COMPACT - Comparative Package for Clustering Assessment (Version 2.2)

Changes to the original Compact version by Roy Varshavsky were implemented by Guy Shaked, February 2012.

Outline

COMPACT is a GUI Matlab tool that enables an easy and intuitive way to compare some clustering methods.

COMPACT is a five-step wizard that envelops some basic Matlab clustering methods and introduces the Quantum clustering algorithm that was originally proposed by Prof. David Horn and Assaf Gottlieb (see The Method of Quantum Clustering, for more details). ***COMPACT*** has introduced the Support Vector Clustering algorithm proposed by Asa Ben-Hur, David Horn, Hava Siegelmann and Vladimir Vapnik (see Support Vector Clustering, for more details). Here, in ***COMPACT 2.2*** a new imp[lamentation of SVC is presented.

COMPACT provides a flexible and customizable interface for clustering data with high dimensionality.

COMPACT allows both textual and graphical display for the clustering results.

How to install

COMPACT is a self-extracting package. In order to install and run the ***COMPACT*** tool, follow these three easy steps:

1. Download the novel (version 2.2) package to your local drive.
2. The older version is available here: COMPACT2.0.zip
3 Add the ***COMPACT*** destination directory to your Matlab path.
4. Within Matlab, type 'compact' at the command prompt.

Steps

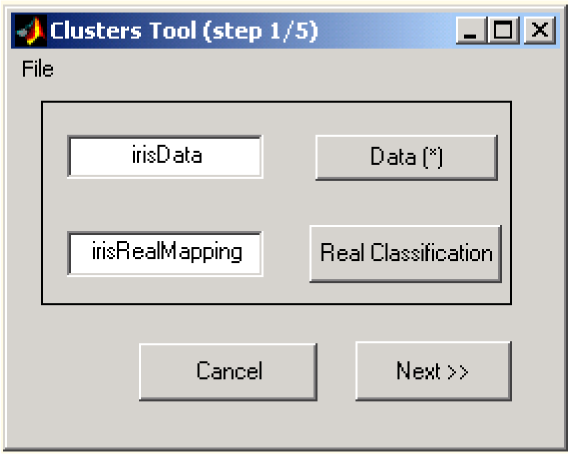
In order to run the ***COMPACT*** properly, you should follow five steps. These steps are mandatory and cannot be skipped. However, in each one of the steps you can return to the previous step(s) and change your settings.

1. Input parameters

COMPACT receives two input parameters, which are Matlab base workspace variables (i.e., must be defined in Matlab workspace before running the ***COMPACT***).

- Data (Mandatory field) - two-dimensional matrix of doubles. Represents the elements (objects to be clustered can be either columns or rows).
- Real classification (Optional field) - one-dimensional vector. Represents the real classification of the elements (i.e., the class of the *i-th* element appears in the *i-th* place in the vector). Therefore, the vectors' length must be equal to the number of elements in the Data matrix (an error occurs otherwise).

The two input parameters can be either typed in or selected from the base workspace.



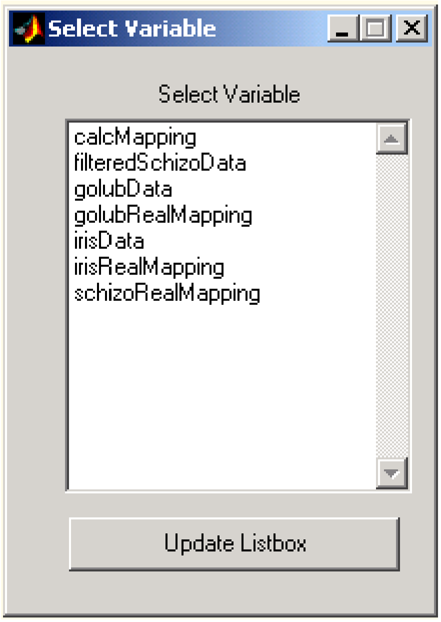
irisData

irisRealMapping

Figure 1: Input parameters

a. Selecting variables from base workspace

Press the 'Data' or the 'Real Classification' button and double click the required variable from the list in the 'Select Variable' dialog.



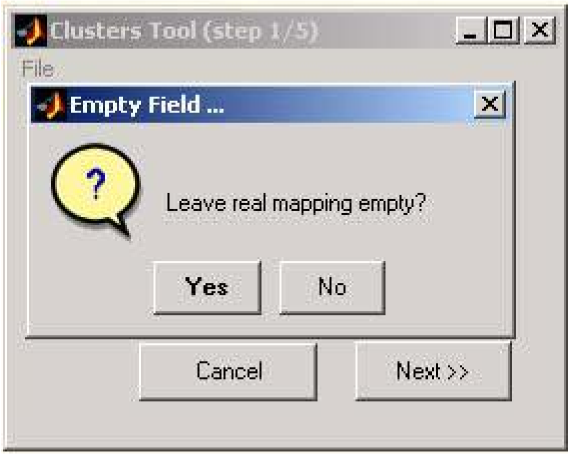
calcMapping
filteredSchizoData
golubData
golubRealMapping
irisData
irisRealMapping
schizoRealMapping

Figure 2: Variables selection dialog

As explained above, the 'Real classification' parameter is optional. If this field remains empty, a popup question appears.

b. Skipping Real Classification input

If you leave the real classification empty, you will be prompted and asked whether to leave the real mapping empty (press 'Yes' button) or specify this input ('No' button). In case where real classification isn't defined, graphical displays and clustering evaluations will be modified and limited (see steps 4 and 6, respectively).



