Chemometrics and Intelligent Laboratory Systems 167 (2017) 23–28

Contents lists available at ScienceDirect

Chemometrics and Intelligent Laboratory Systems

journal homepage: www.elsevier.com/locate/chemometrics

Software description

DD-SIMCA – A MATLAB GUI tool for data driven SIMCA approach

Y.V. Zontov [a, e, *], O.Ye. Rodionova [b], S.V. Kucheryavskiy [c], A.L. Pomerantsev [b, d]

[a] National Research University Higher School of Economics, Moscow, Russia
[b] Semenov Institute of Chemical Physics RAS, Moscow, Russia
[c] Aalborg University, Esbjerg, Denmark
[d] Branch of Institute of Natural and Technical Systems RAS, Sochi, Russia
[e] Federal Institute of Industrial Property (FIPS), Moscow, Russia

1. Introduction

Authentication problems, where the main goal is to determine whether an object is, in fact, what it is declared to be, are in high demand in such areas as counterfeit drug detection, food adulteration, etc. Methods used for solving these problems are called class modeling methods or one-class classifiers (OCC) [1,2]. These methods are meant to distinguish objects of one particular class, also called target class, from all other objects and classes. This approach is similar to non-targeted analysis of adulterants as the acceptance boundaries are developed on the authentic samples purely without the necessity of known adulterant training samples.

Soft independent modeling of class analogy (SIMCA) is one of the OCC methods widely used in chemometrics [3]. The original version, proposed by S. Wold [4], has numerous modifications mostly related to the way of constructing the acceptance boundaries [5–7]. Recently, authors proposed the data driven version of SIMCA (DD-SIMCA), which allows to calculate the errors of misclassification theoretically [8,9].

In this paper, we present DD-SIMCA — a MATLAB GUI tool that extends the MATLAB® environment to provide an easy way for establishment and employment of the data driven SIMCA technique.

2. DD-SIMCA algorithm and implementation

In this section, we briefly present the main steps of the classification algorithm and the way it is implemented in the DD-SIMCA tool. More detailed description and comparison with other modifications of SIMCA method can be found elsewhere [8,9].

The first step is decomposition of a training ($I \times J$) data matrix $\mathbf{X}$ by the Principal Component Analysis (PCA):

$$X = TP^t + E \quad (1)$$

where $\mathbf{T} = \{t_{ia}\}$ is the ($I \times A$) scores matrix; $\mathbf{P} = \{p_{ja}\}$ is the ($J \times A$) loadings matrix; $\mathbf{E} = \{e_{ij}\}$ is the ($I \times J$) matrix of residuals; and A is the number of principal components (PC).

In the second step, the score distance (SD), h_i, and the orthogonal distance (OD), v_i are calculated for each training sample using the results of PCA decomposition,

$$h_i = \mathbf{t}_i^t(\mathbf{T}^t\mathbf{T})^{-1}\mathbf{t}_i = \sum_{a=1}^{A}\frac{t_{ia}^2}{\lambda_a}, \quad v_i = \sum_{j=1}^{J} e_{ij}^2 \quad (2)$$

where λ_a, $a = 1,..,A$ are the diagonal elements of matrix $\mathbf{T}^t\mathbf{T} = \mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_A)$.

The total distance for the sample is calculated as

$$c = N_h\frac{h}{h_0} + N_v\frac{v}{v_0} \propto \chi^2(N_h + N_v) \quad (3)$$

where parameters v_0 and h_0 are the scaling factors, N_h and N_v are the numbers of the degrees of freedom (DoF). These parameters are unknown *a priori*, and they are estimated using a data driven approach explained in Refs. [8,9].

Third step defines the acceptance area or thresholds for the target class. Given the type I error α, the acceptance area is determined as

$$c \le c_{crit}(\alpha), \quad (4)$$

where

$$c_{\mathrm{crit}} = \chi^{-2}(1 - \alpha, N_h + N_v) \quad (5)$$

is the (1−α) quantile of the chi-squared distribution with $N_v + N_h$ DoF [9].

