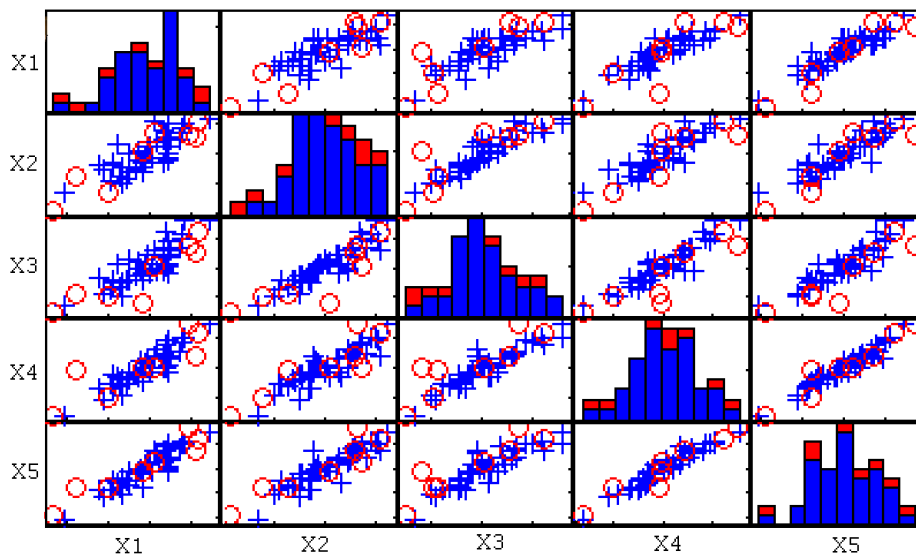


Figure 12: The first five variables of DS17, transformed with negative fourth root, are plotted against each other with outliers found by MCD highlighted. Note the histograms, which account for the different elements that are present in each bin (normal units and outliers).

- `refsteps`, the number of refining iterations in each subsample (default is 3).
- `reftol`, the tolerance for the refining steps (default is 10^{-6}).
- `refstepsbest`, the number of refining iterations for each best subset (default is 50).
- `reftolbest`, the tolerance for the refining steps for each of the best subsets (default is 10^{-10}).
- `best`, the number of “best locations” to remember from the subsamples, which will later be iterated until convergence (default is 5).

6.3 Robust regression analysis, transformations and model selection

The group of functions implementing robust regression estimators is structured almost specularly to the multivariate counterpart and includes S, MM, LMS, LTS and the Forward Search estimators. These functions will be briefly introduced in Subsection 6.3.3 after demonstrating a set of tools for automatic transformation and model selection.

Such tools exploit the data-driven flexible trimming provided by the Forward Search to choose regression models in the presence of outliers. In subsection 6.3.1 we use monitoring of the score test statistic to test whether the response must be transformed. The result is the so called forward score test discussed in Atkinson and Riani (2002b). In subsection 6.3.1 we address the issue of robust model selection, using the distributional results on the added t -test (Atkinson and Riani 2002a) and C_p in the forward search (Riani and Atkinson 2010) and a powerful new version of the C_p plot, which is known as *generalized candlestick plot*.

The data chosen to illustrate how these procedures work are the measurements on ozone concentration used by Breiman and Friedman (1985) when introducing the *ACE* algorithm. These are a series of daily measurements, from the beginning of the year, of ozone concentration (y) and eight meteorological variables (X_1, \dots, X_8) in California. Atkinson and Riani (2000, §3.4) analyse the first 80 observations and find that a time trend (*Time*) should be considered as one of the explanatory variables. Together with the constant term, we therefore have $p = 10$ explanatory variables.

6.3.1 Analysis of transformations

The plot monitoring the score test statistic for transformation given in Figure 13 is produced using the following code:

```
load('ozone.txt','ozone');
y=ozone(:,9);
% Add a time trend to design matrix X
X=[(-40:39)' ozone(:,1:8)];
% Produce the fanplot
[out]=FSRfan(y,X,'plots',1);
```

The plot, that for its characteristic shape is known as a *fan plot*, clearly shows that the response must be transformed by taking logs. The evidence for transformation is diffused

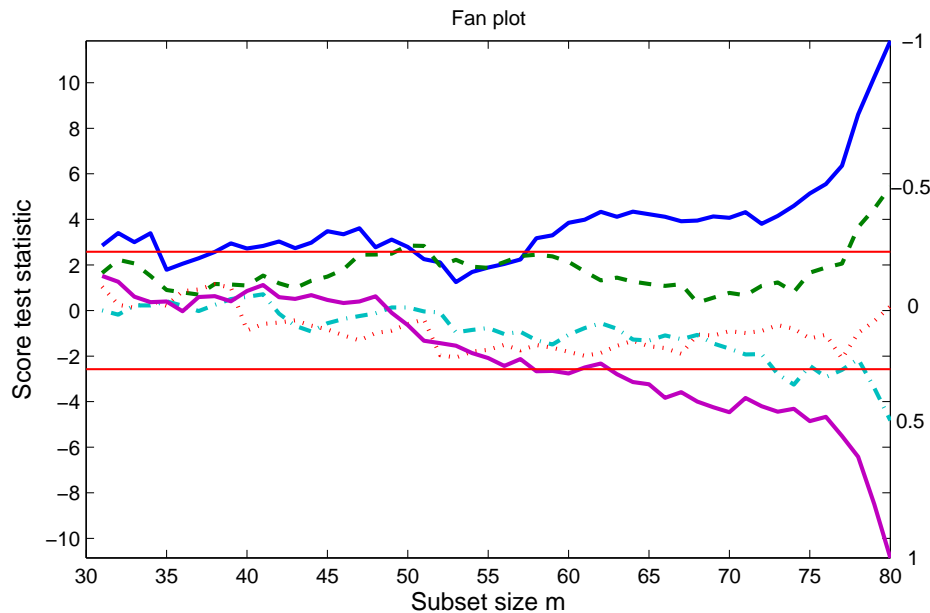


Figure 13: Ozone data: fanplot. The two horizontal bands correspond to the 0.5% and 99.5% quantiles of the standard normal distribution. The trajectory for the logarithmic transformation progresses within the bands throughout the search.

throughout the data and does not depend on the presence of particular observations. More precisely, the hypothesis of no transformation starts to be rejected from m around 60.