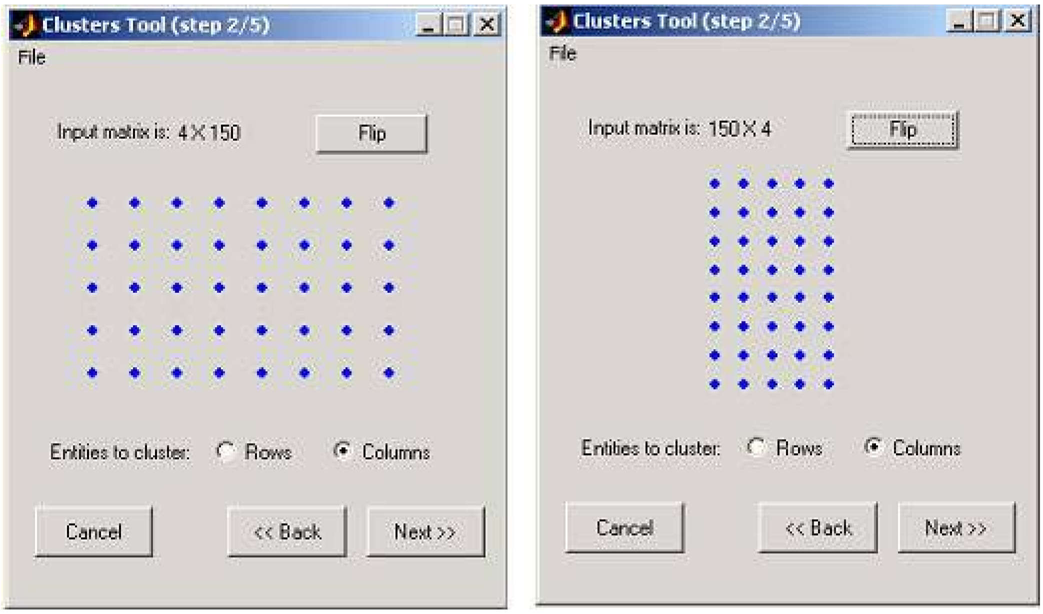
Leave real mapping empty?

Figure 3: Missing real classification field

2. Determining the matrix shape and vectors to cluster

COMPACT reads the Data matrix and displays its schematic shape. You will be asked to select the entities to cluster: rows or columns. When rows are selected, each entity is a row with n(columns) dimensions (and vice versa).

For your convenience, a flip option is available. If you choose to flip the matrix, the schematic display will change accordingly and the Data matrix will be transposed.



Input matrix is: 4 X 150

Entities to cluster: ○ Rows ● Columns

Input matrix is: 150 X 4

Entities to cluster: ○ Rows ● Columns

Figure 4: Matrix shape determination

3. Preprocessing Procedures

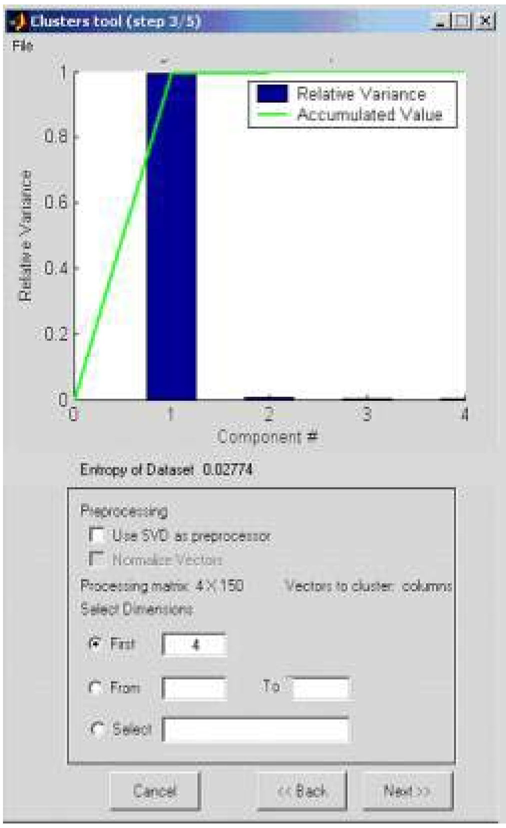
The preprocessing window consists of two sections:

a. **Components' variance graphs** the relative and accumulated variances of the principal components as processed by the SVD (singular values) method, are plotted. The variance of component i (V_i) is defined as:

$$V_i = \frac{s_i^2}{\sum_j s_j^2}$$

b. **Preprocessing parameters -**

- Use SVD method - yes/no check box.
- Normalized selected vectors - yes/no check box (available only when SVD option is selected).
- Relevant dimensions - you can select either the first *x* dimensions (first field), a range of dimensions (second field: from dimension #*i* to dimension #*j*) or a selection of dimensions (e.g., 1, 3-5, 7).