After this step, the model is ready for classification of new samples and can be represented by an acceptance area in the orthogonal vs. score distance (Acceptance plot) defined for the given α-value. The α-value specifies a type I error, i.e. a share of the false negative decisions. Each sample from the training set is characterized by its position in the Acceptance plot and has a status either of a '*regular*' sample, i.e. a sample

* Corresponding author.
E-mail address: yury.zontov@gmail.com (Y.V. Zontov).

http://dx.doi.org/10.1016/j.chemolab.2017.05.010
Received 6 March 2017; Received in revised form 7 May 2017; Accepted 8 May 2017
Available online 19 May 2017

attributed to the target class, or an '*extreme*' sample, which is located out of the acceptance area and is determined as an alien (a non-member). Besides that, a second cut-off level is determined to be used as the outlier border constructed for the given γ-value. This value specifies the probability that at least one regular object from the data set will be erroneously considered as an outlier. Unlike the acceptance area, the outlier area depends on the size of a training set *I*. For a specific γ- value, the greater *I*, the farther the outlier area. For moderate dataset, a common value of γ equals 0.05 or 0.01.

In addition to the Acceptance plot, a special Extreme plot [9] is created. This plot demonstrates the dependence of the observed number of the extreme samples versus the theoretically expected values, calculated as $n = \alpha I$. This dependence is shown in the plot together with the tolerance limits calculated as

$$t_\alpha = n \pm 2\sqrt{\alpha(1-\alpha)I} = n \pm 2\sqrt{n(1-n/I)} \quad (6)$$

The extreme plot helps to analyze the quality of the classification model for the chosen number of PCs.

The last step is classification of test or new objects employing the established model. Classification results are presented in the corresponding Acceptance plot. Additionally, the value of the type II error β, which is the rate of wrong acceptances of aliens as target objects [10] is calculated. It is also possible to perform a reverse evaluation, namely to calculate the type I error α, which corresponds to a given value of the type II error β.

The algorithm and graphical user interface (GUI) were implemented using the MATLAB Object-oriented programming (OOP). The main class, *DDSimca*, contains the implementation of SIMCA and auxiliary algorithms. The instance of this class contains fields, which represent the actual DD-SIMCA model. The class *DDS Task is* used to represent classification results for new data. The code has been tested on MATLAB versions R2010a and R2015b.

The graphical user interface provides an easy way to build DD-SIMCA models and to apply them for classification of test and new samples. Experienced users can also exploit the command-line version that provides numerous additional options, which are documented and exemplified. The DD-SIMCA tool relies on its own implementation of statistical functions and thus does not address the MATLAB Statistics Toolbox (and any other toolboxes).

3. Working with the GUI tool

To illustrate the functionality of the DD-SIMCA Tool we will use a subset (three classes: A4, A5, A6) of the data set Amlodipin [11]. The original data contains NIR spectra of uncoated intact tablets acquired through the PVC blisters. Spectra are preprocessed using the second order Savitzky-Golay differentiation. Tablets in all three classes (the classes correspond to three different manufactures) contain the same amount of identical active pharmaceutical ingredient (API). Each class is presented by 50 spectra (10 spectra from 5 different batches), which are divided into two parts: the training (40 samples) and the test (10 samples) subsets. For illustrations, we use class A6 as the target class (40 spectra for training and 10 spectra for validation) and test spectra from the other two classes, denoted as 'Test A4_A5', for showing how the model works with new samples.

The *DDSGUI* module provides the graphical user interface (GUI) to the tool. Before starting the module, ascertain that all necessary data are loaded into the standard MATLAB workspace. The datasets should be loaded as matrices while the optional text labels for the samples should be represented as cell arrays of strings. The number of elements in label array should correspond to the number of rows in the corresponding data matrix.

The GUI window consists of three panels that can be switched using tabs *Model*, *Prediction* and *Options*.

3.1. The model panel

Fig. 1 depicts the main panel, which contains the user controls for input of the training data set, adjusting the model parameters, viewing and saving the training results. The user controls are grouped into several sections based on their purpose and importance.

The controls in *Data sets* section are used to load data from the MATLAB workspace into the GUI. The «**Training Set**» button invokes a dialog window for selection of the data. The list of datasets includes only arrays that are available in the workspace. The «**Labels**» button invokes a dialog window for selection of a cell array with labels. After the data is loaded, the indicator under the corresponding button is changed from «Not selected» to the actual dimensions of the array (i.e. «[40 × 72]»). The Labels array is optional and is used for output results in the Acceptance plot. If labels are not available, each sample is indicated by its sequential number.