The fan plot can be made dynamic using function `fanplot` and option `databrush`:

```
fanplot(out, 'databrush', 1)
```

The user can select one or more trajectories inside the fanplot. As soon as the user releases the mouse button, the following output automatically appears:

1. In the command window the list of the steps which have been brushed (it is also possible to select non consecutive steps) and the list of the units associated with the brushed steps together with their entry order.
2. The scatter of the response (transformed as specified in the associated score curve which is selected) against each explanatory variable (*yXplot*), with highlighted the units associated to the selected steps in the fanplot. In the *yXplot* using a simple click on the marker in the legend, it is possible to hide/show the brushed/unbrushed units.
3. A plot of monitoring scaled residuals with highlighted the trajectories associated with the selected units in the fan plot. In this plot it is possible to select additional trajectories by simply clicking on the mouse on a particular curve. After a click with the mouse, a pop up yellow window automatically appears (see small rectangle at the bottom of the right part of Figure 15) which lists the units associated with the selected trajectories together with their entry order.

For example, if the user selects the last two steps in the curve associated with $\lambda = 0.5$, which cause a strong rejection of the null hypothesis that the proper transformation is the square root, the additional information which is displayed is shown in Figures 14 and 15.

```

>> fanplot(out,'databrush',1)
To brush, with the mouse select steps in the FAN plot
Brushed steps
  79
  80
Associated brushed units
  71
  53
Steps of entry of brushed units
  79   71   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
  80   53   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN

```

Figure 14: Ozone data: information which automatically appears in the command window if the user has selected the last two steps in the trajectory associated to $\lambda = 0.5$ in the fan plot of Figure 13. Unit 71 joins the subset when $m = 79$, while unit 53 enters the final step. The series of NaN indicates that no other unit joins the subset in the selected steps.

6.3.2 Robust model selection

Monitoring the t -tests for individual regression coefficients in the forward search fails to identify the importance of observations to the significance of the individual regressors. This failure is due to the ordering of the data by the search. It is necessary therefore to use an added-variable test which has the desired properties since the projection leading to residuals destroys the effect of the ordering. In order to monitor the evolution of the added t -test we can use the following simple code:

```

% create variable labels
labels={'Time','1','2','3','4','5','6','7','8'};
[out] = FSRaddt(y,X,'nameX',labels,'plots',1);

```

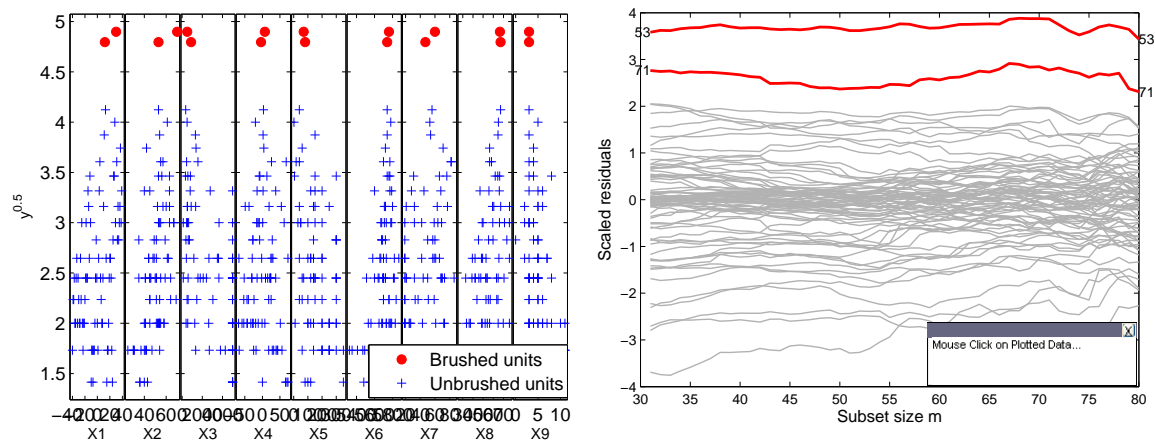


Figure 15: Ozone data: additional plots which automatically appear if the user has selected the last two steps in the trajectory associated to $\lambda = 0.5$ in the fan plot of Figure 13

Figure 16 shows that variable *Time* is clearly significant while, for example, variable 1 fluctuates around 0 and is certainly the first which could be removed in an iterative procedure which deletes the least significant variable at each iteration.

