Review

SAISIR: A new general chemometric toolbox

Christophe B.Y. Cordella a,⇑, Dominique Bertrand b

a Institut National de la Recherche Agronomique, UMR1145 GENIAL, Equipe Ingénierie Analytique pour la Qualité des Aliments/Chimiométrie, 16 rue Claude Bernard, 75005 Paris, France
b Institut National de la Recherche Agronomique, Unité Biopolymères Interactions Assemblages, Equipe Bioinformatique, Rue de la Géraudière-BP 71627, 44316 Nantes Cedex 3, France

A R T I C L E   I N F O
Article history:
Available online 6 December 2013

Keywords:
Chemometric toolbox Data modeling Discrimination MATLAB Multiblock analysis Multidimensional analysis OCTAVE SAISIR SCILAB Regression

A B S T R A C T
The SAISIR Toolbox for MATLAB, OCTAVE and SCILAB is a freely available collection of functions and algorithms for modeling physicochemical, sensorial and multidimensional data with a large range of bilinear and multilinear models. The SAISIR toolbox can be freely downloaded at http://www.chimiometrie.fr/saisirdownload.html. SAISIR contains many functions for loading, saving, manipulating or displaying data. It is well equipped with more than 200 chemometric functions for regression, multidimensional analysis, discrimination and multiblock analysis.

© 2013 Elsevier Ltd. All rights reserved.

Contents

1. Introduction .......................................................... 75
2. Requirements ..................................................... 76
3. List of functions in categories .................................. 76
4. Understanding the SAISIR philosophy of data management .................................................. 77
5. Applications ........................................................ 77
  5.1. Input and import data ........................................ 77
  5.2. Preprocessing and mathematical models .................. 77
  5.3. Principal-component analysis .............................. 78
      5.3.1. Code ......................................................... 78
      5.3.2. Results .................................................. 78
6. Brief review of SAISIR toolbox applications .......... 79
7. SAISIR in the world today ...................................... 79
Appendix A. Code of some SAISIR functions ............... 80
References ......................................................... 82

1. Introduction

In most chemometric works, it is necessary to manipulate large data files and to have the capability to make a complex chain of processing steps. For example, simple data processing may consist in loading data, such as spectra, numeric images, and sensory data, pre-treating the data by specific relevant methods, sampling the data, and eventually processing the data using PCA, PLS, or any appropriate methods (e.g., multiblock table methods, or discrimination). Moreover, it is essential for the user to keep track of the identifiers of the rows and the columns of the data matrix. For example, when processing spectra, it is important to identify the particular role of the variables (labeled, e.g., by a wavenumber or a wavelength value). In the same way, the rows of the data matrices are generally logically identified by names making sense to the chemometrician.
When the chain of procedures is complex, it is not easy to make such a work in graphically-interfaced environments. One may want to keep track of the applied procedures and to be able to re-use them on other data sets, or in cross-validation procedures. The SAISIR (statistics applied to the interpretation of infrared spectra) environment has been developed in order to cope with such situations.

The acronym SAISIR means (in French) “to hold” but also “to understand”. Initially designed for spectral data, it has been extended as a general environment coping with many kinds of data. SAISIR contains many functions for loading, saving, manipulating or displaying data. Contrary to windowed environments, it has the great advantage of allowing batch procedures. It also makes it possible to mix data of any origin (e.g., chemical and physical data, spectroscopic data, and numeric images). Due to its simple way of working, it is always possible to find a practical solution, and the user is never at a complete standstill.

Over 200 functions are available in SAISIR involved in the linear data-processing field, such as graphics (e.g., biplot, curves, confidence ellipse, barycenter representation, and dendrogram), factorial methods (e.g., PCA, correspondence analysis, and factorial distance analysis), discriminative methods (e.g., linear, factorial, quadratic, PLS, and stepwise), and regression methods (e.g., PLS1, PLS2, PCR, ridge, latent root, and stepwise). SAISIR is particularly rich in functions dealing with multiblock tables (e.g., STATIS, ACOM (analyse en co-inertie et composante commune), ACCPS (Common component and specific weights analysis), multiple factor analysis, and canonical analysis) and more. Many of these functions are original and cannot be found elsewhere.

