ASSO

User Manual
for SODAS 2 Software

Project number:	**IST-2000-25161**
Project acronym:	**ASSO**
Project full title:	**Analysis System of Symbolic Official data**

Deliverable number:	**D3.4 b**
Title:	**User Manual** for SODAS 2 Software
Workpackage contributing to the deliverable:	**WP3: Workbench**
Lead participant (short name):	**FUNDP**
Deliverable type:	**Software-Report**
Security:	**Public**
Author:	**ASSO Developers, Partners (Edited by FUNDP)**

Abstract:	**This document contains the user manual coming with the SODAS 2 software of the ASSO project [IST-200-25161].**
Keywords:	**user, documentation, user manual, help file**

Reference:	**ASSO/WP3/D3.4 b**
Version:	**1.0**
Date:	**28/04/2004**
Total number of pages:	**-**
Status:	**Official**
Type of the document:	**Scientific**
Authorized by:	**M. Noirhomme-Fraiture (FUNDP)**

ASSO Copyright © 2004

Distribution List

Participant number	Participant short name	Recipients
1	FUNDP	Monique Noirhomme-Fraiture Edwin Diday Anne de Baenst-Vandenbroucke
2	DECISIA	Alain Morineau
3	TES	Driss Afza
4	FUNDPMa	Jean-Paul Rasson André Hardy
5	INRIA	Yves Lechevallier
6	DAUPHINE	Fabrice Rossi
7	RWTH	Hans Hermann Bock
8	UFPE	Francisco de Assis
9	INE	Carlos Marcelo
10	EUSTAT	Marina Ayestaran Marta Mas
11	STATFI	Seppo Laaksonen
12	FEP	Paula Brito
13	DMS	Carlo Lauro Rosanna Verde
14	DIB	Floriana Esposito Donato Malerba
15	UOA	Haralambos Papageorgiou Maria Vardaki

Amendment History

Version	Date	Actions	Observations
0.1	10 June 2003	First version	Collected by Anne de Baenst
0.2	18 December 2003	New version	First complete version
1.0	28 April 2004	Final version	

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

User Manual

for SODAS 2 Software

ASSO Developers Partners
(Edited by FUNDP)

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **28/04/2004**

Content

This document provides the user manuals for the SODAS 2 software developed in the ASSO Project.

The methods in italics are coming from the previous SODAS 1 software integrated in SODAS 2.

Data Management Methods

Import *Leader*: INRIA

1. DB2SO From DataBase to Symbolic Objects (INRIA)
2. ND2SO From Native Data to Symbolic Objects (FUNDP)

Export *Leader*: DIB

3. SO2DB From Symbolic Objects to DataBase (DIB)

Edit/New *Leader*: FUNDP

4. SOEDIT Symbolic Objects Edition (FUNDP)

Treatment Methods

Descriptive Statistics and Visualisation *Leader*: FUNDP

5. DSTAT Descriptive Statistics (DAUPHINE)
6. VIEW/VSTAR Viewer (FUNDP)

Dissimilarity and Matching *Leader*: DIB

7. DISS/VDISS Dissimilarity Measures (DIB)
8. MATCH Matching Operators (DIB)

Clustering *Leader*: INRIA

9. *DIV* *Divisive Clustering* [*from SODAS 1*] (INRIA)
10. HIPYR/VPYR Hierarchical and Pyramidal Clustering (FEP)
11. SCLUST Dynamic Clustering (INRIA)
12. DCLUST Clustering Algorithm based on Distance Tables (UFPE)
13. SYKSOM Kohonen Self-Organizing Map (RWTH)
14. CLINT Interpretation of Clusters (FEP)
15. SCLASS/VTREE Unsupervised Classification Tree (FUNDPMa)

Factorial *Leader*: DMS

16. SPCA Principal Component Analysis (DMS)
17. SFRD Factorial Discriminant Rule (DMS)
18. SGCA Generalised Canonical Analysis (DMS)
19. SDD Discrimant Description towards Interpretation (DAUPHINE)

Discrimination *Leader*: FUNDPMa

20. *TREE* *Decision Tree* [*from SODAS 1*] (INRIA)
21. *SDT* *Strata Decision Tree* [*from SODAS 1*] (UCM - Madrid)
22. SBTREE Bayesian Decision Tree (FUNDPMa)
23. SFDA Factorial Discriminant Analysis (DMS)

Regression *Leader*: DAUPHINE

24. SREG Regression (DAUPHINE)
25. SMLP Multi-Layer Perceptron (DAUPHINE)

DATA MANAGEMENT METHODS

Import

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

DB2SO User Manual

From DataBase to Symbolic Objects

G. Hébrail (EDF)
M. Csernel, A. El Golli and Y. Lechevallier
INRIA

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **27/09/2003**

DB2S0 : From DataBase to Symbolic Objects

1 General information on DB2SO

DB2SO is the part of SODAS software which enables the user to build a set of assertions from data stored in a relational database.
It is assumed that a set of individuals is stored in the database and that these individuals are distributed into some groups. Then DB2SO can build one assertion for each group of individuals. In this process, mother/daughter variables and taxonomies on variable domains can also be associated with generated assertions.
Theoretical aspects of DB2SO are developed in Chapter 5 of the SODAS Scientific Report.
DB2SO is invoked from the IMPORT Menu of the SODAS software.

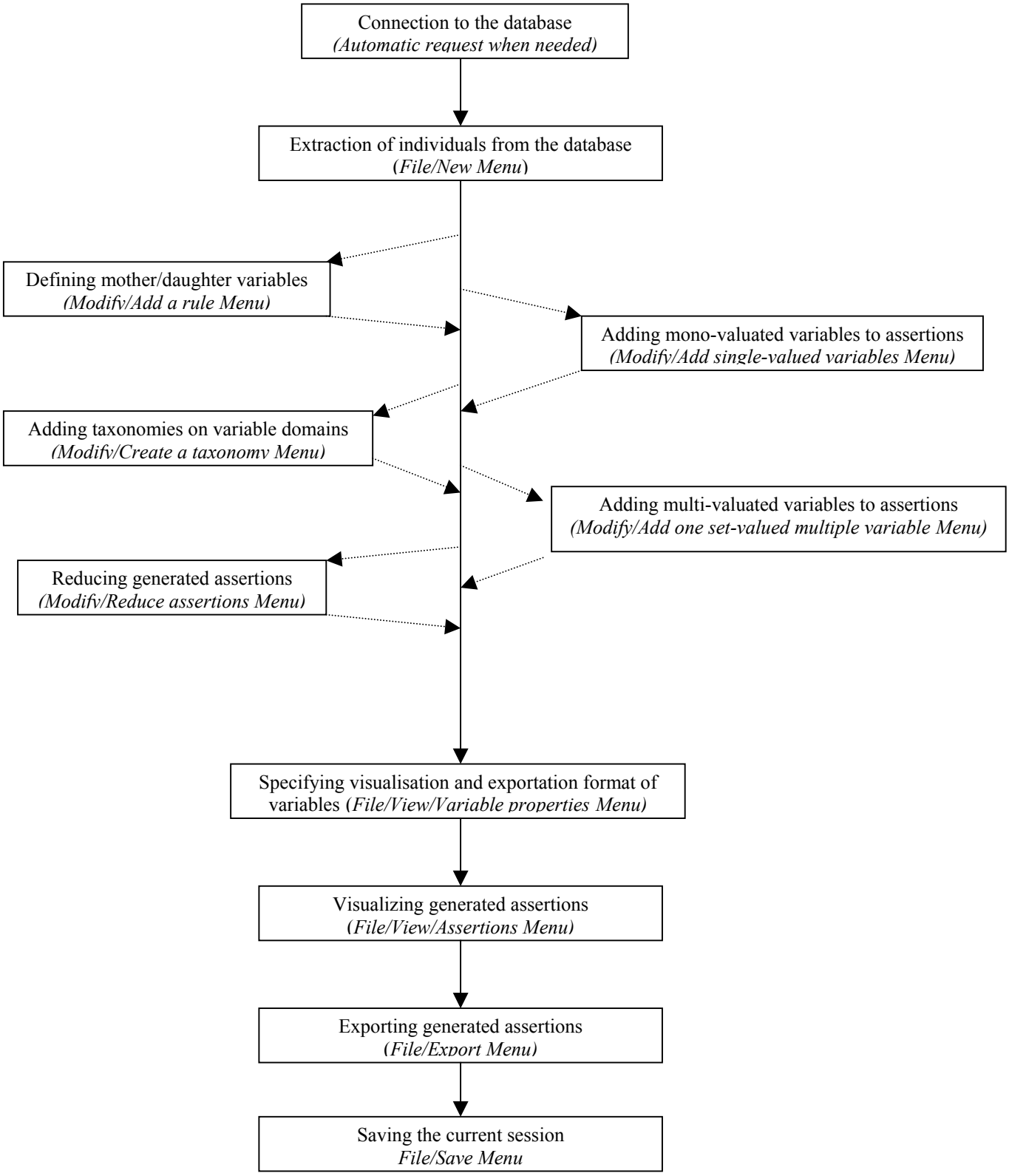
2 Typical use of DB2SO

The typical interaction between the end-user and DB2SO is the following:

+ connection to a database : see p.2
+ retrieving individuals distributed into groups by a SQL query [see the *File/New Menu* on p.5]
. optionally defining mother/daughter variables among variables describing individuals [see the *Modify/Add a dependence Menu* on p.11]
. optionally adding single-valued variables to assertions [see the *Modify/Add single-valued variables Menu* on p.9]
. optionally adding multi-valued variables to assertions [see the *Modify/Add one set-valued multiple variable Menu* on p.9]
. optionally adding taxonomies on variable domains [see the *Modify/Create a taxonomy Menu* on p.10]
. optionally simplifying generated assertions using the reduction facility [see the *Modify/Reduce assertions Menu* on p.12]
+ specifying exportation and visualisation format of variables [see the *Modify/Variable properties Menu* on p.12]
+ visualising all work that you have already done [see the View Menu on p.13]
+ exporting generated assertions to a SODAS file [see the *File/Export (and view)* Menu on p.7]
+ saving the current session to be able to restart it later [see the *File/Save (as) Menu* on p.7]

Above items marked with a "+" are compulsory steps you typically always go through, while those marked with a "." are optional.
Refer to the figure on next page for a synoptic representation of this typical interaction.

Typical interaction between the end-user and DB2SO



3 Connection to the database

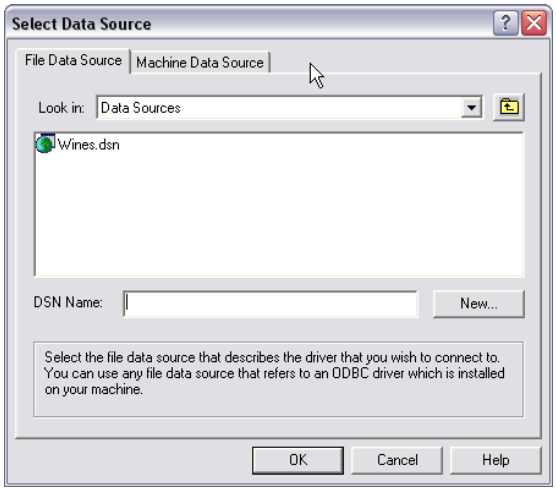
DB2SO access to relational database is done by using the Microsoft ODBC facility. Please refer to Microsoft ODBC documentation which is available on your computer under Windows 95 or Windows NT.
We assume here you have already installed on your PC the necessary ODBC drivers to access your database. Standard ODBC drivers are included with Microsoft software (like EXCEL, MSACCESS drivers). If you need to access ORACLE databases or SAS files, you must ask your retailer for the corresponding driver.
DB2SO does not provide any menu to connect to the database because the user is automatically prompted to connect when necessary, i.e. if not yet connected before running a query.
If the user wants to change the database during a DB2SO session, the procedure is the following :

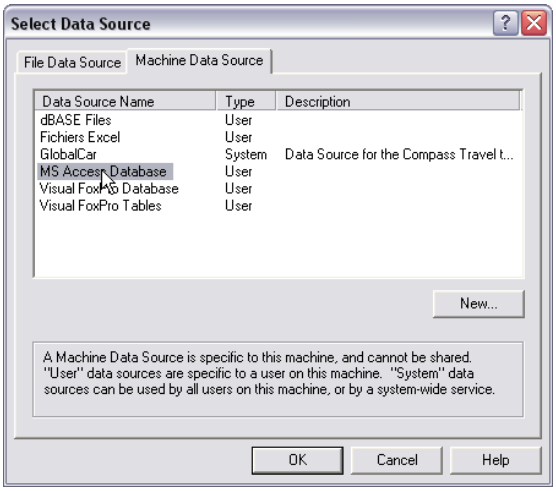
- the user disconnects from the current database by using the *File/Disconnect Menu* (see p.8),

- the user will then be prompted to specify a database before running the next query.

Another way to change the database is to click on the *Modify* button on the window in which SQL queries are written (see the *Writing a SQL query* Section on p.4).
Note that simultaneous connections to several databases are not supported.

When the user is prompted to connect to a database, the left window below appears. The user has to **click on the Machine Data Source button**, which leads to the right window below. In this window, you select the database. If you use an ACCESS database, you have to select the MS ACCESS driver which is available on your computer, and you will then be prompted to choose the file containing the database.

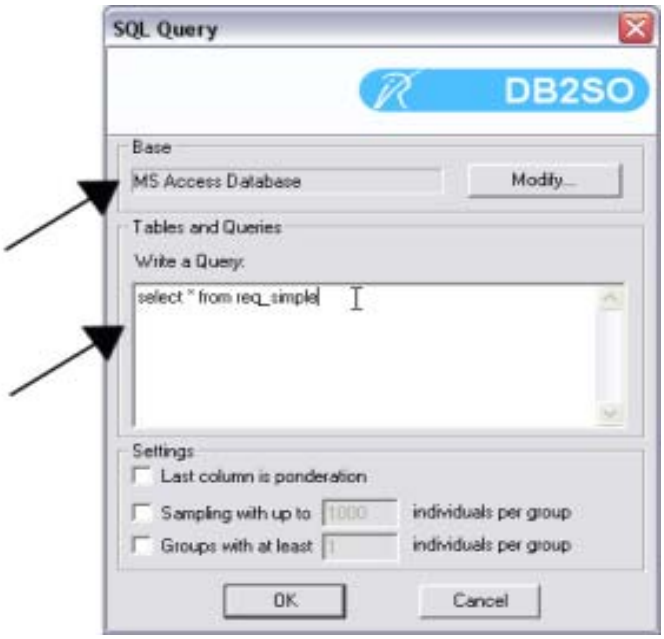




If you use another database system, you have to choose the corresponding ODBC driver.
Note : if you often use the same database file, it is possible to define an entry in the above right window which corresponds to a particular database file : this is done by clicking the New button in this window and follow instructions.

4 Writing a SQL query

In different steps of DB2SO, you will be prompted to write SQL queries. Depending on the step you are performing, a different window appears (below is shown the window for retrieving individuals from the database). But these windows always contain the two parts marked below by an arrow :



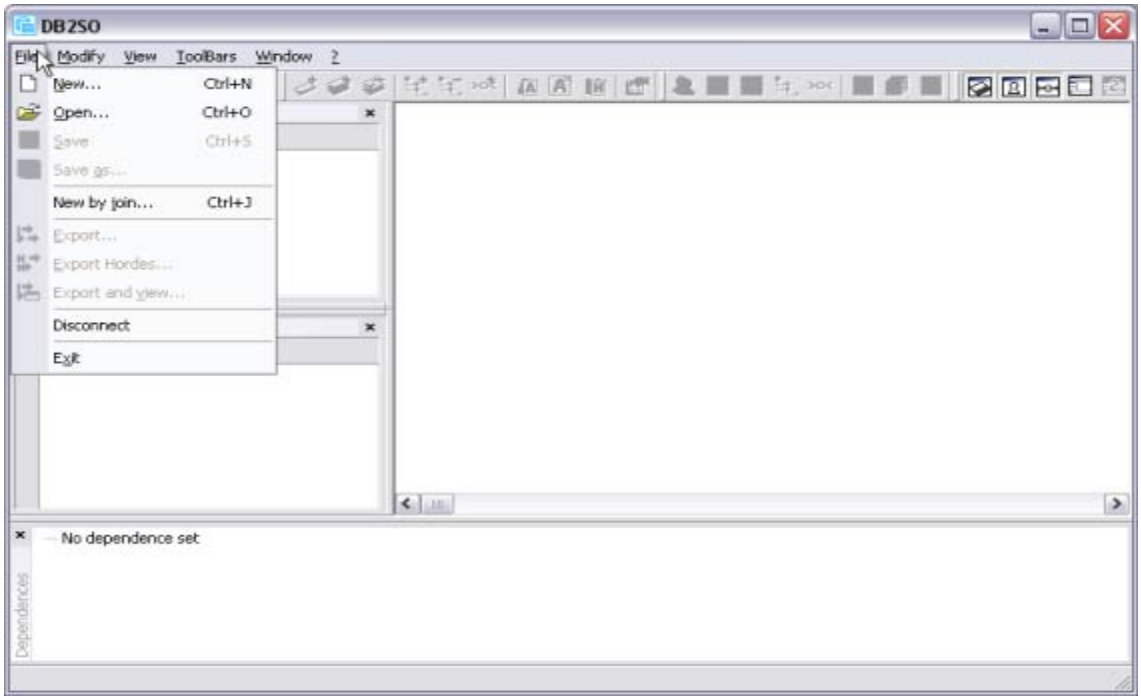
The *Modify* button enables the user to change the database to which the query is submitted. The *SQL query* space is for typing the query in the SQL language.

<u>Important note on SQL</u> :

- **you have to type standard ODBC SQL** in this window but **not** SQL of the database system. For instance, if you use MS ACCESS database, strings are delimited by ″ while they are delimited by ′ in the standard ODBC SQL.

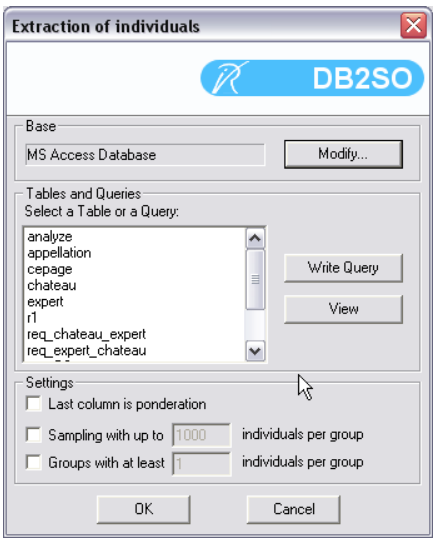
- **if you ignore everything about ODBC SQL**, the best way to run a query is to prepare it in your database system, store it under a name (for instance *Q1*) as a query or as a view, and type the following query in the above window : *select * from Q1*. This will return to DB2SO exactly what the query returns to your favourite database system.

5 File Menu



5.1 File/New Menu

Initialises a new DB2SO session : this is the command which leads you to specify the query which extracts individuals from the database. So, if you are not already connected to a database, you will be prompted by the ODBC menu to choose the database SQL queries will be submitted (see the Section on p.2).
Once connected to the database, the window above appears :



In this window you can choose a table or a query to extract individuals from.
If the last column of the query is a weight associated with each individual, select the '*Last column is ponderation'* option in the window. If you forget to select this option, the last column will lead to an interval variable in generated assertions.

If the query is supposed to return a very large number of tuples, this may overload the main memory: in this case, the user should select the *sampling* option in the window and specify the maximum number of individuals to be kept for each group (1000 by default, as shown in the above window).
If you have privacy concerns, select "Groups with at least" option, any group with less than the specified number of individuals will be ignored.

You can write your own query using "Write query".

In this window, you have to type a **SQL query of type 1**, as specified in Chapter 5 of the SODAS Scientific Report. Please refer to *Annex 1 : SQL query types* on p.19 for a summary of this Chapter 5.
The query must return one row for each individual and the expected structure of rows is (ID_IND, ID_GROUP, ATT1, ..., ATTp), where ID_IND is the individual identifier, ID_GROUP is the group identifier, and ATTi are attributes describing individuals. The number of attributes is variable but a minimum of one is required.

With either the direct choice of table/query or the user-defined query, the following operations are performed :

- individuals are retrieved from the database and stored in main memory,
- the array of assertions is generated[1].

Both the array of individuals and the array of assertions can then be viewed using the *View Menu* (see p.13).

[1] As described in Chapter 5 of the SODAS Scientific Report, numerical variables describing individuals lead to interval variables describing assertions, while non-numerical variables lead to multi-valued – possibly modal – variables.

5.2 File/New by Join Menu

This option allows to build an array of assertions by joining two arrays of assertions stored into **.gaj* files. Files named **.gaj* are files storing DB2SO sessions (see the *File/Save (as) Menu* on p.7).
The join operation takes as input two arrays of assertions. It produces a new array of assertions featuring the intersection of the two sets of assertions described by the union of the two corresponding sets of variables. Please refer to Chapter 5 of the SODAS Scientific Report for more information on the join operation on two arrays of assertions.
After specifying the names of two **.gaj* files, the join of the two arrays of assertions is performed and can be saved in another **.gaj* file, or exported to a SODAS file (**.sds*).
Note that no array of individuals is associated with an array of assertions which is the result of a join operation. Consequently, all DB2SO options related to individuals are inactive in this case. MetaData from the original projects will be ignored.
Join of two SODAS files (**.sds*) is not supported by DB2SO.

5.3 File/Save (as) Menu

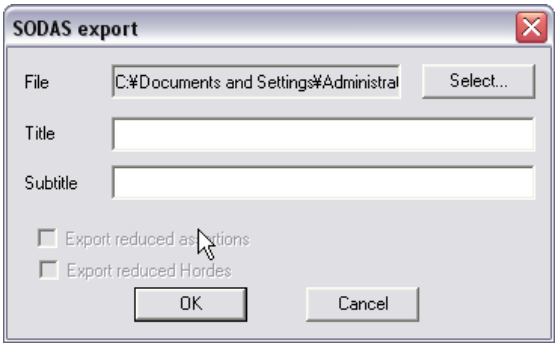
Saves the current session in a **.gaj* file. This enables the user to retrieve later the current state of a DB2SO session, by using the *File/Open Menu* (see p.7).
Just give the name of the file. The extension of these files is *.gaj* by default.
MetaData associated with the current project are saved using the same filename with an appended **.***xml*.

5.4 File/Open Menu

Recovers the session stored in the selected **.gaj* file. Just give the name of the file when prompted.
Note that **.sds* files (see the *File/Export (and view) Menu* Section on p.7) cannot be opened by this command.
MetaData associated with this project can be ignored using the "ignore MetaData" option.

5.5 File/Export (and view) Menu

Builds a SODAS or XML file containing the array of assertions of the current session.
SODAS files are files used as the input of SODAS methods. The extension of SODAS files is *.sds*.



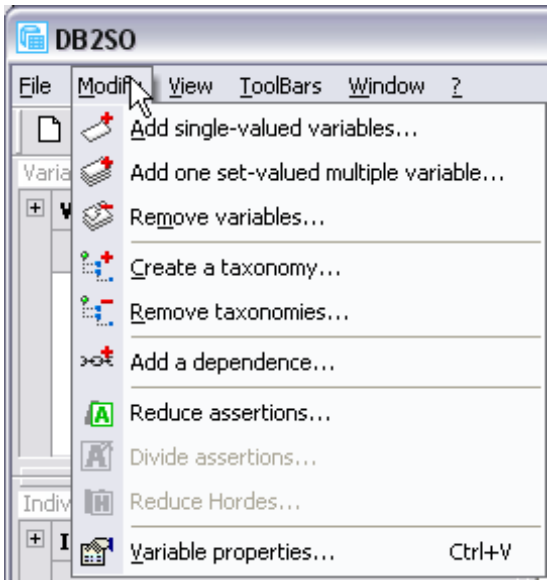
A title and a subtitle can be given as comments to the contents of the SODAS file.

If assertions/hordes have been reduced (see the *Modify/Reduce assertions Menu* on p.12), the user can either export the reduced assertions/hordes (by selecting the option in the window above) or export the non-reduced assertions/hordes.
Assertions are exported according to the properties defined in the *Modify/Variable properties Menu* (see p.12). These properties enable the user to specify which variables describing the assertions are exported and if they are exported as multi-valued (possibly modal) variables in the case of categorical variables.
The *File/Export* command just builds the **.sds* of **.xml* file while the *File/Export and view* one also shows the generated file using Wordpad (**.sds*) or Internet Explorer (**.xml*).
MetaData will be saved using the notation filename_***meta.xml***

5.6 File/Disconnect Menu

Disconnects DB2SO from the database it is currently connected. This command is useful if the user wants to extract data from another database for further operations with DB2SO.

6 Modify Menu



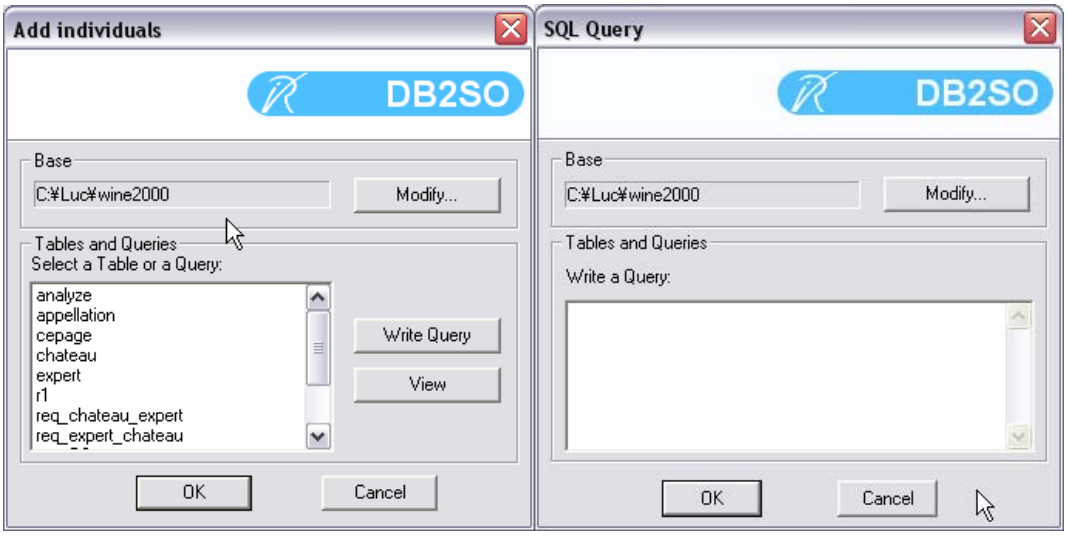
After running a *File/New*, or *File/New by join* command, an array of assertions is generated by DB2SO. In the case of *File/New*, variables describing individuals lead to multi-valued variables describing assertions.
The *Modify Menu* enables the user to modify assertions handled by the current session. Several operations can be performed:

- adding and removing single-valued or multi-valued[2] variables to assertions,
- associating taxonomies with variable domains,
- specifying mother/daughter variables by the mean of rule definition,
- reducing assertions/hordes.

[2] Also called a *set-valued multiple* variable.

- dividing assertions

6.1 Modify/Add single-valued variables Menu



Adds one or several **single-valued** variables to assertions handled by the current session. Values of these variables for the assertions of the current session are extracted from the database by a direct choice or a **SQL query of type 2** (please refer to *Annex 1 : SQL query types* on p.19 for more information about SQL query types in DB2SO).

A SQL query of type 2 has the following structure:
ID_ASSERTION, ASSERTION_ATT1, ..., ASSERTION_ATTp

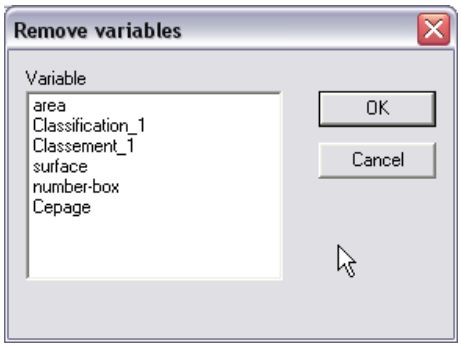
where ID_ ASSERTION is the assertion identifier and ASSERTION_ATTi are one or several variables describing assertions. Only one row should be returned by this query for each assertion ID.
If the query returns a null value for some variables of some assertions, a missing value will be associated with the corresponding assertion for these variables.
If the query returns ASSERTION ID's which do not exist in the array of the current session, information about these ASSERTION ID's is lost.
If the query does not return a row for an ASSERTION ID of the array of the current session, a missing value will be associated with all created variables of this assertion.

6.2 Modify/Add one set-valued multiple variable Menu

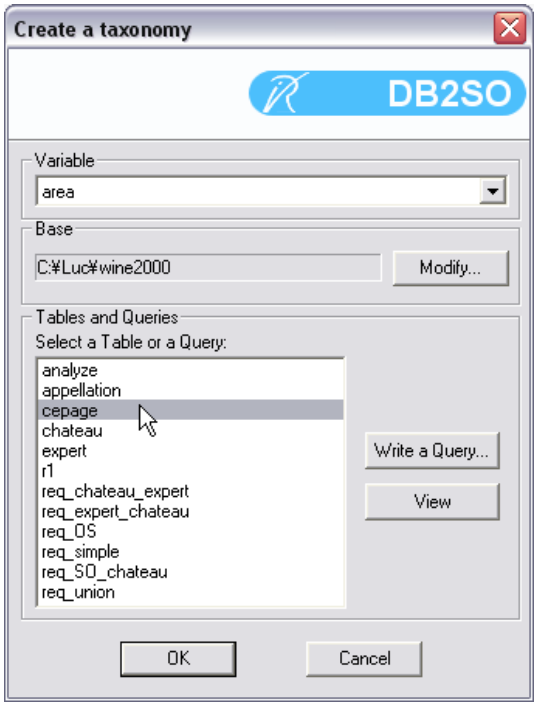
Adds **one** set-valued multiple variable to the array of assertions. Values for this new variable are extracted from the database by a **SQL query of type 3** (please refer to *Annex 1 : SQL query types* on p.19 for more information about SQL query types in DB2SO).
The user just has to do a direct choice or supply a SQL query in a window which is similar to the one for adding a single-valued variable.
In the case where the set of assertion ID's of the current session differs from those returned by the query, same rules apply as described in the preceding Section.

6.3 Modify/Remove variables Menu

Enables the user to remove variables which have been added either by the *Modify/Add single-valued variables Menu* or the *Modify/Add one set-valued multiple variable Menu*.
Just select the variable you want to remove in the window shown below :



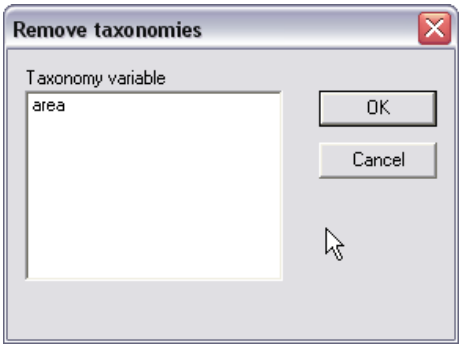
6.4 Modify/Create a taxonomy Menu



Adds a taxonomy structure to the domain of a variable describing generated assertions.
Data associated with the taxonomy is extracted from the database by a direct choice or a **SQL query of type 5a or 5b** (please refer to *Annex 1 : SQL query types* on p.19 for more information about SQL query types in DB2SO), the system recognises if it is a query of type 5a or 5b.

The user has to select in the pop-down list the variable to which the taxonomy will be added. Messages are displayed to the user if data returned by the query is invalid.

6.5 Modify/Remove taxonomies Menu



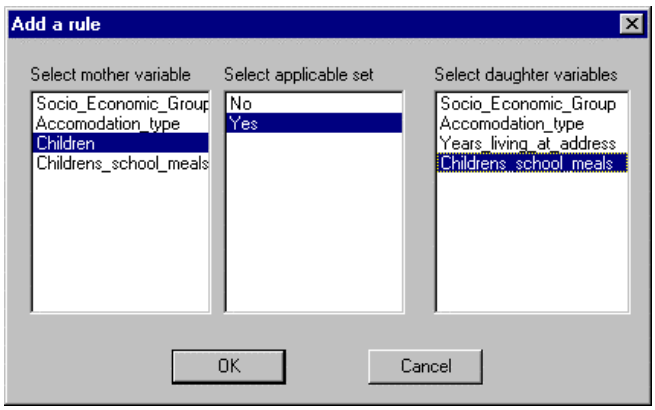
Removes the taxonomy associated with the selected variable.

6.6 Modify/Add a dependence Menu

Enables the user to define mother/daughter variables **among variables describing individuals** (it is not possible to define mother/daughter variables on variables added with the *Modify/Add single-valued variables Menu* or *Modify/Add one set-valued multiple variable Menu*).
Mother/daughter variables are defined by rules of the form:

Salary IS APPLICABLE IF *Activity* IN (*'Working', 'Retired with a pension'*)

where *Salary* and *Activity* are variable names, and *'Working'* and *'Retired with a pension'* are values of the *Children* variable.
This rules mean that there is no value of the *Salary* variable for individuals having a value of the *Activity* variable different from *'Working'* and *'Retired with a pension'*, for instance for individuals having a value *'Unemployed'*.



The specification of one rule is done through the window shown above. In this sample window, the following rule has been specified:

Childrens_school_meals IS APPLICABLE IF *Children* IN (*'Yes'*)

First select the mother variable in the left list, then all values of the mother variable appear in the central list. Select one or several values in the central list. Finally select one or several variables which will be the daughter variables. When several daughter variables are selected, there will be as many rules generated.
After clicking OK in this window, DB2SO checks if individuals respect the definition of the rules:

- individuals supposed to have a NA (NA stands for NON-APPLICABLE) value for the daughter variable are expected to have a NULL value in the database. If it not the case,

DB2SO asks the user a confirmation before forcing the value to NA for all such individuals. **This operation cannot be undone**.

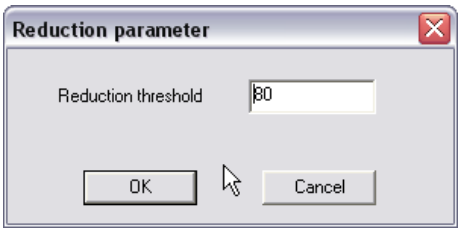
- Individuals having a NULL value in the database and supposed to have a non-NA value for the daughter variable will be considered as having a missing value in DB2SO.

Limitations:

- a variable can be the daughter variable of **only one** mother variable,
- only nominal variables can be mother variables,
- since values of individual daughter variables may be changed by this operation, **adding a rule cannot be undone**. If you are not sure of what you are doing, please save your session before adding rules.

Mother/daughter variables and their associated rules can be displayed by the *View/Dependences Menu* (see p.16).

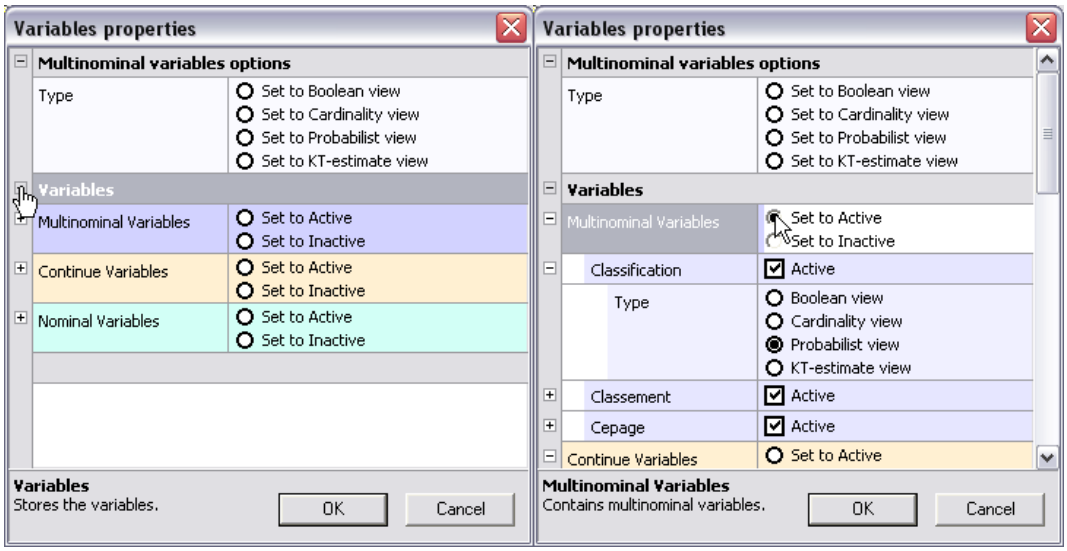
6.7 Modify/Reduce assertions Menu



Performs a reduction of assertions: DB2SO automatically simplifies generated assertions by removing untypical individuals.
Please refer to the SODAS Scientific Report for more information about this reduction process.
The user has to give a threshold which corresponds to the minimum percentage of individuals still recognized by each assertion after the reduction process.

6.8 Modify/Variable properties Menu

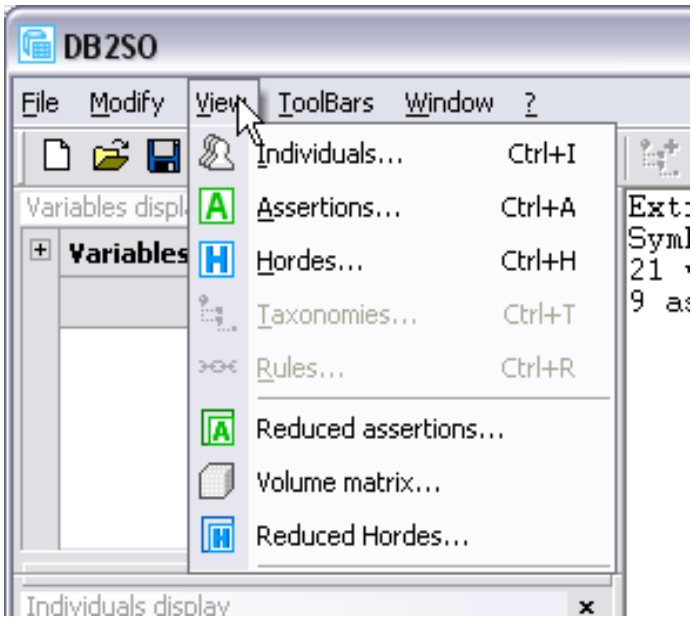
In this window, the user specifies the format of assertions for display (in the *View/Assertions Menu* and the *View/Reduced assertions Menu*) and exportation (in the *File/Export (and view) Menu*). Both variables generated from individual variables and variables directly added to assertions are accessible in this menu.
An *Inactive* variable will be invisible for both view and exportation. For **multinominal** variables, the user can choose between a boolean, cardinality, probabilist or KT-estimate probabilist representations.
Note that cardinality representation is not available if a weight has been associated with individuals (see *File/New Menu* on p.5).



Anchors on the left expand/collapse the tabs.
A **double click** on a tab does the same job but may result in **system overload** if more than two clicks are made.

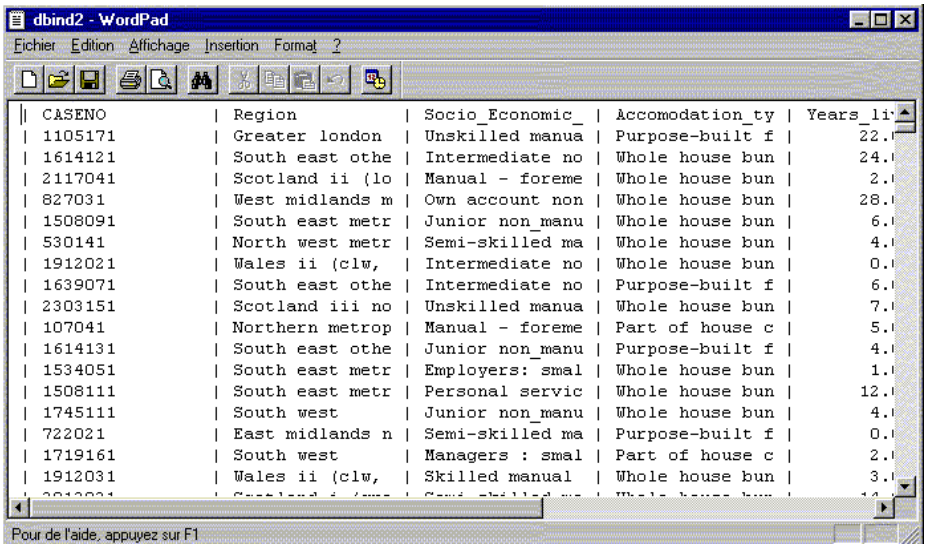
Each specific variable type (i.e. with different properties) has its own tab.
You can activate or de-activate every variable of a type by clicking on the corresponding radio in the type-tab.
You can set the default representation for multinominal variables in the same fashion (*Multinominal variables options* tab).
To activate your changes you **MUST** click on the control (for example a tab) before clicking on OK or the last operation will be discarded. You will know when the changes are applied when the background color will go back from white to its original color.
Side note: the "contagious" options may disappear, this is a windows GUI bug, but they are still "here", they will reappear with the next click on them.

7 View Menu



7.1 View/Individuals Menu

Displays in a Wordpad window the current array of individuals (see the example below). Changes made in Wordpad are ineffective in DB2SO.
Note that you can save what is displayed with Wordpad in a file you may use for other purposes.



dbind2 - WordPad

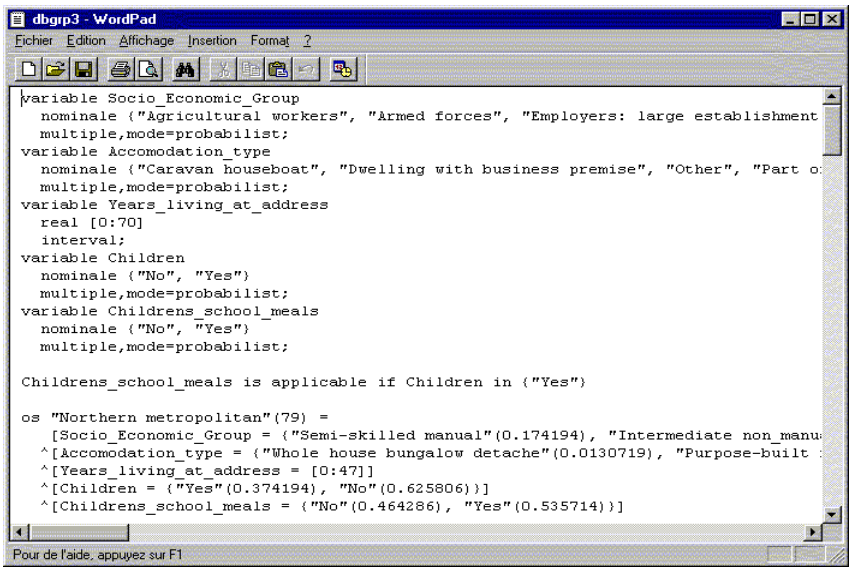
Fichier Edition Affichage Insertion Format ?

CASENO	Region	Socio_Economic_	Accomodation_ty	Years_li
1105171	Greater london	Unskilled manua	Purpose-built f	22.
1614121	South east othe	Intermediate no	Whole house bun	24.
2117041	Scotland ii (lo	Manual - foreme	Whole house bun	2.
827031	West midlands m	Own account non	Whole house bun	28.
1508091	South east metr	Junior non_manu	Whole house bun	6.
530141	North west metr	Semi-skilled ma	Whole house bun	4.
1912021	Wales ii (clw,	Intermediate no	Whole house bun	0.
1639071	South east othe	Intermediate no	Purpose-built f	6.
2303151	Scotland iii no	Unskilled manua	Whole house bun	7.
107041	Northern metrop	Manual - foreme	Part of house c	5.
1614131	South east othe	Junior non_manu	Purpose-built f	4.
1534051	South east metr	Employers: smal	Whole house bun	1.
1508111	South east metr	Personal servic	Whole house bun	12.
1745111	South west	Junior non_manu	Whole house bun	4.
722021	East midlands n	Semi-skilled ma	Purpose-built f	0.
1719161	South west	Managers : smal	Part of house c	2.
1912031	Wales ii (clw,	Skilled manual	Whole house bun	3.

Pour de l'aide, appuyez sur F1

7.2 View/Assertions Menu

Displays in a Wordpad window the generated assertions in the SOL language (see the example below).



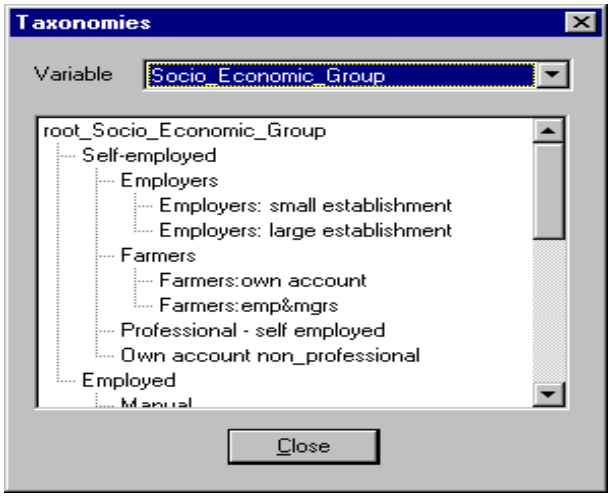
dbgrp3 - WordPad

Fichier Edition Affichage Insertion Format ?

```
variable Socio_Economic_Group
  nominale {"Agricultural workers", "Armed forces", "Employers: large establishment
  multiple,mode=probabilist;
variable Accomodation_type
  nominale {"Caravan houseboat", "Dwelling with business premise", "Other", "Part o
  multiple,mode=probabilist;
variable Years_living_at_address
  real [0:70]
  interval;
variable Children
  nominale {"No", "Yes"}
  multiple,mode=probabilist;
variable Childrens_school_meals
  nominale {"No", "Yes"}
  multiple,mode=probabilist;

Childrens_school_meals is applicable if Children in {"Yes"}

os "Northern metropolitan"(79) =
   [Socio_Economic_Group = {"Semi-skilled manual"(0.174194), "Intermediate non_manu
  ^[Accomodation_type = {"Whole house bungalow detache"(0.0130719), "Purpose-built
  ^[Years_living_at_address = [0:47]]
  ^[Children = {"Yes"(0.374194), "No"(0.625806)}]
  ^[Childrens_school_meals = {"No"(0.464286), "Yes"(0.535714)}]
```

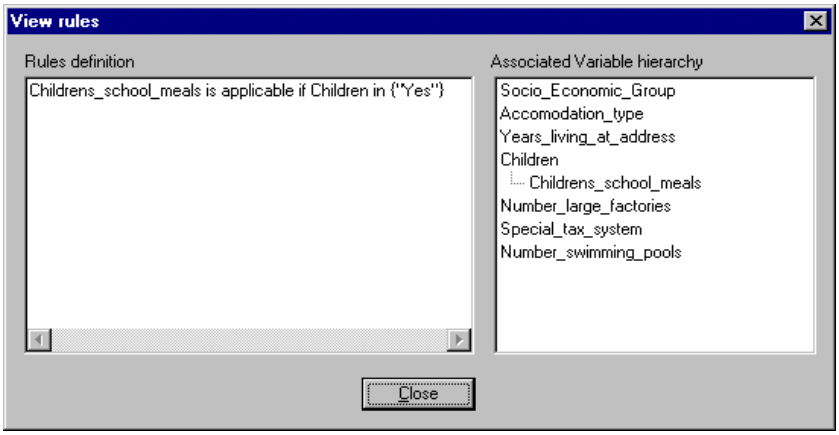
Pour de l'aide, appuyez sur F1

7.3 View/Taxonomies Menu



Selecting this option in the *View Menu* opens the window shown above. Select in the pop-down list the variable (here *Socio_Economic_Group*) for which the associated taxonomy has to be displayed. Then the taxonomy is displayed in the scrollbar window as shown above.

7.4 View/Dependences Menu

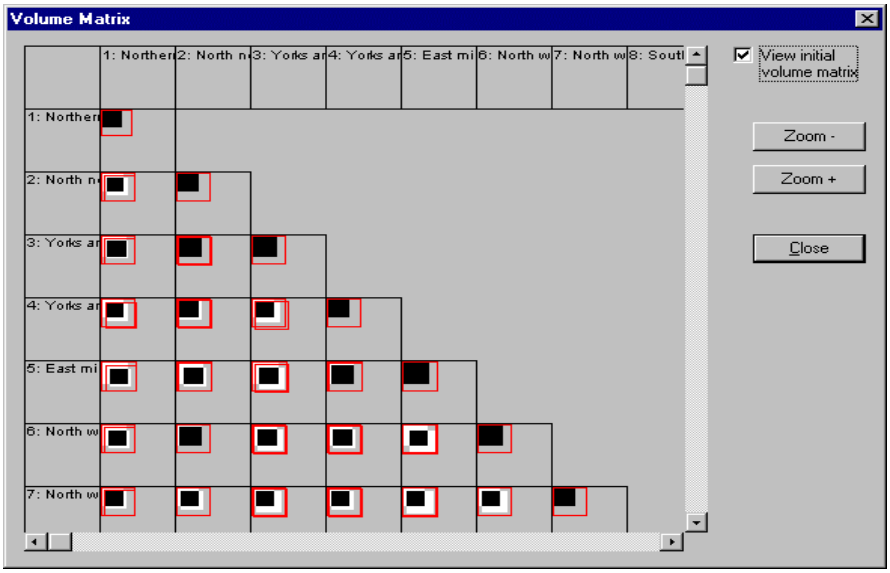


Selecting this option in the *View Menu* opens the window shown above. In the left part, rules representing mother/daughter variables are displayed. In the right part, the hierarchy induced by the definition of mother/daughter variable rules is displayed.

7.5 View/Reduced assertions Menu

Same as the *View/Assertions Menu* except that reduced assertions are displayed, if a reduction process (see *Modify/Reduce assertions Menu* on p.12) has been performed before.

7.6 View/Volume Matrx Menu



Displays the volume matrix associated with the reduction process of assertions. For a definition of the volume of an assertion, please refer to Section 5.4.3 of Chapter 5 of the SODAS Scientific Report.
The matrix is a square matrix in which each cell corresponds to a couple of assertions. Each cell shows :

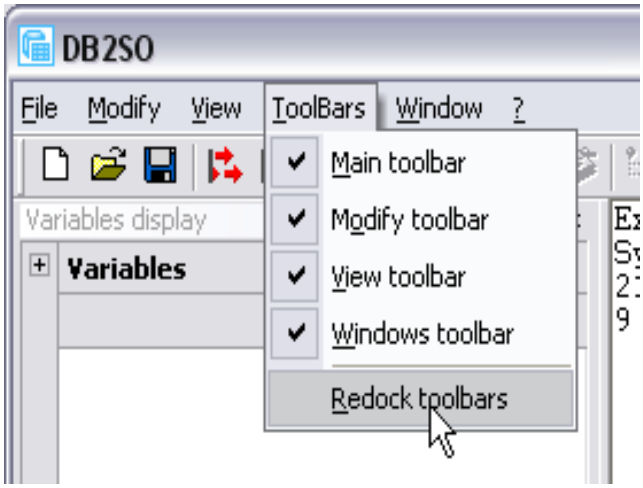
- a white square with a surface proportional to the volume of the corresponding row assertion,

- another white square with a surface proportional to the volume of the corresponding column assertion,

- a black square representing the volume of the intersection of the row and column assertions. (note that the two white squares are laid out so that their intersection is the black square).

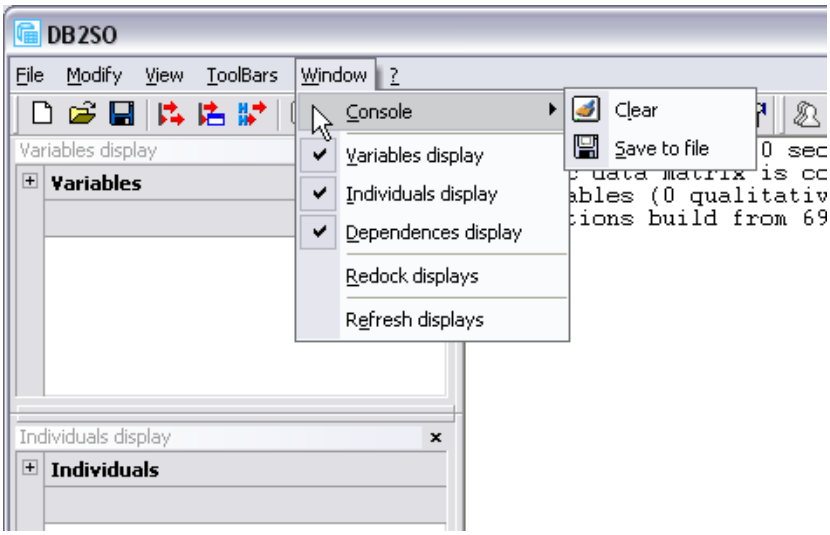
When selecting the *View initial volume matrix*, red line squares show the state of this volume matrix without running the reduction process. This visualisation gives an idea of the effect of assertion reduction (the reduction process is supposed to reduce intersection - overlap - between assertions).
A zoom is available in this window.

8 ToolBars Menu



This menu allows you to display or hide the toolbars associated with the DB2SO menus. *Redock toolbars* will replace all the visible toolbars to a default docked position (usefull if the toolbars are messed up by a window resizing or by an improper un-docking).

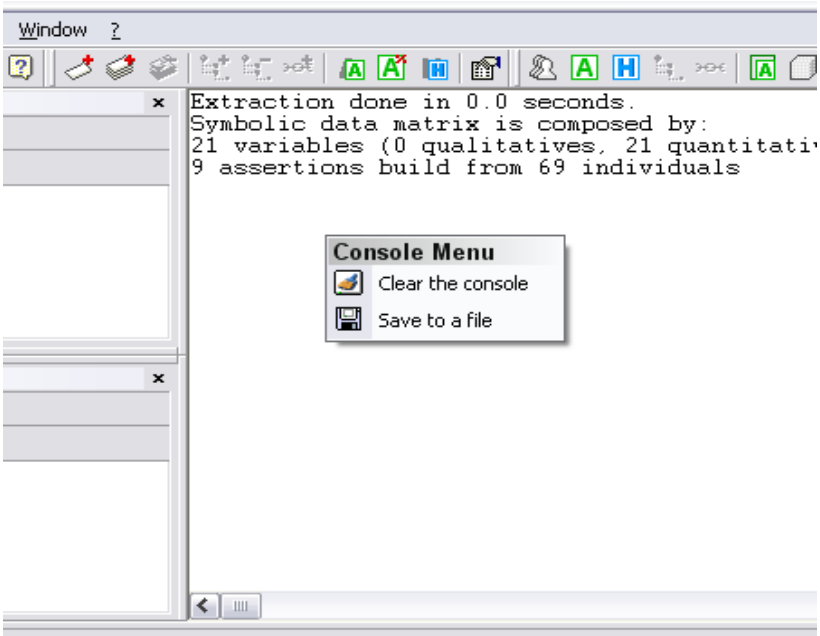
9 Window Menu



This menu allows you to show/hide the information windows (called *displays*), redock them, refresh them, clear the console and save the console content to a text file.

10 The Console

The console is where all texts are displayed. Some of its information may be important to you, that s why we included the Save to File feature. A right-click on the console will popup the console menu.



Annex 1 : SQL query types

SQL query of type 1:

This query type is used to retrieve the following information from the database:

- *individuals ID* and *attributes*, possibly associated with a weight for each individual. These attributes lead to multi-valued variables describing generated assertions.

- *groups ID* which become the names of generated assertions.

- membership of each individual to one or several groups.

To summarize, the structure of this query type is the following:

IND_ID, GROUP_ID, IND_V1, IND_V2, ..., IND_Vk [, IND_WEIGHT]

This query must contain at least three columns (individual ID, group ID and one variable describing individuals). The weight column is optional and the number of individual attributes is variable.

Example:

CASENO	Region	Bedroom	Dining\living	Socio - Economic Group
114051	Northern metropolitan	2	1	Managers : large establishment
114061	Northern metropolitan	2	1	Own account non_professional
114071	Northern metropolitan	2	1	Intermediate non_manual ancill
114101	Northern metropolitan	3	2	Semi-skilled manual
114111	Northern metropolitan	2	1	Managers : large establishment
114131	Northern metropolitan	1	1	Intermediate non_manual ancill
114141	Northern metropolitan	3	1	Unskilled manual
114161	Northern metropolitan	3	1	Managers : large establishment
114171	Northern metropolitan	3	1	Skilled manual

201081	North non-metropolitan	2	1	Semi-skilled manual
201091	North non-metropolitan	4	3	Skilled manual
201101	North non-metropolitan	3	1	Personal service
201111	North non-metropolitan	2	1	Skilled manual
201131	North non-metropolitan	2	1	Junior non_manual
201141	North non-metropolitan	3	2	Unskilled manual
201171	North non-metropolitan	3	1	Intermediate non_manual ancill
...	...			

This query returns one row per household. Individuals are households identified by the first column: CASENO. The group ID is defined by the second column, here REGION. There will be one assertion generated for each region. Households are described by two numerical and one qualitative attribute: the number of two types of rooms in the house and the socio-economic group of the household. Both numerical and qualitative attributes can describe individuals. The only difference is in the generalisation process where numerical variables lead to intervals of values while qualitative ones lead to lists of values or probability distributions. In this query, there is no weight column. If a weight column is present, this must be the last column and the user has to tell it to the system.
The weight column is used in the generalisation process for the computation of probability values associated with modal multi-valued variables. When no weight column is present, all individuals are assumed to have the same weight, equal to 1.

SQL query of type 2:
This query type is used to retrieve the following information from the database:

- *single-valued attributes* describing *groups*: these attributes lead to single-valued variables describing generated assertions.

The structure of this query type is the following:

GROUP_ID, GROUP_V1, GROUP_V2, ..., GROUPE_Vp

This query must contain at least two columns: the group ID and one group attribute. The number of group attributes is variable.

Example:

Region	Number of cars	%Unemployment
East anglia	273000	8
East midlands non-metropolitan	491000	9
Greater london north east	197000	10
Greater london north west	154000	7
Greater london south east	149000	8
Greater london south west	160000	6
North non-metropolitan	260000	10
North west metropolitan	458000	11
...	...	...

This query returns two single-valued attributes describing regions. There is one row per region, thus one value per region for every attribute. Here the two attributes are numerical but they could have been qualitative. Both qualitative and numerical attributes can be returned in the same query.

SQL query of type 3:
This query type is used to retrieve the following information from the database:
- exactly one *native multi-valued (qualitative) attribute* describing *groups*, which lead to a multi-valued variable describing generated assertions

The structure of this query type is the following:
GROUP_ID, MULT_VAL_ATT, CARDINALITY
Example:

Region	Accomodation type	Number of cases
East anglia	Caravan houseboat	1
East anglia	Part of house converted flat o	16
East anglia	Purpose-built flat or maisonet	22
East anglia	Whole house bungalow detache	85
East anglia	Whole house bungalow semi-de	83
East anglia	Whole house bungalow terrace	66
East midlands non-metropolitan	Caravan houseboat	3
East midlands non-metropolitan	Dwelling with business premise	1
East midlands non-metropolitan	Part of house converted flat o	17
East midlands non-metropolitan	Purpose-built flat or maisonet	42
East midlands non-metropolitan	Whole house bungalow detache	134
East midlands non-metropolitan	Whole house bungalow semi-de	182
East midlands non-metropolitan	Whole house bungalow terrace	112
Greater london north east	Dwelling with business premise	1
Greater london north east	Part of house converted flat o	17
...	...	...

The first column of the query describes the group ID, the second one contains modalities of the multi-valued attribute, and the third one is the cardinality of the modality within the group. This query type is useful to acquire multi-valued variables describing assertions when they are not computed from an underlying population. It is not adviced to use this feature if the attribute describes an underlying population: in this case, it is preferable to use a SQL query of type 1 retrieving the underlying population, run the reduction process, and finally join generated assertions to the current ones.

SQL query of type 4: *(not supported by version V2 of DB2SO)*
This query type is used to retrieve the following information from the database:
- exactly one *native interval (numerical) attribute* describing *groups*, which lead to an interval variable describing generated assertions

The structure of this query type is the following:
GROUP_ID, MIN_VALUE, MAX_VALUE
where MIN_VALUE and MAX_VALUE are the bounds of the interval of the new variable describing groups identified by GROUP_ID.

Example:

REGION	LOWER	UPPER
East anglia	1531,59	3058,34
East midlands non-metropolitan	-84,31	259,95
Greater london north east	3525,06	6878,74
Greater london north west	27132,72	32852,73
Greater london south east	33847,21	41307,62
Greater london south west	-52,99	326,95
North non-metropolitan	-65,69	405,41
North west metropolitan	1833,84	3336,49
...	...	...

In this example, this query returns, for each region, a confidence interval for the average income of a particular class of people. This will create an interval numerical variable describing assertions. As for queries of type 3, it is not adviced to use this feature when this data can be calculated from an underlying population.

SQL query of type 5:

This query type is used to retrieve the following information from the database:

- a *taxonomy* on values of a particular variable

This information can be retrieved from the database by two means, corresponding to two different ways of representing a taxonomy in a table:

SQL query of type 5.a ATT_VALUE, LEV1_VALUE, LEV2_VALUE, ..., LEVp_VALUE

This is the representation suited when all leaves of the taxonomy are at the same depth in the tree. A typical example is a query returning TOWN, DEPARTMENT, REGION, COUNTRY to define a taxonomy on a TOWN variable.

SQL query of type 5.b CHILD_ATT_VALUE, PARENT_ATT_VALUE

This is a child/parent representation of the taxonomy which links together the different values of an attribute domain. An example of this representation is given below.

Example of the child/parent representation

Socio - Economic Group	Parent Group
Agricultural workers	Employed
Armed forces	Employed
Employers	Self-employed
Employers: large establishment	Employers
Employers: small establishment	Employers
Farmers	Self-employed
Farmers:emp&mgrs	Farmers
Farmers:own account	Farmers
Intermediate non_manual ancill	Non-manual
Intermediate non_manual foreme	Non-manual
Junior non_manual	Non-manual
Managers	Employed
Managers : large establishment	Managers
Managers : small establishment	Managers
Manual	Employed
Manual - foremen\supervisors	Managers
Non-manual	Employed
Own account non_professional	Self-employed
Personal service	Employed
Professional - employee	Employed
Professional - self employed	Self-employed
Semi-skilled manual	Manual
Skilled manual	Manual
Unskilled manual	Manual

which leads to the following taxonomy:

Employed Agricultural workers Armed forces Personal service Professional - employee Manual Semi-skilled manual Skilled manual Unskilled manual Non-manual Intermediate non_manual ancill Intermediate non_manual foreme Junior non_manual Managers Managers : large establishment Managers : small establishment Manual - foremen\supervisors	Self-employed Professional - self employed Own account non_professional Employers Employers: large establishment Employers: small establishment Farmers Farmers:own account Farmers:emp&mgrs

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

ND2SO User Manual

From Native Data to Symbolic Objects

Edited by FUNDP

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **16/12/2003**

ND2SO : From Native Data to Symbolic Objects

ND2SO is a module designed for the transfer of aggregated data from the standard ASCII format to the SODAS format (.xml). Most of standard statistical programs allow to aggregate individual data to build new variables for these sets of individuals. These new variables can be the minimum, the maximum or the mean of an initial quantitative variable. They can also be the proportion of individuals in some category for a categorical variable or for a discrete quantitative variable. For example, if the aggregated sets contain the inhabitants of towns, a variable can represent the proportion of women in these towns. Other variables can give the proportion of individuals who are less than 24 year old, between 25 and 39, between 40 and 64 and more than 65.

The user could be interested in using these pre-aggregated data to build Symbolic Objects instead of building them directly from the database with the DB2SO module. The ND2SO module will help to perform this transformation. In this chapter, we will refer to "set" to indicate the aggregated data (i.e. as in the original file). Let us notice that ND2SO does not cover all the cases, like taxonomies and hierarchical dependency. Moreover, if the aggregation is not optimised inside the original software (suppression of border individuals in order to have denser sets), ND2SO will not be able to deliver Symbolic Objects of such good quality as those designed with DB2SO.

The help guide (in the Help menu) gives a description of all the menus of this module. It is also available online, and it contains a screen by screen description of this program.

In this user manual, we will present examples of use of this module.

In the principal screen of the workbench, select "Import" in the methods for data management, then select ND2SO. The first screen of ND2SO opens. Click on the File menu; choose Open and then the name of the original file which has to be in the ASCII format. It means that it has been recorded with one aggregated set per line. The values of the different variables are separated by a blank and the lines are separated by x. The values can be numbers (integer or real) or letters. The missing data are represented by ".". The open function displays the original data in the form of a traditional table, with one "set" by line and variables in columns (see Fig. 1).

ND2SO - [Table1]

File Translate Window Help

	K121	K1	K2	K3	K4	K500	K510	HP13035	HP13036
1151	823.5835987	7158.449008	2218.776625	7308.257487	1645.339799	469.1767877	364.671998	72.13474965	39.13019465
1152	519.4523434	5988.689014	1178.8744	6181.240799	1209.847017	94.44306517	269.2345335	173.5525917	122.4227126
1153	427.7655684	5057.305503	1253.339901	5984.104013	1091.573264	216.5047802	236.9988168	64.60285226	18.09100466
1154	337.9570064	5017.177116	1708.968865	6363.352015	1375.50704	158.3593663	216.7149981	13.98002837	54.46419232
1301	436.6887563	5140.136741	974.3017377	5981.827858	1547.664424	206.3501388	161.9382767	202.5983113	57.66456928
1302	436.4101041	5139.168962	966.3040486	6428.358681	1403.394153	209.5962029	233.2652168	48.03284106	78.2779184
1303	419.5998104	4163.558606	694.6770276	5390.062738	1081.762641	129.7074177	164.9639003	57.25525872	31.72986081
1304	378.0551476	5083.506432	1514.175633	6261.262426	1497.330152	196.9276801	229.8067087	61.87536289	39.73942669
1451	613.0017851	6231.532386	1145.775244	7365.107891	1483.925264	361.0715625	445.3107287	47.75553993	62.63809178
1452	571.8737481	6245.337056	1240.77417	7191.40626	1602.091253	166.8710151	406.5259338	91.39079159	60.43818882
1453	527.2042227	5319.103033	1793.60548	6865.257758	1126.091114	215.7846291	288.2140574	32.9033424	40.0693665
1454	529.3413949	5632.247994	1255.579323	6933.565146	1492.114723	194.6173702	330.3800079	85.52633129	22.04882981
1601	671.9681269	4998.624943	678.725547	6834.584518	1084.933845	297.9906546	367.0788248	29.31495781	26.77223378
1602	365.772127	4313.7907	894.6162592	7114.93412	1061.83607	339.3068206	556.5245548	49.84423021	19.11524709
1603	480.2969781	4241.709826	675.8273529	5766.228119	999.9699856	409.0390303	472.2312453	7.211154658	6.881362152
1604	318.910701	4148.674629	1108.786206	6623.526102	1465.146721	294.3599446	291.0622836	7.441334409	4.492135968

Pour l'aide, appuyez sur F1 NUM

Figure 1: Original Data Table

ND2SO allows the design of

- Quantitative single variables;
- Categorical single variables;
- Categorical multi-valued symbolic variables;
- Interval symbolic variables;
- Modal symbolic variables.

To build a **quantitative variable**, you have to:

- Select the appropriate column in the original table in clicking on the head of the column. The values have to be numeric (integer or decimal);
- Select Translate in the menu bar. A window “Variable Label and Type” is opened;
- Give the variable label and choose “Quantitative Single” as type of variable;
- Then validate your choice by clicking the OK button; otherwise, choose the Cancel option and start again.

A new table is displayed with the same column as the one chosen in the original table. To continue the design, click on “Window” and “Tile” to see the first table. There may be a possibility that you have to rearrange the windows.
To build a **categorical single variable:**

The procedure is the same as for a quantitative one, except that you have to choose the type “Categorical Single”. The selected column must contain integers or characters. At the end of the process, a new column is created in the new table.

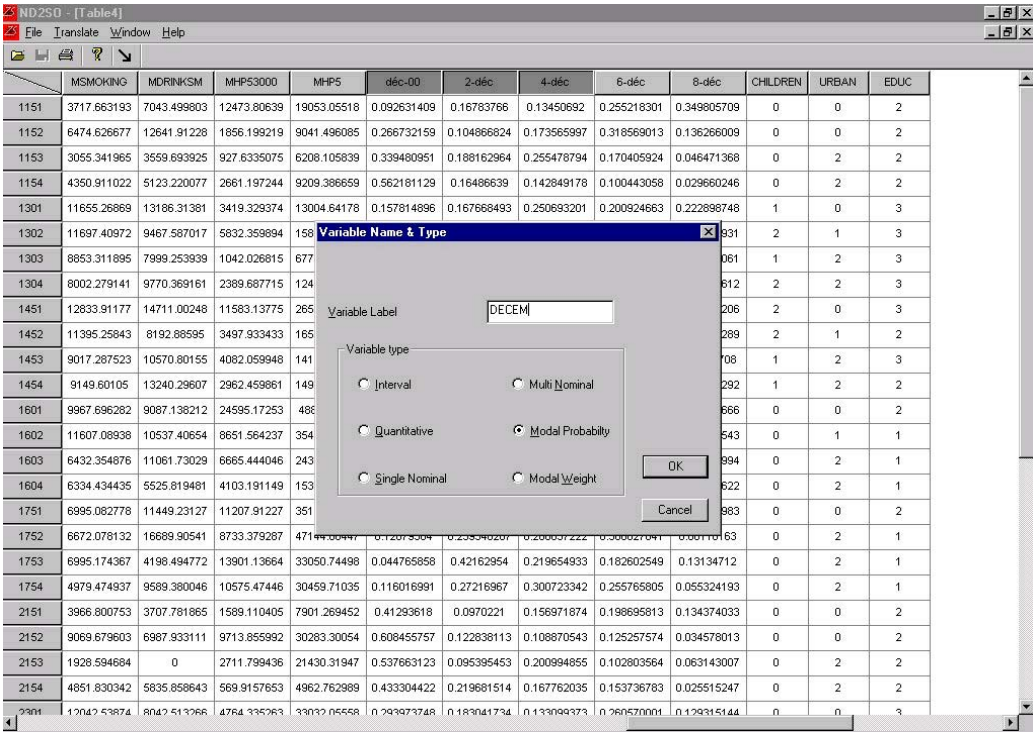


Figure 2: Choice of the variable type

To build a **categorical multi-valued symbolic variable:**

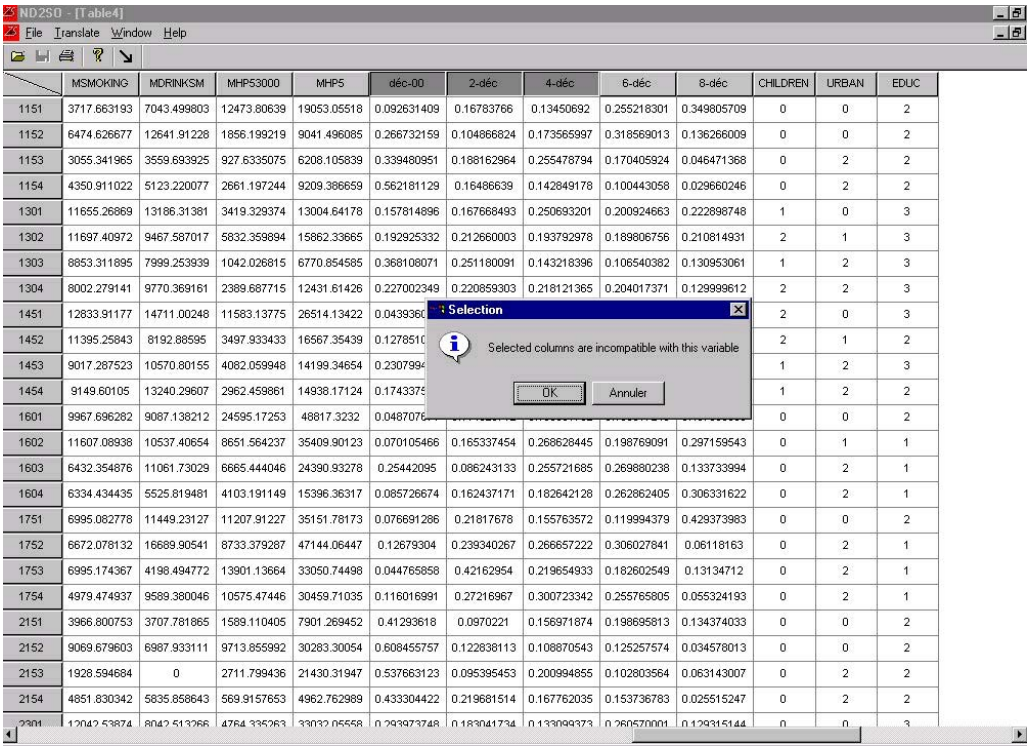
The original table must contain columns with 1 or 0. A “1” means that the category corresponding to this column is present in the line set. You have to:

Select in the original table all the columns corresponding to the categories of the new multi-valued categorical variable;
Select “translate”;
Give the label of the new variable and choose the type “categorical multi-valued”.

A new column is created in the symbolic table. In each cell will be displayed all the categories present for the symbolic object. The label of a category is the name of the corresponding column in the initial table.

To build a **modal symbolic variable:**

You have to select all the columns corresponding to the categories of the new modal variable. The label of the categories will be the names of the corresponding columns in the original table. The values in the original table represent probabilities or frequencies and are therefore numbers between 0 and 1. After giving the name of the new variable and choosing the type "Modal Probability" or "Modal Weight", a new column is created in the symbolic table (see Fig. 2). If "Modal Probability" has been selected, the program will verify that the values sum to one. If not, an error message will appear (see Fig. 3).



	MSMOKING	MDRINKSM	MHP53000	MHP5	déc-00	2-déc	4-déc	6-déc	8-déc	CHILDREN	URBAN	EDUC
1151	3717.663193	7043.499803	12473.80639	19053.05518	0.092631409	0.16783766	0.13450692	0.255218301	0.349805709	0	0	2
1152	6474.626677	12641.91228	1856.199219	9041.496085	0.266732159	0.104866824	0.173565997	0.318569013	0.136266009	0	0	2
1153	3055.341965	3559.693925	927.6335075	6208.105839	0.339480951	0.188162964	0.255478794	0.170405924	0.046471368	0	2	2
1154	4350.911022	5123.220077	2661.197244	9209.386659	0.562181129	0.16486639	0.142849178	0.100443058	0.029660246	0	2	2
1301	11655.26869	13186.31381	3419.329374	13004.64178	0.157814896	0.167668493	0.250693201	0.200924663	0.222898748	1	0	3
1302	11697.40972	9467.587017	5832.359894	15862.33665	0.192925332	0.212660003	0.193792978	0.189806756	0.210814931	2	1	3
1303	8853.311895	7999.253939	1042.026815	6770.854585	0.368108071	0.251180091	0.143218396	0.106540382	0.130953061	1	2	3
1304	8002.279141	9770.369161	2389.687715	12431.61426	0.227002349	0.220859303	0.218121365	0.204017371	0.129999612	2	2	3
1451	12833.91177	14711.00248	11583.13775	26514.13422	0.0439366					2	0	3
1452	11395.25843	8192.88595	3497.933433	16567.35439	0.1278510					2	1	2
1453	9017.287523	10570.80155	4082.059948	14199.34654	0.2307994					1	2	3
1454	9149.60105	13240.29607	2962.459861	14938.17124	0.1743375					1	2	2
1601	9967.696282	9087.138212	24595.17253	48817.3232	0.0487076					0	0	2
1602	11607.08938	10537.40654	8651.564237	35409.90123	0.070105466	0.165337454	0.268628445	0.198769091	0.297159543	0	1	1
1603	6432.354876	11061.73029	6665.444046	24390.93278	0.25442095	0.086243133	0.255721685	0.269880238	0.133733994	0	2	1
1604	6334.434435	5525.819481	4103.191149	15396.36317	0.085726674	0.162437171	0.182642128	0.262862405	0.306331622	0	2	1
1751	6995.082778	11449.23127	11207.91227	35151.78173	0.076691286	0.21817678	0.155763572	0.119994379	0.429373983	0	0	2
1752	6672.078132	16689.90541	8733.379287	47144.06447	0.12679304	0.239340267	0.266657222	0.306027841	0.06118163	0	2	1
1753	6995.174367	4198.494772	13901.13664	33050.74498	0.044765858	0.42162954	0.219654933	0.182602549	0.13134712	0	2	1
1754	4979.474937	9589.380046	10575.47446	30459.71035	0.116016991	0.27216967	0.300723342	0.255765805	0.055324193	0	2	1
2151	3966.800753	3707.781865	1589.110405	7901.269452	0.41293618	0.0970221	0.156971874	0.198695813	0.134374033	0	0	2
2152	9069.679603	6987.933111	9713.855992	30283.30054	0.608455757	0.122838113	0.108870543	0.125257574	0.034578013	0	0	2
2153	1928.594684	0	2711.799436	21430.31947	0.537663123	0.095395453	0.200994855	0.102803564	0.063143007	0	2	2
2154	4851.830342	5835.858643	569.9157653	4962.762989	0.433304422	0.219681514	0.167762035	0.153736783	0.025515247	0	2	2
2201	12042.53874	8042.513266	4764.335263	33032.05558	0.293973748	0.183041734	0.133099373	0.260570001	0.129315144	0	0	3

Figure 3: Error message

To build an **interval variable:**

You have to select the columns in the original file corresponding to min and max. After having selected the type "Interval" in the appropriate window, the program will ask you which one of both columns represents the minimum (see Fig. 4). Min is supposed to be smaller than max for all individuals; otherwise, an error message is displayed (see Fig. 3). A new column is then displayed in the symbolic data table (see Fig. 5).

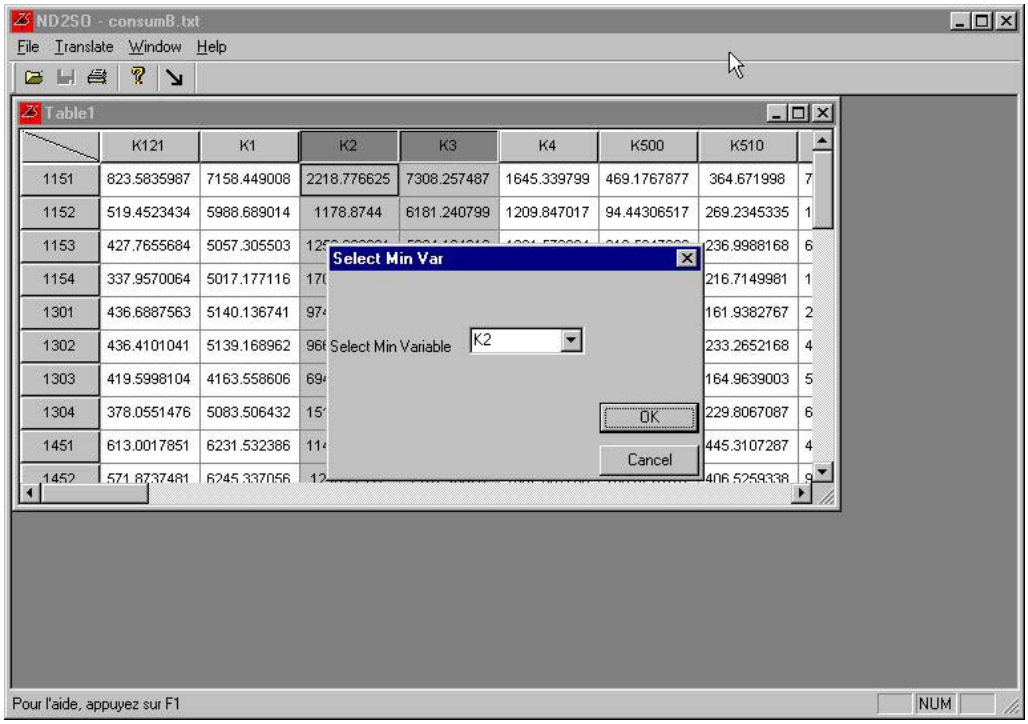


Figure 4: Select Minimum Variable

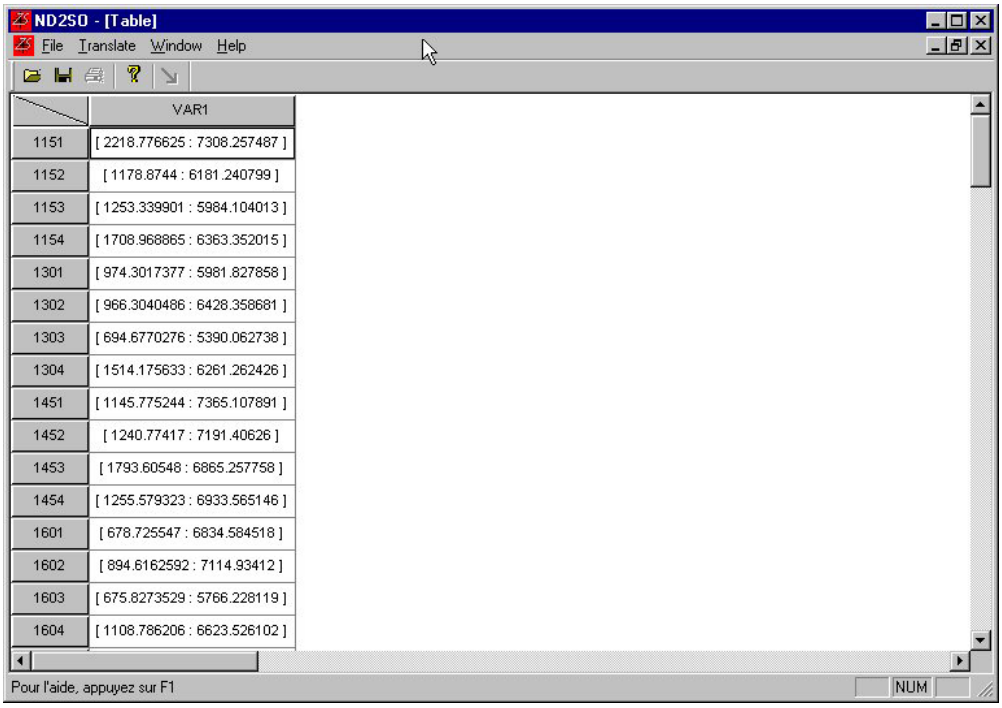


Figure 5: Resulting symbolic data table

Error message for double selection

When a selected column was already used to build a symbolic variable, an error message appears.