If the user decides to consider all possible submodels for the selection of the best model, generally the statistic which is used is Mallows C_p . However, this statistic being an aggregate statistic, that is a function of all the observations, suffers from the well-known lack of robustness of least squares and provides no evidence of whether or how individual observations or unidentified structure are affecting the choice of model, so it is necessary to consider its forward version. Function `FSRms` stores the trajectories of C_p for the best models (i.e. the models which, over the central part of the search, had one of the three smallest values of C_p). For example the code below

```
[outms]=FSRms(y,X,'smallpint',4:6,'labels',labels)
```

stores in matrix `outms.MAL` the values of C_p monitored along the forward search in the range of values of p from 4 to 6 for the best models. In order to summarize the information contained in `outms.MAL`, the user can produce the “candlestick” plot (see Figure 17), simply using the following code

```
cdsplot(outms,'laboutl',1)
```

The vertical lines in the plot summarize the values over the central part of the search. The definition of the candlesticks is:

- Lowest Value; minimum in the central part of the search;

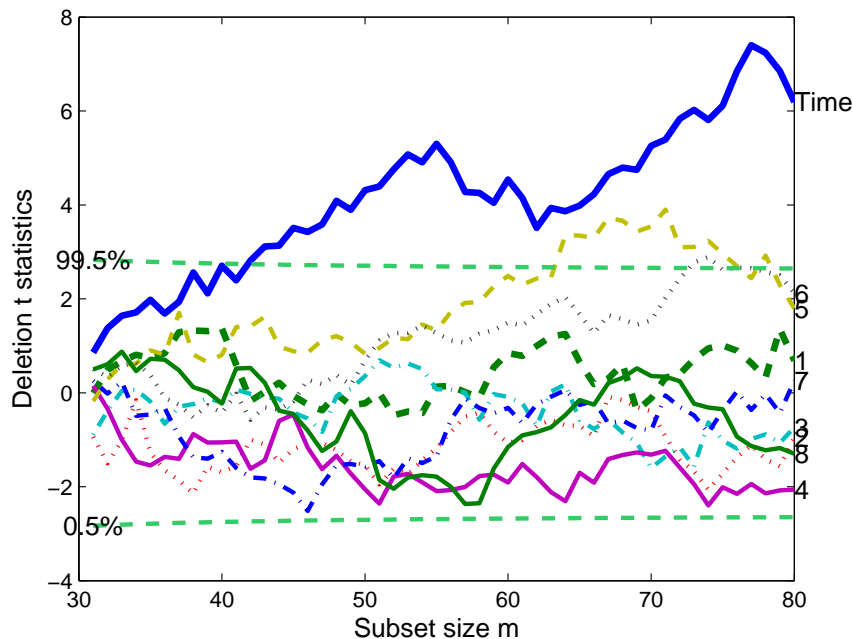


Figure 16: Transformed Ozone data: forward plot of the 9 added-variable t -statistics.

- Central Box; mean and median of the values in the central part of the search; filled if the mean is greater than the median;
- Stars; the values in the final part of the search, if these lie outside the box. As default the last 5% of the steps form the final part. However it is possible to use option `finstep` to specify the central and final part of the search;
- Unfilled Circle; the final value.

Thus, each point in the standard non-robust C_p plot is replaced by a single vertical line and a series of extra symbols. Option `'laboutl'` specifies whether to add the labels of the 'influential units' that is the units which enter the subset in the final part of the search (in this example steps 77-79) and bring the value of the C_p below the minimum or above the maximum value of the central part of the search. More precisely, by setting `'laboutl', 1` we display just the unit number close to its symbol. On the other hand, by setting `'laboutl', 2`, the software adds both the unit number and the associated entry step. In the remaining cases (default), no label is added to the plot.

For example, this plot clearly shows that the model with explanatory variables $Time$, X_4 , X_5 and X_6 has values of C_p which lie well inside the 2.5% and 97.5% confidence bands. However, in the final step (see the circle) the value of C_p is above the 97.5% threshold. On the other hand, for example, the model with explanatory variables $Time$, X_2 , X_5 and X_8 shows values of C_p which are much greater than the threshold provided by the 97.5% level, but only in the final steps the values of C_p return inside the confidence interval of acceptance (masking effect).

Also the candlestickplot is dynamic. Simply adding the option `cpbrush` as follows

```
cdsplot(outms, 'cpbrush', 1, 'laboutl', 1),
```

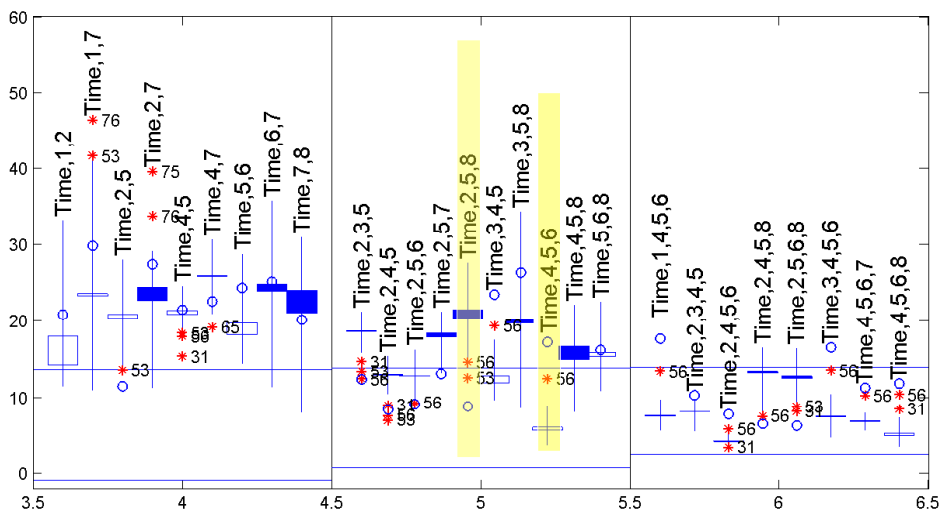


Figure 17: Ozone data: generalized candlestick plot for $C_p(m)$ for the best models in this range with $p = 4, 5, 6$. The horizontal bands denote 2.5% and 97.5% bands of C_p . The candles which have a background strip ($Time$, X_4 , X_5 , X_6 and $Time$, X_2 , X_5 and X_8), denote the two models selected by the user.

it is possible to highlight with the mouse the models of interest in order to examine in detail the associated trajectories of C_p along the forward search. For example, if the user simply selects the two models discussed above ($Time, X_4, X_5, X_6$) and ($Time, X_2, X_5$ and X_8), the plot which automatically comes out is given in Figure 18.