Many chemometric toolboxes and or modeling software (open source) are available on the Internet under GNU GPL license or equivalent and descriptions published in various journals, such as Chemometrics and Intelligent Laboratory Systems, Environmental Modelling & Software, Journal of Cheminformatics and others.

It will not be possible to list here all the toolboxes available. A rapid search especially in Chemometrics and Intelligent Laboratory Systems for the past 10 years (2003–13) lists several contributions aimed at proposing new chemometrics toolboxes for various usages. But among those developed for MATLAB, we found that they are usually specific to a particular field of chemometrics:

- 2D correlation analysis and spectral processing in biology (Biodata) [1];
- multivariate calibration technology (TOMCAT) [2];
- multivariate curve resolution [3];
- multivariate calibrations [4–6];
- experimental design (OptiCat) [7];
- counterpropagation Kohonen neural networks [8];
- related partial least squares (PLS toolbox, commercially available);
- linear discriminant analysis (Michael Kiefte Discrimin toolbox) [9];
- multiway technical processing (N-way toolbox) [10];
- robust analysis (LIBRA) [11]; and,
- FSDA [12].

The Internet is another way of accessing chemometric toolboxes and various algorithms. An interesting example is the KVL website at http://www.models.life.ku.dk/algorithms.

With the SAISIR package, we hope to complete the available offering of chemometric tools by giving access to a free, open source, general toolbox covering a large number of linear procedures. The SAISIR package is provided with supports for English courses to learn how to use the functions of the package and the associated chemometric concepts. Several “scenarios” of data processing are provided under the script-form that we tested and allow the SAISIR environment to be learned quickly.

### 2. Requirements

The collection of functions, algorithms and helper files are provided as MATLAB source files, m-files, with no requirements for any add-ins beyond the standard MATLAB installation. The toolbox has been developed under MATLAB 7.x (MathWorks) but, in this CD-ROM, all functions have been developed for MATLAB, OCTAVE (OCTAVE-Forge project http://octave.sf.net) and SCILAB (Fondation de Coopération Scientifique Digiteo, www.digiteo.fr). Typically, we suggest somewhat more than 32 MB RAM under Microsoft Windows 9x/NT/2000.

### 3. List of functions in categories

Table 1 lists the main categories of functions provided by SAISIR and the number of functions available in each category.

### 4. Understanding the SAISIR philosophy of data management

The SAISIR environment mainly relies on the processing of data tables, as matrices of real (double) numbers. The fundamental notion is that the rows and the columns of the matrices in SAISIR have identifiers (names), which must be kept in any function. Let us, for example, examine the simple data set below. It is the result of the assessment of three apples by a given panelist. The apples are identified by their cultivar names, GALA1, FUJI1 and FUJI2.

The panelist assessed three sensory features of the fruits, namely “global odor” (coded GLO), earth odor (EARTH) and fungus odor (FUNG).

$$
\begin{array}{ccc}
\text{GLO} & \text{EARTH} & \text{FUNG} \\
\text{GALA1} & 2.8 & 1.2 & 0.3 \\
\text{FUJI1} & 2.6 & 0.5 & 0.4 \\
\text{FUJI2} & 7.5 & 0.3 & 0.0 \\
\end{array}
$$

This file includes 3 rows and 3 columns. The rows (representing the observations or individuals) are characterized by the identifiers (GALA1, FUJI1 and FUJI2). In the same way, the columns (variables) are named GLO, EARTH and FUNG. Finally, the numeric data build up the matrix:

$$
\begin{array}{ccc}
2.8 & 1.2 & 0.3 \\
2.6 & 0.5 & 0.4 \\
7.5 & 0.3 & 0.0 \\
\end{array}
$$

Generally, it is necessary to keep these three pieces of information. With SAISIR, the matrix of identifiers is kept in a MATLAB
structure with fields “d” for the numerical data, “i” for the identifiers and “v” for the variables.