Export

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

SO2DB Tutorial

From Symbolic Objects to DataBase

Edited by DIB

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **01/12/2003**

SO2DB : From Symbolic Objects to DataBase

1 Introduction

SO2DB is a software developed to analyze a set of individuals stored into a relational database in order to discover a sub-set of them having a-priori fixed characteristics.

Therefore SO2DB interfaces a relational data model where each relation (table) corresponds to a simple, yet rigorously defined, description of individuals.

A relational database is a collection of tables eventually related by foreign key association (see Figure 1). Each table is composed by a fixed set of named columns and an arbitrary number of unnamed rows. Each column corresponds to an attribute. Each row stored into a table corresponds to an instance of what is represented by the table. A single data value v_{ij} is the value stored in the intersection of a i the row with the j the column. Each named column is associated with a domain, that is, the set of values that could be assigned to the corresponding attribute.

Figure 1. An example of a table(cars) stored in a relational database

Maker_name	Fuel_type	Aspiration	Num_of_doors	Body_style	Drive_wheels
Audi	Gas	Std	4	Sedan	Fwd
Audi	Gas	Turbo	2	Sedan	Fwd
BMW	Gas	Std	4	Wagon	Rwd
FIAT	Gas	Std	4	Sedan	Rwd

The standard query language SQL could be used to query database in order to retrieve individuals that satisfy some explicit conditions. For instance, considering the individual described in Figure 1, we can query the database in order to retrieve all information about the cars having sedan body style. In this case we execute the SQL query:

```
SELECT *
FROM cars
WHERE body_style="Sedan"
```

and retrieve the individuals described by the tuples:

(Audi,Gas,Std,4,Sedan,Fwd)
(Audi,Gas,Turbo,2,Sedan,Fwd)
(FIAT,Gas,Std,4,Sedan,Rwd)

The described process corresponds to retrieve the exact instances, in the table cars, of the symbolic object A intensionally described as A(Body_Syle=Sedan).

A symbolic object (SO) is the description of a class or a group of individuals by means of either set-valued variables or modal variables. More specifically a *boolean symbolic object* (BSO) is described by set-valued variables only, while a *probabilistic symbolic object* (PSO)

is described by modal variables with a relative frequency distribution associated to each of them. A symbolic object could be also described by both set-valued variables and modal variables.

The operation that allows the user to retrieve the instances of a symbolic object (SO) in a set of individuals is named matching. *Matching* is the process of comparing two or more structures to discover their likeness or differences. Similarity judgments in the matching process are directional: they have a *referent* and a *subject*. The former is either a prototype or a description of a class, while the latter is either a variant of the prototype or the description of an instance of the class. In its simplest form, matching is the process of comparing two structures just for equality. The test fails if the structures differ in at least one aspect. In more complex case, the matching compares the description of a class with the description of an individual in order to establish whether the individual can be considered an instance of the class.

Let us consider the symbolic object A and a tuple b retrieved through an SQL query, then Match(A,b) tests whether conditions expressed in A are satisfied by b (but not viceversa).

Match(A(body_style="Sedan)",b) is satisfied for b=((Audi,Gas,Std,4,Sedan,Fwd); (Audi,Gas,Turbo,2,Sedan,Fwd); (FIAT,Gas,Std,4,Sedan,Rwd)).

The symbolic object A represents the description of a class of objects and plays the role of the referent in the matching process, while the row b of the relational table corresponds to the description of an individual plays the role of subject: the problem consists of establishing whether the individual described by b can be considered as an instance of the class described by A. In its simplest form, matching compares two structures of patterns just for equality. More formally, given a symbolic object and a tuple of relational table, the former describing a class of individuals, the latter an individual, we want to check whether there is a match or not. Such a test is called *canonical matching* and returns either 0 (failure) or 1 (success).

What happen if we want to retrieve the instances of the symbolic object B(Body_style=sedan, Num_of_doors in [3,5],Aspiration ="Turbo")?. This corresponds to execute the SQL query:

SELECT *

FROM cars

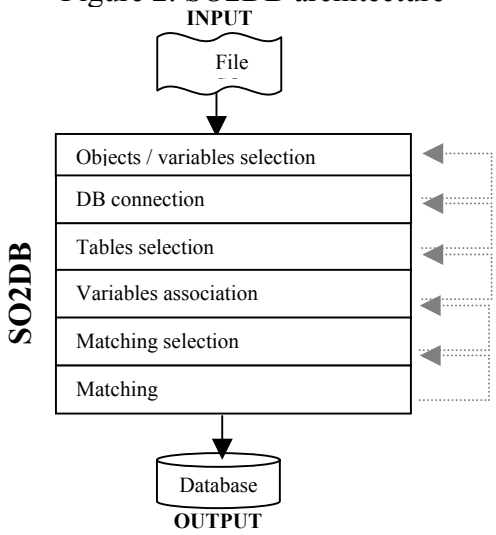
WHERE body_style="Sedan" and (Num_of_doors <=5 or Num_of_doors >=3) and Aspiration ="Turbo".

The set of individuals retrieved by running the SQL query above is empty.

Sometime it could be better to use a more flexible definition of matching that aims to compare two descriptions in order to identify their similarities rather than their equality.

SO2DB (see Figure 2) provides to retrieve the instances of the symbolic objects whose description is stored into an ASSO file. It provides a GUI to support the user in this task and uses the MATCH library to estimate the matching between a SO and the individual stored into a table of a relational database. Retrieved individuals are stored into a table of the input relational database.

Figure 2. SO2DB architecture



2 The input to SO2DB

The input to So2DB is an ASSO file containing the description of SOs and a table of a relational database.

A SOs may be described by the following symbolic variables:

- set-valued variables
  - categorical single (i.e. *town(w)=London* where '*town*' is the variable and *w* the individual)
  - categorical multi-value (i.e. *town(w)={London, Paris, Rome}*)
  - quantitative single valued (i.e. *height(w)=3.5*)
  - interval (i.e. *height(w)=[3,7]*)
- modal variables
  - probability distribution (i.e. *town(w)={London(0.2), Paris(0.7), Rome(0.1)}*)

Constraints (hierarchical dependencies and logical dependencies) are allowed for some dissimilarity measures. Missing values (i.e., unknown values) are also allowed.

According the mixture of these variables, the module can treat BSOs, PSOs and mixed BSOs and PSOs. The last kind of SOs are described by both set-valued and modal variables (i.e. w=[$age(w)$=[20,50] ∧ $sex(w)$={F(0.4), M(0.5), I(0.1)}]).

3 The output of SO2DB

A table stored in the input relational database that contains the instances of a sub-set of symbolic objects selected from the input ASSO file. These instances correspond to the tuples *b*, stored in the input database table, that satisfy the match(A,b) where A is one of the input selected symbolic object.

4 The method

Let us consider a symbolic object *A* and a table *T* of a relational database that describes individuals having the same structure of A, the goal is to retrieve all individuals in *T* that satisfy *A*.

How solve this problem? A solution is the matching that performs the directional comparison between a prototype of a class *C* with the description of a possible instance *b* of *C*. In its simplest form, matching is the process of comparing two structures just for equality. The test fails if the structures differ in at least one aspect. In more complex case, the matching compares the description of a class with the description of an individual in order to establish whether the individual can be considered an instance of the class. This correspond to the *canonical* matching and the *flexible* matching respectively (see MATCH tutorial for more details on the matching process).

In the case of canonical matching, SO2DB retrieve those individuals stored in a selected table of a relational database whose canonical matching against selected input SOs returns 1. This result can be obtained by simply executing the SQL query generated from the description of the SO. However, the canonical matching might be too strict for real-world problems, because of their inherent vagueness. The presence of symbolic variables that represent summarized information insert noise in date and could cause the failure of the matching process. This is the case of symbolic variable generated through the aggregation operators of SQL (Average, Max, Min, etc.). These type of variables could not correspond to any tuple in the table of individuals.

Hence it is possible to use the flexible matching that compares two descriptions in order to identify their similarities rather than their equality. The result of the flexible matching is a number, within the interval [0, 1], that indicates a degree of matching between a symbolic object and a tuple: in particular, flexible matching between A and b is equal to 1 if the canonical matching returns 1; otherwise, it is a value in [0, 1[. The user may introduce a threshold T in [0, 1[that represents the degree of flexible matching. The system outputs the individuals of database whose degree of match against the selected SO's is greater than or equal to T.

It is noteworthy that it is not possible to generate an SQL query from a SO when either the SO is described by summarized or modal variables, or the matching used is the flexible matching. Let's consider the following example:

We want to find the matching for the SO A = [Author = {Green, Bennet}] Title ={ Relational Databases}] [Pub_name = { Algodata Infosystem, New Moon Books }]

on the individuals (in table books):

Author	Title	Pub_name
Green	Relational Databases	Algoda InfoSystem
Green	II World War	New Moon Book
Carson	Art and Literature in Italy	Algoda InfoSystem
O'Leary	Flowers	Algoda InfoSystem
Straight	The blue eye	Algoda InfoSystem
Bennet	Relational Databases	Algoda InfoSystem
Dull	Oceans	Algoda InfoSystem

The query *Q*, obtained by *A* is the following:

SELECT *
FROM books
WHERE Author in ("Green", "Bennet") and Title in ("Relational Databases") and Pub_name in ("Algodata InfoSystem", "New Moon Books ")

By executing *Q* we retrieve the books:
((Green,Relational Databases, Algoda System); (Bennet, Relational Databases, Algoda InfoSyustem))
that is exactly the output obtained by means of the classical matching.
What happen if we decide to use the flexible matching function, *fm*, with a threshold *thr* = 0.85? We retrieve exactly those tuples *b* in *T* having *fm(A,b)* greater than *thr*. This means that we retrieve exactly the books:
((Green,Relational Databases, Algoda System); (Green, II World War, Algoda System); (Bennet, Relational Databases, Algoda InfoSyustem))
since:
fm(A, ((Green,Relational Databases, Algoda System))=1
fm(A, (Green, II World War, Algoda System))>0.85
fm(A, (Bennet, Relational Databases, Algoda InfoSyustem)) =1

Hence, retrieving of individuals in a database is a more complex operation than simply transforming an SO into an SQL query. Hence, the necessity of the module, named SO2DB, to support users in this process.

SO2DB supports the user in choosing all the parameters through a simple graphical wizard where:

1. the user opens an ASSO file that contains the description of a set of symbolic. Then he/she could selects a sub-set (or the entire set) of the variables used to describe each symbolic object;
2. the user chooses a sub-set of the symbolic objects whose description is stored in the input ASSO file;
3. the user selects a table in a relational database that contains individuals to be matched with selected SOs;
4. the user associates each symbolic variable in the symbolic objects with the corresponding attribute in the table and sets some attribute of the table as key attributes for the output;
    - the user chooses the type of matching (either canonical or flexible matching).

In the case of flexible matching, the user has to set a threshold in order to estimate the degree of matching among each SO and the individuals in the database.

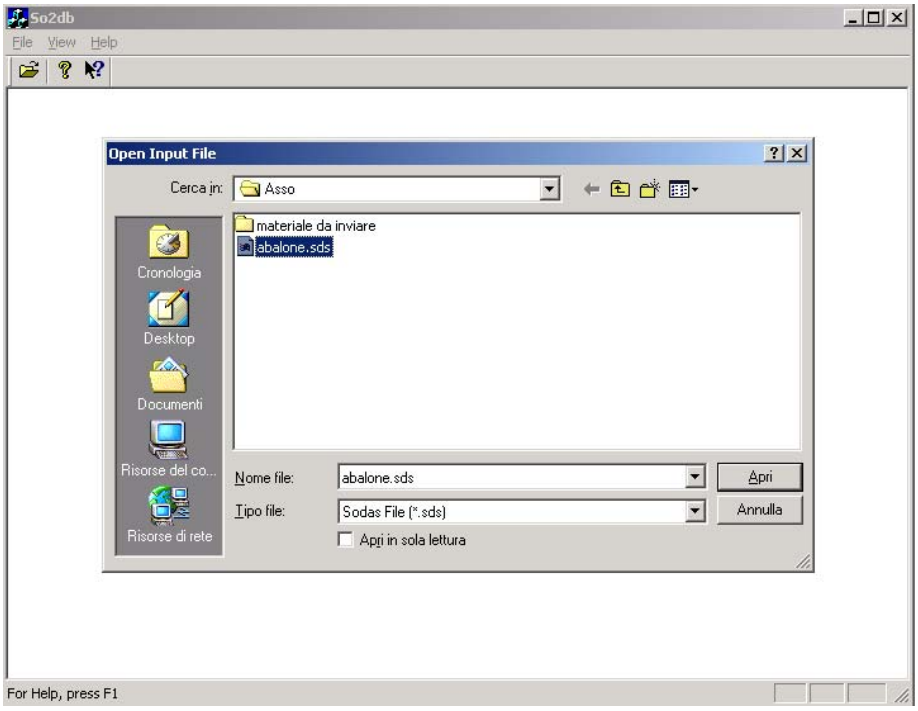
Finally the user chooses the name of the output table and could decide the type of the output between the option *one record for single matching* or *multiple records for single matching*.

5 An example

The SO2DB stand-alone version running command is SO2DB from prompt. Alternatively it could be opened by clicking on *Sodas file->Export->Export Asso SO2DB* of the main menu of the ASSO workbench.

The user opens an ASSO file (abalone.sds as in Figure 3). SO2DB opens a wizard that supports the user in choosing all parameters. Firstly, the user has to select the symbolic variables to be considered in matching process and SOs of interest (see figure 4).

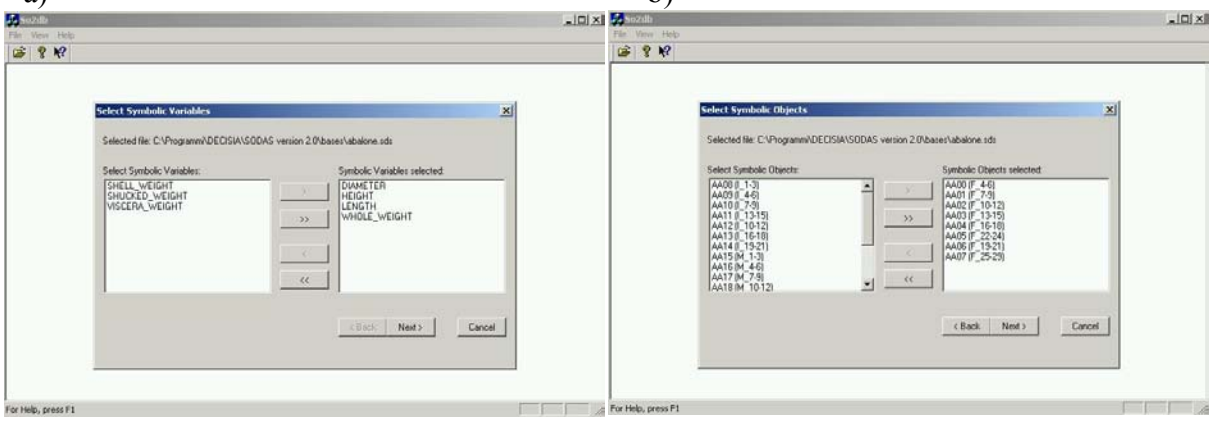
Figure 3. SO2DB: Open an input ASSO file



Then, he/she selects the SOs stored in input ASSO file of interest for the matching process (see Figure 4).

Figure 4. SO2DB: Select (a) Symbolic variables and (b) SOs

a) b)



Then the user opens a relational database (by means of an ODBC data source) and selects the table containing the individual of interest for the matching process (see Figure 5).

Figure 5. SO2DB: Select the table

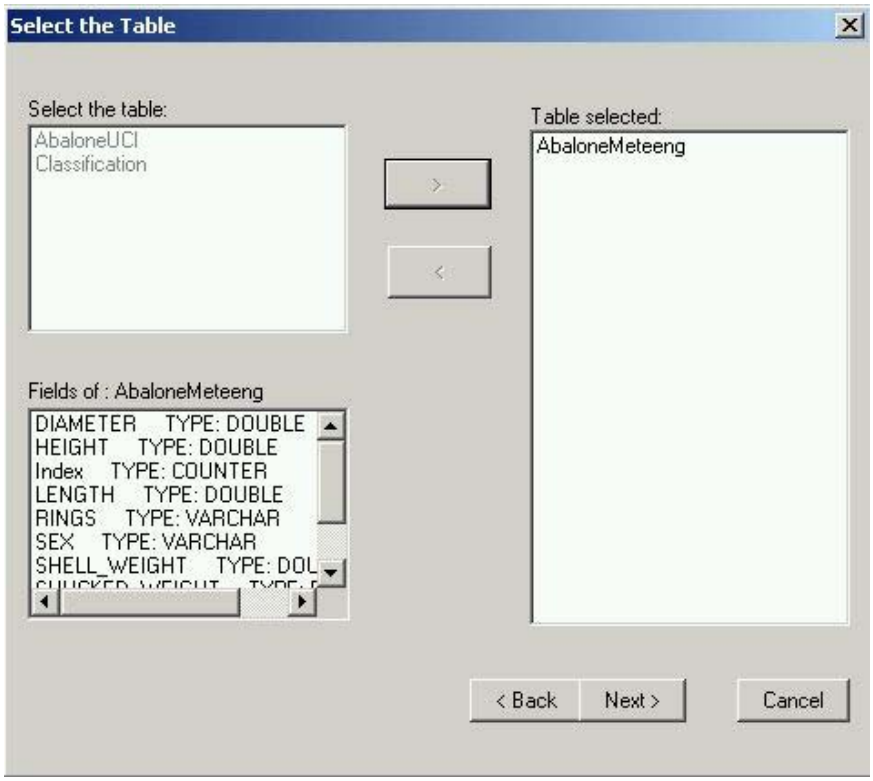
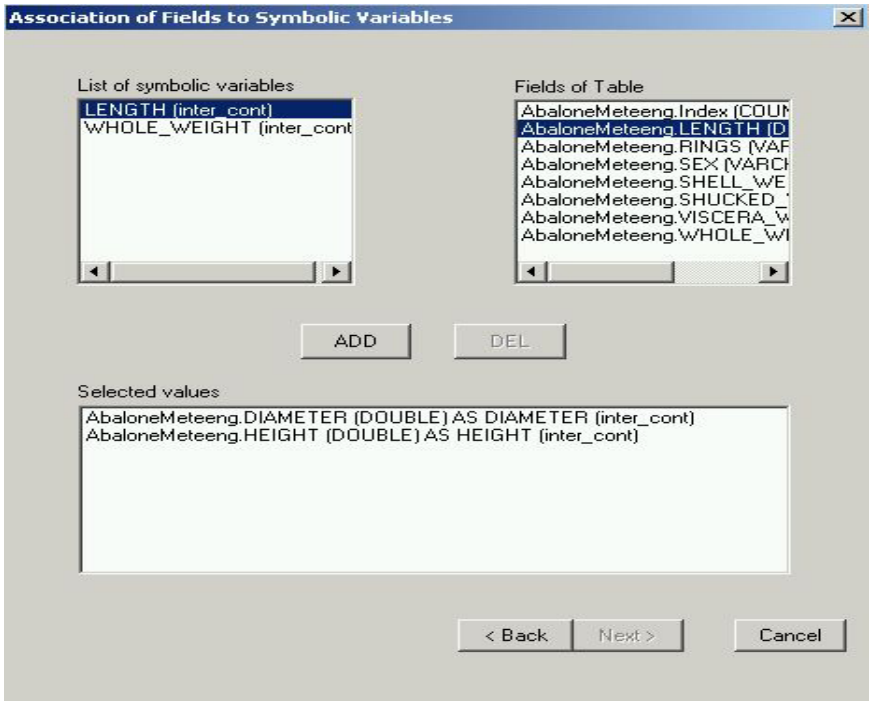


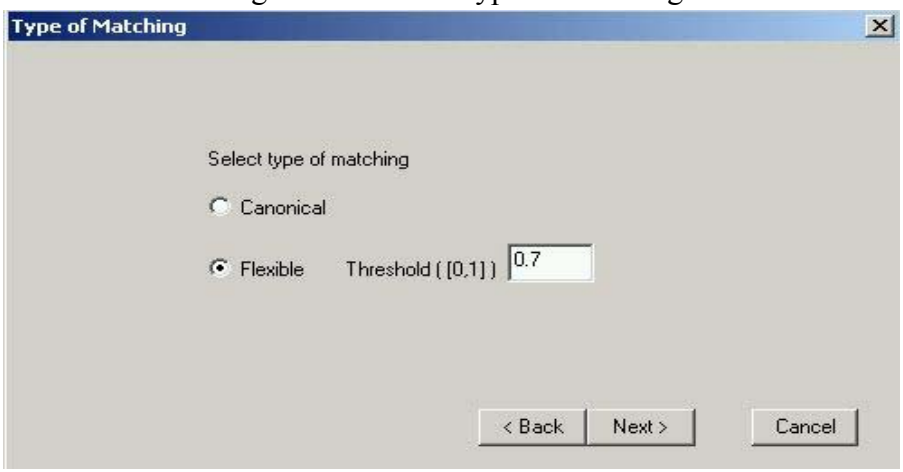
Figure 6. SO2DB: association of fields to Symbolic Variables



Now he/she has to create the binding between each symbolic variable of the SOs and an attribute of the selected table (see Figure 6). Eventually the user might decide to set some attributes of the table as key attributes for the output.

Finally the user decides the matching function. In the example in figure 7 he/she decides to perform a flexible matching with a threshold equal to 0.7.

Figure 7. SO2DB: type of matching



The output is stored in a table of the relational database. Furthermore the user decides if SO2DB has to create multiple output tuples for each individual belonging to the table AbaloneMeeting: one tuple for each matching (*One record for "Single" matching,* see Figure 8). Alternatively he/she might decide to create only a single tuple for each individual belonging to the table AbaloneMeeting and specify in the field *SO* the list of symbolic objects matched by the current tuple(*One record for "Multiple" matching*).

Figure 8. SO2DB: type of output

In the case the user has selected the *One record for "Single" matching* option combined with the *flexible matching* (see Figure 9) the output table "abaloneTest01" stored in abalone database will also contains the information about the degree of the computed flexible matching (*FM*)

Finally *Labels* option allows the user to associate the label to each symbolic objects in the computed output table.

Figure 9. SO2DB: Output

AbaloneTest01 : Tabella

DIAMETER	HEIGHT	LENGTH	WHOLE_WEIGHT	SO	FM
0,48	0,18	0,58	1,2495	AA00	0,7
0,475	0,17	0,595	1,247	AA00	0,7
0,495	0,185	0,595	1,285	AA00	0,7
0,475	0,135	0,6	1,4405	AA00	0,7
0,475	0,155	0,6	1,21	AA00	0,
0,475	0,18	0,6	1,162	AA00	0,7
0,475	0,18	0,6	1,1805	AA00	0,7
0,5	0,155	0,6	1,332	AA00	0,7
0,47	0,165	0,605	1,1775	AA00	0,7
0,475	0,175	0,605	1,382	AA00	0,8
0,48	0,175	0,605	1,1685	AA00	0,7
0,485	0,16	0,605	1,222	AA00	0,7
0,495	0,17	0,605	1,2385	AA00	0,7

Record: 1 di 898

Edit/New

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

SOEDIT User Manual

Symbolic Objects Edition

Edited by FUNDP

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **16/12/2003**

SOEDIT : Symbolic Objects Edition

The SOEDIT module allows you to visualise in a table all the symbolic objects that are present in a ".xml" file. It also allows you to perform basic modifications on symbolic data file like modifying a cell, adding or suppressing a column (i.e. a variable) or a line (i.e. a Symbolic Object), and sorting the Symbolic Objects. It can perform more sophisticated operations like:

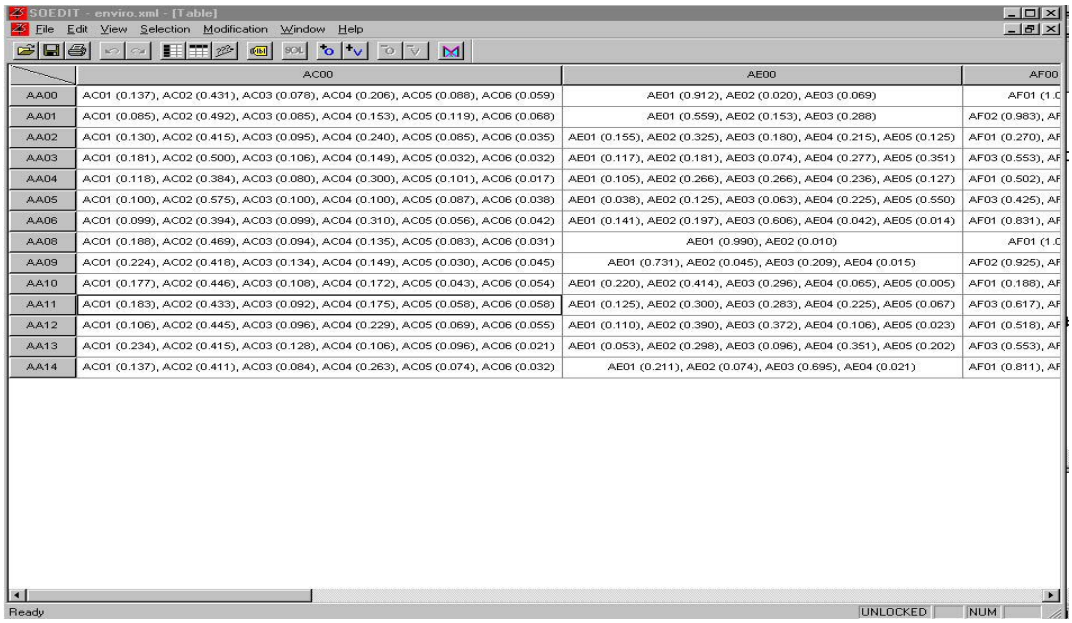
- Merging two symbolic data tables with regard to their lines or columns.
- Generalising Symbolic Objects by union or by intersection.
- Selecting Symbolic Objects according to a rule.
- Visualising the metadata.

The SOEDIT module can also be used to create symbolic data files from scratch. The different screens and menus are described in the help guide, which can is available online.

Here a few examples of use are presented, but the module has many other features that are fully described in the help guide.

1 Edition

In the principal screen of the workbench (after having selected a base for the chaining), select "Edit" in the methods for data management. The first screen of the Edit menu is displayed. With File and Open, simply select the file on which you want to work. The program displays the symbolic data table (Fig. 1).



	AC00	AE00	AF00
AA00	AC01 (0.137), AC02 (0.431), AC03 (0.078), AC04 (0.206), AC05 (0.088), AC06 (0.059)	AE01 (0.912), AE02 (0.020), AE03 (0.069)	AF01 (1.0
AA01	AC01 (0.085), AC02 (0.492), AC03 (0.085), AC04 (0.153), AC05 (0.119), AC06 (0.068)	AE01 (0.559), AE02 (0.153), AE03 (0.288)	AF02 (0.983), AF
AA02	AC01 (0.130), AC02 (0.415), AC03 (0.095), AC04 (0.240), AC05 (0.085), AC06 (0.035)	AE01 (0.155), AE02 (0.325), AE03 (0.180), AE04 (0.215), AE05 (0.125)	AF01 (0.270), AF
AA03	AC01 (0.181), AC02 (0.500), AC03 (0.106), AC04 (0.149), AC05 (0.032), AC06 (0.032)	AE01 (0.117), AE02 (0.181), AE03 (0.074), AE04 (0.277), AE05 (0.351)	AF03 (0.553), AF
AA04	AC01 (0.118), AC02 (0.384), AC03 (0.080), AC04 (0.300), AC05 (0.101), AC06 (0.017)	AE01 (0.105), AE02 (0.266), AE03 (0.266), AE04 (0.236), AE05 (0.127)	AF01 (0.502), AF
AA05	AC01 (0.100), AC02 (0.575), AC03 (0.100), AC04 (0.100), AC05 (0.087), AC06 (0.038)	AE01 (0.038), AE02 (0.125), AE03 (0.063), AE04 (0.225), AE05 (0.550)	AF03 (0.425), AF
AA06	AC01 (0.099), AC02 (0.394), AC03 (0.099), AC04 (0.310), AC05 (0.056), AC06 (0.042)	AE01 (0.141), AE02 (0.197), AE03 (0.606), AE04 (0.042), AE05 (0.014)	AF01 (0.831), AF
AA08	AC01 (0.188), AC02 (0.469), AC03 (0.094), AC04 (0.135), AC05 (0.083), AC06 (0.031)	AE01 (0.990), AE02 (0.010)	AF01 (1.0
AA09	AC01 (0.224), AC02 (0.418), AC03 (0.134), AC04 (0.149), AC05 (0.030), AC06 (0.045)	AE01 (0.731), AE02 (0.045), AE03 (0.209), AE04 (0.015)	AF02 (0.925), AF
AA10	AC01 (0.177), AC02 (0.446), AC03 (0.108), AC04 (0.172), AC05 (0.043), AC06 (0.054)	AE01 (0.220), AE02 (0.414), AE03 (0.296), AE04 (0.065), AE05 (0.005)	AF01 (0.188), AF
AA11	AC01 (0.183), AC02 (0.433), AC03 (0.092), AC04 (0.175), AC05 (0.058), AC06 (0.058)	AE01 (0.125), AE02 (0.300), AE03 (0.283), AE04 (0.225), AE05 (0.067)	AF03 (0.617), AF
AA12	AC01 (0.106), AC02 (0.445), AC03 (0.096), AC04 (0.229), AC05 (0.069), AC06 (0.055)	AE01 (0.110), AE02 (0.390), AE03 (0.372), AE04 (0.106), AE05 (0.023)	AF01 (0.518), AF
AA13	AC01 (0.234), AC02 (0.415), AC03 (0.128), AC04 (0.106), AC05 (0.096), AC06 (0.021)	AE01 (0.053), AE02 (0.298), AE03 (0.096), AE04 (0.351), AE05 (0.202)	AF03 (0.553), AF
AA14	AC01 (0.137), AC02 (0.411), AC03 (0.084), AC04 (0.263), AC05 (0.074), AC06 (0.032)	AE01 (0.211), AE02 (0.074), AE03 (0.695), AE04 (0.021)	AF01 (0.811), AF

Figure 1: Symbolic Data table

It is also possible to see the symbolic data under the form of symbolic assertions:

```
AA02 =
        AC00 = AC01 (0.130), AC02 (0.415), AC03 (0.095), AC04 (0.240), AC05 (0.085), AC06 (0.035)
And     AE00 = AE01 (0.155), AE02 (0.325), AE03 (0.180), AE04 (0.215), AE05 (0.125)
And     AF00 = AF01 (0.270), AF02 (0.730)
And     AG00 = AG01 (0.560), AG02 (0.220), AG03 (0.180), AG04 (0.040)
And     AH00 = [ -195.45 : 90.10 ]
And     AI00 = [ -75.00 : 191.17 ]
And     AJ00 = [ -592.28 : -299.04 ]
And     AK00 = [ -138.88 : 138.29 ]
And     AL00 = [ -161.18 : 124.50 ]
And     AM00 = [ -89.65 : 203.36 ]
And     AN00 = [ -368.92 : -88.57 ]
And     AO00 = [ -238.37 : 20.72 ]
And     AP00 = [ -190.28 : 83.12 ]
And     AQ00 = [ -124.20 : 158.01 ]
And     AR00 = [ -54.88 : 226.65 ]
And     AS00 = [ -120.52 : 164.49 ]
And     AT00 = [ -204.67 : 70.63 ]
```

Figure 2: Symbolic Data Assertions

In order to do that, first select in the table the objects (i.e. lines) for which you want the description by clicking on the first cell of the line. Then select “SOL” in the View menu.

1.1 To modify a value in a cell

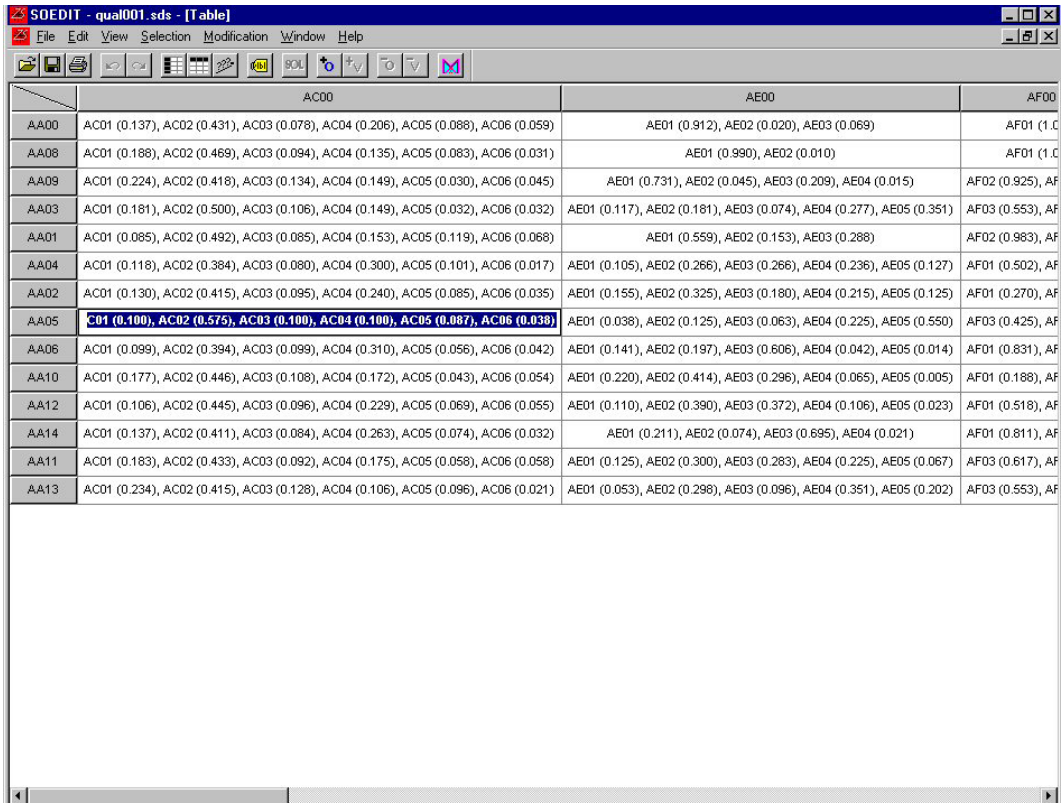
Click on a cell. If the variable’s type is Interval, you just have to modify the value in the cell.

SOEDIT - qual001.sds - [Table]

File Edit View Selection Modification Window Help

	AQ00	AR00	AS00	AT00	AA1
AA00	[-611.26 : -169.30]	[-140.04 : 272.75]	[-542.37 : -146.89]	[-840.00 : -574.57]	10.0000
AA08	[-560.45 : -133.03]	[-40.83 : 273.28]	[-399.76 : -21.17]	[-738.69 : -412.03]	11.0000
AA09	[-336.70 : 106.28]	[129.76 : 451.07]	[41.93 : 482.79]	[-753.14 : -364.43]	10.0000
AA03	[-22.66 : 364.63]	[-60.78 : 371.92]	[-1.31 : 442.54]	[-421.30 : -78.98]	11.0000
AA01	[-747.46 : -81.79]	[-215.07 : 308.08]	[-194.66 : 355.84]	[-669.76 : -235.98]	10.0000
AA04	[-114.89 : 138.60]	[-121.65 : 145.67]	[-223.08 : 13.32]	[223.76 : 471.23]	11.0000
AA02	[-124.20 : 158.01]	[-54.88 : 226.65]	[-120.52 : 164.49]	[-204.67 : 70.63]	10.0000
AA05	[-131.14 : 295.17]	[-199.26 : 285.65]	0 : 48.69]	[45.53 : 421.96]	11.0000
AA06	[-534.94 : 24.78]	[-601.51 : -101.45]	[-480.68 : -28.97]	[241.10 : 659.91]	10.0000
AA10	[-146.63 : 137.06]	[-79.89 : 184.66]	[-67.39 : 207.00]	[-333.55 : -59.44]	11.0000
AA12	[113.03 : 328.04]	[-325.55 : -66.06]	[-36.92 : 221.45]	[160.27 : 414.11]	12.0000
AA14	[-266.50 : 130.98]	[-449.08 : -26.49]	[-327.11 : 82.21]	[384.95 : 700.45]	14.0000
AA11	[31.15 : 327.04]	[-67.36 : 230.84]	[113.42 : 436.93]	[-472.96 : -98.71]	12.0000
AA13	[68.34 : 397.51]	[-271.75 : 170.47]	[-91.41 : 316.16]	[3.65 : 431.00]	12.0000

Figure 3 : Modification of an interval variable



SOEDIT - qual001.sds - [Table]

File Edit View Selection Modification Window Help

	AC00	AE00	AF00
AA00	AC01 (0.137), AC02 (0.431), AC03 (0.078), AC04 (0.206), AC05 (0.088), AC06 (0.059)	AE01 (0.912), AE02 (0.020), AE03 (0.069)	AF01 (1.0
AA08	AC01 (0.188), AC02 (0.469), AC03 (0.094), AC04 (0.135), AC05 (0.083), AC06 (0.031)	AE01 (0.990), AE02 (0.010)	AF01 (1.0
AA09	AC01 (0.224), AC02 (0.418), AC03 (0.134), AC04 (0.149), AC05 (0.030), AC06 (0.045)	AE01 (0.731), AE02 (0.045), AE03 (0.209), AE04 (0.015)	AF02 (0.925), AF
AA03	AC01 (0.181), AC02 (0.500), AC03 (0.106), AC04 (0.149), AC05 (0.032), AC06 (0.032)	AE01 (0.117), AE02 (0.181), AE03 (0.074), AE04 (0.277), AE05 (0.351)	AF03 (0.553), AF
AA01	AC01 (0.085), AC02 (0.492), AC03 (0.085), AC04 (0.153), AC05 (0.119), AC06 (0.068)	AE01 (0.559), AE02 (0.153), AE03 (0.288)	AF02 (0.983), AF
AA04	AC01 (0.118), AC02 (0.384), AC03 (0.080), AC04 (0.300), AC05 (0.101), AC06 (0.017)	AE01 (0.105), AE02 (0.266), AE03 (0.266), AE04 (0.236), AE05 (0.127)	AF01 (0.502), AF
AA02	AC01 (0.130), AC02 (0.415), AC03 (0.095), AC04 (0.240), AC05 (0.085), AC06 (0.035)	AE01 (0.155), AE02 (0.325), AE03 (0.180), AE04 (0.215), AE05 (0.125)	AF01 (0.270), AF
AA05	C01 (0.100), AC02 (0.575), AC03 (0.100), AC04 (0.100), AC05 (0.087), AC06 (0.038)	AE01 (0.038), AE02 (0.125), AE03 (0.063), AE04 (0.225), AE05 (0.550)	AF03 (0.425), AF
AA06	AC01 (0.099), AC02 (0.394), AC03 (0.099), AC04 (0.310), AC05 (0.056), AC06 (0.042)	AE01 (0.141), AE02 (0.197), AE03 (0.606), AE04 (0.042), AE05 (0.014)	AF01 (0.831), AF
AA10	AC01 (0.177), AC02 (0.446), AC03 (0.108), AC04 (0.172), AC05 (0.043), AC06 (0.054)	AE01 (0.220), AE02 (0.414), AE03 (0.296), AE04 (0.065), AE05 (0.005)	AF01 (0.188), AF
AA12	AC01 (0.106), AC02 (0.445), AC03 (0.096), AC04 (0.229), AC05 (0.069), AC06 (0.055)	AE01 (0.110), AE02 (0.390), AE03 (0.372), AE04 (0.106), AE05 (0.023)	AF01 (0.518), AF
AA14	AC01 (0.137), AC02 (0.411), AC03 (0.084), AC04 (0.263), AC05 (0.074), AC06 (0.032)	AE01 (0.211), AE02 (0.074), AE03 (0.695), AE04 (0.021)	AF01 (0.811), AF
AA11	AC01 (0.183), AC02 (0.433), AC03 (0.092), AC04 (0.175), AC05 (0.058), AC06 (0.058)	AE01 (0.125), AE02 (0.300), AE03 (0.283), AE04 (0.225), AE05 (0.067)	AF03 (0.617), AF
AA13	AC01 (0.234), AC02 (0.415), AC03 (0.128), AC04 (0.106), AC05 (0.096), AC06 (0.021)	AE01 (0.053), AE02 (0.298), AE03 (0.096), AE04 (0.351), AE05 (0.202)	AF03 (0.553), AF

Figure 4: Modification of a modal variable

1.2 To add a new Symbolic Object

Choose "Add Symbolic Object" in the Edit menu and enter a code and a label for the new object. The code normally is the key given to the variable or object by DB2SO when it builds the symbolic data. The label is the name of the object or variable as defined by the user, and can be considered as a small description. After these 2 values have been determined, the program asks you to save the Symbolic Table with that new object added to it in a (new) file. That new object is displayed in the table under the shape of a new line. A small message is displayed inviting you to change the values of the variables for that object. The values for the variables are per default the same as the ones of the first object in the table. This way, you have an example of the value format that you have to use. You just have to replace the default values by new ones; you also have to give the true label to the new object.

2 To add a new Symbolic Variable

Choose "Add Variable" in the "Edit" menu. You will first be required to choose the new variable's type, and then to enter a code and a label for that new variable. Depending on the type of variable that you have decided to choose, you may have to give more information. The different possibilities are described below:

For a categorical variable (categorical single, multiple or modal)

First, you have to indicate the number of modalities, and then give for each modality the code and the label. Finally you are invited to give:

- For a categorical single variable: the order of the modalities.

- For a categorical multiple variable: the present categories, filling 1 in a table when the category is present.

- For a modal variable: its categories and related weights (see Fig. 5). Notice that the sum of the weights of one object (i.e. one line) must be equal to 1.

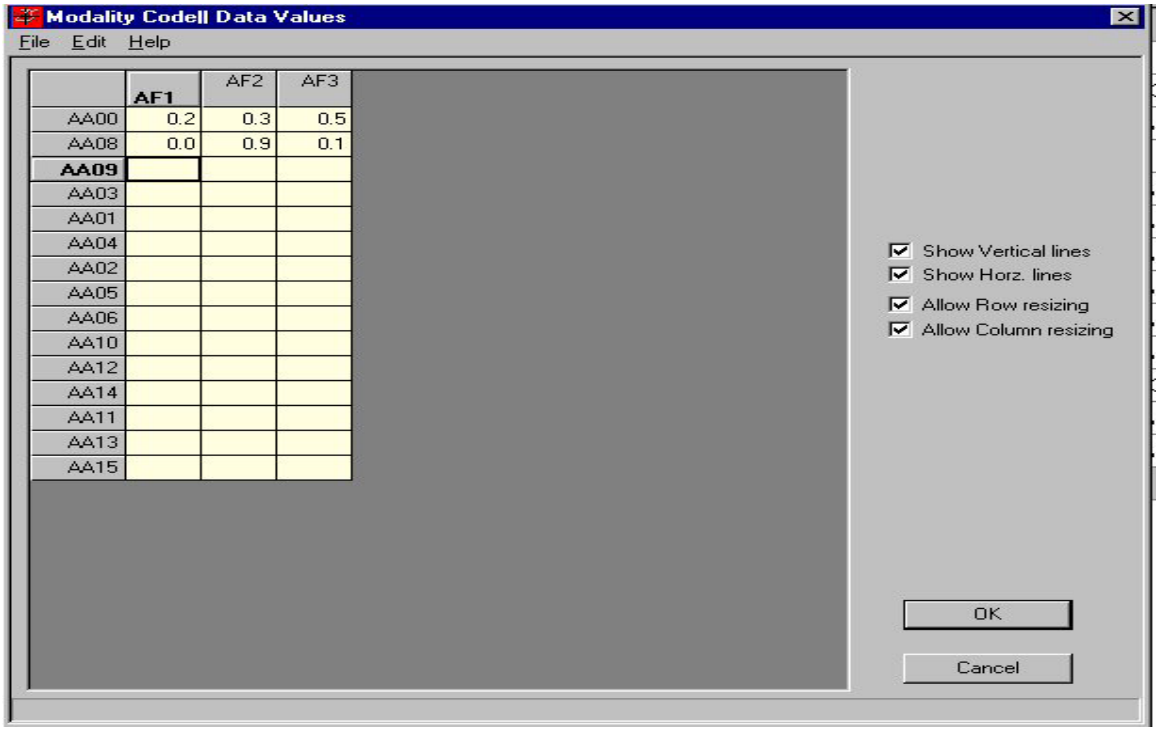


Figure 5: Window for assigning weights to modal variable

For an interval variable

You are invited to give the min and max values and then to save in a new SODAS file.

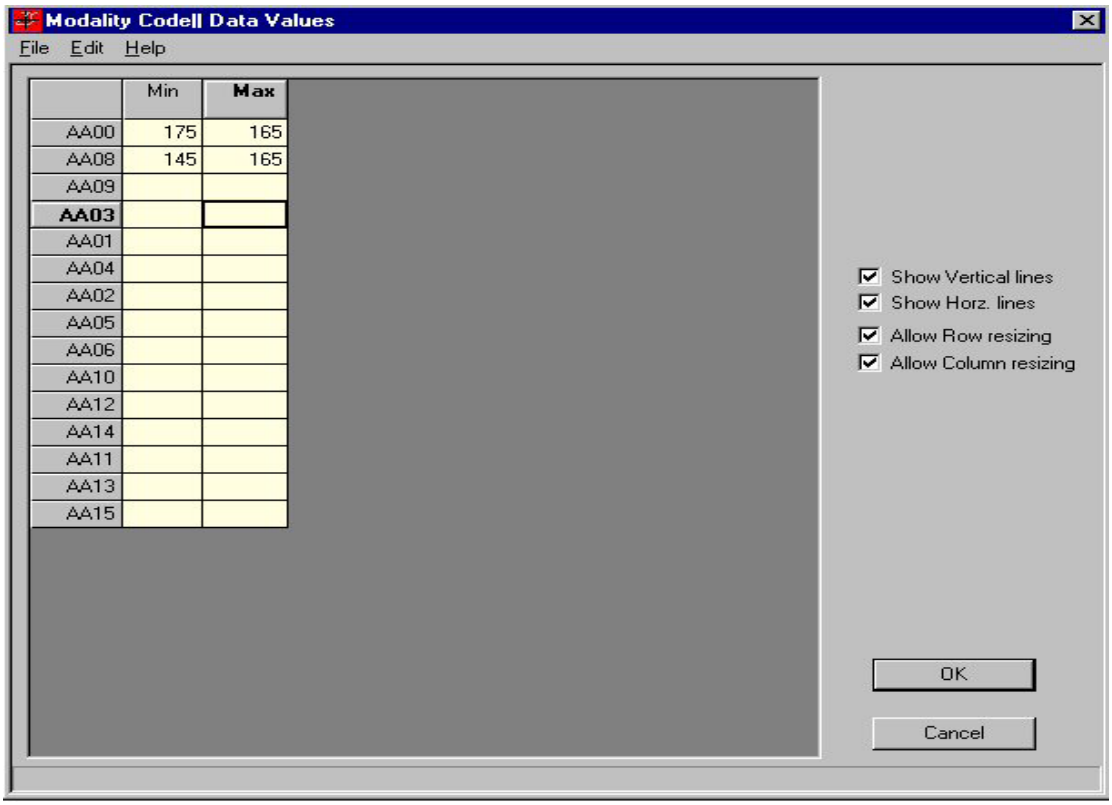


Figure 6: Window for giving min and max values to an interval variable

3 Selection/Sorting/Changing the order

You can decide to select only part of the objects or part of the variables. In order to achieve this, select the **objects** or the **variables** directly in the table by clicking on the first cell of the line/column. You can also use the Selection menu where you choose Symbolic Objects or variables. You can choose part of the **categories** by clicking "Selection" and "Categories". The selection is done in the standard SODAS way: two lists are shown. The list on the top contains the available elements. The list on the bottom contains the selected elements (left/right for categories). Elements can be added and / or removed by using the four buttons.

The order defined in the list will be respected in the table. This is particularly useful when you want to **change the order** of the categories. You can save your selection with the "Save as" option. You can of course open a previous selection.

It is possible to **select objects respecting a rule** on the values of a particular **quantitative variable** (single or interval). First choose "Type of Rules" in the Selection menu and then "Objects Selection Rule by Value". A window invites you to give the label of the variable and to choose the rule. This rule can be of the form: min ≥ or ≤ and/or max ≥ or ≤ "value".

You can also **select Symbolic Objects** for which a **categorical variable** (single or multiple) has a particular categorical value. Choose "Type of Rules" in the Selection menu and then "Objects Selection Rule by Category". You then have to choose the label of the variable and to select the categories for selection.

The Symbolic Objects can be sorted in alphabetical order: select “Type of Sorting” in the “Selection” menu and “SO Sorting by Label”. They can also be rearranged according to the values of a Quantitative single variable or of the min value of an Interval variable. Select “Type of Sorting” in the Selection menu and then select the “SO Sorting by Variable” option. You then have to give the label of the variables.

4 Generalisation of Symbolic Objects

You can create a new Symbolic Object which generalises existing ones. You have the choice between Generalisation by union, Generalisation by intersection, Generalisation by Supremum and Generalisation by Infimum. First, select at least two Symbolic Objects. Then choose in the Edit menu the type of Generalisation that you want to see applied. The program creates a new file with the new generated Symbolic Object. If you want to add this new Symbolic Object to the previous file, simply use the “Merge” option (see below). This is only permitted if the new Symbolic Object’s variables have got the same type as the previous Symbolic Object’s variables.

- With modal variables, Generalisation by Union is possible only if you know the weight of the symbolic object (i.e. the number of individuals in the Symbolic Object).
- Generalisation by intersection if only possible with interval variables and categorical multiple variables
- Generalisation by Supremum or by Infimum is available only for modal variables.

For an interval variable, the Generalisation by Union algorithm takes the minimum among the min values and the maximum among the max values as min and max.

For a categorical variable, if the size of the SOs is known, the algorithm recomputes the exact frequencies.

If you have chosen “Generalisation by intersection”, then for an interval variable, the algorithm computes min = max of the min, and max = min of the max. This interval could be 0. If min > max, generalisation is not possible.

5 Merging Symbolic data tables

According to Symbolic Objects

The first file must be opened before selecting “Merging → SO” in the Edit menu. You are then invited to give the name of the second file to be merged. The two files must have the same objects (i.e. the two files’ objects must have the same label and code), and these objects must be in the **same order**. A new file is created with the variables of the two files. You then have to give the name of the new merged file. If the two files do not contain the same objects, Merging is impossible and an error message is displayed.

According to variables

The first file must be opened before selecting “Merging → variables” in the Edit menu. You are then invited to give the name of the second file to be merged. The two files must have the same variables, and these variables must be in the **same order**. A new file is created with these variables and with the objects of the two files. If the two files do not contain the same variables, an error message notifies you of the fact that Merging is impossible.

6 Metadata

The Edit module allows you to view the metadata which have been recorded for the Symbolic Data file. If you right click on the top cell of a column in the table and position the mouse still above that cell, you will view the characteristics of the corresponding variable. Pay attention to the fact that if you put the mouse above another top cell than the one on which you last right clicked, the data displayed will still be data corresponding to that last cell. You have to click on the cell about which you want information to be displayed.

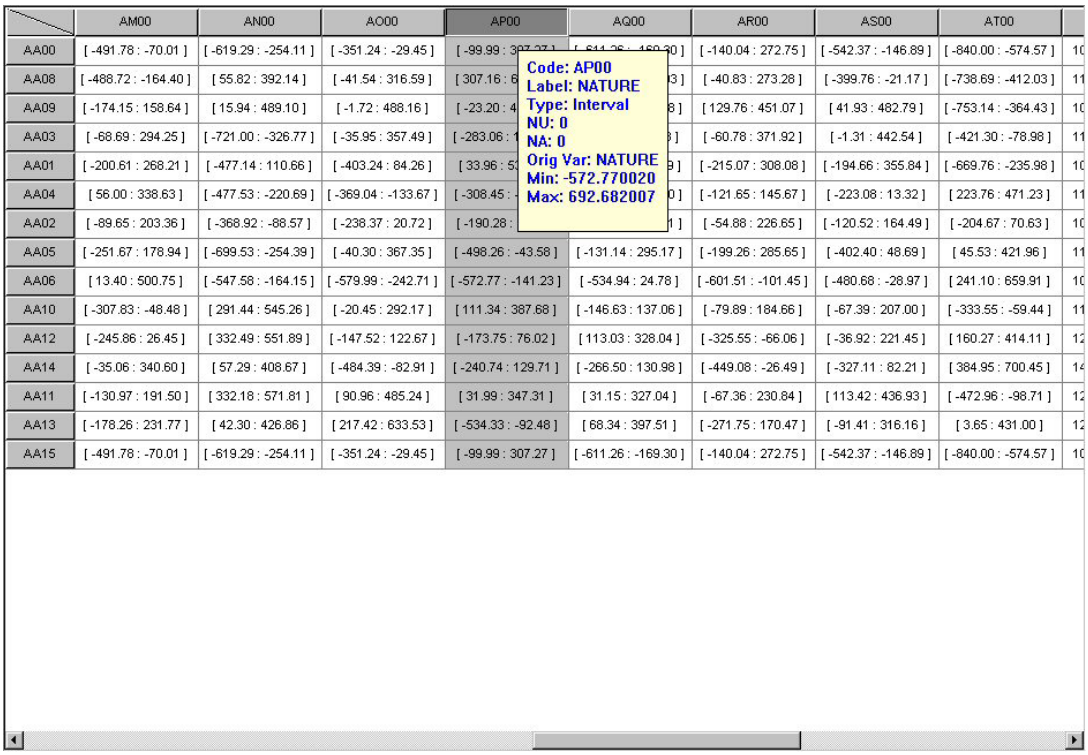
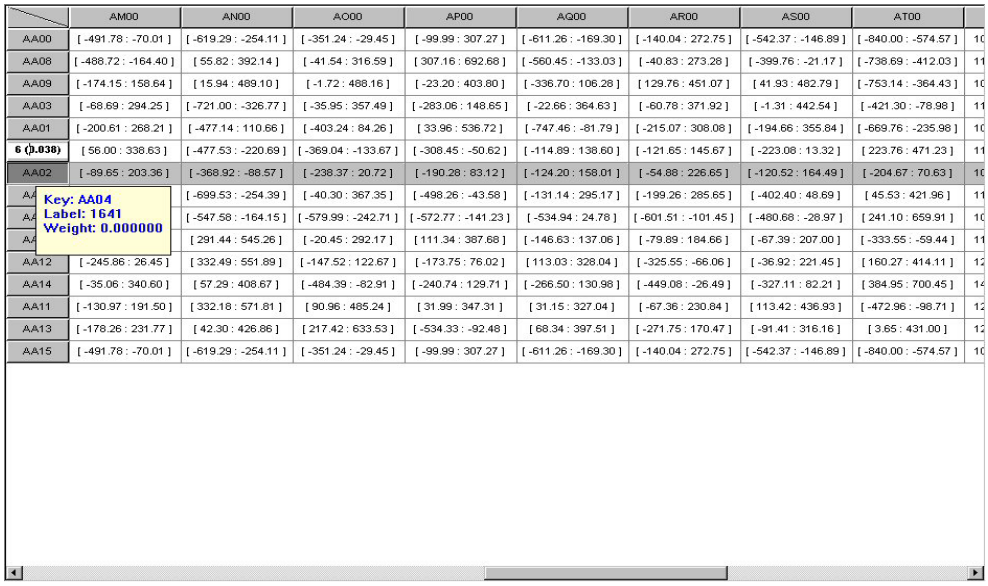


Figure 7: Metadata for a variable



	AM00	AN00	AO00	AP00	AQ00	AR00	AS00	AT00
AA00	[-491.78 : -70.01]	[-619.29 : -254.11]	[-351.24 : -29.45]	[-99.99 : 307.27]	[-611.26 : -169.30]	[-140.04 : 272.75]	[-542.37 : -146.89]	[-840.00 : -574.57]
AA08	[-488.72 : -164.40]	[55.82 : 392.14]	[-41.54 : 316.59]	[307.16 : 692.68]	[-560.45 : -133.03]	[-40.83 : 273.28]	[-399.76 : -21.17]	[-738.69 : -412.03]
AA09	[-174.15 : 158.64]	[15.94 : 489.10]	[-1.72 : 488.16]	[-23.20 : 403.80]	[-336.70 : 106.28]	[129.76 : 451.07]	[41.93 : 482.79]	[-753.14 : -364.43]
AA03	[-68.69 : 294.25]	[-721.00 : -326.77]	[-35.95 : 357.49]	[-283.06 : 148.65]	[-22.66 : 364.63]	[-60.78 : 371.92]	[-1.31 : 442.54]	[-421.30 : -78.98]
AA01	[-200.61 : 268.21]	[-477.14 : 110.66]	[-403.24 : 84.26]	[33.96 : 536.72]	[-747.46 : -81.79]	[-215.07 : 308.08]	[-194.66 : 355.84]	[-669.76 : -235.98]
6 (0.038)	[56.00 : 338.63]	[-477.53 : -220.69]	[-369.04 : -133.67]	[-308.45 : -50.62]	[-114.89 : 138.60]	[-121.65 : 145.67]	[-223.08 : 13.32]	[223.76 : 471.23]
AA02	[-89.65 : 203.36]	[-368.92 : -88.57]	[-238.37 : 20.72]	[-190.28 : 83.12]	[-124.20 : 158.01]	[-54.88 : 226.65]	[-120.52 : 164.49]	[-204.67 : 70.63]
AA		[-699.53 : -254.39]	[-40.30 : 367.35]	[-498.26 : -43.58]	[-131.14 : 295.17]	[-199.26 : 285.65]	[-402.40 : 48.69]	[45.53 : 421.96]
AA		[-547.58 : -164.15]	[-579.99 : -242.71]	[-572.77 : -141.23]	[-534.94 : 24.78]	[-601.51 : -101.45]	[-480.68 : -28.97]	[241.10 : 659.91]
AA		[291.44 : 545.26]	[-20.45 : 292.17]	[111.34 : 387.68]	[-146.63 : 137.06]	[-79.89 : 184.66]	[-67.39 : 207.00]	[-333.55 : -59.44]
AA12	[-245.86 : 26.45]	[332.49 : 551.89]	[-147.52 : 122.67]	[-173.75 : 76.02]	[113.03 : 328.04]	[-325.55 : -66.06]	[-36.92 : 221.45]	[160.27 : 414.11]
AA14	[-35.06 : 340.60]	[57.29 : 408.67]	[-484.39 : -82.91]	[-240.74 : 129.71]	[-266.50 : 130.98]	[-449.08 : -26.49]	[-327.11 : 82.21]	[384.95 : 700.45]
AA11	[-130.97 : 191.50]	[332.18 : 571.81]	[90.96 : 485.24]	[31.99 : 347.31]	[31.15 : 327.04]	[-67.36 : 230.84]	[113.42 : 436.93]	[-472.96 : -98.71]
AA13	[-178.26 : 231.77]	[42.30 : 426.86]	[217.42 : 633.53]	[-534.33 : -92.48]	[68.34 : 397.51]	[-271.75 : 170.47]	[-91.41 : 316.16]	[3.65 : 431.00]
AA15	[-491.78 : -70.01]	[-619.29 : -254.11]	[-351.24 : -29.45]	[-99.99 : 307.27]	[-611.26 : -169.30]	[-140.04 : 272.75]	[-542.37 : -146.89]	[-840.00 : -574.57]

Figure 8: Metadata for an object

When using the icon of the tool bar, you will be able to see all the metadata of the Symbolic Data Table. They are organised within the following paragraphs:

- File general characteristics
- Survey description
- Symbolic objects
- Symbolic variables
- Original variables
- Original Files

For viewing the metadata of a Symbolic Object / variable, you first have to select the label of that Symbolic Object / variable.

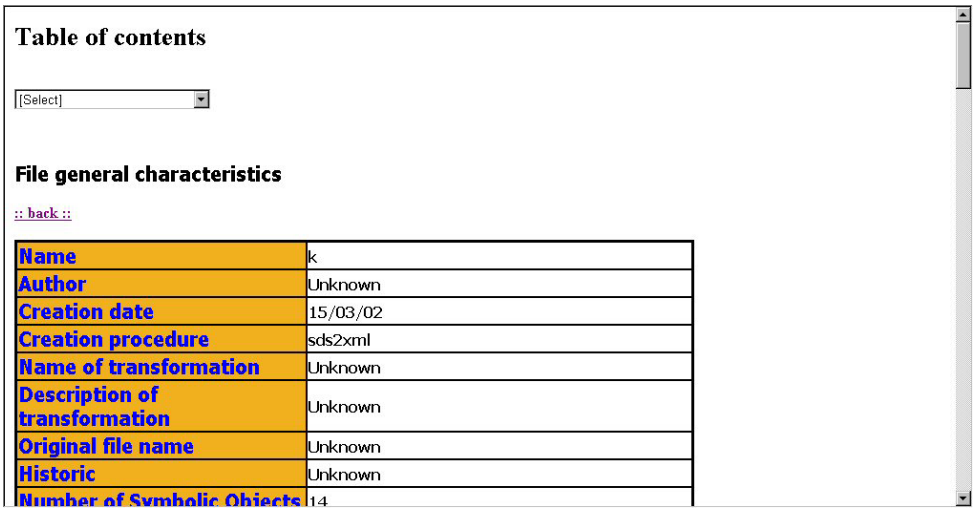


Table of contents

[Select]

File general characteristics

:: back ::

Name	k
Author	Unknown
Creation date	15/03/02
Creation procedure	sds2xml
Name of transformation	Unknown
Description of transformation	Unknown
Original file name	Unknown
Historic	Unknown
Number of Symbolic Objects	14

Figure 9: Metadata "File general characteristics"

TREATMENT METHODS

Descriptive Statistics and Visualisation

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

DSTAT User Manual

Descriptive Statistics

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **11/12/2003**

DSTAT : Descriptive Statistics

1 Introduction

DSTAT stands for “Descriptive STATistics”.

The DSTAT method is actually a set of methods, which aim to extend to the symbolic variables statistical methods usually applied to conventional variables.

	Variable type		
Method	Categorical multi-valued	Interval	Modal
Frequencies for categorical multi-valued	X		
Frequencies for interval		X	
Biplot		X	
Capacities			X
Numeric and symbolic characteristics		X	

Central object: just deals with the individuals (symbolic objects).

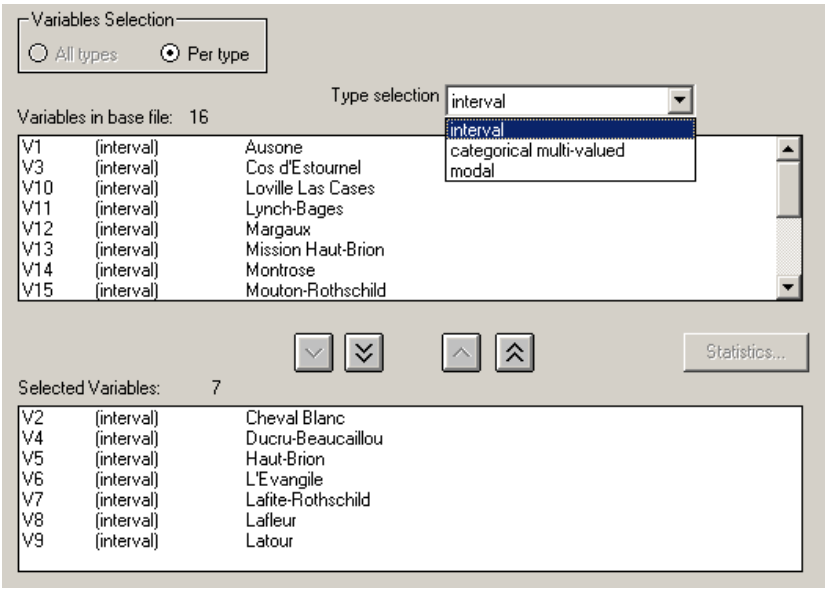
The input to DSTAT is an ASSO symbolic objects file of type .SDS or .XML.

The above three variable types can be mixed in the file.

2 Selecting the variables and the method in the workbench

In the **Variables panel**, you have to select the variables type matching the method you intend to select (or have already selected), then select all or part of the variables inside this type.

Example for interval variables (the other types are handled in the same way):

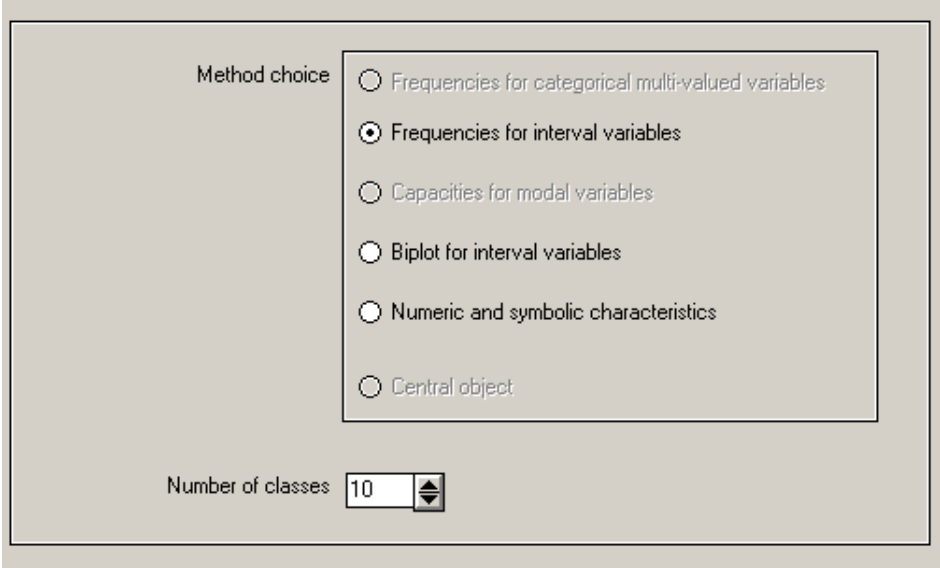


It is advisable to select all the variables, since DSTAT will later let you select those you want to effectively process, in an interactive way; i.e., you may change the selection at will while staying inside DSTAT (no need to exit, return to the workbench then restart).

Selecting part of the variables here is useful only if you wish to restrict the set of variables that will later be offered by DSTAT for selection.

In case of "Frequencies for categorical multi-valued variables", DSTAT will take all the applicable variables regardless of whether the user selected all of them or part of them; it is a must for this method.

The other step (next or previous) is to select the method in the **Parameters panel**.



The methods are enabled according to the contents of the data file, not according to the variable type selected in the other panel, because it is possible to select the method before the variables type. It is up to the user to match both; in case of mismatch the execution does not start and an explanatory warning message shows up.

The **Central object** method just deals with the individuals (symbolic objects); it does not depend on any variable type; it is enabled when the data file contains a distance matrix.

The **Number of classes** parameter only applies to the "Frequencies for interval variables"; it is disabled for the other options.

It tells DSTAT in how many "classes", i.e. sub-intervals, it should divide the overall interval for each variable.

3 Running the method

The F5 key triggers the method execution, which upon completion will display the listing icon and, if applicable, the graph icon.

The ouput depends on the method selected:

	Listing	Graph
Frequencies for categorical multi-valued	X	X
Frequencies for interval	X	X
Capacities	X	X
Biplot		X
Numeric and symbolic characteristics	X (*)	
Central object	X (*)	

(*) The results are also dynamically displayed in a dialog or message window at execution time.

4 Displaying the listing

If there has been an error in the execution, the listing contains a description of that error.

If not, it contains the results of the calculation; example for the Frequencies for interval variables:

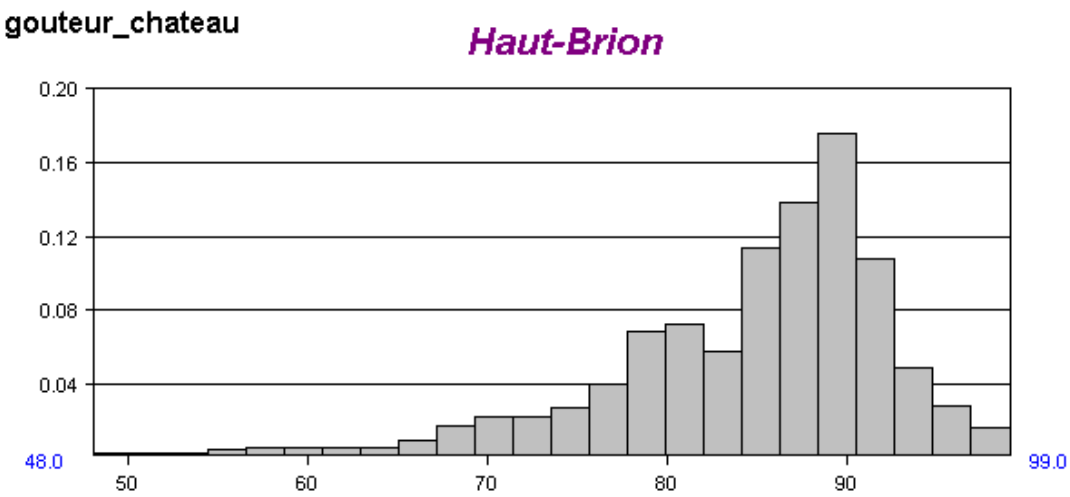
```
-----------------------------------------------------------------------------
 ASSO - DSTAT  RELATIVE FREQUENCIES (INTERVAL)         Dec 08 2003  12:13

 File:  GOUTEUR.XML
 Title: gouteur_chateau
-----------------------------------------------------------------------------

  Ausone

    limits: 35.0 - 98.0    class width: 6.3

    class  1  0.0086
    class  2  0.0086
    class  3  0.0203
    class  4  0.0390
    class  5  0.0882
    class  6  0.1148
    class  7  0.1755
    class  8  0.2989
    class  9  0.2013
    class 10  0.0450

    Central tendancy: 77.5438
    Dispersion: 11.1105

  Cheval Blanc

    limits: 40.0 - 99.0    class width: 5.9
 ...............
```

5 Displaying the graph

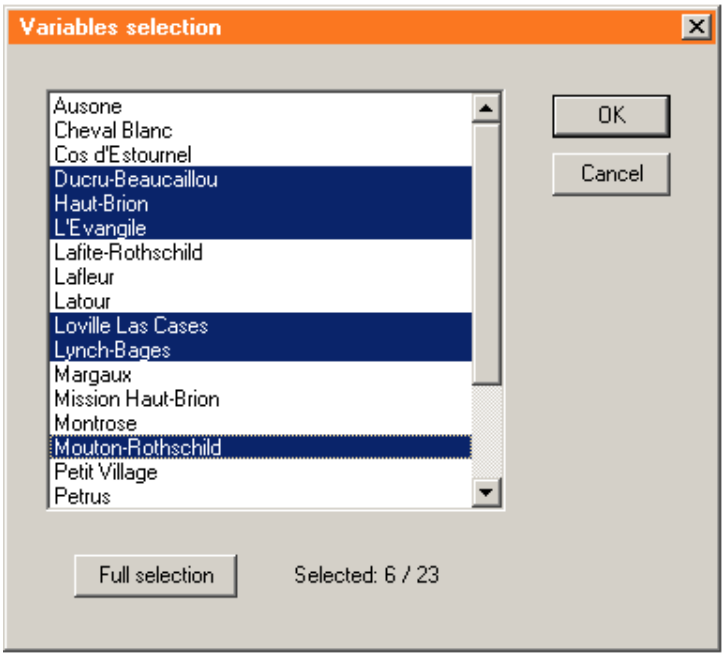
Frequencies for categorical multi-valued
Frequencies for interval

5.1 Capacities

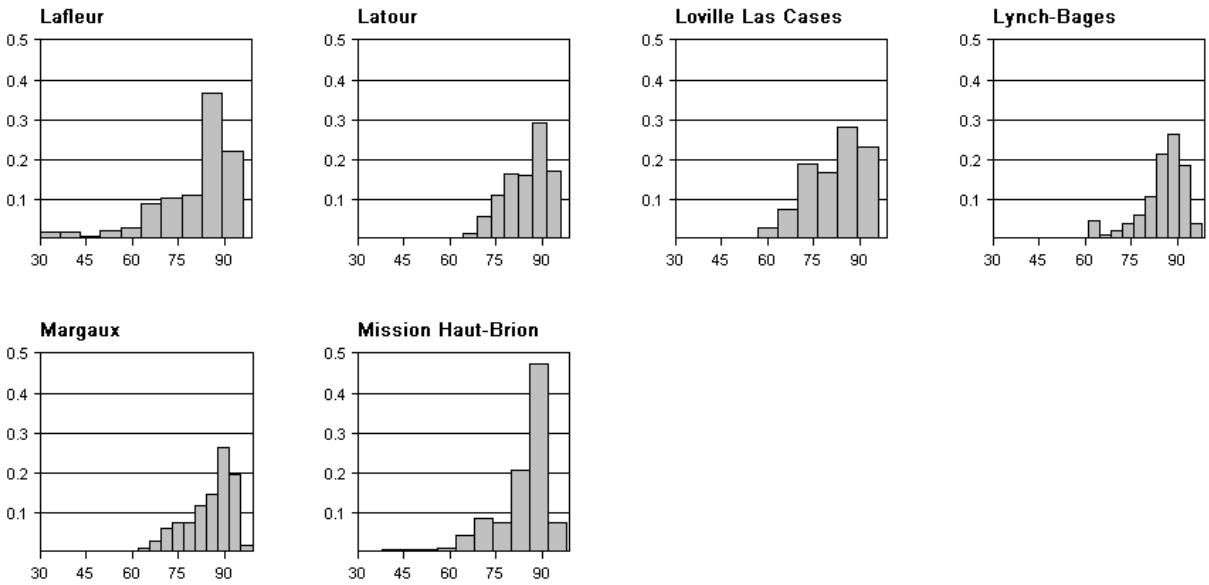
Basically, each of these methods apply to a single variable.
If a single variable has been selected in the workbench, the graph is directly displayed; example for the Frequencies for interval variables:



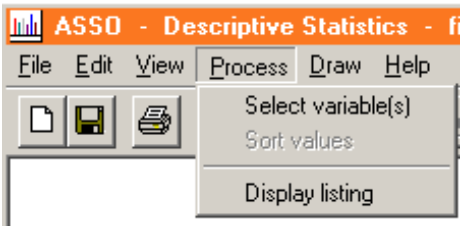
If several variables have been selected in the workbench, the method will be run against each of them successively, and the listing will contain all the results; but for the graph it is necessary to indicate which variable(s) to process, via the following dialog:



Selecting several variables allows to display simultaneouly several smaller graphs for easier comparison of variables; example for the Frequencies for interval variables :

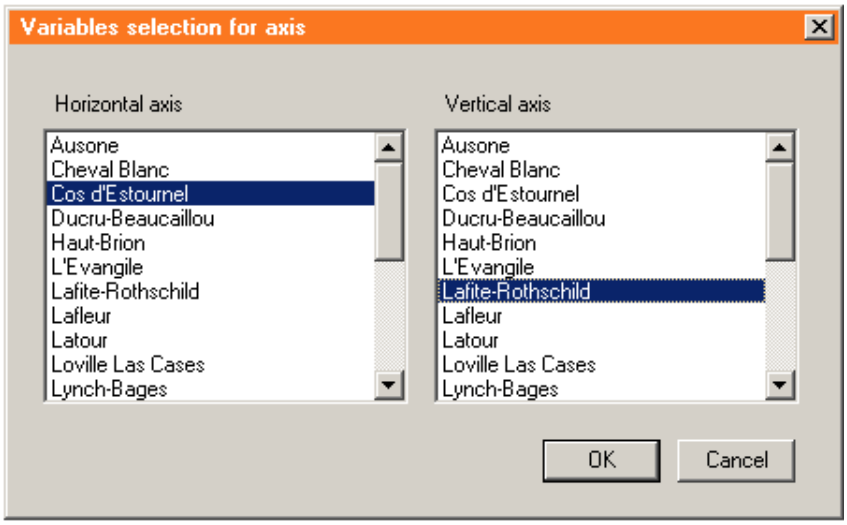


A new variable(s) selection can be done at any time with the dedicated menu command:



5.2 Biplot

The Biplot needs two and only two variables, that it takes as X and Y axis:



A new axis selection can be done at any time with the dedicated menu command.

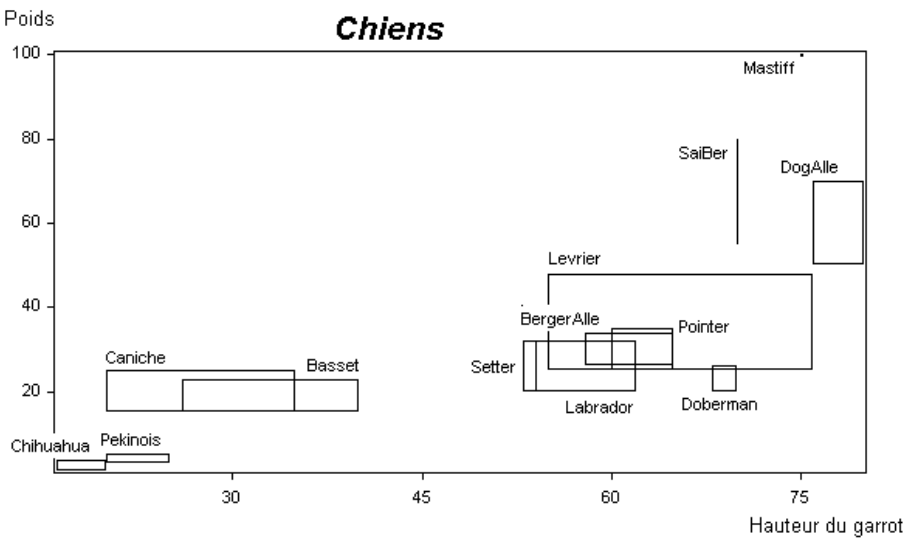
6 Customizing the graph

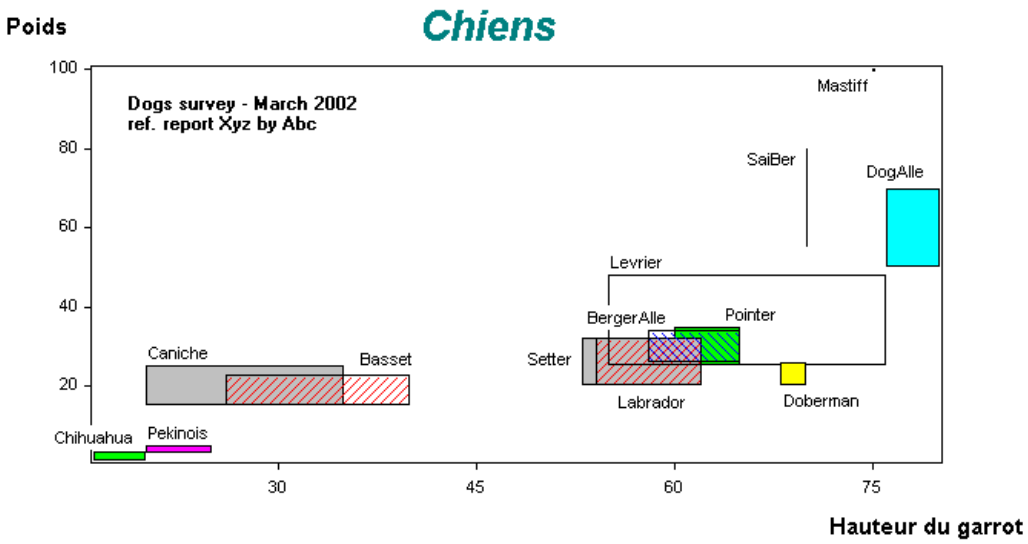
There are many features that allow the user to customize the graph.

They are detailed in the VSTAT and VPLOT internal help guides (VSTAT is the module handling the graphs for the first three methods, VPLOT for the Biplot).

Here are two examples for the Biplot:

- the first one is the graph with the default options, as it shows up the first time
- the second one is after «dressing»





7 Printing, copying and exporting the graph

The usual functions for printing and for copying to the clipboard are available.

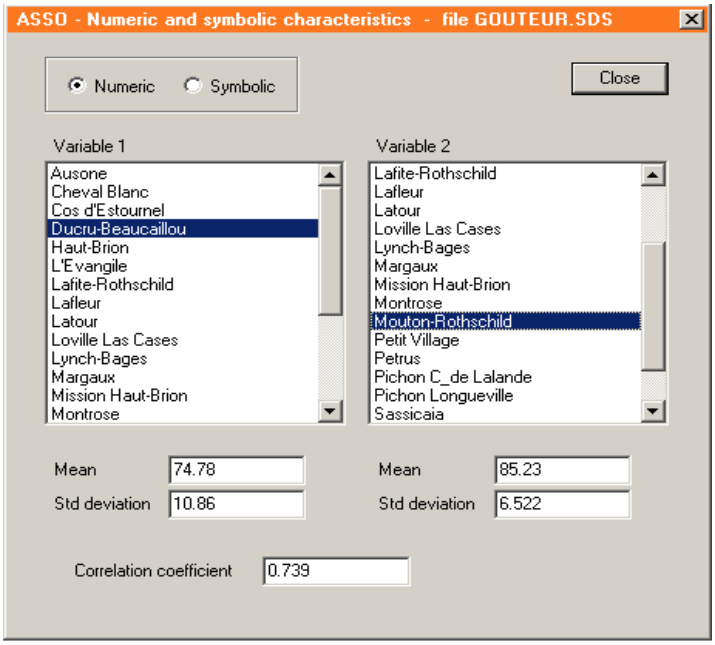
The graphs can also be exported for reuse in word and image processing applications, in three different formats: BMP, PNG and EMF (enhanced metafile).