6.3.3 Robust estimators in regression

The Forward Search in regression is implemented by functions `FSR` and `FSReda`.

`FSR` is conceived for outlier detection. Inference is made by examining the trajectory of the minimum deletion residual among observations that are not in the subset. As in the multivariate case, the signal detection rule is based on consecutive exceedances above the extreme envelopes. This procedure has a size which is very close to the nominal and an average power which is generally greater than that of comparable methods (Torti and Perrotta 2011).

`FSReda` implements the Forward Search approach for exploratory data analysis, by storing along the search many regression statistics such as residuals, leverage, minimum deletion residual outside subset, maximum studentized residual, units belonging to subset in each step and other tests. Then, the record of such statistics can be plotted in traditional or interactive graphical displays.

`LXS` implements Least Trimmed Squares (LTS) and Least Median of Squares (LMS, Rousseeuw 1984) estimators. In these estimators the percentage of trimming is fixed *a priori* and can be controlled using option `h`. As in the multivariate counterpart `MCD` (see section 6.2.7), `h` is linked to the breakdown point in option `bdp`. Using option `rew` it is possible to choose between their raw or re-weighted versions. Option `nsamp` controls the number of samples to extract to find the estimator. An important parameter is `lms`, controlling the type of algorithm used. Default is `lms=1`, which is to execute LMS. `lms=2` runs the FastLTS with all default options, otherwise if `lms` is a scalar different from 1 or 2 standard LTS is used without concentration steps. If `lms` is a structure, the fast algorithm of

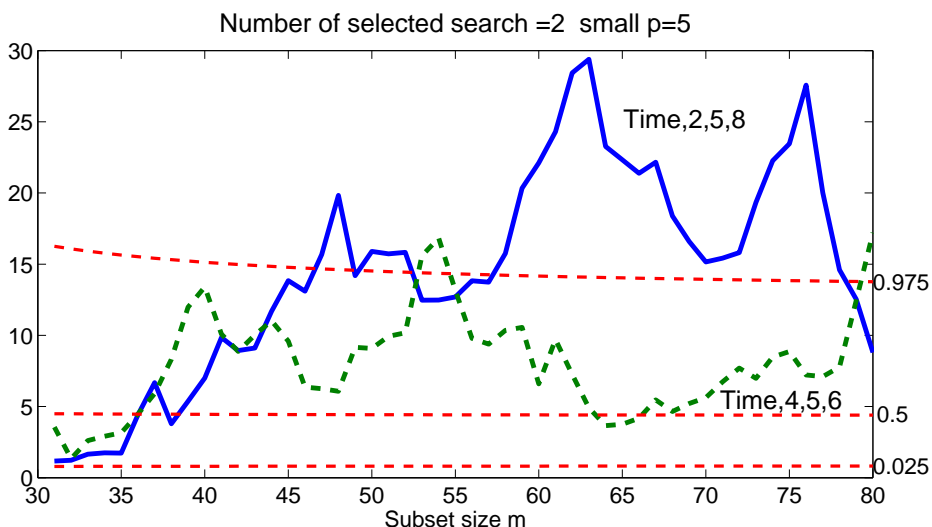


Figure 18: Ozone data: forward plots of $C_p(m)$ for the models the user has selected with the mouse in the candlestick plot. The almost horizontal lines are associated with 2.5%, 50% and 97.5% critical points of the distribution of C_p during the search.

Rousseeuw and Van Driessen (2006) (with concentration steps) is used with the possibility to control different aspects of the algorithm, by setting the same fields listed in Section 6.2.7 for the multivariate counterpart.

`Sreg` and `MMreg` implement *S* and *MM* estimators in linear regression. In the *S* procedure, similarly to what happens for *LXS*, the breakdown point is fixed a priori. Once a robust estimate of the scale is found, one can obtain a nominal efficiency by using the *S* estimate as the starting point for an iterative procedure leading to *MM* estimators. When calling these functions, it is possible not only to set up all options seen for function *LXS*, but also a series of additional options to solve the equation for the scale.

6.4 Interactive Statistical Visualization

This section describes some of the *FSDA* graphical interaction features. The emphasis is on the general interactive paradigms adopted in *FSDA*. This line of work was initiated by the need to simplify the extraction of information from the numerous forward plots produced by the Forward Search, which would require otherwise tedious ad hoc programming of the traditional static plots (Perrotta, Riani, and Torti 2009). These features have now being extended to almost all robust statistical graphics functions in *FSDA*.

6.4.1 Brushing logically linked objects

In Figures 13, 15 and 17 there are examples of interactive plots where a selection made by the user on objects in a given plot is highlighted also in other plots, containing different objects *logically* linked with those selected. Similarly, the points in the index plots of robust Mahalanobis distances produced by the *S*, *MM*, *MVE*, *MCD* and Forward Search methods (Figures 11 and 20), are logically connected with a traditional matrix of scatterplots, so that selections in the index plot give rise to the automatic generation of a scatterplot with the selected units highlighted as in Figure 12.