Entropy of Dataset 0.02774

Preprocessing

□ Use SVD as preprocessor

□ Normalize Vectors

Processing matrix: 4 X 150   Vectors to cluster: columns

Select Dimensions

● First 4

○ From   To

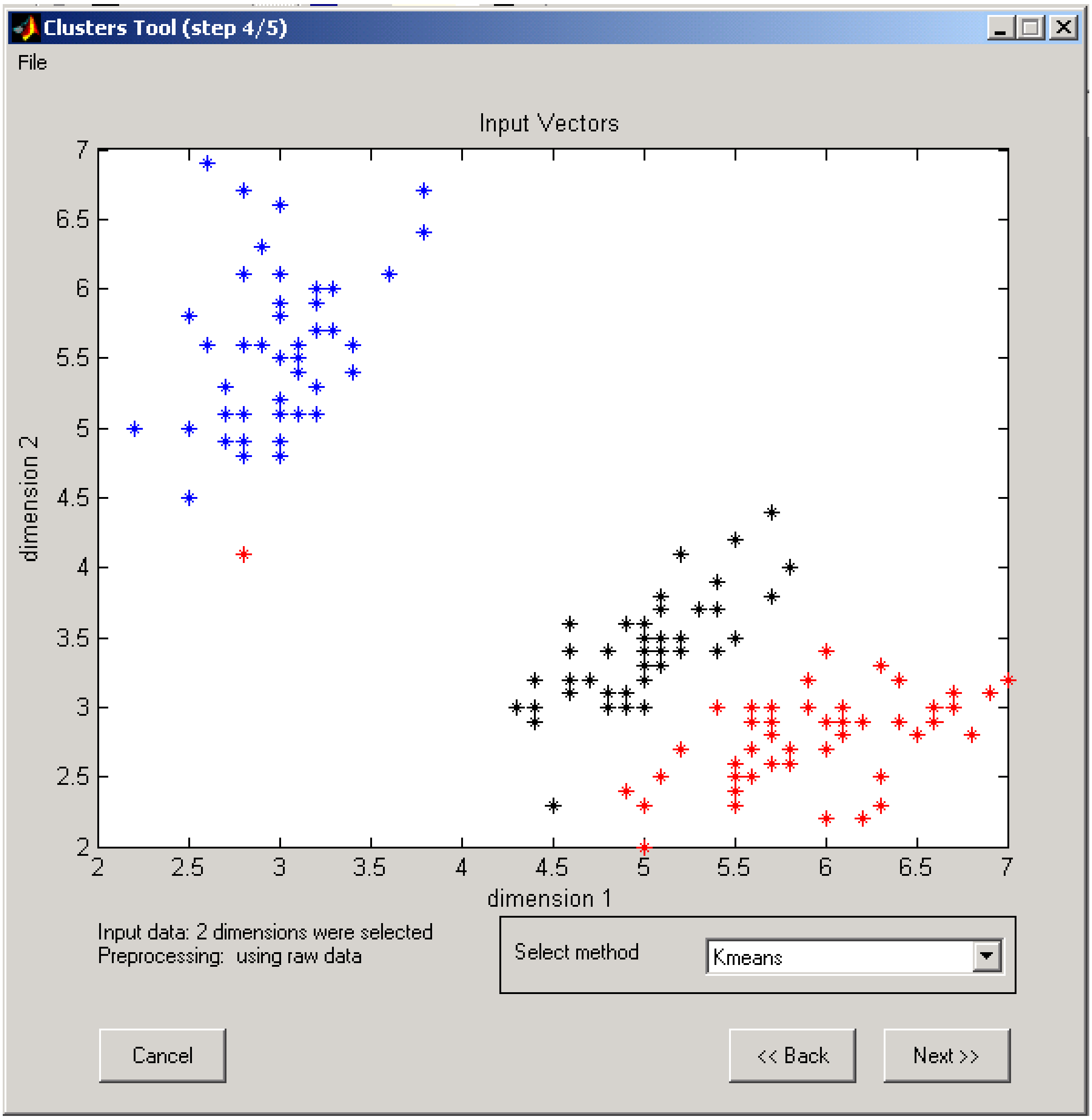
○ Select

Figure 5: Preprocessing procedures

4. Points distribution preview and clustering method selection

The elements of the Data matrix are plotted (3D plot of the first 3 selected dimensions or 2D plot when only 2 dimensions are chosen). Each original class is displayed in a different color. The original classification is taken from the 'Real classification' input parameter (see step 1). If no 'Real classification' is available, all points are plotted in black.

At this stage you are asked to select the clustering method: k-means, fuzzy c-means, competitive neural network, QC (Quantum Clustering algorithm) or SCV (Support Vector Clustering).



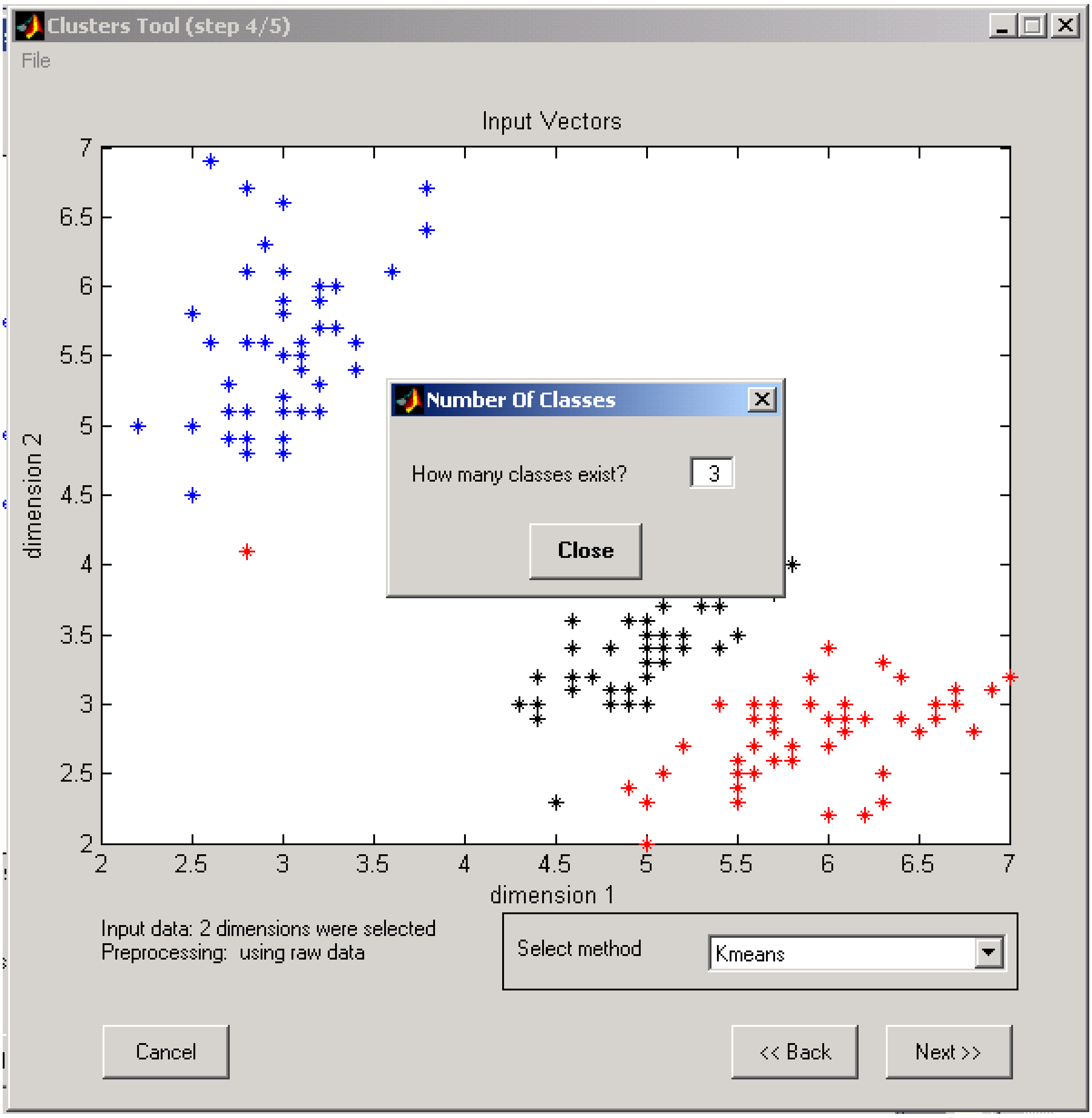
Input data: 2 dimensions were selected
Preprocessing: using raw data