Let us suppose that the previous data are kept in a structure named “apple”. By typing “apple” in the command line, the latter displays:

```matlab
>> apple
apple =
d: [3x3 double]
i: [3x6 char]
v: [3x20 char]
```

It is of course easy to extract the field using the MATLAB extraction command ‘.’ (dot). For example:

```matlab
>> apple.d
GAL1
FUJI1
FUJI2
```

If you are bored with SAISIR, you can easily go out and retrieve the good old MATLAB matrices. For example:

```matlab
>> X=apple.d
X =
2.80 1.20 0.30
2.60 0.50 0.40
7.50 0.30 0.00
```

A particularly important case deals with the data that are digitized curves, such as spectra or chromatograms. In this case, the data are generally stored as matrices with the rows corresponding to the curves. The identifiers of variables are then numbers, transformed in character strings, which normally represent the X-axis labels of the curve (e.g., wavelength, retention time, and shift). Table 2 shows a simplified example of near-infrared data file.

As previously, 1br01, 1br51, 1fu21, and 1fu71 are the identifiers of individuals, whereas 1100, 1102, 1104, 1106, and 1108 are character strings identifying the variables. If the data are in a SAISIR structure, we will have:

```matlab
>> spectra
s:
  d: [4x5 double]
i: [4x31 char]
v: [5x20 char]
```

and:

```matlab
>> spectra.v
1100
1102
1104
1106
1108
```

The numerical nature of the descriptors is exploited in the function devoted to displaying curves (e.g., curve, curves, tcurve, and tcurves).

5. Applications

5.1. Input and import data

Two cases arise, depending on whether the data are already in the workspace of MATLAB, OCTAVE or SCILAB or one wants to import data from an Excel file. If the data are already available in the workspace, they must be structured as a numerical matrix with dimensions 1 x J, where I is the number of samples and J is the number of variables. When dealing with calibration data, the Y-vector must be prepared as a column numerical vector (1 x 1), where the i-th element of this vector represents the Y value of the i-th sample. The names of samples must ideally be prepared as a column cell vector containing all samples identifiers in the first column. Thereafter, when dealing with supervised data, SAISIR provides a function (create_group1) to automate the creation of groups of samples from the list of names (see Appendix). Microsoft csv EXCEL file can be imported directly in a d.i.v. structure. The first column of the XLS file must be the identifiers of the samples or can be void; the first column must be the identifiers of the variables or can be void. CSV files are imported into the workspace with the SAISIR function: excel2saisir (see Appendix for code).

5.2. Preprocessing and mathematical models

Based on its d.i.v. structure, SAISIR also provides a framework for preprocessing all types of data with several common preprocessing algorithms, including normalization, mean-centering, standard normal variate (SNV), nth derivative, and smoothing. These techniques are not discussed here.

When an extended mathematical model of a dataset is built, it is useful to be able to apply exactly the same model to unknown data. For example, when a partial least squares-discriminant analysis is computed on a dataset used as calibration set, it is usually expected that the model thus constructed can be applied to new data. For this reason, SAISIR provides several methods designed to do that, as illustrated by Table 3. All parameters of a chemometric model are stored in the field analysis of the created SAISIR structure. See Appendix for code examples of utilization of excel2saisir, create_group1 and crossplsda functions.

All functions in Table 3 should be used with the result of the function allowing the creation of the corresponding model for the calibration dataset. In the previous example, the crossplsda function was used to compute the PLS-DA model.

The application of this model to new data can be performed by using the applyplsda function that accepts as input at least 3 parameters:

- the SAISIR matrix of predictive variables;
- the SAISIR structure returned by the function crossplsda; and,
- the SAISIR vector of observed groups as optional parameter (Note: this information is accessible by typing “help applyplsda” in the command line of MATLAB, SCILAB or OCTAVE).

See Appendix for a detailed listing of the input and output parameters of the applyplsda function. This listing is extracted from the SAISIR help and the structure of this help is general for all functions of the SAISIR package.
5.3. Principal-component analysis

To illustrate the ease of use of SAISIR, an application using PCA is presented as an example of data exploration. The data set consists of a SAISIR matrix from physicochemical measurements on apricots provided by INRA Avignon [15]. These fruits, belonging to different cultivars, were collected at different stages of maturity. 731 apricot samples were analyzed for 23 parameters, as listed in Table 4.

The measured parameters included physical and rheological measurement (e.g., weight, and firmness) and color measurement in the L'ab’ color space for the faces of the fruit that were exposed (surimposed) or not directly exposed (background) to the rays of the sun. Chemical analyses were also available.