The link is said to be “logical” because the points in the scatterplot of Figure 15 have no variable in common with the scaled residual trajectories in the right panel of the same figure or with the fanplot trajectories of Figure 13, where the selection was done. Note that the native brushing tool of *MATLAB* does not work on these objects, as it is limited to objects physically linked by at least one variable in common. To our knowledge this type of logical link has not been exploited so far by other statistical libraries.

Brushing is enabled by the very flexible option `databrush`, which is implemented in two modalities. A *non persistent* modality, where the selection can be made by the user only once, and a *persistent* modality, where the selection can be repeated multiple times. In addition, there is a persistent *non cumulative* brush option, where every time a brushing action is performed previous selections are removed, and a *cumulative* one where each selection is highlighted and appropriately reported in the legend of the graphs involved.

The non persistent modality with all default options is obtained when `databrush` is 1 (or any other scalar). It also works when `persist` option is set to the default empty string `''`. To change the values of one or more options, it is sufficient to define `databrush` as a structure with the field names of the option(s) to change and the corresponding field value(s) set as desired. Most of the brushing options are specific to a *FSDA* internal function, `selectdataFS`, which is activated whenever a brushing operation is requested by the user. The persistent modality is activated by the `persist` option, which to be cumulative

has to be 'on' (then unit(s) currently brushed are added to those previously brushed, with a different color) and to be non cumulative has to be 'off' (when a new brush is performed, units previously brushed are removed).

6.4.2 Filtering and annotating objects

FSM has found automatically on DS17 (after appropriate transformation) three outliers. Figure 9 shows the function output. A graphical exploration of the result can start from the inspection of the progression of the Mahalanobis distance through the search. This can be obtained with the graphical function `malfwdplot`, after obtaining the desired statistic using `FSMeda`, starting the search from a random subset or using one, say `bs`, determined with the technique of robust ellipses (function `unibiv`):

```
[out] = FSMeda(X.^(-0.25),bs);  
malfwdplot(out);
```

Given that, for large datasets, there are many trajectories to display in `malfwdplot` (or `resfwdplot` in regression), we have introduced options `bground` and `fground` to control the aspect of unimportant trajectories (e.g. plotted in a faint color) and those to be highlighted (e.g. in boldface). In addition, it is possible to activate the `datatooltip`, which will display relevant information on a trajectory selected by the user at a given step of the search with a mouse click. For example, if the user decides to highlight trajectories 12, 32, 38, to use a faint color for the unimportant units (say those with distance smaller than 5) and to add a personalized `datatooltip`, the following code is appropriate:

```
fground = struct; bground = struct;  
fground.funit=[12 32 38];  
bground.bthresh=5;  
bground.bstyle='faint';  
datatooltip.DisplayStyle = 'datatip';  
malfwdplot(out, 'datatooltip', datatooltip, ...  
           'fground', fground, 'bground', bground);
```

The result in Figure 19 shows a selection made for unit 38 at step 43. The annotation of the outliers confirms that the Mahalanobis trajectories of the three outliers have a value comparable with that of the majority of the other units only at the very end of the search.

One might also want to highlight the trajectories of the units that are in the subset at given steps of the search. The interaction modality that we have developed for this purpose is activated by setting the field `SubsetLinesColor`, which also determines the color used to highlight the trajectories in the subset, for example

```
datatooltip.SubsetLinesColor = FSColors.cyan.RGB
```

In this modality if the user does repeated left mouse clicks in the proximity of steps of interest, the trajectories of the units belonging to the subset at the selected steps are dynamically highlighted. A right mouse click terminates this selection modality by marking with a up-arrow the last selected step. The process is very dynamic and cannot be reproduced by one or more figure snapshots.

Sometimes one may also want to remove redundant or unwanted information from a plot. For example, the left panel of Figure 20 which shows the index plot of the robust Mahalanobis distances generated with

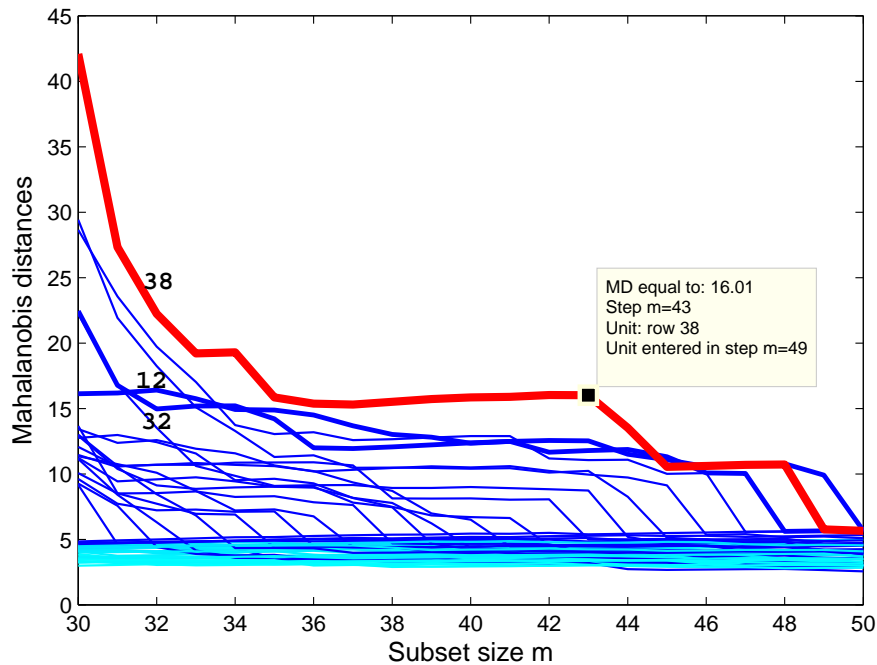


Figure 19: Option `data_tooltip` is activated by the user to get information about specific units (unit 38 in the example), the last step when they enter the subset ($m=49$), and the value of their Mahalanobis distances ($MD=16.01$) at the selected steps of the search ($m=43$).

```
conflev = [0.95 0.975 0.99 1-0.01/(size(X,1))];
malindexplot(out.MAL(:,end-3).^2,17 , 'conflev' , conflev);
```