8 Methods without graphic output

8.1 Numeric and symbolic characteristics

The means and standard deviations are computed for all the variables and for both the numeric and symbolic characteristics, and stored in the listing.

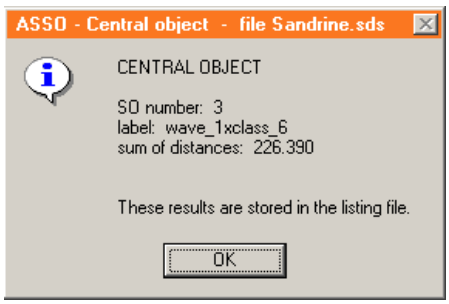
The following dialog allows to compute and display the correlations, and display the characteristics of the variables, at user will :



8.2 Central object

The central object is not concerned with variables.

It is directly computed and listed, and the following window shows up to both give the result and confirm the method completion:



INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

VSTAR User Manual

Viewer

Edited by FUNDP

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **16/12/2003**

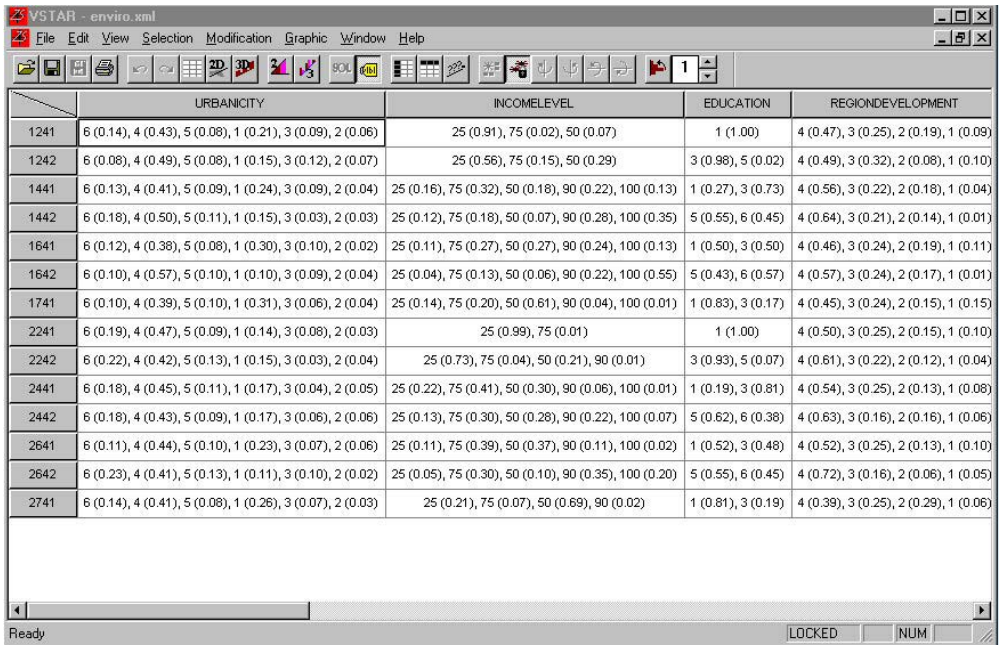
VSTAR : Viewer

1 Introduction

The VSTAR module allows you to display in a table all the Symbolic Objects that are present in a SODAS file (.sds or .xml) and to visualise them in 2D or 3D under the shape of stars. The following facilities are available in the program:

- Different stars representing Symbolic Objects can be superimposed.
- It is possible to build a new object simply by selecting a category in a star (see Breakdown option). You can of course visualise that object too.
- A help guide, which is also available online, is accessible through the Help menu. It describes the different screens of the module. We present here some examples of use. The options are so numerous that it is not possible to cover them all.

Once entering the VSTAR module through the VIEW method, the file of the base selected for chaining is displayed by default in a table (Fig. 1). All Symbolic Objects and variables appear, and among those, the ones selected inside the VIEW module appear in selected mode (i.e. dark grey in the table).



	URBANICITY	INCOMELEVEL	EDUCATION	REGIONDEVELOPMENT
1241	6 (0.14), 4 (0.43), 5 (0.08), 1 (0.21), 3 (0.09), 2 (0.06)	25 (0.91), 75 (0.02), 50 (0.07)	1 (1.00)	4 (0.47), 3 (0.25), 2 (0.19), 1 (0.09)
1242	6 (0.08), 4 (0.49), 5 (0.08), 1 (0.15), 3 (0.12), 2 (0.07)	25 (0.56), 75 (0.15), 50 (0.29)	3 (0.98), 5 (0.02)	4 (0.49), 3 (0.32), 2 (0.08), 1 (0.10)
1441	6 (0.13), 4 (0.41), 5 (0.09), 1 (0.24), 3 (0.09), 2 (0.04)	25 (0.16), 75 (0.32), 50 (0.18), 90 (0.22), 100 (0.13)	1 (0.27), 3 (0.73)	4 (0.56), 3 (0.22), 2 (0.18), 1 (0.04)
1442	6 (0.18), 4 (0.50), 5 (0.11), 1 (0.15), 3 (0.03), 2 (0.03)	25 (0.12), 75 (0.18), 50 (0.07), 90 (0.28), 100 (0.35)	5 (0.55), 6 (0.45)	4 (0.64), 3 (0.21), 2 (0.14), 1 (0.01)
1641	6 (0.12), 4 (0.38), 5 (0.08), 1 (0.30), 3 (0.10), 2 (0.02)	25 (0.11), 75 (0.27), 50 (0.27), 90 (0.24), 100 (0.13)	1 (0.50), 3 (0.50)	4 (0.46), 3 (0.24), 2 (0.19), 1 (0.11)
1642	6 (0.10), 4 (0.57), 5 (0.10), 1 (0.10), 3 (0.09), 2 (0.04)	25 (0.04), 75 (0.13), 50 (0.06), 90 (0.22), 100 (0.55)	5 (0.43), 6 (0.57)	4 (0.57), 3 (0.24), 2 (0.17), 1 (0.01)
1741	6 (0.10), 4 (0.39), 5 (0.10), 1 (0.31), 3 (0.06), 2 (0.04)	25 (0.14), 75 (0.20), 50 (0.61), 90 (0.04), 100 (0.01)	1 (0.83), 3 (0.17)	4 (0.45), 3 (0.24), 2 (0.15), 1 (0.15)
2241	6 (0.19), 4 (0.47), 5 (0.09), 1 (0.14), 3 (0.08), 2 (0.03)	25 (0.99), 75 (0.01)	1 (1.00)	4 (0.50), 3 (0.25), 2 (0.15), 1 (0.10)
2242	6 (0.22), 4 (0.42), 5 (0.13), 1 (0.15), 3 (0.03), 2 (0.04)	25 (0.73), 75 (0.04), 50 (0.21), 90 (0.01)	3 (0.93), 5 (0.07)	4 (0.61), 3 (0.22), 2 (0.12), 1 (0.04)
2441	6 (0.18), 4 (0.45), 5 (0.11), 1 (0.17), 3 (0.04), 2 (0.05)	25 (0.22), 75 (0.41), 50 (0.30), 90 (0.06), 100 (0.01)	1 (0.19), 3 (0.81)	4 (0.54), 3 (0.25), 2 (0.13), 1 (0.08)
2442	6 (0.18), 4 (0.43), 5 (0.09), 1 (0.17), 3 (0.06), 2 (0.06)	25 (0.13), 75 (0.30), 50 (0.28), 90 (0.22), 100 (0.07)	5 (0.62), 6 (0.38)	4 (0.63), 3 (0.16), 2 (0.16), 1 (0.06)
2641	6 (0.11), 4 (0.44), 5 (0.10), 1 (0.23), 3 (0.07), 2 (0.06)	25 (0.11), 75 (0.39), 50 (0.37), 90 (0.11), 100 (0.02)	1 (0.52), 3 (0.48)	4 (0.52), 3 (0.25), 2 (0.13), 1 (0.10)
2642	6 (0.23), 4 (0.41), 5 (0.13), 1 (0.11), 3 (0.10), 2 (0.02)	25 (0.05), 75 (0.30), 50 (0.10), 90 (0.35), 100 (0.20)	5 (0.55), 6 (0.45)	4 (0.72), 3 (0.16), 2 (0.06), 1 (0.05)
2741	6 (0.14), 4 (0.41), 5 (0.08), 1 (0.26), 3 (0.07), 2 (0.03)	25 (0.21), 75 (0.07), 50 (0.69), 90 (0.02)	1 (0.81), 3 (0.19)	4 (0.39), 3 (0.25), 2 (0.29), 1 (0.06)

Figure 1: Symbolic Objects in a table

The selection can be directly changed by clicking the columns or the Selection button in the menu bar. The process of selecting / deselecting has the same conventions as in the VIEW module. To deselect a preselection in the table, you can click on the upper left cell.

Once you have selected at least one Symbolic Object and three variables, you can visualise the corresponding objects in 2D or 3D. This can be done simply by clicking the icons or , or by selecting View in the menu bar and then “2D Graphic” or “3D Graphic”.

2 2D Graphic

If you have selected “2D Graphic”, all the preselected Symbolic Objects are displayed in the form of 2D Zoom Stars in separate windows. The windows appear in cascade mode, but you can switch to tile mode through clicking on the Window menu, and then on “Tile”.

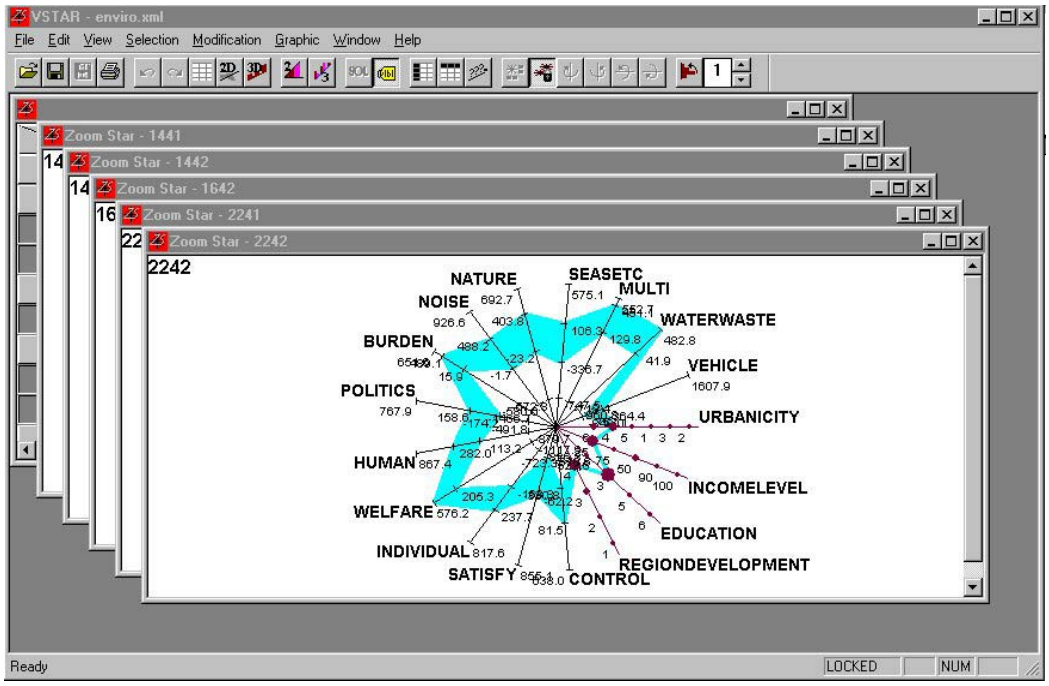


Figure 2: 2D Zoom stars in cascade mode

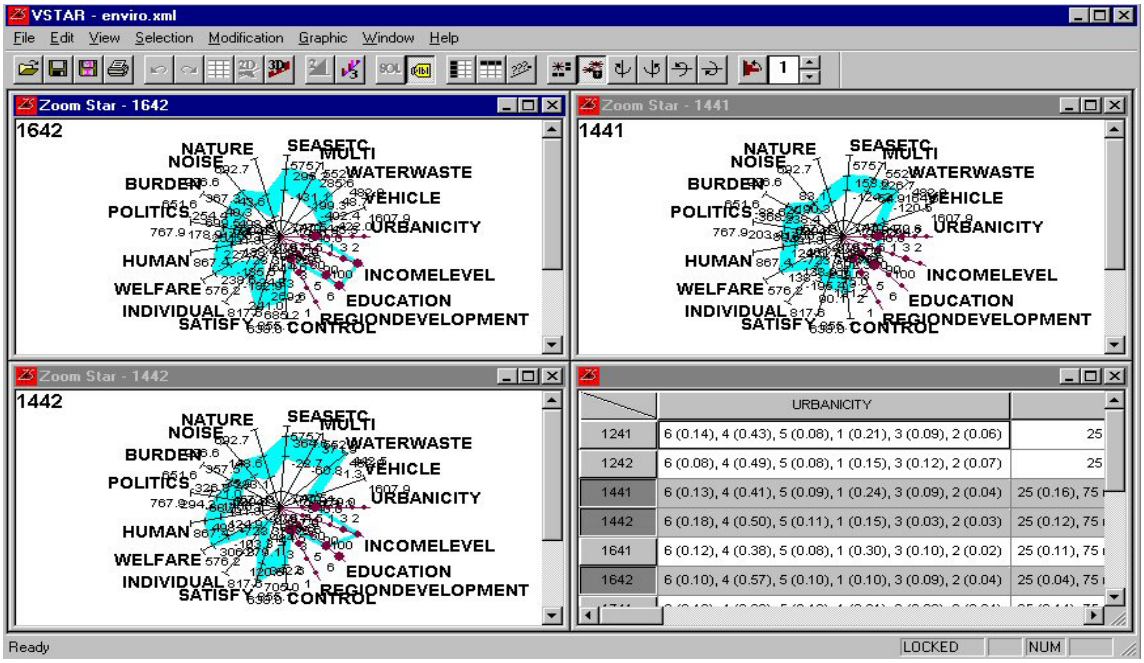
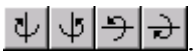


Figure 3: 2D Zoom stars in tile mode

If you want the same actions to be performed on all Zoom Stars, simply select “Lock” in the Graphic menu. The 2D Zoom Star representation has the following conventions:

Variable type	***Axis description***
Quantitative	Graduated axis
Categorical	Dots equally distributed on the axis
Categ.: Not Weighted	Axis drawn in one colour
Categ.: Weighted	Axis drawn in another colour
Missing value	Axis drawn in grey

You can turn and rotate the stars thanks to the following buttons:  . When the labels are too large, it is a good way to make them appear properly. There is also an option allowing you to reduce labels. Select the View menu and then “Labels”. Choose the length of the labels in the frame near the “Truncated at” instruction (see Fig. 4).

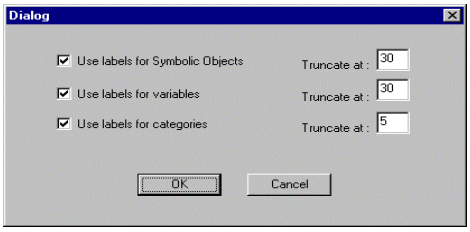


Figure 4: Truncational labels

If “Use Labels for Symbolic Objects / Variables / Categories” is not selected, the corresponding items’ codes will be displayed instead of the labels (see Fig. 5). This can also be changed using this icon: .

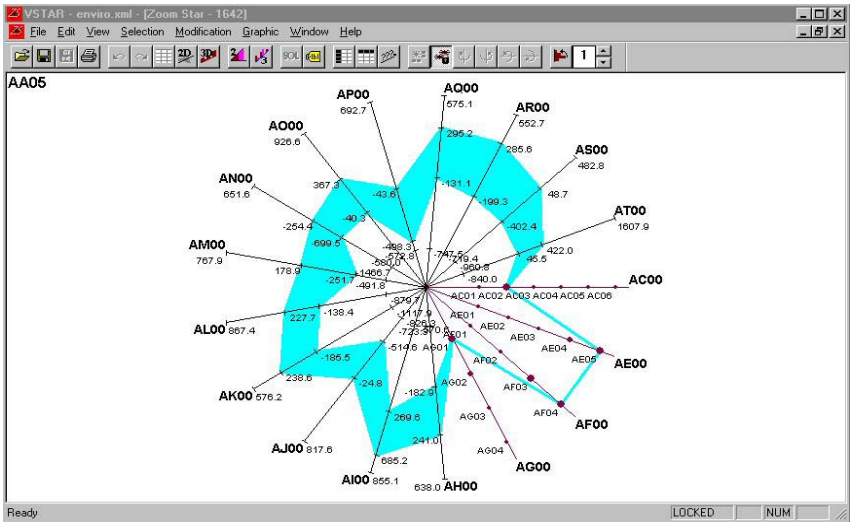


Figure 5: 2D Zoom Star with codes instead of labels

3 Interpreting the picture

For training, we advise you to start with only one selected Symbolic Object. Figure 2 represents a kind of **accident**. Large surfaces indicate that the dispersion of the variables is large with respect to the objects present in the whole file.

The scales are computed using the minimum among the min values and the maximum among the max values of the whole file. The size of the dots is proportional to the frequency of the categories. In order to have a more accurate figure of frequencies, click on a modal axis. A histogram is displayed in a separate window; it corresponds to the detailed distribution of the modal variable.

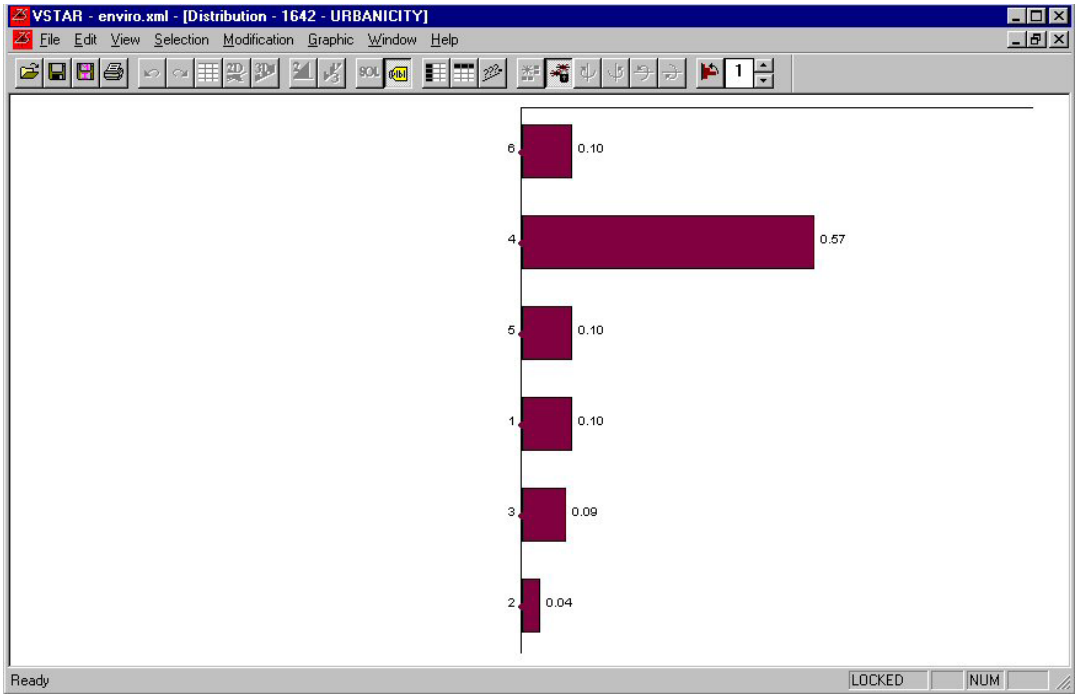


Figure 6: Histogram for the URBANICITY variable

If a hierarchy sign appears on the axes, it means that there is taxonomy for this variable. To see the details of the taxonomy (see Fig. 8), you can click the icon with the left button of the mouse. You can see the details of the taxonomy by selecting “Taxonomy” in the View menu.

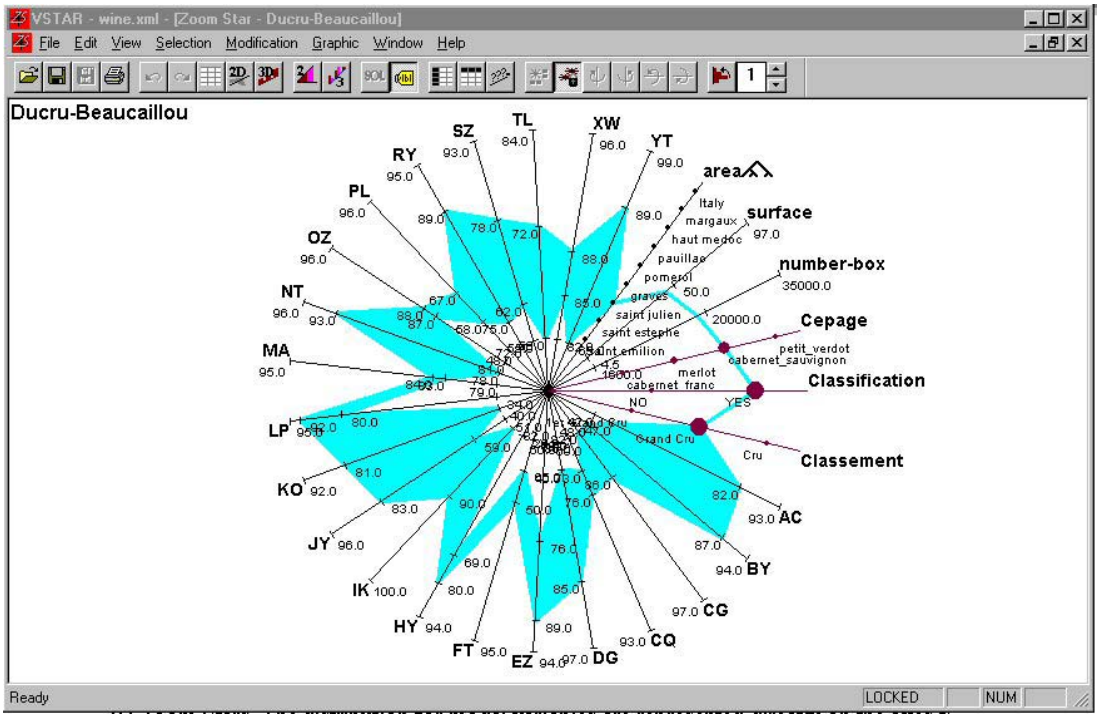


Figure 7: 2D Zoom Star with a taxonomical variable (AREA)

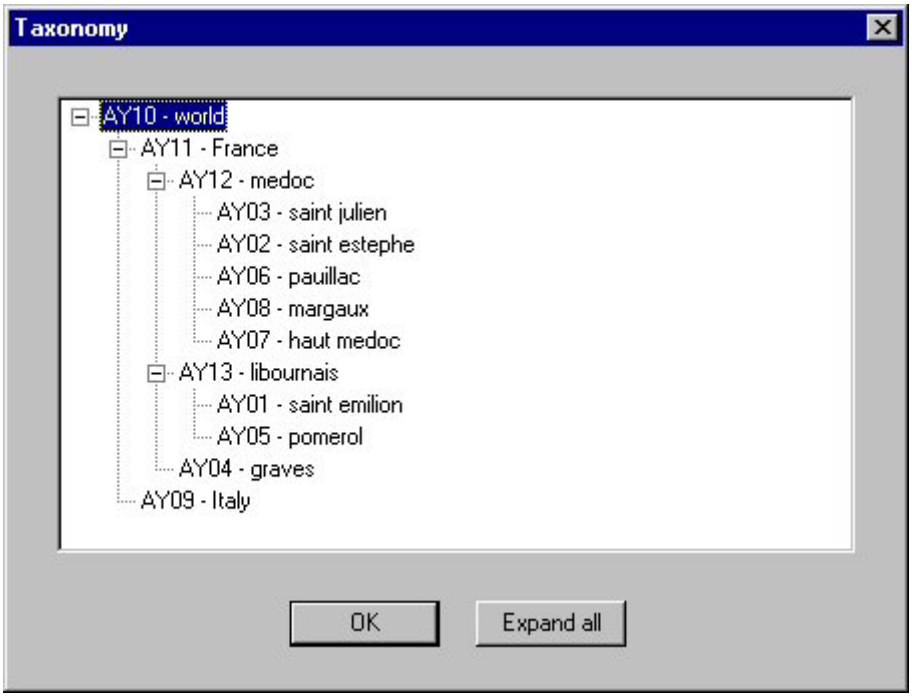


Figure 8: Taxonomy details

If a small line starting on axes is displayed, it indicates the existence of a **dependency**. If you click on the right button of the mouse, then the entire axes is drawn. You can see the details of dependencies through selecting “Dependencies” in the View menu.

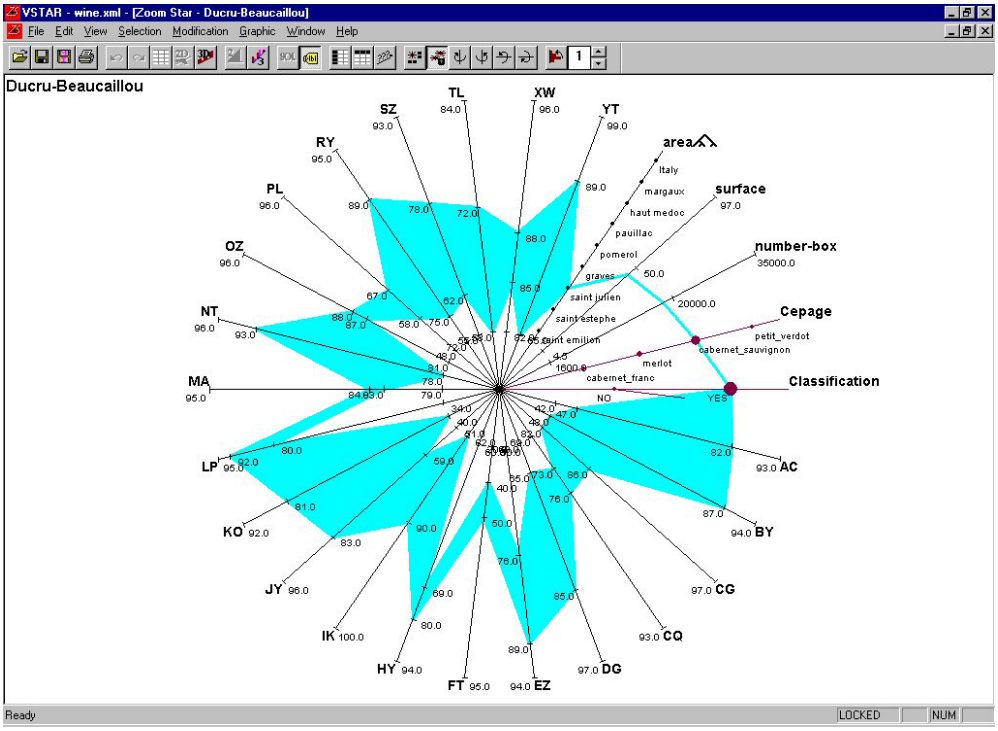


Figure 9: 2D Zoom star with dependency

4 3D Graphic

If you have selected the "3D Graphic" option, then the pre-selected Symbolic Objects will be displayed under the shape of 3D Zoom Stars. Then the distribution for modal variables is to be represented directly on the axes.

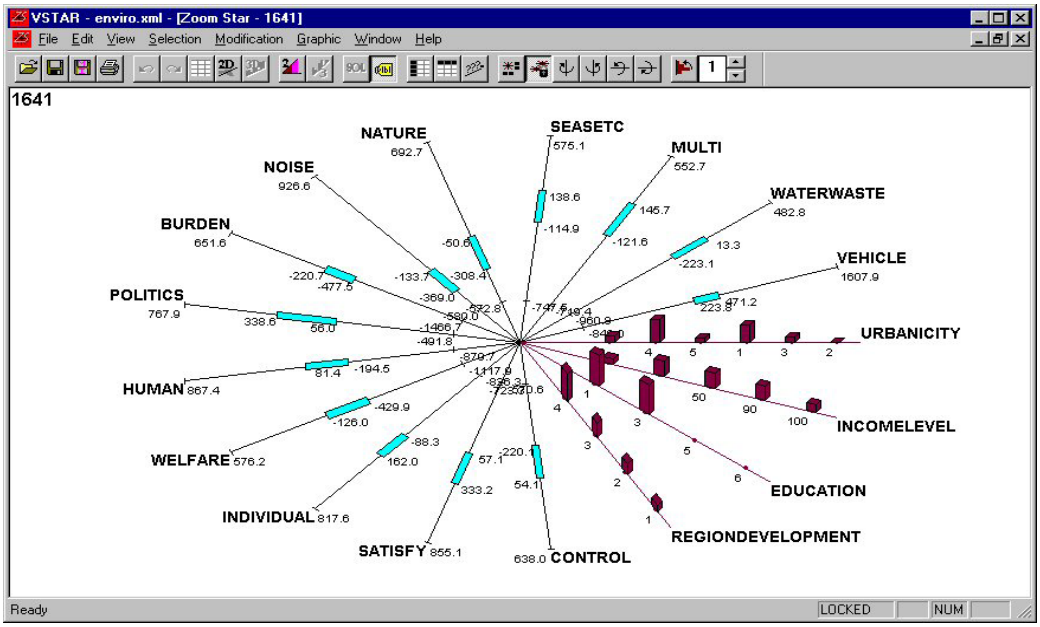


Figure 10: 3D Zoom star

The height of histograms can be modified through clicking on the small arrows located on the following icon in the tool bar: 1.

The star can be turned right or left and up or down, using the options in the Graphic menu or Add Buttons and clicking on the following icons: .

The other options and possibilities are the same as those existing for the 2D Zoom Star.

5 Superimposition

It is possible to superimpose Zoom Stars in 2D ( ) or in 3D ( ). First, select in the table the Symbolic Objects that you want to see represented. Then, choose "Superimpose 2D" or "Superimpose 3D". It is then possible to compare several Symbolic Objects by superimposition. In order to make this task easier, the different objects are given different default colours (See Fig. 11).

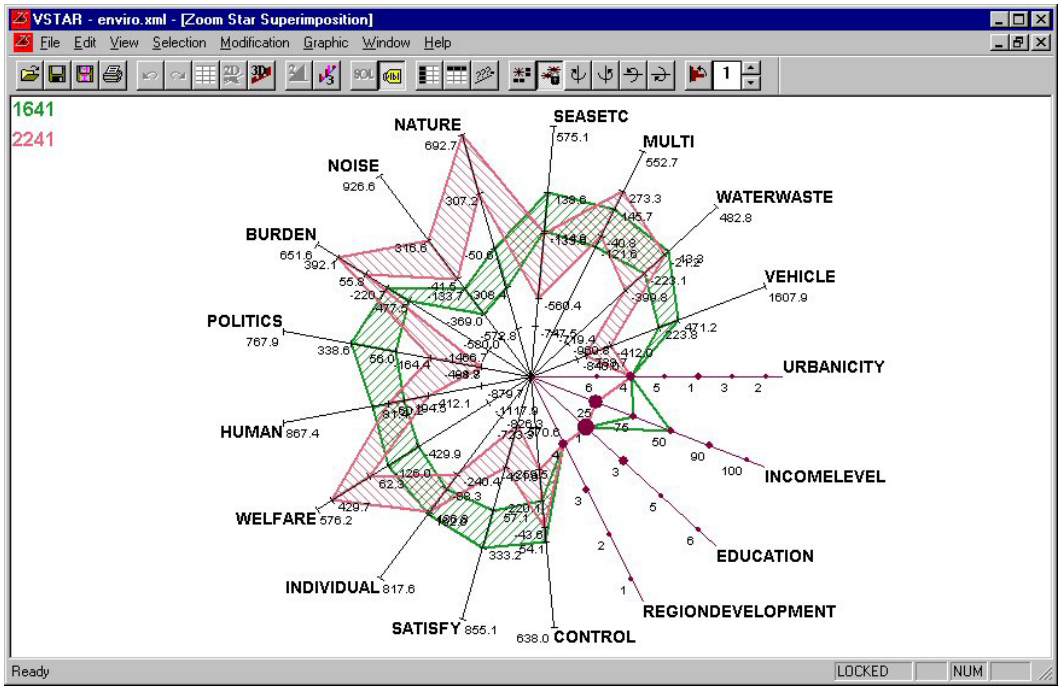


Figure 11: Superimposition of two Symbolic Objects

6 Graphical facilities (in the Graphic menu)

The program allows you to add text (for example a title). This is particularly useful when you want to edit the figure. You can also change the predefined colours ("Set Colour") or change the size of the fonts ("Set Fonts"). You can save the image through using "Save Image File" in the File menu.

7 Breakdown or Drill down

Breakdown or Drill down is a new interactive way to create a Symbolic Object inside a given class. When selecting a category on an axis of a zoom star, a search in the individuals database is generated and the corresponding object is automatically created. It represents the set of individuals who match the selected characteristic. For example if we import data on sociological investigation, we can interactively create the objects corresponding to gender (women or men), or the class of individuals corresponding to status (single person, widower, cohabitation, divorced). The two symbolic objects will be displayed on the same graph (with superimposition).

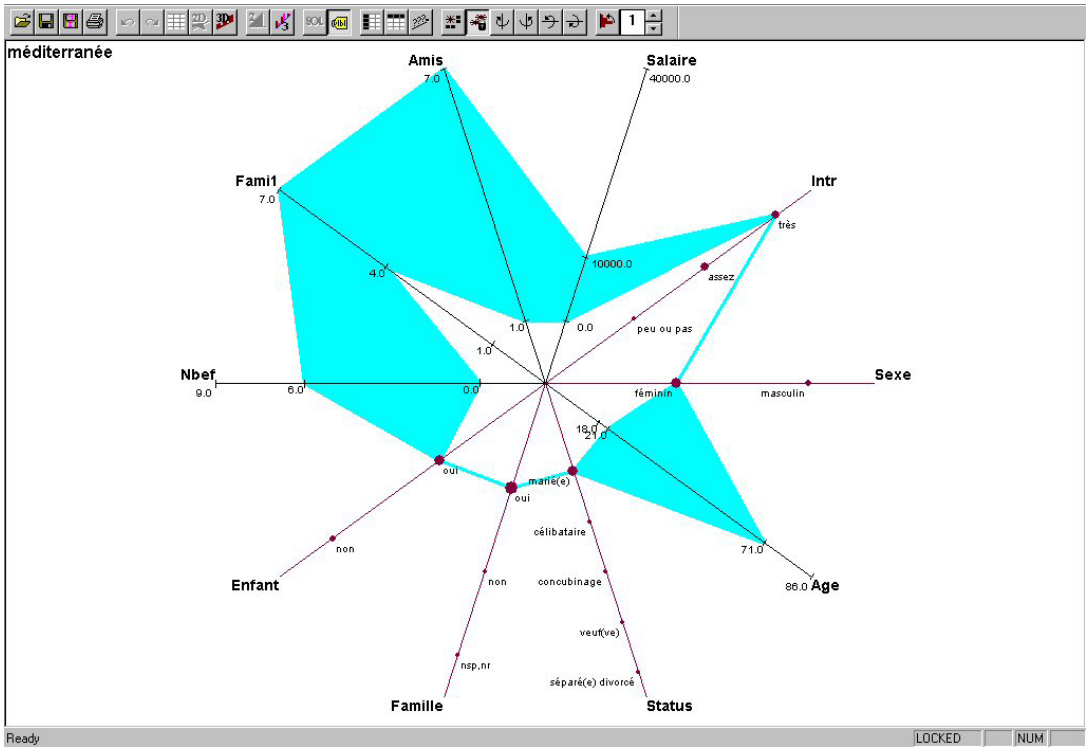


Figure 12: Original objects for breakdown

To perform a Breakdown, the following sequence of actions has to be followed:

1. Display the desired object in 2D/3D (see Fig. 12).
2. Right click twice on the desired category of modal variable (for example “Masculin” on the “Sexe” variable).
3. A window appears in which you are required to choose the original database (Fig. 13)
4. The new Symbolic Object is automatically created and represented by the superimposition process (Fig. 14).
5. It is added in the original file but it is saved with another name (“BkDown” + Original name).

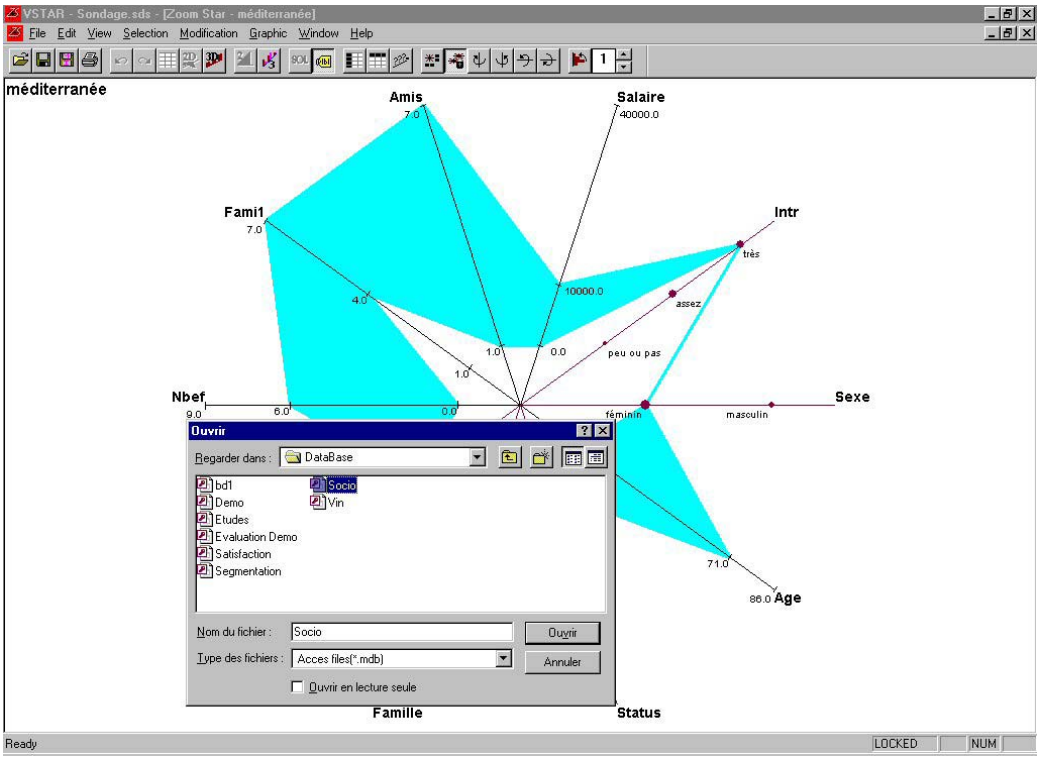


Figure 13: Breakdown - Choice of the original database

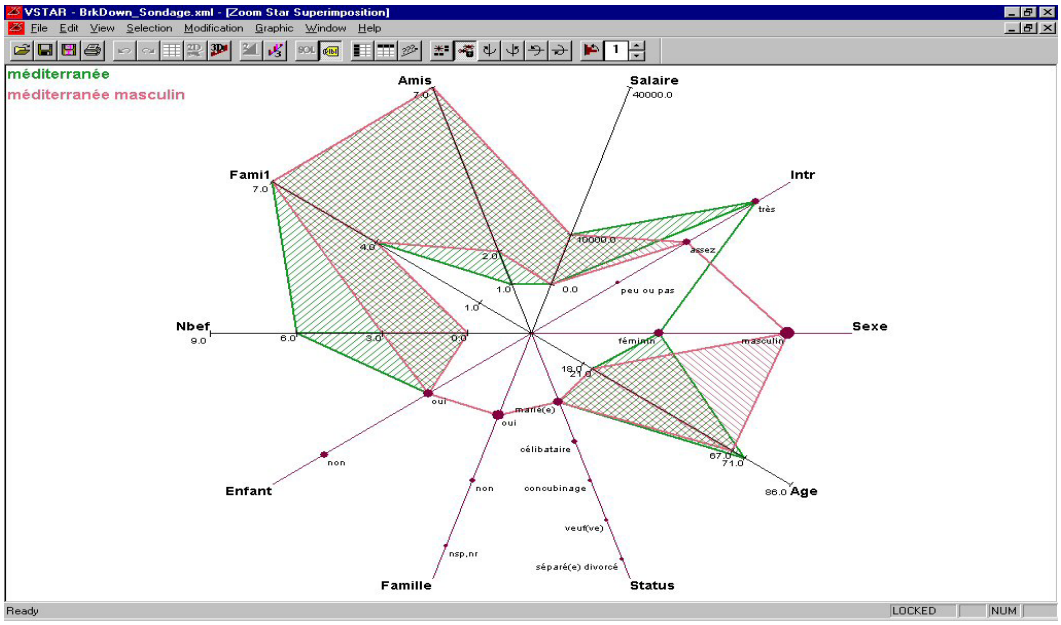


Figure 14: Breakdown - Superimposition of the new and the old objects

The last figure shows in green the initial object ("Méditerranée") and in pink the set of men inside the object "Méditerranée".

Dissimilarity and Matching

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

DISS-VDISS User Manual

Dissimilarity Measures

Edited by DIB

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **01/12/2003**

DISS-VDISS : Dissimilarity Measures

1 Introduction

Symbolic data analysis generalises some standard statistical data mining methods, such as those developed for classification and clustering tasks, to the case of symbolic objects (SOs). These objects, informally defined as "aggregated data", synthesize information concerning about a more or less homogeneous classes or groups of individuals by means of *set-valued* or *modal* variables. A variable Y defined for all elements k of a set E is termed *set-valued* with the domain Y if it takes its values in $\boldsymbol{P}(\mathrm{Y})=\{U \mid U \subseteq \mathrm{Y}\}$, that is the power set of Y. A *modal* variable is a set-valued variable with a measure or a (frequency, probability or weight) distribution associated to $Y(k)$. The description of a class or a group of individuals by means of either set-valued variables or modal variables is termed *symbolic object* (SO). More specifically a *boolean symbolic object* (BSO) is described by set-valued variables only, while a *probabilistic symbolic object* (PSO) is described by modal variables with a relative frequency distribution associated to each of them. A symbolic object could be also described by both set-valued variables and modal variables.

A set of symbolic data, which involves the same variables to describe different (possibly overlapping) classes of individuals, can be described by a single table, named symbolic data table, where rows correspond to distinct symbolic data while columns correspond descriptive variables. Symbolic data tables are more complex to be analyzed than standard data tables, since each item at the intersection of a row and a column can be either a finite set of values, or an interval or a probability distribution. The main goal of the research area known as symbolic data analysis is that of investigating new theoretically sound techniques to analyze such table.

Comparing SOs is an important step of symbolic data analysis. It can be useful either to cluster some SOs or to discriminate between them, or even to order SOs according to their degree of generalization. Dissimilarity operators have been defined both for BSOs and PSOs.

At present the following dissimilarity measures for BSOs have been implemented in the module DISS:

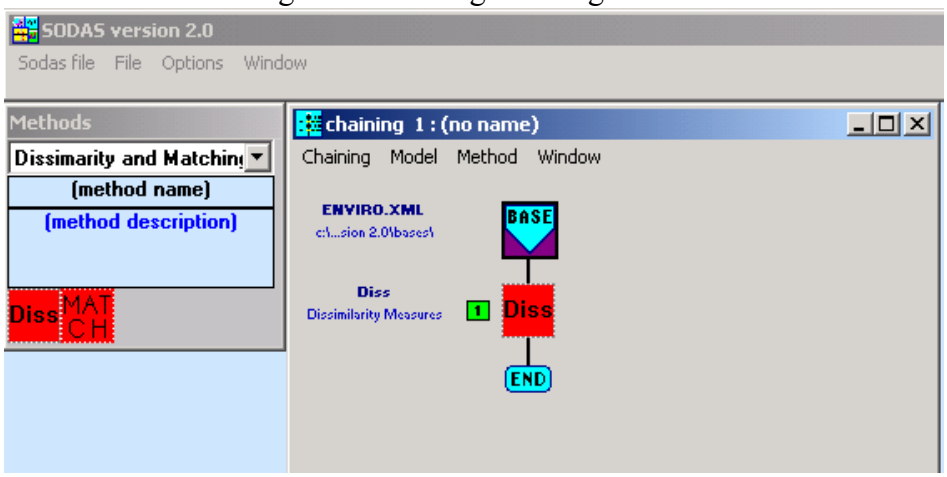
- U_1: Gowda and Diday's dissimilarity measure;
- U_2: Ichino and Yaguchi's first formulation of a dissimilarity measure;
- U_3: Ichino and Yaguchi's dissimilarity measure normalized w.r.t. domain length;
- U_4: Ichino and Yaguchi's normalized and weighted dissimilarity measure;
- SO_1: De Carvalho's dissimilarity measure;
- SO_2: De Carvalho's extension of Ichino and Yaguchi's dissimilarity;
- SO_3: De Carvalho's first dissimilarity measure based on description potential;
- SO_4: De Carvalho's second dissimilarity measure based on description potential;
- SO_5: De Carvalho's normalized dissimilarity measure based on description potential;
- SO_6: dissimilarity measure based on flexible matching among BSOs ;
- C_1: De Carvalho's normalized dissimilarity measure for constrained BSOs.

In the case of PSOs DISS implements a set of dissimilarity measures based on different measures of divergence between two discrete probability distributions, which are associated to each $Y(k)$ for some multi-valued variable Y.

2 The input to DISS

The module DISS aims at evaluating the degree of *dissimilarity* between a set of symbolic objects stored into the ASSO file associated to the running chain (see Figure 1).

Figure 1. Running chaining with DISS



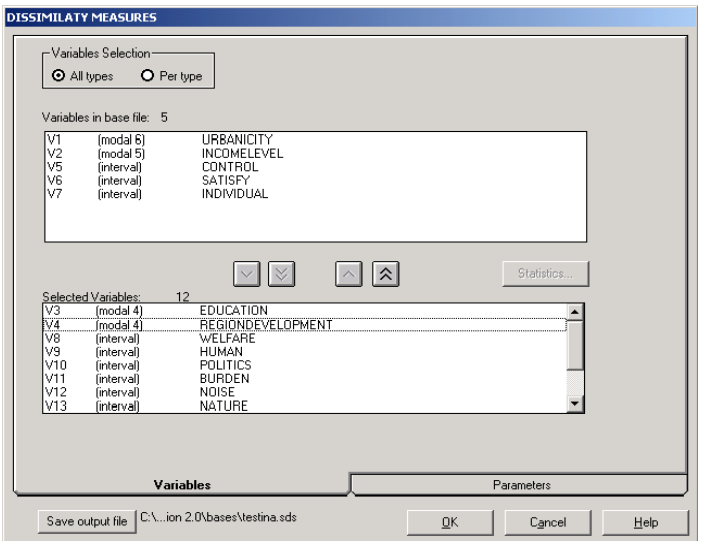
Generally a symbolic object is described by a collection of set-valued variables and/or modal variables, where a set-valued variable could be:

- a categorical single-value variable(i.e. *town(w)=London* where '*town*' is a variable describing the individual *w*),
- a categorical multi-value variable(i.e. *town(w)={London, Paris, Rome}*),
- a quantitative single-value variable(i.e. *height(w)=3.5*),
- an interval variable(i.e. *height(w)=[3,7]*);

while a modal variable describes a probability distribution (i.e. *town(w)={London(0.2), Paris(0.7), Rome(0.1)}*).

The user selects the variables to be analyzed (see Figure 2) in computing the dissimilarity between each pair of objects stored in input ASSO file.

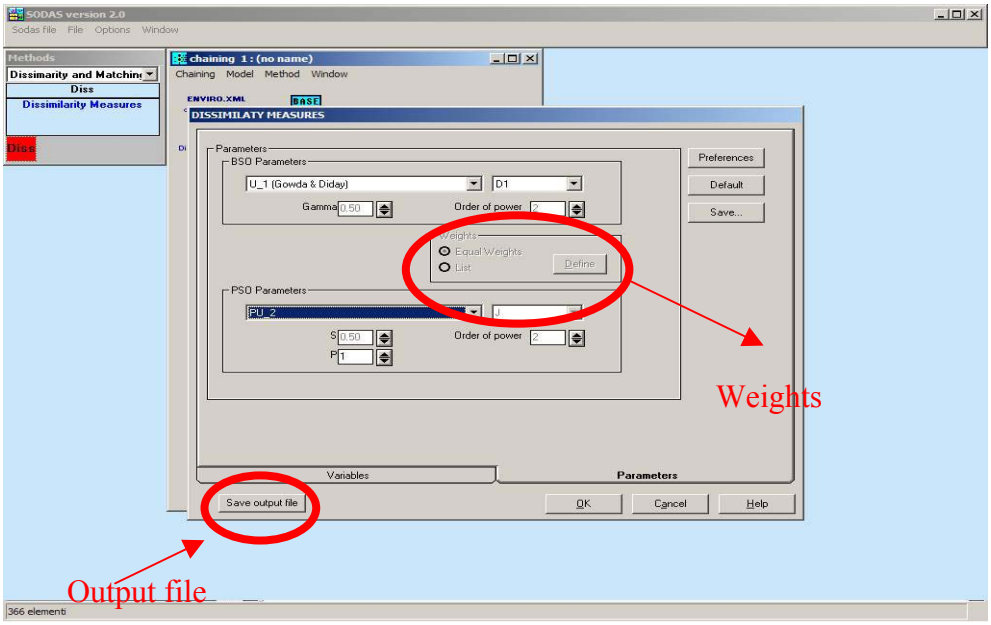
Figure 2. Selection of variables



According to these variables, the dissimilarity measures can be selected for both BSOs and PSOs (see Figure 3). In the case of mixed BSOs and PSOs the user has to select both a dissimilarity measure for BSOs and a dissimilarity measure for PSOs. The output dissimilarity matrix is then obtained by combining the two matrices computed using the two selected measures.

Dissimilarity measures could be computed by associating the same weight (1/Number of selected variables) to each selected variables (*Equal Weight* option). Alternatively the user could decide to associate a different weight to each selected variable (*List* option).

Figure 3: Dissimilarity functions selection



The user has also to specify the output file (NAME+PATH).

These choices are automatically stored in the parameter file(.pad) that manages the communication between the ASSO workbench and the DISS module.

This parameter file is composed by six sections. The first section contains:

- The ASSO file (PATH+NAME) that is the input of the entire running chaining (i.e. SDS_IN = "C:\ASSO\DISS\enviro.xml"),
- The ASSO file (PATH+NAME) that contains the output of DISS (i.e. SDS_OUT="C:\ASSO\DISS\output1.sds"),
- The log file (PATH+NAME) that contains the log of the DISS computation (i.e. LOG = "C:\ASSO\DISS\example1.LOG"),
- The output file (PATH+NAME) that reports the output dissimilarity matrix (i.e. OUT = "C:\ASSO\DISS\example1.DI0").

The second section contains the weights eventually associated to each selected variable. In the case that a list of weights is introduced: the sum of weights must be 1. The third section, also called BSO section, contains all parameters to compute the selected dissimilarity measure for BSOs. Similarly, the forth section, also called PSO section, contains all parameters to compute the selected dissimilarity measure for PSOs. The fifth section contains the list of the selected variables. Finally the sixth section contains the modality to combine (by sum or by product) the dissimilarity matrices obtained in the case that both modal and non-modal

variable have been selected in the fifth section. A complete example of a parameter file is shown in Figure 4.

Figure 4: An example of parameter file for DISS

```
SDS_IN = "C:\Documents and Settings\ASSO\Desktop\ASSOFILES\enviro.xml"
SDS_OUT="C:\Documents and Settings\ASSO\Desktop\ASSOFILES\output.sds"
LOG = "C:\Programmi\DECISIA\SODAS version 2.0\filieres\BQKB0Y01.LOG"
OUT = "C:\Programmi\DECISIA\SODAS version 2.0\filieres\BQKB0Y01.LST"

PROC_DISTANCE

::---Weights for selected variables
WSEL = LIST
WEIGHTS = 0.3,0.2,0.2,0.1,0.2
::---List of parameters for BSO
SIMFUN  = U_1
::---List of parameters for PSO
SIMFUN  = PU_2
P = 1
::---Selected variables
```

The parameter file together with the running chaining ASSO file are input to the module DISS. Furthermore, DISS automatically read the metadata stored in the metadata file named *meta<name of the input asso file>.xml* that is stored in the same path of the input ASSO file. In the case that metadata file is not available in the required path DISS shows a message and it continues the running.

3 The output of DISS

The output of DISS is a triangle matrix because of the symmetric property of dissimilarity measures (i.e given two SOs *a* and b, $d(a,b)=d(b,a)$). It is written in a new ASSO file which is a copy of the input file, containing the symbolic data array, plus the appended dissimilarity matrix. Other outputs of DISS module are a text file, containing only the dissimilarity matrix, and a report file visualizing all the information about the input, the selected parameters and the dissimilarity values among groups of SOs. DISS module also re-writes the metadata file associated to the selected input file. This file contains the metadata about the input SOs updated with the additional information on the dissimilarity measures for BSO and/or PSO used in the computation of the triangle matrix and the selected symbolic variables.

4 DISS Dissimilarity matrix: a visualization

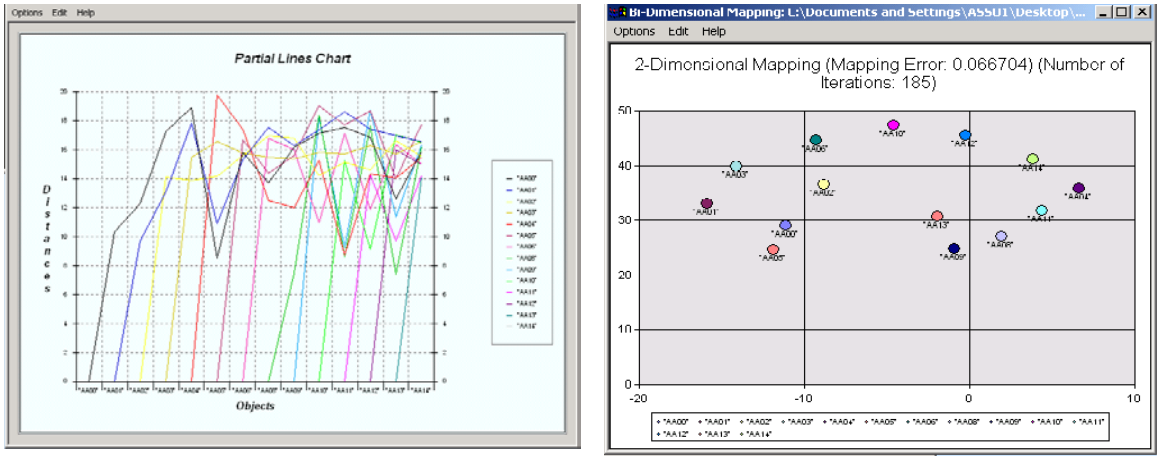
The output dissimilarity matrix computed by DISS could be visualized by means of the visualization module VDISS which also show some information stored in the metadata file associated to the current running chain. VDISS shows the matrix in a *table format*, in a bidimensional space *scatterplot* as well as in *line graph*.

The bidimensional scatterplot (see Figure 5.a) allows the visualization of the nonlinear mapping of the SOs into a 2-dimensional space. The nonlinear mapping is based on

Sammon's algorithm that takes in input a matrix of distances between all the SOs and returns a collection of points in the bidimensional space such that their Euclidean distances preserve the "structure" of the original dissimilarity matrix.

In the line graph (see Figure 5.b), dissimilarities are reported along vertical axis, in the same order in which they are reported in the dissimilarity matrix. Each line is used to represent the dissimilarity between a given SO and the other SOs represented in the matrix. The number of lines in each graph is equal to the number of SOs minus one.

Figure 5: VDISS visualization of a dissimilarity matrix



5 DISS method

The module DISS aims at evaluating the degree of dissimilarity between either SOs BSOs or PSOs. In the following, dissimilarity measures are detailed.

5.1 Dissimilarity measures for BSOs

Let a and b be two BSOs:

$$a = [Y_1=A_1] \wedge [Y_2=A_2] \wedge \ldots \wedge [Y_p=A_p]$$

$$b = [Y_1=B_1] \wedge [Y_2=B_2] \wedge \ldots \wedge [Y_p=B_p]$$

defined by p variables. Each variables Y_j takes values in a domain Y_j and A_j and B_j are subsets of Y_j. It is possible to define a dissimilarity measure between two BSOs a and b by aggregating dissimilarity values computed independently at the level of single variables Y_j (*componentwise dissimilarities*). A classical aggregation function is the generalized Minkowski metric. However, another class of measures defined for BSOs is based on the the notion of *description potential,* $\pi(a)$, which is defined as the *volume* of the Cartesian product $A_1 \times A_2 \times \ldots \times A_p$. For this class of measures, no componentwise decomposition is necessary, so that no function is required to aggregate dissimilarities computed independently for each variable.

The dissimilarity measures actually defined for BSOs (see Table 1) are

U_1: Gowda and Diday's dissimilarity measure;

U_2: Ichino and Yaguchi's first formulation of a dissimilarity measure;

U_3: Ichino and Yaguchi's dissimilarity measure normalized w.r.t. domain length;

U_4: Ichino and Yaguchi's normalized and weighted dissimilarity measure;

SO_1: De Carvalho's dissimilarity measure;

SO_2: De Carvalho's extension of Ichino and Yaguchi's dissimilarity;
SO_3: De Carvalho's first dissimilarity measure based on description potential;
SO_4: De Carvalho's second dissimilarity measure based on description potential;
SO_5: De Carvalho's normalized dissimilarity measure based on description potential;
SO_6: dissimilarity measure based on flexible matching among BSOs (section 4.5);
C_1: De Carvalho's normalized dissimilarity measure for constrained BSOs.

Table 1. Dissimilarity measures for BSO's.

Name	***Componentwise dissimilarity measure***	***Objectwise dissimilarity measure***
U_1	$D^{(j)}(A_j, B_j) = D_\pi(A_j, B_j) + D_s(A_j, B_j) + D_c(A_j, B_j)$ where $D_\pi(A_j, B_j)$ *is* due to *position* $D_s(A_j, B_j)$ is due to *spanning* $D_c(A_j,B_j)$ is due to *content*	$d(a, b) = \sum_{j=1}^{p} D^{(j)}(A_j, B_j)$
U_2	$\phi(A_j, B_j) = \lvert A_j \oplus B_j \rvert - \lvert A_j \otimes B_j \rvert + \gamma\,(2\lvert A_j \otimes B_j \rvert - \lvert A_j \rvert - \lvert B_j \rvert)$ where *meet* ($\otimes$) and *join* ($\oplus$) are two Cartesian operators.	$d_q(a, b) = \sqrt[q]{\sum_{j=1}^{p} [\phi(A_j, B_j)]^q}$
U_3	$\psi(A_j, B_j) = \frac{\phi(A_j, B_j)}{\lvert \mathcal{Y}_j \rvert}$	$d_q(a, b) = \sqrt[q]{\sum_{j=1}^{p} [\psi(A_j, B_j)]^q}$
U_4	$\psi(A_j, B_j) = \frac{\phi(A_j, B_j)}{\lvert \mathcal{Y}_j \rvert}$	$d_q(a, b) = \sqrt[q]{\sum_{j=1}^{p} w_j [\psi(A_j, B_j)]^q}$
SO_1	$d_i(A_j,B_j)$ $i=1, \ldots,5$	$d_q^i(a, b) = \sqrt[q]{\sum_{j=1}^{p} [w_j\, d_i(A_j, B_j)]^q}$
SO_2	$\psi'(A_j, B_j) = \frac{\phi(A_j, B_j)}{\mu(A_j \oplus B_j)}$	$d'_q(a, b) = \sqrt[q]{\sum_{j=1}^{p} \frac{1}{p} [\psi'(A_j, B_j)]^q}$
SO_3	None	$d'_1(a, b) = \pi(a \oplus b) - \pi(a \otimes b) + \gamma\,(2\pi(a \otimes b) - \pi(a) - \pi(b))$
SO_4	none	$d'_2(a,b) = \frac{\pi(a \oplus b) - \pi(a \otimes b) + \gamma(2\pi(a \otimes b) - \pi(a) - \pi(b))}{\pi(a^E)}$ where $\pi(a^E) = [Y_1 \in Y_1] \wedge \ldots \wedge [Y_p \in Y_p]$
SO_5	none	$d'_3(a,b) = \frac{\pi(a \oplus b) - \pi(a \otimes b) + \gamma(2\pi(a \otimes b) - \pi(a) - \pi(b))}{\pi(a \oplus b)}$
SO_6	none	$d(a,b)=[1-((fm(a,b)+fm(b,a))/2)]$

C_1	$d_i(A_j,B_j)$ $i=1, ...,5$	$d_q^i(a, b) = \sqrt[q]{\frac{\sum_{j=1}^{p}\left[w_j\, d_i\left(A_j, B_j\right)\right]^q}{\sum_{J=1}^{p}\delta(j)}}$, where δ(j) is the indicator function

U_1 and U_2 are summative dissimilarity measures. They can be applied to BSOs described by interval variables, by quantitative single valued variables (as long as domains are restricted by a lower and a upper bound for U_1) and by categorical single and multi valued variables. They can not be used in presence of constrains (hierarchical dependencies and logical dependencies) in the BSOs. U_1 values are normalized between *0* and *3p* where *p* is the number of the symbolic variables describing a BSO, while U_2 values are greater than or equal to *0*. U_3 and U_4 are quite similar to U_2 perhaps the former with the values normalized in the interval *[0,p]* and the latter in the interval *[0,1]*. SO_1 and SO_2 are similar to U_4 and their range is *[0,1]* too. SO_3 values are greater than or equal to *0*, SO_4 and SO_5 ranges in *[0,1]*. C_1 is the dissimilarity measures used only for constrained BSOs with logical or hierarchical dependences.

The dissimilarity measures for BSOs implemented in the module DISS and the corresponding parameters are shown in table 2.

Table 2. Dissimilarity measures available in the DISS module for BSO's, and related

Dissimilarity measure	***Parameters***	***Constraints***	***Default***
U_1 (Gowda & Diday)	none		
U_2 (Ichino & Yaguchi)	Gamma Order of power	[0 .. 0.5] 1 .. 10	0.5 2
U_3 (Normalized Ichino & Yaguchi)	Gamma Order of power	[0 .. 0.5] 1 .. 10	0.5 2
U_4 (Weighted Normalized Ichino & Yaguchi)	Gamma Order of power List of weights per variable	[0 .. 0.5] 1 .. 10 Sum(weights) = 1.0	0.5 2 Equal weights
C_1 (Normalized De Carvalho)	Comparison function Order of power	D_1, D_2, D_3, D_4, D_5 1 .. 10	D_1 2
SO_1 (De Carvalho)	Comparison function Order of power List of weights per variable	D_1, D_2, D_3, D_4, D_5 1 .. 10 Sum(weights) = 1.0	D_1 2 Equal weights
SO_2 (De Carvalho)	Gamma Order of power	[0 .. 0.5] 1 .. 10	0.5 2
SO_3 (De Carvalho)	Gamma	[0 .. 0.5]	0.5

SO_4 (Normalized De Carvalho)	Gamma	[0 .. 0.5]	0.5
SO_5 (Normalized De Carvalho)	Gamma	[0 .. 0.5]	0.5
SO_6 ()	None		

5.2 Dissimilarity measures for PSOs

Let *a* and *b* be two PSOs:

$$a = [Y_1 \in A_1] \wedge [Y_2 \in A_2] \wedge \ldots \wedge [Y_n \in A_n] \text{ and } b = [Y_1 \in B_1] \wedge [Y_2 \in B_2] \wedge \ldots \wedge [Y_n \in B_n]$$

where each variable Y_j is modal and takes values in the domain Y_j and A_j , B_j are subsets of Y_j. A dissimilarity function between *a* and *b* can be built by aggregating dissimilarity coefficients between probability distributions through the generalized and weighted Minkowski's metric (see *PU_1*in table 3):

$$d_p(a,b) = \sqrt[p]{\sum_{k=1}^{n} \left[c_k \; m(A_k, B_k)\right]^p}$$

where, $\forall k \in \{1, \ldots, n\}$, $c_k>0$ are weights with $\Sigma_{k=1..n} c_k=1$ and, by setting $P=A_k$ and $Q=B_k$, $m(P,Q)$ is a dissimilarity coefficient between probability distributions that is one of the following comparison functions:

- J: *J-coefficient*
- CHI2 : χ^2 – *divergence*
- CHER: *Chernoff's distance*
- REN: *Rènyi's distance*
- LP: *Minkowski's L_p distance*

Alternatively, the dissimilarity coefficients can be aggregated through the product. Therefore, by adopting appropriate precautions and considering only the Minkowski's L_p distance (see *PU_2* in table 3), we obtain the following normalized dissimilarity measure between PSOs:

$$d'_p(a,b) = 1 - \frac{\prod_{i=1}^{n}\left(\sqrt[p]{2} - \sqrt[p]{\sum_{y_i}\left|p(y_i) - q(y_i)\right|^p}\right)}{\left(\sqrt[p]{2}\right)^n} = 1 - \frac{\prod_{i=1}^{n}\left(\sqrt[p]{2} - \sqrt[p]{L_p}\right)}{\left(\sqrt[p]{2}\right)^n}$$

where each y_i corresponds to a value of the *i*-th variable domain.

Note that this dissimilarity measure is symmetric and normalized in [0,1]. Obviously $d'_p(a,b) = 0$ if *a* and *b* are identical and $d'_p(a,b) = 1$ if the two objects are completely different.

Finally the dissimilarity measure between PSOs could be computed by comparing them using a matching function. A matching operator evaluates the degree of match of a SO against another SO. Formally the main difference between dissimilarity measures and matching functions is that the latter does not satisfy the symmetry property. So, similarity judgement in the matching process is directional: there is a referent and a subject. The referent is either a prototype or a description of a class, while the subject is either a variant of the prototype or a description of an instance (individual) of a class. Therefore it is possible to compute the dissimilarity (see *PU_3* in table 3) between two PSOs *a* and *b* by considering both the matching between *a* and *b,* and the matching between *b* and *a* as follows:

$$d(a,b) = [1-((fm\ (a,b)+fm(b,a))/2)]$$

where *fm* stands for *flexible matching* measure.

Table 3. Dissimilarity measures available for PSO's.

Name	Componentwise dissimilarity measure		Objectwise dissimilarity measure
PU_1	Indicated $P = A_k$ and $Q = B_k$, $d(A_k, B_k)$ is given by:		$d_q(a,b) = \sqrt[q]{\sum_{k=1}^{n}\left[c_k\ d^*(A_k, B_k)\right]^q}$
	J-coefficient	$J(P,Q) := I(Q \mid P) + I(P \mid Q)$ where $I(Q \mid P) = \sum_{y\in Y} \log\left\{\frac{q(y)}{p(y)}\right\} \cdot q(y)$	
	Symmetrized χ^2- *divergence*	$d^*(P,Q) := \left(\sum_{y\in Y} \frac{\lvert p(y) - q(y)\rvert^2}{p(y)} + \sum_{y\in Y} \frac{\lvert q(y) - p(y)\rvert^2}{q(y)}\right)$	
	Symmetrized Chernoff distance	$d^*(P,Q) := -\log d^{(s)}(P,Q) - \log d^{(s)}(Q,P)$ where $0 < s < 1$ e $d^{(s)}(P,Q) := \sum_{y\in Y} q^s(y) \cdot p^{1-s}(y)$.	
	Symmetrized Rènyi distance	$d^*(P,Q) := \frac{\log d^{(s)}(P,Q)}{s-1} + \frac{\log d^{(s)}(Q,P)}{s-1}$	
	Minkowski Lp distance	$d_1(P,Q) := \sum_{y\in Y} \lvert p(y) - q(y)\rvert^p$. dove $p > 0$.	
PU_2	LP = $\sum_{y\in Y} \lvert p(y) - q(y)\rvert^p$ is the Minkowski *Lp*		$d(a,b) = 1 - \frac{\prod_{i=1}^{n}\left(\sqrt[p]{2} - \sqrt[p]{\mathrm{LP}}\right)}{\left(\sqrt[p]{2}\right)^n}$

	distance.	
PU_3	none	$d(a,b) = [1-((fm(a,b)+fm(b,a))/2)]$

The dissimilarity measures for PSOs implemented in the DISS module and the corresponding parameters are shown in table 4.

Table 4. Dissimilarity measures available in the module DISS for PSO's, and related parameters.

Dissimilarity measure	***Parameters***	***Constraints***	***Default***
PU_1	Compfun Order of power S P List of weights per variable	J,CHI2,CHER,REN,LP 1..10 (0,1) 1..10 Sum(weights) = 1.0	J 2 0.5 1 Equal weights
In particular :			
PU_1 (J)	Order of power List of weights per variable	1..10 Sum(weights) = 1.0	2 Equal weights
PU_1(CHI2)	Order of power List of weights per variable	1..10 Sum(weights) = 1.0	2 Equal weights
PU_1(REN)	S Order of power List of weights per variable	(0,1) 1..10 Sum(weights) = 1.0	0.5 2 Equal weights
PU_1(CHER)	S Order of power List of weights per variable	(0,1) 1..10 Sum(weights) = 1.0	0.5 2 Equal weights

PU_1(LP)	P Order of power List of weights per variable	1..10 1..10 Sum(weights) = 1.0	1 2 Equal weights
PU_2	P	1..10	2
PU_3	none		

6 Examples

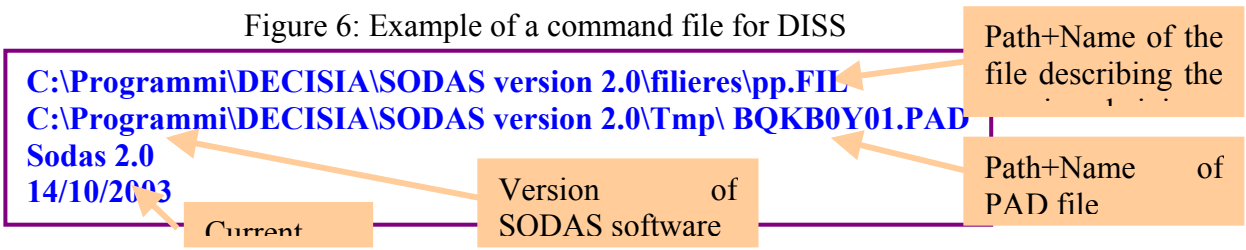
6.1 DISS Stand-alone version

DISS module stand-alone version running command: **DISS BQKB0Y01.cmd**

Pre-conditions →

Diss module stand-alone version works with the following files:

Figure 6: Example of a command file for DISS



1. **BQKB0Y01.cmd** -> it contains the path of file parameter file (see Figure 6).

2. **BQKB0Y01.PAD** -> it contains all the information needed to execute directly Diss module (see Figure 7).

Figure 7. Example of a parameter file for DISS

```
SDS_IN = " C:\Documents and Settings\Administrator\Desktop\AssoFiles \enviro.xml"
1.1        SDS_OUT=" C:\Documents and Settings\Administrator\Desktop\AssoFiles \output.sds"
LOG = "C:\Programmi\DECISIA\SODAS version 2.0\filieres\BQKB0Y01.LOG"
OUT = "C:\Programmi\DECISIA\SODAS version 2.0\filieres\BQKB0Y01.LST"
PROC_DISTANCE
::---Weights for selected variables
WSEL = LIST
WEIGHTS = 0.3,0.2,0.2,0.1,0.2
::---List of parameters for BSO
SIMFUN  = U_1
::---List of parameters for PSO
SIMFUN  = PU_2
P = 1
```

3. **DissDLL.dll** -> it contains all the methods used to compute the dissimilarities

It is important that Diss.exe and DissDLL.dll are in the same directory!

Diss module execution →

Open MS-DOS Window
Work directory position
Digit **diss.exe BQKB0Y01.cmd** as in the example in figure 8.

Figure 8. Example of running of DISS

```
C:\WINNT\System32\cmd.exe

C:\Documents and Settings\Administrator\Desktop\asso\DISS02-10>Diss.exe BQKB0Y01
.cmd

 Compiler.exe - 0 error
Parsing the file C:\Documents and Settings\Administrator\Desktop\AssoFiles\envir
o.xml  with the version:Version 2.1 of the Asso library

 The parser detected no Syntatctic Errors

 The parser detected no semantic Errors

 Parsing OK

Parsing the Metadata file C:\Documents and Settings\Administrator\Desktop\AssoFi
les\metaenviro.xml with the Version 2.1 of the Asso library

 The parser detected no Syntatctic Errors


C:\Documents and Settings\Administrator\Desktop\asso\DISS02-10>pause
Premere un tasto per continuare . . .
```

According to the described **BQKB0Y01.PAD** file DISS generates:

Output.sds that is written with the *XML format* and contains both a summary of the information about individuals and their symbolic description and the dissimilarity matrix computed by DISS. This file could be analysed by means of VDISS module

Note that this file could be also visualized in html format by putting the xsl file "asso2.2.xsl" in the same directory of the xml file (we assume that the output xml file includes line <?xml-stylesheet type="text/xsl" href="asso2.2.xsl"?>).

- Parser.dat that contains log information about the parser of the input ASSO file (in this case enviro.xml) and metadata file (e.g. metaenviro.xml).
- Error.tmp that contains error messages.
- BQKB0Y01.TXT that contains the number of individuals and dissimilarity measure computed by DISS.
- BQKB0Y01.LOG that contains log information.
- BQKB0Y01.LST (see Figure 9) that is the report file generated by DISS and contains information about the selected variables, the dissimilarity measure chosen for BSO and PSO, the dissimilarity matrix computed by DISS,…

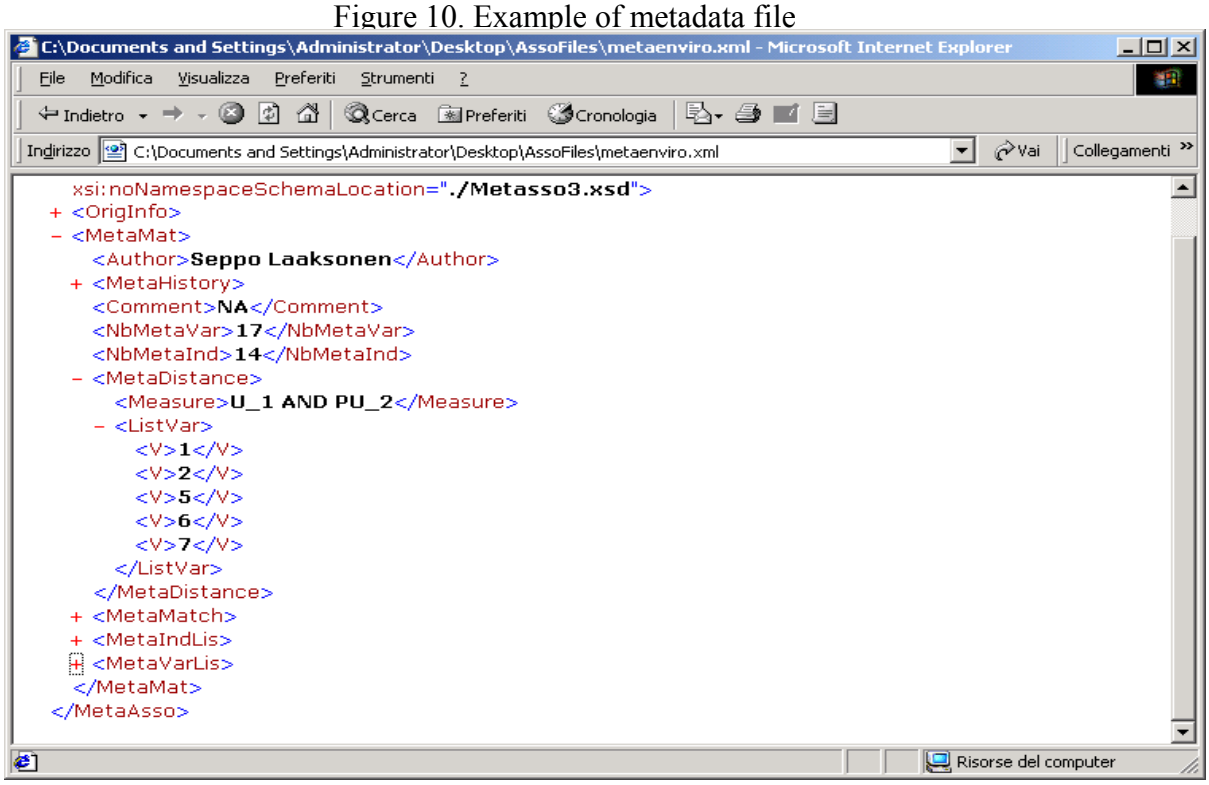
Figure 9: Output of DISS

```
 Page 1                    ASSO              12/01/03
            Asso The Statistical Package for Symbolic Data Analysis
                                Version 2.2  -  26 November 2003

          **************D I S T A N C E   M E A S U R E S*************
Data Information:
   COMBINE
    Input Asso File: C:\Documents and Settings\ASSO1\Desktop\ASSOFILES\enviro.xml

 14        Mixed Probabilistic and Boolean Symbolic Objects (PSOs - BSOs) read.
 17        Variables selected for each mixed SO: 1 2 5 6 7
Selected Distance Function: PU_2   Distance Function for PSO derived by aggregating Lp
    PI:    1
Selected Distance Function for boolean variables: U_1

Distance Matrix

  PSO        "AA00"      "AA01"      "AA02"      "AA03"

"AA00"    0.0000e+000
"AA01"    1.0322e+001  0.0000e+000
"AA02"    1.2277e+001  9.6805e+000  0.0000e+000
"AA03"    1.7254e+001  1.3079e+001  1.4108e+001  0.0000e+000
"AA04"    1.8867e+001  1.7790e+001  1.3861e+001  1.5466e+001
-----------------------------------------------------------------------
Page 2                    SODAS              12/01/03
  PSO        "AA00"      "AA01"      "AA02"      "AA03"

"AA05"    8.5181e+000  1.0901e+001  1.4129e+001  1.6538e+001
"AA06"    1.5827e+001  1.5283e+001  1.5607e+001  1.5736e+001
"AA08"    1.3710e+001  1.7540e+001  1.6964e+001  1.5439e+001
"AA09"    1.6157e+001  1.6280e+001  1.6779e+001  1.5331e+001
"AA10"    1.7131e+001  1.7330e+001  1.4200e+001  1.5826e+001
"AA11"    1.7531e+001  1.8623e+001  1.5148e+001  1.5713e+001
"AA12"    1.6846e+001  1.7382e+001  1.4607e+001  1.6291e+001
"AA13"    1.2586e+001  1.6948e+001  1.6591e+001  1.5722e+001
"AA14"    1.5845e+001  1.6566e+001  1.5546e+001  1.6558e+001

  PSO        "AA04"      "AA05"      "AA06"      "AA08"
"AA04"    0.0000e+000
"AA05"    1.9766e+001  0.0000e+000
"AA06"    1.7434e+001  1.6677e+001  0.0000e+000
"AA08"    1.2479e+001  1.4368e+001  1.6782e+001  0.0000e+000
"AA09"    1.1989e+001  1.5574e+001  1.5980e+001  7.4430e+000
"AA10"    1.5209e+001  1.8984e+001  1.0959e+001  1.8355e+001
"AA11"    8.7415e+000  1.7697e+001  1.7134e+001  8.6222e+000
"AA12"    1.4310e+001  1.8674e+001  1.1843e+001  1.7650e+001
-----------------------------------------------------------------------
Page 3                    SODAS              12/01/03

  PSO        "AA04"      "AA05"      "AA06"      "AA08"
```

BQKB0Y01.VDI that contains the complete path and the name (specified in SDO_OUT) of the output file generated by DISS. The name BQKB0Y01.VDI is obtained by combining the name of the pad file with the extension .VDI. The file BQKB0Y01.VDI is stored in the directory where the base chaining file (<name of chain>.FIL) is stored.

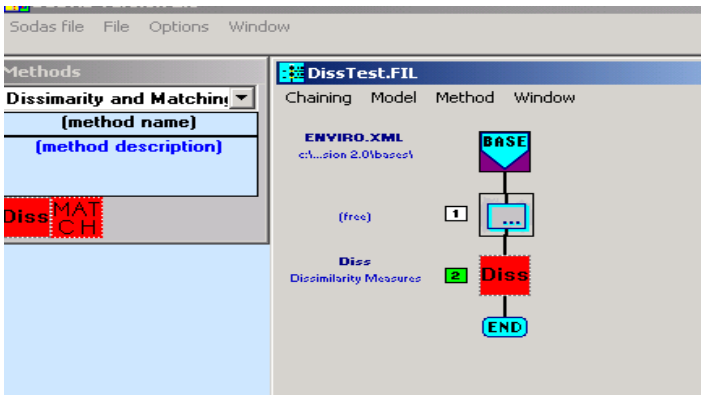
DISS also updates the metadata file associated to the input ASSO file by adding in *metaenviro.xml* information about the dissimilarity measures and the variables selected by the user (see Figure 10).