The input data can be column-wise centered and/or scaled by the standard deviation using checkboxes in the *Preprocessing* section.

The *Model parameters* section provides a possibility to choose the main parameters, such as number of the PCs, value of the type I error, significance level for outliers, a type of the acceptance area, and a method for estimation of the chi-square distribution parameters.

The type I error is an important parameter, crucial for construction of the acceptance area. The «Auto » checkbox provides an option to calculate α in such a way that all training samples are attributed as the target class members and thus are located inside the acceptance area. The outlier significance level allows to change the outlier area (indicated by the red curve on the Acceptance plot).

The program provides a possibility to switch between two types of acceptance area — «chi-square» and «rectangle». The first type is a triangular area [8], constructed using the sum of the normalized SD and OD given in Eq. (3), while the second type represents the classical area, where the cut-off levels are constructed for each type of distance independently.

Two methods of estimation of parameters of the chi-square distribution for the OD and SD distances are available. «Classical» method is based on the method of moments. The «Robust» method [9] takes into account possible outliers in the PCA model. The robust method is recommended in the initial stage of analysis, when data may be contaminated by outliers.

A group of buttons at the bottom left part of the Model panel provides user controls for the most important tasks: building the model based on the selected parameters, showing windows with plots, saving and loading the model, clearing the current model, and resetting the parameters to the default values.

The right part of the panel contains the results of modeling for the data under consideration. The *Current model* section contains information regarding a newly developed or loaded model, including the predefined α-value, the number of PCs, the method used for estimation of the chi-square parameters, the type of preprocessing, the form of the acceptance area, and the estimated values for the DoF for the orthogonal and score distances.

The choice of the number of PCs highly depends on the ultimate goal and the quality of the data at hand. A user can easily change the number of PCs and the value of the type I error comparing the results by building acceptance plots. Discussion regarding the "optimal" dimensionality for one-class classifies is presented in Ref. [12].

The *Results and statistics* section represents the total number of samples used for the model building, as well as the number of samples, which appear to be extremes or outliers based on the current set of the model parameters. In case of extreme or outlying samples, a special warning "*Extreme objects in the training set!*" is displayed.

For the training data set, *a posteriori* sensitivity can be calculated using results from the «*Result and Statistics*» section as

$$\text{Sensitivity} = 100\%(\text{Samples} - \text{Extremes})/\text{Samples} \quad (7)$$