while the right panel shows what the user gets by clicking with the mouse on three of the four confidence band legends: the unwanted bands disappear and the corresponding legends are displayed in greyish. This modality is activated by a function called `clickableMultiLegend` and is generally applicable to many other FSDA graphical objects. For example, in the scatterplot matrix of Figure 12, the blocks of the grouped histograms relating to the detected outliers and the corresponding circles would become both transparent if the user clicks on an outlier legend that, like in this specific case, has not necessarily to be in the Figure concerned. In such a case, also the objects associated to the outliers in the figure, where the legends are located, would disappear. To our knowledge, this feature is new for general scatterplot matrices.

7 Developments

Like other well maintained statistical software libraries, FSDA is constantly enriched with new functions and methods. Future releases will gradually integrate methods of Principal Components, Discriminant Analysis, Cluster Analysis and Nonlinear Regression.

The feedback received from many users (a functional mailbox `toolboxfs@unipr.it` is associated with the FSDA) advocates for friendly graphical user interfaces to the most important statistical, visualization and data acquisition and transformation functions. Others advance the need to address big datasets and big collections of datasets or, in other words, the need of very scalable statistical functions.

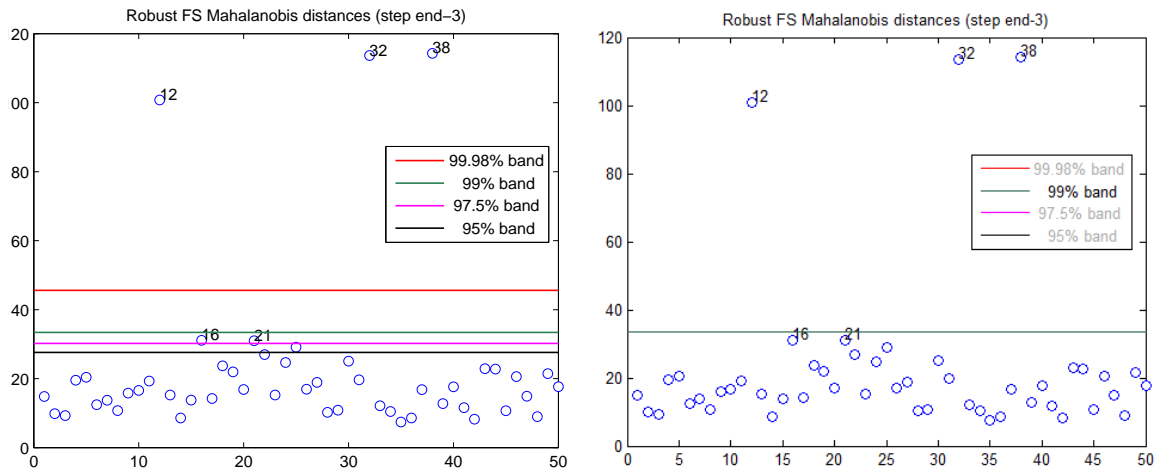


Figure 20: Index plot of robust Mahalanobis distances for transformed DS17. The plot can be brushed to produce a matrix of scatterplots with selected units highlighted as in Figure 12. On the right panel, Function `clickableMultiLegend` has activated a graphic modality to hide or show symbols inside all active plots (or similar multi-plot) by clicking on the legend.

Scalability issues go beyond the performance aspects discussed in Section 4 and require reconsideration of even very fundamental (and thus very used) statistical functions. One of such functions is the median of a set of values, widely used in applications requiring the repeated application of robust estimators. An example is the Median Absolute Deviation (MAD) criterion, which is used to robustly standardize the data in many robust procedures, e.g. the MCD. MATLAB computes the median with the standard but expensive approach of sorting the numbers, which takes $O(n \log n)$ comparisons, while just the middle element is needed and the ordering of the others is of no interest. In statistics this problem was already touched by Tukey (1978) and more recently (focusing more on storage than time aspects) by Rousseeuw and Bassett (1990). In computer science the issue has been tackled as a particular case of the *selection problem*, i.e. finding the i -th order statistic in a set of n distinct elements, with a minimum number of comparisons. Note that an efficient algorithm for the order statistics can lead to a considerable speedup of the fast algorithms of the MCD and LTS estimators (Sections 6.2.6 and 6.3.3). In fact such algorithms are based on repeated concentration steps, each one requiring the extraction of the h -th ordered value of an objective function (the covariance determinant or the sum of squared residuals respectively), which is typically done (inefficiently) after sorting the full set of n values.

Good overviews of the selection problem issue are in the books of Knuth (1998), p. 214–215 and Dasgupta, Papadimitriou, and Vazirani (2008), p. 60–61, together with an efficient $O(n)$ divide-and-conquer solution which in more than 20 years has evolved from using at most $5.4305n$ comparisons (Blum, Floyd, Pratt, Rivest, and Tarjan 1973) to $2.95n$ comparisons (Dor and Zwick 1999). We are testing this type of solution in view of introducing in FSDA fast versions of the median and order statistics functions. For example we observed that the default algorithm provided by C++ (`nth_element`), derived from the quicksort of Hoare (1961), leads in practice to a 50% gain on average, at least for sets of 100 to 500 elements.