Figure 10. Example of metadata file



6.2 DISS modules in ASSO Workbench

Figure 11. A running chaining with DISS

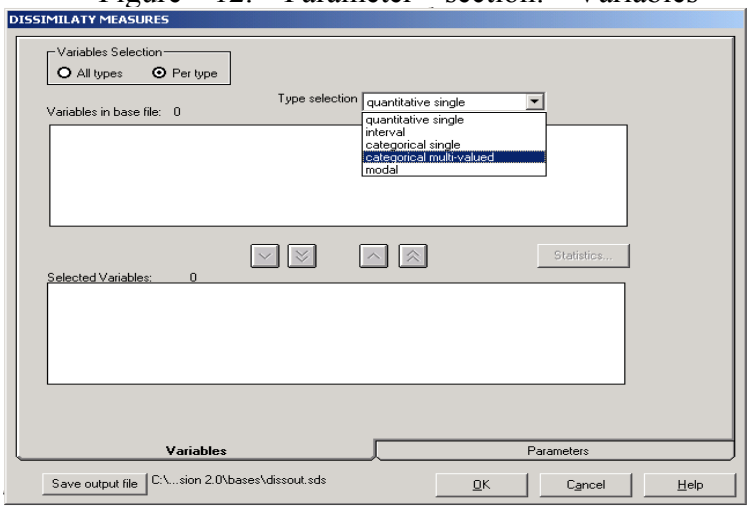


ASSO user chooses "*Dissimilarity and matching*" from the set of methods available and he/she opens an existing ASSO chain. Alternatively the user may decide to create a new ASSO chaining.

For instance, the user decides to create a new chaining named **DissTest** associated to the ASSO file **enviro.xml**. A metadata file named **metaenviro.xml** is associated to the input ASSO file. Then he/she adds DISS to the current running chaining by clicking on the "*base*" block and choosing "*insert method*". A new empty block is added to the chaining. The user selects DISS method and drags it on the empty block (see Figure 11) .Enviro.xml contains 14

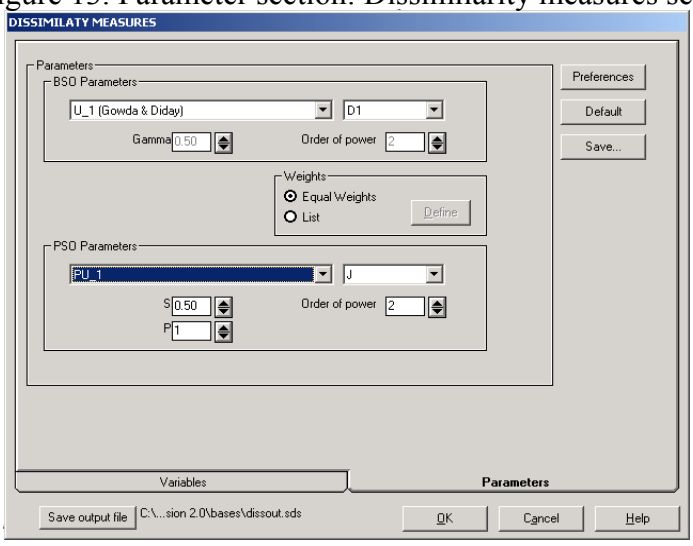
symbolic objects described by 17 mixed variables, both boolean and modal. In the *Parameter Section* the user may select all these variables or only a subset of them (see Figure 12).

Figure 12. Parameter section: Variables



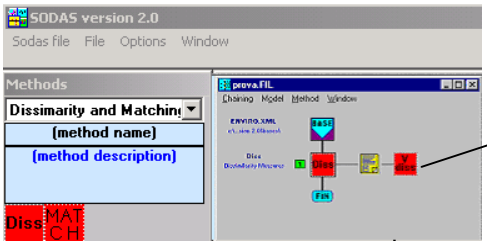
The user has also to choose the dissimilarity measure to be used to evaluate the dissimilarity between each pair of SOs stored in input ASSO file. In this example, the user selects all variables so the SOs to be analysed are mixed BSOs and PSOs. Therefore the user has to choose both a dissimilarity measure for BSOs and a dissimilarity measure for PSOs (see Figure 13).

Figure 13. Parameter section: Dissimilarity measures selection



Finally the user executes DISS method by choosing "*run*" from "*Method*" menu. A new block corresponding to the report file generated by DISS is added to the current running chaining. The user could analyse this report by clicking on it (see Figure 14).

Figure 14: DISS output



Page 1 ASSO 10/03/03
Asso The Statistical Package for Symbolic Data Analysis

Version 1.0 - 27 March 2003

******D I S T A N C E M E A S U R E S*****

Data Information:

Input Asso File: C:\Programmi\DECISIA\SODAS version 2.0\bases\enviro.xml

14 Mixed Probabilistic and Boolean Symbolic Objects (PSOs - BSOs) read.

17 Variables selected for each mixed SO: 1 -- 17

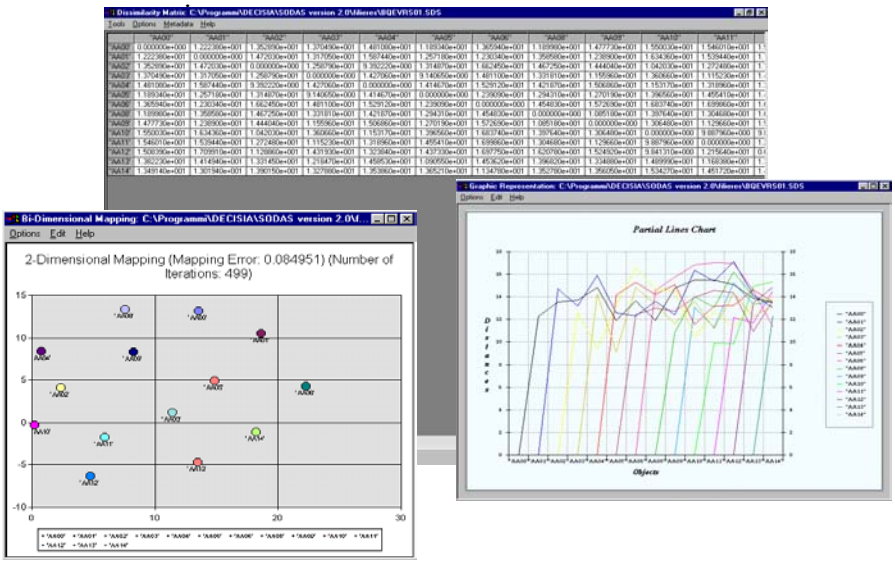
Selected Distance Function: PU_1 Generalized Minkowski's measure for PSO

Selected Comparison Function for probabilistic variables: LP Minkowski's Lp distance

D- 1

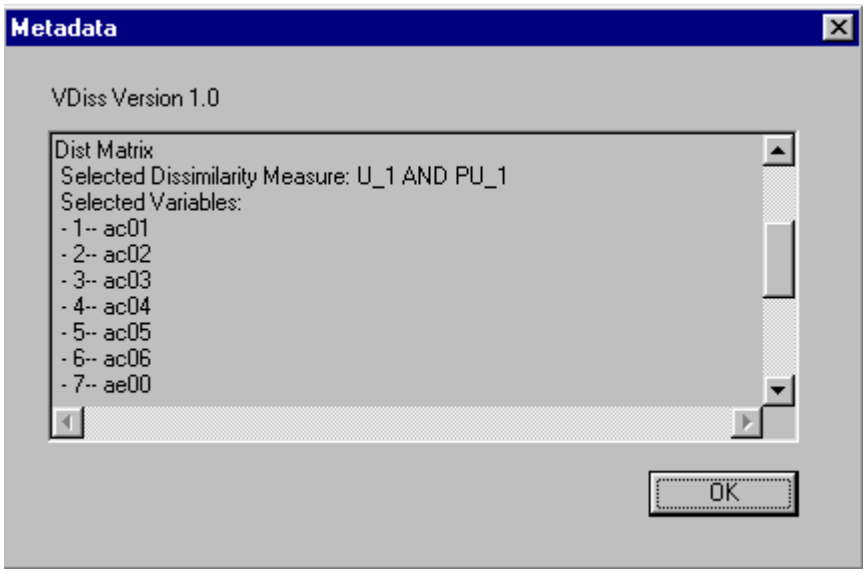
The dissimilarity matrix computed by DISS could be also visualised by means of the visualisation module VDISS (see Figure 15). VDISS allows the visualisation of the dissimilarity matrix in a table format as well as a *scatterplot* of the Sammon's nonlinear mapping into a bidimensional space. It is also possible to visualise dissimilarity matrix by means of *Line charts.*

Figure 15: VDISS visualization of a dissimilarity



VDISS also allows the visualisation of the metadata associated to the computation of the current dissimilarity matrix (see Figure 16).

Figure 16: Visualisation of metadata



INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

MATCH User Manual

Matching Operators

Edited by DIB

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **24/11/2003**

MATCH : Matching Operators

1 Introduction

Matching is the process of comparing two or more structures to discover their likeness or differences. Similarity judgments in the matching process are directional: they have a *referent* and a *subject*. The former is either a prototype or a description of a class, while the latter is either a variant of the prototype or the description of an instance of the class. In its simplest form, matching is the process of comparing two structures just for equality. The test fails if the structures differ in at least one aspect. In more complex case, the matching compares the description of a class with the description of an individual in order to establish whether the individual can be considered an instance of the class.

Matching two structures is crucial into symbolic classification, pattern recognition or expert systems. Matching operators have been defined both for BSO and PSO.

In MATCH two forms of matching are implemented: the *canonical* matching and the *flexible* matching. The former checks for an exact match whereas the latter one computes a degree of matching.

The library of matching functions available with the MATCH module is also used in the module SO2DB in order to retrieve individuals in a relational database corresponding to some characteristics described by a symbolic object.

2 The input to MATCH

The module allows the computation of the matching between a set of symbolic objects stored into a ASSO file.

Generally a symbolic object is described by a collection of set-valued variables and/or modal variables, where a set-valued variable could be:

- a categorical single-value variable (i.e. *town(w)=London* where '*town*' is a variable describing the individual *w*),
- a categorical multi-value variable (i.e. *town(w)={London, Paris, Rome}*),
- a quantitative single-value variable (i.e. *height(w)=3.5*),
- an interval variable (i.e. *height(w)=[3,7]*);

while a modal variable describes a probability distribution (i.e. *town(w)={London(0.2), Paris(0.7), Rome(0.1)}*).

According to these variables, the matching measures can be defined for BSOs, PSOs, mixed BSOs and PSOs. The last kind of SOs are described by both set-valued and modal variables (i.e. w=[$age(w)$=[20,50] ∧ $sex(w)$={F(0.4), M(0.5), I(0.1)}]).

Two forms of matching between BSOs have been implemented in MATCH:

- canonical matching (CM) that checks for an exact match,
- flexible matching (FM) that computes a degree of matching.

Only a probabilistic flexible matching (PFM) has been implemented for PSOs.

These choices are automatically stored in the parameter file (.pad) that manages the communication between the ASSO workbench and the MATCH module.

Figure 1: An example of parameter file for MATCH

```
SDS_IN = "C:\Documents and Settings\ASSO\Desktop\ASSOFILES\enviro.xml"
SDS_OUT="C:\Documents and Settings\ASSO\Desktop\ ASSOFILES\output.sds"
LOG = "C:\Programmi\DECISIA\SODAS version 2.0\filieres\BQKB0Y01.LOG"
OUT = "C:\Programmi\DECISIA\SODAS version 2.0\filieres\BQKB0Y01.DI0"

PROC_DISTANCE

::==== list of parameters for BSO =====
SIMFUN = FM
::==== list of parameters for PSO =====
.1.1.1.1      SIMFUN = PFM
COMPFUN = HELL
```

This parameter file is composed of four sections. The first section contains:

1. The ASSO file (PATH+NAME) that is the input of the entire running chaining (i.e. SDS_IN = "C:\ASSO\DISS\enviro.xml"),
2. The ASSO file (PATH+NAME) that is the output of MATCH (i.e. SDS_OUT="C:\ASSO\DISS\output1.sds"),
3. The log file (PATH+NAME) that contains the LOG of the MATCH computation (i.e. LOG = "C:\ASSO\DISS\example1.LOG"),
4. The output file (PATH+NAME) that reports the matching matrix (i.e. OUT = "C:\ASSO\DISS\example1.DI0").

The second section, also called BSO section, contains all parameters to compute the selected matching function for BSOs. Similarly, the third section, also called PSO section, contains all parameters to compute the selected matching function for PSOs. Finally the fourth section contains the list of the selected variables. A complete example of a parameter file is shown in Figure 1.

The parameter file together with the running chaining ASSO file are input to the module MATCH. Furthermore, MATCH automatically read the metadata stored in the metadata file named *meta<name of the input asso file>.xml* that is stored in the same path of the input ASSO file. In the case that metadata file is not available in the required path, MATCH shows a message and continues the execution.

3 The output of MATCH

The output is a square matrix stored in an output SODAS file. In particular, the result of the computation of the matching function between the i-th SO and the j-th SO in the input SODAS file is written in the (i, j)-entry of the matrix. Missing values are handled by simply substituting the whole set of possible values for each occurrence of a missing value.

MATCH module also updates the metadata file associated to the input ASSO file by adding information about matching measures for BSO and/or PSO used in the computation of the matching square matrix and the sub-set of selected symbolic objects.

4 Matching measures for boolean symbolic objects

In the case of matching between BSOs two matching functions were provided in the MATCH module: one for canonical matching and the other for flexible matching (see Table 1 for their theoretical definition).

Table 1: Matching functions for SO's

Name	***Definition***
	Let a, b be two BSO's: $a = [x_1 = A_1] \wedge \ldots \wedge [x_n = A_n]$, $b = [x_1 = B_1] \wedge \ldots \wedge [x_n = B_n]$
CM	*canonical-matching(a,b) = true* *if* $B_i \subseteq A_i$ *for each i=1, 2, …, n,* *canonical-matching (a,b) = false* *otherwise.*
FM	*flexible-matching(a,b)* = 1 *if canonical-matching(a,b)=true,* $\textit{flexible-matching(a,b)} \underset{def}{=} \max_{b' \in S_a} P(b \mid b')$ *otherwise* where $S_a = \{b' \in S \mid \textit{Canonical-matching(a,b')=true}\}$ and $P(b \mid b') = \prod_{i=1}^{n} P(b_i \mid b') = \prod_{i=1}^{n} P([x_i = Value_i] \mid [x_i = Value'_i]) = \prod_{i=1}^{n} P(\delta_i(Value'_i, X) \geq \delta_i(Value'_i, Value_i))$

The following assumptions on the distance δ_i are made whenever such distance is not specified:

- for continuous-valued variables, we assume that δ_i is L_1-norm:

$$\delta_i(x, y) = | x - y |,$$

- for nominal variables, we assume that δ_i is the binary distance:

$$\delta_i(x, y) = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{otherwise} \end{cases}$$

By supposing that values in the domain O_i are uniformly distributed, where:

$$P([x_i = Value_i] \mid [x_i = Value'_i]) = \frac{|O_i| - 1}{|O_i|}.$$

Therefore:

$$\textit{flexible-matching(a,b)} = \max_{b' \in S_a} \left(\prod_{i=1}^{n} \sum_{j=1}^{p} \frac{1}{p} P(b_{ij} \mid b'_i) \right)$$

where p is the number of categories for the variable j in the symbolic object b.

The matching operators for BSOs implemented in the MATCH module and the corresponding parameters are shown in table 2.

Table 2: Matching functions for BSO's

Dissimilarity measure	***Parameters***	***Constraints***	***Default***	***Example in .pad***
CM	none			. . . ::==== list of parameters for BSO ===== SIMFUN = CM . . .
FM	none			. . . ::==== list of parameters for BSO ===== SIMFUN = FM . . .

5 Matching measures for probabilistic symbolic objects

An extension of flexible matching is applicable to the case of a pair of two PSO's.

Let *a*, *b* be two PSO's; then we can define "flexible matching between a and b" as follows:

$$flexible\text{-}matching(a,b) = \max_{b' \in S_a}\left(\prod_{i=1}^{n} P(b'_i)\sum_{j=1}^{p} P(b_{ij})\,P(b_{ij} \mid b'_i)\right)$$

where $P(b'_i)$ is the probability of the symbolic object that plays the role of class while $P(b_{ij})$ is the probability of the other symbolic object.

As a particular case, this formula can be also used to compute the flexible matching of a PSO against the description of an individual (i.e., a PSO having single-valued variables).

In the matching module the following non-symmetric comparison function between two modal (probabilistic) symbolic variables P and Q taking values on the same range *Y* are also implemented:

- *I-divergence*

$$I(Q \mid P) = \sum_{y \in Y} \log\left\{\frac{q(y)}{p(y)}\right\} \cdot q(y)$$

This coefficient is also called ***Kullback-Leibler divergence***, ***relative*** or ***cross-entropy*** or ***information distance***. Moreover, it is used to calculate the error generated by approximating *Q* with *P*.

- *χ^2-divergence*

$$d(P,Q) := \sum_{y \in Y} \frac{|p(y) - q(y)|^2}{p(y)}$$

In classical statistics, this coefficient is known as the *noncentrality parameter* which controls the performance of the χ^2 goodness-of-fit test.

- *Hellinger coefficient of the order t*

$$s^{(t)}(P,Q) := \sum_{y \in Y} q^{t}(y) \cdot p^{1-t}(y).$$

with $0<t<1$.

The Hellinger coefficient is a similarity-like coefficient with $0 \leq s^{(t)}(P,Q) \leq 1$, $s^{(t)}(P,Q))=1$ for $P=Q$ and $s^{(t)}(P,Q)=0$ if P and Q have disjoint supports. Hellinger's special case $t=1/2$ yields a symmetric coefficient which has been termed ***Bhattacharyya distance***: in this case we can't involve this coefficient in the computation of flexible matching.

I-divergence and χ^2*-divergence* are two dissimilarity coefficient and so they are not suitable to computing a degree of matching. Thus we have to transform these one in similarity coefficient. We can obtain this transformation through the following formula:

$s(P,Q) := e^{-x}$

where x is one of two previous coefficient.

All these coefficients will be aggregated by product because the matching operators return a value in the interval [0,1]. Then

$$\textit{flexible-matching}(a,b) = \prod_{i=1}^{n} s(A_i, B_i).$$

The matching operators for PSOs implemented in the MATCH module and the corresponding parameters are shown in table 3.

Table 3: Matching functions for PSO's

Dissimilarity measure	***Parameters***	***Constraints***	***Default***	***Example in .pad***
PFM	COMPFUN S	I | CHI2 | HELL (0,1)	HELL 0.5	
In particular:				
PFM(none)	none			… ::==== list of parameters for PSO ===== SIMFUN = PFM …
PFM(I)	none			… ::==== list of parameters for PSO ===== SIMFUN = PFM COMPFUN = I …

PFM(CHI2)	none			… ::==== list of parameters for PSO ===== SIMFUN = PFM COMPFUN = CHI2 …
PFM(HELL)	S	(0,1)	0.5	… ::==== list of parameters for PSO ===== SIMFUN = PFM COMPFUN = HELL S = 0.5 …

6 Example - Stand-alone version

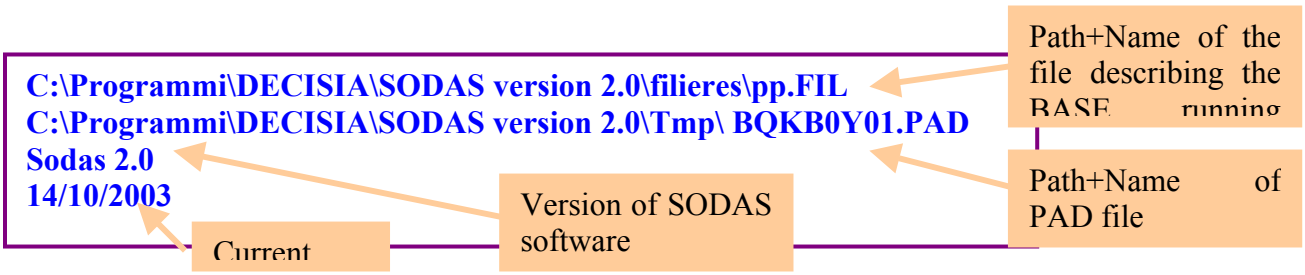
MATCH stand-alone version running command: **MATCH BQKB0Y01.cmd**

Pre-conditions →

Match module stand-alone version works with the following files:

1. **MATCH.cmd** -> it contains the path of file parameter file (see Figure 2).

Figure2 : Example of a command file for MATCH



2. **BQKB0Y01.PAD** -> it contains all the information needed to execute directly MATCH module (see Figure 3).

3. **Match.dll** -> it contains all the methods used to compute the matching measures

   It is important that Match.exe and MatchDLL.dll are in the same directory!

Match module execution →

Open MS-DOS Window
Work directory position
Digit **match.exe BQKB0Y01.cmd** as in the example in figure 34

Figure 3. Example of a parameter file for MATCH

```
SDS_IN = " C:\Documents and Settings\Administrator\Desktop\AssoFiles \enviro.xml"
1.2     SDS_OUT=" C:\Documents and Settings\Administrator\Desktop\AssoFiles
        \output.sds"
LOG = "C:\Programmi\DECISIA\SODAS version 2.0\filieres\BQKB0Y01.LOG"
OUT = "C:\Programmi\DECISIA\SODAS version 2.0\filieres\BQKB0Y01.LST"

PROC_DISTANCE
::==== list of parameters for BSO =====
SIMFUN = FM

::==== list of parameters for PSO =====
SIMFUN = PFM
COMPFUN = HELL
```

Figure 4. Example of running of MATCH

```
Seleziona C:\WINNT\System32\cmd.exe

C:\AssoTest\Match>Match.exe prova.cmd

 Compiler.exe - 0 error
Parsing the file C:\Documents and Settings\Administrator\Desktop\AssoFiles\envir
o.xml  with the version:Version 2.1 of the Asso library

 The parser detected no Syntatctic Errors

 The parser detected no semantic Errors

 Parsing OK

Parsing the Metadata file C:\Documents and Settings\Administrator\Desktop\AssoFi
les\metaenviro.xml with the Version 2.1 of the Asso library

 The parser detected no Syntatctic Errors


C:\AssoTest\Match>pause
Premere un tasto per continuare . . .
```

According to the described **BQKB0Y01.PAD** file MATCH generates:

Figure 5. Example of report file generated by MATCH

```
 Page 1                                ASSO                        11/24/03
        Asso The Statistical Package for Symbolic Data Analysis

                        Version 2.2 - 24 November 2003

* F L E X I B L E   A N D   P R O B A B I L I S T I C   F L E X I B L E   M A T C H I N G *

  Data Information:

  Input Sodas File: C:\Documents and Settings\Administrator\Desktop\AssoFiles\enviro.xml
   14     Mixed Boolean and Probabilistic SOs read.
   17     Mixed Variables selected for each SO: 1 -- 9

      Selected Comparison Function for PSOs: HELL   Hellinger Coefficient of the order s
       S:   0.4

 Matching Matrix

     Mixed SO            "AA00"        "AA01"        "AA02"        "AA03"

  "AA00"        1.0000        0.0000        0.1148        0.0000
  "AA01"        0.0000        1.0000        0.2326        0.0129
  "AA02"        0.0420        0.4080        1.0000        0.0000
  "AA03"        0.0000        0.0087        0.0000        1.0000
  "AA04"        0.0372        0.1252        0.3338        0.0000
  ---------------------------------------------------------------------------
  Page 2                          ASSO                        11/24/03
     Mixed SO            "AA00"        "AA01"        "AA02"        "AA03"

  "AA05"        0.0000        0.0082        0.0000        0.4005
  "AA06"        0.0043        0.0075        0.0167        0.0000
  "AA08"        0.1183        0.0000        0.0680        0.0000
  "AA09"        0.0000        0.1909        0.0841        0.0105
  "AA10"        0.0073        0.1106        0.1030        0.0000
  "AA11"        0.0000        0.0122        0.0000        0.0624
  "AA12"        0.0069        0.0403        0.0909        0.0000
  "AA13"        0.0000        0.0025        0.0000        0.1248
  "AA14"        0.0000        0.0000        0.0000        0.0000

     Mixed SO            "AA04"        "AA05"        "AA06"        "AA08"

  "AA04"        1.0000        0.0000        0.2552        0.0423
  "AA05"        0.0000        1.0000        0.0000        0.0000
  "AA06"        0.1490        0.0000        1.0000        0.0050
  "AA08"        0.0626        0.0000        0.0179        1.0000
  "AA09"        0.0460        0.0137        0.0000        0.0000
  "AA10"        0.1008        0.0000        0.0262        0.0419
  "AA11"        0.0000        0.1113        0.0000        0.0000
  "AA12"        0.1737        0.0000        0.1182        0.0279
  ---------------------------------------------------------------------------
  Page 3                          ASSO                        11/24/03
     Mixed SO            "AA04"        "AA05"        "AA06"        "AA08"
```

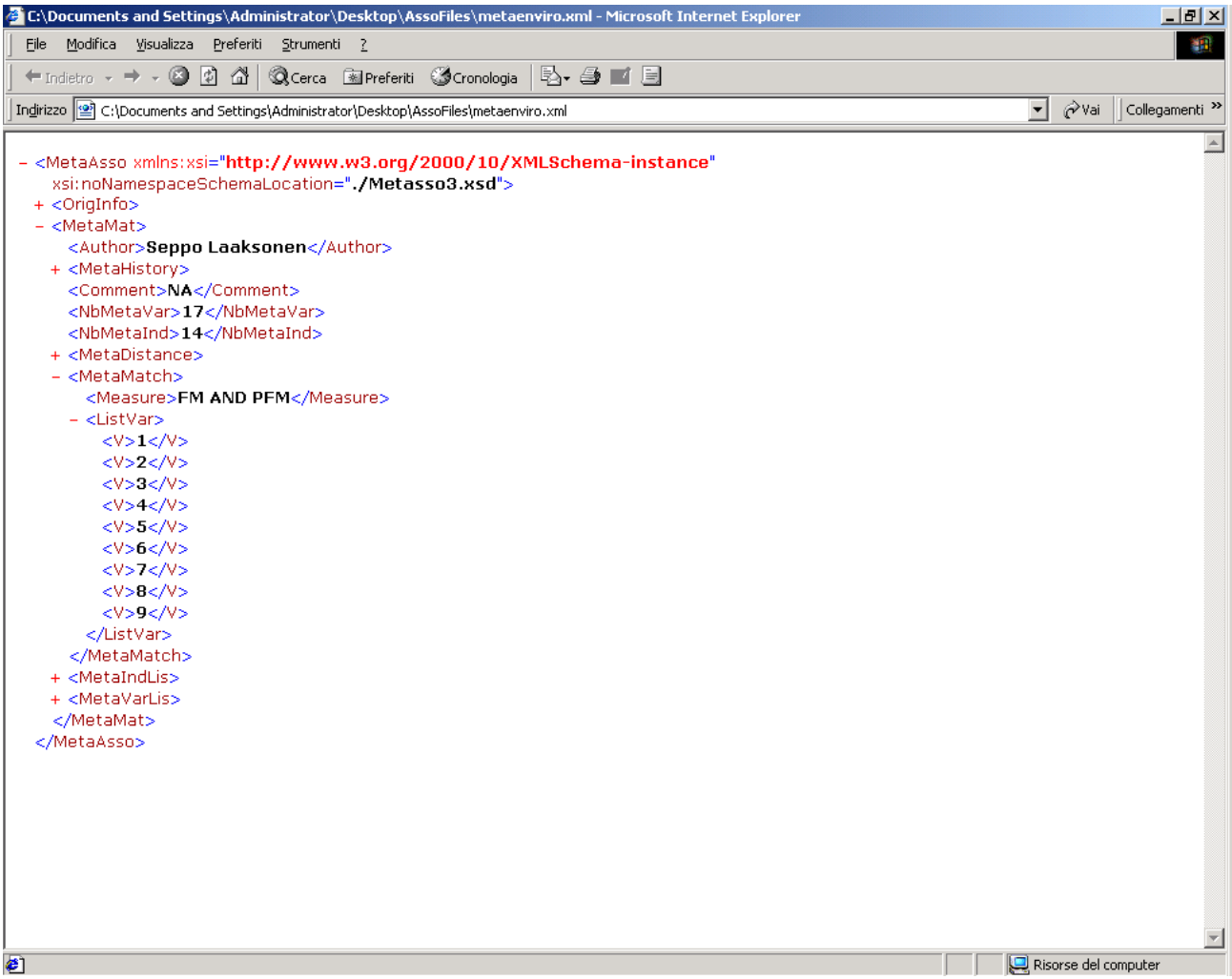
Output.sds that is written with the *XML format* and contains both a summary of the information about individuals and their symbolic description and the matching matrix computed by MATCH.

Note that this file could be also visualized in html format by by putting the xsl file "asso2.2.xsl" in the same directory of the xml file (we assume that the output xml file includes line <?xml-stylesheet type="text/xsl" href="asso2.2.xsl"?>).

- Parser.dat that contains log information about the parser of the input asso file (in this case enviro.xml) and metadata file (e.g. metaenviro.xml).
- Error.tmp that contains error messages.
- BQKB0Y01.TXT that contains the number of individuals and matching measure computed by MATCH.
- BQKB0Y01.LOG that contains log information.
- BQKB0Y01.LST that is the report file (see figure 5 generated by MATCH and contains information about the selected variables, the matching measure chosen for BSO and PSO, the matching matrix computed by MATCH,…

MATCH also updates the metadata file associated to the ASSO input file by adding in *metaenviro.xml* information about the matching measures and the variables selected by the user (see Figure 6).

Figure 6. Example of metadata file



Clustering

INFORMATION SOCIETY TECHNOLOGIES
PROGRAMME

HIPYR/VPYR User Manual

Hierarchical and Pyramidal Clustering

Edited by FEP

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

ASSO Project
Workpackages WP6.1 and 8.3 - Modules HIPYR and VPYR

Hierarchical and Pyramidal Clustering

Clustering and Visualizing Symbolic Data by using the modules HIPYR and VPYR

A Short Tutorial for Users

Paula Brito

Version 1: December 1, 2003

1 Clustering of Symbolic Data

Clustering is a multivariate statistical technique that aims at collecting similar individuals in homogeneous classes, on the basis of observed values in a set of variables. The constructed classes may be organized according to different structures; hierarchical and pyramidal clustering methods produce a structure of nested clusters. The module HIPYR allows to construct a hierarchy or a pyramid on a set of symbolic data. HIPYR assumes that you have a set of entities, hereafter called "individuals" (persons, institutions, cities, objects, etc.) that you wish to organize in a nested clustering structure. Individuals, associated to symbolic objects, are described by quantitative single, interval, nominal, multinominal and /or modal variables (mixed types are allowed). If a dissimilarity matrix is available, it may be used for a numerical clustering. Taxonomies and hierarchical dependencies defined on vari-

ables may be taken into account.
As an example, consider a data set consisting of the staff of some given teaching institutions, described by age (interval variable), marital status (categorical multi-valued variable) and staff category (modal variable):

	Age	Marital Status	Staff Category
Institution 1	[20,45]	{single, married}	Administration (30%), Teaching (70%)
Institution 2	[30,50]	{married, divorced}	Administration (20%), Teaching (60%), Cleaning (20%)
...	...	...	...
Institution n	[25,60]	{single, married, widow}	Administration (20%), Teaching (80%)

Table 1: Example of Data Array

Here the aim would be to gather similar institutions in clusters, so as to identify groups of institutions with similar staff.
HIPYR constructs the cluster structure based either:

- on the the individual-variable data set ;
- on dissimilarity values between the elements of the data set.

2 Principles of the method

The general aim of a clustering method is to aggregate the objects of a set E in homogeneous clusters. In the case of hierarchical or pyramidal clustering, the clusters formed are organized in a tree-like structure. In an ascending bottom-up approach, the most similar objects are merged together, then similar clusters are merged, until a unique cluster, gathering all the elements of E is formed. The clustering model to be used is to be chosen by the user: in the case of a hierarchy, each level corresponds to a partition; in the case of a pyramid we get, at each level,

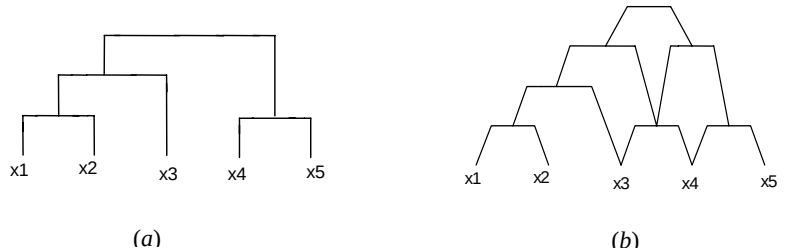


Figure 1: (a): Hierarchy (b): Pyramid

a family of overlapping clusters, but all clusters are intervals of a total linear order on E. Hence, a pyramid provides both a clustering and a seriation on the data. The pyramidal model, leading to a system of clusters which is richer than that produced by hierarchical clustering, allows for the identification of clusters that the hierarchical model would not identify; the existence of a compatible order on the objects leads, however, to a structure which is much simpler than lattices.
Two approaches are considered :

- The clustering is based on the symbolic objects data set. In this case, each cluster formed is associated to a conjunction of properties in the input variables, which constitutes a necessary and sufficient condition for cluster membership. Clusters are hence "concepts", described both extensionally, by the set of its members, and intentionally, by a symbolic object expressing its properties. This means that each cluster formed is by construction associated to a symbolic object, that generalizes its members, and such that no element outside the cluster meets the description given by this symbolic object. At each step, an additional numerical criterion is defined that allows choosing among the possible aggregations the "best" one. The method uses the "generality degree" which, for interval and categorical multi-valued variables evaluates the proportion of the underlying domain that is covered by a symbolic object and, for modal variables, in how much the given distribution is close to the uniform distribution. Two measures are available: the generality degree value of the formed cluster and the increase in the generality degree resulting by the formation of

the cluster.

- The clustering is based on a dissimilarity matrix between the elements of E. In this case,these dissimilarites must be computed by the DISS module and the input of HIPYR is a file containing a triangular dissimilarity matrix. Then, a classical hierarchical or pyramidal clustering algorithm is applied. The most similar clusters are aggregated at each step; aggregation measures, that evaluate the dissimilarity between clusters, such as complete linkage or single linkage, are considered. In this case, the clusters are not automatically associated to a discriminant description.

In the first case, the description of each cluster is given by the associated symbolic object, each cluster is defined by the set of its members together with its description (as a complete symbolic object that generalizes its members). In the second case, only dissimilarity values are available, so no description is given for the clusters.
Once the structure has been completed, a comparison measure between individuals may be obtained. This is an induced dissimilarity measure : for any two individuals, the induced dissimilarity is equal to the height of the cluster where they first appear together when exploring the structure bottom-up. This induced dissimilarity may then be compared with the dissimilarity or generality degree values directly obtained from the data.
If the hierarchy/pyramid is built from a symbolic data table, we obtain an inheritance structure, in the sense that each cluster inherits the properties associated to its predecessors. This fact allows to generate rules between clusters. Two methods are considered:

- Fusion method (pyramids only)

  Let (p_1, s_1), (p_2, s_2) be two concepts in the pyramid, and (p, s) be another concept such that $p = p_1 \cap p_2$. We can then write the rule:

$$s_1 \wedge s_2 \Rightarrow s$$

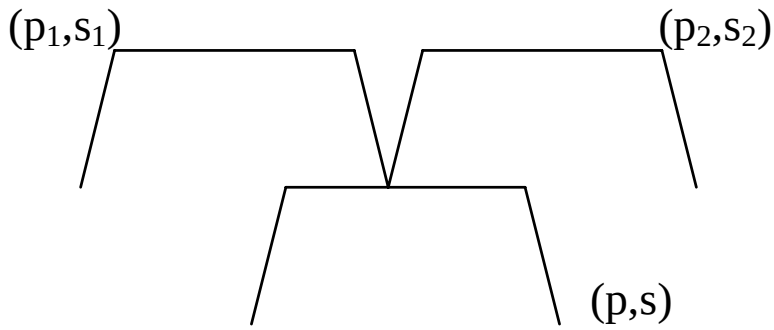


Figure 2: Situation for fusion rule

- Splitting method (hierarchies and pyramids)

  Let (p, s) be a concept obtained by merging (p_1, s_1) with (p_2, s_2), that is, $p = p_1 \cup p_2$, $s = s_1 \cup s_2$. Then,

$$s \Rightarrow s_1 \vee s_2$$

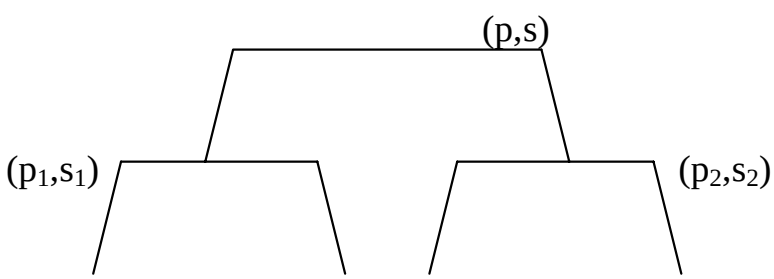


Figure 3: Situation for fission rule.

The visualization of the hierarchy or pyramid is performed by VPYR.

2.1 Pruning

Pruning of a hierarchy or pyramid consists in suppressing clusters of the hierarchy or pyramid with the aim of obtaining a structure which is easier to interpret, without an important loss of information. Let C be

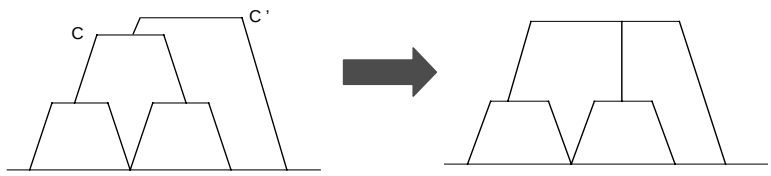


Figure 4: Pruning a Pyramid

a cluster with a single predecessor C' (in the case of pyramids a cluster may have up to two predecessors). Let f be the index function. Suppose that $f(C') - f(C) < \epsilon$ where $\epsilon > 0$ is a fixed threshold. We can then suppress C from the structure, without a great loss. The choice of ϵ depends on the degree of simplification we wish to achieve: a too small ϵ will almost not change the structure, a large ϵ will very strongly simplify it, and remove a lot of clusters; usually ϵ will be a suitable percentage of the maximum height. Once the graphical representation is displayed, the user may ask for a simplification of the structure, choosing the pruning parameter ϵ. In HIPYR, pruning may be performed from the graphical representation.

3 HIPYR's output: listing and graphical representations

The HIPYR algorithm produces as output a file .sds containing both the input data and a new set of symbolic objects: the clusters of the hierarchy or pyramid. Supplementary results are provided in a text file as:

- File and Options used
- List of clusters, and, for each cluster:
  - the clusters merged to form the present cluster
  - membership list
  - index value

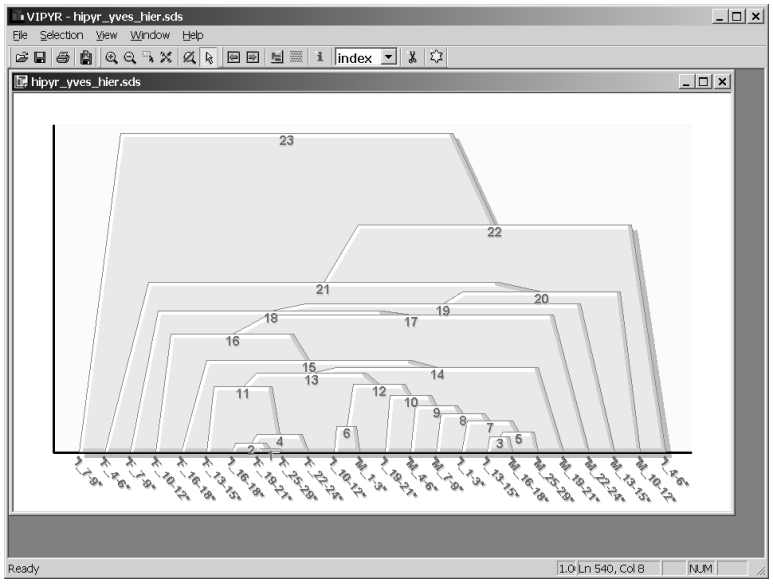


Figure 5: Graphical representation of a hierarchy

  - symbolic object: if the hierarchy or pyramid was built by a symbolic clustering method, then this symbolic description constitutes a necessary and sufficient condition for cluster membership.

- Evaluation value: it evaluates the fit between the the obtained structure and the original data, the lower its value the better the fit.

- The induced dissimilarity matrix, if requested.

A graphical output is provided by VPYR; several options are then available to explore the structure (hierarchy or pyramid):

- A cluster is selected by clicking on it. Then the user may get the description of the cluster in terms of a list of chosen variables or its representation by a Zoom Star.

- The user can prune the hierarchy or pyramid using the aggregation heights as a criterion. First, the rate of simplification must

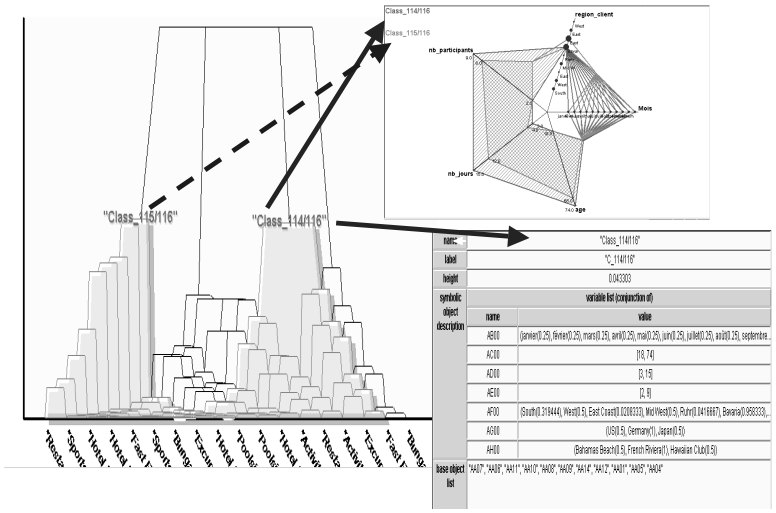


Figure 6: Interpretation tools on the graphical representation

be selected, then the user must click in the simplification button in the menu, this lunches a new graphic window with a simplified graphic.

- If the hierarchy/pyramid are built from a symbolic data table, rules may be generated and saved in a specified file.
- Should the user be interested in a particular cluster, he may obtain a window with the structure restricted to this cluster and its successors.

4 How is HIPYR working?

Let $E = \{w_1, \ldots, w_n\}$ be the set of units (individuals, groups,...) we wish to cluster. If the clustering is to be performed on the basis of a dissimilarity matrix, then the classical ascending hierarchical/pyramidal algorithm is applied, using a chosen aggregation index (dissimilarity between clusters) δ. In HIPYR, four options are available : single linkage (dissimilarity between two clusters is the minimum dissimilarity between members of one cluster and members of the other cluster), complete linkage (dissimilarity between two clusters is the maximum dissimilarity between members of one cluster and members of the other cluster), average linkage (dissimilarity between two clusters is the average of dissimilarities between members of one cluster and members of the other cluster) and diameter (dissimilarity between two clusters is the maximum dissimilarity between members of either cluster). Clusters of units are constructed in a recursive way: At each step a new cluster C is formed by merging suitable previously constructed clusters C_α and C_β, according to the desired clustering structure:

- if the structure is a hierarchy: none has been merged before;
- if the structure is a pyramid: none has been merged twice, and there is a linear order on E such that all clusters previously formed and C are intervals of this order.

Let P_t denote the set of clusters formed after step t and $S_t \subseteqq P_t \times P_t$ the set of pairs of elements of P_t that may be merged at step t+1, according to the chosen model.

- Initialisation: $P_0 = E$, $S_0 = P_0 \times P_0$, $C_i = \{w_i\}$, $f(C_i) = 0, i = 1, \ldots, n$
- Aggregation/Generalisation:
  After step t:
  $P_t = \{C_h, h = 1, \ldots, m\}$, $S_t = \{(C_h, C_{h'}) \subseteqq P_t \times Pt : C_h$ may be merged with $C_{h'}\}$
  While $E \neq P_t$:
  Let $(\alpha, \beta) : \delta(C_\alpha, C_\beta) = Min\{\delta(C_h, C_{h'})$ for $(C_h, C_{h'}) \in S_t\}$
  Then $C_{m+1} = C_\alpha \cup C_\beta$

$f(C_{m+1}) = Max\{\delta(C_\alpha, C_\beta), f(C_\alpha), f(C_\beta)\}$
$P_{t+1} = P_t \cup \{C_{m+1}\}$

If the hierarchy or pyramid are built o the basis of the data set - symbolic clustering - then the algorithm has some different features. Let again $E = \{w_1, \ldots, w_n\}$ be the set of units we wish to cluster, and s_i the symbolic object associated to $w_i, , i = 1, \ldots, n$. The initial set of concepts is $\{(w_1, s_1), , (w_n, s_n)\}$: it is assumed that all $(w_i, s_i), i = 1, , n$, are concepts. Let $G(s)$ be the generality degree of a symbolic object s. Let $s = s_\alpha \cup s_\beta$, then the clusters C_α and C_β to be merged should fulfill the following conditions:

- **a.** C_α and C_β can be merged together according to the desired clustering structure.
- **b.** $ext_E(s) = C$, i.e., no element of E outside C belongs to the extent of s (i.e. fulfills the conditions expressed by s).
- **c.** A numerical criterion is minimum. This criterion may be the generality degree of the resulting symbolic object s, $G(s)$ or the increase of the generality degree.

Then the concept corresponding to the new cluster is (C, s). If no pair of clusters C_α, C_β fulfills conditions **a** and **b**, the algorithm proceeds by trying to merge more than two clusters at a time (with the suitable adaptation of the merging conditions). By using the minimum generality degree criterion to choose among the pairs of clusters C_α, C_β fulfilling conditions **a** and **b**, the algorithm forms clusters first which are associated to less general symbolic objects. The new cluster C is indexed by $f(C) = G(s)$, the value of the generality degree of s. The algorithm comes to an end when the cluster E is formed which corresponds to a concept (E, s), for a suitable s. The following algorithm constructs an indexed hierarchy or a pyramid indexed in the broad sense, such that each cluster formed corresponds to a concept. Let again P_t denote the set of clusters formed after step t, Q_t the corresponding set of concepts and $S_t \subseteqq P_t \times P_t$ the set of pairs of elements of P_t that may be merged at step t+1, according to the chosen model. For the sake of simplicity of the presentation, we shall assume that $S_t \neq \varnothing$ in all steps.

- Initialisation: $P_0 = E$, $Q_0 = \{(w_1, s_1), , (w_n, s_n)\}$, $S_0 = P_0 \times P_0$, $C_i = \{w_i\}$, $f(C_i) = 0, i = 1, \ldots, n$

- Aggregation/Generalisation:
  After step t: $P_t = \{C_h, h = 1, \ldots, m\}$, $Q_t = \{(C_h, s_h), h = 1, \ldots, m\}$, $S_t = \{(C_h, C_{h'}) \subseteqq P_t \times Pt : C_h$ may be merged with $C_{h'}\}$
  While $E \neq P_t$:
  1 Let $(\alpha, \beta) : G(s_\alpha \cup s_\beta) = Min\{G(s_h \cup s_{h'})$ for $(C_h, C_{h'}) \in S_t\}$
  If $ext_E(s_\alpha \cup s_\beta) = C_\alpha \cup C_\beta$
  Then
  $C_{m+1} = C_\alpha \cup C_\beta$
  $s_{m+1} = s_\alpha \cup s_\beta$
  $f(C_{m+1}) = G(s_\alpha \cup s_\beta)$
  $P_{t+1} = P_t \cup \{C_{m+1}\}$
  $Q_{t+1} = Q_t \cup \{(C_{m+1}, s_{m+1})\}$
  Else
  $S_t = S_t \setminus (C_\alpha, C_\beta)$
  Go to 1

It is also possible in HIPYR to use the minimum increase in generality instead of the minimum absolute generality as criterion to merge clusters.

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

DCLUST User Manual

Clustering Algorithm based on Distance Tables

Edited by UFPE

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **08/04/2004**

Clustering algorithm based on Distance tables (DCLUST)

1 Introduction

The aim of the module DCLUST is to cluster interactively a large set of symbolic objects into a reduced (fixed) number of homogeneous classes, on the basis of a proximity table.

We use a clustering criterion which is based on the sum of dissimilarities between the individuals belonging to the same cluster, and try to minimize this clustering criterion by a suitable choice of the classes.

The proximity tables are provided by procedures from SODAS software (e.g., DI, DIM or DISS).

2 Principle of the method

The aim of the clustering process is to be able to group the objects of a set E in k homogeneous clusters on the basis of a dissimilarity table. The proposed approach is an application of the dynamical clustering algorithm applied to a dissimilarity table (Lechevallier, 1974). The algorithm follows the main principles of the method.
The number k of classes must be fixed, but a different number of classes can be request in order to look for the best partition in k classes. That can be performed by moving k between two integers corresponding to the minimum and maximum number of classes.
Moreover, the algorithm can be re-initialized in order to improve the final partition and the value of the optimized criterion.

3 DCLUST algorithm

- *Initialization*

  Let $L_o = \{s_1^{(o)}, \ldots, s_k^{(o)}\}$ be the initial prototypes, random objects of E

- *Allocation step t*:

  An object s_h is assign to the class $C_i^{(t)}$, iff $\psi(s_h, s_i^{(t-1)})$ is minimum.

- *Representation step t*:

  For $i = 1, \ldots, k$, find a prototype $s_i^{(t)}$ representing class $C_i^{(t)} \in P^{(t)}$ which minimizes $\sum_{s \in C_i^{(t)}} \psi(s_i^{(t)}, s)$.

- *Stopping rule or stability*

  If $P^{(t+1)} = P^{(t)}$ **then** STOP **else** GO TO *allocation step*

Alternatively, the following algorithm scheme can be used:

- *Initialization*

  Let $P^{(o)} = (C_1^o, \ldots, C_k^o)$ be the initial random partition of E in k classes.

- *Allocation step t*:

  For all h, any objects $s_h \in E$ is assign to the class C_i, iff $\sum_{s \in C_i^{(t)}} \psi(s_h, s)$ is minimum:

  $C_i^{(t+1)} = \{s_h \in \text{ E } \mid \sum_{s \in C_i^{(t)}} \psi(s_h, s) \leq \sum_{s \in C_m^{(t)}} \psi(s_h, s) \; \forall \; m \neq i\}$

- *Stopping rule or stability*

  If $P^{(t+1)} = P^{(t)}$ **then** STOP **else** GO TO *allocation step*

4 Input Data

The DCLUST algorithm can be performed on ordinal or interval scaled dissimilarity matrices. The dissimilarity table must be given by SODAS file (*e.g.*, using the procedures DI, DIM and DISS in SODAS software.

5 Output Data and Results

If the input is just a given dissimilarity table it is possible only to use the alternative algorithm DCLUST. There are no symbolic description of the classes. In that case, the DCLUST algorithm produces as output a ASSO file containing the original symbolic data table where it's added an indicator variable with the index of the classes of the final partition. If the input is the symbolic data table plus the proximity table, the basic DCLUST algorithm produces as output a file .sds containing a new set of symbolic objects: the clusters of the final partition, which are described by a description vector of the corresponding prototypes. In both basic and alternative DCLUST algorithm, it is added to the original symbolic data table an indicator variable with the index of the classes of the final partition.
Supplementary results are furnished in a text file as:

- List of classes (membership list, binary membership vector, class summary),
- Description vector associated to the prototypes (basic DCLUST algorithm),
- The relation for each variable,
- The extension mapping (basic DCLUST algorithm).

6 Parameters of DCLUST

DCLUST provides several different options for treating data, which differ, *e.g.*, by the definition of dissimilarities, cluster prototypes, update formulas etc. Therefore, the user must specify the values of a variety of parameters which characterize special options. Insofar the user is able to tune the parameters such that the resulting clustering method fits optimally his special application problem.
On the other hand, it is also possible to run DCLUST in a automatic way, *i.e.*, without caring too much about the correct choice of all parameters, or just by tuning only a part of the parameters: In fact,

DCLUST provides default (standard) values for all parameters, and replaces non-specified parameter values by standard values.
Besides, the menu of DCLUST ensures that the selected parameter combinations are consistent. That means, that it is not possible to combine some parameters in a way that makes no sense or is contradictory. That is why in some cases a couple of parameters cannot be chosen and appears grey in the parameter window.
The several options and parameters are explained below in the same order as they appear in the window.

6.1 Initialization

As every clustering method DCLUST assigns each object to that class, which has the most fitting prototype, initial class prototypes are needed to start the clustering method. The module offers two different options (Random prototypes, Random partition). If the first option is selected, the method will initially create k prototypes, whose prototypes are k random objects in the set of objects. If the second option is selected, the method will initially create k no empty clusters by randomization of the set of objects.

6.2 Number of classes

The number k of classes must be fixed, but a different number of classes can be request in order to look for the best partition in k classes. That can be performed by moving k between two integers corresponding to the minimum and maximum number of classes.

6.3 Number of runs

DCLUST treats the n data table in a sequential way. That means, it takes the first object, assigns it to a cluster, takes the second object, assigns it to a cluster, etc. When all n objects are assigned to their clusters DCLUST checks if a local optimal partition is attained. If this is not yet the case, the module DCLUST starts a new cycle in which all n objects are processed again in the previous order.

6.4 Number of iterations

To prevent an endless loop DCLUST stops after at most a maximal number of iterations or cycles which is determined by this parameter. Increasing this parameter can result in a linear increase in running time.

References

Bock, H. H., Diday, E. (eds) (2000), *Analysis of Symbolic Data*, Springer Verlag. Heidelberg.

Borg I., Groenen P. (1997). *Modern Multidimensional Scaling – Theory and Applications*, Springer–Verlag, New York.

Carroll, J.D. (1972). *Individual Differences and Multidimensional Scaling.* in Multidimensional Scaling Theory and Applications in the Behavioral Sciences, vol I, Theory, New York: Seminar Press.

Celeux, G. , Diday, E. , Govaert, G. , Lechevallier, Y. , Ralambondrainy, H. (1988) - Classification Automatique des Données : Environnement Statistique et Informatique - Dunod, Gauthier-Villards, Paris

Cox T. and Cox M. (1994). *Multidimensional Scaling*, Chapman and Hall,New York.

De Carvalho, F.A.T, Verde, R., Lechevallier, Y. (1999): A dynamical clustering of symbolic objects based on a context dependent proximity measure. In: H. Bacelar-Nicolau, F.C. Nicolau, J. Janssen (eds.): *Proc. IX International Symposium - ASMDA'99.* LEAD, Univ. de Lisboa, 237–242.

De Carvalho, F.A.T., Souza, R. M. C. (1999). New metrics for constrained Boolean symbolic objects. In: Studies and Research: *Proceedings of the Conference on Knowledge Extraction and Symbolic Data Analysis (KESDA'98).* Office for Official Publications of the European Communities, Luxembourg, 175–187.

Diday, E. (1972) Optimisation en classification automatique et reconnaissance des formes. *Revue francaise d'Automatique, Informatique et Recherche Opérationnelle (RAIRO)* **3**, 61.

Duda, R.O, Hart, P.E. (1973): *Pattern Classification and Scene Analysis.* Wiley, New York.

Gower, J. C. (1966) Some distances properties of latent root and vector methods using multivariate analysis. *Biometrika*, 53, 325–338.

Lebart, L., Morineau, A. and Piron, M.: 1995, *Statistique exploratorie multidimensionelle*, Dunod, Paris.

Lechevallier, Y. (1974): *Optimisation de quelques critères en classification automatique et application à l'étude des modifications des protéines sériques en pathologie clinique.* Thèse de l'université Paris VI.

Verde, R., De Carvalho, F.A.T. (1999): Dependence rules influence on factorial representation of Boolean symbolic objects. *Proceedings of the Conference on Knowledge Extraction and Symbolic Data Analysis (KESDA'98).* Office for Official Publications of the European Communities, Luxembourg, 287-300.

Verde, R., De Carvalho, F.A.T, Lechevallier, Y. (2000): A Dynamical Clustering Algorithm for Multi-nominal Data. In: H. A. L. Kiers, J.-P. Rasson P. J. F. Groenen, M. Schader (eds.): *Data Analysis, Classification, and related methods.* IFCS2000, Namur 387–394.

Verde, R., De Carvalho, F.A.T, Lechevallier, Y. (2001): A dynamical clustering algorithm for symbolic data. Tutorial "Symbolic Data Analysis", GfKl conference, Munich.

INFORMATION SOCIETY TECHNOLOGIES
PROGRAMME

SCLUST User Manual

Dynamic Clustering

Edited by INRIA

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

ASSO Tutorial for SCLUST

Project number :	**IST-2000-25161**
Project acronym :	**ASSO**
Project full title :	**Analysis System of Symbolic official data**

Deliverable number :	**WP6 - 0.1**
Title :	**Tutorial for SCLUST**
Workpackage contributing to the deliverable :	**Symbolic dynamic CLUSTering method**
Lead participant (short name) :	**INRIA**
Deliverable type	**Draft report**
Security :	**Rest.**
Author :	**Patrice Bertrand, André Hardy, Y. Lechevallier**

Abstract :	**This tutorial describes SCLUST module**
Keywords :	**Preliminary report**

Reference :	**ASSO/WP6/D6.3**
Version :	**0.3**
Date :	**30/11/03**
Total number of pages :	**28**
Status :	**Temporary**
Type of the document :	**Technical**
Authorized by :	**WP6 group**

Distribution List

Participant number	Participant short name	Recipients
1	FUNDP	Monique Noirhomme
		Edwin Diday
		Anne de Baenst
2	CISIA	Alain Morineau
3	TES	Driss Afza
4	FUNDPMa	Jean-Paul Rasson
		Andre Hardy
5	INRIA	Yves Lechevallier
		Marc Csernel
		Patrice Bertrand
6	DAUPHINE	Fabrice Rossi
7	RWTH	Hans Hermann Bock
8	UFPE	Francisco de Assis de Carvalho
9	INE	Carlos Marcelo
10	EUSTAT	Marina Ayestaran
11	STATFI	Seppo Laaksonen
12	FEP	Paula Brito
13	DMS	Carlo Lauro
		Rosanna Verde
14	DIB	Floriana Esposito
15	UOA	Haralambos Papageorgiu

Amendment History

Version	Date	Actions	Observations
0.0	November 2003	First version	

Title page

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

Project documentation

Tutorial for SCLUST module

Edited by

WP6 group

ASSO

Project acronym : **ASSO**
Project full title : **Analysis System of Symbolic Official data**
Proposal/Contract no. : **IST-2000-25161**

Date of preparation : **12/06/2001**

Contents

1 Content of the document **5**

2 Symbolic dynamic CLUSTering method (SCLUST) **5**
2.1 Principle of the SCLUST method 5
2.1.1 Prototypes for the clusters 5
2.1.2 Representation space 7
2.1.3 Representation function 7
2.1.4 Allocation function 8
2.1.5 The optimisation problem 8
2.1.6 Remark 8
2.2 SCLUST algorithm 8

3 The determination of the number of clusters for symbolic objects described by interval variables **9**
3.1 Introduction 9
3.2 The clustering problem 10
3.3 Methods for the determination of the number of clusters . . . 11
3.3.1 The Caliński and Harabasz method 11
3.3.2 The C-index 11
3.3.3 The Γ-index 12
3.4 Modelisation 12

4 Examples **13**
4.1 Symbolic artificial data set 13
4.2 Real data set: Oils and Fats 15
4.3 Real data set: 60 meteorological stations in China 15

5 Stability of a cluster **16**

6 Input Data **20**
6.1 Partitioning for Classical Data 20
6.2 Partitioning for Boolean Symbolic Data 21
6.3 Partitioning for Modal Symbolic Data 21
6.4 Partitioning for Mixed Symbolic Data 21
6.5 Remarks: 22

7 Output Data and SCLUST Results **23**

1 Content of the document

The tutorial is composed of three parts. In the first one we present a dynamical clustering algorithm, SCLUST. Its aim is to partition a set of symbolic objects, described by set-valued variables, in a predefined number k of clusters. In that method, it is possible to activate a module that produces sets of indices which allow the determination of the best number of clusters. The second part explain how these indices are produced and how to interpret them. In the third part we are interested in the validation of the partition produced by the algorithm. Some stability indices are produced in order to analyse the cohesion and the isolation of the clusters produced by SCLUST.

2 Symbolic dynamic CLUSTering method (SCLUST)

2.1 Principle of the SCLUST method

The aim of the clustering process is to be able to collect the objects of a set E into k homogeneous clusters. The proposed approach is an extension to symbolic data of the classical dynamic clustering algorithm (Dynamic clouds, (Diday, 1971)).

So the proposed Dynamic Clustering Method determines iteratively a series of partitions that improve at each step the underlying clustering criterion. The algorithm is based on:

- prototypes for representing the clusters
- context-dependent proximity functions for assigning the elements (symbolic objects) to the clusters at each step.

The clustering criteria to be optimized is based on the sum of proximities between objects and the prototype of the assigning clusters.

The main steps of the method are the following. In order to illustrate them, we will take the particular cases of interval or multivalued symbolic objects.

2.1.1 Prototypes for the clusters

$E = \{x_1, \ldots, x_n\}$ is a set of n objects described by p variables. The dynamic clouds clustering method need the specification of prototypes representing the clusters. In the case of classical quantitative variables, the representation of the object x_k is given by

$$x_k = (x_{k1}, \cdots, x_{kp}).$$

The prototypes of the clusters can be means, axes, groups of objects, ...

When the prototypes are the centroid of the clusters, they can be expressed as

$$g^{(\ell)} = (g_1^{(\ell)}, \cdots, g_p^{(\ell)})$$

where $g^{(\ell)}$ is the centroid of cluster C_ℓ
and

$$g_j^{(l)} = \frac{1}{n_l} \sum_{x_i \in C_l} x_{ij}.$$

In the case of interval variables, each object x_i is described by p interval variables. The interval variable Y_j is a set-valued variable defined by

$$\begin{array}{llll} Y_j\colon & \mathrm{E} & \rightarrow & \mathcal{B}_j \\ & x_k & \mapsto & Y_j(x_k) \quad = x_{kj} = [\alpha_{kj}, \beta_{kj}] \subset R. \end{array}$$

So each object can be represented by a hyperrectangle in an euclidean p-dimensional space.

The symbolic description of x_i is given by $x_i = ([\alpha_{i1}, \beta_{i1}], \ldots, [\alpha_{ip}, \beta_{ip}])$ and the prototype of cluster C_ℓ is the hyperrectangle of gravity of C_ℓ defined by

$$g^{(\ell)} = \left(\left[\frac{1}{n_\ell} \sum_{x_i \in C_\ell} \alpha_{i1}, \frac{1}{n_\ell} \sum_{x_i \in C_\ell} \beta_{i1} \right], \ldots, \left[\frac{1}{n_\ell} \sum_{x_i \in C_\ell} \alpha_{ip}, \frac{1}{n_\ell} \sum_{x_i \in C_\ell} \beta_{ip} \right] \right).$$

In the case of multivalued variables, $E = \{x_1, \ldots, x_n\}$ is a set of n objects described by p multivalued variables $Y_1, \ldots, Y_p$ with domains $\mathcal{Y}_1, \ldots, \mathcal{Y}_p$ respectively.

Let m_j denote the number of categories taken by Y_j. The frequency value $q_{j,x_k}(c_s)$ associated to the category c_s $(s = 1, \ldots, m_j)$ of the variable Y_j is given by

$$q_{j,x_k}(c_s) = \begin{cases} \frac{1}{|Y_j(x_k)|} & \text{if} \quad \mathrm{c_s} \in \mathrm{Y_j(x_k)} \\ 0 & \text{otherwise.} \end{cases}$$

The symbolic representation of the object $x_k \in E$ is a $(m_1 + \ldots + m_p)$-dimensional vector, given by

$$x_k = ((q_{1,x_k}(c_1), \ldots, q_{1,x_k}(c_{m_1})), \ldots, (q_{p,x_k}(c_1), \ldots, q_{p,x_k}(c_{m_p}))).$$

So the original data matrix $\underline{X} = (Y_j(x_k))$ is transformed into a frequency matrix $\tilde{X}$

	Y_1			$\cdots$	Y_p		
	1	$\cdots$	m_1	$\cdots$	1	$\cdots$	m_p
x_1	$q_{1,x_1}(c_1)$	$\cdots$	$q_{1,x_1}(c_{m_1})$	$\cdots$	$q_{p,x_1}(c_1)$	$\cdots$	$q_{p,x_1}(c_{m_p})$
$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$
x_k	$q_{1,x_k}(c_1)$	$\cdots$	$q_{1,x_k}(c_{m_1})$	$\cdots$	$q_{p,x_k}(c_1)$	$\cdots$	$q_{p,x_k}(c_{m_p})$
$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$
x_n	$q_{1,x_n}(c_1)$	$\cdots$	$q_{1,x_n}(c_{m_1})$	$\cdots$	$q_{p,x_n}(c_1)$	$\cdots$	$q_{p,x_n}(c_{m_p})$

where for all $x_k \in E$, and for all $j \in \{1, \cdots, p\}$, $\sum_{i=1}^{m_j} q_{j,x_k}(c_i) = 1$.

The prototype $g^{(\ell)}$ of C_ℓ is defined as

$$g^{(\ell)} = (\frac{1}{n_l} \sum_{x_i \in C_l} q_{1,x_i}(c_1), \ldots, \frac{1}{n_l} \sum_{x_i \in C_l} q_{1,x_i}(c_{m_1})), ..., (\frac{1}{n_l} \sum_{x_i \in C_l} q_{p,x_i}(c_1), \ldots, \frac{1}{n_l} \sum_{x_i \in C_l} q_{p,x_i}(c_{m_p})))$$

where n_ℓ is the number of objects in cluster C_ℓ.

2.1.2 Representation space

In the case of interval symbolic objects, the objects space is the space $\mathcal{I}^p$ of the closed bounded hyperrectangles. The representation space $\mathcal{L}$ of a cluster is also $\mathcal{I}^p$. The adequation measure D is defined by:

$$D : \mathcal{P}(E) \times \mathcal{L} \rightarrow R^+$$

such that

$$\forall A \in \mathcal{P}(E) \;\; and \;\; \forall x_i = ([\alpha_{i1}, \beta_{i1}], \ldots, [\alpha_{ip}, \beta_{ip}]) \in \mathcal{I}^p, \quad D(A, x_i) = \sum_{a \in A} d^2(a, x_i)$$

where d is a dissimilarity measure and $\mathcal{P}(E)$ the set of all subsets of E. Three distances are available in SCLUST for symbolic objects described by interval variables: the $\mathcal{L}_1$ and $\mathcal{L}_2$ distances, and the Hausdorff distance [BOC 00].

2.1.3 Representation function

The representation function g associates, to each partition $P = (C_1, \ldots, C_k)$ into k clusters, its representation $L = (L_1, \ldots, L_k)$, defined by:

$$\begin{array}{rrcl} g: & \mathcal{P}_k & \rightarrow & \mathcal{L}_k \\ & (C_1, \ldots, C_k) & \mapsto & (g^{(1)}, \ldots, g^{(k)}) \end{array}$$

where $g^{(\ell)}$ is the prototype of the cluster C_ℓ.

2.1.4 Allocation function

The allocation function f is defined by:

$$\begin{array}{rrcl} f: & \mathcal{L}_k & \rightarrow & \mathcal{P}_k \\ & (g^{(1)}, \ldots, g^{(k)}) & \mapsto & (C_1, \ldots, C_k) \end{array}$$

where

$$C_\ell = \{x \in E \ \mid \ d(x, g^{(\ell)}) \leq d(x, g^{(m)}), \ \ \forall\, m \in \{1, \ldots, k\}\}.$$

2.1.5 The optimisation problem

The problem is to find $(P^*, L^*) \in \mathcal{P}_k \times \mathcal{L}_k$ that minimizes the adequation criterion W between the partition $P = (C_1, \ldots, C_k)$ and its representation $L = (g^{(1)}, \ldots, g^{(k)})$ defined by:

$$W(P, L) = \sum_{\ell=1}^{k} D(C_\ell, g^{(\ell)}) = \sum_{\ell=1}^{k} \sum_{x_i \in C_\ell} d^2(x_i, g^{(\ell)})$$

where $g^{(\ell)}$ is the prototype of cluster C_ℓ.

So the purpose of the method is to minimize the within-clusters inertia W and thus to maximize the between-clusters inertia B of the partition.

2.1.6 Remark

SCLUST supposes that the number k of clusters has been fixed by the user. Nevertheless, a different number of clusters can be given in order to determine a better partition. Moreover, the algorithm can be re-initialized in order to improve the final partition and the value of the optimized criterion.

Part 3 of this document presents systematic methods in order to determine the best number of clusters.

2.2 SCLUST algorithm

- *Initialization*

  Let $P_o = \{C_{1o}, \ldots, C_{ko}\}$ be the initial random partition of E into k clusters.

- *Representation step t*: $L^{(t)} = g(P^{(t)})$

  For $i = 1, \ldots, k$, compute a prototype $g^{(i)}$ representing cluster $C_i \in P^{(t)}$

- *Allocation step t*: $P^{(t+1)} = f(L^{(t)})$

  For all h, any object $x_h \in E$ is assign to the class C_i, iff $d(x_h, g^{(i)})$ is minimum:

  $C_i^{(t+1)} = \{x_h \in E \ \mid \ d(x_h, g^{(i),t}) \leq d(x_h, g^{(j),t}) \ \ \forall \ \ j \neq i\}$

- *Stopping rule or stability*

  If $P^{(t+1)} = P^{(t)}$ **then** STOP **else** GO TO *representation step*

Note that the algorithm can be initialized by a random set of k objects of E. Then, the descriptions of these objects are assumed as the initial prototypes $g^{(1)}, \ldots, g^{(k)}$. Then, the initial partition is obtained by the allocation step.

Alternatively, the following algorithm scheme can be used:

- *Initialization*

  Lets L_o= $\{g^{(1),0}, \ldots, g^{(k),0}\}$ be the initial prototypes, random objects of E

- *Step t*: For all $x_h \in E$:

  - *Allocation phase*:
    An object x_h is assign to the class C_i, iff $d(x_h, g^{(i)})$ is minimum.
  - *Representation phase*:
    The prototype $g^{(i)}$ is upgraded by including the last allocated object.

- *Stopping rule or stability*

  If $P^{(t+1)} = P^{(t)}$ **then** STOP **else** GO TO *previous step*

In the first algorithm, the description of each prototype $g^{(i)}$ changes when all the objects have been assigned to the class. In the second one, it is modified after the assignment of an object to the class C_i

3 The determination of the number of clusters for symbolic objects described by interval variables

3.1 Introduction

One of the important problems in cluster analysis is the objective assessment of the validity of the clusters found by a clustering algorithm. The problem

of the determination of the "best" number of clusters has often been called the central problem of cluster validation.

Optimization methods for cluster analysis assume usually that the number of groups has been fixed a priori by the user. An important problem is then to select one solution in the sequence of partitions into ℓ clusters ($\ell =$ 1, 2, ..., K) given by a clustering algorithm. So the approach here is to ask the clustering algorithm to produce the best partitions into ℓ clusters ($\ell =$ 1, 2, ..., K), and each of the methods for the determination of the number of clusters to give a series of K values (a value is associated to each partition into ℓ clusters). The analysis of these values should allow the user to determine the best number of clusters in his data set.

We investigate here the problem of the determination of the number of clusters for symbolic objects described by interval variables.

Three stopping rules will be applied to the sets of partitions into ℓ clusters given by the SCLUST symbolic clustering procedure.

3.2 The clustering problem

The clustering problem we are interested in is the following.
$E = \{x_1,\ x_2,\ ...,\ x_n\}$ is a set of n objects characterized by the value of p interval variables $Y_1,\ Y_2,\ ...,\ Y_p$. We try to find a partition $P = \{C_1,\ C_2,\ ...,\ C_k\}$ of the set E into k clusters. The interval variable Y_j ($j = 1, ..., p$) measured on each object is defined by

$$\begin{array}{llll} Y_j & \mathrm{E} & \rightarrow & \mathcal{B}_j \\ & x_k & \mapsto & x_{kj} \quad = Y_j(x_k) = [\alpha_{kj}, \beta_{kj}] \subset R \end{array}$$

where $\mathcal{B}_j$ is the set of all closed bounded intervals of R.

The data matrix will take the form

$$X = (x_{kj})_{n \times p}$$

where x_{kj} is an interval.
The symbolic description of the object $x_k \in E$ is given by

$$x_k = ([\alpha_{k1}, \beta_{k1}], ..., [\alpha_{kp}, \beta_{kp}]).$$

The geometric visualization of a symbolic object of the interval type x_k is a hyperrectangle in the euclidean space R^p.
The methods for the determination of the number of clusters are applied to sets of partitions produced by the symbolic clustering procedure SCLUST.

3.3 Methods for the determination of the number of clusters

Many different stopping rules have been published in the scientific literature. The most detailed and complete comparative study that has been carried out appears to be that undertaken by Milligan and Cooper (1985). They conducted a Monte Carlo evaluation of thirty indices for the determination of the number of clusters, and they investigate the extent to which these indices were able to detect the correct number of clusters in a series of simulated data sets containing a known structure.

SCLUST is a partitioning symbolic clustering algorithm, but it is not a hierarchical one. So we have selected among the best stopping rules from the Milligan and Cooper (1985) analysis, the three best methods that are applicable to sets of partitions given by a non-hierarchical clustering procedure: the Caliński and Harabasz (1974) method (M1), the C-index (Hubert and Levin (1976)) (M2) and the Γ-index (Baker and Huber (1976)) (M3).

3.3.1 The Caliński and Harabasz method

The Caliński and Harabasz index is computed as

$$CH = [B/(c-1)]/[W/(n-c)] \tag{1}$$

where n is the total number of objects, and c the number of clusters in the partition. W and B denote, respectively, the total within-cluster sum of squared distances (about the centroids), and the total between-clusters sum of squared distances.

The maximum value of the index is used to indicate the correct number of clusters in the data set.

3.3.2 The C-index

That index need the computation of the sum of all within-cluster pairwise dissimilarities. If the partition has r such dissimilarities, we denote by V_{min} (respectively, V_{max}) the sum of the r smallest (respectively, largest) pairwise dissimilarities.

The C-index is then defined by

$$C = \frac{V - V_{min}}{V_{max} - V_{min}}. \tag{2}$$

The minimum value of that index across the partitions into ℓ clusters (ℓ = 1, ..., K) is used to indicate the optimal number of clusters, taking into consideration that the best minimal value is 0.

3.3.3 The Γ-index

Here comparisons are made between all within-cluster pairwise dissimilarities and all between-cluster pairwise dissimilarities. A comparison is defined as consistent (respectively, inconsistent) if a within-cluster dissimilarity is strictly less (respectively, greater) than a between-cluster dissimilarity.

The Γ-index is computed as

$$\Gamma = \frac{\Gamma_+ - \Gamma_-}{\Gamma_+ + \Gamma_-} \tag{3}$$

where Γ_+ (respectively Γ_-) represents the number of consistent (respectively, inconsistent) comparisons.

The maximum value of the Γ-index indicates the correct number of clusters. Let us remark that the absolute maximum of that index is 1.

An analysis of these stopping rules for the determination of the best number of natural clusters has been made for classical quantitative data sets test by Hardy and Deschamps (1999).

3.4 Modelisation

In order to handle symbolic interval data, we have chosen to model an interval by its Middle and its Lenght ((M, L) representation). So each symbolic object of the interval type can be represented as a point in a $2p$-dimensional space where p is the number of interval variables measured on each object.

The Calińsky and Harabasz method, the C-index, the Γ-index need the definition of a distance between the objects.

Let us consider two symbolic objects x_1 and x_2 described by p interval variables.
Let us denote by

$$x_1^j = (M_j(x), L_j(x)), \quad x_1^j \in R^2 \quad (j = 1 \ldots, p)$$
$$x_2^j = (M_j(y), L_j(y)), \quad x_2^j \in R^2 \quad (j = 1 \ldots, p)$$

the (M, L) representations of the objects x_1 and x_2, for each of the p interval variables.

The distance between x and y will be defined by

$$D(x, y) = \sum_{j=1}^{p} d(x_1^j, x_2^j).$$

It is in fact the sum of the p distances between the (M, L) representations x_1^j and x_2^j $(j = 1, \cdots, p)$ of the objects x_1 and x_2. In that formula, d is an

euclidean distance.

We have a distance matrix for the set E of objects. The three methods for the determination of the number of clusters can be implemented.

4 Examples

4.1 Symbolic artificial data set

The data set is composed of 30 objects described by two interval variables. So each object can be represented by a rectangle in the two-dimensional space R^2. The data were generated in order to present a well-defined structure into three clusters (Fig. 1). The original data are tabulated in Tab.1.

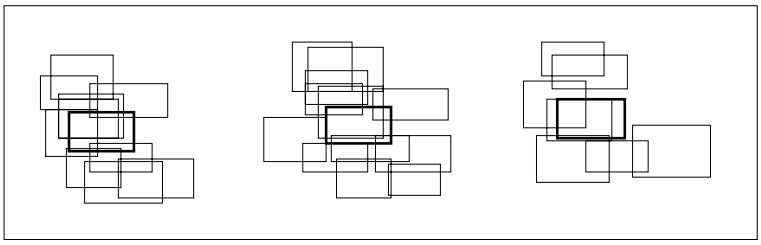


Figure 1: Visualisation of the data

The three rectangles printed in bold are the prototypes of the three clusters i.e. the hyperrectangles of gravity of the three natural clusters.

First we apply the SCLUST procedure to the original symbolic data, and we consider the best partitions into l clusters (l = 1, ..., K). We apply the (M, L) modelisation, compute the distance matrix and apply the three methods for the determination of the number of clusters.

If we fix the number of clusters to 3, SCLUST retrieves the natural structure into 3 clusters.

Concerning the determination of the number of clusters, the indices associated to the three stopping rules are tabulated in Tab. 2.

For the Calińsky and Harabasz method (M1), the maximum value of the index indicates the best number of clusters. It is attained with three clusters. For the C-index, we have to find the minimum value of the index. It indicates that the best number of clusters is also equal to three. Finally, the maximum value for the Γ-index is obtained for $k = 3$.

objects	variable 1	variable 2
1	[3.37; 6.28]	[$_0$.35; 1.10]
2	[0.81;3.17]	[3.35;5.11]
3	[2.19;4.69]	[1.57;1.72]
4	[1.06;3.59]	[1.92;3.55]
5	[2.33;5.31]	[2.72;3.95]
6	[0.63;2.64]	[1.21;3.04]
7	[0.43;2.59]	[2.96;4.28]
8	[2.10; 5.08]	[$_0$.63; 1.01]
9	[1.38;5.13]	[-0.04;1.53]
10	[1.05;3.42]	[1.91;3.37]
11	[13.21;16.12]	[2.63;3.78]
12	[10.07;12.43]	[3.66;5.58]
13	[10.46;12.96]	[0.58;1.66]
14	[11.05;13.58]	[1.93;3.87]
15	[10.66;13.64]	[3.64;5.35]
16	[13.79;15.81]	[-0.30;0.87]
17	[10.60;12.77]	[2.78;4.04]
18	[11.63;14.62]	[0.95;2.00]
19	[11.77;13.93]	[-0.44;1.14]
20	[9.02;11.39]	[1.00;2.68]
21	[13.27;16.18]	[0.64;2.03]
22	[10.64;13.00]	[3.13;4.50]
23	[19.68;22.07]	[4.30;5.59]
24	[21.39;23.84]	[0.63;1.68]
25	[20.14;22.99]	[1.71;3.30]
26	[19.64;22.07]	[3.81;5.07]
27	[19.01;21.44]	[2.32;4.12]
28	[23.21;26.17]	[0.43;2.38]
29	[19.48;22.32]	[0.24;1.99]
30	[19.90;22.37]	[1.80;3.37]

Table 1: Symbolic data table

SCLUST	M1	M2	M3
k=5	156.35205	0.00574	0.95018
k=4	165.37531	0.00527	0.96158
k=3	**189.95941**	**0.00002**	**0.99990**
k=2	67.04706	0.05495	0.81996
k=1	-	-	-

Table 2: Values of the indices: artificial data set

4.2 Real data set: Oils and Fats

Ichino's data [Bock et al, 2000] are reproduced in the following table. That real data set consists of eight types of oil, described by four quantitative interval variables: Specific gravity, Freezing point, Iodine value, Saponification value. The original data are tabulated in Tab.3 and the indices of the first three methods for the determination of the number of clusters in Tab.4.

Sample	Specific gravity	Freezing point	Iodine value	Saponification value
linseed oil	[0.930 : 0.935]	[-27 : -18]	[170 : 204]	[118 : 196]
perilla oil	[0.930 : 0.937]	[-5 : -4]	[192 :208]	[188 : 197]
cottonseed oil	[0.916 : 0.918]	[-6 : -1]]	[99 : 113]	[189 : 198
sesam oil	[0.92 : 0.926]	[-6 : -4]]	[104 :116]	[187 : 193
camelia oil	[0.916 : 0.917]	[-21 : -15]	[80 :82]	[189 : 193]
olive oil	[0.914 : 0.919]	[0 : 6]	[79 : 90]	[187 : 196]
beef tallow	[0.86 : 0.87]	[30 : 38]	[40 : 48]	[190 : 199]
hog fat	[0.858 : 0.864]	[22 : 32]	[53 : 77]	[190 : 202]

Table 3: Ichino's data

k	M1	M2	M3
7	0.34849	0.08378	0.57333
6	2.22292	0.15840	0.51304
5	2.54987	0.19063	0.50877
4	3.61575	0.19063	0.50877
3	13.46313	0.03633	0.63636
2	3.16596	0.42138	0.25000
1	-	-	-

Table 4: Values of the indices: Ichino data

The maximum value of the Calinski and Harabasz index is attained for $k = 3$. The C-index is the closest to 0 also when $k = 3$. The maximum value of the Γ-index is 0.63636; it corresponds to 2 clusters.

4.3 Real data set: 60 meteorological stations in China

This data set contains the monthly temperatures observed in 60 meteorological stations of China. According a natural representation of the temperatures, they are coded in a table as the interval of the minima and maxima for each month. For our example we have considered the temperatures of the year 1988 and we have built a table of dimension 60 rows and 12 columns, corresponding to the number of stations and to the number of months of the year.