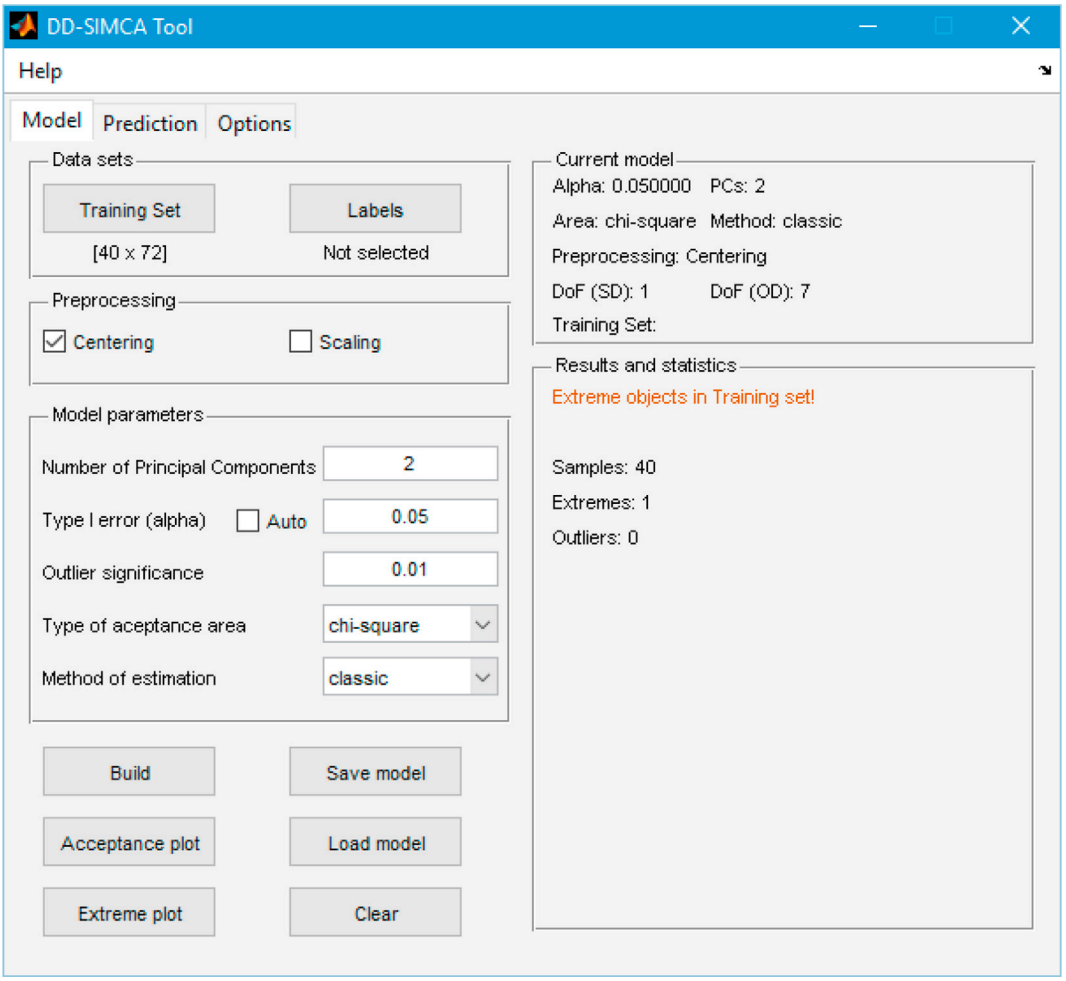


Fig. 1. Model panel.

It is recommended to compare *a posteriori* sensitivity with the predefined α-value. In the example, α value is set to 0.05. This means that the cut-off level is calculated assuming that 5% (out of forty training objects) are expected to be extremes. As can be seen from Fig. 1, only one object is identified as an extreme. Thus *a posteriori* sensitivity equals 97.5%, which is close to *a priori* estimate.

The results of modeling can be visualized using the «**Acceptance plot**» button and the «**Extreme plot**» button (see section 3.4).

The constructed model may be saved to the MATLAB workspace and loaded from it. The corresponding buttons are located in the lower left-hand area of the *Model* panel. The default name for the model is *DD_SIMCA*. The appropriate name may be chosen (the whitespaces are not allowed). The model is an instance of *DDSimca* class and can be saved as a ***.mat** file for the future employment.

3.2. The Prediction panel

The *Prediction* panel allows to apply the constructed model to new objects. The layout and most of the controls are similar to the *Model* panel (Fig. 2). The *Current model* section contains information regarding the parameters of the current classification model. This information duplicates the information shown in the similar section of the *Model* panel.

For prediction, the new or test dataset should be loaded from the MATLAB workspace using button «**New set**» and, optionally, using button «**Labels**» in the same way as it is done for the training data. The «**Decide**» button applies the current model to the new data. The results of classification are visualized using the Acceptance plot (see section 3.4) while corresponding performance statistics are shown in the *Results and statistics* section. By default, the value of the type II error, β, for the new set is calculated. Additionally, the overall number of new samples and the number of samples that are considered as extraneous regarding the current model as well as the chosen α value are reported. If a new dataset consists of several subsets, then the β-value is calculated for the subset that is the closest to the acceptance area and other subsets are not taken into account [10]. In this case the corresponding warning «*The New Set contains more than one class of external objects!*» is displayed. Optionally, it is possible to select the «*Calculate type I error*» check-box and to input a desired β-value that will be used for the corresponding α-error calculation.

If the new set is a test set, with samples from the target class, the sensitivity can be calculated in the same way as for the training set (see Eq. (7)). In the other cases the specificity is calculated as:

$$\text{Specificity} = 100\%(\text{External objects}/\text{Samples}) \quad (8)$$

In the current example, objects from two alien classes are combined into the 'new set'. DD-SIMCA classifies them as extraneous objects; therefore the *a posteriori* specificity is equal to 100%.