Select method: Kmeans

Figure 6: Clustering method determination

5. Parameters for clustering algorithms

a. Parameter for K-means algorithm (Matlab kmeans function) - number of clusters



How many classes exist? 3

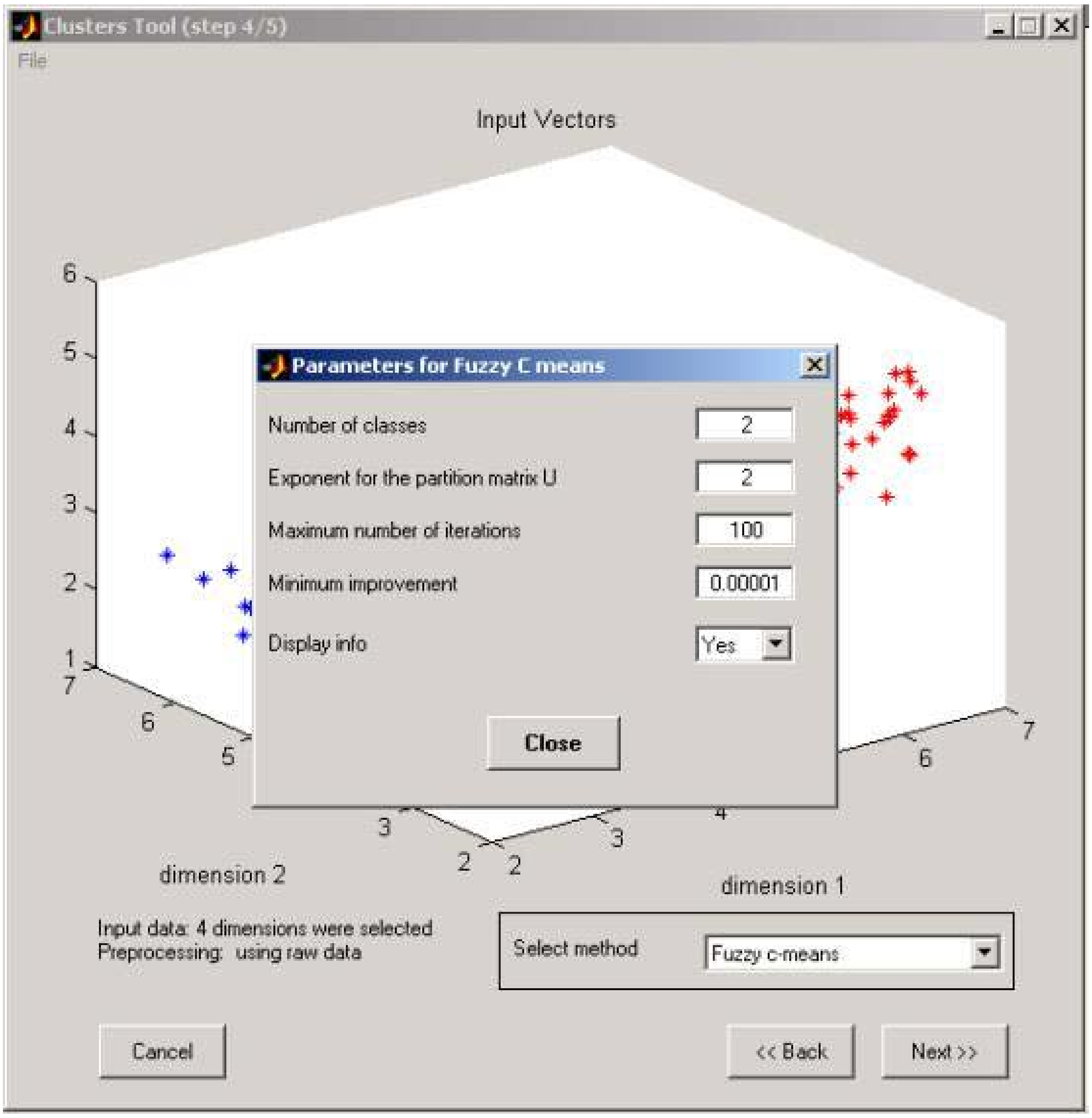
Input data: 2 dimensions were selected
Preprocessing: using raw data

Select method Kmeans

Figure 7: K-means input parameter

b. Parameters for Fuzzy C-means algorithm (Matlab fcm function):

- Exponent for the partition matrix U (default: 2.0)
- Maximum number of iterations (default: 100)
- Minimum amount of improvement (default: 1e-5)
- Info display during iteration (default: Yes).



Parameters for Fuzzy C means

Number of classes: 2

Exponent for the partition matrix U: 2

Maximum number of iterations: 100

Minimum improvement: 0.00001

Display info: Yes

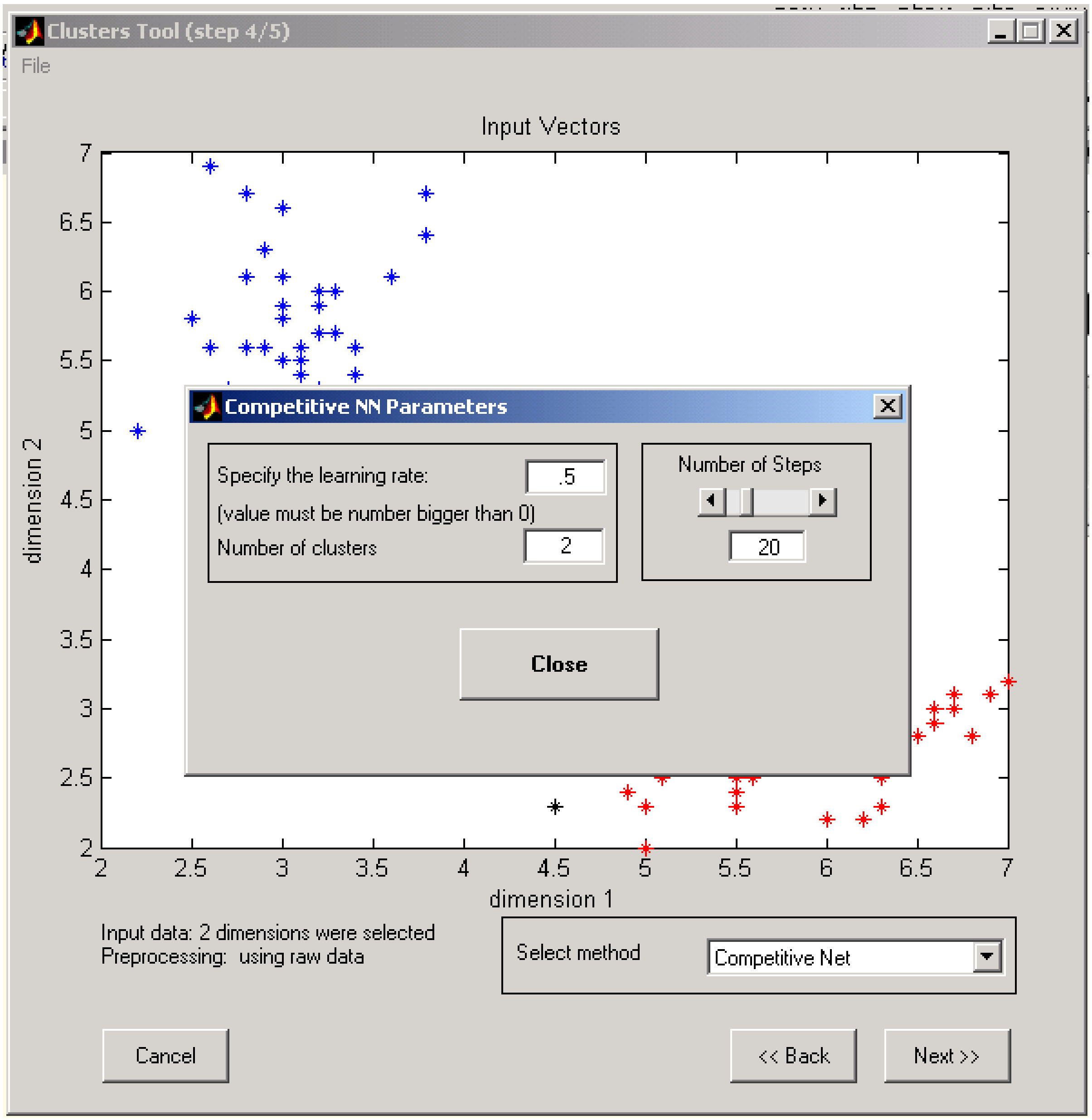
Input data: 4 dimensions were selected
Preprocessing: using raw data