SCLUST has been applied on these symbolic data described by interval variables, in order to obtain a partition into 5 clusters. For instance, the station "ChangSha" is described by the 12 intervals of the monthly temperatures:

```
 [January = [2.7:7.4]]        ^[February = [3.1:7.7]]
^[March = [6.5:12.6]]         ^[April = [12.9:22.9]]
^[May = [19.2:26.8]]          ^[June = [21.9:31]]
^[July = [25.7:34.8]]         ^[August = [24.4:32]]
^[September = [20:27]]        ^[October = [15.3:22.8]]
^[November = [7.6:19.6]]      ^[December = [4.1:13.3]]
```

Table 5 presents the data set. The full data can be found at http://dss.ucar.edu/datasets/ds578,5/data.

Meteorological stations	January	February	...	December
AnQing	[1.8 : 7.1]	[5.2 : 11.2]	...	[4.3 : 11.8]
BaoDing	[-7.1 : 1.7]	[-5.3 : 4.8]	...	[-3.9 : 5.2]
BeiJing	[-7.2 :2.1]	[-5.3 : 4.8]	...	[-4.4 : 4.7]
BoKeTu	[-23.4 : -15.5]	[-24 : -14]	...	[-21.1 : -13.1]
ChangChun	[-16.9 : -6.7]	[-17.6 : -6.8]	...	[-15.9 : -7.2]
ChangSha	[2.7:7.4]	[3.1:7.7]	...	[4.1:13.3]
ZhiJiang	[2.7 : 8.2]	[2.7 : 8.7]	...	[5.1 : 13.3]

Table 5. Minima and maxima monthly temperatures

In SCLUST, the number of clusters has been fixed to 5. The algorithm is reiterated 50 times and the best solution is found for the minimum value of the criterion equal to: W=3848.97. It is worth to noting that the obtain partition of the 60 elements follows the geographical contiguity of the stations.

According to the kind of representation of the classes by intervals proposed in the partitioning algorithm on interval data, the prototype of each class is the interval which minimizes the Hausdorff distance from all the elements belonging to the class.

Concerning the methods for the determination of the number of clusters, for the Calińsky and Harabasz method (M1), the maximum value of the index is attained with two clusters. The best number of clusters given by the C-index and the Γ-index is $k = 6$.

5 Stability of a cluster

Stability of a cluster measures the impact of removing a few objects from the data set, on the isolation and the cohesion of a cluster, when this cluster was obtained from a symbolic partitioning method.

Three stability indices are computed. Two of them estimate the inherent stability in the isolation (respectively, cohesion) of the examined cluster. By combining these two values of stability, a third index estimates the overall stability of the cluster. Using Monte Carlo tests, levels of significance assess how likely the values taken by the stability indices are under a null model that specifies the absence of cluster stability.

The aim of the procedure is to assess the stability of an arbitrary cluster of a partitioning, under the deletion of a few symbolic objects from the whole set S of symbolic objects. The method was introduced by Bel Mufti (1998) and by Bel Mufti and Bertrand (2001), for classical numerical data sets. In case of symbolic data sets, the method is strictly identical, except for the computation of convex hulls of data sets, which is needed in step **6** of the procedure described hereafter.

We denote by C the cluster whose stability is examined, and we assume that C is a cluster of a partition $\mathcal{P} = \{\mathcal{C}_1, \ldots, \mathcal{C}_k\}$ which was obtained by applying a partitioning method $\mathcal{M}$ to data set S. We do not make any assumption about the type of the examined data set S, which may therefore be either numerical or symbolic.

1. **Sampling the data set S.** A large number N_1 (e.g. $N_1 = 100$) of samples $S^{(1)}, \ldots, S^{(N_1)}$ are randomly selected from S. In order that the cluster structure of these samples should not depart significantly from the cluster structure $\mathcal{P}$ of S, each sample is drawn from S in the following way:

   - first, choose a large percentage value p, for example $p = 90\%$;
   - second, select randomly and without replacement, $[\, p \mid C_i \mid]$ symbolic objects [1], from each cluster $\mathcal{C}_i$ of $\mathcal{P}$, for $i = 1, \ldots, k$.

   It can be noticed that the sizes of the samples $S^{(1)}, \ldots, S^{(N_1)}$ are identical. Denoting by n_s this common size, we clearly have $n_s \approx [\, p\, n\,]$.

2. **Partitioning the samples.** Each sample $S^{(j)}$ (for $j = 1, \ldots, N_1$), is partitionned into k clusters by applying the method $\mathcal{M}$. We denote by $\mathcal{P}^{(1)}, \ldots, \mathcal{P}^{(N_1)}$ the so-obtained k-partitions on samples $S^{(1)}, \ldots, S^{(N_1)}$.

3. **Measuring the stability of a cluster, w.r. to its isolation.**

   We measure the stability of the cluster C, with respect to the effects of sampling on the isolation of the cluster C, by computing the index value $I(C)$:

$$I(C) = 1 - \frac{2}{n_s\,(n_s - 1) N_1} \sum_{j=1}^{N_1} \sum_{i=1}^{k} \mid C \cap C_i^{(j)} \mid \mid \overline{C} \cap C_i^{(j)} \mid,$$

[1] we denote by $[\, x\,]$ the integer-part of any real number x.

where $C_1^{(j)}, \ldots, C_k^{(j)}$ denote the clusters of the partition $\mathcal{P}^{(j)}$.

4. **Measuring the stability of a cluster, w.r. to its compactness.**

   We measure the stability of the cluster C with respect to the effects of sampling on the compactness of the cluster C, by computing the index value $H(C)$:

$$H(C) = 1 - \frac{2}{n_s\,(n_s-1)N_1} \sum_{j=1}^{N_1} \sum_{1 \le i < i' \le k} \mid C \cap C_i^{(j)} \mid \mid C \cap C_{i'}^{(j)} \mid .$$

5. **Computing an overall measure of cluster stability.**

   We measure the stability of the cluster C, with respect to the effects of sampling on both isolation and compactness of the cluster C, by computing the index value $G(C)$:

$$G(C) = I(C) + H(C) - 1$$

6. **Computing levels of significance.** We determine the level of statistical significance of the observed values $I(C)$, $H(C)$ and $G(C)$, using the following three-steps procedure of test:

   6.1 Randomly generate data sets are according to a data null model specifying the absence of cluster stability.

   6.2 Compute the distribution of the values of the stability index for clusters that are obtained by applying the partitioning method $\mathcal{M}$ to simulated data sets generated in step **6.1**.

   6.3 Compute the level of statistical significance of the observed value of the stability index measured for cluster C, on the basis of the previous empirical distribution. If the level of significance is low, e.g. less than 5%, then the hypothesis of no cluster stability is rejected.

Symbolic data null model

It should be noticed that step **6** in the above procedure, i.e. the determination of the levels of significance, is very similar to the cluster validation test proposed by Gordon (1994). In particular, the choice of an appropriate null model for data, requires careful attention (cf. Gordon (1996)).
The symbolic data null model used by this approach, consists in randomly selecting symbolic objects within the convex hull of the examined symbolic data set S. The computation of the convex hull of a symbolic data set, is not straightforward: this requires to code the examined symbolic objects as points of an euclidean space. When the variables are classical quantitative, such a coding is clearly not needed. When the variables are interval valued,

a simple coding consists in representing each interval as a point (x, y) where x (resp. y) is the center (resp. length) of the interval. The other types of symbolic variables are not taken into account by this task.

When the number variables is too large (i.e. greater than 10), an alternative of interest in order to avoid the combinatorial explosion due to the computation of convex hulls, is to use the ellipsoidal model. The ellipsoidal null model consists in randomly selecting the symbolic objects from a multivariate normal distribution whose variance-covariance matrix is the data set variance-covariance matrix. Then, the ellipsoidal null model still requires that symbolic objects be coded as points of an euclidean space.

Output Data of stability of a cluster

There are two levels of results. The first level provides the numerical values of the stability indices that are computed, for each examined cluster C, by using the input data of the first level of input. In other words, the first level of results contains:

- the index value $I(C)$ which estimates the inherent stability in the isolation of the examined cluster C,
- the index value $H(C)$ which estimates the inherent stability in the compactness of the examined cluster C,
- the index value $G(C)$ which estimates the inherent stability in both the isolation and compactness of the examined cluster C.

It should be noticed that $I(C)$, $H(C)$ and $G(C)$ are *relative* measures of stability. In other words, these measures allow only to compare the degrees of stability of clusters having approximatively the same size and which were obtained by the same partitioning method applied to the same data set. In particular, if two clusters have not been determined on the same data set, then these values do not indicate which of the two clusters can be viewed as more stable.

The levels of statistical significance of the values $I(C)$, $H(C)$ and $G(C)$, as defined in step **6** of the last paragraph, are more intrinsic stability measures. The second level of results contains these levels of significance of the values $I(C)$, $H(C)$ and $G(C)$ provided by the first level of results. It should be noticed that this second level of results requires the input data of the second level.

Results

There are two levels of results. The first level provides the numerical values of the stability indices that are computed, for each examined cluster C, by using the input data of the first level of input. In other words, the first level of results contains:

- the index value $I(C)$ which estimates the inherent stability in the isolation of the examined cluster C,

- the index value $H(C)$ which estimates the inherent stability in the compactness of the examined cluster C,
- the index value $G(C)$ which estimates the inherent stability in both the isolation and compactness of the examined cluster C.

It should be noticed that $I(C)$, $H(C)$ and $G(C)$ are *relative* measures of stability. In other words, these measures allow only to compare the degrees of stability of clusters having approximatively the same size and which were obtained by the same partitioning method applied to the same data set. In particular, if two clusters have not been determined on the same data set, then these values do not indicate which of the two clusters can be viewed as more stable.
The levels of statistical significance of the values $I(C)$, $H(C)$ and $G(C)$, as defined in step **6** of the last paragraph, are more intrinsic stability measures. The second level of results contains these levels of significance of the values $I(C)$, $H(C)$ and $G(C)$ provided by the first level of results. It should be noticed that this second level of results requires the input data of the second level.

6 Input Data

The SCLUST algorithm can be performed on data described by all the types of variables as well as mixed data. Missing data are also admitted.

6.1 Partitioning for Classical Data

Here the input data are described by classical variables Y_j, even if mixed types of variables Y_j: quantitative and qualitative (nominal or ordinal) ones. In contrast, a class prototype can be described by both classical and symbolic variables. Different kinds of distance or dissimilarity measures are proposed as allocation functions.

INPUT Variables	Prototypes G_i described by	Distances $\psi_j(s, G_i)$
quantitative	quantitative (real-valued)	Euclidian
nominal	nominal (categories)	χ^2
ordinal	ordinal (levels)	Euclidean rank
quantitative	interval	Hausdorff distance
nominal	categorical multi-valued	context-dependent
nominal	modal (frequency distr.)	context-dependent

Table 5: Default parameters for classical input data

6.2 Partitioning for Boolean Symbolic Data

Here the data are symbolic ones, described by symbolic variables of two types: categorical multi-valued, and interval variables. This method allows to construct a partition of SO's defined by both types. However, the class prototypes are always described by symbolic variables.

INPUT Variables	Prototypes G_i described by	Distance $\psi_j(s, G_i)$
interval	interval	Hausdorff distance
categorical multi-valued	modal	context-dependent
interval	modal	free and context-dependent

Table 6: Default parameters for Boolean symbolic input data

We accepted also the possibility of missing data (NULL).

6.3 Partitioning for Modal Symbolic Data

Here the input is a set of modal symbolic objects, described by probabilistic or histogram-like variables. The prototypes can be similarly described by variables of the same type as the descriptors of the input data. Suitable dissimilarity measures defined between distributions are proposed.

INPUT Variable	Prototype G_i described by	Distance $\psi_j(s, G_i)$
modal (probability distrib.)	modal (probability distrib.)	ϕ^2
histogram-like	histogram-like	χ^2

Table 7: Default parameters for symbolic modal input data

Remark: If the histogram-like symbolic descriptors of the symbolic input data are defined on different domains, a suitable context-dependent dissimilarity measure can be used.

6.4 Partitioning for Mixed Symbolic Data

All the proposed distance functions for p variables are determined by sums of dissimilarities corresponding to the univariate symbolic component descriptors Y_j. The most appropriate dissimilarity functions have been listed in the tables above in dependence of the type of variables.

In practice, however, symbolic data to be clustered are typically described by different types of variables. In order to compute an overall dissimilarity measure two approaches are proposed here:

1) *Weighted linear combination of dissimilarity measures:* If $s = (a, R, \Delta)$ is a symbolic object where $\Delta = (d^1, ..., d^p)$ is the description vector with coordinate values $d^1, ..., d^p$ for the p variables, and if $(G_i^1, ..., G_i^p)$ is the description of the prototype G_i of a class C_i, then the overall dissimilarity between s and G_i is here obtained by a linear combination of the dissimilarity measures computed with respect to the different types of (classical or symbolic) variables:

$$\Psi(s, G_i) = \sum_{k=1}^{K} \lambda_k \sum_{j \in V_k} \xi_j \psi_k(s^j, G_i^j) \tag{4}$$

where:

V_k is the set of variables of type k (k = quantitative, interval, modal, ...)
λ_k is a normalizing factor for the set of variables of type k
ξ_j is a normalizing factor for the j-th variable Y_j.

2) *Categorization (discretization, segmentation, ...)* of the domains of the descriptors Y_j in order to transform all the variables to the same type.

6.5 Remarks:

1 The classical data can be treated as a particular case of the symbolic one.

2 All the methods have a default option.

3 It is not possible to have (strictly) modal symbolic objects in input and Boolean symbolic objects in output (because we do not accept to have an output more complex than the input).

4 There is an option for selecting the most suitable normalized dissimilarity measure for mixed data.

5 There is an option for selecting different systems of weights as coefficients of the linear combination of the univariate dissimilarity measures, whereas the variables are mixed.

6 Introduction of order constraints in SCLUST for categorized quantitative (interval) descriptors in the global strategy.

7 Logical dependence and hierarchical dependence (Mother-Daughter) are not considered in SCLUST.

7 Output Data and SCLUST Results

The SCLUST algorithm produces as output a file .sds containing a new set of symbolic objects: the clusters of the final partition, which are described by a description vector of the corresponding prototypes. We propose to add to the original symbolic data table an indicator variable with the index of the classes of the final partition.

Supplementary results are furnished in a txt file as:

- List of classes (membership list, binary membership vector, class summary),
- Description vector associated to the prototypes,
- The relation for each variable,
- The extension mapping.

References

Albert, P., Brito, P., Bertrand, P., Diday, E., Jacques, G. (1991), Logiciel de Classification Pyramidale, Rapport ILOG, Compte rendu de fin d'étude d'une recherche financée par le MRT.

Baker, F. B. and Hubert, L. J. (1976): Measuring the power of hierarchical cluster analysis. *Journal of the American Statistical Association*, 70, 31–38.

Beale, E. M. L. (1969) Euclidean cluster analysis. *Bulletin of the International Statistical Institute* 43 (2), 92-94.

Bel Mufti, G. (1998) Validation d'une classe par estimation de sa stabilité. *Thèse*, Université Paris-Dauphine, Paris.

Bel Mufti, G., Bertrand P. (2001) Stability analysis of individual clusters. *preprint*.

Bertrand, P. (1986), Etude de la Représentation Pyramidale, Thèse de 3ème Cycle, Univ. Paris-IX Dauphine.

Bock, H.H. (1999) Clustering and neural network approaches. In: W. Gaul, H. Locarek-Junge (eds.): *Classification in the information age.* Springer Verlag, Heidelberg, 42-57.

Bock, H. H., Diday, E. (eds) (2000), *Analysis of Symbolic Data*, Springer Verlag. Heidelberg.

Brito, P. (1991), Analyse de Données Symboliques. Pyramides d'héritage. Thèse, Mathématiques De La Décision, Univ. Paris-IX Dauphine.

Brito, P. (1998), Symbolic Clustering of Probabilistic Data, In : Advances In Data Science And Classification, Eds. Rizzi, A., Vichi, M., Bock, H.-H., Springer-Verlag, Pp. 385-390.

Brito, P., De Carvalho, F. (1998), Symbolic Clustering in the Presence of Hierarchical Rules, in : Proc. of KESDA'98 - Conference, Knowledge Extraction from Statistical Data, Luxemburgo, Abril 98.

Brito, P. (1991), Analyse de Données Symboliques. Pyramides d'héritage. Thèse, Mathématiques de la Décision, Univ. Paris-IX Dauphine.

Brito, P. (1994), Use of Pyramids in Symbolic Data Analysis, in : "New Approaches in Classification and Data Analysis", eds E. Diday et al ., Springer-Verlag

Brito, P. (1998), Symbolic Clustering Of Probabilistic Data, In : Advances In Data Science And Classification, Eds. Rizzi, A., Vichi, M., Bock, H.-H., Springer-Verlag, Pp. 385-390.

Brito, P., De Carvalho, F. (1998), Symbolic Clustering in the Presence of Hierarchical Rules, in : Proc. of KESDA'98 - Conference, Knowledge Extraction from Statistical Data, Luxemburgo, Abril 98.

Brito, P. ; De Carvalho, F., Symbolic Clustering of Constrained Probabilistic Data, in : "Exploratory Data Analysis in Empirical Research", Otto Opitz, Manfred Schwaiger (eds.), volume in Series "Studies in Classification, Data Analysis and Knowledge Organization", Springer Verlag, Heidelberg, 2001 (submitted).

Borg I., Groenen P. (1997). *Modern Multidimensional Scaling – Theory and Applications*, Springer–Verlag, New York.

Calinski, T., Harabasz, J. (1974): A dendrite method for cluster analysis. *Communications in Statistics* 3, 1-27.

Carroll, J.D. (1972). *Individual Differences and Multidimensional Scaling.* in Multidimensional Scaling Theory and Applications in the Behavioral Sciences, vol I, Theory, New York: Seminar Press.

Cazes, P., Chouakria, A., Diday, E., Schektman, Y.: 1997, Extension de l'analyse en composantes principales à des données de type intervalle, *Revue de Statistique Appliquée* **XIV**(3), 5–24.

Celeux, G. , Diday, E. , Govaert, G. , Lechevallier, Y. , Ralambondrainy, H. (1988) - Classification Automatique des Données : Environnement Statistique et Informatique - Dunod, Gauthier-Villards, Paris

Chouakria, A., Verde, R., Diday, E., Cazes, P.: (1996), Généralisation de l'analyse factorielle des correspondances multiple à des objects symboliques. In Proc. *Quatriemes Journées de la Societé Francophone de Classification*, Vannes.

Chouakria, A., Diday, E., Cazes, P.: (1998), An improved factorial representation of symbolic objects, *KESDA'98 27-28 April*, Luxembourg.

Chouakria, A., Diday, E., Cazes, P.: (1998), Vertices Principal Components With an Improved Factorial Representation. In: A.Rizzi, M.Vichi, and H.H. Bock (Eds.): *Advances in Data Science and Classification.* Springer, Heidelberg, 397-402.

Cox T. and Cox M. (1994). *Multidimensional Scaling*, Chapman and Hall,New York.

Csernel, M., De Carvalho, F.A.T. (1999). Usual Operations with Symbolic Data under Normal Symbolic Form. *Applied Stochastic Models in Business and Industry* **15**, 241–257.

D'Ambra, L., Lauro, C. : 1982, Analisi in componenti principali in rapporto a un sottospazio di riferimento, *Rivista di Statistica Applicata* **15**(1), 51–67.

De Carvalho, F.A.T, Verde, R., Lechevallier, Y. (1999): A dynamical clustering of symbolic objects based on a contxt dependent proximity measure. In: H. Bacelar-Nicolau, F.C. Nicolau, J. Janssen (eds.): *Proc. IX International Symposium - ASMDA'99.* LEAD, Univ. de Lisboa, 237–242.

De Carvalho, F.A.T., Souza, R. M. C. (1999). New metrics for constrained Boolean symbolic objects. In: Studies and Research: *Proceedings of the Conference on Knowledge Extraction and Symbolic Data Analysis (KESDA'98).* Office for Official Publications of the European Communities, Luxembourg, 175–187.

Diday, E. (1971): La méthode des Nuées Dynamiques. *Revue de Statistique Appliquée*, 19, 2, 19–34.

Diday, E.: 1987, Introduction á l'approche symbolique en analyse des données, *Journées Symbolique-Numerique*, Université Paris Dauphine.

Diday, E. (1972) Optimisation en classification automatique et reconnaissance des formes. *Revue francaise d'Automatique, Informatique et Recherche Opérationnelle (RAIRO)* **3**, 61.

Duda, R.O, Hart, P.E. (1973): *Pattern Classification and Scene Analysis.* Wiley, New York.

Gettler-Summa, M.(1992): Factorial Axis Interpretation by Symbolic Objects. In *Journées Symbolique-Numerique*. Eds: Diday and Kodratoff, Pinson. Paris.

Gettler-Summa, M., Périnel, E. and Ferraris, J. (1994): New Automatic aid to symbolic cluster interpretation . In New approaches in Classification and data analysis. Springer-Verlag.

Gettler-Summa, M.(1998): MGS in SODAS; marking and generalisation by symbolic objects in the symbolic official data analysis software, Cahier 9935, LISE-CEREMADE Université Dauphine, Paris.

Gettler-Summa, M.(2000): Making and Generalisation by Symbolic Objects in SODAS. In: H. A. L. Kiers, J.-P. Rasson P. J. F. Groenen, M. Schader (eds.): *Data Analysis, Classification, and related methods*. IFCS2000, Namur 417–422.

Goodman, L.A.,Kruskal, W.H. (1954). Measures of association for cross-classifications. *Journal of the American Statistical Association* 49, 732-764.

Gordon, A.D. (1998). How many clusters? An investigation of five procedures for detecting nested cluster structure. In: *Proceedings of the IFCS-96 Conference*. Kobe, 109-116.

Gordon, A. D. (1994): Identifying genuine clusters in a classification. *Computational Statistics and Data Analysis*, **18**, 561–581.

Gordon, A. D. (1996): Null models in cluster validation. In: W. Gaul and D. Pfeifer (Eds.): *From Data to Knowledge: Theoretical and Practical Aspects of Classification, Data Analysis, and Knowledge Organization*. Springer, Berlin, 32–44.

Gower, J. C. (1966) Some distances properties of latent root and vector methods using multivariate analysis. *Biometrika*, 53, 325–338.

Hardy, A. (1994): An examination of procedures for determining the number of clusters in a data set. In: E. Diday, Y. Lechevallier, M. Schader, P. Bertrand, B. Burtschy (Eds): New approaches in classification and data analysis. Springer Verlag, Berlin, 178-185.

Hardy, A. (1996): On the number of clusters. *Computational Statistics & Data Analysis*, 23, 83–96.

Hardy, A.,Andre, P. (1998): An investigation of nine procedures for detecting the structure in a data set. In: A. Rizzi, M. Vichi, H.-H. Bock (Eds): *Proceedings of the 6th Conference of the International Federation of Classification Societies*. Springer-Verlag, Berlin, 29-36.

Hardy, A., Lallemand, P. (2001): Application du test des Hypervolumes à des données symboliques de type intervalle. *Extraction des connaissances et apprentissage*, 1, 4, 287–292

Hardy, A., Lallemand, P. (2002): Determination of the number of clusters for symbolic objects described by interval variables. *Classification, Clustering and Data Analysis*, Springer Verlag, 311–318 Hardy, A., Lallemand, P. (2003): Trois approches pour la modélisation des données symboliques de type intervalle. *Méthodes et perspectives en Classification*, Presses Académiques de Neuchâtel, 137–140

Huber, L. J., Levin, J. R.: 1976, A general statistical framework for assessing categorical clustering in free recall. *Psychological Bulletin*, 83, 1072–1080.

Lauro, C., Verde, R., Palumbo, F.: 2000, Factorial Discriminant Analysis on Symbolic Objects, in Bock, H.H. and Diday, E. (eds): 1999, *Analysis of Symbolic Data*, Springer Verlag. Heidelberg.

Lauro, C., Palumbo, F.: 2000, Factorial Methods with cohesion constraints on Symbolic Objects. In Atti di IFCS2000, Springer-Verlag, .

Lauro, C., Palumbo, F.: 1998, New approaches to principal components analysis to interval data, *International Seminar on New Techniques and Technologies for Statistics, NTTS'98, 4/6 nov. 1998*, Sorrento, Italy.

Lauro, C., Palumbo, F.: 2000, Principal Component Analysis of Interval Data: a Symbolic Data Analysis Approach, Computational Statistics (forthcoming).

Lebart, L., Morineau, A. and Piron, M.: 1995, *Statistique exploratorie multidimensionelle*, Dunod, Paris.

Lechevallier, Y. (1974): *Optimisation de quelques critères en classification automatique et application à l'étude des modifications des protéines sériques en pathologie clinique.* Thèse de l'université Paris VI.

Milligan, G.W.,Cooper, M.C. (1985): An examination of procedures for determining the number of clusters in a data set. *Psychometrika* 50, 159-179.

Palumbo F. and Gettler-Summa, M.(2000): Interpretation of Symbolic Principal Componant Analysis on Interval Data.SIS2000, Florence.

Torgenson, W. S. (1952) Multidimensional scaling: 1 Theory and method, *Psychometrika*, 17, 401–419.

Torgenson, W. S. (1958) *Theory and methods of scaling.* New York: Wiley

Verde, R., (1997): Symbolic object decomposition by factorial techniques. *Franco-Indian Meeting*, LISE-CEREMADE, Université Paris IX Dauphine.

Verde, R. and De Angelis, P. (1997): Symbolic objects recognition on a factorial plan, *NGUS'97*, Bilbao Spain.

Verde, R., Giordano, G. and Gettler-Summa, M. (1997): Symbolic objects for multiattribute preference data, *OSDA'97*, Darmstadt, Germany.

Verde, R. and Lauro, C. (1993): Non symmetrical data analysis of multiway fuzzy coded matrices, *ISI*, Firenze.

Verde, R. (1998): Generalised Canonical Analysis on symbolic objects, In: M. Vichi, and O. Opitz (Eds.): *Classification and Data Analysis*, Springer Verlag, Heidelberg, 195–202.

Verde, R., De Carvalho, F.A.T. (1999): Dependence rules influence on factorial representation of Boolean symbolic objects. *Proceedings of the Conference on Knowledge Extraction and Symbolic Data Analysis (KESDA'98)*. Office for Official Publications of the European Communities, Luxembourg, 287-300.

Verde, R., De Carvalho, F.A.T, Lechevallier, Y. (2000): A Dynamical Clustering Algorithm for Multi-nominal Data. In: H. A. L. Kiers, J.-P. Rasson P. J. F. Groenen, M. Schader (eds.): *Data Analysis, Classification, and related methods*. IFCS2000, Namur 387–394.

Verde, R., De Carvalho, F.A.T, Lechevallier, Y. (2001): A dynamical clustering algorithm for symbolic data. Tutorial "Symbolic Data Analysis", GfKl conference, Munich.

Winsberg, S., Desoete, G. (1997) Multidimensional scaling with constrained dimensions: CONSCAL, *British Journal of Mathematical and Statistical Psychology*, 50, 55-72.

INFORMATION SOCIETY TECHNOLOGIES
PROGRAMME

SYKSOM User Manual

Kohonen Self-Organizing Map

Edited by RWTH

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

ASSO Project - Workpackage WP6.2 - Module SYKSOM
User Manual: The ASSO Module SYKSOM
(Slightly revised version from April 4, 2004

Visualizing symbolic data tables by Kohonen maps: The SODAS module SYKSOM

Hans-Hermann Bock
University of Aachen, Germany

1. Introduction
2. The general idea underlying a Kohonen map
   2.1 The linear projection paradigm in PCA and SPCA
   2.2 The Kohonen approach for symbolic data
3. Building a Kohonen map with SYKSOM
4. The output of the SYKSOM module
5. Visualization and interpretation of the Kohonen map
   5.1 The VMAP module and the Kohonen map
   5.2 A general strategy for visualizing Kohonen maps
   5.3 The VPLOT display
6. Summary of the benefits from Kohonen maps
7. How is the SYKSOM algorithm working?

1. Introduction

Visualizing data in the form of illustrative diagrams and searching, in these diagrams, for structures, clusters, trends, dependencies or anomalies (e.g., outliers) is one of the main motives and strategies in exploratory statistics, data analysis and data mining. In the case of symbolic data, powerful methods are provided by suitable 'symbolic' adaptations of classical methods such as Symbolic Principal Component Analysis, Symbolic Generalized Canonical (or Factor) Analysis, etc. that produce, for example, two-dimensional displays in the space of the first and second 'factors' (see the SODAS software modules SPCA and SGCA).

An alternative visualization of symbolic data is obtained by constructing a *Kohonen map* with the SODAS module SYKSOM (for the case of interval-type data): Instead of displaying the individual items $k = 1, ..., n$ (data points, data rectangles) by n points or rectangles in a two-dimensional 'factor space' as in SPCA (see Fig. 1), the n items are first clustered into a (quite smaller) number m of 'mini-clusters' (this is essentially a data reduction step) and then these mini-clusters are assigned to (represented by) by the vertices of a rectangular lattice $\mathcal{L}$ of points in the plane such that 'similar' clusters (in the original data space) are represented by neighbouring vertices in the lattice $\mathcal{L}$ (see Fig. 2 and Fig. 9).

In order to visualize the properties of the clusters, the SYKSOM module determines, during the cluster construction process, suitable cluster representatives or prototypes: These prototypes can be easily visualized, in a third step, by clicking with the mouse at the corresponding vertex of the screen, or by displaying all cluster prototypes simultaneously (see Fig. 4, and 10 to 11).

In the SODAS software Kohonen maps are constructed by the module SYKSOM that assumes a data set with n items or individuals that are described by p interval variables. The visualization of clusters is obtained by the modules VPLOT and VMAP (accessible directly from the result output of the chaining), and by other SODAS modules for displaying the properties of clusters (including also, e.g., additional categorical variables).

Two remarks might be in order: First, it should be emphasized that the construction of a Kohonen map is particularly well suited for the analysis of a large number n of items. In fact, the implicitly involved clustering process reduces the large number n to a smaller number m of clusters (or vertices) whose properties can be much more easily visualized than those of the original number n of data where SPCA or SGCA might yield unwieldy displays which are difficult to interpret. Second, we will show below that in the Kohonen approach both clustering and prototype construction are conducted in a simultaneous process where the data are entered in a sequential way and current results (clusters, prototypes) are iteratively modified and updated by incorporating a new (the subsequent) data item. Insofar the procedure can be termed 'self-organizing' which explains the acronym SYKSOM that combines the SYmbolic context with the classical notation 'Kohonen's Self-Organizing Map'

In the following section 2 we describe the basic idea that underlies the Kohonen and SYKSOM approach, then specify in section 3 SYKSOM's input data and describe how this module is run for a set of interval-type data. Section 4 may be considered the most important one since it shows how to visualize the constructed clusters and their prototypes. Finally, section 5 provides an idea on how the sequential SOM algorithm works. A detailed description will be given in the ASSO scientific report and the monograph to be published with Dekker publishers.

2. The general idea underlying a Kohonen map

2.1 The linear projection paradigm in PCA and SPCA

A well-known method for visualizing data is provided by the classical *principal component analysis* (PCA): This method starts from n p-dimensional data vectors $x_1, ..., x_n \in \mathbb{R}^p$ which represent n sampled objects $k = 1, ..., n$ (individuals, items,...) such that the p components of the vector $x_k = (x_{k1}, ..., x_{kp})'$ are the values of p quantitative variables observed for the k-th object. Basically, PCA selects, as illustrated in Fig. 1, in the (usually high-dimensional) Euclidean space $\mathbb{R}^p$ an appropriate low-dimensional (linear) hyperplane H with a given low dimension s (typically $s = 2$) and projects the data points $x_1, ..., x_n$ onto this hyperplane H. The resulting configuration of projected points $y_1, ..., y_n \in H$ is then displayed on a screen and can reveal interesting features or structures in the set of objects or data points, in particular clusterings (see Fig. 1) and trends. A suitable generalization to symbolic interval-type data is described in chapter 9 of Bock & Diday (2000)[1] and implemented by the module SPCA of the SODAS software.

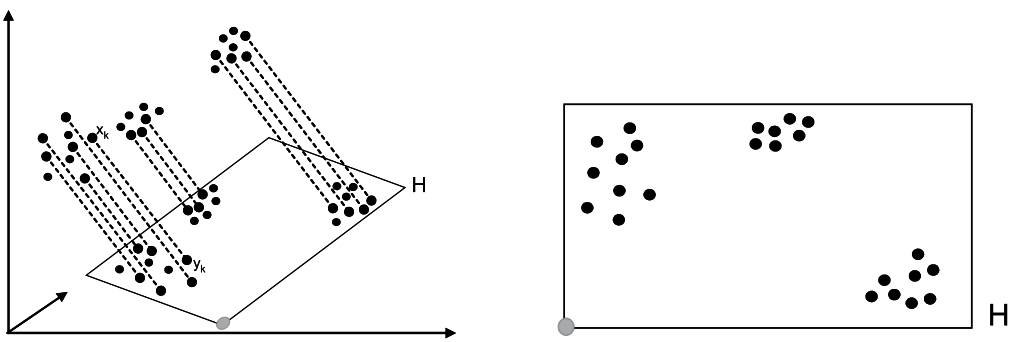


Fig. 1: PCA of n data points $x_k \in \mathbb{R}^3$, with two-dimensional display on the screen.

Both PCA and SPCA are oriented towards a situation where the data $x_1, ..., x_n$ are essentially concentrated in the neighbourhood of an s-dimensional *linear* manifold of $\mathbb{R}^p$. However, in practice this linearity assumption is rarely fulfilled, since the cloud of data points is often concentrated near a *nonlinear* manifold F of $\mathbb{R}^p$ which is bent down or

[1]Bock, H.-H., and Diday, E.: Analysis of symbolic data. Exploratory methods for extracting statistical information from complex data. Springer Verlag, Heidelberg, 2000, pp 425.

back (see Fig. 2). Then PCA might return an erroneous, artificial, or non-interpretable configuration, possibly a methodological artifact.

2.2 The Kohonen approach for symbolic data

In order to overcome this difficulty, Kohonen (1982, 1995) proposed an alternative visual representation which does not use a Euclidean space (i.e., a hyperplane H in $\mathbb{R}^p$), but represents the data (more specifically: the data clusters) as the vertices $P_1, ..., P_m$ of a *rectangular* $b \times a$ *lattice* $\mathcal{L}$ with a specified number b, a of rows and columns, respectively, such that each vertex P_i of this lattice represents a homogeneous *cluster* C_i of objects and also the *prototype* z_i for this cluster.

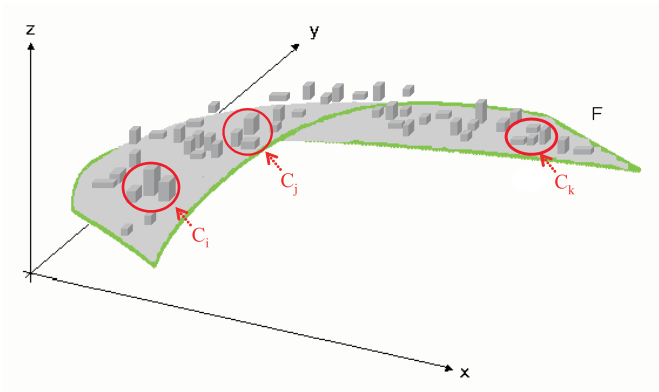
Whereas the classical Kohonen approach is designed for data *points*, the SYKSOM module provides an adaptation to the case of symbolic interval-type data, i.e. a set of n interval-type data vectors $x_1, ..., x_n \subset \mathbb{R}^p$ such that the k-th object is described by a vector $x_k = ([a_{k1}, b_{k1}], ..., [a_{kp}, b_{kp}])$ of one-dimensional intervals and geometrically represented by a hypercube $Q_k = [a_k, b_k]$ of $\mathbb{R}^p$ (also termed a p-dimensional interval, a hyper-rectangle etc.; see Fig. 3). Table 1 provides an example of an interval-type data matrix for $n = 8$ items with $p = 3$ interval-type variables where x_k is given by the k-th row, and Fig. 2 illustrates a case where the data hypercubes $Q_1, ..., Q_n$ are concentrated near a bent hyper-surface F such that SPCA would be inappropriate.

$k \setminus j$	Var 1	Var 2	Var 3
1	$[8.5, 10.0]$	$[13.0, 15.2]$	$[5.0, 8.2]$
2	$[6.3, 9.1]$	$[14.1, 16.0]$	$[6.3, 7.2]$
3	$[7.9, 11.8]$	$[11.6, 13.5]$	$[4.9, 6.5]$
4	$[9.0, 11.0]$	$[10.9, 12.5]$	$[7.1, 8.1]$
5	$[6.3, 7.2]$	$[12.9, 15.0]$	$[6.2, 7.4]$
6	$[7.1, 7.9]$	$[11.5, 12.9]$	$[4.8, 5.7]$
7	$[7.5, 9.4]$	$[13.2, 15.0]$	$[6.6, 8.1]$
8	$[6.6, 7.8]$	$[12.4, 13.2]$	$[5.7, 7.2]$

Tab. 1: *A data table for $n = 8$ items (cities $k = 1, ..., 8$) and $p = 3$ interval-type variables (price ranges in 1000 Euro for three car brands $j = 1, 2, 3$)*

The symbolic Kohonen approach, implemented by the ASSO module SYKSOM, combines essentially the following steps:

(1) The n data hypercubes are clustered into m non-overlapping clusters $C_1, ..., C_m \subset \{1, ..., n\}$ of (items with) 'similarly located' hypercubes (where $m = b \cdot a$ is the number of vertices of the selected $b \times a$ lattice $\mathcal{L}$). This is essentially a data aggregation step where the size of the constructed clusters is typically small in relation to the total number n of individuals. Therefore we will call these clusters 'mini-clusters' in the sequel.



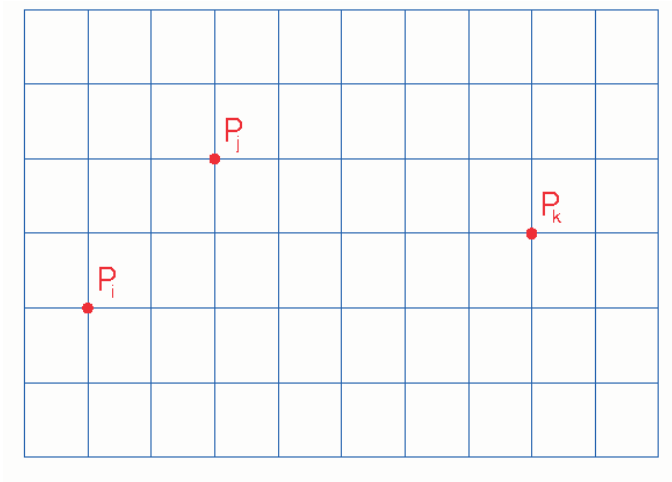


Fig. 2: *A set of hypercubes Q_k in R^3 concentrated near a nonlinear manifold F with clusters C_i to be displayed by the $m = b \cdot a = 7 \cdot 11 = 77$ vertices P_i in the rectangular lattice $\mathcal{L}$ in a topologically correct form.*

(2) Each mini-cluster C_i is characterized by a *prototype hypercube* z_i of $\mathbb{R}^p$ (as a class representative).

(3) Each mini-cluster C_i, and thereby each prototype z_i, is assigned to a vertex P_{ν_i} of the lattice $\mathcal{L}$ (with $\nu_i \in \{1, ..., n\}$ for all i).

(4) This assignment process is conducted in a way such that any two prototypes z_i, z_j (clusters C_i and C_j) which are *neighbouring* <u>*in* $\mathbb{R}^p$</u> (i.e., which have a small dissimilarity or distance from each other) are typically assigned to two vertices P_{ν_i}, P_{ν_j} of $\mathcal{L}$ which are *neighbouring* <u>*in the lattice* $\mathcal{L}$</u> (i.e., such that the 'path distance' between the vertices P_{ν_i} and P_{ν_j} along the edges of $\mathcal{L}$ is small)[2].

[2]The SYKSOM algorithm uses a labelling of clusters and vertices such that C_i and z_i are assigned to the vertex P_i (i.e., with $\nu_i \equiv i$). Therefore we use this labelling throughout this text.

In this way, the SYKSOM algorithm builds a partition $(C_1, ..., C_m)$ of objects and determines, for each mini-cluster C_i, a 'typical' hypercube z_i that is called the 'prototype' of C_i.

It is expected that, in the end, the lattice $\mathcal{L}$ together with the m mini-clusters and cluster prototypes provides an illustrative 'flat representation' of the (unknown) bent and twisted surface F which is visualized by the Kohonen map and may be interpreted by using the graphical modules VMAP and VIEW (see below).

3. Building a Kohonen map with SYKSOM

SYKSOM starts from a $n \times p$ table of interval data which describe the behaviour of n objects $k = 1, ..., n$ (persons, items,...) with regard to p interval-type variables $j = 1, ..., p$. A small-sized example for $n = 8$ cities (items) and $p = 3$ variables (the range of prices for three car brands $j = 1, 2, 3$) is given in Table 1. Let $x_{kj} = [a_{kj}, b_{kj}]$ denote the interval which forms the entry of the cell (k, j) of this table. Typically, this interval measures the range of observed values of the j-th variable for individuals from item k, e.g., in Table 1 the minimum price a_{kj} and the maximum price b_{kj} of a car of brand j in the city k.

Considering all p variables (columns) simultaneously, the item k is described by the row k of the table, i.e. the vector of intervals

$$x_k = \begin{pmatrix} [a_{k1}, b_{k1}] \\ \vdots \\ [a_{kp}, b_{kp}] \end{pmatrix}, \qquad \text{for example:} \qquad x_4 = \begin{pmatrix} [\ 9.0, 11.0] \\ [10.9, 12.5] \\ [\ 7.1,\ \ 8.1] \end{pmatrix} \tag{1}$$

or, equivalently, by the rectangle (hypercube)

$$Q_k = [a_k, b_k] \subset \mathbb{R}^p$$

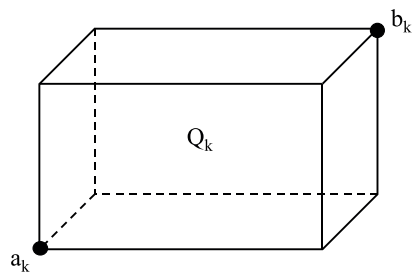


Fig. 3: *The hyperrectangle $Q_k = [a_k, b_k]$ representing item k*

in the p-dimensional space $\mathbb{R}^p$ with sides $[a_{kj}, b_{kj}]$ for $j = 1, ..., p$ where

$$a_k = \begin{pmatrix} a_{k1} \\ \vdots \\ a_{kp} \end{pmatrix} \qquad \text{and} \qquad b_k = \begin{pmatrix} b_{k1} \\ \vdots \\ b_{kp} \end{pmatrix} \tag{2}$$

are the „left lower vertex“ and the „right upper vertex“ of Q_k (see Fig. 3). We will identify the interval-type data *vector* x_k with the corresponding hypercube $Q_k \in \mathbb{R}^p$. Thus, our data table corresponds to a cloud of n rectangles in $\mathbb{R}^p$ which may be more or less clustered as in Fig. 4 or arranged near a manifold F as in Fig. 2.

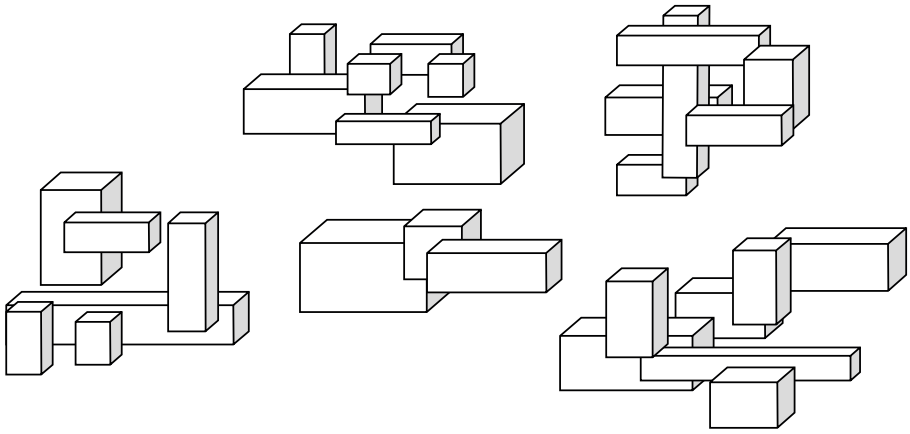


Fig. 4: $n = 29$ *items* $k = 1, ..., 29$ *(rows of the data table) represented as 29 rectangles* Q_k *in the p-dimensional space (with* $p = 3$*), here with an obvious clustering structure*

Remark 3.1:
SYKSOM is able to process also p-dimensional quantitative *single-valued* data, just by converting a single data point $x \in \mathbb{R}^p$ into a p-dimensional one-point interval $[x, x]$. The corresponding rectangle Q_k then has the volume 0.

In order to produce a Kohonen map for a given interval-type data matrix, the SYKSOM module of the SODAS software has to build a chaining as follows (see Fig. 5):

• First, the data are entered in the form of a SODAS data file with the extension `.sds` or `.xml` (see the corresponding SODAS modules).

• Then the SYKSOM module has to be inserted into the chaining from the list of methods provided under the heading `Clustering`.

• Now we have to specify the 'parameters' (clicking the right mouse button). First we select, in the `Variables` window, the variables that should be used for the Kohonen map. Only quantitative (single-point or interval-type) variables are possible here.

Then we specify, in the `Parameters` specification window displayed in Fig. 6, the methods, parameters, distance measures etc. that should be used for the cluster construction process. The available options are explained below. A detailed and precise explanation of the methods and parameters is given in the scientific report and the forthcoming monograph on the SODAS software.

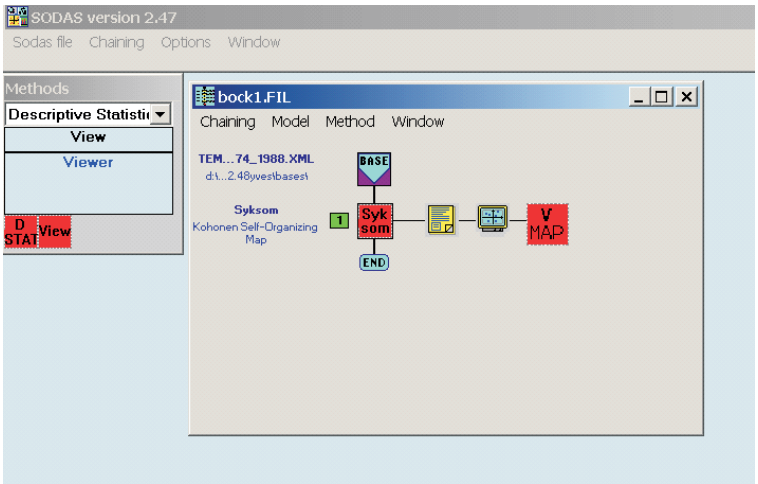


Fig. 5: *The chaining for the SYKSOM module*

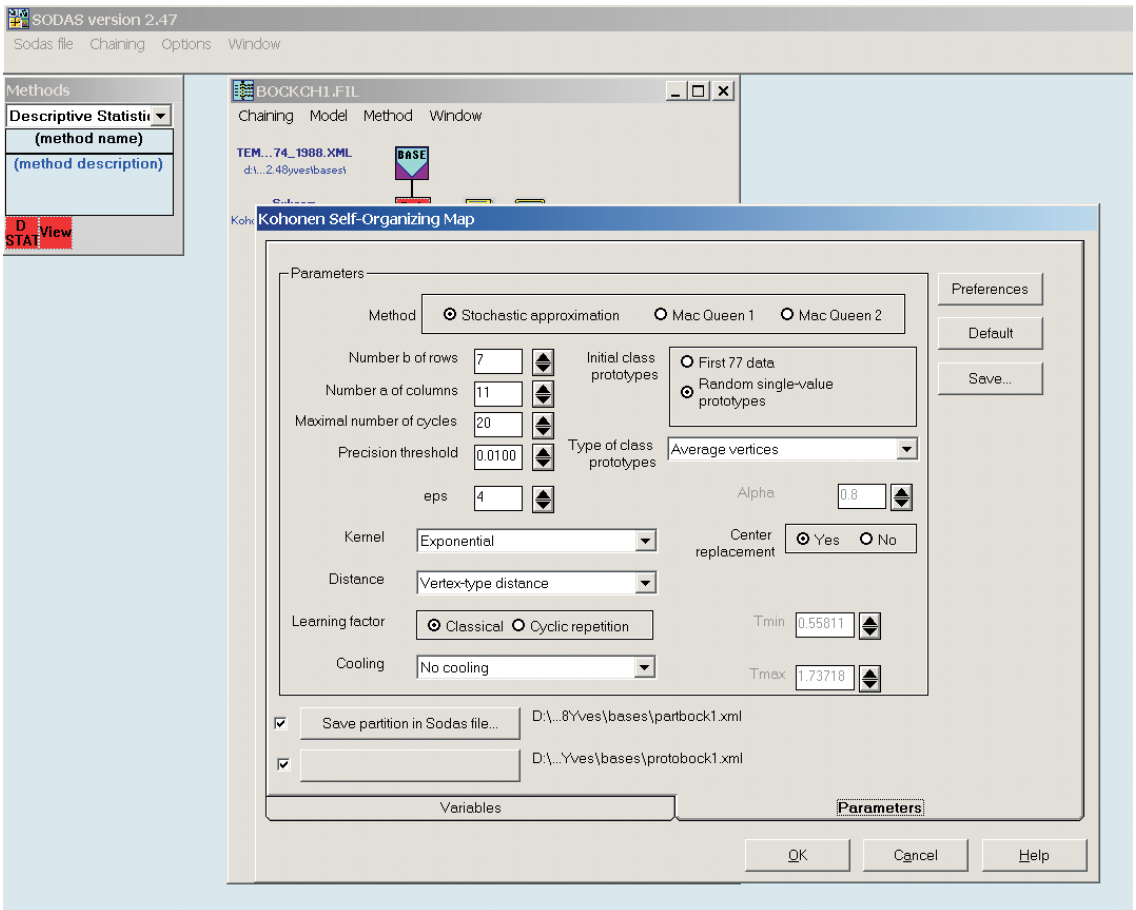


Fig. 6: *The parameter specification window of SYKSOM*

The method:
The module SYKSOM provides three different clustering methods that are called *StochApprox* (stochastic approximation), *MacQueen1* and *MacQueen2*. The selection of a method has some influence on other parameters, in particular, on the list of parameters to select. The method *StochApprox* is the direct generalization of Kohonen's classical SOM algorithm and should typically be used. A brief description is given in section 7 below[3].

The lattice:
The user has to choose the number of rows b and columns a of the lattice $\mathcal{L}$. Note that it makes no sense to introduce too small clusters. A good rule might be that $n/m \geq 5$ such that clusters have at least 5 members on the average.

Type of class prototypes:
The prototype of each cluster C_i will be a hypercube z_i in $\mathbb{R}^p$ that is determined from the data hypercubes that belong to the cluster. There are three options: 'Average vertices', 'envelope-type' and 'truncated envelope-type' (this latter choice needs a percentage α of data hypercubes covered by z_i).

Initialization:
The algorithm starts with $m = b \cdot a$ suitable cluster prototypes (hypercubes). The user can choose either the first m data rectangles (this choice might introduce some bias) or - also as the default - a set of m randomly determined singleton prototypes (points from $\mathbb{R}^p$ that will be blown up to real hypercubes with positive volume in the course of the algorithm).

Iteration parameters:
All n data hypercubes are cyclically processed several times. The iteration is stopped if the clustering has obtained an almost stationary state (controlled by the precision threshold) or if a maximum number of cycles was run.

Distance measures and neighbourhoods:
The algorithm assigns a new data hypercube x_k to the 'most similar' cluster prototype z_{i^*}. The dissimilarity between hypercubes can be measured in three ways: vertex-type distance, Hausdorff-type L_1 distance, and Hausdorff-type L_2 distance. The cluster prototypes z_j in the neigbourhood (in $\mathcal{L}$!) of the 'most similar' prototype z_{i^*} are updated in each turn, at least if the path distance between P_j and P_{i^*} (in $\mathcal{L}$!) is not larger than the parameter ϵ. The amount of updating is determined by this path distance $\delta(P_j, P_{i^*})$ and a decreasing 'kernel function' K (it is proportional to $K(\delta(P_j, P_{i^*}))$, a sequence of 'learning factors' (that can be repeated after each cycle of n steps), and a cooling parameter T that shrinks the neigbourhood of vertices (in $\mathcal{L}$) in the course of the algorithm (the parameters T_{min} and T_{max} are automatically calculated). There are four options for K: two threshold versions, a Gaussian, and an Exponential kernel.

[3]The algorithms MacQueen1 and MacQueen2 generalize a classical sequential clustering method by MacQueen and resolve slightly modified problems. Their use needs a deeper understanding of the underlying approach, the interpretation may be difficult.

Saving the output:
After the end of the algorithm, the resulting mini-clusters $C_1, ..., C_m$ (an m-partition of $\{1, ..., n\}$) and the class prototypes $z_1, ..., z_m$ can be saved in special files if the `Save partition as SODAS file` and `Save prototypes as SODAS file` options are ticked and suitable file names are chosen (after clicking twice).

4. The output of the SYKSOM module

As mentioned before, the SYKSOM algorithm produces an m-partition $\mathcal{C} = (C_1, ..., C_m)$ of classes (mini-clusters) C_i from the set of items $\{1, ..., n\}$, and for each mini-cluster C_i a prototype hypercube $z_i = [u_i, v_i]$ with a 'lower left' vertex u_i and a 'upper right' vertex $v_i \in \mathbb{R}^p$. The classes and prototypes are assigned to the $m = b \cdot a$ vertices of the lattice $\mathcal{L}$ in a (hopefully) 'topologically correct' way, i.e. such that the large-scale configuration of the n given rectangles (and: of the m prototype hypercubes z_i) in $\mathbb{R}^p$ is approximately reproduced by the two-dimensional display in $\mathcal{L}$. More specifically, the SYKSOM module provides the following output files:

1. A list with the member lists of all m mini-clusters $C_1, ..., C_m \subset \{1, ..., n\}$ and all cluster prototypes (hypercubes). This information is available by clicking the left-most (yellow) box in the 'result' part of the chaining. It is stored as a file with extension `.lst` in the sub-directory `...\filieres`.

2. A list of the n original data (i.e., including eventually existing categorical, multimodal,... variables as well and not only the interval variables which were selected for the construction of the map) together with a new, additional categorical variable that specifies, for each individual, the index $i \in \{1, ..., m\}$ of the cluster C_i to which this individual has been assigned. This information can be saved as a `.xml` file in the subdirectory `...\bases` by ticking (and clicking twice) `Save partition as SODAS file` in the SYKSOM `Parameter specification window` before running SYKSOM. This file can be used, in an additional chaining, as input for many other ASSO modules, e.g., when visualizing cluster-specific zoom stars with VIEW which include all variables and not only the interval variables (as it is the case for the VMAP module, see below). Or by using CLINT for this file.

3. A list with all m cluster prototypes (hypercubes) $z_i = [u_i, v_i]$ (i.e., the lower and upper vertices $u_i, v_i \in \mathbb{R}^p$ of z_i in terms of p coordinate-wise intervals $[u_{ij}, v_{ij}]$, $j = 1, ..., p$) can be saved in a `.xml` file in the sub-directory `...\bases` by ticking (and clicking twice) `Save prototype in SODAS file` in the SYKSOM `Parameter specification window` before running SYKSOM. This file can be used as input (in another chaining) for VIEW, DIV, HIPYR etc. where Zoom Star visualizations or a hierarchical/pyramidal classifications of the prototypes are obtained.

4. A list that is stored as a `.vm0` file in the sub-directory `...\filieres` and contains several parameters and results (clusters, prototypes, scalings,...). This file is the input to the visualization tool VMAP (see section 5.2).

5. By clicking the right-most icon in the chaining (Fig. 5) we obtain the desired Kohonen map in the form of a rectangular lattice of cells (cases) with b rows and a columns representing the $b \cdot a$ vertices of the lattice $\mathcal{L}$ (see Fig. 7)[4]. In this display, each cell represents a mini-cluster and, by construction, neighbouring cells should imply similar class prototypes. This display is the basis for the statistical and exploratory analysis of the given data and of the constructed mini-clusters by the module VMAP. This will be explained in the next section.

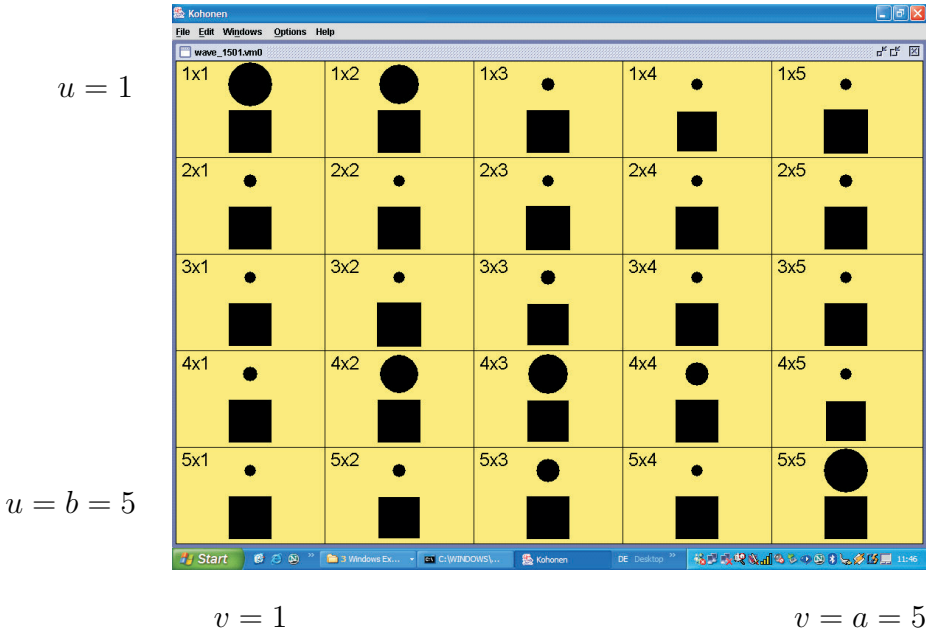


Fig. 7: *Display of a 5×5 lattice $\mathcal{L}$ (the 25 cells of the display correspond to the 25 vertices of $\mathcal{L}$). Circles represent the cluster size $|C_i|$, squares the geometric mean of the p side lengths of the prototype rectangles.*

Remark 4.1:
Our notation is chosen such that for all indices $i = 1, ..., m = b \cdot a$ the mini-cluster C_i is assigned to the vertex P_i. The position of a vertex P_i in the lattice is often specified in terms of the integer lattice coordinates (v, u) or $v \times u$ in $\mathcal{L}$ (for $v = 1, ..., b, u = 1, ..., a$), with P_1 in the upper left corner. Thus we use the following notation scheme:

$$\begin{array}{llllllll}
P_1 & \hat{=} (1,1) & P_2 & \hat{=} (1,2) & \cdots & P_a & \hat{=} & (1,a) \\
P_{a+1} & \hat{=} (2,1) & P_{a+2} & \hat{=} (2,2) & \cdots & P_{2a} & \hat{=} & (2,a) \\
\vdots & \vdots & \vdots & \vdots & & \vdots & & \\
P_{a(b-1)+1} & \hat{=} (b,1) & P_{a(b-1)+2} & \hat{=} (b,2) & \cdots & P_m & \hat{=} & \hat{=}(b,a)
\end{array}$$

This is visualized in Fig. 8 below for the case of a 6×8 lattice.

[4] For typographical reasons, each cell (box) (v, u) of this display corresponds to the vertex (v, u) of the lattice $\mathcal{L}$.

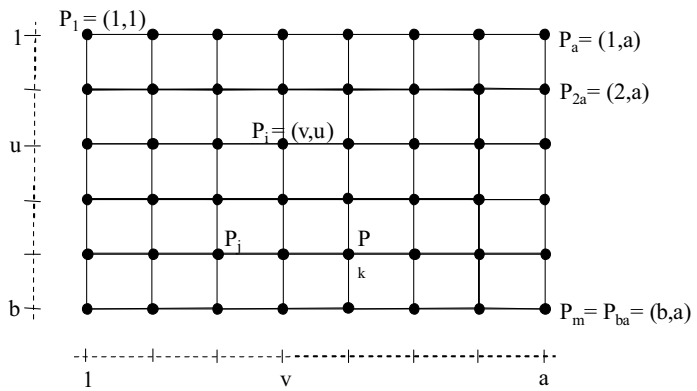


Fig. 8: *Coordinates in a rectangular lattice* $\mathcal{L}$ *with* $b = 6$ *rows,* $a = 8$ *columns and* $m = b \cdot a = 6 \cdot 8 = 48$ *vertices corresponding to* 48 *mini-clusters.*

5. Visualization and interpretation of the Kohonen map

Neither the (mini-) clusters $C_1, ..., C_m$ nor the class prototypes $z_1, ..., z_m$ constructed by the algorithm SYKSOM can be directly seen in the lattice $\mathcal{L}$. However, the usefulness and attractiveness of the Kohonen approach resides in the fact that there exist various tools for visualizing the mini-clusters, their properties, and their similarities:

(1) the VMAP module (by double-clicking the VMAP icon in the chaining),

(2) the VPLOT module (by clicking the middle icon in the chaining), and

(3) the module VIEW (referring to the saved partition and prototype files).

5.1 The VMAP module and the Kohonen map

The VMAP module displays the Kohonen map in the form of Fig. 7. Each cell (vertex of $\mathcal{L}$) corresponds to a mini-cluster C_i and contains two icons, a circle and a square.

– The circle indicates the size $|C_i|$ of a cluster C_i (with area proportional to $|C_i|$).

– The square displays the volume of the corresponding prototype hypercube $z_i = [u_i, v_i]$: its area is proportional to the geometric mean $[\prod_{j=1}^{p}(v_{ij} - u_{ij})]^{1/p}$ of the p side lengths $v_{ij} - u_{ij}$ $(j = 1, ..., p)$ of the rectangle $z_i = [u_i, v_i]$. Up to the exponent $1/p$, this is the volume of z_i (standardization is such that the maximum volume is 1).

Moreover, by clicking a cell with the *left* mouse button yields the member list of the corresponding class (together with the class size and the volume of z_i). When clicking a cell with the *right* mouse button, we obtain a detailed description of the corresponding class prototype either in the form of a Zoom Star (`Polaire`) or a *bar diagram*(`Parallèle`). This latter one displays, for the selected mini-cluster C_i, the p sides $[u_{ij}, v_{ij}]$ of the corresponding prototype rectangle $z_i = [u_i, v_i] = ([u_{i1}, v_{i1}], ..., u_{ip}, v_{ip}])$ in dependence on the variables coded by $j = 1, ..., p$ (see Fig. 9).

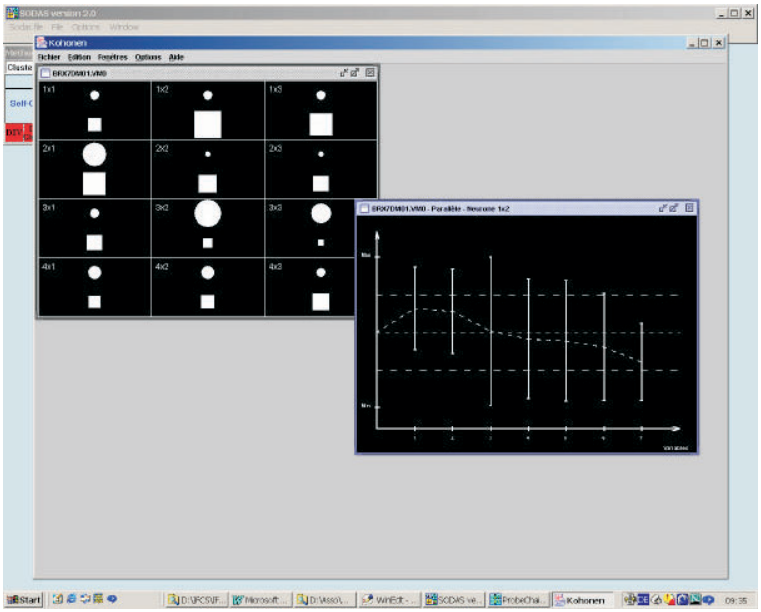


Fig. 9: *A Kohonen map of type* 4×3 *with the bar diagram of the mini-cluster* C_2 *represented by the vertex* $P_2 \hat{=} 1 \times 2$.

The option `Proximité` provides a (smoothed) version of the map where the dissimilarity between all classes to the selected class are displayed by the grey level. The option `Toutes` displays all three displays.

The user can also generate these displays for two or several clusters simultaneously. This option provides the possibility to compare easily the properties of different clusters.

Remark:
Note that in these displays the cells are marked with 'neuron' instead of 'cluster' or 'class', following a terminology broadly used in informatics. Also note that VMAP uses the result file `.vm0` stored in a subdirectory ...\`filieres` and needs a suitable version of Java software (currently we use the version *j2re-1_4_1_05-windows-i586-i.exe*).

5.2 A general strategy for visualizing Kohonen maps

The previously described VMAP module provides a class description for a single or some few mini-cluster(s) C_i. However, the user can also display the properties of all m classes simultaneously just by utilizing the SODAS module VIEW for the `.xml` prototype file specified and saved in the `Parameter specification window`.

More generally, when describing the properties of all clusters simultaneously in one single display, we obtain what is called a *Kohonen map (in the strong sense)* or a *semantic map*. A simple, but illustrative example is given by Fig. 10 where the data originate from a sociological survey and we consider, in order to describe the $m = 66$ mini-clusters, the

interval-type variable 'income': The five labels a, b, c, d, e indicate the ordinal income level that prevails in the corresponding mini-cluster (a means 'low income', e means 'huge income'). So we see that, e.g., low incomes occur in two large clusters of mini-clusters (two 'super-clusters'), North-East and West, say. Moreover, there is an outlying mini-cluster of type a near the South-West corner, surrounded by mini-clusters from the b category.

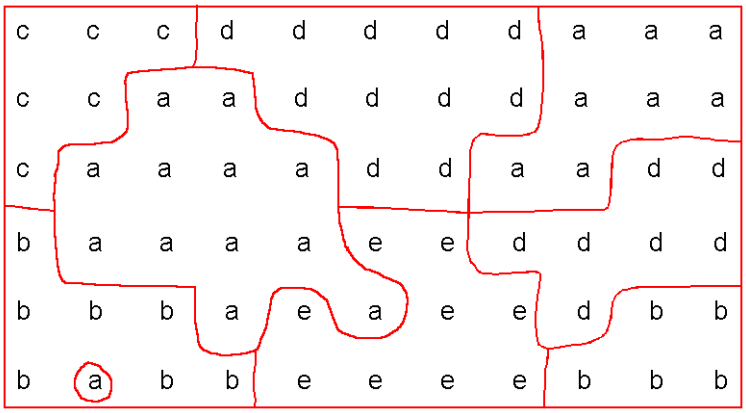


Fig. 10: *A Kohonen map where the 66 mini-clusters (vertices) have been marked with labels $a, b, ..., e$ which describe specific properties of the clusters (here: the variable 'income' with levels: a = 'low', ..., e = 'huge') such that various homogeneous regions or 'super-clusters' can be recognized in the map where, e.g., 'low income' (= a) prevails.*

More generally, we can mark each cluster C_i (cell, vertex P_i) of the lattice $\mathcal{L}$ with a label, icon, letter, zoom star, profile etc. which describes the properties of this cluster (see Fig. 11): This provides a visual representation of cluster properties where, by construction, neighbouring vertices should exhibit similar cluster descriptions or labels.

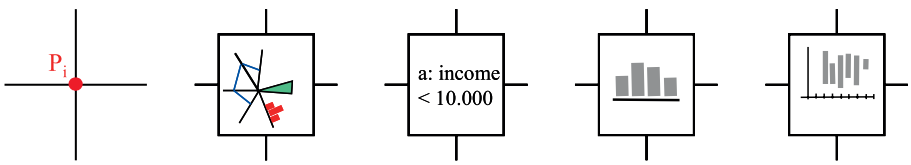


Fig. 11: *Typical icons, codes and diagrams that are used for displaying specific properties of the clusters.*

For illustration, Fig. 12 and Fig. 13 have been generated in this way (by using the options provided by VIEW). They display the class properties for interval-type data, one in the form of bar diagrams, the other one in the form of Zoom Stars.

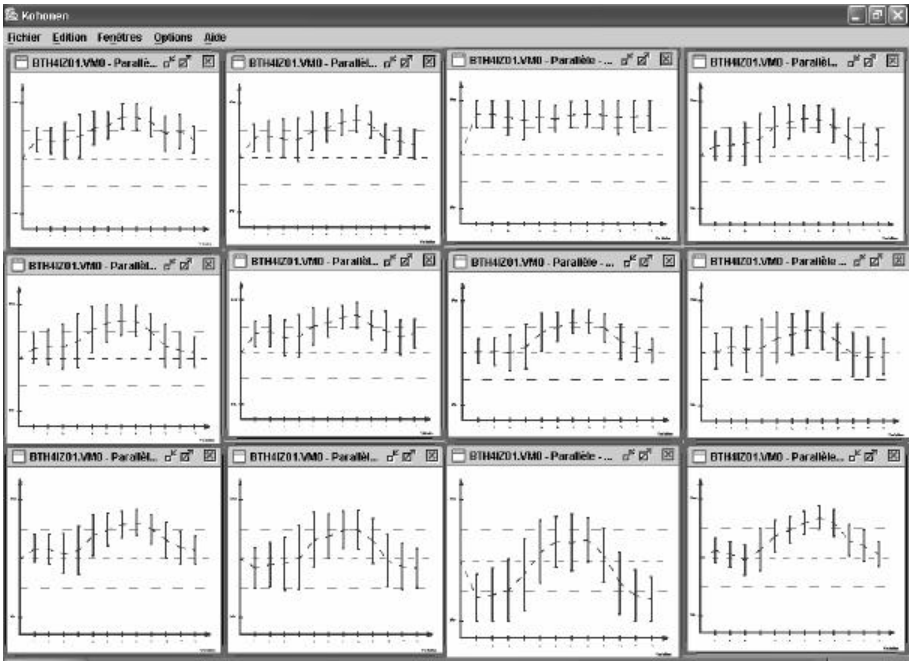


Fig. 12: *Bar diagram of the sides of the class-specific prototype rectangles z_i in the case of $p = 12$ interval-type variables (monthly temperatures), simultaneously for all $m = 12$ mini-clusters $C_1, ..., C_{12}$.*

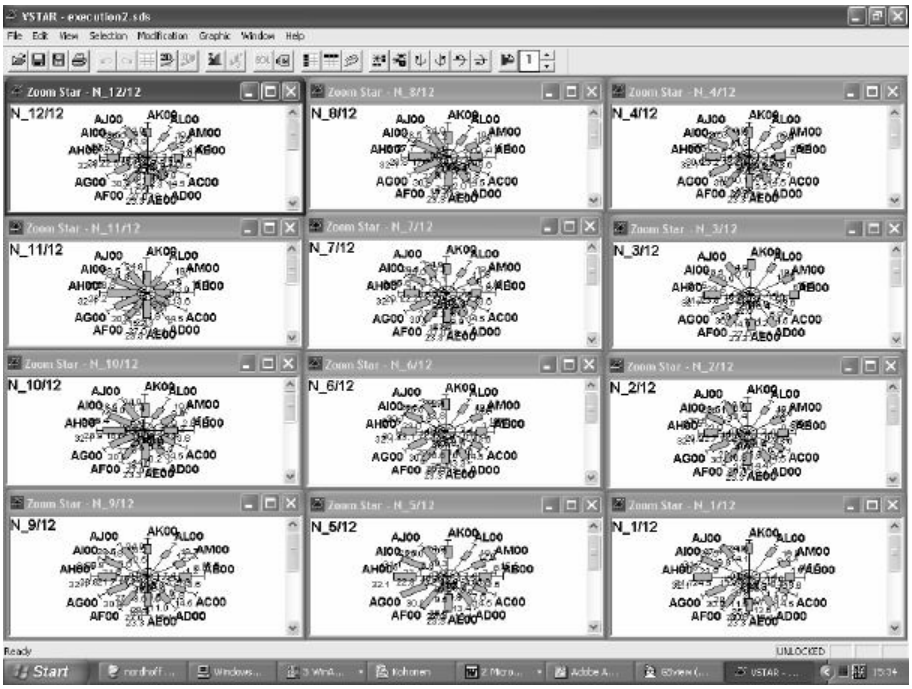


Fig. 13: *Zoom star diagrams of class-specific prototype rectangles z_i in the case of $p = 12$ interval-type variables, simultaneously for all $m = 12$ mini-clusters $C_1, ..., C_{12}$.*

This landscape with icons, letters, diagrams etc. provides an excellent tool for resolving many exploratory problems of data analysis and to reveal the overall structure of the underlying data. From the plethora of problems and possible strategies we mention the following ones:

(1) Recognizing the overall distribution of properties in the map and in the feature space, thereby revealing the topological structure of the data table;

(2) Revealing similar mini-clusters (similar patterns or prototypes) located in different domains of the underlying feature space;

(3) Scenting and locating interesting or remarkable patterns (prototypes) in the data set (and insofar in the feature space);

(4) Locating (in $\mathcal{L}$!) any contiguous regions where the mini-clusters have the same or similar properties. Typically, a contiguous set of similar mini-clusters will be aggregated to form one single, large and homogeneous *super-cluster* (a cluster of mini-clusters), and the system of all super-clusters will provide a natural classification of the set of objects (the data table, the feature space) into homogeneous, practicable and interpretable segments;

(5) Detecting any regions in the map where the mini-clusters have a *prespecified* property; this is exemplified in Fig. 10 where, e.g., the label e means that in the corresponding mini-cluster the persons with a 'huge income' prevail;

(6) Displaying class-specific properties relating to different variables and thereby discovering hidden dependencies among variables, and checking if these dependencies or other trends are the same on the whole map or not;

(7) Detecting 'aberrant' or 'outlying' mini-clusters whose elements behave quite differently either from the bulk of data or from the surrounding mini-clusters;

(8) Assigning new objects (new data hypercubes) to one of the mini-clusters and thereby locating it into the context of the n original objects and their the data.

5.3 The VPLOT display

The module VPLOT provides a geometrical display of the mini-clusters in the space of two arbitrarily selected variables. More specifically, after having clicked the middle box in the 'result' part of the chaining, we may select two of the p interval-type variables j and j', say, and can display the projection of the class prototypes $z_1, ..., z_m$ onto the two-dimensional Euclidean space that is spanned by the selected variables j and j' (see Fig. 14). This provides some idea about how clusters are distributed in this two-dimensional space and (by trying several different pairs of variables j, j') even in the whole space $\mathbb{R}^p$. Such displays can also reveal the amount of overlap and separation that may exist among the cluster prototypes.

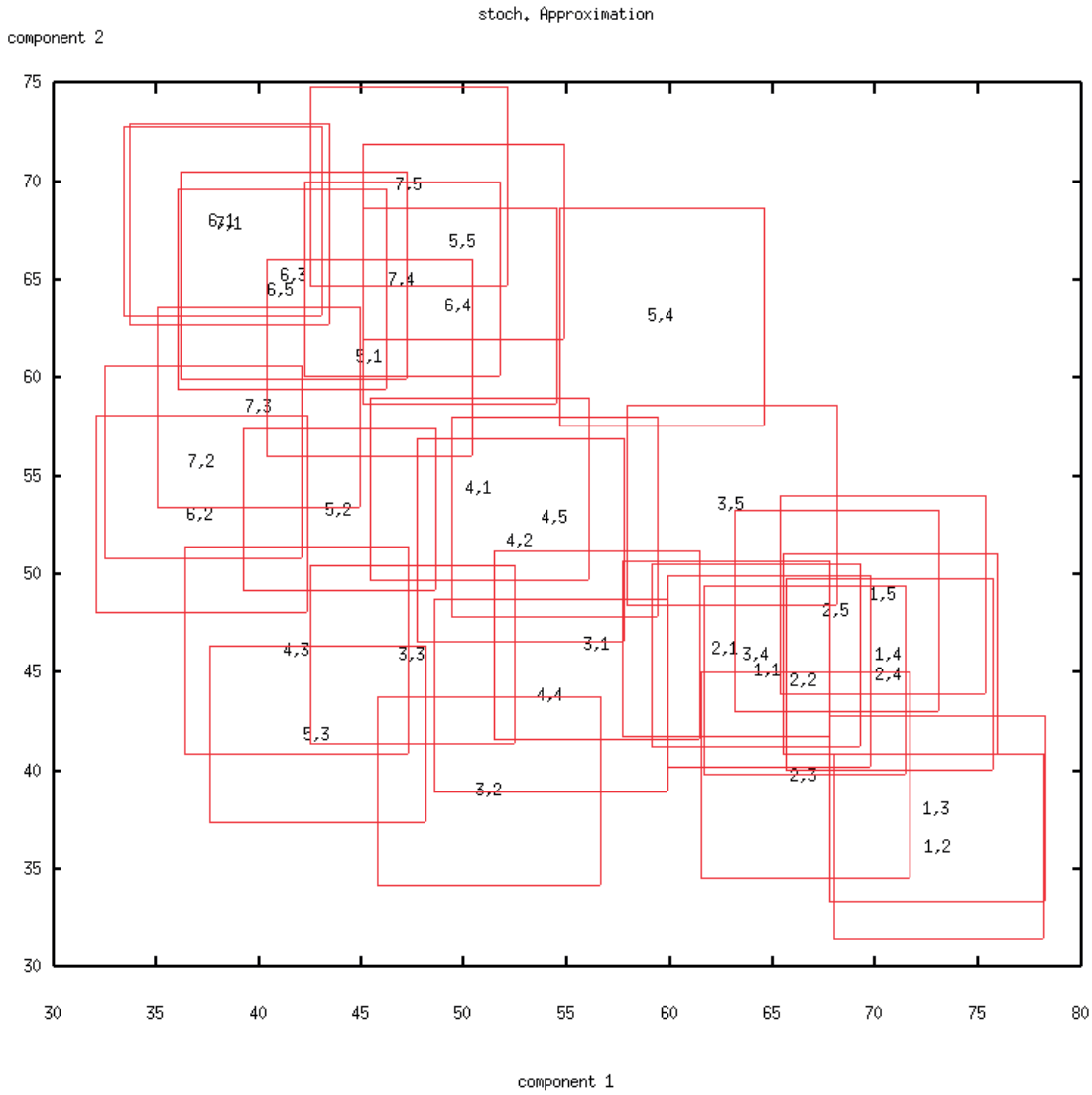


Fig. 14: *Projection of the class prototype rectangles $z_1, ..., z_m$ (here: $m = 7 \cdot 5 = 35$) onto the space of two selected variables $j = 1$ and $j' = 2$ (the underlying sample consists of 10.000 randomly simulated rectangles in a square of $\mathbb{R}^5$).*

6. Summary of the benefits from Kohonen maps

Kohonen maps and the SYKSOM algorithm share some useful features which make them very useful and effective for practical use, in particular for the analysis of large data tables and symbolic data:

As mentioned before, the very benefit from the SYKSOM module in the ASSO software results from the possibility to display visually the properties of the resulting (mini-)clusters at the location of the corresponding vertices in the lattice $\mathcal{L}$ (i.e., on the screen), in the form of the Kohonen map.

(a) The in-built clustering process reduces the large amount n of original data hypercubes (objects) to a quite smaller set of m mini-clusters or aggregates (data aggregation).

(b) An eventually *non-linear* configuration of the data (and cluster prototypes) in $\mathbb{R}^p$ is embedded into the two-dimensional lattice in a way which can follow the bends and twists of the underlying (hidden) manifold F and is in contrast to other methods such as PCA which implicitly assume some *linear* data structure.

(c) In this way the map provides a global overview on the whole data set, the properties of the constructed clusters can be easily displayed and compared to each other (e.g., by using VMAP or VIEW, labels and Zoom Stars).

(d) The method does not involve any parametric model.

(f) The clustering and assignment process is conducted in a sequential way which enters the data hypercubes $Q_1, Q_2, ...$ one after each other (and not all simultaneously): Insofar the results can be updated in when new data should be available.

Note that SYKSOM and VMAP can only display properties which concern the interval-type variables which were used for the cluster construction, but not other (e.g., categorical) variables which have been recorded in the original data base. However, the class membership of the items (stored in the saved partition file) can be included, as a new categorical variable, into the original data table (that may include other qualitative or modal values as well). Then the clusters can be analyzed with respect to *all* recorded variables just by opening a new chaining, converting the clusters to symbolic objects and analyzing them with the module VIEW.

7. How is the SYKSOM algorithm working?

The Kohonen map approach is often assigned to the *neural network* domain or to the field of *sequential learning*. This resides in the fact, that the underlying algorithms (and also SYKSOM) enter the data in a *sequential order* $x_1, x_2, x_3, ...$, thus *not simultaneously* as a bulk $\{x_1, ..., x_n\}$ of n data vectors. This sequential process works essentially as follows:

Suppose that, at some 'time' t, i.e. after having observed t data rectangles $x_1, ..., x_t$, a preliminary result has been obtained in the form of m mini-clusters $C_1^{(t)}, ..., C_m^{(t)}$ (assigned to the vertices $P_1, ..., P_m$) and m class prototypes $z_1^{(t)}, ..., z_m^{(t)}$. Then, in the next step, we include ('observe') the next data rectangle x_{t+1} and assign it to the 'closest', i.e., most similar, class (this implies that some dissimilarity measure has been introduced). For the new classes $C_1^{(t+1)}, ..., C_m^{(t+1)}$ we calculate some updated prototypes $z_i^{(t+1)}, ..., z_m^{(t+1)}$. Afterwards we proceed to the next data x_{t+2} etc.