The results may be saved as an instance of the *DDS Task* class in the MATLAB workspace using the «**Save results**» button. The default name is *DDS_TASK*. The results from the workspace may be saved as a **.mat** file for further usage. It is important to mention that the *DDS Task* instance

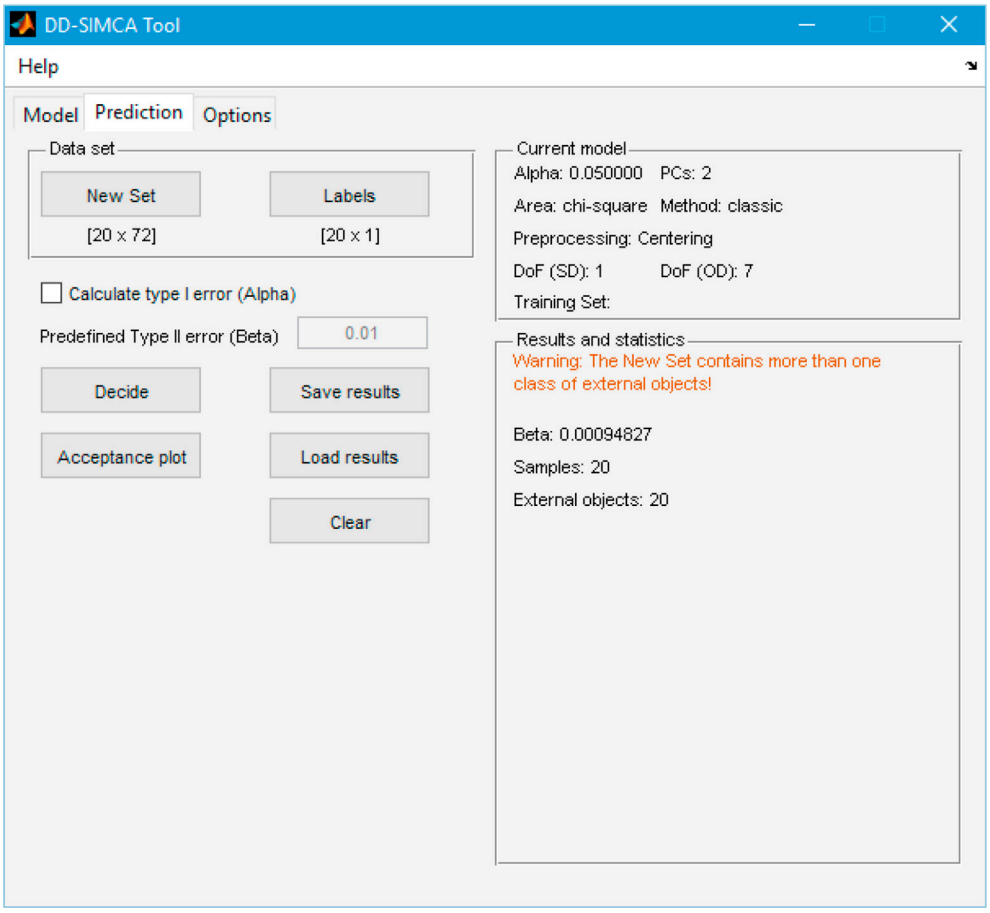


Fig. 2. Prediction panel.

already contains the *DDSimca* model.

A previously saved the *DDSTask* object can be loaded from the workspace to the GUI using the «**Load results**» button.

3.3. Options panel

The Options panel (Fig. 3) contains various settings for the Acceptance plot, such as axes transformation and sample names. By default, the results in the Acceptance plot are presented using the logarithmic scale for better visualization.

$$x_{tr} = \ln(1 + x/x_0) \quad (9)$$

This transformation may be revoked by selecting '*none*' in the *Axes transformation* dropdown list. The checkboxes in the *Options* panel allow selecting whether the labels of the samples should be shown in the plot.

3.4. Data visualization

The DD-SIMCA Tool provides two ways to represent the classification results graphically - the Acceptance and the Extreme plots. The Acceptance plot for the Training set (Fig. 4a) provides a graphic representation of the acceptance area, the area inside the green curve, outlier cut-off (red curve), regular samples (green dots), extremes (yellow squares), and outliers (red squares) for the samples used for model building. If the checkbox "Show sample labels for Training Set" is inactive (as shown in Fig. 3), markers in Fig. 4a are shown without labels.

The Acceptance plot for the new set (Fig. 4b) provides a graphical representation of the decision area (green curve), samples that fit to the model (green dots) and samples considered as non-members (red dots).