Select method: Fuzzy c-means

Figure 8: Fuzzy C-means input parameters

c. Parameters for Competitive neural network algorithm:

- Learning rate (0 - 1)
- Number of clusters
- Number of learning iterations.



Specify the learning rate: .5
(value must be number bigger than 0)
Number of clusters 2
Number of Steps 20

Input data: 2 dimensions were selected
Preprocessing: using raw data

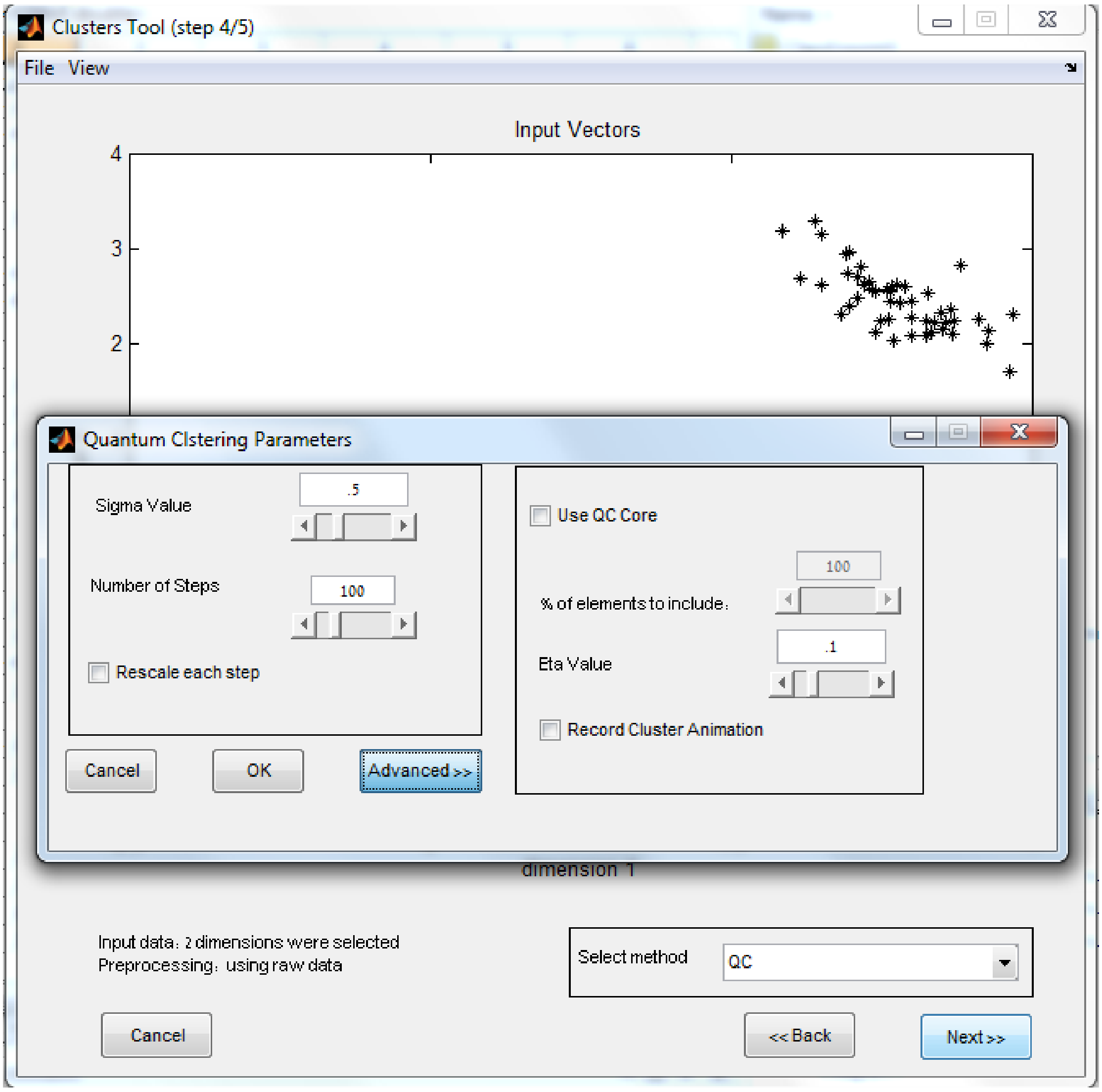
Select method Competitive Net

Figure 9: Competitive Network Input parameters

d. Parameters for Quantum Clustering algorithm*

- Sigma value
- Number of steps
- Rescaling the vectors after each step
- Using QC Core
- % of elements to include – available only with QC core option
- Eta Value – responsible or step size
- Recording the clustering animation
- Advanced button – opens the right pane

* For more details see The Method of Quantum Clustering.