After all n available data have been processed, we repeat the procedure in a cyclic way. After a (prespecified) number of cycles or when attaining (almost) a stationary result, the algorithm stops and provides the definite classes and prototypes.

When specifying the various steps of the algorithm, the user will have to specify various options and choices. This is briefly explained in section 3[5].

• *Input:*
A sequence of interval-type vectors (hypercubes, rectangles) $x_1, x_2, ...$ or $Q_1, Q_2, ...$ in $\mathbb{R}^p$. Specification of the numbers b and a of rows and columns of the lattice $\mathcal{L}$.

• *Initialisation step $t = 0$:*
Define an initial set of $m = ba$ (symbolic) cluster prototypes $z_1^{(0)}, ..., z_m^{(0)}$ in $\mathbb{R}^p$ and m empty classes $C_1^{(0)}, ..., C_m^{(0)}$ of objects. For all i, assign the class $C_i^{(0)}$ to the vertex P_i of the lattice.

• *Iteration $t \to t+1$* (i.e., including the $(t+1)$st data x_{t+1}):
At the begin of the step t, the data $x_1, ..., x_t$ were considered, the algorithm has determined a partition $C_1^{(t)}, ..., C_m^{(t)}$ of the first t objects together with a system of cluster prototypes (rectangles) $z_1^{(t)}, ..., z_m^{(t)}$.

1. Consider (observe, include) the $(t+1)$st data rectangle x_{t+1}.

2. Determine, from the set of all current class prototypes $z_1^{(t)}, ..., z_m^{(t)}$ (i.e., those constructed from $x_1, ..., x_t$), a prototype $z_{i^*}^{(t)}$ which is closest to the new data rectangle x_{t+1} in the sense

$$d(x_{t+1}, z_{i^*}^{(t)}) = \min_{i=1,...,m} d(x_{t+1}, z_i^{(t)}).$$

Here d is a measure for the *dissimilarity* of the underlying rectangles. SYKSOM provides three different choices for d (see section 2.3.2).

3. Assign the object $t+1$ (i.e., the data rectangle x_{t+1}) to the closest class $C_{i^*}^{(t)}$ leaving all other classes unchanged. This yields the new mini-clusters

$$\begin{aligned} C_{i^*}^{(t+1)} &:= C_{i^*}^{(t)} \cup \{t+1\} && \text{for } j = i^* \\ C_j^{(t+1)} &:= C_j^{(t)} && \text{for all } j \neq i^*. \end{aligned}$$

4. Update the prototype $z_{i^*}^{(t)}$ of the old, modified class $C_{i^*}^{(t)}$ *and also all other prototypes* $z_j^{(t)}$ with $j \neq i^*$, with a suitable 'updating formula' which essentially means shifting the class prototypes more or less towards the new data vector x_{t+1}. As a matter of fact, the amount of shifting depends on the *degree of neighbourhood* between the corresponding vertices P_{i^*} and P_j in the lattice $\mathcal{L}$. Without mentioning the exact updating formula here, we should say here that the shift will be large for classes $C_j^{(t+1)}$ where, in the lattice $\mathcal{L}$, P_j has a small path distance $\delta(P_j, P_{i^*})$ from P_{i^*} whereas the modification is small (or even 0) if P_j is far away from P_{i^*} in $\mathcal{L}$, i.e., for a large distance $\delta(P_j, P_{i^*})$.

This updating step yields m new class prototypes (rectangles) in $\mathbb{R}^p$:

$$z_1^{(t+1)}, z_2^{(t+1)}, ..., z_m^{(t+1)}.$$

[5] An exact, complete presentation is given in the forthcoming monograph by Diday and Noirhomme.

• *Iteration cycles and stopping:*
The steps 1. to 4. are iterated until all n available data rectangles were processed. After this first cycle the algorithm repeats the same procedure in subsequent cycles (where all n data are processed again in each cycle). Thereby the cluster prototypes obtained at the end of cycle $j-1$ are used as the initial cluster prototypes for the next cycle j. – The iteration stops after a full cycle when either a maximum number C of cycles was obtained or if the total (relative) difference between the prototypes obtained after cycles $j-1$ and j, respectively, are smaller than a precision threshold $\delta > 0$ (stopping criterion). Both parameters C and δ can be selected in the `Parameter Specification Window`. In order to avoid extended computation times for very large data sets or a large size m of the lattice, the SYKSOM module disregards the precision threshold for $C < 20$.

INFORMATION SOCIETY TECHNOLOGIES
PROGRAMME

CLINT User Manual

Interpretation of Clusters

Edited by FEP

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

ASSO Project
Workpackage WP6.1 - Module CLINT

Clustering Interpretation

Interpreting Clusters by using the module CLINT

A Short Tutorial for Users

Paula Brito

Version 1: December 4, 2003

1 Objectives of the Module

The module CLINT is used to obtain an interpretation of classes (subsets of the data set). These may result from a clustering algorithm or be directly identified in the given symbolic data array. The classes are identified by nominal or multinominal variables: a class is formed by the set of individuals (symbolic objects) sharing a given category of the defining variable.
As an example, consider a data array consisting of the staff of some given teaching institutions, described by age (interval variable), marital status (categorical multi-valued variable) and staff category (modal variable), on which a clustering method has been applied resulting on 2 classes, indicated by variable "class" (table 1). In this case, the user may wish to interpret either or both of the two clusters defined by variable "class", or any class resulting from selecting institutions having staff with a given "marital status" (for example, institutions with "single" people, that is, {Institution 1, Institution 2, Institution 3}).
Classes are interpreted on the basis of a given set of variables, by de-

	Age	Marital Status	Staff Category	Class
Institution 1	[20,45]	{single, married}	Administration (30%), Teaching (70%)	Class 2
Institution 2	[30,50]	{single,married, divorced}	Administration (20%), Teaching (60%), Cleaning (20%)	Class 1
Institution 3	[25,60]	{single, married, widow}	Administration (20%), Teaching (80%)	Class 2
Institution 4	[28,55]	{married, divorced}	Administration (10%), Teaching (80%), Cleaning (10%)	Class 1

Table 1: Example of Data Array with a "partition" variable

scriptions in the form of symbolic objects and by numerical indicators related to variable contributions.

2 Input of the CLINT module

The input for the module CLINT is a symbolic data array, where individuals, associated to symbolic objects, are described by quantitative single, interval, nominal, multinominal and /or modal variables (mixed types are allowed). There must be at least one nominal or multinominal variable, which will be used to define the classes to interpret. Taxonomies and hierarchical dependencies defined on variables may be taken into account.
The user must select the variable which defines the classes to be interpreted. A class is formed by the set of individuals (symbolic objects) sharing a given category of this variable. To interpret clusters of a previously obtained partition, hierarchy or pyramid, the corresponding variable must be chosen. Partitions are represented by nominal variables, hierarchies and pyramids are represented by multinominal variables. By default, the first nominal or multinominal variable in the data array is chosen. Then, the user must select the categories corresponding

to the classes he wishes to interpret: the "Aggregation Variable" must be chosen prior to the categories selection. By default, all categories are select, which means that all classes defined by this variable will be interpreted. Also, the user chooses the variables that will be used to interpret the classes. CLINT accepts all types of variables.

3 Description of the cluster

For each cluster, CLINT determines

- A long description, given by the union, on all chosen variables, of all symbolic objects representing the members of the cluster. This is the most specific generalization of the cluster members, and it takes into account the variability, within the cluster, of each chosen variable.
- A short description: the simplest symbolic object (that is, on the least number of variables) such that all members in the cluster belong to its extension, and it is discriminant (i.e., no element outside the cluster belongs to its extension).
- The contribution of the cluster to the total variability of the data array, the importance of each variable for each cluster and the contribution of each cluster to the variability of each variable.

In the example of the previous section, a long description of Class 1 = {Institution 2, Institution 4} would be
$[Age \in [28, 55]] \wedge$ [Marital status $\in$ {single, married, divorced}] $\wedge$
[staff category $\leq$ { Administration (80%), Teaching (20%), Cleaning (20%}]
and a short description could be [Marital status $\supset$ {divorced}]

3.1 Determining the long description

The long description of a class of symbolic objects is obtained by a generalisation operation. This procedure differs according to the variable type. Let $s_1, \ldots, s_q$ be a set of symbolic objects we wish to generalise. Generalisation is performed variable wise.

3.1.1 Interval Variables

In the presence of an interval variable Y, the symbolic objects $s_h, h = 1, \ldots, q$, contain events of the form $e_h = [Y \subseteqq I_h]$, with $I_h = [a_h, b_h], h = 1, \ldots, q$. In the generalised object, the event corresponding to this variable will be $e = [Y \subseteqq [Min\{a_h\}, Max\{b_h\}]$, since this is the smallest interval that contains all the intervals $I_h = [a_h, b_h], h = 1, \ldots, q$.
Example: Consider the set of symbolic objects $s_1, \ldots, s_4$ defined on variables age and salary, each one describing a group of people $C_1, \ldots, C_4$, respectively.
$s_1 = [age \subseteqq [20, 45]] \wedge [salary \subseteqq [1000, 3000]]$
$s_2 = [age \subseteqq [35, 40]] \wedge [salary \subseteqq [1200, 3500]]$
$s_3 = [age \subseteqq [25, 45]] \wedge [salary \subseteqq [2000, 4000]]$
$s_4 = [age \subseteqq [30, 50]] \wedge [salary \subseteqq [2000, 3200]]$
The generalised object is: $s = [age \subseteqq [20, 50]] \wedge [salary \subseteqq [1000, 4000]]$

3.1.2 Nominal and Multinominal Variables

For a nominal or multi-valued variable Y with domain $O = \{m_1, \ldots, m_k\}$, the events in the symbolic objects $s_1, \ldots, s_q$ have the form: $e_h = [Y = v_h]$ with $v_h \subseteqq O$ (if Y is a classical nominal variable) or $e_h = [Y \subseteqq V_h]$ with $V_h \subseteqq O$ (if Y is a multi-valued variable). For generalisation purposes, $e_h = [Y = v_h]$ is equivalent to $e_h = [Y \subseteqq \{v_h\}]$, so both cases can be considered together. Then, generalisation is made by applying the union operator to the sets associated to the corresponding events; in the generalised object, the event corresponding to the variable in question will be: $e = [Y \subseteqq V]$, with $V = \bigcup_{h=1}^{q} V_h$ since V is the smallest set that contains all the $V_h, h = 1, \ldots, q$.
Example: Consider a set of symbolic objects $s_1, \ldots, s_4$, defined on variables sex and nationality, each one describing a group of people $C_1, \ldots, C_4$, respectively:
$s_1 = [\text{sex} \subseteqq \{M\}] \wedge [\text{nationality} \subseteqq \{\text{French, English}\}]$
$s_2 = [\text{sex} \subseteqq \{M, F\}] \wedge [\text{nationality} \subseteqq \{\text{German, English}\}]$
$s_3 = [\text{sex} \subseteqq \{F\}] \wedge [\text{nationality} \subseteqq \{\text{French, Belgian}\}]$
$s_4 = [\text{sex} \subseteqq \{M, F\}] \wedge [\text{nationality} \subseteqq \{\text{French, Portuguese}\}]$
The generalised object is:

$s = [\text{sex} \subseteqq \{M, F\}] \wedge$
[nationality $\subseteqq$ {French, Belgian, German, English, Portuguese}]

3.1.3 Modal Variables

For a modal variable Y, with underlying domain $O = \{m_1, \ldots, m_k\}$, generalisation can be defined in two ways, each with a particular semantics:

- i) Generalisation by the maximum
  In this case, generalisation of events is performed by taking, for each category m_j , the maximum of its probabilities.
  The generalisation of $e_1 := [Y \leq \{m_1(p_1^1), \ldots, m_k(p_k^1)\}], \ldots, e_q := [Y \leq \{m_1(p_1^q), \ldots, m_k(p_k^q)\}]$ is
  $e := [Y \leq \{m_1(p_1), \ldots, m_k(p_k)\}]$ where $p_\ell = Max\{p_\ell^h, h = 1, \ldots, q\}, \ell = 1, \ldots, k$.
  Notice that the generalised event is typically no longer probabilistic, in the sense that the obtained distribution is no longer a probability distribution, since $p_1 + \ldots + p_k > 1$ is possible.

  Example: Let $e_1 := [Y \leq \{\text{administration}(30\%), \text{teaching}(70\%)\}]$
  $e_2 := [Y \leq \{\text{administration}(60\%), \text{teaching}(20\%), \text{secretary}(20\%)\}]$
  The generalised event is:
  $[Y \leq \{\text{administration}(60\%), \text{teaching}(70\%), \text{secretary}(20\%)\}]$

- ii) Generalisation by the minimum: In this case, generalisation of events is performed by taking, for each category , the minimum of its probabilities.
  The generalisation of $e_1 := [Y \geq \{m_1(p_1^1), \ldots, m_k(p_k^1)\}], \ldots, e_q := [Y \geq \{m_1(p_1^q), \ldots, m_k(p_k^q)\}]$ is
  $e := [Y \geq \{m_1(p_1), \ldots, m_k(p_k)\}]$ where $p_\ell = Min\{p_\ell^h, h = 1, \ldots, q\}, \ell = 1, \ldots, k$.
  The generalised event is typically no longer probabilistic since in this case $p_1 + \ldots + p_k \leq 1$ is possible.

  Example: Let now $e_1 := [Y \geq \{\text{administration}(30\%), \text{teaching}(70\%)\}]$
  $e_2 := [Y \geq \{\text{administration}(60\%), \text{teaching}(20\%), \text{secretary}(20\%)\}]$
  The generalised event is:
  $[Y \geq \{\text{administration}(30\%), \text{teaching}(20\%)\}]$

3.2 Contribution of the cluster to the total variability of the data array

Let $E = \{w_1, \ldots, w_n\}$ be the set of units (individuals, groups,...) under consideration. The total variability of the data array is evaluated by the generality degree of the symbolic object associated to the whole set E. Let s_E be the corresponding symbolic object, that is $s_E = \bigcup_{i=1}^{n} s_i$, where s_i is the ith symbolic object in the data base, so s_E generalizes all elements of the array. Let C be a class, and s_c the (long) symbolic object generalizing its members. The contribution of a cluster C to the variability of the data base can be assessed by

$$Cont(C) = \frac{G(s_c)}{G(s_E)} \tag{1}$$

Notice that $0 < Cont(C) \leq 1$, and that $Cont(.)$ is increasing with respect to the symbolic order between symbolic objects. High values of $Cont(C)$ indicate that the cluster is almost as heterogeneous as the whole data array, low values indicate that the cluster is globally homogeneous.

3.3 Contribution of a variable to the variability of a cluster

Let C be a cluster, and $G(s_C)$ the generality degree of its associated long symbolic object.
Let $s_C = \bigwedge_{j=1}^{p} e_j^C$ with $e_j^C = [y_j \subseteqq V_j^C]$.

$$G(s_C) = \prod_{j=1}^{p} G(e_i^C) = \frac{\prod_{j=1}^{p} c(V_j^C)}{\prod_{j=1}^{p} c(O_i)} \tag{2}$$

Then, the contribution of a variable y_j to the variability of a cluster can be measured by

$$Cont(y_j, C) = \frac{c(V_j^C)}{\sum_{j=1}^{p} c(V_j^C)} \tag{3}$$

for $G(s_c) \neq 1$. Notice that $G(s_c) = 1$ iff $G(e_j^c) = 1, i = 1, \ldots, p$, that is when s_c is the "full" symbolic object (for all interval or multi-values variables, the whole observation set is present, and all distributions for histogram variables are uniform). Also notice that $\sum_{j=1}^{p} Cont(y_j, C) = 1$ and that $0 < Cont(y_j, C) \leq \frac{1}{p-1}$. This contribution evaluates the different roles of the variables in the cluster variability. High values indicate that the differences between the cluster elements are highly due to this variable; low values indicate that the cluster elements do not vary too much concerning this variable.

Example :
Consider a symbolic object s, describing a group p of people, $C = (p, s)$, defined on variables age, sex, nationality and staff category:

$s = [age \subseteqq [20, 45]] \wedge [sex \subseteqq \{M\}] \wedge [nationality \subseteqq \{French, English\}] \wedge [staff \leqq \{administration(0, 30), teaching(0, 70)\}] = e_1 \wedge e_2 \wedge e_3 \wedge e_4$

$c(V_1) = 45 - 20 = 25$
$c(V_2) = 1$
$c(V_3) = 2$
$c(V_4) = \sqrt{0,3} + \sqrt{0,7} = 4,38$

$Cont(age, C) = \frac{25}{25+1+2+4,38} = \frac{25}{32,38} = 0,77$

$Cont(sex, C) = \frac{1}{32,38} = 0,03$

$Cont(nationality, C) = \frac{2}{32,38} = 0,06$

$Cont(staffcat., C) = \frac{4,38}{32,38} = 0,135$

3.4 Contribution of a cluster to the variability of a variable

The contribution of a cluster C to the variability of a variable y_j is measured by :

$$Cont(C, y_j) = \frac{c(V_j^c)}{c(V_j^E)} = \frac{G(e_j^c)}{G(e_j^E)} \tag{4}$$

It evaluates for how much of the variable variability the cluster is responsible.
We have $0 < Cont(C, y_i) \leq 1$. High values indicate that, for this variable, the cluster is nearly as heterogeneous as the whole data array, low values indicate that the cluster may be considered homogeneous as concerns this variable.

4 Output: Interpretation Results

The output of CLINT is

- a Zoom Star representation of the selected classes
- a text file containing, for each selected class:
  - Membership list
  - Long symbolic object describing the cluster
  - Short (simple) symbolic object describing the cluster
  - Contribution of the cluster to the variability of the data base
  - Contribution table, gathering, for each of the chosen variables, the contribution of the cluster to the variability of the variable and the contribution of the variable to the variability of the cluster.

INFORMATION SOCIETY TECHNOLOGIES
PROGRAMME

SCLASS User Manual

Unsupervised Classification Tree

Edited by FUNDPMa

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

ASSO Project
Workpackage WP6.? - Module SCLASS

Unsupervised Divisive Classification

Clustering of interval data and its visualization

by using the modules SCLASS and VTREE

A Short Tutorial for Users

Jean-Paul Rasson - Pascale Lallemand - Séverine Adans

Version 1: October 3, 2003

One of the most common tasks in data analysis is the detection and construction of 'homogeneous' groups C_1,$C_2, \ldots$ of objects in a population E such that objects from the same group show a high similarity whereas objects from different groups are typically more dissimilar. Such groups are usually called 'clusters' and must be constructed on the basis of the data which were recorded for the objects.

SCLASS is a divisive clustering method for a symbolic $n \times p$ data array $\underline{X}$. By definition of a divisive clustering method, the algorithm starts with all objects in one larger cluster, and splits successively each cluster into (two) smaller ones until a suitable stopping rule prevents further divisions.
This algorithm proceeds also in a monothetic way. In other words, the algorithm assumes as input n data vectors and proceeds such that each split is carried out by using only one single variable (which is selected optimally).

1 Interval type data for Clustering Tree

This algorithm studies the case where a set of Symbolic Objects are described by p interval-valued variables $Y_1, \ldots ,Y_p$.
The interval-valued variable $Y_j (j = 1, \ldots ,p)$ is measured for each item of the basic set $E = \{1, \ldots ,n\}$. For each item $k \in E$, we denote the interval $Y_j(k)$ by $[\underline{y}_{jk},\bar{y}_{jk}]$, thus $\underline{y}_{jk}$ (resp. $\bar{y}_{jk}$) is the lower (resp. the upper) bound of the interval $Y_j(k) \subseteq \mathcal{R}$.

A small example is given by table 1

k \ j	Var 1	Var 2	Var 3
1	[8.5, 10]	[13.0, 15.2]	[5.0, 8.2]
2	[6.3, 9.1]	[14.1, 16.0]	[6.3, 7.2]
3	[7.9, 11.8]	[11.6, 13.5]	[4.9, 6.5]
4	[9.0, 11.0]	[10.9, 12.5]	[7.1, 8.1]
5	[6.3, 7.2]	[12.9, 15.0]	[6.2, 7.4]
6	[7.1, 7.9]	[11.5, 12.9]	[4.8, 5.7]
7	[7.5, 9.4]	[13.2, 15.0]	[6.6, 8.1]
8	[6.6, 7.8]	[12.4, 13.2]	[5.7, 7.2]

TAB. 1 – *a symbolic interval data table*

The item 4 is then described by the row 4 of the table; i.e. the vector of intervals

$$x_4 = \begin{pmatrix} [9.0,11.0] \\ [10.9,12.5] \\ [7.1,8.1] \end{pmatrix}$$

2 What is Clustering Tree?

SCLASS is a recursive algorithm for sharing a given population of symbolic objects into classes.
In this type of method, nodes are split recursively by choosing the best interval variable. The original contribution of this method lies in the way of splitting a node. Indeed, the cut will be based on the only assumption that the distributions of points can be modeled by non-homogeneous Poisson process where the intensity will be estimated by the kernel method. The cut will be made in order to maximize the likelihood function.

2.1 General Hypothesis: Non Homogeneous Poisson process

We consider a problem of clustering where observed points are generated by a Non Homogeneous Poisson Process with intensity $q(.)$ and are observed in D, where D is the union of g disjoint convex fields.

The likelihood function, for the observations $\underline{x} = (x_1, x_2, \ldots, x_n)$ with $x_i \in R^d, i = 1, \ldots, n$ is:

$$F_D(\underline{x}) = \frac{1}{(\rho(D))^n} \prod_{i=1}^{n} 1_D(x_i).q(x_i)$$

where $\rho(D) = \int_D q(x)dx$ is the integrated intensity;
$q(.)$ is the process' intensity ($q(x) > 0 \ \forall x$).

Consequently, if the intensity of the process is known, the solution of the likelihood maximum will correspond to g disjoint convex fields containing all the points and for which the sum of their integrated intensities is minimal. When the intensity is unknown, it will be estimated.

2.2 Kernel Method

To estimate the intensity of a Non Homogeneous Poisson Process, we will use a non-parametric method: the **Kernel method**.
The Kernel estimator is defined by :

$$\hat{q}(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{x - X_i}{h}),$$

where - K is the kernel with these properties:
symetric, continu, $\int K(x)dx = 1$, $K \geq 0$;
- h is the window width also called the **smoothing parameter**.

The kernel estimator is a sum of 'bumps' placed around the observations. The kernel function K determines the shapes of the bumps while the window width h determines their width. How determines this parameter?

2.3 Bumps and Multi-modalities

Silverman has showed interest for the search of modes within the framework of the clustering. He distinguishes the concept of bumps and modes: a mode in a density f will be a local maximum, while a bump will be characterized by an interval in such way that the density is concave on this interval but not on an larger interval.

In the framework of the density estimation by the kernel method,
- for very great values of h, the density estimation is unimodal;
- for h decreasing, the number of mode increases.

This behaviour is described by Silverman: "*the number of modes is a decreasing function of the smoothing parameter* h". This is showed only for the NORMAL kernel.

Consequently, to realize the estimation of the intensity of the Non Homogeneous Poisson Process, we will use the kernel method with the normal kernel. This kernel is defined by:

$$K_{\mathcal{N}}(t) = \frac{1}{\sqrt{2\pi}}e^{-\frac{t^2}{2}}$$

Since we use the normal kernel, there is a critical value h_{crit} of the smoothing parameter for which the estimation changes from the unimodality to multi-modality. Our criterion of split seeks this value.

2.4 Splitting criteria

For each variable, by a dichotomic process, we find the greatest value of parameter h giving a number of modes of the associated intensities strictly larger than 1. Once this h determined, we cut into two convex disjoint fields for which the likelihood function is maximum; i.e. for which the integrated density is smallest.

Proceeding variable by variable, we will be able to select the one which generates the greatest likelihood function.

This procedure is recursively performed until the number of points in a node is less than a predefined value.

2.5 Pruning method

At the end of the splitting process, we end with a huge tree. The best subtree is selected. Indeed, we have developed, under the hypothesis of non homogeneous Poisson Process, a tree pruning method that takes the form of a classical test of hypothesis: the **Gap Test** (Kubushishi).

The Gap Test, in the case of two classes D_1 and D_2, tests the hypothesis

H_0: there is $n = n_1 + n_2$ points in $D_1 \cup D_2$

against the hypothesis

H_1: there are n_1 points in D_1 and n_2 points in D_2, with $D_1 \cap D_2 = \emptyset$.

In other words, the null assumption involves what we call a bad cut whereas the alternative assumption gives a good cut.

The pruning method built here traverses the tree branch by branch from its root to its end, in order to index the good cuts (Gap Test satisfied) and the bad cuts (Gap Test non-satisfied). The ends of the branches for which there are only bad cuts are pruned.

2.6 Application to INTERVAL data

How apply this new clustering method to symbolic data of interval type?

Note M the center of an interval and L the length of an interval.
The usual distance used for interval variables is the Hausdorff distance:

$$d_H([a_1,b_1],[a_2,b_2]) = max(|a_1 - a_2|,|b_1 - b_2|)$$

or (Chavent and Lechevallier)

$$d_H([a_1,b_1],[a_2,b_2]) = |M_1 - M_2| + |L_1 - L_2|.$$

We can work on the space $(M,L) \subseteq \mathcal{R} \times \mathcal{R}^+$, where each interval is represented by its center and its length.
Consequently, as we use a divisive method, separations must respect the order of the classes' centers.

Thus, if there is a density ρ_1 on the axis M and another ρ_2 on the axis L, we must minimize, in the most general case of a Non Homogeneous Poisson process the integrated intensity:

$$\int_{M_i}^{M_{i+1}} \rho_1(m)dm + \int_{min(L_i,L_{i+1})}^{max(L_i,L_{i+1})} \rho_2(l)dl \tag{1}$$

and to choose as bipartition this one generated by any point being located inside the interval which maximizes 1.

3 SCLASS output

The SCLASS algorithm produces as output a file .sds containing a new set of symbolic objects: the clusters of the partition.
Moreover, a new categorical multiple variable is created. This variable enumerates the nodes to which the object belong.
An internal output file (.vt0) contain the structure of the tree (used like input of VTREE).
A listing file contains the database's characteristics, the parameters' value and the detailed splits of the tree.
A sds file, where each node is represented by a symbolic object, is also created.

4 Example

The above clustering method has been examined with the well-known Ichino's oils dataset.
The data set is composed of 8 oils described in term of four interval variables

Sample	Specific Gravity	Freezing point	Iodine Value	Saponification Value
linseed oil	[0.930;0.935]	[-27;-18]	[170;204]	[118;196]
perilla oil	[0.930;0.937]	[-5;-4]	[192;208]	[188;197]
cottonseed oil	[0.916;0.918]	[-6;-1]	[99;113]	[189;198]
sesam oil	[0.920;0.926]	[-6;-4]	[104;116]	[187;193]
camelia oil	[0.916;0.917]	[-21;-15]	[80;82]	[189;193]
olive oil	[0.914;0.919]	[0;6]	[79;90]	[187;196]
beef tallow	[0.860;0.870]	[30;38]	[40;48]	[190;199]
hog fat	[0.858;0.864]	[22;32]	[53;77]	[190;202]

TAB. 2 – *Table of oils and fats*

The created results file is the following:

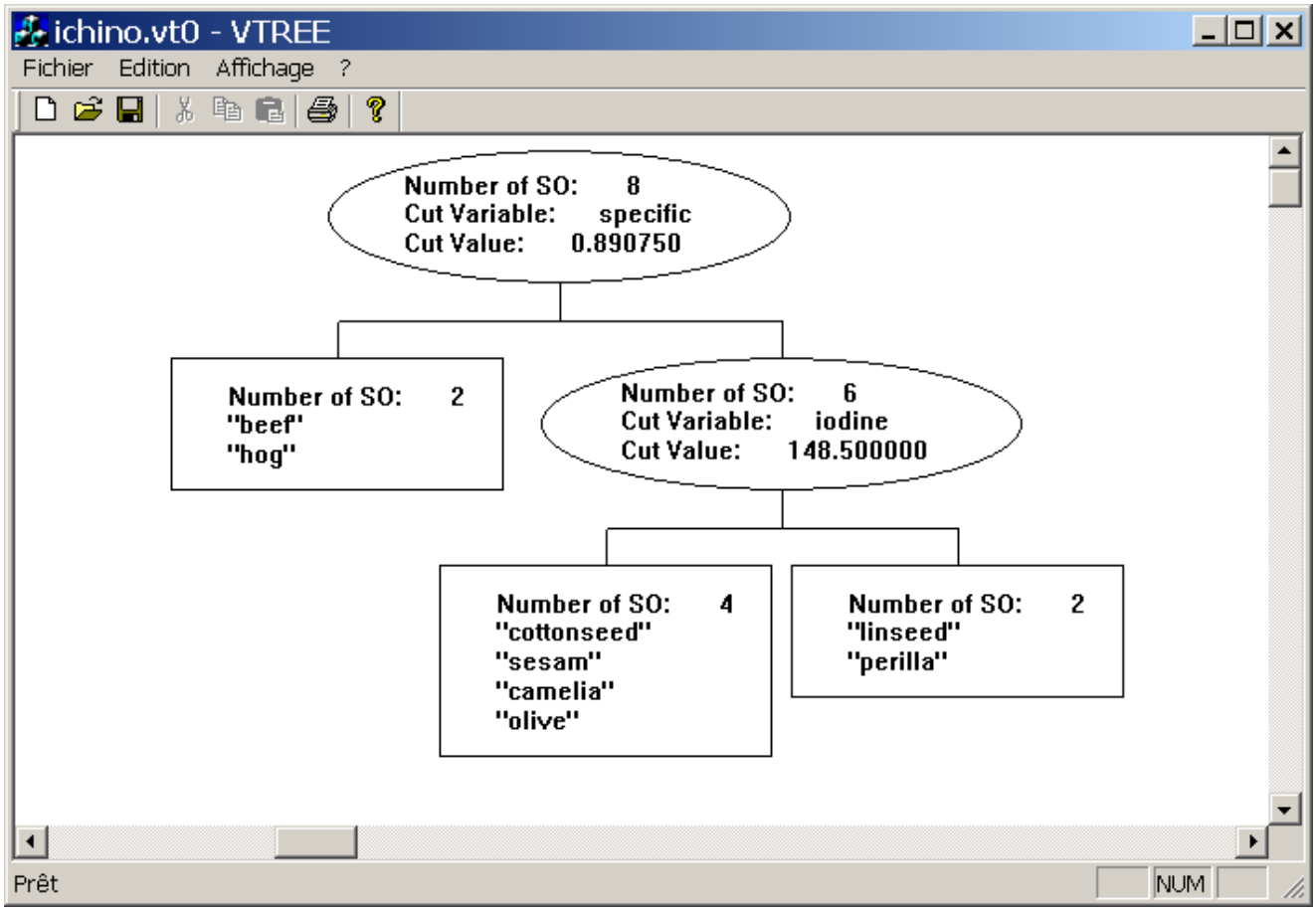
```
-------------------------------------------------------
 BASE= D:\asso_11_2003\sclass_29_11\ichino.sds
 Number of OS = 8
 Number of variables = 4
 METHOD=SCLASS Version 1.0    08/2002
-------------------------------------------------------

Sodas The Statistical Package for Symbolic Data Analysis
Version 1.0 - 09/2002
MODULE: SCLASS
Symbolic Unsupervised Classification Tree

-------------------------------------------------------
---------------------------------------------------------
```

```
-------------------------------------------------------
Sodas File          : D:\asso_11_2003\sclass_29_11\ichino.sds
Log File            : D:\asso_11_2003\sclass_29_11\ichino.LOG
Listing File        : D:\asso_11_2003\sclass_29_11\filieres\ichino.LST
LaTeX File          : D:\asso_11_2003\sclass_29_11\filieres\ichino.tex
Partition file      : D:\asso_11_2003\sclass_29_11\filieres\ichinoCluster.sds
Node file             : D:\asso_11_2003\sclass_29_11\filieres\node.sds
-------------------------------------------------------
-------------------------------------------------------
-------------------------------------------------------

 Learning Set         :     8
 Number of variables  :     3
 Min. number of object by node :     5
 Alpha Value    : 0.500000

 GROUP OF SELECTED VARIABLES :
 ============================

 ( Pos )    Name          Type

 (   1 ) specific     INTERVAL
 (   2 ) freezing     INTERVAL
 (   3 ) iodine       INTERVAL

 LIST OF SYMBOLIC OBJECTS IN THE SET :
 =====================================

 "linseed"        "perilla"        "cottonseed"     "sesam"          "camelia"
 "olive"          "beef"           "hog"

  ========================
  Split of the node :    1
  =========================

  Number of Symbolic objects in the node:      8pt
  ---------------------------------------

   Criteria of cut :
  -----------------

 Cut variable :(  1) specific
 Cut value :   0.89
 Smoothing parameter CENTER :   0.02
 Smoothing parameter LENGTH :   0.00
 Rule : if value of i <    0.89 -> the SO i is in the left node (next even node)
        if value of i >    0.89 -> the SO i is in the right node (next odd node)

Node :   2    Cardinal :     2pt
====================================
 (6)  "beef"
 (7)  "hog"

Node :   3    Cardinal :     6pt
====================================
 (0)  "linseed"
```

```
 (1)  "perilla"
 (2)  "cottonseed"
 (3)  "sesam"
 (4)  "camelia"
 (5)  "olive"

  ========================
  Split of the node :    2
  ========================

  Number of Symbolic objects in the node:      2pt
  ---------------------------------------

     TERMINAL NODE

 the stop-splitting is true : the size of the node is too small
 value of the stop-splitting rule:     5

  ========================
  Split of the node :    3
  ========================

  Number of Symbolic objects in the node:      6pt
  ---------------------------------------

   Criteria of cut :
  -----------------

 Cut variable :(  3) iodine
 Cut value : 148.50
 Smoothing parameter CENTER :  36.10
 Smoothing parameter LENGTH :   3.40
 Rule : if value of i <  148.50 -> the SO i is in the left node (next even node)
        if value of i >  148.50 -> the SO i is in the right node (next odd node)

Node :   4    Cardinal :     4pt
====================================
 (2)  "cottonseed"
 (3)  "sesam"
 (4)  "camelia"
 (5)  "olive"

Node :   5    Cardinal :     2pt
====================================
 (0)  "linseed"
 (1)  "perilla"

  ========================
  Split of the node :    4
  ========================

  Number of Symbolic objects in the node:      4pt
  ---------------------------------------

     TERMINAL NODE
```

```
the stop-splitting is true : the size of the node is too small
value of the stop-splitting rule:     5

 ========================
 Split of the node :    5
 ========================

 Number of Symbolic objects in the node:      2pt
 ---------------------------------------

    TERMINAL NODE

the stop-splitting is true : the size of the node is too small
value of the stop-splitting rule:     5
```

The graphic representation after the pruning procedure of the results is:



The resulting 3-clusters partition is:

- C_1: (linseed, perilla),
- C_2: (hog, beef),
- C_3: (camelia, cottonseed, sesam, olive),

References

[1] H. H. Bock and E. Diday. *Analysis of Symbolic Data, Exploratory methods for extracting statistical information from complex data.* Springer-Verlag, 2000.

[2] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees.* Wadsworth, 1984.

[3] M. Chavent. *Analyse des Données Symboliques: Une méthode divisive de classification.* PhD thesis, Université Paris IX-Dauphine, 1992.

[4] M. Chavent and Y. Lechevallier. Dynamical clustering of interval data: Optimization of an adequacy criterion based on hausdorff distance. In K. Jajuga, A. Sokolowski, and H. H. Bock, editors, *Classification, Clustering, and Data Analysis*, pages 53–60, Berlin, Germany, juillet 2002. Springer.

[5] T. Kubushishi. *On some Applications of the Point Process Theory in Cluster Analysis and Pattern Recognition.* PhD thesis, Facultés Universitaires Notre-Dame de la Paix, 1996.

[6] J. P. Rasson and T. Kubushishi. The gap test: an optimal method for determining the number of natural classes in cluster analysis. In E. Diday, Y. Lechevallier, M. Schader, P. Bertrand, and B. Burtschy, editors, *New approaches in Classification and Data Analysis*, pages 186–193, 1994.

[7] B.W. Silverman. Using kernel density estimates to investigate multimodality. *Journal of Royal Statistic Society, B*, 43:97–99, 1981.

[8] B.W. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, 1986.

Factorial

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

SPCA User Manual

Principal Component Analysis

Edited by DMS

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **03/02/2004**

SPCA : Principal Component Analysis

1 Introduction

The SPCA aims to look for the best representation of the a set of Symbolic Data described by interval variables on a factorial plane. A first PCA analysis of SO's was proposed by Cazes et al. (1997).

In ASSO/SODAS 2 software we have implemented three approaches of the PCA on SO's in order to improve the representation of the SO's on a reduced factorial sub-space in terms of compactness of the images of each SO and separation between SO's. These alternative proposed approaches have been denoted as: Vertices SPCA *Principal Component Analysis on Symbolic objects under cohesion vertices constraints*; RT-PCA *Range Transformation Principal Component Analysis* and a mixed strategy.

We remind that the cited first approach by Cazes st al. (1997) optimized as criterion the total variance of all the vertices of the hypercubes, treating them as independent each other.

In the new approach to PCA, Vertices SPCA, by introducing a cohesion constraint on the vertices of each hypercube which represents a SO, the criterion maximizes becomes the variance between SO's instead than the total variance.

In order to retain the same kind of geometrical representation of the SO's in the original space, on the factorial plane they are visualized as rectangles with sides parallel to the axes.

The second approach in such context of analysis is the RT-PCA. It is based on a Range-Transformation of the SO descriptions in order to put in evidence the main differences among SO's on the factorial plane. From a geometrical point of view, this approach consists in translating the minimum vertices of the hypercubes associated to the SO's in the origin of the origin of the representation space. Then, PCA is performed on the maximum vertices of the hypercubes. A comparison among SO's can be easily realized according the size and the shape of their images on the factorial plane.

The last approach to PCA proposed in the software consists in a Mixed Strategy which allows to represents SO's as rectangles projecting the vertices of the hypercube on the factorial axes achieved by RT-PCA procedure.

2 The input to SPCA

All the SPCA approaches are performed on a set of n SO's, described by p descriptors, all continuous at interval. The rows of the matrix $\mathbf{Z}$ correspond to the coordinates of the vertices of the hypercube representing the SO in a p dimensional space.

Moreover, logical relations among the descriptors and missing values (NULL) are admitted.

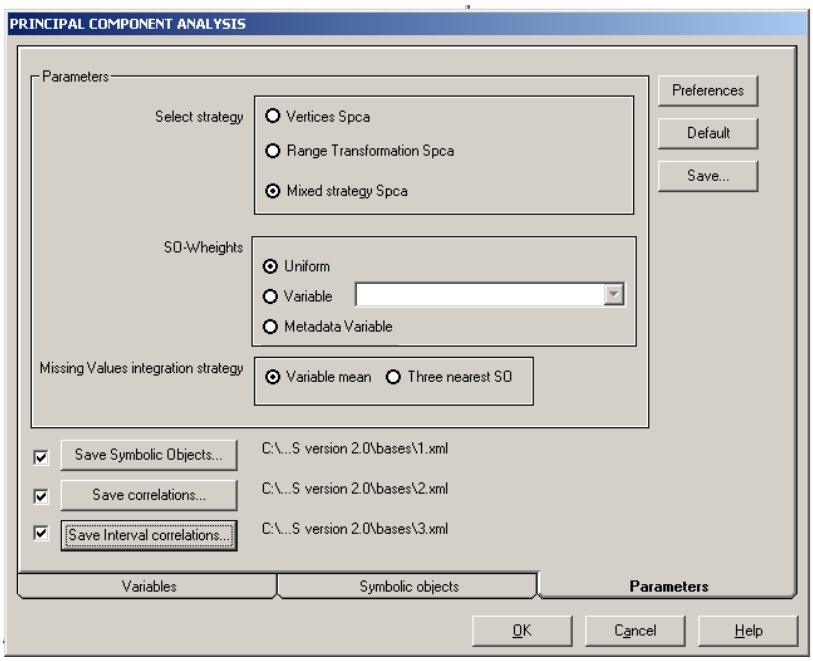


Figure 1: Parameters window for SPCA

3 The output of SPCA

The SPCA results are: the coordinates of the vertices of the hypercubes and the coordinates of the extremes of the intervals of the rectangles which reproduce the projections of the vertices of the hypercubes on each factorial axis. The latter will be available in a SOM file. These results will be display by using V-PLOT module. In particular, the graphical output of SPCA is constituted by four plots:

1) The SO's rectangular plot: the objects are represented on the factorial planes by means of their minimum and maximum coordinates as described in the text output file.

2) The SO's convex hull plot: on the base of the two axes chosen in the parameters window, the SO's are represented by means of the convex hull of the original vertices projected onto the factorial plane. This kind of visualization represents the original hypercube projected onto the factorial plane, while rectangular plot represent the minimum rectangle area that contains the projection of the original hypercube onto the factorial plane.

3) Circle of correlation plot: as for classical PCA, the original descriptors are represented by means of their correlation with the factorial axes in the unit circle.

4) Interval correlation plot: in this plot, are represented the interval correlation of each variable with the factorial axes.

In analogy with the classical PCA method, the other main results reported in the textual output file are: the absolute value and the percentage of inertia explained by each axis, some indices to evaluate the quality of the representation of the SO's on the plane, the contribution of each object and of each descriptor to the achievement of the several factorial axes.

4 SPCA method

SPCA, as all the other factorial technique is based on multi-step procedures and it is constituted by the following steps:

i) SO descriptors transformation (or coding);

ii) numerical treatment of the transformed descriptors;

iii) interpretation of the results according to the original nature of the data. This approach is known as symbolic-numerical-symbolic (Diday, 1987).

In particular, SPCA is performed on a set of n SO's, described by p interval variables. The symbolic data matrix in input is denoted $\mathbf{Z}$ (n x p).

The first phase (coding phase) of the SPCA procedure consists in a numerical transformation of the symbolic matrix $\mathbf{Z}$ by performing a Cartesian product of all the bound of the intervals in the symbolic description of each SO. In the general case, for p descriptors, at each rows of $\mathbf{Z}$ is associated a matrix $\mathbf{Z}_i$ will have 2^p rows and p columns. The matrix $\mathbf{Z}_i$ associated to the description of the i-th SO, assuming $p = 2$ for simplicity, is the following:

$$Z_i = \begin{array}{c} \begin{array}{cc} z_1 & z_2 \end{array} \\ \begin{pmatrix} \underline{y_{i,1}} & \underline{\overline{y_{i,2}}} \\ \underline{\overline{y_{i,1}}} & \overline{y_{i,2}} \\ \overline{y_{i,1}} & \underline{\overline{y_{i,2}}} \\ \overline{y_{i,1}} & \overline{y_{i,2}} \end{pmatrix} \end{array}$$

where the elements are respectively the lower and the upper bound of the intervals of values ($\left[\underline{y_{i,1}};\overline{y_{i,1}}\right],\left[\underline{y_{i,2}};\overline{y_{i,2}}\right]$) assumed by the SO for the two numerical descriptors y_1 and y_2.

Furthermore, in order to take in account logical conditions in the symbolic objects description, we propose to decompose the SO's in sub-objects, whose descriptions are consistent with the condition expressed by the logical rules.

Therefore, the number of SO's to be consider in the analysis increase of the number of sub-objects. For simplicity, we leave equal to n the total number of SO's, including both the original one and the adding sub-objects.

The global coding matrix $\mathbf{Z}$ is obtained by overlaying the n matrices $\mathbf{Z}_i$ (with $1 \leq i \leq n$). The matrix $\mathbf{Z}$ has $N = n$ x 2^p rows and p columns, and it represents the matrix of associated to the numerical representation of n SO's.

Without loss of generality we can also assume that the z_j variables are standardized.

To relate the vertices of the same hypercube, a vertices cohesion constraint is defined in the analysis. The latter is introduced by the indicator matrix $\mathbf{A}$ (N x n), which describes the belonging of the N vertices to the n SO's.

Whereas a SO has been decomposed, in the numerical transformation phase, into sub-objects such that their descriptions are consistent with the condition imposed by logical rules. However, in order to preserve the unity of the decomposed SO, in the analysis, all the sub-objects will be considered as belonging to the same original SO. It consists to give them the same code in the indicator matrix $\mathbf{A}$.

The principal axes are obtained, in the space $\mathbf{R}^p$, as solutions of the following characteristic equation:

$$\frac{1}{N}\left[\mathbf{Z'A(A'A)}^{-1}\mathbf{A'Z}\right]\tilde{\mathbf{v}}_m = \frac{1}{N}\mathbf{Z'P_A Z} = \tilde{\lambda}_m \tilde{\mathbf{v}}_m$$

where: $\tilde{\mathbf{v}}_m$ is defined under the orthonormality constraints, already expressed in V-PCA. Considering that $\mathbf{P_A} = \mathbf{A(A'A)}^{-1}\mathbf{A'}$ is an orthogonal projector.

The coordinates of the vertices of the *i*-th SO on the axis m, are given by the vector of $\mathbf{R}^{2p}$:

$$\tilde{\psi}_{i,m} = Z_i \tilde{\mathbf{v}}_m$$

Notice that the coordinates can be directly obtained by the analysis in $\mathbf{R}^n$ based on the following eigen-equation:

$$(\mathbf{A'A})^{-1/2}(\mathbf{A'ZZ'A})(\mathbf{A'A})^{-1/2}\tilde{\mathbf{w}}_m = \tilde{\lambda}_m \tilde{\mathbf{w}}_m$$

Then, the SO's are represented by the minimum covering area rectangles with edges parallel to the axes.

The most immediate interpretation of factorial axes can be obtained with reference to the variables having maximal contributions, observing by means of correlations between axes and variables. In fact, the measures of descriptors contribution to the axes are expressed, as in the classical PCA, as the squared correlation variable/factor, see Lebart et al. (1995).

In the choice of analysis strategy is possible to select the following options:

1) Vertices SPCA: is the PCA performed on the vertices of objects under cohesion constraint.

2) RT-PCA (Range transformation PCA) perform a transformation of the range of all intervals in order to analyze the size and the shape of objects without their position

3) MIXED STRATEGY combines Vertices SPCA and RT-PCA in order to improve SO's representation taking into account their differences in terms of scale and structural (*size* and *shape*) characteristics. (Lauro Palumbo, 2001)

5 Examples

In the following example Vertices SPCA, RT-PCA and Mixed strategy PCA is performed on the file "car.xml", which describes 11 characteristics of 33 kind of cars. As SPCA works only on interval variables, in the variable selection window, only the interval variables selection is allowed. Once ASSO/SODAS 2 software starts after the selection of the base file (car.xml). We have to select "Factorial Methods" and to drag and drop the icon "SPCA" into the chain, parameterize and run the method. If the analysis has been performed two icons appears to the right of "SPCA", the first one represent the text output, the latter the graphical one.

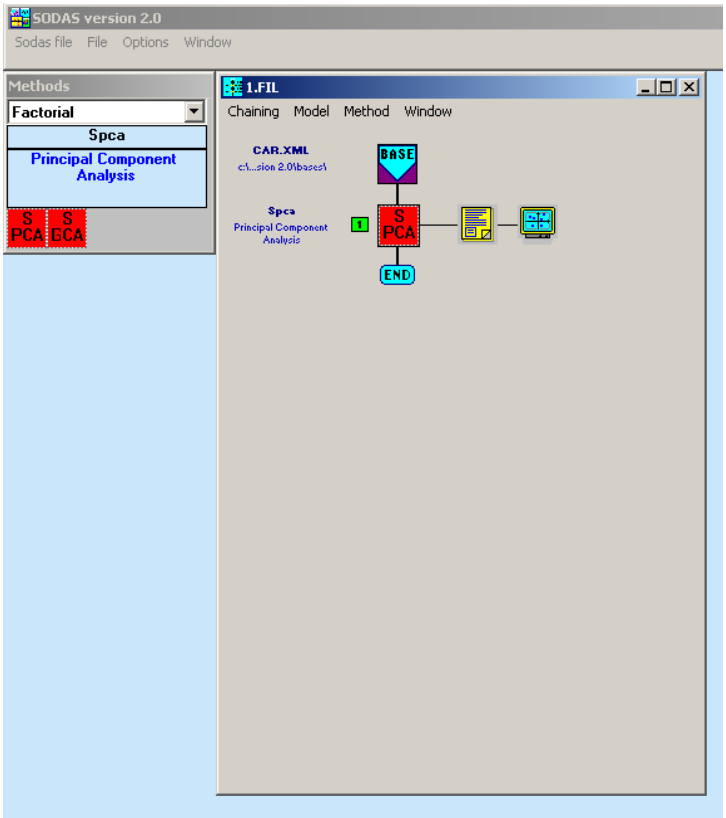


Figure 2: Asso workbench for SPCA

According to the selected strategy we may have the following textual and graphical results:

1) **Vertices SPCA**

The text output file is composed by the following section:

a) the selection of SO's and variables;
b) the eigenvalues of the analysis with the explained inertia percentage and histograms;
c) the SO's interval coordinates on factorial axes;
d) the absolute contributions of the SO's to the axes;
e) the quality measure of the SO's representation on each axis;
f) the point correlations between variables and factors;
g) the interval correlations between variables and factors.

```
      ****** PRINCIPAL COMPONENTS ANALYSIS ******


   File:C:\DOCUMENTS AND SETTINGS\NOTEBOOK\DESKTOP\SOFT LAST\EXECUTABLES\CAR.XML

 SELECTIONS
   Symbolic Object:33
      "Alfa  145" ==> "AA00"
      "Alfa  156" ==> "AA01"
      "Alfa  166" ==> "AA02"
      "Aston Martin" ==> "AA03"
      "Audi A3" ==> "AA04"
      "Audi A6" ==> "AA05"
```

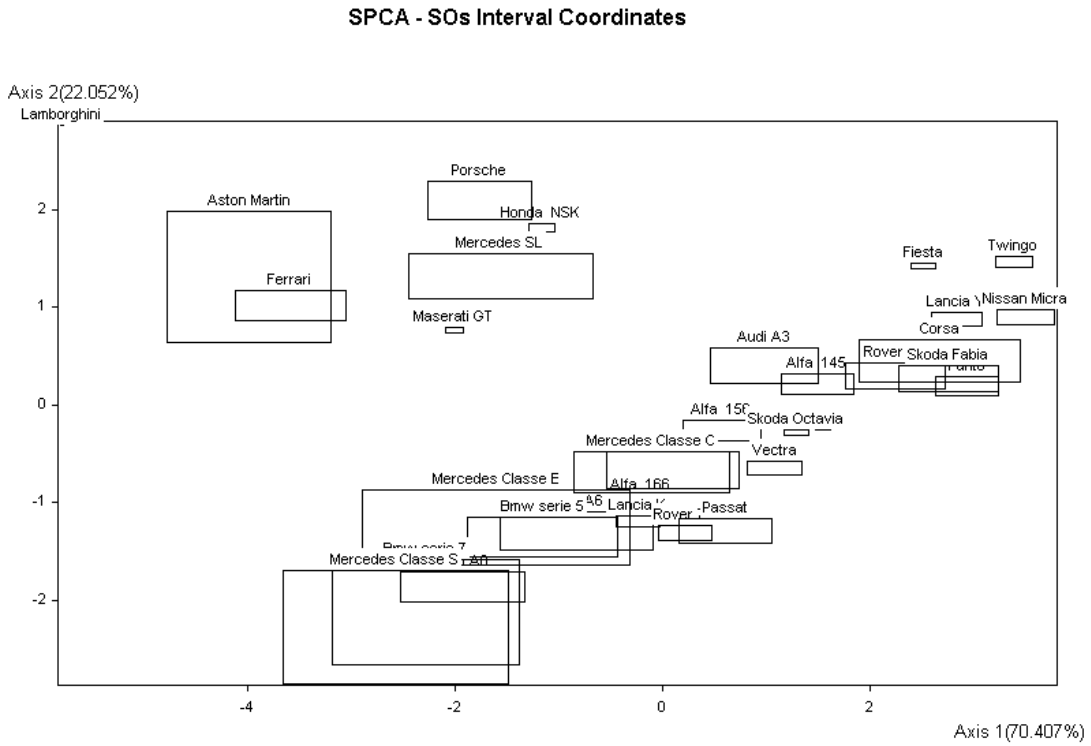
```
        "Audi A8" ==> "AA06"
        .......
        "Skoda Octavia" ==> "AA31"
        "Passat" ==> "AA32"


     Variables:8
        Price ==> AB00
        Cubic_Capacity ==> AC00
        Max_Speed ==> AF00
        Acceleration ==> AG00
        Axes_distance ==> AH00
        Length ==> AI00
        Width ==> AJ00
        Heigth ==> AK00


        SO-PCA

Eigenvalues        Inertia         %        Cum. %       Histogram
     L1         5.13353        70.40707       70.40707      ____________________________________________
     L2         1.60784        22.05174       92.45880      ________________
     L3         0.21330        2.92551        95.38431      ___
     L4         0.13040        1.78841        97.17272      __
     L5         0.10417        1.42875        98.60146      __
     L6         0.07301        1.00140        99.60287      __
     L7         0.01546        0.21197        99.81483      _
     L8         0.01350        0.18516        100.00000     _

SPCA   Coordinates [Min;Max]  (33,8)=
Objects             Factor 1                Factor 2                      Factor 3
     Factor 4
"Alfa  145"          [ 1.14100;   1.84780]  [ 0.11082;   0.31397] [-0.06432;   0.82334]
"Alfa  156"          [ 0.19459;   0.94696]  [-0.37174;  -0.15413] [-0.01600;   0.86767]
"Alfa  166"          [-0.40740;  -0.00618]  [-1.02382;  -0.92307] [-0.15226;   0.25788]
"Aston Martin"       [-4.78288;  -3.19601]  [ 0.64490;   1.98084] [-1.02928;   0.24687]
"Audi A3"            [ 0.46268;   1.50063]  [ 0.21496;   0.58893] [ 0.00201;   1.31515]
"Audi A6"            [-1.56305;  -0.09336]  [-1.49132;  -1.09285] [-0.44989;   1.21015]
"Audi A8"            [-2.52382;  -1.33335]  [-2.01762;  -1.70759] [-0.65930;   0.80531]
"Bmw serie 3"        [-0.53627;   0.74511]  [-0.85850;  -0.48360] [-0.46883;   1.08710]
.....
"Skoda Octavia"      [ 1.17990;   1.41362]  [-0.31171;  -0.24863] [ 0.12627;   0.40022]
"Passat"             [ 0.16241;   1.05809]  [-1.41552;  -1.16101] [-0.39022;   0.69902]

Contributions of the SOs to the axes  (33,8)=
Objects             Factor 1  Factor 2       Factor 3       Factor 4
"Alfa  145"           0.01292  0.00086        0.01396        0.00088
"Alfa  156"           0.00202  0.00135        0.01648        0.00689
"Alfa  166"           0.00045  0.01779        0.00168        0.05399
"Aston Martin"        0.09333  0.03478        0.01829        0.10521
"Audi A3"             0.00559  0.00306        0.03595        0.00051
"Audi A6"             0.00489  0.03154        0.03065        0.01447
"Audi A8"             0.02154  0.06499        0.02447        0.00704
"Bmw serie 3"         0.00021  0.00854        0.01613        0.00039
"Bmw serie 5"         0.00896  0.03467        0.02654        0.03377
"Bmw serie 7"         0.03309  0.08516        0.06807        0.04015

Quality measure of the SOs representation  (33,8)=
Objects             Factor 1  Factor 2       Factor 3       Factor 4
```

```
"Alfa  145"           0.77065  0.01580        0.06293        0.00213
"Alfa  156"           0.38784  0.08040        0.23929        0.05376
"Alfa  166"           0.04974  0.60070        0.01387        0.23991
"Aston Martin"        0.81278  0.09355        0.01203        0.03723
"Audi A3"             0.40546  0.06858        0.19684        0.00150
"Audi A6"             0.23268  0.46355        0.11020        0.02798
"Audi A8"             0.45752  0.42633        0.03927        0.00607
"Bmw serie 3"         0.02091  0.26180        0.12100        0.00156
"Bmw serie 5"         0.35388  0.42296        0.07919        0.05420
"Bmw serie 7"         0.49095  0.39015        0.07630        0.02420
....
"Skoda Fabia"         0.87325  0.00875        0.03418        0.00373
"Skoda Octavia"       0.68491  0.03202        0.03481        0.07069
"Passat"              0.14275  0.61999        0.03766        0.02738

Correlations between variables and factors  (8,8)=
Var.        Factor 1   Factor 2       Factor 3       Factor 4
Price      -0.73021    0.26649        0.17508        0.41443
Cubic_Capa -0.72194    0.13489        0.23993        0.40853
Fuel       -0.83257    0.25093        0.30423        0.17792
Traction    0.89392   -0.05524        0.17721        0.20931
Max_Speed  -0.54232   -0.74332       -0.05354        0.03931
Accelerati -0.72384   -0.60038        0.07398       -0.04145
Axes_dista -0.91546   -0.07179        0.00670       -0.06990
Length      0.61689   -0.70317        0.08164        0.14402

Interval correlations between variables and factors  (8,8)=
Variables     Factor 1                Factor 2                      Factor 3
Price      [-0.99999;  -0.73021]     [ 0.26649;   1.00000] [-0.99999;   0.17508]
Cubic_Capa [-0.93934;  -0.62981]     [ 0.13489;   0.93841] [-0.93929;   0.23993]
Fuel       [-0.88674;  -0.66030]     [ 0.25093;   0.88498] [-0.88632;   0.30423]
Traction   [ 0.53768;   0.89392]     [-0.82684;  -0.05524] [ 0.17721;   0.82854]
Max_Speed  [-0.54232;  -0.09762]     [-0.74332;   0.53208] [-0.53352;  -0.05354]
Accelerati [-0.72384;  -0.34197]     [-0.60038;   0.58175] [-0.58412;   0.07398]
Axes_dista [-0.91546;  -0.70220]     [-0.07179;   0.83255] [-0.83393;   0.00670]
Length     [ 0.52143;   0.83804]     [-0.83931;  -0.52138] [ 0.08164;   0.83825]
```

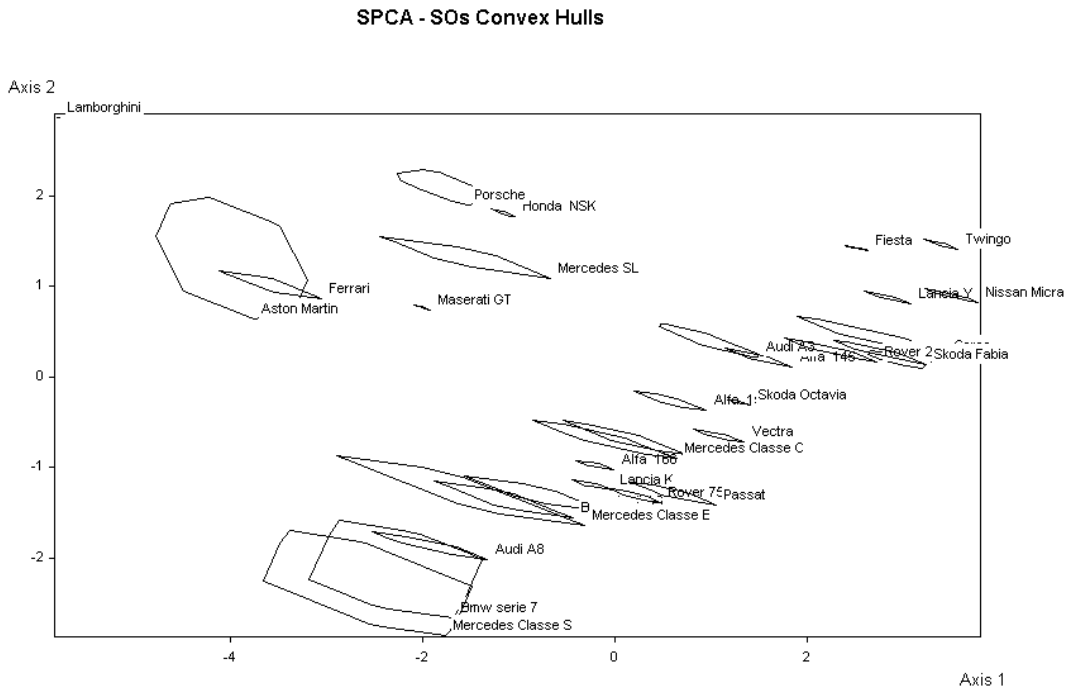
Figure 3: Textual output for Vertices SPCA

In **Vertices SPCA** we have four graphical output:

i) The SO's rectangular plot: the objects are represented on the factorial planes by the minimum and maximum coordinates as described in the text output file.

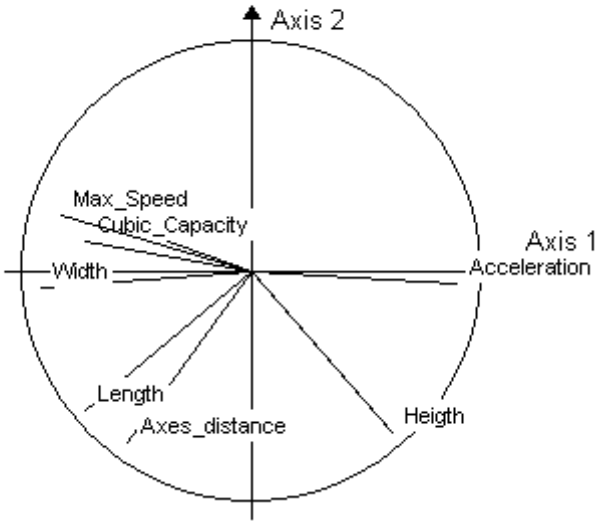


ii) The SO's convex hull plot: selected two axes in the parameters window, the SO's are represented by convex hulls of the original hypercube vertices projected on the factorial plane. This kind of visualization corresponds to the smallest convex polygon including all the hypercube vertices of an hypercube projected on the factorial plane, while rectangular representation corresponds to the minimum rectangle area containing the projection of the hypercube vertices on the factorial plane.



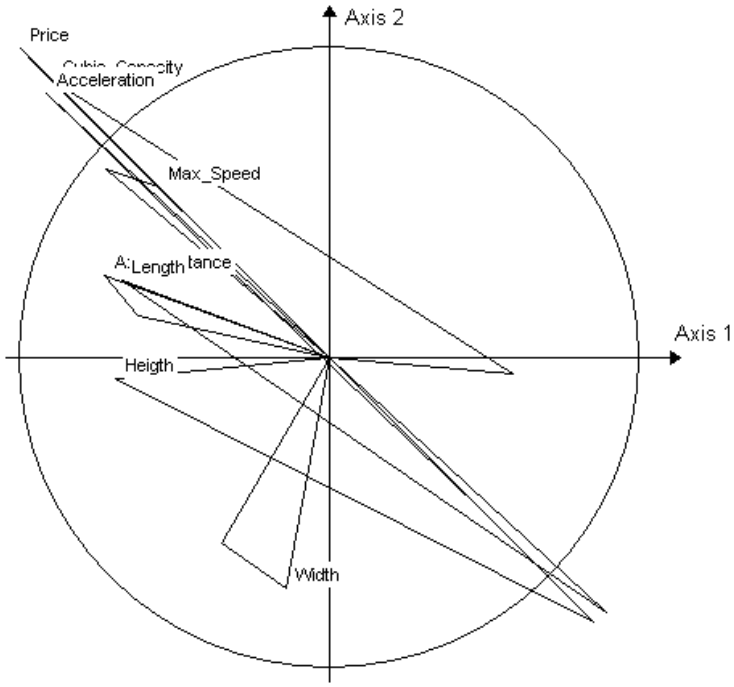
iii) Circle of correlation plot: as for classical PCA, the original descriptors are represented according to their correlation with the factorial axes in the unit circle.

SPCA - Variables circle of correlations



iv) interval correlation plot: in this plot are represented the interval correlations of each variable with the factorial axes.

SPCA - Variables Int Coordinates



2) **RT PCA** (Range Transformation) strategy

The text output file is composed by the following section:

a) the selection of SO's and variables;

b) the eigenvalues of the analysis with the explained inertia percentage and histograms;

c) the SO's max point coordinates on factorial axes;
d) the absolute contributions of the SO's to the axes;
e) the quality measure of the SO's representation on each axis;
f) the point correlations between variables and factors.

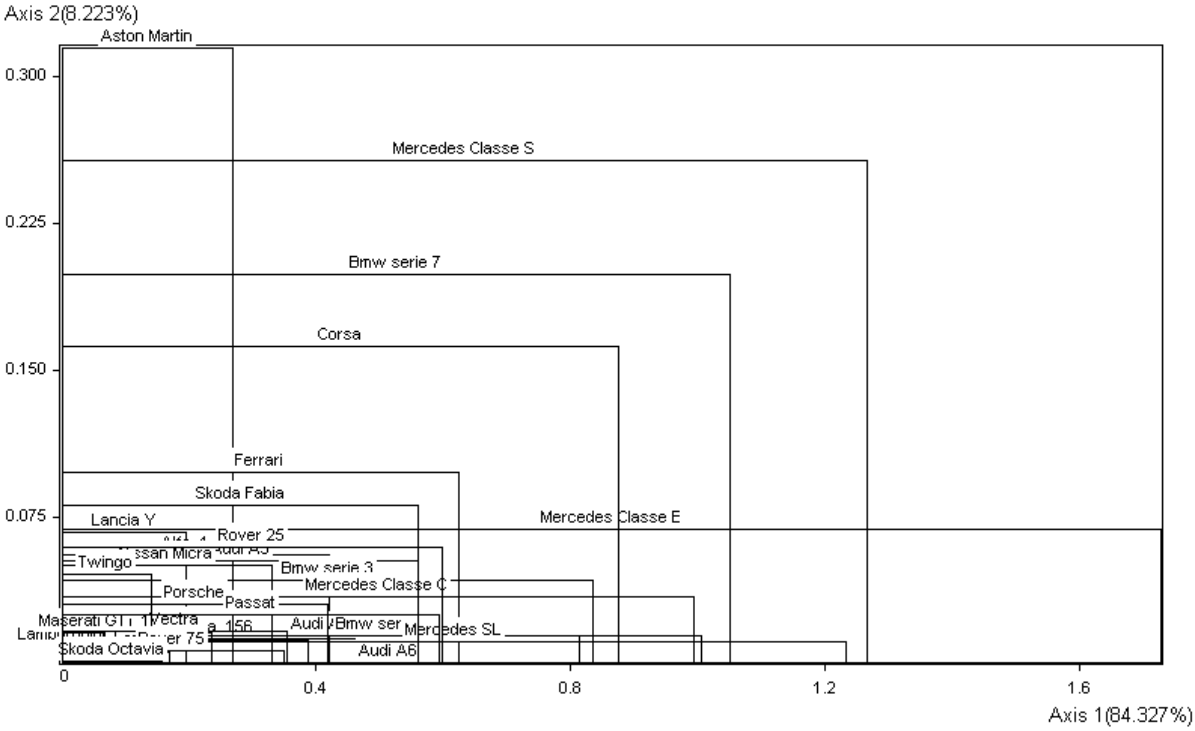
```
                ****** PRINCIPAL COMPONENTS ANALYSIS ******

        File:C:\DOCUMENTS AND SETTINGS\NOTEBOOK\DESKTOP\SOFT LAST\EXECUTABLES\CAR.XML

  SELECTIONS
        Symbolic Object:33
                "Alfa  145" ==> "AA00"
                "Alfa  156" ==> "AA01"
                "Alfa  166" ==> "AA02"
                "Aston Martin" ==> "AA03"
                "Audi A3" ==> "AA04"
                "Audi A6" ==> "AA05"
                "Audi A8" ==> "AA06"
                "Bmw serie 3" ==> "AA07"
                "Bmw serie 5" ==> "AA08"
                "Bmw serie 7" ==> "AA09"
                ......
                "Passat" ==> "AA32"


        Variables:8
                Price ==> AB00
                Cubic_Capacity ==> AC00
                Max_Speed ==> AF00
                Acceleration ==> AG00
                Axes_distance ==> AH00
                Length ==> AI00
                Width ==> AJ00
                Heigth ==> AK00


                RT-PCA

Eigenvalues        Inertia         %        Cum. %       Histogram
L1              18.27105       84.32661       84.32661      ______________________________
L2              1.78177        8.22342        92.55003      _
L3              1.00570        4.64162        97.19166      _
L4              0.31282        1.44375        98.63541
L5              0.13176        0.60813        99.24354
L6              0.12037        0.55554        99.79908
L7              0.03649        0.16839        99.96747
L8              0.00705        0.03254        100.00000

RT-PCA   Coordinates  (33,8)=
Objects             Factor 1  Factor 2       Factor 3       Factor 4
"Alfa  145"           0.42173  0.05575        0.00096        0.00081
"Alfa  156"           0.50188  0.01285        0.00075        0.00002
"Alfa  166"           0.23668  0.01555        0.02720        0.00219
"Aston Martin"        0.26799  0.31412        0.47927        0.00581
"Audi A3"             0.56022  0.05273        0.05522        0.01035
"Audi A6"             1.02464  0.00007        0.01769        0.00242
"Audi A8"             0.81462  0.01413        0.00260        0.00092
```

```
.....
"Skoda Octavia"       0.15676  0.00111        0.00219        0.00171
"Passat"              0.59456  0.02513        0.00000        0.00000

Contributions of the SOs to the axes  (33,8)=
Objects             Factor 1  Factor 2       Factor 3       Factor 4
"Alfa  145"           0.02308  0.03129        0.00095        0.00260
"Alfa  156"           0.02747  0.00721        0.00075        0.00005
"Alfa  166"           0.01295  0.00873        0.02705        0.00699
"Aston Martin"        0.01467  0.17630        0.47655        0.01858
"Audi A3"             0.03066  0.02959        0.05491        0.03310
"Audi A6"             0.05608  0.00004        0.01759        0.00773
"Audi A8"             0.04459  0.00793        0.00258        0.00294
"Bmw serie 3"         0.04567  0.02390        0.00030        0.00008
.......
"Skoda Octavia"       0.00858  0.00062        0.00218        0.00547
"Passat"              0.03254  0.01410        0.00000        0.00001

Quality measure of the SOs representation  (33,8)=
Objects             Factor 1  Factor 2       Factor 3       Factor 4
"Alfa  145"           0.87392  0.11552        0.00198        0.00168
"Alfa  156"           0.95876  0.02454        0.00144        0.00003
"Alfa  166"           0.79612  0.05232        0.09150        0.00736
"Aston Martin"        0.24724  0.28980        0.44216        0.00536
"Audi A3"             0.80841  0.07609        0.07968        0.01494
"Audi A6"             0.97417  0.00007        0.01682        0.00230
"Audi A8"             0.96681  0.01677        0.00308        0.00109
"Bmw serie 3"         0.94560  0.04825        0.00034        0.00003
"Bmw serie 5"         0.96132  0.01320        0.01518        0.00824
......
"Skoda Octavia"       0.92871  0.00660        0.01297        0.01014
"Passat"              0.95714  0.04045        0.00001        0.00000

Correlations beetween variables and factors  (8,8)=
Var.       Factor 1   Factor 2       Factor 3       Factor 4
Price       0.74260    0.55876        0.36271        0.65550
Cubic_Capa  0.86191    0.07137       -0.31727        0.43061
Fuel        0.58622    0.15546       -0.09771        0.06540
Traction    0.58671    0.04437       -0.24441       -0.07940
Max_Speed   0.25564    0.84664        0.45155        0.76253
Accelerati  0.27564    0.65932        0.18312        0.85620
Axes_dista -0.13509    0.45385        0.88332        0.13180
Length     -0.13779    0.57410        0.97727        0.05278
```

In **RT PCA** we have two graphical output:

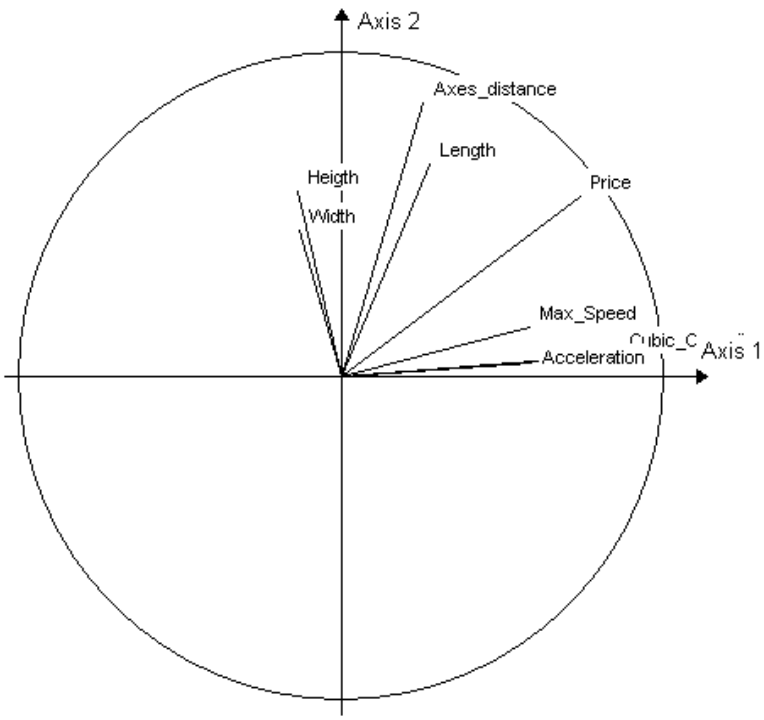
i) The SO's rectangular plot: the objects are represented onto the factorial planes by means of their minimum and maximum coordinates as described in the text output file.

SPCA - RT Object Coordinates



ii) Circle of correlation plot: as for classical PCA, the original descriptors are represented by means of their correlation with the factorial axes in the unit circle.

SPCA - Variables circle of correlations



3) **Mixed Strategy**

Mixed Strategy combines Vertices SPCA and Range Transformation SPCA. The text output reports the following results:

a) the selection of SO's and variables;
b) the eigenvalues of the analysis with the explained inertia percentage and histograms;
c) the SO's interval coordinates on factorial axes;
d) the absolute contributions of the SOs to the axes;
e) the quality measure of the SOs representation on each axis;
f) the point correlations between variables and factors;
g) the interval correlations between variables and factors.

```
                ****** PRINCIPAL COMPONENTS ANALYSIS ******

        File:C:\DOCUMENTS AND SETTINGS\NOTEBOOK\DESKTOP\SOFT LAST\EXECUTABLES\CAREN.XML

  SELECTIONS
        Symbolic Object:33
                "Alfa  145" ==> "AA00"
                "Alfa  156" ==> "AA01"
                "Alfa  166" ==> "AA02"
                "Aston Martin" ==> "AA03"
                "Audi A3" ==> "AA04"
                "Audi A6" ==> "AA05"
                "Audi A8" ==> "AA06"
                "Bmw serie 3" ==> "AA07"
                "Bmw serie 5" ==> "AA08"
                "Bmw serie 7" ==> "AA09"
                ......
                "Passat" ==> "AA32"


        Variables:8
                Price ==> AB00
                Cubic_Capacity ==> AC00
                Max_Speed ==> AF00
                Acceleration ==> AG00
                Axes_distance ==> AH00
                Length ==> AI00
                Width ==> AJ00
                Heigth ==> AK00



                SO-PCA

.........(results of Vertices SPCA)


                RT-PCA

...... (Results of Range transformation SPCA)
```

```
                MIX-PCA

Eigenvalues        Inertia         %        Cum. %       Histogram
L1              1.69028        93.41035       93.41035      ________________________________
L2              0.08500        4.69762        98.10796      _
L3              0.02183        1.20656        99.31452      _
L4              0.01240        0.68547        99.99999
L5              0.00000        0.00000        99.99999
L6              0.00000        0.00000        99.99999
L7              0.00000        0.00000        99.99999
L8              0.00000        0.00000        100.00000

MIXPCA   Coordinates [Min;Max]  (33,8)=
Objects             Factor 1                 Factor 2                      Factor 3
"Alfa  145"          [-0.00039;   0.52688]  [-0.96430;  -0.08138] [ 0.52266;   1.18234]
"Alfa  156"          [-0.49384;   0.07814]  [-0.74913;   0.05065] [ 0.18891;   0.79967]
"Alfa  166"          [-1.16480;  -0.85042]  [-0.25646;   0.01673] [ 0.18471;   0.37323]
"Aston Martin"       [ 0.53772;   1.94973]  [ 1.46828;   2.67576] [-3.01119;  -1.31043]
"Audi A3"            [ 0.01510;   0.81463]  [-0.97595;   0.33704] [ 0.07595;   1.20274]
"Audi A6"            [-1.28774;  -0.16142]  [-0.82658;   0.54088] [-0.13969;   0.90803]
"Audi A8"            [-1.81111;  -0.92089]  [-0.45582;   0.89693] [-0.51635;   0.51528]
"Bmw serie 3"        [-0.91780;   0.04775]  [-0.98351;   0.49107] [-0.13443;   1.00944]
"Bmw serie 5"        [-1.29215;  -0.17205]  [-0.59379;   0.64309] [-0.39723;   0.72670]
"Bmw serie 7"        [-2.41679;  -0.63989]  [-0.06900;   1.05396] [-0.80709;   0.35067]
.........
"Skoda Octavia"      [-0.39308;  -0.21592]  [-0.78358;  -0.55734] [ 0.48398;   0.68607]
"Passat"             [-1.25947;  -0.58551]  [-1.36970;  -0.34483] [ 0.18589;   0.97251]


Contributions of the SOs to the axes  (33,8)=
Objects             Factor 1  Factor 2       Factor 3       Factor 4
"Alfa  145"           0.00271  0.00939        0.01867        0.02003
"Alfa  156"           0.00231  0.00415        0.00748        0.01070
"Alfa  166"           0.03985  0.00041        0.00191        0.00500
"Aston Martin"        0.05941  0.10185        0.10587        0.14049
"Audi A3"             0.00669  0.01031        0.01625        0.01953
"Audi A6"             0.02431  0.00145        0.00905        0.03876
"Audi A8"             0.07190  0.00465        0.00573        0.02192
"Bmw serie 3"         0.00810  0.00752        0.01140        0.02138
"Bmw serie 5"         0.02591  0.00010        0.00728        0.01610
"Bmw serie 7"         0.09054  0.00624        0.00818        0.02041
........
"Skoda Octavia"       0.00361  0.01065        0.00789        0.00565
"Passat"              0.03303  0.02007        0.01087        0.01385

Quality measure of the SOs representation  (33,8)=
Objects             Factor 1  Factor 2       Factor 3       Factor 4
"Alfa  145"           0.02438  0.13690        0.28731        0.05894
"Alfa  156"           0.06669  0.19483        0.37062        0.10140
"Alfa  166"           0.65618  0.01084        0.05399        0.02697
"Aston Martin"        0.07791  0.21688        0.23782        0.06037
"Audi A3"             0.07300  0.18280        0.30374        0.06985
"Audi A6"             0.17424  0.01689        0.11108        0.09103
"Audi A8"             0.22997  0.02416        0.03137        0.02297
..........
"Skoda Octavia"       0.03820  0.18328        0.14324        0.01960
"Passat"              0.32083  0.31656        0.18091        0.04410

Correlations beetween variables and factors  (8,8)=
```

Var.	Factor 1	Factor 2	Factor 3	Factor 4
Price	0.31476	0.90315	-0.62147	-0.08267
Cubic_Capa	0.21771	0.90214	-0.55225	-0.04583
Fuel	0.25069	0.91720	-0.68546	-0.28490
Traction	0.08584	-0.52230	0.80004	0.64838
Max_Speed	-0.68413	0.25396	-0.12700	-0.12306
Accelerati	-0.56315	0.41997	-0.31673	-0.25996
Axes_dista	-0.15710	0.76899	-0.69583	-0.42004
Length	-0.48206	-0.76749	0.85106	0.45036

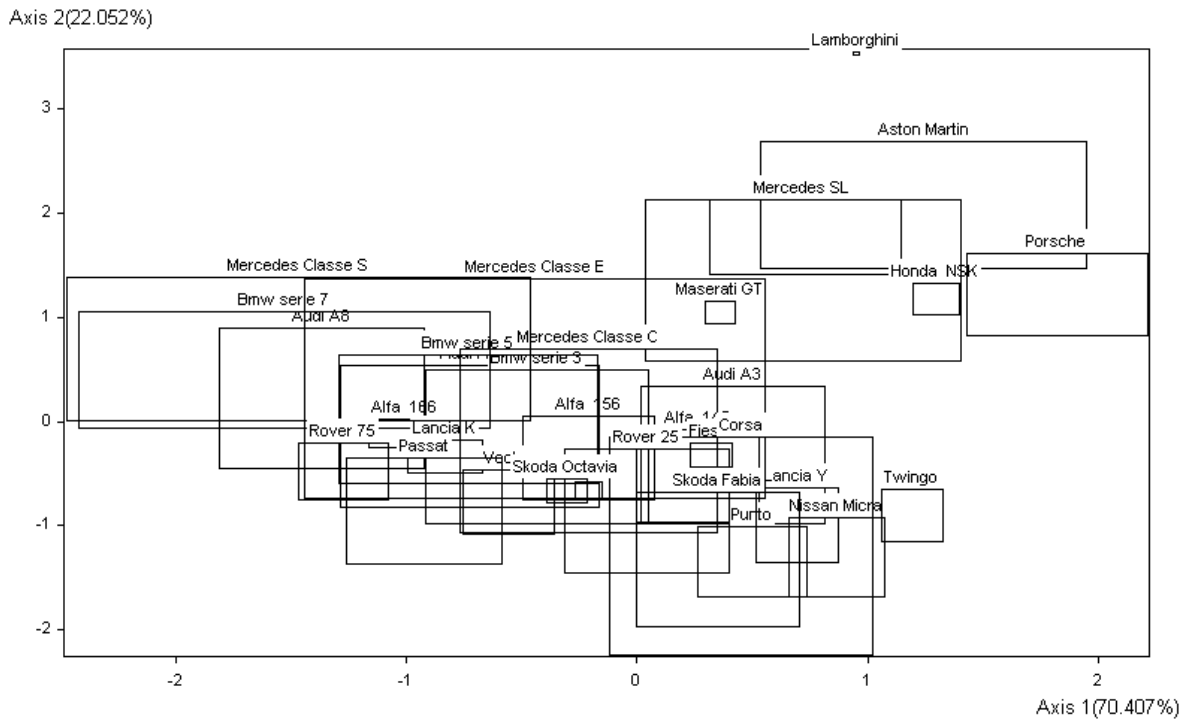
Interval correlations beetween variables and factors (8,8)=

Variables	Factor 1	Factor 2	Factor 3
Price	[0.31476; 0.99999]	[0.90315; 0.99999]	[-1.00000; -0.62147]
Cubic_Capa	[0.21771; 0.93509]	[0.62852; 0.93522]	[-0.93374; -0.55225]
Fuel	[0.25069; 0.90618]	[0.68598; 0.91720]	[-0.90423; -0.68546]
Traction	[-0.80473; 0.08584]	[-0.80475; -0.52230]	[0.59460; 0.80520]
Max_Speed	[-0.68413; 0.52555]	[0.10965; 0.52610]	[-0.52513; -0.10462]
Accelerati	[-0.56315; 0.59846]	[0.35991; 0.59891]	[-0.59750; -0.31673]
Axes_dista	[-0.15710; 0.85282]	[0.67736; 0.85336]	[-0.84901; -0.67466]
Length	[-0.84733; -0.48206]	[-0.84706; -0.51725]	[0.51529; 0.85106]

In Mixed Strategy SPCA we have four graphical output:

i) The SO's rectangular plot: the objects are represented on the factorial planes by minimum and maximum coordinates as described in the text output file.