Fig. 4 clearly depicts several groups of alien objects. Returning back to the *Results and statistics* section (Fig. 2) we can see, that the program has found inhomogeneity in the 'new set' and provides the corresponding

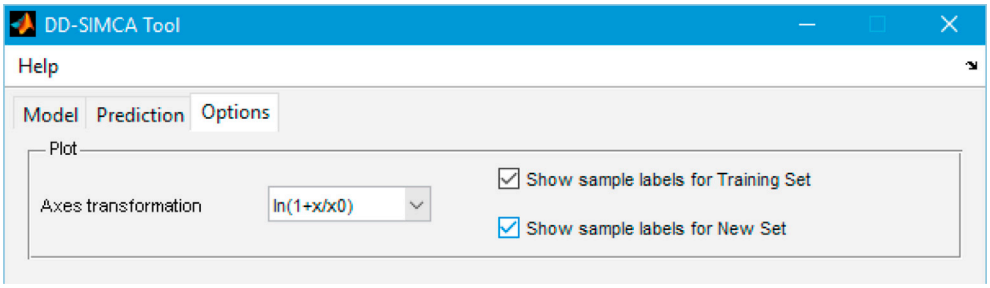
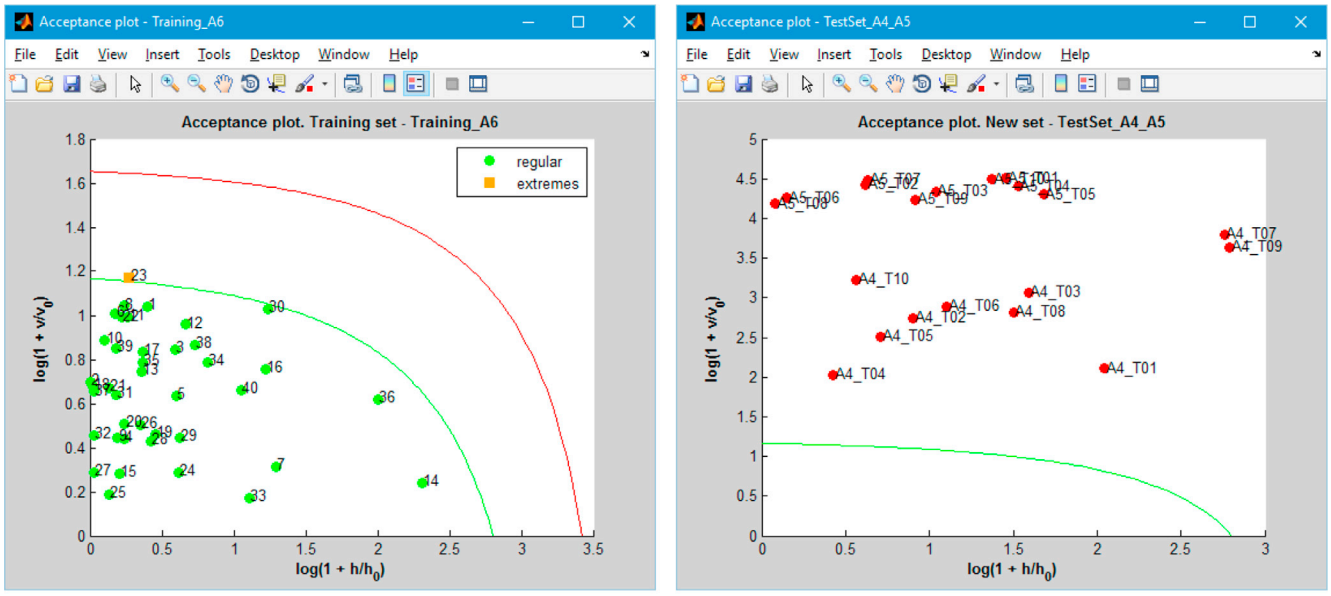


Fig. 3. Options panel.



a) Training set A6 b) New set, Test set A4_A5

Fig. 4. Acceptance plots for target class A6, α = 0.05, 2PCs.