Sigma Value .5

Number of Steps 100

□ Rescale each step

□ Use QC Core

% of elements to include: 100

Eta Value .1

□ Record Cluster Animation

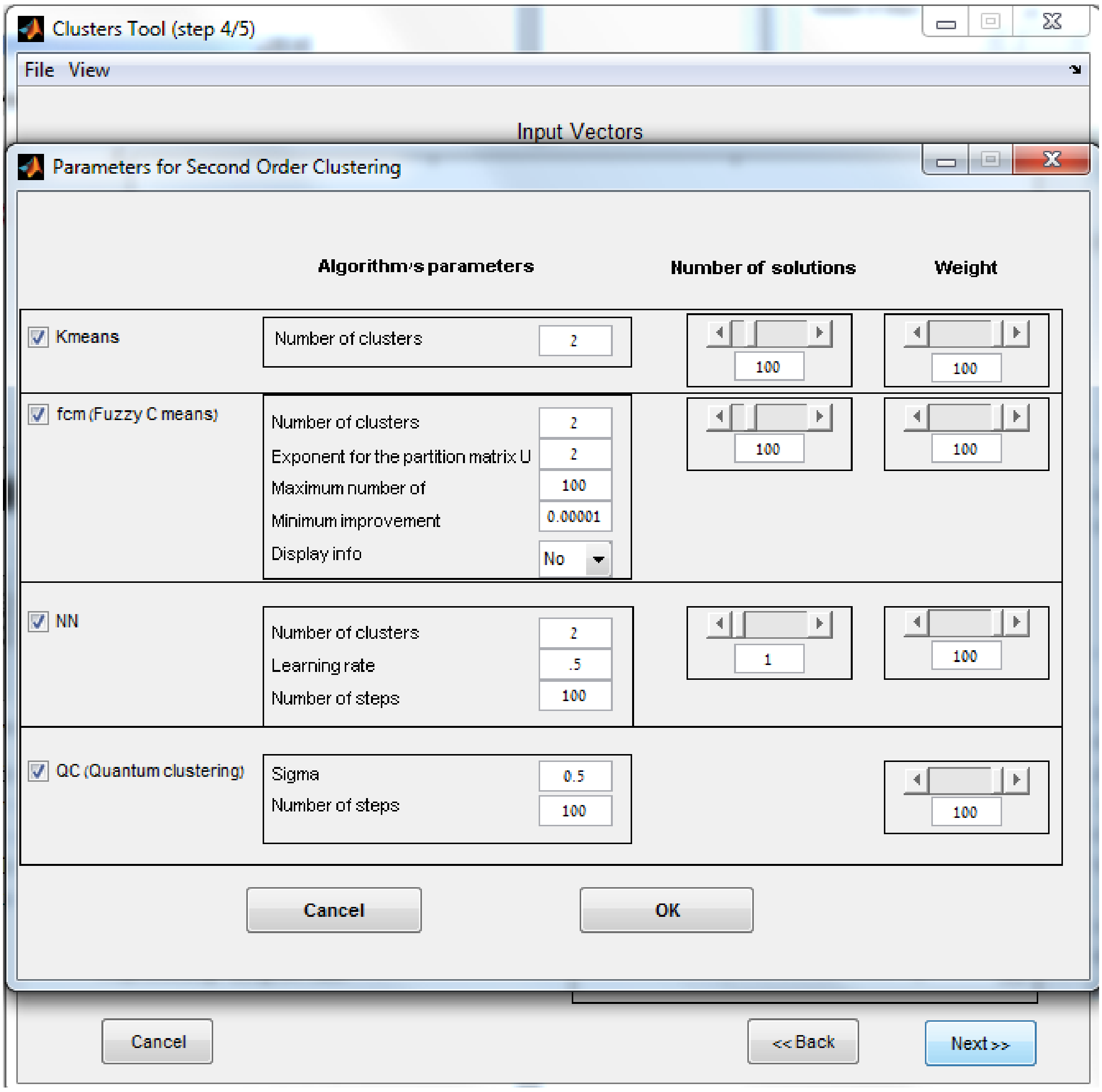
Input data: 2 dimensions were selected
Preprocessing: using raw data

Select method QC

Figure 10: Quantum clustering Input parameters

e. Parameters for second order clustering – Combining the methods together

- Choose the method to combine
- The input parameters as described in section a-d.



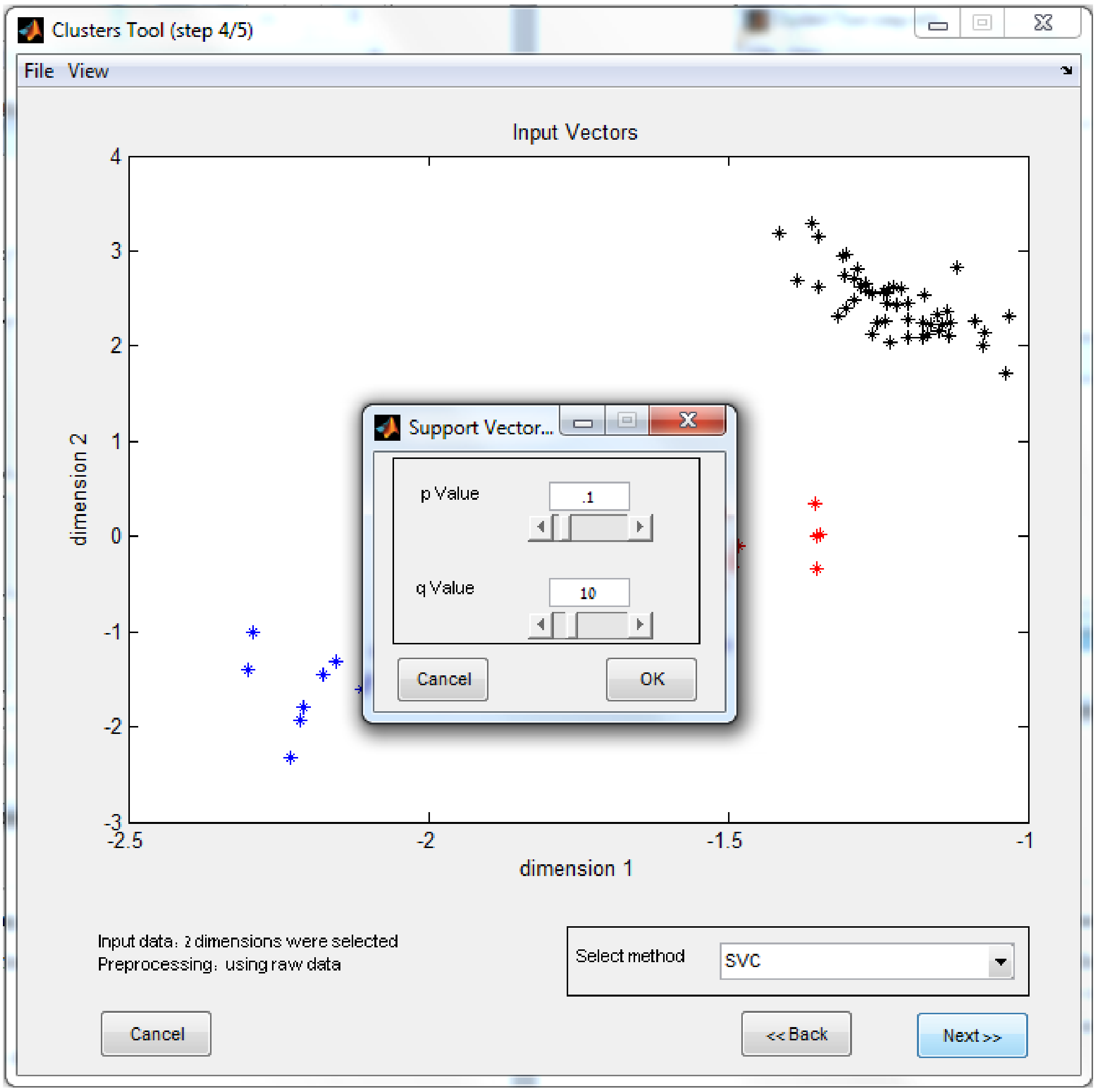
	Algorithm's parameters	Number of solutions	Weight
☑ Kmeans	Number of clusters 2	100	100
☑ fcm (Fuzzy C means)	Number of clusters 2 Exponent for the partition matrix U 2 Maximum number of 100 Minimum improvement 0.00001 Display info No	100	100
☑ NN	Number of clusters 2 Learning rate .5 Number of steps 100	1	100
☑ QC (Quantum clustering)	Sigma 0.5 Number of steps 100		100