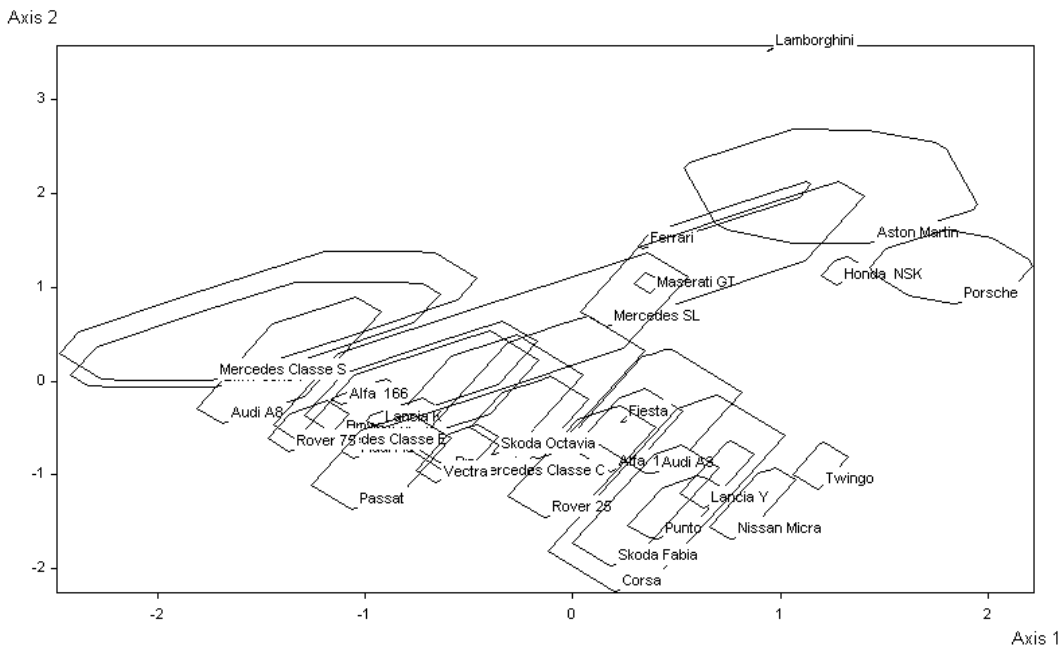
SPCA - SOs Interval Coordinates



ii) The SO's convex hull plot: selected two axes in the parameters window, the SO's are represented by convex hulls of the original hypercube vertices projected on the factorial plane. This kind of visualization corresponds to the smallest convex polygon including all the

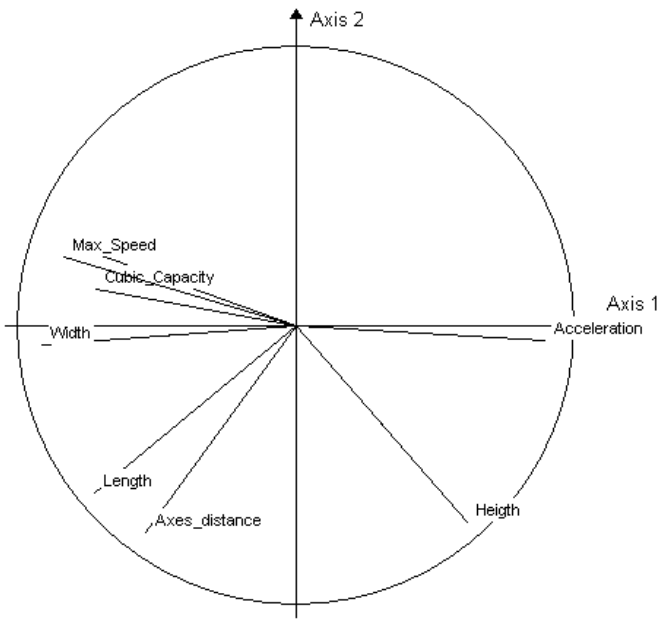
hypercube vertices of an hypercube projected on the factorial plane, while rectangular representation corresponds to the minimum rectangle area containing the projection of the hypercube vertices on the factorial plane.

SPCA - SOs Convex Hulls



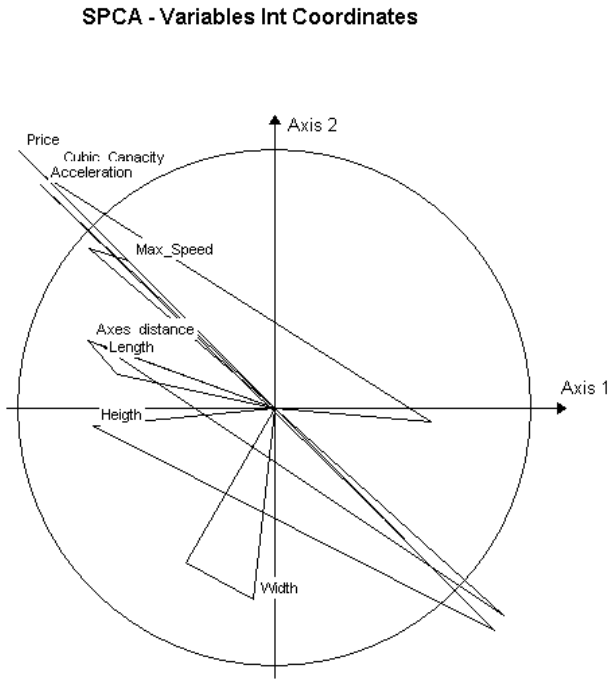
iii) Circle of correlation plot: as for classical PCA, the original descriptors are represented by means of their correlation with the factorial axes in an unit circle.

SPCA - Variables circle of correlations



iv) Interval correlation plot: in this plot are represented the interval correlations of each

variable with the factorial axes.



References

Bock, H. H., Diday, E. (eds) (2000), Analysis of Symbolic Data, Springer-Verlag. Heidelberg.

Cazes, P., Chouakria, A., Diday, E., Schektman, Y.: 1997, Extension de l'analyse en composantes principales à des données de type intervalle, *Revue de Statistique Appliquée* XIV(3).

D'Ambra, L., Lauro, C. : 1982, Analisi in componenti principali in rapporto a un sottospazio di riferimento*, Rivista di Statistica Applicata* 15(1), 51-67.

Lauro, C., Palumbo, F.: 2000, Factorial Methods with cohesion constraints on Symbolic Objects. In Proceeding of IFCS2000, Springer-Verlag, .

Lauro, C., Palumbo, F.: 2001, Principal Component Analysis of Interval Data: a Symbolic Data Analysis Approach, *Computational Statistics*.

Lebart, L., Morineau, A. and Piron, M.: 1995, Statistique exploratoire multidimensionnelle, Dunod, Paris.

Verde, R.: 1997, Symbolic object decomposition by factorial techniques. Indo-French Meeting, LISE-CEREMADE, Université Paris IX Dauphine.

Verde, R. and De Angelis, P.: 1997, Symbolic objects recognition on a factorial plan, NGUS'97, Bilbao Spain.

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

SFDR Tutorial

Factorial Discriminant Rule

Edited by DMS

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **03/02/2004**

SFDR : Factorial Discriminant Rule

1 Introduction

The aim of the SFDR Factorial Discriminant Rule application on a new set of Symbolic Objects is to use the geometrical rule achieved by SFDA to affect these objects to the *r* classes. SFDA aims at defining the factorial sub-space which discriminates at the best the *a priori* classes defined in the SO training set descriptions. The procedure, as shown in the SFDA tutorial, is performed by a quantification and selection of the SO's descriptors following by FDA on the quantified new descriptors. The SGCA has been proposed as the quantification procedure of the coded descriptors.

2 The input to SFDR

SFDR is performed on a set of SO's, described by the same set of descriptors than the training set SO's in SFDA. As input file SFDR requests also a .fdr file which contains the rule and the intermediate quantification parameter for the projection of the new objects on the factorial discriminant axes, that is an output of SFDA module. Mixed variables, logical dependence rules, taxonomies and missing values (NULL) are admitted.

3 The output of SFDR

SFDR results are: the coordinates of the vertices of the hypercubes (SO's and classes); the coordinates of the extremes of the intervals of the rectangles which reproduce the hypercube images on each factorial axis, and the classification tables of the new objects.

4 SFDA method

It is constituted by the following steps:

i) a quantification of the new SO's descriptors;

ii) the projection of the new objects on the subspace identified by the previous SFDA;

iii) the assignment of the new objects to the classes.

The first phase of SFDR procedure consists in a SO descriptors quantification on the base of their nature and on the base of the quantification parameters of the SGCA quantification step performed in SFDA.

The second phase is the projection of the new objects on the factorial subspace carried out by the previous SFDA.

The coordinates of the hypercubes vertices, associated to each SO, are computed as follows: $\Psi_m = \tilde{\Phi}\upsilon_m$.

The last phase is represented by the assignment of the new objects to a class on the basis of one of the available geometrical classification rules. We consider that both SO's and classes are represented on factorial plan by rectangles. The classification of a generic SO to a class C_i is defined according to two events:

i) if a SO image is included in the representation of C_i (of the training set) it is assigned to the same class of C_i;

ii) if a SO is partially or completely outside to any C_i representation, or it is in an overlapping area between two or more classes representations, we are going to consider a suitable measure of similarity to assign the element to only one class. The new element will be assigned to the class according to the highest similarity value of the assignment function.

For defining a geometric classification rule, we propose three approaches, two based on the Description Potential π(.) measure, defined by De Carvalho (1992) as the volume obtained by the Cartesian product of the SO descriptors values; whereas the last one is based on the Hausdorff distance between polygons.

In particular, the three geometrical classification rules need the specification of the following parameters:

1) *Ichino - De Carvalho's dissimilarity index*: it is based on the following measure between projected objects represented as hypercubes:

$d(\hat{\omega}_j, \hat{\omega}_s) = \sqrt[m]{\sum_{\alpha} \left(p_\alpha \psi_\alpha \left(\hat{\omega}_j, \hat{\omega}_s \right) \right)^m}$ where $\hat{\omega}_j$ and $\hat{\omega}_s$ are the factorial representation of two SO *j* and *s*, p_α is the α-*th* eigenvalue and *m* is the number of retained factorial axes,

$$\psi_\alpha \left(\hat{\omega}_j, \hat{\omega}_s \right) = \frac{\mu\left(S_\alpha^s \oplus S_\alpha^j\right) - \mu\left(S_\alpha^s \cap S_\alpha^j\right) + \gamma\left(2\mu\left(S_\alpha^s \cap S_\alpha^j\right) - \mu\left(S_\alpha^s\right) - \mu\left(S_\alpha^j\right)\right)}{\mu\left(S_\alpha^s \oplus S_\alpha^j\right)},$$

$\mu\left(S_\alpha^j\right)$ is the length of the coordinates interval of object *j* on the α-th axis, $\mu\left(S_\alpha^s \oplus S_\alpha^j\right)$ is the length of the coordinates interval of the conjunction of the two coordinates intervals of the objects *j* and *s* on the α-th axis and $\mu\left(S_\alpha^s \cap S_\alpha^j\right)$ is the length of the intersection of the coordinates intervals of the two objects on the α-th axis.

In order to compute the dissimilarity between two objects it needs to choice a suitable metric (1=city block, 2=Euclidean,...), a bound γ (in]0;1[) that allows to emphasize the intersection between objects.

2) *Descriptor Potential Increase* index: this measure is based on the concept of potential descriptor increasing. It is based on the computing of how a class image has to enlarge in order to contain the object to be affected. The dpi is given by the following formula:

$$\mathrm{dpi}\left(\hat{\omega}_j, \hat{\omega}_s\right) = \frac{\prod_{\alpha=1}^{m} \mu\left(S_\alpha^s \oplus S_\alpha^j\right) - \prod_{\alpha=1}^{m} \mu\left(S_\alpha^j\right)}{\prod_{\alpha=1}^{m} \mu\left(S_\alpha^j\right)}$$

Where $\hat{\omega}_j$ is the representation of the *j*-th class, $\hat{\omega}_s$ is the representation of the object s to be affected, $\mu\left(S_\alpha^j\right)$ is the coordinates interval length of the object class *j* on the α-th axis and $\mu\left(S_\alpha^s \oplus S_\alpha^j\right)$ is the coordinates interval length of the conjunction of the two intervals which represent the objects on the α-th axis.

3) *Hausdorff distance*: it is based on the Hausdorff distance between two objects computed on the coordinates intervals on the factorial axes. This measure needs only the choice of the metric (1=city block, 2=Euclidean,...).

If the distance between a new SO *s* and a class is evaluated with respect to all the SO's belonging to such class (in 1) and 3)) , the classification criterion needs of choosing the single, average or complete linkage, as in classical clustering techniques.

5 Examples

No example is available until it will be not integrated in the workbench

References

Bock, H. H. and Diday, E. (eds): 2000, *Analysis of Symbolic Data*, Springer-Verlag, Heidelberg.

Goodman, L.A.,Kruskal, W.H. (1954). Measures of association for cross-classifications. *Journal of the American Statistical Association* 49, 732-764.

Lauro, C., Verde, R., Palumbo, F.: 2000, Factorial Discriminant Analysis on Symbolic Objects, in Bock, H.H. and Diday, E. (Eds): 1999, Analysis of Symbolic Data, Springer Verlag. Heidelberg.

Lauro, C., Palumbo, F.: 2000, Factorial Methods with cohesion constraints on Symbolic Objects. In Proceeding of IFCS2000, Springer-Verlag.

Lebart, L., Morineau, A. and Piron, M.: 1995, Statistique exploratorie multidimensionelle, Dunod, Paris.

Verde, R.: 1997, Symbolic object decomposition by factorial techniques. Indo-French Meeting, LISE-CEREMADE, Université Paris IX Dauphine.

Verde, R. and De Angelis, P.: 1997, Symbolic objects recognition on a factorial plan, NGUS'97, Bilbao Spain.

Verde, R. : 1999 Generalised Canonical Analysis on Symbolic Objects. In *Classification and Data Analysis, Theory and Application*. Vichi M. - Opitz O. (Eds), Springer-Verlag, Heidelberg, 195-202.

Verde, R. and Lauro, C.: 1993, Non symmetrical data analysis of multiway fuzzy coded matrices, ISI, Florence.

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

SGCA User Manual

Generalised Canonical Analysis

Edited by DMS

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: 0**3/02/2004**

SGCA : Generalised Canonical Analysis

1 Introduction

The Generalized Canonical Analysis extended to Symbolic Objects (GCA) aims to analyse SO's described by different kind of descriptors. It looks for the more suitable factorial axes in order to study the relationships among SO's and SO descriptors on a Cartesian plane. The analysis is performed on a coding matrix of the SO descriptions where each rows identifies the coordinates of the vertices of the hypercubes associated to each SO in the original representation space. As SPCA, in the analysis we consider a cohesion constraint in order to preserve the unitary of the SO in the analysis. The criterion optimizes is an additive criterion and each component is expression of the capability power of SO each descriptor.

The SGCA can be considering as a general Factorial Analysis procedure. For instance, the SGCA has been used in the intermediate, quantification step, of the Factorial Discriminant Analysis on SO's (see SFDA). As well as, a particular case is given by Multiple Correspondence Analysis when all the SO's are described by multi-nominal variables.

2 The input to SGCA

SGCA can be performed on a set of SO's described by any kind of descriptor ("quantitative single", "interval", "categorical single", "categorical multi-valued" and "modal"). Mixed variables, logical dependence rules, taxonomies and missing values (NULL) are admitted.

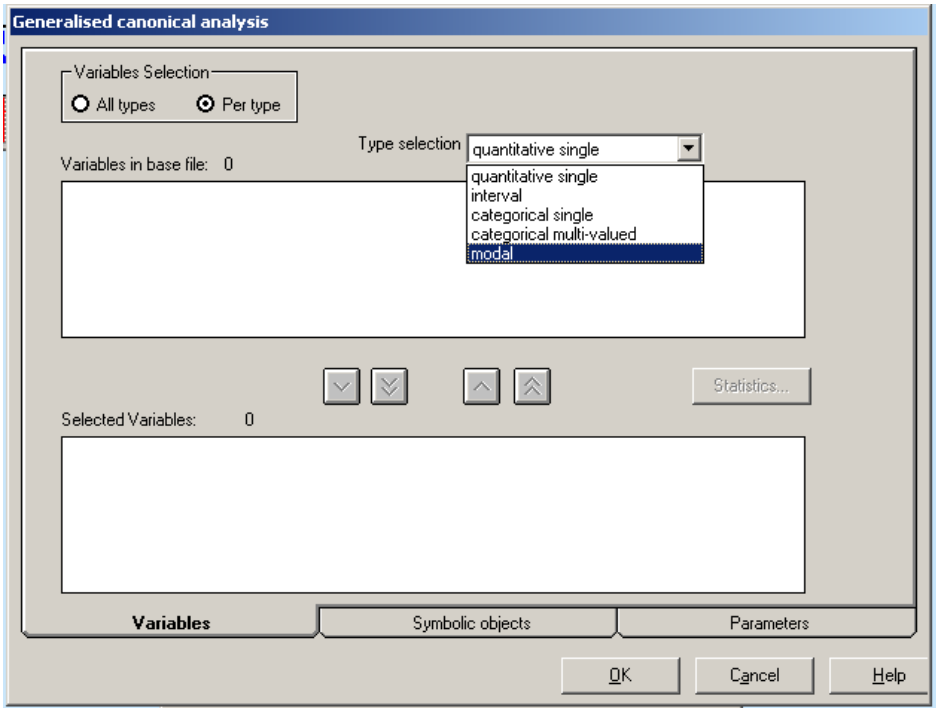


Figure 1: Variables Selection window

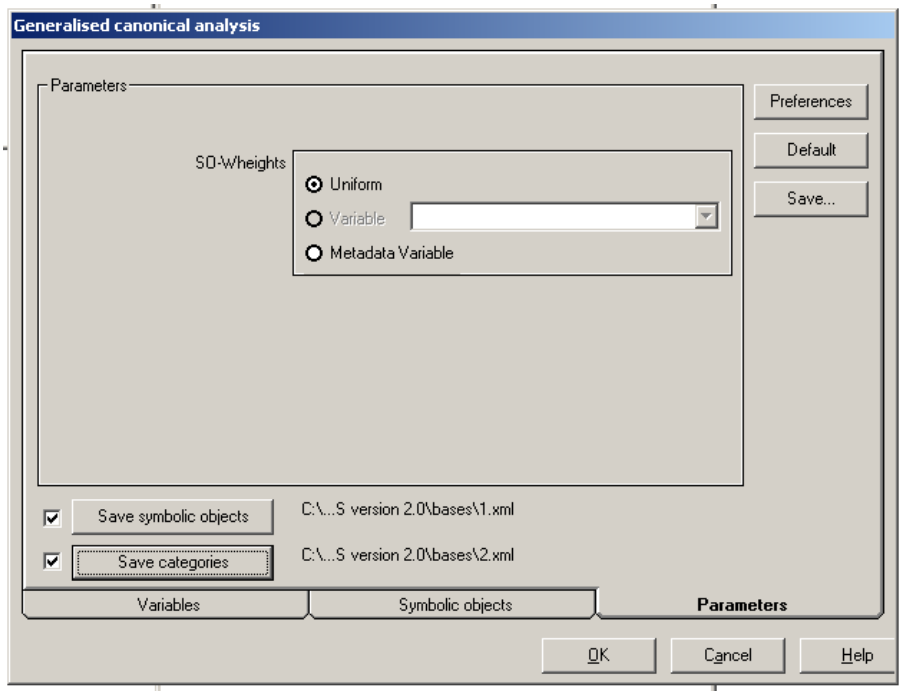


Figure 2: SGCA parameters window, the user have only to choose the weight variable

3 The output of SGCA

The SGCA results are: the coordinates of the extremes of the intervals of the rectangles which reproduce the hypercube images on each factorial axis and the coordinates of all the categories of the descriptors. The latter will be available in a .sds file. These results can be displayed by using VPLOT module. The other main results are: the absolute value and the percentage of inertia explained by each axis, some indexes to evaluate the quality of the representation of the SO's on the factorial plane, the contribution of each object and of each descriptor categories to the factorial axes.

4 SGCA method

SGCA as all the other factorial techniques on SO's is a *symbolic-numerical-symbolic* (Diday, 1987) and it is performed in the following steps:

i) a suitable coding of the SO descriptors;

ii) a factorial analysis on the coding matrix;

iii) a symbolic interpretation of the results according to the nature of the analysed data.

10.1.1 First phase: SO's descriptors coding of the symbolic object descriptors

In order to perform the factorial technique, based on the optimization of a numerical criterion, both numerical and nominal multi-valued SO descriptors have to be homogenised.

The first phase of the proposed approach consists in a numerical coding of the descriptors y_j's (for $j = 1,...,p$) of each SO in a table $\mathbf{Z}_{ij}$. The chosen system of coding differs according to the type of descriptor. In particular, if y_j is:

- a "categorical multi-valued" variable, then the coding table $\mathbf{Z}_{ij}$ is a binary matrix of values

(0/1). The i-th SO is coded with respect to k_{ij} rows of $\mathbf{Z}_{ij}$, being k_{ij} the categories of the descriptor y_j presented by the SO;

- a "modal" variable, then the i-th SO is coded in the table $\mathbf{Z}_{ij}$ according to the frequencies or probabilities that it assumes for the categories of y_j;

- a "quantitative single" or "interval" variable, then it is categorized and codified according to a fuzzy system of coding (e.g. using Basic splines functions)

which allows to preserve the numerical information of the original numerical variables. Usually, linear function are used to coded a numerical variable, with values in [0,1], with respect to three categories (e.g. low-medium-high). The two bounds of the intervals of values, assumed by the i-th SO are coded on two different rows of the coding matrix $\mathbf{Z}_{ij}$.

In the space of the coding descriptors, the geometrical representation of the generic SO i-th, in terms of hypercubes, is obtained by the Cartesian product of the $\mathbf{Z}_{ij}$ rows (for $j=1,...,p$) and it is denoting $\mathbf{Z}_{ij}$. By a vertical-concatenation of all the matrices $\mathbf{Z}_{ij}$ (for all the SO's: $i=1,...,n$) we obtain p matrices $\mathbf{Z}_j$ (for $j=1,...,p$), of dimensions $N\times k_j$, where: k_j is the number of categories of y_j when it is a "categorical multi-valued" or a "modal" variable; whereas, if y_j is numerical ("quantitative single", "interval") k_j is the number of categories (usually k_j=3) in which the variable has been categorized and coded according a fuzzy system. Finally the number N of rows depends on the number of variables of different type which influence the number of coding rows of each $\mathbf{Z}_{ij}$ in several way. Let's be q the number of "categorical multi-valued" variables and g the number of "interval" descriptors.

We remember that each SO is codified with respect to the "categorical multi-valued" on a number of rows of $\mathbf{Z}_{ij}$ equal to the categories k_{ij} of y_j that it assumes and for the "interval" variables on a number of rows of $\mathbf{Z}_{ij}$ equal to 2. Furthermore, "quantitative single" variables, as well as "modal" variables, are coded in fuzzy way on one row of the respective matrices $\mathbf{Z}_{ij}$, such that they do not contribute to increase the dimensions of the matrices $\mathbf{Z}_{ij}$. Thus, the number of rows of each $\mathbf{Z}_{ij}$ is $N_i = 2^g \times \prod_{j=1}^{q} k_{ij}$ and the number of rows of the matrices $\mathbf{Z}_{ij}$ (for $j=1,...,p$) is simply $N = \sum_{i=1}^{n} N_i$.

Finally the global coding matrix, denoted $\mathbf{Z}_{N\times K}$, containing all the coordinates of the hypercube vertices representing the n SO's, can be also seen as constituted by the juxtaposition of the p binary or fuzzy coding matrices:

$$Z = [Z_1|...|Z_j|...|Z_p]$$

where: the number N of rows of the upgraded coding matrices $\mathbf{Z}_j$ (for $j=1,...,p$). The total number K of categories of all transformed variables in the SO descriptions is equal to: $K=3\times K = 3\times g + \sum_{j=1}^{q} k_j$ (where k_j is equal to the all the possible values in D_j of the "categorical" or "modal" descriptor y_j).

Furthermore, logical relations among the descriptors, as well as taxonomies will be suitably admitted in the technique.

Usually, logical conditions reduce the space of symbolic objects description, so that, we propose to decompose the SO's in sub-objects, whose descriptions are consistent with the condition expressed by the logical rules. Therefore, the number of SO's to be consider in the analysis increase of the number of sub-objects. For simplicity, we leave equal to n the total number of SO's, including both the original one and the adding sub-objects. The sub-objects are even coded according to the same system of coding.

In the coding phase, in order to take into account hierarchical structure among the categories,

we propose to re-code the values that each SO takes for the taxonomical descriptors. The new coding is carried out with respect to the terminal leaves, using a suitable system of fuzzy coding proportional to the number of the terminal leaves belonging to the same branch. Equivalently it is possible to transform the description of symbolic objects relatively to nominal categories, in an alike description given by the lower level categories (terminal ealves) of the tree structure. In order to display the intermediate categories it is possible to consider the barycentre of the represented leaves.

Generalised Canonical Analysis on the matrix Z under cohesion constraint

The method is performed on the global coding matrix $\mathbf{Z}_{N\times K}$ that can also be considered as a juxtaposition of the p fuzzy and/or disjunctive coding matrices: $\mathbf{Z} = [\mathbf{Z}_1|...|\mathbf{Z}_j|...|\mathbf{Z}_p]$.

Consistently with our proposal to keep the cohesion among the vertices of the same SO, we introduce the cohesion constraint. Let $\mathbf{A}_{N\times n}$ be an indicator matrix that identifies the belonging of the N vertices to the different n SO.

Moreover, in order to preserve the unity of the decomposed SO, consistently with the conditions expressed by logical rules, they will be assigned with the same code to their original SO in the indicator matrix **A**.

The maximized criterion expresses the explicative power of the coded descriptors with respect to the set of hypercube vertices representing geometrically each SO. The factorial axes are obtained as solution of the following characteristics equation:

$$\frac{1}{N}\mathbf{A'Z\Sigma}^{-1}\mathbf{Z'A}\tilde{\xi}_m = \tilde{\mu}_m\tilde{\xi}_m$$

under the ortho-normality constraints : $\tilde{\xi}'_m\,\tilde{\xi}_m = 1$ and $\tilde{\xi}'_m\,\tilde{\xi}_{m'} = 0$ (for $m\neq m'$).

The maximum number of non-trivial eigenvalues is (K-p+1) (provided that the total number of all categories $K = \sum_{j=1}^{p} k_j$ is less than N).

If the matrix **A** is supposed centred, the trace of the matrix decomposed in the analysis, but for a constant term, corresponds to the sum of the Goodman-Kruskal predictivity τ indices (1954), computed with respect to each categorized descriptor (binary or fuzzy coded):

$$tr\left(\mathbf{A'Z\Sigma}^{-1}\mathbf{Z'A}\right) = \sum_{j=1}^{p}\tau_{Z_j}$$

Denoting $\tilde{\beta}_m = \sqrt{\tilde{\mu}_m}\mathbf{\Sigma Z'A}$ they are the eigenvectors of dual space.

The coordinates of the vertices of the i-th SO on the m-th factorial axis are:

$$\tilde{\phi}_{i,m} = \mathbf{Z}_i\tilde{\beta}_m \quad m=1,...,K\text{-}p+1$$

5 Example

In the following example Vertices SGCA is performed on the file "car.xml", which describes 11 characteristics of 33 cars. As SGCA works on all kind of variables, in the variable selection window, all kind of variables selection is allowed. Once ASSO/SODAS 2 starts after

selecting base file (car.xml), we have to select “Factorial Methods” and to drag and drop the icon “SGCA” into the chain, parameterize it and run the method. If the analysis has been performed two icons appears to the right of “SGCA”, the first one represent the text output, the latter the graphical one.

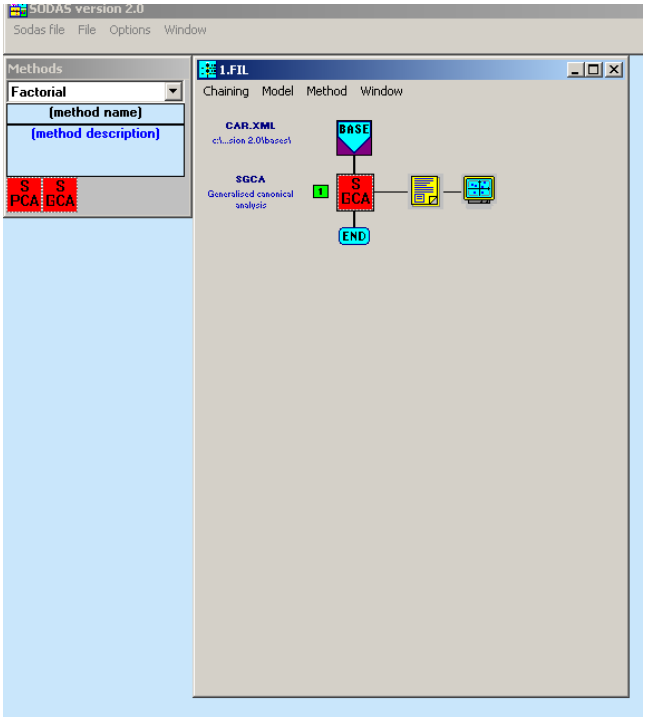


Figure 3: SGCA Workench window

5.1 Textual Output

The text output file is composed by the following section:

a) selection SO’s and variables;

b) eigenvectors and eigenvalues of the analysis with the explained inertia;

c) SO’s interval coordinates on factorial axes;

d) descriptors categories coordinates on factorial axes;

e) descriptors categories absolute contributions to factorial axes;

f) factorial axes relative contributions to descriptors categories representation;

g) SO’s absolute contributions to factorial axes;

h) factorial axes relative contributions to SO’s representation;

i) axes description section, based on the contribution of each categories to the axes. For each axis the categories are ordered by their absolute contribution to it and by the sign of the categories coordinates for it; not all the categories are displayed but only those one that have a contribution greater than the mean contribution of all categories to the axis. The sign (- or +) represent the orientation of axes, then when a category contributes with a + (resp. -) sign means that it is useful to explain the positive (resp. negative) versus of the axis.

```
Proc: Symbolic Object Generalized Canonical Analysis

----------------------------------

 Selected Variables

  1 ====>  (Interval) Price
  2 ====>  (Interval) Cubic_Capacity
  3 ====>  (MultiNom) Fuel
  4 ====>  (MultiNom) Traction
  5 ====>  (Interval) Max_Speed
  6 ====>  (Interval) Acceleration
  7 ====>  (Interval) Axes_distance
  8 ====>  (Interval) Length
  9 ====>  (Interval) Width
 10 ====>  (Interval) Heigth
----------------------------------

 Selected OBJECTS

  1 ====> Alfa  145
  2 ====> Alfa  156
  3 ====> Alfa  166
  4 ====> Aston Martin
  5 ====> Audi A3
  6 ====> Audi A6
  7 ====> Audi A8
 .......
 32 ====> Skoda Octavia
 33 ====> Passat
----------------------------------


 EIGEN VECTORS numb of components==>33

Eigen Vect   1 -0.118  -0.066   0.011  -0.020....
Eigen Vect   2 -0.125   0.081   0.015  -0.033....
Eigen Vect   3 -0.172  -0.198  -0.036   0.313....
Eigen Vect   4 -0.037   0.108   0.038   0.180....
Eigen Vect   5 -0.050   0.110  -0.019   0.403....
Eigen Vect   6 -0.038  -0.133  -0.044  -0.039....
Eigen Vect   7 -0.157  -0.213   0.044  -0.060....
Eigen Vect   8  0.233   0.385   0.256   0.105....
Eigen Vect   9 -0.105   0.221   0.236  -0.336....
Eigen Vect  10 -0.108   0.036  -0.113  -0.145....
Eigen Vect  11  0.045  -0.070  -0.027  -0.283....
Eigen Vect  12  0.681  -0.279  -0.089  -0.194....
Eigen Vect  13  0.086  -0.014   0.177  -0.101....
Eigen Vect  14  0.395   0.050  -0.158   0.206....

Eigenvalues            Inertia  Perc. of Inertia  Cum. Inertia
Eigenvalue n.   1     0.02917      48.483           48.483
Eigenvalue n.   2     0.01364      22.669           71.152
Eigenvalue n.   3     0.00705      11.725           82.877
Eigenvalue n.   4     0.00315       5.240           88.117
Eigenvalue n.   5     0.00237       3.944           92.061
Eigenvalue n.   6     0.00177       2.937           94.998
```

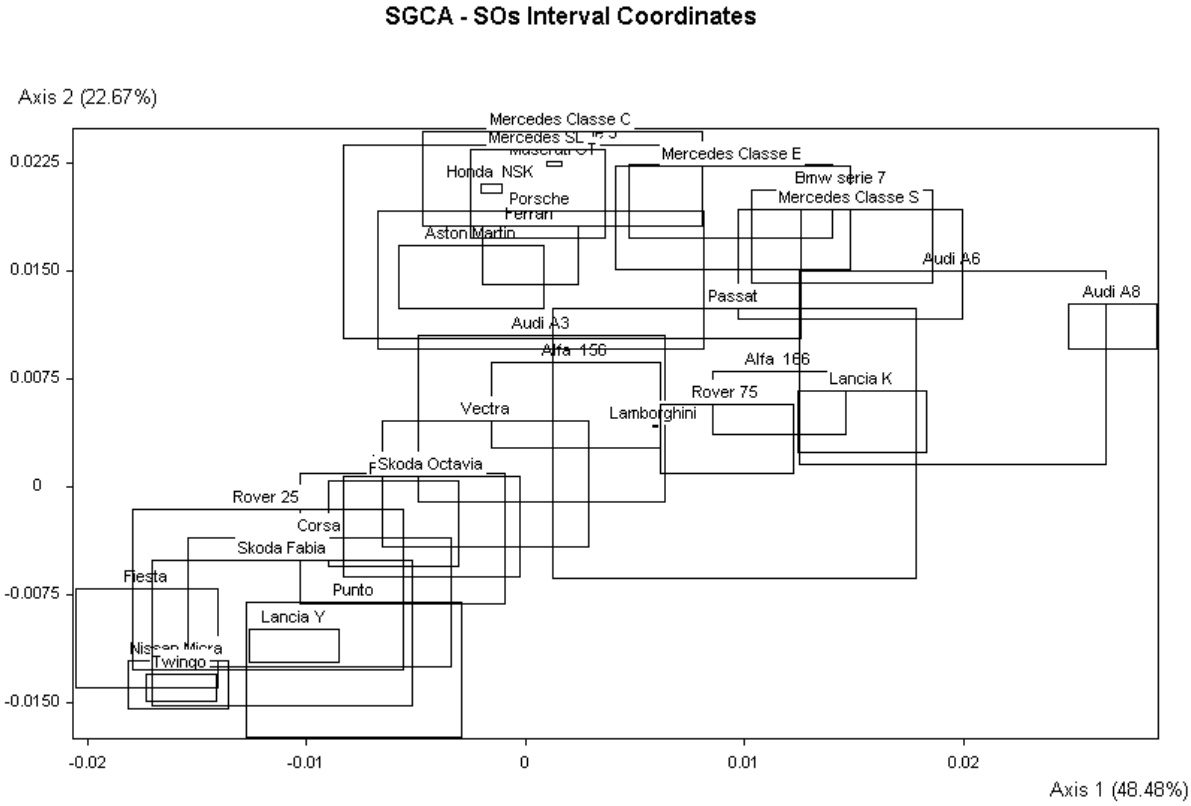
```
Eigenvalue n.   7     0.00099       1.648           96.646
Eigenvalue n.   8     0.00087       1.446           98.093
Eigenvalue n.   9     0.00033       0.554           98.646
Eigenvalue n.  10     0.00029       0.485           99.132
Eigenvalue n.  11     0.00017       0.290           99.422
Eigenvalue n.  12     0.00016       0.264           99.686
Eigenvalue n.  13     0.00011       0.187           99.874
Eigenvalue n.  14     0.00008       0.126          100.000

    Coordinates at interval OBJS

                        Factor 1                Factor 2                  Factor 3

Alfa  145   [-0.01027349;-0.00092814]  [-0.00805685; 0.00098466]  [-0.01246340;-0.00907266]
Alfa  156   [-0.00153474; 0.00614971]  [ 0.00271581; 0.00867354]  [-0.01097328;-0.00884009]
Alfa  166   [ 0.00855727; 0.01460638]  [ 0.00369880; 0.00804041]  [-0.00840862;-0.00680025]
Aston Mart  [-0.00576328; 0.00083732]  [ 0.01245332; 0.01679105]  [ 0.01040933; 0.01237243]
Audi A3     [-0.00489718; 0.00638890]  [-0.00106351; 0.01056175]  [-0.01087535;-0.00676749]
Audi A6     [ 0.01254182; 0.02651724]  [ 0.00157475; 0.01505741]  [-0.00938872;-0.00295365]
Audi A8     [ 0.02477819; 0.02879532]  [ 0.00964339; 0.01271807]  [-0.00491108;-0.00083194]
.......
Skoda Octa  [-0.00830258;-0.00023711]  [-0.00621292; 0.00073019]  [-0.01164840;-0.00880830]
Passat      [ 0.00126959; 0.01786581]  [-0.00628567; 0.01242306]  [-0.01324718;-0.00843350]


----------------------------------------
Categories Coordinates on Factorial Axes
----------------------------------------


 Categories coordinates


Price     -LOW    -0.00600  -0.00137  -0.00097   0.00019
Price     -MED     0.00429   0.00117   0.00025  -0.00018
Price     -HI     -0.00446  -0.00293   0.00385   0.00061
Cubic_Capa-LOW    -0.00431  -0.00328  -0.00070   0.00013
Cubic_Capa-MED     0.00140   0.00153  -0.00013  -0.00013
Cubic_Capa-HI      0.00112  -0.00209   0.00244   0.00050
Fuel      -Pet     0.00036   0.00029   0.00008   0.00010
Fuel      -Die    -0.00356  -0.00287  -0.00080  -0.00103
Traction  -Fro     0.00074  -0.00200  -0.00060   0.00005
Traction  -Rea    -0.00326   0.00242   0.00161  -0.00067
Traction  -Int     0.00316  -0.00000  -0.00118   0.00081
........
Heigth    -LOW     0.00409  -0.00797   0.00400   0.00021
Heigth    -MED    -0.01051   0.00389   0.00032   0.00112
Heigth    -HI      0.01189  -0.00392  -0.00067  -0.00132


---------------------------------
Categories absolute contributions
---------------------------------
Price     -LOW      0.0318    0.0048    0.0092    0.0029
Price     -MED      0.0162    0.0035    0.0006    0.0026
Price     -HI       0.0176    0.0221    0.1448    0.0295
Cubic_Capa-LOW      0.0164    0.0276    0.0048    0.0014
Cubic_Capa-MED      0.0017    0.0060    0.0002    0.0013
Cubic_Capa-HI       0.0011    0.0112    0.0581    0.0204
```

```
Fuel      -Pet      0.0001    0.0002    0.0001    0.0009
Fuel      -Die      0.0112    0.0212    0.0063    0.0852
Traction  -Fro      0.0005    0.0102    0.0035    0.0002
Traction  -Rea      0.0094    0.0150    0.0253    0.0365
Traction  -Int      0.0088    0.0000    0.0137    0.0532
.......
Heigth    -LOW      0.0148    0.1633    0.1568    0.0035
Heigth    -MED      0.0975    0.0389    0.0010    0.1016
Heigth    -HI       0.1248    0.0395    0.0044    0.1402

---------------------------------
Categories relative contributions
---------------------------------
Price     -LOW      0.7194    0.0801    0.0778    0.0065
Price     -MED      0.7135    0.1128    0.0098    0.0116
Price     -HI       0.1512    0.1398    0.4647    0.0257
Cubic_Capa-LOW      0.3857    0.4764    0.0418    0.0033
Cubic_Capa-MED      0.2442    0.6186    0.0082    0.0184
Cubic_Capa-HI       0.0288    0.2159    0.5662    0.0540
Fuel      -Pet      0.2244    0.3125    0.0468    0.1734
Fuel      -Die      0.2244    0.3125    0.0468    0.1734
Traction  -Fro      0.0441    0.6850    0.1185    0.0021
Traction  -Rea      0.2765    0.3239    0.2777    0.1087
Traction  -Int      0.3790    0.0000    0.2198    0.2317
........
Heigth    -LOW      0.0474    0.3845    0.1874    0.0011
Heigth    -MED      0.6729    0.1973    0.0025    0.0710
Heigth    -HI       0.7063    0.1640    0.0092    0.0803


---------------------------------
OBJECTS absolute contributions
---------------------------------
Alfa  145       0.0292    0.0124    0.0116    0.0009    0.0004
Alfa  156       0.0092    0.0052    0.0154    0.0073    0.0020
Alfa  166       0.0010    0.0008    0.0021    0.0036    0.0002
Aston Mart      0.0033    0.0035    0.1529    0.0804    0.1068
Audi A3         0.0148    0.0001    0.0080    0.0301    0.0038
Audi A6         0.0266    0.0000    0.0003    0.0001    0.0000
Audi A8         0.0600    0.0022    0.0131    0.0186    0.0056
.........
Skoda Octa      0.0263    0.0070    0.0112    0.0032    0.0005
Passat          0.0017    0.0025    0.0161    0.0255    0.0077

---------------------------------
OBJECTS relative contributions
---------------------------------
Alfa  145       0.7233    0.1792    0.0905    0.0008    0.0009
Alfa  156       0.5091    0.1675    0.2681    0.0159    0.0092
Alfa  166       0.3384    0.1523    0.2267    0.0500    0.0071
Aston Mart      0.0496    0.0306    0.7297    0.0482    0.1366
Audi A3         0.7742    0.0033    0.1327    0.0625    0.0170
Audi A6         0.9953    0.0005    0.0038    0.0002    0.0000
Audi A8         0.8886    0.0186    0.0611    0.0109    0.0070
.......
Skoda Octa      0.7672    0.1181    0.1032    0.0038    0.0012
Passat          0.1678    0.1397    0.4926    0.0981    0.0629
```

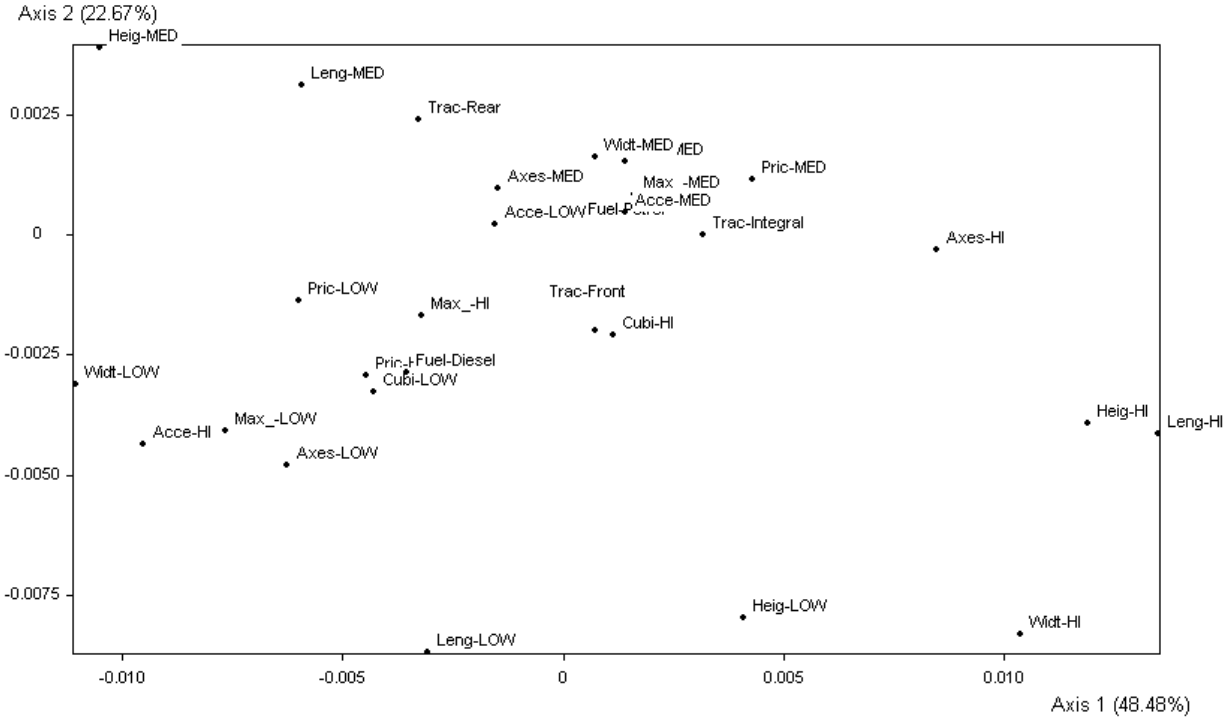
```
Axes description contr, direction (+ positive, - negative)


     Axis 1

Variable (Category)  contr   direction
---------------------------------------
Width     -LOW       0.1079     -
Heigth    -MED       0.0975     -
Accelerati-HI        0.0798     -
Max_Speed -LOW       0.0515     -
Axes_dista-LOW       0.0347     -
---------------------------------------
          Central Zone
---------------------------------------
Axes_dista-HI        0.0629     +
Width     -HI        0.0947     +
Heigth    -HI        0.1248     +
Length    -HI        0.1598     +
---------------------------------------


     Axis 2

Variable (Category)  contr   direction
---------------------------------------
Length    -LOW       0.1949     -
Width     -HI        0.1774     -
Heigth    -LOW       0.1633     -
Axes_dista-LOW       0.0595     -
Accelerati-HI        0.0488     -
Length    -HI        0.0441     -
Max_Speed -LOW       0.0427     -
Heigth    -HI        0.0395     -
---------------------------------------
          Central Zone
---------------------------------------
Heigth    -MED       0.0389     +
---------------------------------------


     Axis 3

Variable (Category)  contr   direction
---------------------------------------
---------------------------------------
          Central Zone
---------------------------------------
Cubic_Capa-HI        0.0581     +
Axes_dista-HI        0.1008     +
Max_Speed -HI        0.1412     +
Price     -HI        0.1448     +
Heigth    -LOW       0.1568     +
Width     -HI        0.2229     +
---------------------------------------
```

5.2 Graphical output

1) the coordinates of the extremes of the intervals of the rectangles which reproduce the hypercube images on each factorial axis



2) coordinates of descriptors categories



References

Bock, H. H. and Diday, E. (Eds): 2000, *Analysis of Symbolic Data*, Springer.

Lauro, C., Palumbo, F.: 2000, Factorial Methods with cohesion constraints on Symbolic Objects. In Proceeding of IFCS2000, Springer-Verlag, .

Lebart, L., Morineau, A. and Piron, M.: 1995, Statistique exploratoire multidimensionelle, Dunod, Paris.

Verde, R.: 1997, Symbolic object decomposition by factorial techniques. Indo-French Meeting, LISE-CEREMADE, Université Paris IX Dauphine.

Verde, R. and De Angelis, P.: 1997, Symbolic objects recognition on a factorial plan, NGUS'97, Bilbao Spain.

Verde, R. : 1999 Generalised Canonical Analysis on Symbolic Objects. In *Classification and Data Analysis, Theory and Application*. Vichi M. - Opitz O. (Eds), Springer-Verlag, Heidelberg, 195-202.

Verde, R. and Lauro, C.: 1993, Non symmetrical data analysis of multiway fuzzy coded matrices, ISI, Florence.

Discrimination

INFORMATION SOCIETY TECHNOLOGIES
PROGRAMME

SDT User Manual

Strata Decision Tree

M.C. Bravo Llatas
UCM (Madrid)

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: **15/01/2004**

User manual for Strata Decision Tree. SODAS Software

M.C. Bravo Llatas

January 2004

Universidad Complutense de Madrid, Centro de Proceso de Datos, Av. Paraninfo s/n, 28040 Madrid, Spain; *carmen@sim.ucm.es*

1 Introduction

Tree-growing methods or segmentation trees (Breiman et al. (1984), Quinlan (1986)) are in general non-parametric classification techniques. A population of individuals or objects is divided into predefined classes, which have to be explained (described) by a set of *qualitative* variables (the *independent* variables or *predictors)* . The aim is to explain or predict the class that an individual belongs to, known the values independent variables have for this individual and looking for logical dependencies between the independent variables and the *criterion* or *class* variable. These dependencies are described by rules of prediction for the criterion variable by means of the independent variables. The growing methods are recursive and obtain at each step a binary partition of the underlying population, described by one independent variable, which maximises the *quality gain of prediction* or explanation of the criterion variable by the independent variables from one step to the next. This quality of prediction or explanation can be seen as an information content measure of the tree with respect to the predefined classes. Several information criteria are proposed in the literature (Breiman et al. (1984), Quinlan (1986), Ciampi et al. (1996)).

Frequently, the underlying population is not only subdivide into the classses to be investigated but is simultaneously composed of groups S_1,..., S_{m} of individuals, which will be called *strata*, such as individuals of a country are divided into regions, individuals of a region are divided into towns, individuals of an educational system are divided into schools, etc... Then, it is interesting to explain how strata influence the rules obtained by a tree-segmentation method. Some rules could be applied to some strata, and others can not.

The **Strata Decision Tree (SDT)** program implements a generalised recursive tree-building algorithm (Breiman et al. (1989)) for populations partitioned into strata. Common predictor (including modal probabilistic variables and hierarchical dependencies) and criterion variables describe population in all strata. A modal probabilistic variable associates to input data units a probability distribution over a set of categories. Hierarchical dependence between two variables identifies when a variable is non applicable for specific values of the other one.

Objectives are to explain or predict class variable by the predictors, affected by stratum membership; obtain sets of strata where this explanation / prediction is the same; describe a stratum by these class variable explanations or prediction rules.

Method considers strata structure in all steps of the algorithm. Symbolic objects describe decisional nodes and strata. A stratum is described by a set of symbolic objects that represent rules for prediction of the class variable, with the importance they have in the stratum. General formalisation can be extended to other symbolic data.

2 Input

Let be Ω a set of individuals, $E = \{S_1, ..., S_m\} \subset \mathcal{P}(\Omega)$ a partition of Ω. Thus, each element of E, $S_i \subset \Omega$ is a group of individuals, called a stratum (for $\omega \in \Omega$, $M(\omega) = i \iff \omega \in S_i$). Let individuals $\omega \in \Omega$ be described by the predictors Y_j, $j = 1, ..., p$ and the class variable Z. Two different input variable types[1] are considered:

1. *Categorical single-valued* data. Variables Y_j are categorical single-valued mappings from Ω to $\mathcal{Y}_j = \{1, ..., l_j\}$

2. *Modal probabilistic* data. Variables Y_j are symbolic variables, more especifically modal probabilistic variables with finite domain $\mathcal{Y}_j$, that is, for $\omega \in \Omega$, $Y_j(\omega) = q_j^\omega$ is a probability distribution over $\mathcal{Y}_j$, identified with $(1\, q_j^\omega(1), ..., l_j\, q_j^\omega(l_j))$. The symbolic data description $Y_j(\omega)$ can represent either the uncertainty for an individual or the variation for a group of individuals regarding categories $\mathcal{Y}_j$.

[1] SDT software is able to treat both types of variables simultaneously.

In both cases, Z is a categorical single-valued mapping from Ω to $\mathcal{Z} = \{1, \ldots, s\}$.[2]

For an example of an input data unit see (5) in section 5 .

3 Output

A **decision tree** can be represented by an *organised* set of *assertions,* Ciampi et al. (1996), Bravo & Garcia-Santesmases, 1997. Each decisional node, described by the assertion $t_k = \beta_k \wedge \alpha_k \wedge \mu_k$, represents a set of strata for which the same rule for prediction of the class variable can be applied. Tree is represented by:

$$T = \{\beta_k \wedge \alpha_k \wedge \mu_k\}_{k=1,\ldots,K} \tag{1}$$

where:

- K is the number of decisional nodes[3];
- β_k is a conjunction of events (of $B = \{b = [Y_j \in D_j], b^c = [Y_j \in \mathcal{Y}_j - D_j] | D_j \subset \mathcal{Y}_j\}$; D_j is a subset of the space of categories $\mathcal{Y}_j$; for modal probabilistic data, $\sim$ replaces $\in$) defined in the predictors Y_j;
- α_k is a modal probabilistic symbolic event describing the prediction for Z;
- $\mu_k = [M \in S^k]$ with $S^k \subseteq \{1, \ldots, m\}$ is a Boolean event in the variable M. The μ_k is true for all individuals $\omega \in \Omega$ that belong to a stratum in S^k. Stratum indicators in S^k are identified in *steps 3, 4* of the algorithm (see section 4).
- Function $\wedge$ is the product.

Assertion $\beta_k \wedge \mu_k$ describes the population (its extension) for which the prediction of Z is described by α_k. Value $\beta_k \wedge \mu_k(\omega)$ gives for categorical single-valued data, ω membership to node k, and for modal probabilistic

[2] Current version of SDT (2.23) software admits binary predictors and class variable. For non-binary predictors, some data transformation may be done in input data file. See SDT User Manual for more details.

[3] Here, all terminal nodes are called decisional nodes. In SDT software output three different names are given to these nodes: decisional, terminal-divide and terminal depending on the condition they have been obtained. See section 4 for more details.

data, the probability of node k given ω, that is, for stratum indicators in S^k, the probability of descriptions D_j (in β_k), given ω.

For example, a **decisional node** described by:

$$[sex = f] \wedge [salh25 = yes] \wedge [clerk \sim (no(0.10), yes(0.90))]$$
$$\wedge [NACE \in \{services, electric\}]$$

gives for individuals in services and electricity $NACE$ sectors the rule *if sex is woman and mean gross hourly earnings is in the first quartile then estimated probability to be clerk is* 0.9.

Each **stratum** is also described by an *organised* set of *weighted assertions*, the decisional tree node descriptions where the stratum belongs to, with weights that give the relative importance they have in this stratum, Bravo and García-Santesmases (2000a, 2000b), Bravo, 2001. A stratum is described by different 'segments' of objects described by the values of the predictors and the value for the prediction of the class Z in these segments, as well as by certain percentages, which give the weight these segments have in the stratum. Stratum S_i $(i = 1, ..., m)$, can be described by:

$$S_i : \{w_k^i(\beta_k \wedge \alpha_k) \mid k = 1, \ldots, K\} \tag{2}$$

where:

- $w_k^i \in [0, 1]$ is the relative *weight* of the decisional node k to the stratum S_i.
- Let be $\mu_k = [M \in S^k]$,

$$w_k^i = \begin{cases} \frac{Card(Ext_{S_i}(\beta_k))}{Card(S_i)} & \text{For categorical single-valued data, } i \in S^k \\ \frac{\sum_{\omega \in S_i} \beta_k(\omega)}{Card(S_i)} & \text{For modal probabilistic data, } i \in S^k \\ 0 & i \notin S^k \end{cases}$$

- β_k, α_k assertions of decisional node k in Y_j, Z.
- $\sum_{k \in \{1,\ldots,K\}} w_k^i = 1$ for all $S_i \in E$

An example of a stratum can be seen in (6) in section 5.

Looking at the weights in (2), it can be detected strata that share rules, that is, those with non zero weights in the same node simultaneously. The importance of these rules in each stratum is given by these relative contributions. See section 5 for a complete example.

4 Algorithm

The aim is to build recursively an *organised* set of assertions $T = \{t_k\}_{k=1...K}$ (see (1)), by binary partitioning the population and combining at each step maximisation of an extended information content (EIC) measure of the tree with respect to E and selection of new decisional nodes. The EIC criterion measures quality of prediction for the class variable in a new partition, taking into account stratum membership in the cut. For modal probabilistic predictors, partitions are uncertainty partitions. After maximisation, quality of prediction is tested for subsets of strata in order to build *decisional nodes*. A decisional node is a leaf for some strata, while the other strata follow the recursive method. For the latest strata, a stopping criterion is also checked. In each step of the algorithm, T is composed of *exploratory* (obtained from the recursive partition, they can be binary split further on) and *decisional nodes* (split from an exploratory node, they are terminal). Quality of prediction for the class variable by the predictors and strata is given by the information content measure (IC)[4] of the tree with respect to Ω, Bravo and García-Santesmases, 2000a, Bravo, 2001.

[4]The IC and EIC measures are defined as:

$$IC\{T,\Omega\} = -\sum_{k=1}^{K} P(\beta_k \wedge \mu_k) Ent(Z|\beta_k \wedge \mu_k) \tag{3}$$

$$EIC\{T,r,b,E\} = IC\{T(r),\Omega\} \tag{4}$$

$$-P(\beta_r \wedge b \wedge \mu_r) \sum_{i \in S^r} P([M=i]|\beta_r \wedge b \wedge \mu_r) Ent(Z|\beta_r \wedge b \wedge [M=i])$$

$$-P(\beta_r \wedge b^c \wedge \mu_r) \sum_{i \in S^r} P([M=i]|\beta_r \wedge b^c \wedge \mu_r) Ent(Z|\beta_r \wedge b^c \wedge [M=i])$$

where $T(r) = T - \{\beta_r \wedge \alpha_r \wedge \mu_r\}$ is the tree that results from T when the node r is removed; $P(.)$ is the estimated probability of a node; $P(.|a)$ is the estimated conditional probability of a stratum to the node described by a and $Ent(Z|.)$ is the entropy (Quinlan (1990)) for Z in the corresponding node.

For monoevaluated data and a Boolean assertion a, $P(a) := \frac{Card(Ext_\Omega(a))}{Card(\Omega)}$ and $P([M = i]|Ext_\Omega(a)) := \frac{Card(Ext_{S_i}(a))}{Card(Ext_\Omega(a))}$ are estimated in a frequentist way.

For probabilistic data and an assertion $\beta \wedge \mu$ ($\mu = [M \in S]$), probabilities are estimated by $P(\beta \wedge \mu) := \frac{\sum_{\omega \in \Omega} \beta \wedge \mu(\omega)}{Card(\Omega)}$, and $P([M = i]|\beta \wedge \mu) := \frac{\sum_{\omega \in S_i} \beta(\omega)}{\sum_{\omega \in \Omega} \beta \wedge \mu(\omega)}$, for $i \in S$ and zero in other case.

Descriptions for Z in α_k are given by probability distributions over $\{1, ..., s\}$. For monoevaluated data, probabilities are estimated as relative frequencies of each class in a node. For probabilistic data, probability of class $l \in \{1, ..., s\}$ in node k is estimated by $\frac{\sum_{\omega \in \Omega} \beta_k \wedge [Z=l] \wedge \mu_k(\omega)}{\sum_{\omega \in \Omega} \beta_k \wedge \mu_k(\omega)}$.

Information content measure is the opposite of a weighted entropy for Z variable in decisional nodes. Extended information content measure is based in internal entropy in strata in successor nodes. Decisional node criterion is based in a threshold for these internal entropies.

Let X be the set of exploratory nodes in a step. Very briefly, algorithm main steps than can be seen in figure 1 are:

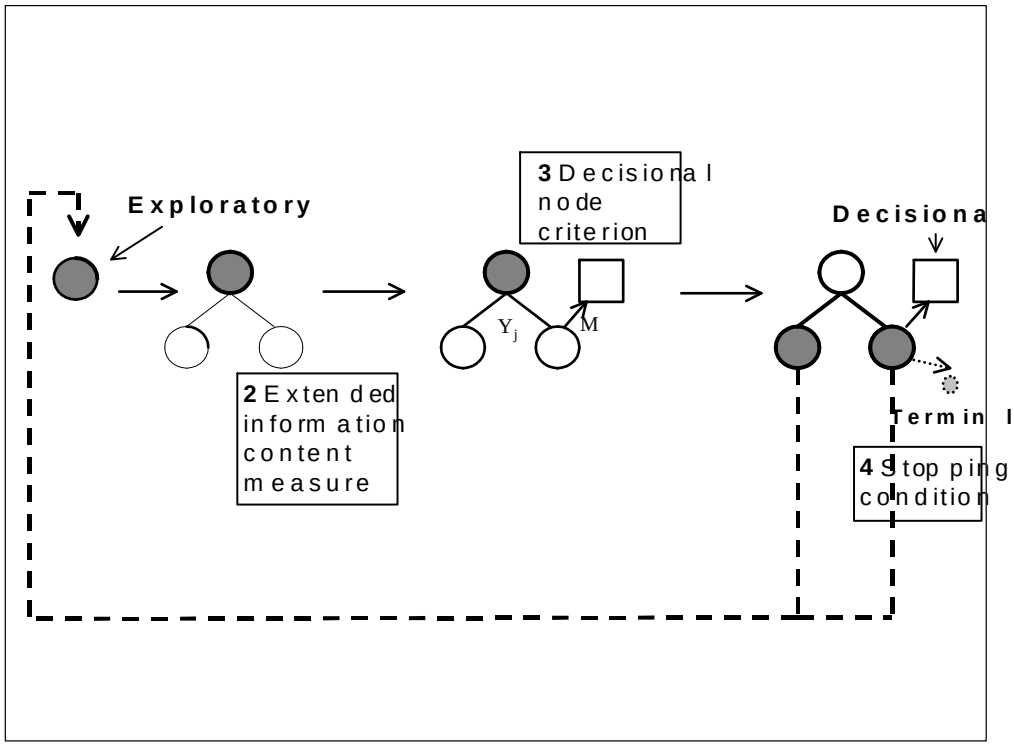


Figure 1: Algortihm main steps

Step 0: Initialisation and evaluation of IC at initial step, $IC\{T, \Omega\}$, that is the minus entropy of Z in the whole population. The first exploratory node contains the whole population (and all strata).

Step 1: Check admissibility condition. For each $r \in X$ (if any), build $B_r \subseteq B$ set of admissible splitting statements to be explored from node r, also considering information given by NA (non-applicable) rules and *maximum*

depth level[5] permitted.

Step 2: Obtain the best split. For each $r \in X$, maximise in $b \in B_r$, the EIC measure, $EIC\{T, r, b, E\}$ of T expanded from node r by split b with respect to E. Maximise in $r \in X$ these measures and select the best node r' and split. This node r' is removed from X (given that we explore it now). Make the split and add to parent node descriptions (in Y_j, M) the description of the new split.

Step 3: Decisional node criterion.* For the new children nodes, i.e., the new exploratory nodes, check the set of strata for which the decisional node condition is satisfied (ex. minimum probability for a stratum and a class of Z to be split in a decisional node). Split from these nodes these strata to form new (a) **decisional node**(s).

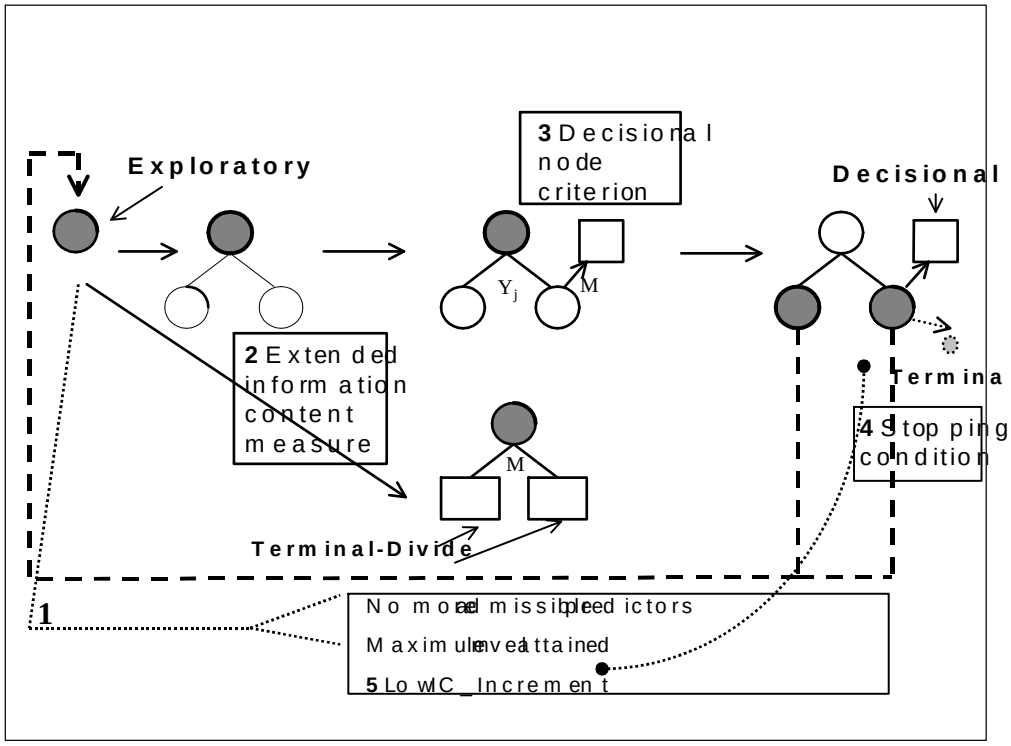


Figure 2: Complete algorithm steps

[5]Terms with * are input parameters in SDT software

Step 4: Strata terminal node condition.* For the new exploratory nodes (if not empty), check the set of strata for which the stopping propagation condition from node is satisfied (ex. low weight). Remove these strata from exploratory nodes, obtaining a **terminal node**.

Step 5: Check minimum improvement of IC.* If this relative improvement is small, then: (1) algorithm *Steps* 2,3,4 are undone and; (2) the explored *parent* node r' is split into two terminal nodes, splitting the subset of strata by their quality of prediction of Z. This latter action is also taken when the maximum depth level is attained or an exploratory node has no admissible splits (in *step 1*). These new nodes will be called in SODAS, **terminal-divide nodes** to distinguish them from nodes obtained by decisional node condition or stopping criterion for strata.

Update node descriptions. New decisional/terminal node descriptions add to parent node descriptions in Y_j, descriptions in M with the strata they contain. For exploratory nodes, update their descriptions in M, removing the strata split in *Steps* 3,4. Compute $IC\{T, \Omega\}$, go to *Step 1*.

In figure 2, complete algorithm steps are shown.

5 Example

Method has been applied to modal probabilistic data obtained from "T25IT Italy: Monthly earnings by local unit size, NACE (economic activity) and ISCO (profession)" data about Statistics on the structure and distribution of earnings (SES), 1995. Consolidated data refers to 5.000.000 employees. More details on original data in Bravo & García-Santesmases (2000), Bravo (2001).

Set Ω is composed of 720 data units described by modal probabilistic data, considering input data weights of original data. Groups described come from the combination of the categories of 22 economic sectors, 7 professions, and 7 company sizes. $E \subset \mathcal{P}(\Omega)$ contains subsets of data units that belong to the same $NACE$ sector ($m = 5$). $NACE$ sectors considered are: mining and quarrying; manufacturing; electricity, gas and water supply; construction; and, services. Class variable is the truth variable *manual* ($s = 2$), Predictors are *sex* and truth variables for thresholds in original variable quartiles: $h50$ for mean weekly hours; $sal75$ for mean gross monthly earnings; $b25$ for mean monthly value of periodic bonuses, $salh50$ for mean gross hourly earnings and *cvm* median for monthly earnings coefficient of variation. An example

of one data unit $\omega \in \Omega$ is:

$$\omega : (sex(\omega) = (f(0.64), m(0.36)), sal75(\omega) = yes, b25(\omega) = (yes(0.04), no(0.96)) \tag{5}$$
$$salh50(\omega) = (yes(0.64), no(0.36)), nace(\omega) = services, manual(\omega) = no)$$

that represents a set of individuals of *services* sector, $non - manual$, with mean gross salary less than first quartil and probability distributions for *sex* $(f(0.64), m(0.36))$, for $b25$, $(yes(0.04), no(0.96))$ and for $salh50$, $(yes(0.64), no(0.36))$.

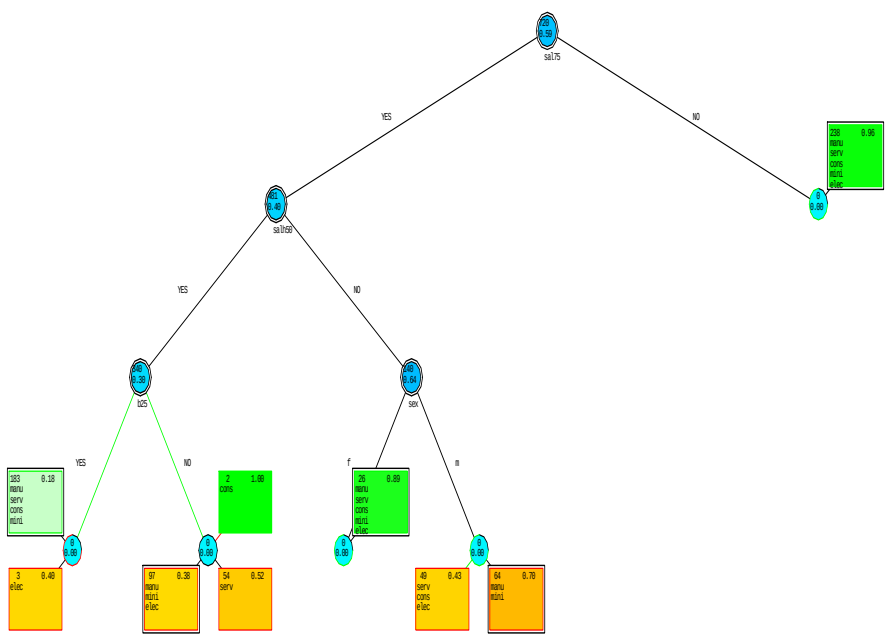


Figure 3: Decision Tree on SES data, 3 levels

Figure 3[6] shows decisional tree built in 3 levels. Initial information content measure is -0.677260 and final value is -0.432755. Rounded nodes are exploratory nodes and squared nodes are decisional nodes. Light coloured nodes, represent prediction rules for *manual* employees and dark coloured nodes for $non - manual$ employees. Nodes shows weights and estimated

[6]This graph is an output of SDT method in SODAS.

probability for $non - manual$ employees. Decisional nodes show strata as well. Decisional nodes in this tree are:

$11d1 : [sal75 \sim no] \wedge [manual \sim (no0.96, yes0.04)] \wedge [nace \in \{manufact, services, construc, mining, electric\}]$

$30d0 : [sal75 \sim yes] \wedge [salh50 \sim yes] \wedge [b25 \sim yes] \wedge [manual \sim (no0.18, yes0.82)] \wedge [nace \in \{manufact, services, construc, mining\}]$

$31d1 : [sal75 \sim yes] \wedge [salh50 \sim yes] \wedge [b25 \sim no] \wedge [manual = no] \wedge [nace = construc]$

$32d1 : [sal75 \sim yes] \wedge [salh50 \sim no] \wedge [sex \sim f] \wedge [manual \sim (no0.89, yes0.11)] \wedge [nace \in \{manufact, services, construc, mining, electric\}]$

Table 1 gives weights for these decisional nodes in strata. Sum of the elements of each column is not 1 because in the table are only the decisional nodes obtained by decisional node condition. All NACE sectors share $11d1$ rule, with a relative importance into them around 30 or 35 per cent. Rule 30d0 is shared by $mining$, $manufacturing$, $construction$ and $services$ sectors, having a relative importance into the first three of 30%, while in $services$ sector it decreeses to 11%. NACE sectors with the exception of $construction$ share rule $32d1$ with a relative importance into them of 4%. $Mining$ and $manufacturing$ sectors share 3 rules $11d1$, $30d0$ and $32d1$ with

	mining	$manfacturing$	$electricity$	c$onstruction$	$services$
$11d1$	0.34	0.32	0.34	0.3	0.37
$30d0$	0.37	0.32		0.28	0.11
$31d1$				0.08	
$32d1$	0.04	0.04	0.04	$\times$	0.04

Symbol $\times$ specifies a value lower than 0.01

Table 1: Table of decisional node weights in strata. Probabilistic SES data.

similar weights in both strata, (weights = 0.34, 0.37 and 0.04 for $mining$ and weights = 0.32, 0.32 and 0.04 for $manufacturing$). Thus, 70% of $manual$ explanation in both sectors is for the same prediction rules. $services$ sector also shares these rules, first and third with similar relative importance and second decreeses to 11%.

As an example, *mining* sector is described by:

$$mining : \{0.34[sal75 \sim no] \wedge [manual \sim (no(0.96), yes(0.04)], \quad (6)$$
$$0.37[sal75 \sim yes] \wedge [salh50 \sim yes] \wedge [b25 \sim y.] \wedge [manual \sim (no(0.18), y.(0.82))],$$
$$0.04[sal75 \sim yes] \wedge [salh50 \sim no] \wedge [sex \sim f] \wedge [manual \sim (no(0.89), y.(0.11))],$$
$$0.25Other\}$$

that is, by three rules with respective relative importance in the sector of 0.34, 0.37 and 0.04. Nodes with *mining* sector have a double sorrounding line.

6 Conclusion

Advantages of method presented are the analysis of symbolic data, the *generalisation of a stratum by symbolic objets* that represent prediction rules for class variable by the predictors giving a conjoint interpretation of strata in the *context* of all strata and not isolatedly, and *classification of strata by common prediction rules*. Also that inclusion of strata information in all steps of the algorithm gives in only one tree, common prediction rules for strata and favours good predictors for some strata.

7 Bibliography

References

[1] BOCK,H.H., DIDAY,E., Eds.(2000), .*Analysis of Symbolic Data. Exploratory Methods for Extracting Statistical Information from Complex Data*, Springer, Heidelberg.

[2] BRAVO,M.C. (2000), Strata Decision Tree Symbolic Data Analysis Software. En: H.A.L. Kiers, J.P. Rasson, P.J.F Groenen, M. Shader (eds.) *Data Analysis, Classification and Related Methods*, Springer, Heidelberg, 409-415.

[3] ———-(2001): *Análisis de Segmentación en el Análisis de Datos Simbólicos,* Tesis Doctoral, Fac. Ciencias Matemáticas, Universidad Complutense de Madrid.

[4] BRAVO,M.C., GARCÍA-SANTESMASES,J.M. (1997), Segmentation Trees for Stratified Data. In: Jansen,J., Lauro,C.N. Eds: *Applied Stochastic Models and Data Analysis: The Ins/Outs of Solving Real Problems*. Curto, Nápoles, 37-42.

[5] ———(2000a): Segmentation Trees for Stratified Data. In: Bock,H.H. and Diday,E. Eds.: *Analysis of Symbolic Data. Exploratory Methods for Extracting Statistical Information from Complex Data*. Springer, Heid., 266-293.

[6] ———(2000b), Symbolic Object Description of Strata by Segmentation Trees. In: *Computational Statistics*. Physica Verlag, Heidelberg, **15**, 13-24.

[7] BREIMAN,L., FRIEDMAN,J.H., OLSHEN,R.A., STONE,C.J. (1984) *Classification and Regression Trees*. Wadsworth, Belmond, Ca.

[8] CIAMPI,A., DIDAY,E., LEBBE,J., PÉRINEL,E. and VIGNES,R. (1996): Recursive partition with probabilistically imprecise data. In: Diday,E. et al. Eds.: *Ordinal and symbolic data analysis*. Springer Verlag. 201–212.

[9] QUINLAN,J.R. (1990), Probabilistic Decision Trees, In: Kodratoff,Y., Michalski,R. Ed. *Machine Learning, an Artificial Intelligence A., III*. Kaufmann, 140-152.

INFORMATION SOCIETY TECHNOLOGIES
PROGRAMME

SBTREE User Manual

Bayesian Decision Tree

Edited by FUNDPMa

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

ASSO Project
Workpackage WP7.2 - Module SBTREE

Bayesian Decision Tree

Discriminant Analysis on interval data and the Bayesian tree visualization by using the modules SBTREE and VTREE

A Short Tutorial for Users

Jean-Paul Rasson - Pascale Lallemand - Séverine Adans

Version 1: December 15, 2003

SBTREE is a tree-growing algorithm, which is merging approaches of density estimation and decision tree.
This tree-growing algorithm is proposed in order to treat explicitly interval symbolic variables. The aims are:

1. to describe in the form of a binary tree, the various classes of the given partition;
2. to build a discriminant decision rule which is able to classify new objects (with unknown class membership) to one class of the given partition.

1 The Underlying Population and the Variables

We consider a population $\Omega = \{1, \ldots, n\}$, where each object is described by two categories of variables:

- p predictors $Y_1, \ldots, Y_p$: interval-valued variables;
- the criterion variable C: a categorical single variable.
  This variable contains the class of each training object and 99 for each object to be classified.

A small example is given by table 1

k \ j	Var1	Var2	Class
1	[0.5:2.5]	[4:5]	1
2	[3:4]	[2.5:4]	1
3	[4.5:5]	[3.5:6]	1
4	[8:9.5]	[1:3]	2
5	[7.5:10]	[3:3.5]	2
6	[10.5:11]	[3.5:6]	2
7	[3:4.5]	[-6:-1]	99

TAB. 1 – *A symbolic data table*

2 What is Bayesian Tree?

SBTREE, based on density estimation, aims to classify new objects to one class of a prior partition. Each split is carried out by selecting the most discriminant variable. The classification step is performed according to the Bayesian decision rule thanks to kernel density estimation. This description requires a class-specific density estimate and an optimal choice of the corresponding window bandwidths. Suitable prior probabilities are also computed.

The Bayesian decision tree is constructed by repeated splits of subsets of the training set into two descendant subsets.
The entire construction of a tree, then, resolves around three elements:

- the selection of the splits;
- the decisions when to declare a node terminal or to continue splitting;
- the assigment of each terminal node to a class.

At the end, the proposed method gives rules in order to classify the news objects.

3 Tree-Growing Method

In SBTREE, we are limited to the case of two classes: C_1 and C_2.
The original contribution of this method lies in the way of splitting a node. Indeed, the cut will be based on the Bayesian rule.

3.1 Bayesian Rule

Suppose that for the population Ω, the data vector $\tilde{x}$ is distributed with a density function $f_k(\tilde{x})$ $(k = 1,2)$. Suppose also that the prior probabilities p_k $(k = 1,2)$ are known.
By definition, the Bayesian rule assigns $\tilde{x}$ to the population with the largest $p_k f_k(\tilde{x})$ $(k = 1,2)$.

So, for each variable $(i = 1, \ldots, p)$, we consider the set of questions

$$p_1 f_{1,i}(\tilde{x}) \overset{?}{>} p_2 f_{2,i}(\tilde{x}), \quad \forall \tilde{x}, \quad \tilde{x} \in \Omega.$$

If $p_1 f_{1,i}(\tilde{x})$ is greater than $p_2 f_{2,i}(\tilde{x})$, the object $\tilde{x}$ is considered belong to the first class, else to the second one.

Note that if the densities are unknown, we have to estimate them by means of the kernel method.

3.2 Kernel Method

To estimate the intensity of a Non Homogeneous Poisson Process, we will use a non-parametric method: the **Kernel method**.
The Kernel estimator is defined by :

$$\hat{f}_l(x) = \frac{1}{n_l} \sum_{i=1}^{n_l} \frac{1}{h} K(\frac{x - X_i}{h}), \quad l = 1{,}2$$

where - K is the kernel with these properties:
symetric, continu, $\int K(x)dx = 1$, $K \geq 0$;
- h is the window width also called the **smoothing parameter**.

The kernel estimator is a sum of 'bumps' placed around the observations. The kernel function K determines the shapes of the bumps while the window width h determines their width. How compute h?

3.3 Bumps and Multi-modalities

Silverman has showed interest for the search of modes within the framework of the clustering. He distinguishes the concept of bumps and modes: a mode in a density f will be a local maximum, while a bump will be characterized by an interval in such way that the density is concave on this interval but not on an larger interval.

In the framework of the density estimation by the kernel method,
- for very great values of h, the density estimation is unimodal;
- for h decreasing, the number of mode increases.

This behaviour is described by Silverman: "*the number of modes is a decreasing function of the smoothing parameter* h". This is showed only for the NORMAL kernel.
Consequently, to realize the estimation of the intensity of the Non Homogeneous Poisson Process, we will use the kernel method with the normal kernel. This kernel is defined by:

$$K_{\mathcal{N}}(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}.$$

Since we use the normal kernel, there is a critical value h_{crit} of the smoothing parameter for which the estimation changes from the unimodality to multi-modality. Our criterion of split seeks this value.

3.4 Prior Probabilities

The user has choice between two prior probabilities, which are represented by a parameter. So he must select:
- 1 for equal prior probabilities: $\hat{p}_k = \frac{1}{2}$, for $k = 1{,}2$;
- 2 for proportions observed in the training data sets: $\hat{p}_k = \frac{n_k}{n}$, for $k = 1{,}2$.