The dataset was therefore a 731 x 23 numerical matrix. The 723 observations were identified with a particular coding (in the field .i of the data structure). The first two characters indicated the nature of the cultivar, whereas the third indicated the degree of maturity, from 1 (not mature) to 4 (very mature). For example, the code “mo3” indicated that the sample belonged to the cultivar coded “mo”, at maturity “2” (intermediate).

The following code shows how to perform PCA on this dataset with graphical outputs allowing chemical interpretation.

### 5.3.1. Code

```matlab
(L1) load chemistry
(L2) mypca = normed_pca(chemistry);
(L3) mypca
(L4) mypca.score
(L5) figure; colored_map1(mypca.score,1,2,1,3);
(L6) figure; correlation_circle(mypca,chemistry,1,2);
```

### 5.3.2. Results

The first line (L1) of this code loads the dataset entitled “chemistry” in the workspace. By typing `chemistry` in the command line, the command window displays:

```matlab
d: [731x23 double] %data
i: [731x40 char] %sample names
```

Line (L2) performs a normalized PCA on “chemistry” and produces in the workspace a new variable “mypca”. The content of “mypca” is given by (L3) as follows:

```matlab
mypca =
```

```matlab
score: [1x1 struct] % Scores of samples
eigenv: [1x1 struct] % Loadings
eigenv: [1x1 struct] % Eigenvalues of the data matrix
average: [1x1 struct] % Average of observations
var_score: [1x1 struct] % Scores of the variables
std: [1x1 struct] % Standard deviation of the columns % of the matrix
```

Each field of “mypca” is a structure in the MATLAB sense that contains other fields as given by line (L4):

```matlab
mypca.score
```

```matlab
ans =
```

```matlab
d: [731x23 double]
i: [731x40 char]
v: [23x11 char]
```

(L5) and (L6) create respectively plots for scores and loadings. The result is presented in Figs. 1 and 2. In Fig. 1, the observations are colored according to their cultivar and their maturity. In this way, two observations belonging to the same cultivar at the same maturity bring the same color. Simultaneous interpretation of the scores and loadings allows us to notice that the first axis A1 (31.8% of the total variance) changes with the rate of fruit ripening. Indeed, the order of samples from left to right is clearly indicated.
on the scores plot: mo1, mo2, mo3; go1, go2, go3, 371, 372, 373, etc. The index corresponds to the age of the fruit. Meanwhile, the loadings plot shows the first axis is strongly and positively correlated with sucrose “saccharose (g/100gMF)” content and “p.c.perte”, whereas it is highly and negatively correlated with “firmness”. These parameters directly reflect the maturity of the fruits studied.

Detailed analysis showed that the second component relates to the nature of the cultivar. Observation of the correlation plot in Fig. 2 also shows that the second component is mainly correlated with the color. This simply shows that the cultivars, whatever the stage of maturity, have specific colors.

Basically, with the SAISIR syntax, just 3 lines of code are needed to perform normalized PCA and to obtain corresponding graphical outputs, whereas the user would usually write many more lines of code.

6. Brief review of SAISIR toolbox applications

The SAISIR toolbox has been used in many scientific and technological fields to study the quality of food or to follow the influence of a process on the quality of a food product, such as fruit and vegetables, tomatoes [13,14], apricots [15], apples [16–18], olives [19], carrots [20], whole fruit [21], studies of cheeses in order to characterize varieties [22], seed [23,24], wines for prediction of maturity and sensory analysis [25,26], meats and biological tissue for pathological study, muscle-type determination [27], microbiological studies and studies of polymer materials [28]. Further references are available on request.