Of course there are other functions that may have to be addressed to improve scala-

bility, but the median and order statistics examples well illustrates the related benefits and complexities.

We are also testing the potential benefits of a different approach to scalability, which is to write parallel code segments that MATLAB distributes automatically over different cores, CPUs and even machines or clusters. Currently this requires a separate Parallel Processing toolbox but, in the absence of it, MATLAB simply runs the parallel code in the traditional sequential mode.

8 Conclusions

FSDA is designed to couple robust statistical methods with interactive and flexible data exploration instruments, conceived to appreciate the effect of outliers or multiple populations on the estimates produced by the statistical method in use. We have seen this applied to the index plot of robust distances produced by the well known robust methods of regression and multivariate analysis (LMS, LTS, MCD, MVE, S and MM). We have also seen this in a variety of Forward Search plots, for which the extraction of information critically depends on the possibility of interacting with such plots.

Demonstrations have been conducted using real datasets related to a bio-pharmaceutical problem. We have analyzed such datasets in co-operation with the end users. The FSDA was used in the preliminary exploration of their chemical analysis samples, for checking robustly the normality of the data, finding appropriate transformations for non-normal and contaminated data and, finally, for detecting multivariate outliers. We hope we have demonstrated that these statistical tasks can be all simplified by the adoption of the FSDA toolbox.

Acknowledgments

We are grateful to Luigi Colombo, Simona Riva and Walter Pollini (respectively General Manager, Associate Director and consultant of RBM S.p.A., Merck Serono Non-Clinical Development) for introducing us to the biochemical problems and related statistical issues treated in the paper and for the interest shown in our tools.

The FSDA project is supported by the Faculty of Economics of the University of Parma (Italy) and by the Joint Research Centre (JRC) of the European Commission. Several colleagues have contributed to the conception, design, development and maintenance of FSDA. Special thanks go to Andrea Cerioli and Fabrizio Laurini (University of Parma), Anthony C. Atkinson (LSE, UK), Vytis Kopustinskas (JRC, for the development of several FSDA routines), Christophe Damerval (JRC, who tested the portability of FSDA in OCTAVE and SCILAB), Massimiliano Gusmini (who designed the FSDA logo) and Patrizia Calcaterra (who keeps the setup software updated and patiently addresses any problem with the development platforms).

Last but not least we acknowledge the fruitful interactions had with Francesca Perino and Giovanna Galliano (The MathWorks, Italy).

References

- Atkinson, A. C. and M. Riani (2000). *Robust Diagnostic Regression Analysis*. New York: Springer–Verlag.
- Atkinson, A. C. and M. Riani (2002a). Forward search added variable t tests and the effect of masked outliers on model selection. *Biometrika* 89, 939–946.
- Atkinson, A. C. and M. Riani (2002b). Tests in the fan plot for robust, diagnostic transformations in regression. *Chemometrics and Intelligent Laboratory Systems* 60, 87–100.
- Atkinson, A. C., M. Riani, and A. Cerioli (2004). *Exploring Multivariate Data with the Forward Search*. New York: Springer–Verlag.
- Atkinson, A. C., M. Riani, and A. Cerioli (2010). The forward search: theory and data analysis (with discussion). *Journal of the Korean Statistical Society* 39, 117–134. doi:10.1016/j.jkss.2010.02.007.
- Becker, C. and U. Gather (1999). The masking breakdown point of multivariate outlier identification rules. *Journal of the American Statistical Association* 94, 947–955.
- Blum, M., R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan (1973). Time bounds for selection. *Journal of Computer and System Sciences* 7(4), 448 – 461.
- Box, G. E. P. and D. R. Cox (1964). An analysis of transformations (with discussion). *Journal of the Royal Statistical Society, Series B* 26, 211–246.
- Butler, R. W., P. L. Davies, and M. Jhun (1993). Asymptotics for the minimum covariance determinant estimator. *The Annals of Statistics* 21, 1385–1400.
- Cerioli, A. (2010). Multivariate outlier detection with high-breakdown estimators. *Journal of the American Statistical Association* 105(489), 147–156.
- Cook, R. D. and D. M. Hawkins (1990). Comment on Rousseeuw and van Zomeren (1990). *Journal of the American Statistical Association* 85, 640–4.
- Dasgupta, S., C. Papadimitriou, and U. Vazirani (2008). *Algorithms*. McGraw-Hill.
- Daszykowski, M., K. Kaczmarek, Y. V. Heyden, and B. Walczak (2007). Robust statistics in data analysis – a review: Basic concepts. *Chemometrics and Intelligent Laboratory Systems* 85(2), 203 – 219.
- Daszykowski, M., S. Serneels, K. Kaczmarek, P. V. Espen, C. Croux, and B. Walczak (2007). Tomcat: A matlab toolbox for multivariate calibration techniques. *Chemometrics and Intelligent Laboratory Systems* 85(2), 269 – 277.
- Dor, D. and U. Zwick (1999). Selecting the median. *SIAM Journal on Computing* 28, 1722–1758.
- Filzmoser, P., S. Serneels, R. Maronna, and P. V. Espen (2009). Robust multivariate methods in chemometrics. In S. D. Brown, R. Tauler, and B. Walczak (Eds.), *Comprehensive Chemometrics, Chemical and Biochemical Data Analysis, Volume 3*, pp. 681 – 722. Oxford: Elsevier.
- Filzmoser, P. and V. Todorov (2011). Review of robust multivariate statistical methods in high dimension. *Analytica Chimica Acta. In Press, Corrected Proof*.