Figure 11: Second order clustering Input parameters

f. Parameters for Support Vector Clustering algorithm*

- P – percentage of outliers
- q value

* For more details see Support Vector Clustering



p Value .1

q Value 10

Input data: 2 dimensions were selected
Preprocessing: using raw data

Select method SVC

Figure 12: Support Vector Clustering Input parameters

6. *COMPACT* results

Once the ***COMPACT*** finishes its run, the results are displayed in both graphical and textual formats.

a. Real classification plot (upper left graph): same as in step 4

b. Algorithms classification (upper right graph): points are plotted by the same rule as in the Real Classification graph, but are tagged by the algorithm (k-means, Fuzzy C means, Competitive net, QC or SVC).

c. Classification alternative display (lower right graph): Assuming that the points are ordered by the real classification (i.e., the Real Classification array looks like: 1 1...2 2...3 3...), each change in the background (white-gray-white...) represents a new cluster in the Real classification. Each one of the points is represented as a bin whose location on the x-axis equals its location in the Real classification and in the Data matrices, and its location on the y-axis represents the tag proposed by the algorithm. Perfect classification is therefore represented as homogenous rectangles.

d. Purity, Efficiency and Jaccard scores (lower left graph): These criteria for clustering assessment and are defined as follows:

$$Efficiency = \frac{n_{11}}{n_{11} + n_{10}},\ Purity = \frac{n_{11}}{n_{11} + n_{01}},\ Jaccard = \frac{n_{11}}{n_{11} + n_{01} + n_{10}}$$

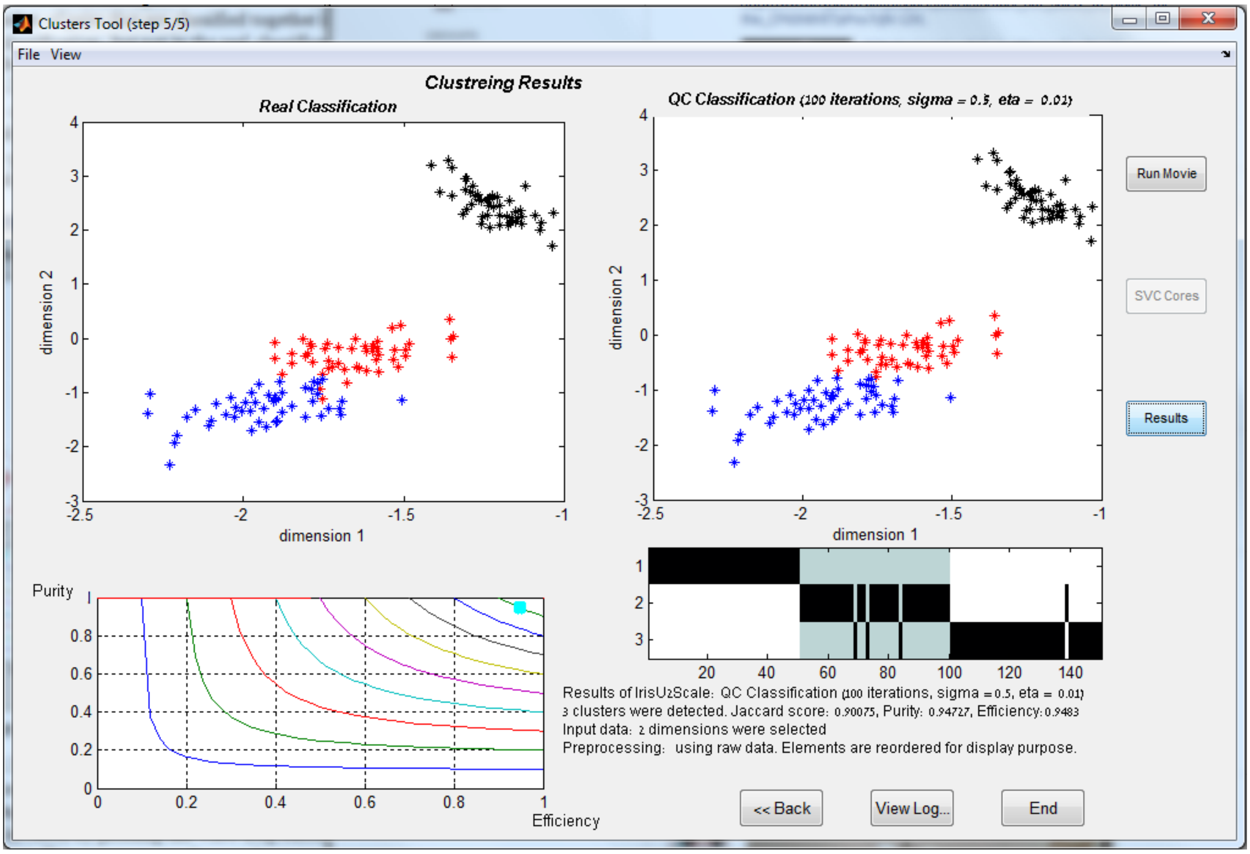
where:

- n_{11} is the number of pairs that are classified together, both in the real classification and in the algorithm's classification.
- n_{10} is the number of pairs that are classified together in the real classification, but not in the algorithm's classification.
- n_{01} is the number of pairs that are classified together in the algorithm's classification, but not in the real classification.

e. Textual summary of the clustering results (lower right textual area).

f. Starting from version 2.2, 2 new capabilities were included

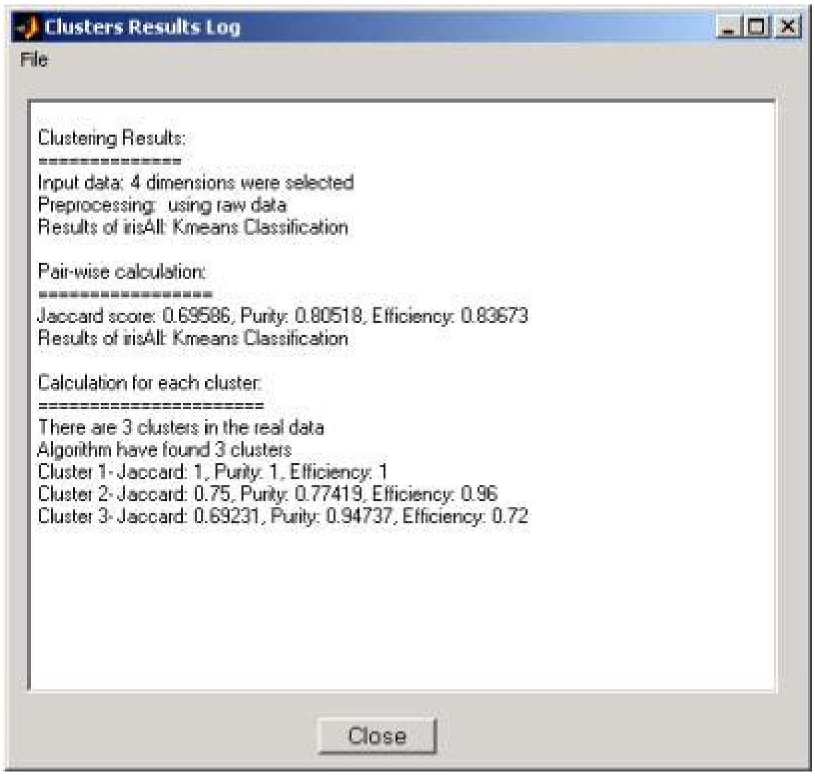
- Run Movie - show an animation of the clustering done by QC (available only in the QC method)
- SVC cores – plot only the cores found by the SVC in colors (available only in the SVC metod)



Results of IrisU2Scale: QC Classification (100 iterations, sigma = 0.5, eta = 0.01)
3 clusters were detected. Jaccard score: 0.90075, Purity: 0.94727, Efficiency: 0.9483
Input data: 2 dimensions were selected
Preprocessing: using raw data. Elements are reordered for display purpose.

Figure 13: *COMPACT* results and clustering scores

f. Starting from version 1.3, the results can be displayed also in a log window: the Clusters Results Log dialog is a window that displays the summary of the clustering results and quality in a textual way (see Figure 12). The log window is invoked by pressing the View Log button from the ***COMPACT*** result step. This summary can be saved as a text (.txt) or matlab (.mat) files (menu: File/Save/)



```
Clustering Results:
===============
Input data: 4 dimensions were selected
Preprocessing:  using raw data
Results of irisAll: Kmeans Classification

Pair-wise calculation:
==================
Jaccard score: 0.69586, Purity: 0.80518, Efficiency: 0.83673
Results of irisAll: Kmeans Classification

Calculation for each cluster:
=======================
There are 3 clusters in the real data
Algorithm have found 3 clusters
Cluster 1- Jaccard: 1, Purity: 1, Efficiency: 1
Cluster 2- Jaccard: 0.75, Purity: 0.77419, Efficiency: 0.96
Cluster 3- Jaccard: 0.69231, Purity: 0.94737, Efficiency: 0.72
```

Figure 12: *COMPACT* textual results

7. *COMPACT* output

Pressing the 'End' button at this stage will terminate the application. A new variable is then added to the Matlab workspace: calcMapping.
The calcMapping variable is a one-dimensional vector that represents the calculated classification of the elements (i.e., the proposed class of the *i-th* element appears in the *i-th* place in the vector). Actually it has similar properties to the RealClassification input parameter.

8. Optional extensions

COMPACT is a set of self explanatory and documented Matlab functions and as such can be extended. Users that are familiar with Matlab and wish to change features in the current tool are welcome to do so. In addition, we designed this tool that adding a new clustering method is done by changing only one function (see ***COMPACT*** extension).

9. Requirements

- **Project name:** COMPACT - Comparative Package for Clustering Assessment
- **Project home page:** http://horn.tau.ac.il/compact,
- **Operating system(s)**: Platform independent tested on MS-Windows (2000, XP,7), Linux and Unix
- **Programming language:** Matlab
- **Other requirements:** Matlab 7 or higher (toolboxes: fuzzy logic toolbox, version 2.1, Neural Network Toolbox 4.0.1, statistics toolbox, version 4.1).
- **Compact 2.2 requirements –** Matlab 2009b or higher, 64-bit operating system (for SVC and QC animation)

- **License:** Matlab
- **Any restrictions to use by non-academics**: currently open for all academic users. Adequate referencing required. Non-academic users are required to apply for permission to use this product.

} **Reference:** Roy Varshavsky, Michal Linial, David Horn, COMPACT: A Comparative Package for Clustering Assessment, Lecture Notes in Computer Science, Volume 3759, Oct 2005, Pages 159 - 167

Example

iris.mat is a workspace that includes two variables: (a) irisData: a 4X150 double array and (b) irisRealClassification: a 1X150 real mapping array.