7. SAISIR in the world today

We have distributed the SAISIR package since 2010. Today, more than 350 CD-ROMs have been freely distributed around the world (France, Spain, Italy, Portugal, The Netherlands, Belgium, Germany, Brazil, Thailand, Hong Kong, USA, Lebanon, Switzerland, Tunisia, UK, China, Japan, Morocco, India, Poland, Turkey, and Canada) and many downloads have been made from Internet. We hope to contribute to the popularization of chemometrics by facilitating access to this discipline by scientists who may not be chemometricians and to trigger some interest from the community of chemometricians.
Appendix A. Code of some SAISIR functions

(a) Function "create_group1" for the automatic creation of groups of samples

```python
groups = create_group1(SAISIR_Mat, Pos_of_1st_Letter, Pos_of_Lst_Letter);
```

With

- **SAISIR_Mat:** SAISIR Structure (with d.i.v fields)
- **Pos_of_1st_Letter:** Position of the 1st desired char in the string of the sample name
- **Pos_of_Lst_Letter:** Position of the last desired char in string of the sample name

For example, if the samples list is:

```python
SampleList =
    'Gala01'
    'Gala02'
    'Gala03'
    'Pink01'
    'Pink02'
    'Pink03'
    'Roya01'
    'Roya02'
    'Roya03'
```

The variable `groups` will contain:

```python
groups.d : [1;1;1;2;2;2;3;3;3]
groups.i : Gala01
           Gala02
           Gala03
```
(b) Function “excel2saisir” to import Excel csv data file

```matlab
[Output_Variable] = excel2saisir (filename, nchar (optional), start(optional), xend(optional))
```

Input arguments:
- **filename**: (string) name of the Excel file in .csv format
- **nchar** : (integer, optional) number of character kept in the identifiers (default: 20)
- **Start** : (integer, optional) Index of the beginning of the observations to be loaded
- **xend** : (integer, optional) Index of the final of the observations to be loaded (greater than start)

(c) Example of pretreatments and mathematical computing model

This example can be utilized directly in MATLAB/OCTAVE or SCILAB after setting up the SAISIR package.

SAISIR functions are in bold font. “Spectra” dataset is one of the numerous datasets provided with the SAISIR package for learning; it contains 500 infrared spectra of apple fruits of various maturities. Lines beginning “or” are preceded by “%” and are code comments that will not be interpreted by MATLAB, SCILAB or OCTAVE.

```
Spectra = excel2saisir(‘spectra’); % Load data from XLS file
My_SNVData = snv(Spectra); % SNV pretreatment
groups = create_group1(Spectra,1,1); % group of maturity of the fruit
Maxdim = 20; % here it is needed to scan the dimensions
clear nscorrect;
partition = 1/3; % proportion calibration/validation
[n,p] = size (mySNVData.d);
% the random_select function fixes samples with 0 = calibration, 1 = % validation, the process is randomly performed
Selection = random_select(n, round (n*partition));
for ndimension = 1:Maxdim
    my_plsda = crossplsda(mySNVData, groups, ndimension, selection);
    nscorrect (ndimension) = my_plsda.nscorrect1;
end;
```

(d) Example of pretreatments and mathematical computing model

The `my_plsda` output variable contains results and all parameters of the PLS-DA model:

```
confusion1 <1 x 1 struct>
d <4 x 4 double>
i <4 x 10 char>
v <4 x 10 char>
nincorrect1 333
ncorrect1 167
sconfusion1 <4 x 4 double>
confusion <1 x 1 struct>
d <4 x 4 double>
i <4 x 10 char>
v <4 x 10 char>
ncorrect 333
ncorrect 167
sconfusion <4 x 4 double>
info ‘no index: max of predicted Y; 1: mahalanobis distance on latent’
```

This model can then be applied to new data using the “applyplsda” function described below:

```
[Valid_plsda] = applyplsda(X, plsdatype, actual_group (optional))
% return the predicted group on (unknown data X)
% Input arguments:
% ===============
% X : SAISIR matrix of predictive variables
% plsdatype : structure returned by function ‘plsda’
% actual_group (optional): SAISIR vector of observed groups.
% Observations with the same group number belong to the same group.
% Output argument:
% ==============
% Valid_plsda contains following fields:
% confusion1 : matrix of confusion, method 1 (if ‘actual_group’ defined)
% ncorrect1 : Number of correct classifications, method 1 (if ‘actual_group’ defined)
% predgroup1 : predicted group (method1)
% confusion : matrix of confusion, method 0 (if ‘actual_group’ defined)
% ncorrect : Number of correct classifications, method 0 (if ‘actual_group’ defined)
% predgroup : predicted group (method0)
% Method 1 : (attribution to index of max of predicted Y)
% Method 0 : (shortest Mahalanobis distance calculated on PLS scores)
```