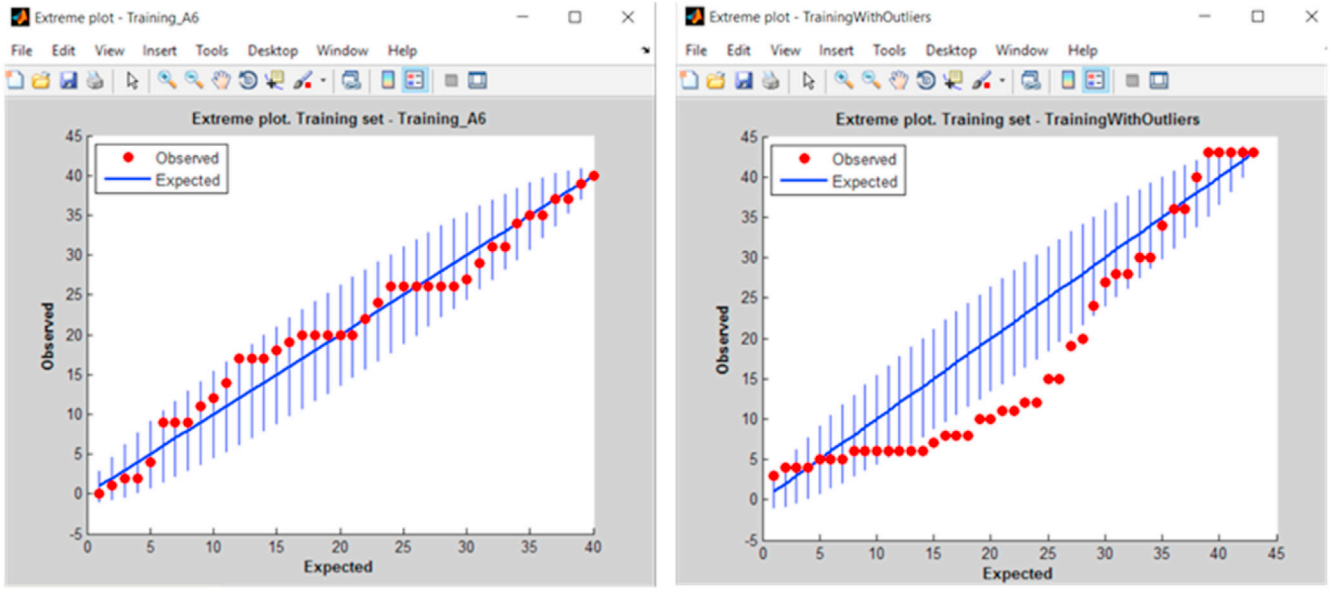
warning. Thus, the estimation of the β-value is given for the nearest subset only. So, there is 0.01% chance that alien object(s) from class A4 can wrongly be attributed to class A6.

Extreme plot (Fig. 5a) is built using training set samples. It allows checking the training data set for the presence of possible outliers. One can say that something is wrong with the data if the number of observed extreme objects does not fall inside the tolerance area (shown in vertical lines). In Fig. 5a we can see that training set A6 does not contain outliers for the model with two PCs.

An example of the Extreme plot for data contaminated with outliers is presented in Fig. 5b. This illustrative dataset is named "training_with_outliers.mat". It is worth mentioning that the more PCs are applied the more 'ideal' is the Extreme plot (red dots are closer to the diagonal). This can be explained by the fact that complex model uses some PCs for description of the outliers.

4. Conclusions

The DD-SIMCA Tool implements all the features of the Data-Driven SIMCA method in MATLAB scripting language. The tool, as well as a demonstration dataset, are freely available via GitHub [13] or via the supplementary materials.



a) Training set A6, 2 PCs. b) Set with outliers. 2 PCs.

Fig. 5. Extreme plot.

The authors are grateful to the program testers. Their technical remarks have been corrected in the stage of manuscript/software revision. General comments and proposals for further possible software improvements are presented below.

5. Validation

Ph.D. Marina Cocchi[1]
[1] Università degli Studi di Modena e Reggio Emilia, Department of Chemical and Geological Sciences.

I tested the software on Mac OS X 10.11.6 Matlab R2015b. No bugs or malfunctioning were encountered. The interface is easy to use, the description provided in the paper and the help in the GUI are clear for the users.

Possible improvements:

1. It would be nice to load data from a folder different from the working one.
2. A nice future improvement would be to show as well the scores and loadings plot for the class model inside the interface.

Ph.D. Sergei Zhilin[1,2]
[1] CSort LLC, R&D Department, Barnaul, Russia.
[2] Altai State University, Department of Informatics, Barnaul, Russia.