- Fisher, R. and F. Yates (1963). *Statistical tables for biological, agricultural and medical research* (6th ed.). London: Oliver & Boyd.
- Hardin, J. and D. M. Rocke (2005). The distribution of robust distances. *Journal of Computational and Graphical Statistics* 14, 910–927.
- Hoare, C. A. R. (1961). Algorithm 64: Quicksort. *Communications of the ACM* 2(7), pp. 321.
- Hössjer, O. (1992). On the optimality of S estimators. *Statistics and Probability Letters* 14, pp. 413–419.
- Huber, P. J. and E. M. Ronchetti (2009). *Robust Statistics, 2nd edition*. New York: Wiley.
- Knuth, D. E. (1997, November). *The Art of Computer Programming, Volume 2: Seminumerical Algorithms* (3rd ed.). Addison-Wesley Professional.
- Knuth, D. E. (1998, May). *The Art of Computer Programming, Volume 3: Sorting and Searching* (2nd ed.). Addison-Wesley Professional.
- Maronna, R. A., D. R. Martin, and V. J. Yohai (2006). *Robust Statistics: Theory and Methods*. New York: Wiley.
- Perrotta, D., M. Riani, and F. Torti (2009). New robust dynamic plots for regression mixture detection. *Advances in Data Analysis and Classification* 3, 263–279. doi:10.1007/s11634-009-0050-y.
- Pison, G., S. Van Aelst, and G. Willems (2002). Small sample corrections for LTS and MCD. *Metrika* 55, 111–123.
- Riani, M. and A. C. Atkinson (2000). Robust diagnostic data analysis: Transformations in regression (with discussion). *Technometrics* 42, 384–398.
- Riani, M. and A. C. Atkinson (2010). Robust model selection with flexible trimming. *Computational Statistics and Data Analysis* 54, 3300–3312. doi: 10.1016/j.csda.2010.03.007.
- Riani, M., A. C. Atkinson, and A. Cerioli (2009). Finding an unknown number of multivariate outliers. *Journal of the Royal Statistical Society, Series B* 71, 447–466.
- Rousseeuw, P. J. (1984). Least median of squares regression. *Journal of the American Statistical Association* 79, 871–880.
- Rousseeuw, P. J. (1985). Multivariate estimation with high breakdown point. In W. Grossmann, G. Pflug, I. Vincze, and W. Wertz (Eds.), *Mathematical Statistics and Applications*, Volume B, pp. 283–297. Dordrecht: Reidel.
- Rousseeuw, P. J. (1991). Tutorial to robust statistics. *Journal of Chemometrics* 5(1), 1–20.
- Rousseeuw, P. J. and J. Bassett, Gilbert W. (1990). The remedial: A robust averaging method for large data sets. *Journal of the American Statistical Association* 85(409), pp. 97–104.
- Rousseeuw, P. J., M. Debruyne, S. Engelen, and M. Hubert (2006). Robustness and outlier detection in chemometrics. *Critical Reviews in Analytical Chemistry* 36(3–4), 221–242.

- Rousseeuw, P. J. and K. Van Driessen (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics* 41, 212–223.
- Rousseeuw, P. J. and K. Van Driessen (2006). Computing LTS regression for large data sets. *Data Mining and Knowledge Discovery* 12, 29–45.
- Rousseeuw, P. J. and B. C. van Zomeren (1990). Unmasking multivariate outliers and leverage points. *Journal of the American Statistical Association* 85, 633–9.
- Salibian-Barrera, M. and V. J. Yohai (2006). A fast algorithm for s-regression estimates. *J. Computat. Graphic. Statist* 15, 414–427.
- Shankar, B. and al. (2008). Recommendations for the validation of immunoassays used for detection of host antibodies against biotechnology products. *Journal of Pharmaceutical and Biomedical Analysis* 48, 1267–1281. Gopi Shankar, Viswanath Devanarayan, Lakshmi Amaravadi, Yu Chen Barrett, Ronald Bowsher, Deborah Finco-Kent, Michele Fiscella, Boris Gorovits, Susan Kirschner, Michael Moxness, Thomas Parish, Valerie Quarmby, Holly Smith, Wendell Smith, Linda A. Zuckerman, Eugen Koren.
- Tallis, G. M. (1963). Elliptical and radial truncation in normal samples. *Annals of Mathematical Statistics* 34, 940–944.
- Torti, F. and D. Perrotta (2011). Size and power of tests for regression outliers in the forward search. In S. Ingrassia, R. Rocci, and M. Vichi (Eds.), *New Perspectives in Statistical Modeling and Data Analysis*. Heidelberg: Springer-Verlag.
- Tukey, J. W. (1978). The ninther: A technique for low-effort robust (resistant) location in large samples. In H. A. David (Ed.), *Contributions to Survey Sampling and Applied Statistics in Honor of H. O. Hartley*, pp. 251–257. New York: Academic Press.
- Verboven, S. and M. Hubert (2005). LIBRA: a MATLAB library for robust analysis. *Chemometrics and Intelligent Laboratory Systems* 75, 127–136. doi:10.1016/j.chemolab.2004.06.003.
- Verboven, S. and M. Hubert (2010). Matlab library LIBRA. *WIREs Computational Statistics* 2, 509–515.
- Yohai, V. J. (1987). High breakdown-point and high efficiency robust estimates for regression. *The Annals of Statistics* 15(2), pp. 642–656.