The authors present the graphical user interface for their updated MATLAB program implementing Data-Driven SIMCA method (DD-SIMCA).

Graphic interface to DD-SIMCA provides the following functionalities: loading of training/test sets from MATLAB environment, one-class classifier model building, statistics output, visualization of diagnostic plots, prediction of class membership and test objects statuses (inlier, extreme or outlier), saving/loading of models and predictions to disk. Before model building, a user is able to specify the number of principal components, levels of type I error and outlier significance. Classical and robust estimation method and two types of acceptance area (rectangle and chi-square) are implemented, so one can compare the results of modeling by classical SIMCA and its data-driven variant which treats extreme objects and outliers more thoroughly.

A user can visually investigate objects statuses using acceptance plots for training and test sets. Two-level decision area shown on the plot can be based on the rectangle form or on chi-square curve. On default, axes have the logarithmic scale but this transformation can be turned off in order to use usual space of normalized score and orthogonal distances. If needed, samples on plots may be equipped with labels.

Extreme objects and its relation to the theoretical tolerance area can be shown on the extreme objects plot, which is useful for visual detection of potential outliers in a training set and estimation of outliers and extreme objects percent.

The new version of the code of the DD-SIMCA procedure lying behind the GUI is refactored to an object-oriented paradigm. It makes an employment of the procedure by third-party developers easier and more natural. Other valuable modifications of the program are aimed at the computations speeding-up. Most of the time-consuming steps are optimized for speed, so a processing of large datasets (e.g., derived from hyperspectral images) is available now.

It is worth to note that comprehensive help pages describing GUI and inner structure of MATLAB classes are provided with the program.

The new version of DD-SIMCA program provides the possibilities for comfortable use both in interactive and programmatic mode.

References

[1] D. Tax, One-class Classification: Concept-Learning in the Absence of Counter-Examples, Doctoral Dissertation, University of Delft, The Netherlands, 2001.
[2] O.Ye Rodionova, A.V. Titova, A.L. Pomerantsev, Discriminant analysis is an inappropriate method of authentication, Trends Anal. Chem. 78 (4) (2016) 17–22.
[3] T. Naes, T. Isaksson, T. Fearn, T. Davies, Multivariate Calibration and Classification, Wiley, Christerer, 2002.
[4] S. Wold, Pattern recognition by means of disjoint principal components models, Pattern Recognit. 8 (1976) 127–139.
[5] P. Nomikos, J.F. MacGregor, Multivariate SPC charts for monitoring batch processes, Technometrics 37 (1995) 41–59.
[6] M. Hubert, P.J. Rousseeuw, K. Vanden Branden, ROBPCA: a new approach to robust principal component analysis, Technometrics 47 (2005) 64–79.
[7] M. Daszykowski, K. Kaczmarek, I. Stanimirova, Y. Vander Heyden, B. Walczak, Robust SIMCA-bounding influence of outliers, Chemom. Intell. Lab. Syst. 87 (2007) 95–103.
[8] A.L. Pomerantsev, Acceptance areas for multivariate classification derived by projection methods, J. Chemom. 22 (2008) 601–609, http://dx.doi.org/10.1002/cem.1147.
[9] A.L. Pomerantsev, O.Ye Rodionova, Concept and role of extreme objects in PCA/SIMCA, J. Chemom. 28 (2014) 429–438, http://dx.doi.org/10.1002/cem.2506.
[10] A.L. Pomerantsev, O.Ye. Rodionova, On the type II error in SIMCA method, J. Chemom. 28 (2014) 518–522, http://dx.doi.org/10.1002/cem.2610.
[11] Y.V. Zontov, K.S. Balyklova, A.V. Titova, O.Ye. Rodionova, A.L. Pomerantsev, Chemometric aided NIR portable instrument for rapid assessment of medicine quality, J. Pharm. Biomed. Anal. 131 (2016) 87–93, http://dx.doi.org/10.1016/j.jpba.2016.08.008.
[12] O.Ye. Rodionova, P. Oliveri, A.L. Pomerantsev, Rigorous and compliant approaches to one-class classification, Chemom. Intell. Lab. Syst. 159 (2016) 89–96, http://dx.doi.org/10.1016/j.chemolab.2016.10.002.
[13] Github https://github.com/yzontov/dd-simca.git.