3.5 Cut Variable

Each split is carried out by selecting the most discriminant variable, that is, the one which leads to the purest nodes. Indeed, thanks to the bayesian rule, computed for each variable, we are able to determine which are the misclassified objects. So the most discriminant variable will be the one which minimizes the impurity measure, i.e. the number of these misclassified objects.

3.6 Splitting Criteria

For each variable, by a dichotomic process, we find the greatest value of parameter h giving a number of modes of the associated intensities strictly larger than 1. Once this h determined, we cut into two fields for which the impurity measure is minimum; i.e for which the number of objects misclassified is smallest.
Proceeding variable by variable, we will be able to select the one which generates the purest nodes. By a dichotomic process, we compute the cut value for this best variable, such that we have $p_1 f_{1,i}(\tilde{x}) = p_2 f_{2,i}(\tilde{x})$.

This procedure is recursively performed until some stopping rules are fulfilled:
- the number of points in a node is less then a predefined value;
- there is only one class in a node.

3.7 Pruning Method

At the end of the splitting process, we end with a huge tree. The best subtree is selected. Indeed, we have developed, under the hypothesis of non homogeneous Poisson Process, a tree-pruning method that takes the form of a classical test of hypothesis: the **Gap Test** (Kubushishi).

The Gap Test, in the case of two classes D_1 and D_2, tests the hypothesis
H_0: there is $n = n_1 + n_2$ points in $D_1 \cup D_2$
against the hypothesis
H_1: there are n_1 points in D_1 and n_2 points in D_2, with $D_1 \cap D_2 = \emptyset$.

In other words, the null assumption involves what we call a bad cut whereas the alternative assumption gives a good cut.

The pruning method built here runs through the tree branch by branch from its root to its end, in order to index the good cuts (Gap Test satisfied) and the bad cuts (Gap Test non-satisfied). The ends of the branches for which there are only bad cuts are pruned.

3.8 Assignment of each terminal node to a class

Regardless of the number of points in each class, the left node must be assigned to the class 1 and the right node to the class 2. Indeed, the posterior probability to belong to class 1 is greater than the one relative to class 2 for individuals of the left node and conversely for those of the right node.

3.9 Application to INTERVAL data

How apply this new clustering method to symbolic data of interval type?

Note M the center of an interval and L the length of an interval.
The usual distance used for interval variables is the Hausdorff distance:

$$d_H([a_1,b_1],[a_2,b_2]) = max(|a_1 - a_2|,|b_1 - b_2|)$$

or (Chavent and Lechevallier)

$$d_H([a_1,b_1],[a_2,b_2]) = |M_1 - M_2| + |L_1 - L_2|.$$

We can work on the space $(M,L) \subseteq \mathcal{R} \times \mathcal{R}^+$, where each interval is represented by its center and its length.
Consequently, as we use a divisive method, separations must respect the order of the classes' centers.

Thus, if there is a density ρ_1 on the axis M and another ρ_2 on the axis L, we must minimize, in the most general case of a Non Homogeneous Poisson process the integrated intensity:

$$\int_{M_i}^{M_{i+1}} \rho_1(m)dm + \int_{min(L_i,L_{i+1})}^{max(L_i,L_{i+1})} \rho_2(l)dl \tag{1}$$

and to choose as bipartition this one generated by any point being located inside the interval which maximizes 1.

4 Symbolic Discriminant Analysis

At each terminal node, a symbolic object is allocated. This symbolic object, described by the path in the tree that provides it, defining a decision rule able to classify new objects in one of the two classes.

5 SBTREE Output

The SBTREE algorithm produces as output a listing file containing:

- the tree growing strategy;
- the symbolic discrimination rule, and the result of the symbolic discriminant analysis.

An internal output file (.vt0) contain the structure of the tree (used like input of VSTAR).

6 Example

The dataset is composed of eight objects, described by four interval variables and a categorical variable. This variable is the class owner of the object, or is settled at 99 for the objects to be classified.

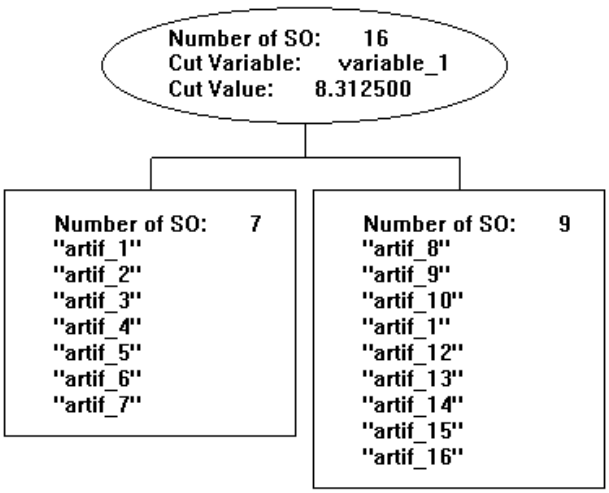
Sample	Specific Gravity	Freezing point	Iodine Value	Saponification Value	Class
linseed oil	[0.930;0.935]	[-27;-18]	[170;204]	[118;196]	1
perilla oil	[0.930;0.937]	[-5;-4]	[192;208]	[188;197]	1
camelia oil	[0.916;0.917]	[-21;-15]	[80;82]	[189;193]	2
olive oil	[0.914;0.919]	[0;6]	[79;90]	[187;196]	2
beef tallow	[0.860;0.870]	[30;38]	[40;48]	[190;199]	2
hog fat	[0.858;0.864]	[22;32]	[53;77]	[190;202]	2
cottonseed oil	[0.916;0.918]	[-6;-1]	[99;113]	[189;198]	99
sesam oil	[0.920;0.926]	[-6;-4]	[104;116]	[187;193]	99

The created results file is the following:

```
-------------------------------------------------------
 BASE= C:\Mes documents\ASSO\SBTREE\Bases\ichino.sds
 Number of OS = 8
 Number of variables = 5
 METHOD=SBTREE Version 1.0    13/2003
-------------------------------------------------------
Sodas The Statistical Package for Symbolic Data Analysis
Version 1.0 - 09/2002
MODULE: SBTREE
Bayesian Decision Tree
-----------------------------------------------------------
-----------------------------------------------------------
Sodas File          : C:\Mes documents\ASSO\SBTREE\Bases\ichino.sds
Log File            : C:\Mes documents\ASSO\SBTREE\ichino.LOG
Listing File        : C:\Mes documents\ASSO\SBTREE\filieres\ichino.LST
---------------------------------------------------------
---------------------------------------------------------
 Learning Set         :     8
 Number of variables  :     4
 Min. number of object by node :     3
 Alpha Value    : 0.500000

 GROUP OF SELECTED VARIABLES :
 ============================

 ( Pos )    Name          Type
 (   1 ) specific           INTERVAL
 (   2 ) freezing           INTERVAL
 (   3 ) iodine             INTERVAL
 (   4 ) saponification     INTERVAL

 LIST OF SYMBOLIC OBJECTS IN THE SET :
 =====================================

 "linseed"        "perilla"        "camelia"        "olive"          "beef"
 "hog"            "cottonseed"     "sesam"
```

```
-----------------------------------------------
Creation of the tree based on the training set
-----------------------------------------------

  =========================
  Split of the node :    1
  =========================

  Number of Symbolic objects in the node:      6pt
  ---------------------------------------

  Criteria of cut :
  -----------------

 Cut variable :(  1) specific
 Cut value :   0.00
 Smoothing parameter CLASS 1 :   0.00
 Smoothing parameter CLASS 2 :   1.70
 Rule : if value of the SO <    0.00 -> the SO is in the left node (next even node)
        if value of the SO >    0.00 -> the SO is in the right node (next odd node)

Node :   2    Cardinal :     2pt
====================================
 (0)  "linseed"
 (1)  "perilla"

Node :   3    Cardinal :     4pt
====================================
 (2)  "camelia"
 (3)  "olive"
 (4)  "beef"
 (5)  "hog"

  =========================
  Split of the node :    2
  =========================

  Number of Symbolic objects in the node:      2pt
  ---------------------------------------

     TERMINAL NODE
 the stop-splitting is true : the size of the node is too small
 value of the stop-splitting rule:     3

  =========================
  Split of the node :    3
  =========================

  Number of Symbolic objects in the node:      4pt
  ---------------------------------------

     TERMINAL NODE
 stop splitting: only one class within the node
```

The graphic representation after the pruning procedure of the results is:




```
-----------------------------------------------
Symbolic Discrimination Rule
-----------------------------------------------

        CLASSES:
        --------
        C1:     2 TRAINING OBJECT(S)
        C2:     4 TRAINING OBJECT(S)
        ----------------------------
                6 TRAINING OBJECT(S)

             2 OBJECT(S) TO CLASSIFY

        PRIOR PROBABILITIES:
        --------------------
            C1: 0.500000
            C2: 0.250000

        CUT RULE:
        ---------
        CUT NUMBER   CUT VARIABLE   CUT VALUE
                 1               1    0.000000

        CLASSIFICATION:
        ---------------
        OBJECT   CLASS 1       CLASS 2
             6         0          1
             7         0          1
```

References

[1] M. Bardos. *Analyse discriminante, Application au risque et scoring financier.* Dunod, Paris, 2001.

[2] V. Bertholet. *Apports de l'estimation de densité aux arbres de discrimination : Applications réelles.* PhD thesis, Facultés Universitaires Notre-Dame de la Paix, 2003.

[3] H. H. Bock and E. Diday. *Analysis of Symbolic Data, Exploratory methods for extracting statistical information from complex data.* Springer-Verlag, 2000.

[4] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees.* Wadsworth, Belmont, Calif., 1984.

[5] M. Chavent and Y. Lechevallier. Dynamical clustering of interval data optimization of an adequacy criterion based on haussdorff distance. In K. Jajuga, A. Sokolowski, and H. H. Bock, editors, *Classification, Clustering, and Data Analysis*, pages 53–60, Berlin, Germany, juillet 2002. Springer.

[6] T. Kubushishi. *On some Applications of the Point Process Theory in Cluster Analysis and Pattern Recognition.* PhD thesis, Facultés Universitaires Notre-Dame de la Paix, 1996.

[7] J. P. Rasson and T. Kubushishi. The gap test: an optimal method for determining the number of natural classes in cluster analysis. In E. Diday, Y. Lechevallier, M. Schader, P. Bertrand, and B. Burtschy, editors, *New approaches in Classification and Data Analysis*, pages 186–193, 1994.

[8] B.W. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, 1986.

[9] S.W. Silverman. Using kernel density estimates to investigate multimodality. *Journal of Royal Statistic Society, B*, 43:97–99, 1981.

INFORMATION SOCIETY TECHNOLOGIES PROGRAMME

SFDA User Manual

Factorial Discriminant Analysis

Edited by DMS

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

Date: 0**3/02/2004**

SFDA : Factorial Discriminant Analysis

1 Introduction

The aim of the Factorial Discriminant Analysis on SO's (SFDA) (Lauro et al. 2000) is to define suitable geometrical classification rules, even using a more discriminant SO's visualization by convex hulls, in order to assign new SO's to one of the *a priori* classes. The SFDA aims at defining the best discriminant subspace which is defined first via a descriptors quantification and selection step, then performing a FDA on the new descriptors. The SGCA has been proposed as a quantification procedure for the descriptors coding in the SFDA. In addition, the addictiveness of the optimized criterion in the SGCA provides a suitable way to select the best discriminant predictors. In fact, in the quantification phase (i.e. SGCA) the criterion optimised is a generalization of a sum of predictivity indices (like the τ of Goodman-Kruskal).

The preliminary phase of selection of the most discriminant descriptors can be performed according to the same criterion decomposed in the analysis.

An interpretation of the factorial discriminant axes in terms of symbolic assertions is also proposed as well as a study of the stability of the geometrical discriminant configuration.

2 The input to SFDA

SFDA is performed on a set of SO's, belonging to a priori *r* classes identified by a nominal descriptor in the symbolic data file. They can be described (in a symbolic table) by any kind of SO descriptor ("quantitative single", "interval", "categorical single", "categorical multi-valued" and "modal"). Mixed variables, logical dependence rules, taxonomies and missing values (NULL) are admitted.

In SFDA module is also present a procedure for an automatic selection of the most discriminant descriptors based on a generalization of the Goodman-Kruskal predictivity τ index. The procedure for the descriptors selection ranks the descriptors according the value of the like-τ index, computed for each descriptor. Fixed a bound, the most discriminant descriptors are selected and listed in the text output file.

The classification rule in SFDA is a geometrical one and it is performed according a proximity measure. A new element is assign to the class with respect to it present the minimum dissimilarity to the whole class or to the elements belonging to class (using the single, average or complete linkage). Three dissimilarity functions can be chosen in the classification procedure and their values are evaluated on the coordinates of the SO's and SO classes on the factorial plane.

Moreover, the validation of the procedure is performed on a new set of SO's (or classical individuals), described by the same descriptors and extracted by the same population of the training set. For the SO's of this test set is assumed known the class membership, in order to build the classification table. In the case, the test set is not furnished in input, the same training test is used for default to validate the classification rules and compute the ratio of misclassified.

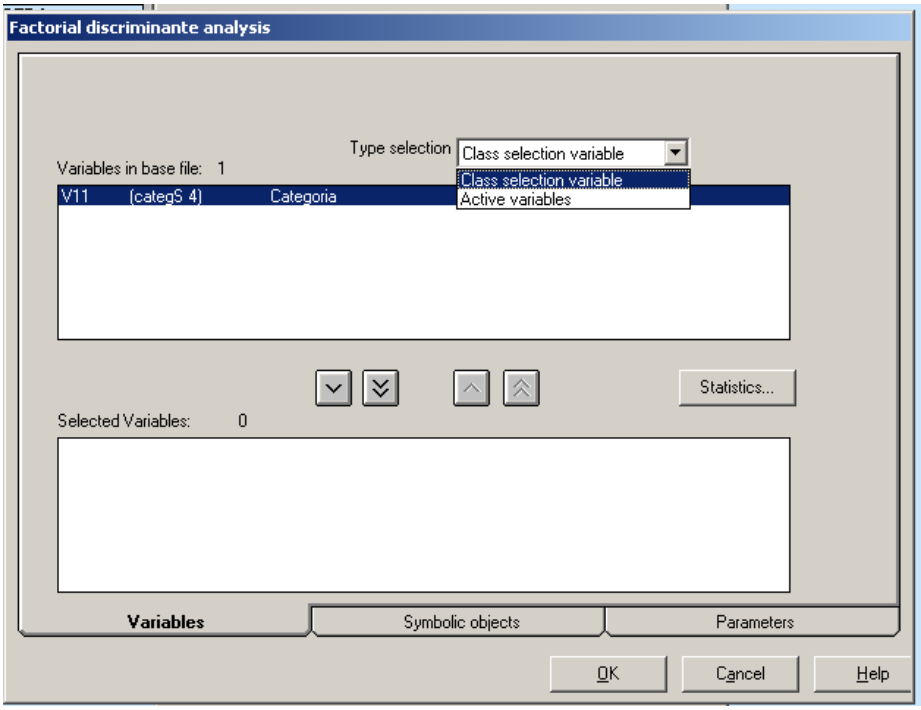


Figure 1: Descriptors selection window for SFDA

The "Class Selection variable" have to be necessarily "Categorical single" and it corresponds to the indicator variable of the a priori classification of the training set SO's.

3 The output of SFDA

The SFDA results are: the coordinates of the vertices of the hypercubes (SO's and classes) and the coordinates of the extremes of the intervals of the rectangles which reproduce the hypercube images on each factorial axis. The latter can be stored in a SOM file. These images of the SO's on factorial planes can be displayed by using VPLOT module.

The other results concern: a list of the SO's descriptors ordered according to their predictivity power; a list of the SO descriptors selected in the analysis; the list of SO's in the training and in the test set; classification tables and the misclassification ratio for the chosen classification rule; the descriptions of the factorial discriminant axes in terms of symbolic assertions; a summary table with the results of the validation procedure.

4 SFDA method

Similarly to the other factorial methods, SFDA is a *symbolic-numerical-symbolic* approach (Diday, 1987). It is constituted by the following steps:

i) quantification of the SO descriptors;

ii) FDA on the quantified predictors;

iii) symbolic interpretation of the results (factorial discriminant axes and classification rules) according to the nature data.

The first phase of SFDA procedure consists in a quantification of the SO's descriptors.

It is performed by applying the SGCA for: the coding, the selection and the quantification of the descriptors of the different nature. The result is given by numerical predictors: factorial

coordinate-vectors ($\tilde{\mathbf{\Phi}} = \left[\tilde{\phi}_1,...,\tilde{\phi}_M\right]$) of the hypercube vertices associated to the training set SO's.

The second phase assumes the $\tilde{\phi}_j$, $\forall j = 1,...,M$ as the new predictors.

Then, denoting with $\mathbf{C}_{N\times r}$ the indicator matrix of the training set SO's with respect to the r a prori class, the factorial discriminant axes are obtained as solutions of the following eigenequation:

$$\left[\left(\tilde{\mathbf{\Phi}}'\mathbf{H}\tilde{\mathbf{\Phi}}\right)^{-1}\left(\tilde{\mathbf{\Phi}}'\mathbf{HC}\right)\left(\mathbf{C'HC}\right)^{-1}\left(\mathbf{C'H}\tilde{\mathbf{\Phi}}\right)\right]\boldsymbol{\upsilon}_m = \kappa_m \boldsymbol{\upsilon}_m,$$

where: the column vectors of $\tilde{\mathbf{\Phi}}$ are centred, while the columns of the matrix $\mathbf{C}$ are not centred; $\mathbf{H}$ is the diagonal matrix of the weights on the hypercubes vertices of the classes. The diagonal terms of $\mathbf{H}$ are equal to: $\frac{N_i}{N}$ (i = 1,...,N), depending on the number of the SO's in each class. Moreover, κ_m and $\boldsymbol{\upsilon}_m$ are the m-th eigenvalue and eigenvector of the matrix in the equation.

The coordinates of the hypercubes vertices associated to each symbolic object are computed as follows: $\mathbf{\Psi}_m = \tilde{\mathbf{\Phi}}\boldsymbol{\upsilon}_m$.

The number of axes to be retained in the FDA analysis is chosen on the basis of the usual criteria, i.e. the percentage of variance of the predictors explained from the first $q \leq min$ (N, M-1) axes.

The last phase is represented by the definition of alternative geometrical classification rules. We consider both SO's and classes are represented on factorial planes by rectangles. The classification of a generic SO in a class C_i is defined according to two events:

i) if a SO image is included in the representation of C_i (of the training set) it is assigned to this class C_i;

ii) if a SO is partially or completely outside to all the representation of the classes or it is in an overlapping area between two or more classes representations, we are going to consider a suitable measure of similarities to assign the element to only one class C_i. The new element will be assigned to the class according to the highest similarity value.

For defining a geometric classification rule, we propose three approaches, two based on the Description Potential π(.), defined by De Carvalho (1992) as the volume obtained by the Cartesian product of the SO descriptor values, the last one is based on the Hausdorff distance between polygons.

In particular, the three geometrical classification rules need the following parameters:

1) *Ichino - De Carvalho's dissimilarity index*: it is based on the following measure between projected objects represented as hypercubes:

$d(\hat{\omega}_j,\hat{\omega}_s) = \sqrt[m]{\sum_\alpha \left(p_\alpha \psi_\alpha\left(\hat{\omega}_j,\hat{\omega}_s\right)\right)^m}$ where $\hat{\omega}_j$ and $\hat{\omega}_s$ are the factorial representation of two SO j and s, p_α is the α-*th* eigenvalue and m is the number of retained factorial axes,

$$\psi_\alpha\left(\hat{\omega}_j,\hat{\omega}_s\right) = \frac{\mu\left(S_\alpha^s \oplus S_\alpha^j\right) - \mu\left(S_\alpha^s \cap S_\alpha^j\right) + \gamma\left(2\mu\left(S_\alpha^s \cap S_\alpha^j\right) - \mu\left(S_\alpha^s\right) - \mu\left(S_\alpha^j\right)\right)}{\mu\left(S_\alpha^s \oplus S_\alpha^j\right)},$$

$\mu\left(S_\alpha^j\right)$ is the length of the coordinates interval of object j on the α-th axis, $\mu\left(S_\alpha^s \oplus S_\alpha^j\right)$ is the

length of the coordinates interval of the conjunction of the two coordinates intervals of the objects j and s on the α-th axis and $\mu\left(S_\alpha^s \cap S_\alpha^j\right)$ is the length of the intersection of the coordinates intervals of the two objects on the α-th axis.

In order to compute the dissimilarity between two objects it needs to choice a suitable metric (1=city block, 2=Euclidean,...), a bound γ (in]0;1[) that allows to emphasize the intersection between objects.

2) *Descriptor Potential Increase* index: this measure is based on the concept of potential descriptor increasing. It is based on the computing of how a class image has to enlarge in order to contain the object to be affected. The dpi is given by the following formula:

$$\mathrm{dpi}\left(\hat{\omega}_j, \hat{\omega}_s\right) = \frac{\prod_{\alpha=1}^{m} \mu\left(S_\alpha^s \oplus S_\alpha^j\right) - \prod_{\alpha=1}^{m} \mu\left(S_\alpha^j\right)}{\prod_{\alpha=1}^{m} \mu\left(S_\alpha^j\right)}$$

Where $\hat{\omega}_j$ is the representation of the j-th class, $\hat{\omega}_s$ is the representation of the object s to be affected, $\mu\left(S_\alpha^j\right)$ is the coordinates interval length of the object class j on the α-th axis and $\mu\left(S_\alpha^s \oplus S_\alpha^j\right)$ is the coordinates interval length of the conjunction of the two intervals which represent the objects on the α-th axis.

3) *Haudorff distance*: it is based on the Hausdorff distance between two objects computed on the coordinates intervals on the factorial axes. This measure needs only the choice of the metric (1=city block, 2=Euclidean,...).

If the distance between a new SO s and a class is evaluated with respect to all the SO's belonging to such class (in 1) and 3)), the classification criterion needs of choosing the single, average or complete linkage, as in classical clustering techniques.

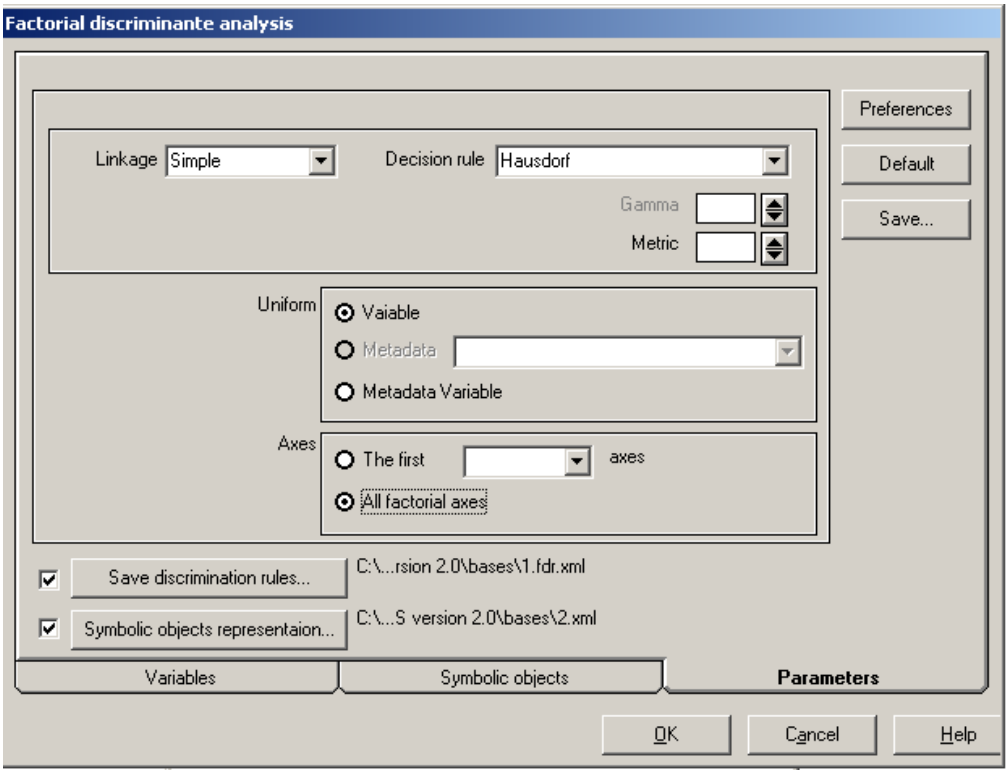


Figure 2: Classification parameters window

Globally, the quality of the SO's visualization can be evaluated according to the percentage of total variability explained by the factorial axes. As regards the quality of the representation of each SO, some measures have to be defined in order to take into account the SO wholeness. Alternatively, we propose to evaluate it with respect to the coordinates of all the hypercube vertices belonging to the same SO.

5 Examples

The following example is performed on the file "OS2.sds", which describes 12 objects (sex-age typology of people in Portugal) described by 38 variables. Once ASSO starts after selecting base file (OS2.sds), we have to select "Discrimination and regression Methods" and to drag and drop the icon "SFDA" into the chain, parameterizing it and run the method. The "categorical single" variable that identifies the belonging of training set SO's to the classes is the 38th (*"Partition into 3 clusters"*). In this example we use the automatic selection procedure in order to select predictors that have the best discrimination power with respect the a priori classification. The procedure is based on a generalization of the Goodman-Kruskal's τ predictivity index. If the analysis has been performed two icons appears to the right of "SFDA", the first one represents the text output, the latter the graphical one.

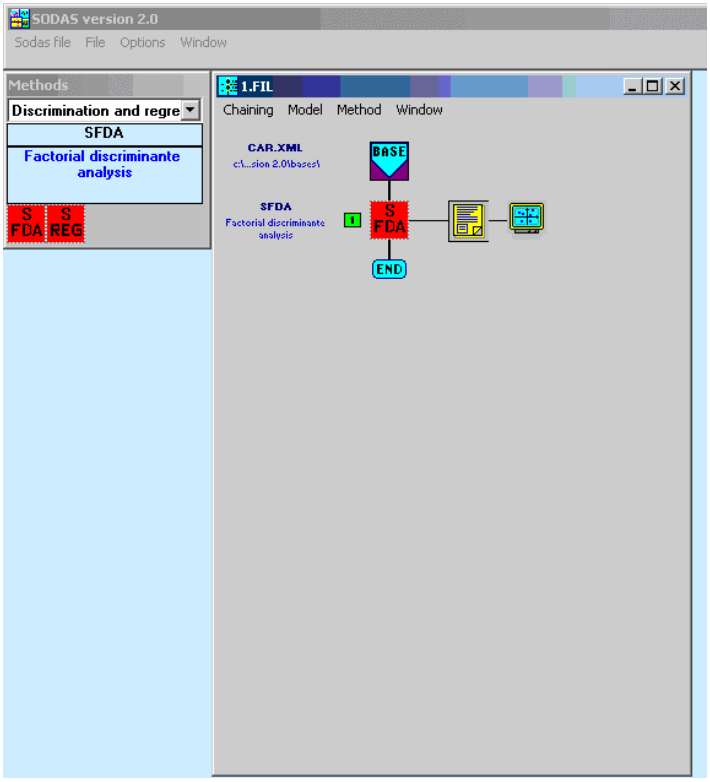


Figure 3: SFDA Workbench window

The textual output of SFDA is constituted by the following sections:

a) the class ID variable selected with the name of the classes (the categories of the "categorical single" variable);

b) the result of the automatic selection procedure: the variable (except for the class ID variable) are ordered by the associated predictivity index;

c) the SO's grouped on the basis of the class ID variable;

d) the eigenvectors and the eigenvalues of the SFDA;

e) the interval coordinates of SO's and of the classes of SO's;

f) the distance matrix between objects on the base of the dissimilarity or distance chosen in the parameter window (in this case, *Hausdorff distance* and an *Euclidean metric*);

g) the assignment table on the base of the linkage criterion chosen in the parameter window (in this case, the *average* linkage);

h) the classification table and the correct classification ratio

```
Proc: Symbolic Object Factorial Discriminant  Analysis

----------------------------------
ClassID variable ===>> Partition into  3 clusters

Class 1  ===>> Cluster 1/3
Class 2  ===>> Cluster 2/3
Class 3  ===>> Cluster 3/3

----------------------------------
Preprocessing of Proc: SFDA   Automatic selection of variables


 Automatic Variables Selection procedure based on Tau Index (Goodman Kruskal)

  CLassID       38 Partition into  3 clusters

Selection|   NO var|      Tau|   CumTau| % of Tau| Cum% Tau|
-------------------------------------------------------------------------------
 Refused |       21|   0.0873|   0.0873|    0.419|    0.419  |Real_state,_renting_
 Refused |       15|     0.12|    0.207|    0.574|    0.993  |Financial_intermedia
 Refused |       18|    0.128|    0.335|    0.617|     1.61  |Mining_and_quarrying
 Refused |       14|    0.155|     0.49|    0.744|     2.35  |Electricity,_gas_and
 Refused |        9|    0.161|    0.652|    0.775|     3.13  |Other_professional_s
 Refused |       31|    0.191|    0.842|    0.915|     4.04  |Service_workers_and_
 Refused |       16|    0.254|      1.1|     1.22|     5.26  |Hotels_and_restauran
 Refused |       25|    0.278|     1.37|     1.33|      6.6  |              Clerks
 Refused |       23|    0.402|     1.78|     1.93|     8.53  |Wholesale_and_retail
 Refused |        8|    0.426|      2.2|     2.04|     10.6  |       Family_worker
 Refused |       29|     0.46|     2.66|     2.21|     12.8  |Plant_and_machine_op
 Refused |        1|    0.473|     3.14|     2.27|     15.1  |           Full-time
 Refused |        2|    0.473|     3.61|     2.27|     17.3  |           Part-time
 Refused |       33|    0.525|     4.13|     2.52|     19.8  |Technicians_and_asso
 Refused |       22|    0.536|     4.67|     2.57|     22.4  |Transport,_storage_a
 Accepted|       24|    0.552|     5.22|     2.65|     25.1  |        Armed_forces
 Accepted|       19|    0.566|     5.79|     2.72|     27.8  |      Other_services
 Accepted|       20|    0.571|     6.36|     2.74|     30.5  |Public_administratio
 Accepted|       26|    0.664|     7.02|     3.19|     33.7  |Craft_and_related_tr
 Accepted|       37|    0.667|     7.69|      3.2|     36.9  |       Widow/Widower
 Accepted|       32|    0.676|     8.36|     3.25|     40.2  |Skilled_agricultural
 Accepted|       17|    0.677|     9.04|     3.25|     43.4  |       Manufacturing
 Accepted|       13|    0.679|     9.72|     3.26|     46.7  |        Construction
 Accepted|       34|    0.708|     10.4|      3.4|     50.1  |Divorced_or_Separate
 Accepted|       27|    0.713|     11.1|     3.42|     53.5  |Elementary_occupatio
 Accepted|        5|    0.722|     11.9|     3.47|       57  |  University_studies
 Accepted|       12|    0.733|     12.6|     3.52|     60.5  |Agriculture,_cattle,
 Accepted|       30|    0.738|     13.3|     3.54|       64  |       Professionals
```

```
 Accepted|        6|     0.74|     14.1|     3.55|     67.6  |     Without_studies
 Accepted|        3|    0.746|     14.8|     3.58|     71.2  |Primary_studies_or_l
 Accepted|       10|    0.787|     15.6|     3.78|     74.9  |Self-employed_with_e
 Accepted|       28|    0.831|     16.4|     3.99|     78.9  |Legislators,_senior_
 Accepted|       35|    0.832|     17.3|     3.99|     82.9  |              Married
 Accepted|        4|    0.855|     18.1|     4.11|       87  |Secondary_studies_-_
 Accepted|       36|     0.87|       19|     4.17|     91.2  |               Single
 Accepted|       11|     0.91|     19.9|     4.37|     95.6  |Self-employed_withou
 Accepted|        7|    0.924|     20.8|     4.44|      100  |             Employee


----------------------------------

 Selected Variables

  1 ====>  (Interval) Primary_studies_or_less
  2 ====>  (Interval) Secondary_studies_-_and_professional
  3 ====>  (Interval) University_studies
  4 ====>  (Interval) Without_studies
  5 ====>  (Interval) Employee
  6 ====>  (Interval) Self-employed_with_employees
  7 ====>  (Interval) Self-employed_without_employees
  8 ====>  (Interval) Agriculture,_cattle,_hunt,_forestry_and_fishing
  9 ====>  (Interval) Construction
 10 ====>  (Interval) Manufacturing
 11 ====>  (Interval) Other_services
 12 ====>  (Interval) Public_administration
 13 ====>  (Interval) Armed_forces
 14 ====>  (Interval) Craft_and_related_trades_workers
 15 ====>  (Interval) Elementary_occupations
 16 ====>  (Interval) Legislators,_senior_officials_and_managers
 17 ====>  (Interval) Professionals
 18 ====>  (Interval) Skilled_agricultural_and_fishery_workers
 19 ====>  (Interval) Divorced_or_Separate
 20 ====>  (Interval) Married
 21 ====>  (Interval) Single
 22 ====>  (Interval) Widow/Widower
----------------------------------

 Selected OBJECTS

.......................................

  Class   1

  1 ====> Man / 15 to 24 years old
  2 ====> Man / 25 to 34 years old
  3 ====> Woman / 15 to 24 years old
  4 ====> Woman / 25 to 34 years old
  5 ====> Woman / 35 to 44 years old
.......................................

  Class   2

  6 ====> Man / 65 and more years old
  7 ====> Woman / 55 to 64 years old
  8 ====> Woman / 65 and more years old
.......................................
```

```
  Class   3

  9 ====> Man / 35 to 44 years old
 10 ====> Man / 45 to 54 years old
 11 ====> Man / 55 to 64 years old
 12 ====> Woman / 45 to 54 years old
.........................................

----------------------------------
Eigenvalues FDA:
 N° Eigenvalue Percentage Cumulated
-----------------------------------
  1    0.0090   80.2272   80.2272
  2    0.0017   15.2593   95.4865
  3    0.0005    4.5135  100.0000
-----------------------------------

 eigenvectors FDA  number of components  9

Eigen vect   1  -0.866231   -0.488909   -0.038402   -0.084149    0.003606
Eigen vect   2   0.496914   -0.825178   -0.135243   -0.203406    0.046663
Eigen vect   3   0.041657   -0.268826    0.229000    0.742441   -0.143449

    Coordinates at interval OBJS
                              Factor 1              Factor 2              Factor 3
Man / 15 to 24 years  [-0.013269; 0.001323]  [-0.063326;-0.047675]  [-0.025983;-0.018067]
Man / 25 to 34 years  [-0.034656;-0.021991]  [-0.033969;-0.017851]  [ 0.002310; 0.009500]
Man / 35 to 44 years  [-0.057968;-0.041899]  [-0.005901; 0.010260]  [-0.001605; 0.005642]
Man / 45 to 54 years  [-0.068574;-0.051279]  [ 0.010562; 0.027185]  [-0.003434; 0.004177]
Man / 55 to 64 years  [-0.050197;-0.019999]  [ 0.027885; 0.044190]  [-0.008614; 0.003836]
Man / 65 and more ye  [ 0.011618; 0.041571]  [ 0.018438; 0.031508]  [-0.000550; 0.013130]
Woman / 15 to 24 yea  [-0.015587; 0.009685]  [-0.057443;-0.038779]  [-0.008824; 0.003562]
Woman / 25 to 34 yea  [-0.006403; 0.006444]  [-0.033803;-0.020983]  [ 0.008272; 0.014165]
Woman / 35 to 44 yea  [-0.018553;-0.003806]  [-0.007774; 0.008922]  [ 0.010252; 0.016082]
Woman / 45 to 54 yea  [-0.014716; 0.002782]  [ 0.016221; 0.032820]  [ 0.014308; 0.019917]
Woman / 55 to 64 yea  [ 0.014537; 0.035575]  [ 0.019167; 0.030096]  [ 0.012015; 0.022044]
Woman / 65 and more   [ 0.040962; 0.067920]  [ 0.005176; 0.022549]  [-0.001399; 0.014555]

Coordinates at interval of Classes
                Factor   1             Factor   2             Factor   3
Class   1 [-0.034656; 0.009685]  [-0.063326; 0.008922]  [-0.025983; 0.016082]
Class   2 [ 0.011618; 0.067920]  [ 0.005176; 0.031508]  [-0.001399; 0.022044]
Class   3 [-0.068574; 0.002782]  [-0.005901; 0.044190]  [-0.008614; 0.019917]


Hausdorff distance matrix
  Metric = 2


            Man / 15 t  Man / 25 t  Man / 35 t  Man / 45 t  Man / 55 t  Man / 65 a ....
Man / 15 t     0.00000
Man / 25 t     0.04697     0.00000
Man / 35 t     0.07692     0.03673     0.00000
Man / 45 t     0.09569     0.05667     0.02006     0.00000
Man / 55 t     0.10140     0.06488     0.04099     0.03598     0.00000
Man / 65 a     0.09414     0.08056     0.08646     0.09361     0.06362     0.00000
.......
------------------------
```

```
Average Linkage
Assignment Matrix  the rows are the objects the columns the classes
in brakets () the apriori class of the misclassified object

           Class 1   Class 2   Class 3
Man / 15 t   1.0        -         -
Man / 25 t   1.0        -         -
Man / 35 t    -         -        1.0
Man / 45 t    -         -        1.0
Man / 55 t    -         -        1.0
Man / 65 a    -        1.0        -
Woman / 15   1.0        -         -
Woman / 25   1.0        -         -
Woman / 35    -         -        1.0( 1)
Woman / 45    -        1.0( 3)    -
Woman / 55    -        1.0        -
Woman / 65    -        1.0        -

Classification matrix
 the rows are the apriori classes
 the column are the assigned classes

From \ To| Class   1 Class   2 Class   3
----------------------------------------
Class   1|     4.000         -     1.000
Class   2|         -     3.000         -
Class   3|         -     1.000     3.000

Correct classification ratio ===>83.333 %
```

The graphical output is the representation of the classes and of the SOs on the factorial plane.

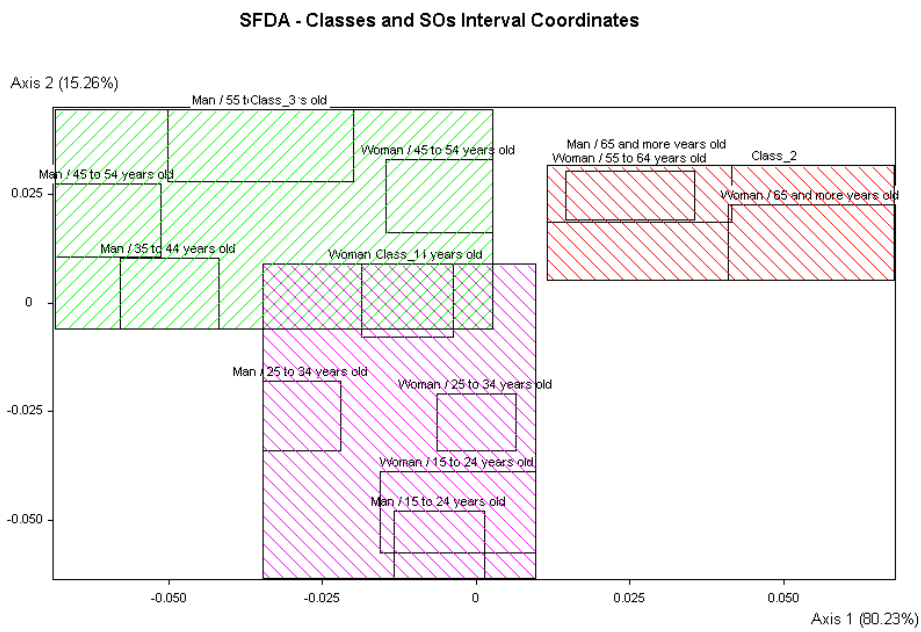


Figure 4: Graphical output of SFDA

References

Bock, H. H. and Diday, E. (eds): 2000, *Analysis of Symbolic Data*, Springer-Verlag, Heidelberg.

Goodman, L.A.,Kruskal, W.H. (1954). Measures of association for cross-classifications. *Journal of the American Statistical Association* 49, 732-764.

Lauro, C., Verde, R., Palumbo, F.: 2000, Factorial Discriminant Analysis on Symbolic Objects, in Bock, H.H. and Diday, E. (eds): 1999, *Analysis of Symbolic Data*, Springer Verlag. Heidelberg.

Lauro, C., Palumbo, F.: 2000, Factorial Methods with cohesion constraints on Symbolic Objects. In Proceeding of IFCS2000, Springer-Verlag, .

Lebart, L., Morineau, A. and Piron, M.: 1995, Statistique exploratorie multidimensionelle, Dunod, Paris.

Verde, R.: 1997, Symbolic object decomposition by factorial techniques. Franco-Indian Meeting, LISE-CEREMADE, Université Paris IX Dauphine.

Verde, R. and De Angelis, P.: 1997, Symbolic objects recognition on a factorial plan, NGUS'97, Bilbao Spain.

Verde, R. : 1999 Generalised Canonical Analysis on Symbolic Objects. In *Classification and Data Analysis, Theory and Application.* Vichi M. - Opitz O. (Eds.), Springer-Verlag, Heidelberg, 195-202.

Verde, R. and Lauro, C.: 1993, Non symmetrical data analysis of multiway fuzzy coded matrices, ISI, Florence.

Regression

INFORMATION SOCIETY TECHNOLOGIES
PROGRAMME

SREG User Manual

Regression

Edited by DAUPHINE

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

ASSO Project

Symbolic Linear Regression
by using the module SREG
A short tutorial for users

AFONSO Filipe, LIMAM Mehdi
DIDAY Edwin.

Ceremade / Université de Paris 9 Dauphine
Place du Maréchal de Lattre de Tassigny
75775 PARIS Cedex 16 FRANCE.

1 Introduction

In the module SREG, we implement methods for linear regression with symbolic data. Methods of regression with symbolic interval and histogram-valued data have already been studied by Billard and Diday. Here, we will first present methods to regress taxonomic variables. Several methods have been suggested and we will compare the different approaches. Second, we will propose a method to regress hierarchical variables. In these two cases, the problem to solve is to define the data matrix to regress.

2 State of the art

2.1 The linear regression

We want to explain a variable Y with a linear equation of explanatory variables X_i (i=1…k): $Y=b_0+b_1X_1+\ldots b_kX_k + \varepsilon = \beta X+\varepsilon$ where ε is the residual error vector such as $E(\varepsilon)=0$ and $Var(\varepsilon)=\sigma^2 I$ ([9]).

In order to find the best vector β, we minimize the medium error $\| \varepsilon \|^2 = \|Y-\beta X\|^2$ by annulling the derivative(/β):

$\|Y-\beta X\|^2 = \| Y \|^2 + < X'X\beta, \beta> -2 < X'Y, \beta>$

$$\frac{\partial}{\partial \beta}\|Y - XB\| = 0 = 2X'X\beta - 2X'Y.$$

We obtain: $\beta^* = (X'X)^{-1}X'Y$.

⇒ In the following sections, we will highlight the matrix X and Y to regress.

Note :If there is only one explanatory variable then we will have the formulas

$$b_1 = \frac{Cov(X,Y)}{Var(Y)} \text{ and } b_0 = \overline{Y} - b_1\overline{X}.$$

2.2 Linear regression with interval-valued data ([3], [4])

In the case of simple linear regression, for each individual i, $Y(i)=[a_i,b_i]$ and $X(i)=[c_i,d_i]$. Under the assumption of distribution uniformity, we show:

- $Cov(X,Y)=(1/4n)\ \Sigma\ (b_i+a_i)(c_i+d_i)-(1/4n^2)\ \Sigma\ (b_i+a_i)\ \Sigma\ (c_i+d_i)$.
- $Y_m=(1/2n)\ \Sigma\ (a_i + b_i)$.
- $S_y^2=(1/4n)\ \Sigma\ (b_i + a_i)^2- (1/4n^2)\ [\Sigma\ (b_i + a_i)]^2$.

For the multiple linear regression, in order to calculate $(X'X)^{-1}X'Y$, we use the symbolic sums and cross products as given in the previous formulas.

2.3 Linear regression with histograms ([4])

In the case of simple linear regression, for each individual i, $Y(i)=y_{i1}[a_{i1},b_{i1}], \ldots, y_{is}[a_{is},b_{is}]$ and $X(i)=x_{i1}[c_{i1},d_{i1}], \ldots, x_{it}\ [c_{it},d_{it}]$ where $\Sigma_k\ y_{ik}= \Sigma_l\ x_{il}=1$.
Under the assumption of distribution uniformity, we show:

- $Cov(X,Y)=(1/4n)\ \Sigma_i\ \{\Sigma_k\ y_{ik}(b_{ik}+a_{ik})\}\ \{\Sigma_l\ x_{il}(c_{il}+d_{il})\}-(1/4n^2)\ [\Sigma_i\ \{\Sigma_k\ y_{ik}(b_{ik}+a_{ik})\}]\ [\Sigma_i\ \{\Sigma_l\ x_{il}(c_{il}+d_{il})\}]$.
- $Y_m=(1/2n)\ \Sigma_i\ \{\Sigma_k\ y_{ik}(b_{ik}+a_{ik})\}$.
- $S_y^2=(1/4n)\ \Sigma_i\ \{\Sigma_k\ y_{ik}(b_{ik}+a_{ik})\}^2-(1/4n^2)\ [\Sigma_i\ \{\Sigma_k\ y_{ik}(b_{ik}+a_{ik})\}]^2$.

For the multiple linear regression, in order to calculate $(X'X)^{-1}X'Y$, we use the symbolic sums and cross products as given in the previous formulas.

3 The symbolic data needed as input

3.1 Linear regression with qualitative variables

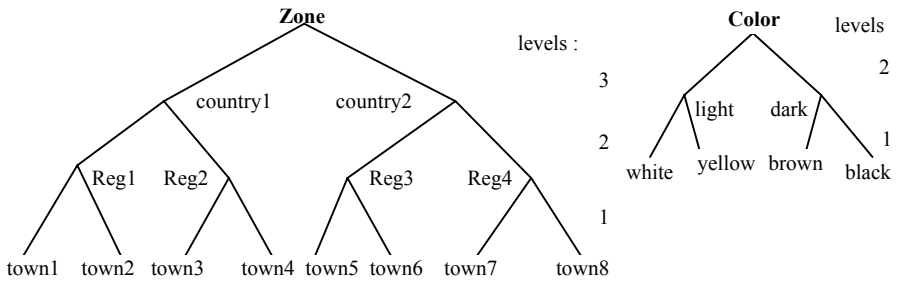
In presence of qualitative variables, each modality of the qualitative variable becomes a binary variable ($X_m(w) = 1$ if w has the modality m and $X_m(w) = 0$ if w does not have the modality m). We remove a modality of each explanatory variable in order to obtain a reversible matrix (X'X). Indeed, if k is the number of variables and if V_{im} refers to the value of the modality m of the variable I (0 or 1) then ∀ i, j from 1 to k

$$\sum_m V_{im} = \sum_m V_{jm} = 1$$ that gives us (X'X) non-reversible.

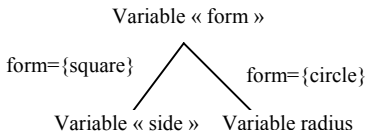
If the case of a dependent qualitative variable we have to code it ([11]). The best code is centred and reduced.

3.2 Taxonomic variables against Hierarchical variables

Taxonomic variables are variables organized in a tree in order to express several levels of generality. Taxonomies are, for example, useful to classify the fauna and the flora, some regions of the globe (town, region, country)…. We will study an example with two taxonomic variables "color" and "zone":



Hierarchical variables (i.e. mother-daughters variables) are variables organized in a hierarchy. Some variables, the daughters, will exist only if another variable, the mother, takes a restricted set of values. The hierarchy can have several levels. A variable X has a daughter Y which has a daughter Z…. For example, the variable "brand of the computers in your firm?" will have a meaning, for the firm f, if the variable "Are there computers in your firm?" takes the modality "yes" for this firm. Also, if the variable "form" takes the modality "circle" then we will inform the variable "radius" and if "form" takes the modality "square" then we will inform the variable "side»:



4 Linear regression with taxonomies

4.1 Input data

We handle values at different levels of generality. In the following example, Color and Zone are the taxonomic explanatory variables and income is the quantitative dependent variable.

We know the tonality but we do not know the color

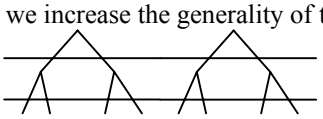
Individual	Color	Zone	income
1	yellow	town1	4400
2	black	town 2	4000
3	white	town 3	3500
4	black	town 3	2000
5	light	Region1	2500
6	yellow	country1	2800
7	black	town 5	3800
8	brown	Region3	3000
9	yellow	town 7	4200
10	light	town 8	3800

We know the region or the country but not the town.

4.2 Method 1: method by decomposition

Principle of the method when the explanatory variables are taxonomic

We suggest doing one regression by level. For each level, we remove the values at higher levels and we increase the generality of the values at lower levels.



One regression by level

With the previous example, we obtain three regressions. At the first level, we remove the individuals 5, 6, 8, 10: 6/10*100=60% of the individuals are used. At the second level, we increase the generality of the data at the first level and we remove the data at the third level. We remove the individual 6: 9/10*100=90% of the individuals are used. At the third level, we increase the generality of all the data to the higher level: 100% of the individuals are used

First level			
I	Color	Zone	income
1	yellow	town1	4400
2	black	town 2	4000
3	white	town 3	3500
4	black	town 3	2000
7	black	town 5	3800
9	yellow	town 7	4200

Second level			
I	Color	Zone	income
1	light	Region1	4400
2	dark	Region1	4000
3	light	Region2	3500
4	dark	Region2	2000
5	light	Region1	2500
7	dark	Region3	3800
8	dark	Region3	3000
9	light	Region4	4200
10	light	Region4	3800

Third level			
I	Color	Zone	Y
1	light	country1	4400
2	dark	country1	4000
3	light	country1	3500
4	dark	country1	2000
5	light	country1	2500
6	light	country1	2800
7	dark	country2	3800
8	dark	country2	3000
9	light	country2	4200
10	light	country2	3800

Principle with a taxonomic dependent variable

First, we decompose the dependent variable as we did with the explanatory variables and second, we code the variable. We obtain a linear equation at each level and therefore predictions at each level. The predictions at the highest levels will be used to improve the predictions at the lowest levels. For example, if “color” is the dependent variable then we will give the codes white=1, yellow=2, brown=3, black=4. If we obtain, for a new individual, a prediction equals to 1.8 then we will not be able to conclude. But if we do another regression at the second level, by coding light=1 and dark=2, and if the equation gives us the prediction 1.2 then we will conclude that the color should be light and moreover we will be able to say that

the color should be yellow. Therefore, predictions of taxonomic variables should be better than predictions of classical qualitative variables.

Conclusions on the method by decomposition

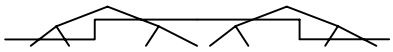
This method does not modify the real data set. We clearly decompose each level by suggesting an own regression. Moreover, the decomposition by level is the best method in the case of a taxonomic dependent variable. However, this method should be less efficient with the increasing of the number of taxonomic variables. Indeed, a lot of values would be removed. Also, if the taxonomic variable has several levels then we will have to deal with several equations.

So, we suggest a method by aggregation and a « divisive » method with only one regression and with all the individuals.

4.3 Method 2: method by aggregation

Principle of the method

If an individual takes a value at a level j then the values sons (only the sons) of it in the tree will be aggregated up to the level j (if light is present in the matrix then white and yellow will be aggregated up to light but if dark is not present then the values black and brown will not be aggregated up). We obtain an unique regression through several levels. With this method all the individuals are used.



Multi-level regression.

With the previous example, we do the regression after having increased the generality of the light colors and of the individuals of the country1 and the region3:

I	Color	Zone	Y
1	light	Country1	4400
2	black	Country1	4000
3	light	Country1	3500
4	black	Country1	2000
5	light	Country1	2500
6	light	Country1	2800
7	black	Region3	3800
8	brown	Region3	3000
9	light	town7	4200
10	light	town8	3800

We will obtain an unique equation through the three levels:

Y = cste+a_1*light+a_2*brown+b_1* Country1+b_2*Region3+b4*town7.

Conclusions on the method by aggregation

This method modifies the initial data matrix but the values remain faithful to the real data and we obtain an unique regression. However, if there are only a few values at the highest levels then we will have to aggregate up all the individuals. Moreover, if the dependent variable is taxonomic then it will be difficult to find multi-level codes. So, we suggest another method.

4.4 Method 3: "divisive" method

Principle of the method

We will obtain an unique equation at the lowest level by replacing all the values at levels>=2 by all theirs sons at the lowest level with the weights w = 1/m (m = number of sons) or w = P_m = proportion of the modality m in the data or every P_m such as $\Sigma P_m=1$. With this method all the individuals are used.

In our example, we replace light by ½yellow, ½white, country1 by ¼town1, ¼ town2, ¼ town3, ¼ town4, region1 by ½ town1, ½town2 and region3 by ½ town5, ½ town6. After the decomposition in binary variables, we will have:

I	White	yellow	brown	black	town1	town2	town3	town4	town5	town6	town7	town8
1	0	1	0	0	1	0	0	0	0	0	0	0
2	0	0	0	1	0	1	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0	0	0	0	0
4	1	0	0	1	0	0	1	0	0	0	0	0
5	½	1/2	0	0	1/2	1/2	0	0	0	0	0	0
6	0	1	0	0	1/4	1/4	1/4	1/4	0	0	0	0
7	0	0	0	1	0	0	0	0	1	0	0	0
8	0	0	1	0	0	0	0	0	1/2	1/2	0	0
9	0	1	0	0	0	0	0	0	0	0	1	0
10	½	1/2	0	0	0	0	0	0	0	0	0	1

We obtain only one regression at the lowest level:

$Y = cst + a_1*white + a_2*yellow + a_3*brown + b_1*\ town1 + b_2*\ town2 + b_3*\ town3 + b_4*\ town4 + b_5*\ town5 + b_6*\ town6 + b_7*\ town7$.

Conclusions on the "divisive" method

We only have one regression at the lowest level and all the individuals are used. Nevertheless, we modify the real data. Indeed, the weights may not be faithful to the real data set.

4 .5 Method 4: multi-level "divisive" method

We propose a method that keeps the advantages of the method by decomposition but with all the individuals.

Principle of the method

We define one equation by level with all the individuals. This method is a combination of the methods 1 and 3. So, for each level, we increase the generality of the values at lower levels and we decrease the generality of the values at higher levels to the relevant level like in the third method.

Step 1: For the variable « Zone », we do a regression at the level of the cities by replacing Region1 by ½town1 ½ town2, Region2 by ½ town3 ½ town4, …, country1 by ¼ town1 ¼ town2 ¼ town3 ¼ town4 and country2 by ¼ town5 ¼ town6 ¼ town7 ¼ town8. We do the same with "Color". We obtain the matrix:

I	white	yellow	brown	black	town1	town2	town3	town4	town5	town6	town7	town8
1	0	1	0	0	1	0	0	0	0	0	0	0
2	0	0	0	1	0	1	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0	0	0	0	0
4	1	0	0	1	0	0	1	0	0	0	0	0
5	1/2	1/2	0	0	1/2	1/2	0	0	0	0	0	0
6	0	1	0	0	1/4	1/4	1/4	1/4	0	0	0	0
7	0	0	0	1	0	0	0	0	1	0	0	0
8	0	0	1	0	0	0	0	0	1/2	1/2	0	0
9	0	1	0	0	0	0	0	0	0	0	1	0
10	1/2	1/2	0	0	0	0	0	0	0	0	0	1

Step 2: For the variable « Zone », we do a regression at the level of the regions by aggregating up the cities to their region in the tree (town3 is replaced by region2…) and by replacing country1 by ½Region1 ½Region2 and country2 by ½Region3 ½Region4. We do the same with "Color". We obtain the matrix:

I	light	dark	Region1	Region2	Region3	Region4
1	1	0	1	0	0	0
2	0	1	1	0	0	0
3	1	0	0	1	0	0
4	0	1	0	1	0	0
5	1	0	1	0	0	0
6	1	0	1/2	1/2	0	0
7	0	1	0	0	1	0
8	0	1	0	0	1	0
9	1	0	0	0	0	1
10	1	0	0	0	0	1

Step 3: For the variable « Zone », we do a regression at the level of the countries by aggregating up the cities and the regions to their country (town3 is replaced by country1 and region3 by country2…). We obtain the matrix:

I	light	dark	country1	country2
1	1	0	1	0
2	0	1	1	0
3	1	0	1	0
4	0	1	1	0
5	1	0	1	0
6	1	0	1	0
7	0	1	0	1
8	0	1	0	1
9	1	0	0	1
10	1	0	0	1

Notes: the equation at the first level is identical to the method 3 and the third level is identical to the third level of the method 1.

Conclusions on the multi-level "divisive" method

This method has the same advantages than the first one but all the individuals are used. However, this method inherits its disadvantages from the method 3.

4.6 Conclusions on the four methods

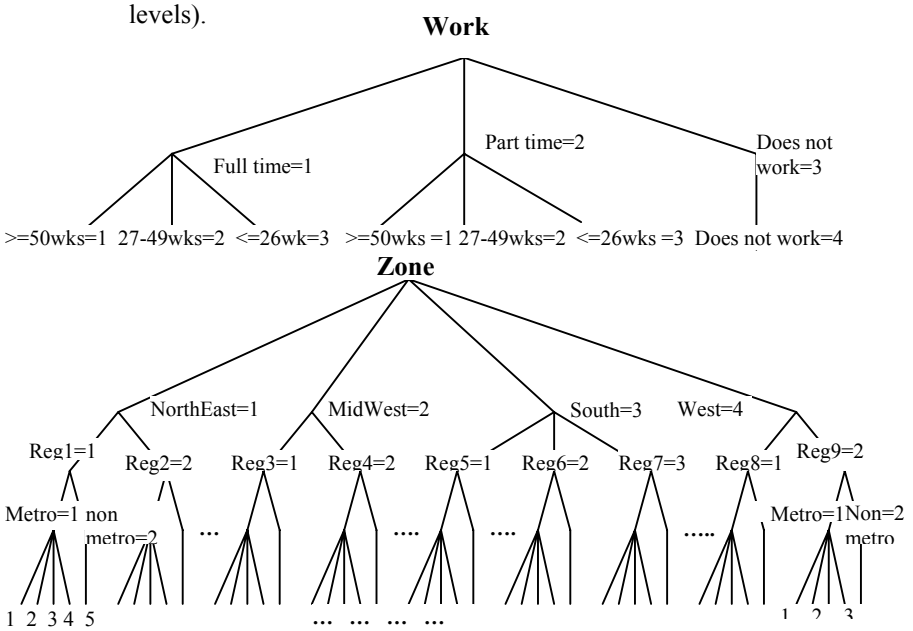
The multi-level methods have a lot of advantages. These methods respect the structure of the taxonomy. In particular, if all the values were at the lowest level then the "divisive" method and the method by aggregation would consider the variable like a classical qualitative variable whereas the multi-level methods would do each regression. Moreover, The multi-level methods bring us more information. A taxonomic variable could be able to explain a dependent variable at a level k but not at a level k+1 (for example, the variable "town" could be able to explain the variable "income" but this relation might disappear with the replacement of the cities by the regions). Reciprocally, when there are a lot of missing values at the level k, a regression can be interesting at a level k+1 but not interesting at a level k. Also, in the case of a taxonomic dependent variable, the predictions at the highest levels of the method by decomposition will improve the predictions at the lowest levels.

4.7 Comparison of the methods with a real data set

Test 1-All the explanatory variables are taxonomies.

Now, we compare the methods with a real-data set. We have:

- n = 10 000 individuals from the population of US.
- A quantitative dependent variable "Income";
- 2 explanatory variables with taxonomies: Zone (4 levels) and work (2 levels).



Test 2-With non-taxonomic variables.

We can add non-taxonomic variables to the taxonomic variables in order to see the differences in the results. So, we add to the previous example seven quantitative variables “age”, “gluc”, “chol”, “hemo”, “hemat”, “redBlood”, whiteBlood” and three qualitative variables “race”, “agegroup” and “diabetes”.

We build an array with the weights of the taxonomic variables and the non-taxonomic variables in the regression (in the previous test, weight of the taxonomies = 100%): $W(V)=\Sigma_m|coef(V,m)|$ where coef(V,m) is the coefficient of the modality m of the variable V in the regression. The weight of the taxonomic variables in the regression falls from 75% to 6% between the first and the fourth level.

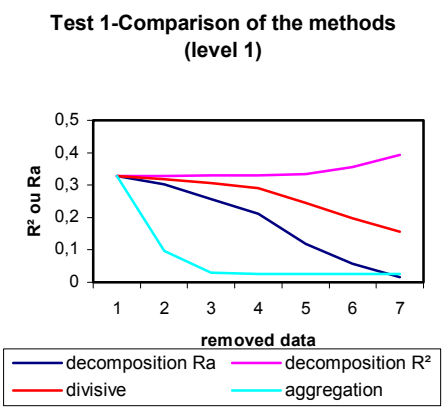
	Weights of the variables in the regression				
	level 1		**level 2**	**level 3**	**level 4**
Variable V	$\Sigma_m\|coef(V,m)\|$	%	%	%	%
constant	11689,8	**6,33%**	**7,10%**	**21,44%**	**31,04%**
Total of the taxonomies	139643,566	**75,66%**	**65,05%**	**26,70%**	**6,07%**
Total of the non taxonomies	33231,7624	**18,01%**	**27,84%**	**51,87%**	**62,89%**
Total	184565,128	100,00%	100,00%	100,00%	100,00%

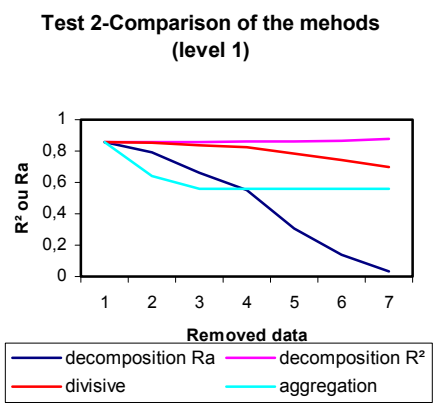
In order to compare the methods, we will do several regressions. For each regression, we will progressively remove some values and we will replace them by values at higher levels of generality. In fact, we increase the generality of the data.

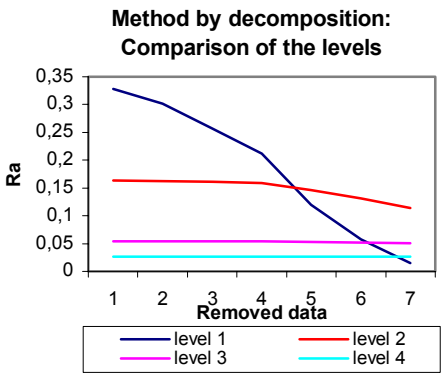
The determination coefficients R^2 will give us the best methods:
$R^2=\Sigma(y_i^*-y_{mean})^2/\ \Sigma\ (y_i-y_{mean})^2=SSE/SST\rightarrow 1$ when $SSE \rightarrow SST$ (Y = dependent variable, Y* = Y calculated with the equation).

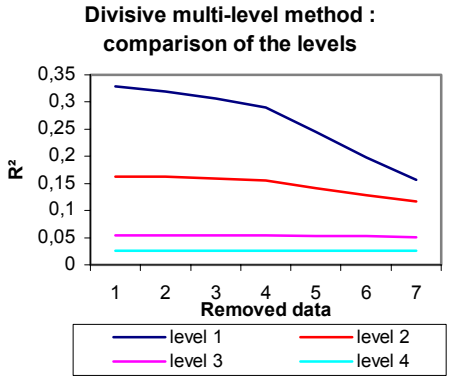
<u>Note:</u> For the method 1, we calculate an adjusted determination coefficient Ra. The idea is to penalize the coefficient when we remove individuals. So, we have $Ra=R^2*p$ where p=proportion of the individuals in the regression.

We represent the determination coefficients according to the removed data.







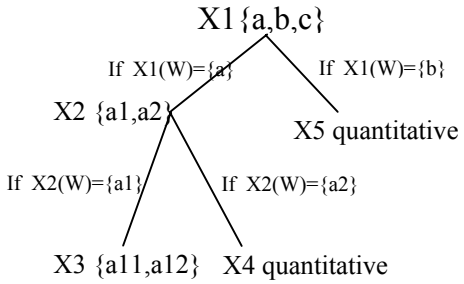


The multi-level "divisive" method has obtained the best results. The quality of the regression persists with the replacement of values at the lower levels by values at higher levels. We advocate this method because of the benefits obtained with the multi-level methods. However, the level 1 remains a long time the best level . So, we will use this level ("divisive" method) if we prefer only one equation. The method by decomposition is quite good when there are not non-taxonomic variables. In opposite of the divisive method, we have bad results with the second test because, like the taxonomic variables, the non-taxonomic variables are penalized by the removal of individuals. The adjusted determination coefficient Ra brings us to look the levels >= 2. In fact, if we consider that the number of individuals is sufficient in spite of the suppression of some of them then this method will be the best one. Indeed, with this method, the values are strictly correct. The method 2 will be able to give good results only if we remove a little part of the modalities.

5 Linear regression with hierarchies (mother-daughters variables)

5.1 Input data

We will use an example with five explanatory variables X1, X2, X3, X4, X5 organized in a hierarchy and a dependent quantitative variable Y. We see in the data matrix that we have to deal with non-sense values (NS):



i	Y	x1	x2	x3	x4	x5
1	5000	a	a1	a11	NS	NS
2	4500	a	a1	a11	NS	NS
3	4000	a	a1	a12	NS	NS
4	3500	a	a2	NS	10	NS
5	3000	a	a2	NS	20	NS
6	2500	b	NS	NS	NS	50
7	2200	b	NS	NS	NS	60
8	2000	b	NS	NS	NS	70
9	1800	b	NS	NS	NS	80
10	1000	c	NS	NS	NS	NS
11	1100	c	NS	NS	NS	NS
12	1200	c	NS	NS	NS	NS

5.2 Principle and algorithm of the regression

Note: The mother-daughters variables are the explanatory variables.

We do a regression with the mother and eventually the classical variables. Then, we improve the equation with the regressions of the daughters on the residuals.

(i) We do a regression with the variable at the top of the hierarchy and eventually the classical variables on Y.

(ii) We calculate the residuals Y-Y* where Y is the dependent variable and Y* is the variable Y calculated with the previous equation.

(iii) We do a regression with the daughters of the previous variable on the residual Y-Y*. In fact, we do a regression with the variables that have the most important number of relevant individuals. So, we do the regression with the relevant individuals (we remove the individuals with NS).

(iv) We add the new equation to the previous one.

(v) We come back to step 2.

5.3 Example

(1) We do a regression with Y and X1. We obtain the equation:

Y = 1100+2900*a(x1)+1025*b(x1).

(2) We calculate the residuals Y-Y*.

i	1	2	3	4	5	6	7	8	9	10	11	12
Y	5000	4500	4000	3500	3000	2500	2200	2000	1800	1000	1100	1200
Y*	4000	4000	4000	4000	4000	2125	2125	2125	2125	1100	1100	1100
Y-Y*	1000	500	0	-500	-1000	375	75	-125	-325	-100	0	1000

(3) We select the remaining variable(s) with the most important number of relevant individuals. Here, we select X2. We do the regression of X2 on the residuals Y-Y* with the individuals 1, 2, 3, 4, 5. We regress the matrix:

i	Y-Y*	X2
1	1000	a1
2	500	a1
3	0	a1
4	-500	a2
5	-1000	a2

We obtain the equation: Y-Y*= -750+1250*a1(x2).

(4) We add this new equation to the previous one conditionally to X1(w)={a}.

(5) We calculate the residuals Y-Y*.

i	1	2	3	4	5	6	7	8	9	10	11	12
Y	5000	4500	4000	3500	3000	2500	2200	2000	1800	1000	1100	1200
Y*	4500	4500	4500	3250	3250	2125	2125	2125	2125	1100	1100	1100
Y-Y*	500	0	-500	250	-250	375	75	-125	-325	-100	0	1000

(6) We select the remaining variable(s) with the most important number of relevant individuals. Here, we select X5. We do the regression of X5 on the residuals Y-Y* with the individuals 6, 7, 8, 9. We regress the matrix:

i	Y-Y*	X5
6	375	50
7	75	60
8	-125	70
9	-235	80

We have the equation Y-Y* = 1495-23*x5.

(7) We add this new equation to the previous one conditionally to X1(w)={b}.

(Next steps) We do the same with X3. X4 is not done because it remains only two individuals.

The algorithm stops because there are no more variables. We finally have the equation:

y=(+1100+2900*a(x1)+1025*b(x1)) R^2=0.86
If x1=a, (-750+1250*a1(x2))
If x1=b, (+1495-23*x5)
If x2=a1, (-500+750*a11(x3)) R^2=0.98

Verification of the quality of the regression.

i	1	2	3	4	5	6	7	8	9	10	11	12
Y	5000	4500	4000	3500	3000	2500	2200	2000	1800	1000	1100	1200
Y*	4750	4750	4000	3250	3250	2470	2240	2010	1780	1100	1100	1100
Y-Y*	250	-250	0	0	0	30	-40	-10	20	-100	0	1000

⇒R^2=0.9865.

5.4 Conclusions on the mother-daughters method

We first do a classical regression and second we add the regressions on the residuals to the initial equation. Each “daughter” variable improves the regression with the “mother”. This method remains efficient with big hierarchies because the

variables at the bottom have less importance than the variables at the top. Also, we can note that we can't apply the Fisher test to the whole regression. However, we can apply the test to each "sub-regression" and stop the algorithm when a "sub-regression" is refused.

6 Linear regression with diagrams

6.1 Input data

We show the method with an example. Let "Colors" be a diagram-valued explanatory variable and Y the quantitative dependent variable. We define a data matrix:

i	Colors	Y
1	(0.6) white, (0.4) yellow	1000
2	(0.3) blue, (0.7) red	1100
3	(0.2) white, (0.8) yellow	1200
4	(0.5) white, (0.5) blue	1300
5	(0.9) blue, (0.1) red	800

6.2 Principle of the method

Instead of building a binary matrix, we will build a matrix with the weights of the modalities. We will regress the matrix:

I	white	yellow	blue	red	Y
1	0.6	0.4	0	0	1000
2	0	0	0.3	0.7	1100
3	0.2	0.8	0	0	1200
4	0.5	0	0.5	0	1300
5	0	0	0.9	0.1	800

Note: we have to delete a modality in order to reverse the matrix.

In the case of several diagram-valued explanatory variables, the method remains the same.

7 SREG module

These different methods have been implemented in the SODAS software for ASSO ([2]). The SREG module provides methods and tests for multiple linear regression on symbolic data like intervals, histograms, taxonomies, mother-daughters variables, multi-nominal variables and diagrams.

8 Conclusions and perspectives

With these new methods, the linear regression has been extended to the taxonomic, the mother-daughters and the diagram-valued variables. However, there

are several ways to improve this work. Linear regression can be extended to other variables. For example, we can look for methods of regression when each value of a variable is a variable. Also, we can look for new tests in the case of symbolic linear regression. Moreover, it is interesting to study others symbolic regressions. Finally, we can think about methods to extract rules with symbolic regressions.

References

[1] BERTRAND P., GOUPIL F.: Descriptive Statistics for Symbolic Data; In Analysis Data Sets, Bock H.H., Diday E.; Springer; 2000.

[2] BIDSORF R., DIDAY E.: Symbolic Data Analysis and the SODAS Software in Official Statistics; In Data Analysis, Classification and Related Methods; Kiers et al. Editors; Springer; 2001.

[3] BILLARD L.& DIDAY E.: Symbolic Regression Analysis; In Classification, Clustering and Data Analysis; Bock, Jajuga, Sokolowski, Springer; 2002.

[4] BILLARD L.& DIDAY E.: Regression analysis for Interval-Valued Data; In Data analysis, classification and related methods (Kiers, Rasson, Groenen, Schader); Springer; 2001.

[5] BOCK HH. & DIDAY E.: Exploratory methods for extracting statistical information from complex data; In Analysis Data Sets, chapter 8 Similarity and dissimilarity; Springer Verlag; 2000.

[6] BRITO P., DIDAY E., KODRATOFF Y., MOULET M.: "Induction symbolique numérique à partir de données" ; Cépadues; 2000.

[7] DE CARVALHO F.: Méthodes descriptives en analyse de données symboliques; Doctoral thesis at University of Dauphine; 1992.

[8] DIDAY E.: Analyse des données symboliques: théorie et outil pour la fouille de connaissances; TSI (Technique et Science Informatiques), Vol 19, n°1-2-3 ; 2000.

[9] PRUM B.: Modèle linéaire : Comparaison de groupes et régression; Editions INSERM; 1996.

[10] RODRIGUEZ O.: Classification et modèles linéaires en analyse des données symboliques; Doctoral thesis at University of Paris Dauphine ; 2001.

[11] TENENHAUS M.: La régression qualitative; Revue de Statistiques Appliquées, vol. XXVII, n°2; 1979.

[12] ZIANI D.: Sélection de variables sur un ensemble d'objets symboliques; Doctoral thesis at University of Paris Dauphine; 1996.

INFORMATION SOCIETY TECHNOLOGIES
PROGRAMME

SMLP User Manual

Multi-Layer Perceptron

Edited by DAUPHINE

Project acronym: **ASSO**
Project full title: **Analysis System of Symbolic Official data**
Proposal/Contract no.: **IST-2000-25161**

ASSO Project
Workpackage WP6.2
Module SMLP

NEURAL NETWORKS MODULE

An Outline with options and formulas

Aboa Yapo Jean-Pascal

Lise-ceremade, Université Paris IX Dauphine
aboa@ceremade.dauphine.fr
Version by November 2003

CONTENTS

1 Input **3**
1.1 The Original dataset 3
1.2 Recoding data 7
1.3 The weights Matrix 11

2 Background **12**
2.1 The Multi-Layer Perceptron 12
2.2 Forward propagation 14
2.3 Activation Functions 14
2.4 Learning Method 16
2.5 Parameter Optimization Algorithm: Gradient descent 16
2.5.1 Simple Gradient 17
2.5.2 Conjugate Gradient : BFGS method 17
2.6 Back-propagation 18
2.6.1 Presentation 18
2.6.2 Algorithm of back-propagation 19

3 Output **20**
3.1 General results 20
3.2 Coding the output values 21

4 Way of working **25**

5 Example **27**

1 Input

1.1 The Original dataset

The neural networks method is constructed from a set of objects described by variables.

The basic building block with which the statistician deals is the variable. Statistical data are result of successive observations of some characteristic of a group. The observations, which are recorded as differences in magnitude or number are the value of the variable.

In a general situation we will consider a table $(X_j^{(i)}, Y_k^{(i)})_{i=1,\ldots n, j=1,\ldots p, k=1,\ldots c}$ of values on sets V_j as described below. Let o_i denote the object number i described by the ith-row $(X^{(i)} = (X_1^{(i)}, \ldots X_p^{(i)}), Y^{(i)} = (Y_1^{(i)}, \ldots Y_c^{(i)}))$. The whole set $\{s_1, \ldots, s_n\}$ will be denoted by S.

S	X_1	X_2	$\ldots$	X_p	Y_1	Y_2	$\ldots$	Y_c
s_1	$x_1^{(1)}$	$x_2^{(1)}$	$\ldots$	$x_p^{(1)}$	$y_1^{(1)}$	$y_2^{(1)}$	$\ldots$	$y_c^{(1)}$
s_2	$x_1^{(2)}$	$x_2^{(2)}$	$\ldots$	$x_p^{(2)}$	$y_1^{(2)}$	$y_2^{(2)}$	$\ldots$	$y_c^{(2)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_n	$x_1^{(n)}$	$x_2^{(n)}$	$\ldots$	$x_p^{(n)}$	$y_1^{(n)}$	$y_2^{(n)}$	$\ldots$	$y_c^{(n)}$

Table 1: Data table for n item $(s_1, \ldots, s_n)$ and variables (X, Y)

We have many kind of variables (numerical variable, interval-valued variable, nominal variable, multinominal variable, probabilistic variable).
We give an example for each type of variable.

- **Numerical variable** :
  We consider a table of numerical data for 6 item (s_i) $i = 1, \ldots, 6$ of objects and 2 numerical variables :
  X_1 : $Height$
  X_2 : $Weight$

S	X_1	X_2
s_1	1.65	68
s_2	1.70	69
s_3	1.60	65
s_4	1.82	75
s_5	1.90	80
s_6	1.55	60

Table 2: A data table for 6 item $(s_1, \ldots, s_6)$ and 2 numerical type variables (Height, weight)

- **Interval-valued variable** :
  We consider a table of interval-type data for 6 item (s_i) $i = 1, \ldots, 6$ of objects and 2 numerical variables :
  $var1$: $Height$
  $var2$: $Weight$

S	$Var1$	$Var2$
s_1	$[1.60, 1.70]$	$[63.0, 73.0]$
s_2	$[1.65, 1.75]$	$[64.0, 74.0]$
s_3	$[1.55, 1.65]$	$[60.0, 70.0]$
s_4	$[1.77, 1.87]$	$[70.0, 80.0]$
s_5	$[1.85, 1.95]$	$[75.0, 85.0]$
s_6	$[1.50, 1.60]$	$[55.0, 65.0]$

Table 3: A data table for 6 item $(s_1, \ldots, s_6)$ and 2 interval type variables (Height, weight)

- **Nominal variable** :
  We consider a table of nominal data for 6 item (s_i) $i = 1, \ldots, 6$ of objects and 2 nominal variables :
  $var1$: $Length(H = high, S = small)$
  $var2$: $Color(G = Greem, Y = Yellow)$

S	$Var1$	$Var2$
s_1	H	G
s_2	H	Y
s_3	S	Y
s_4	H	G
s_5	S	G
s_6	H	Y

Table 4: A data table for 6 item $(s_1, \ldots, s_6)$ and 2 nominal type variables (Length, Color)

- **Multinominal variable**
  We consider a table of multinominal data for 6 item (s_i) $i = 1, \ldots, 6$ of objects and 2 multinominal variables :
  X_1 : with modalities A, B, C
  X_2 : with modalities E, F

S	X_1	X_2
s_1	$\{A, B\}$	$\{E, F\}$
s_2	$\{A\}$	$\{F\}$
s_3	$\{A, B, C\}$	$\{E\}$
s_4	$\{C\}$	$\{E, F\}$
s_5	$\{B, C\}$	$\{E, F\}$
s_6	$\{A, B, C\}$	$\{E\}$

Table 5: A data table for 6 item $(s_1, \ldots, s_6)$ and 2 multinominal type variables (X_1, X_2)

- **Probabilistic variable**

  We consider a table of probabilistic data for 6 item (s_i) $i = 1, \ldots, 6$ of objects(for example the set of districts of a country) and 1 probabilistic variable :
  X_1 denotes the *occupation domain* :

  $X_1 = A \quad for$ Agriculture
  $X_1 = B \quad for$ Primary Production
  $X_1 = C \quad for$ manufacturin
  $X_1 = D \quad for$ Service
  $X_1 = E \quad for$ other cases

S	X_1
s_1	$(A10\%, B3\%, C11\% D42\% E34\%)$
s_2	$(A12\%, B6\%, C10\% D40\% E32\%)$
s_3	$(A15\%, B5\%, C13\% D37\% E30\%)$
s_4	$(A11\%, B9\%, C12\% D39\% E29\%)$
s_5	$(A13\%, B7\%, C15\% D30\% E35\%)$
s_6	$(A16\%, B8\%, C14\% D42\% E20\%)$

Table 6: A data table for 6 item $(s_1, \ldots, s_6)$ and 1 probabilistic type variable (Occupation)

1.2 Recoding data

In our program each observation is expected to be a numerical rating or a count of evens, but non-numeric codes can be used, if they are recoded in a numeric value.
Each variable needs a specific recoding model as described below, using an example in each case :

- **Numerical variable**

  In the case of numerical variable, no recoding is expected. The value of any object for a numerical variable is already a numeric value. The value of the numerical variable from the original database are directly used, without any modification.

  Original observation

S	Z_1	Z_2
s_1	1.65	68

  Recoding observation

S	Z_1	Z_2
s_1	1.65	68

- **Interval-valued variable**

  The value of an Interval-value variable is defined by the side : $[a, b]$, where a is the minimun of the side and b is the maximum of the side.

The recoding model for this kind of variable is : We construct an 2-dimensional value (m, σ),where m and σ are respectively the mean of a and b and the difference between a and b.

$$m = (a + b)/2$$

$$\sigma = b - a$$

Original observation

S	Z
s_1	$[1.60, 1.70]$

Recoding observation

S	m	σ
s_1	1.65	0.10

- **Nominal variable**

The value of a nominal variable Z are categorial value. Let $\{A_1, \ldots, A_m\}$ the set of m modalities of the nominal variable Y, so the recoding model for this type of variable is defined as follows :

A_1	is recoded as	$(1, 0, 0, \ldots, 0)$
A_2	is recoded as	$(0, 1, 0, \ldots, 0)$
$\vdots$	$\vdots$	$\vdots$
$\vdots$	$\vdots$	$\vdots$
A_m	is recoded as	$(0, 0, \ldots, 0, 1)$

Original observation

S	Y
s_1	A_k

Recoding observation

S	Z
s_1	$(0, \ldots, 0, 1, 0, \ldots, 0)$

The $m - dimensional$ value $(0, \ldots, 0, 1, 0, \ldots, 0)$ defined in the last table, place the value 1 at the k-th range.

- **Multinominal variable**

The multinominal variable is a generalization of the nominal variable, so the value of a multinominal variable with m modalities are recoded as :

$\{A_1\}$	is recoded as	$(1, 0, 0, 0, \ldots, 0)$
$\{A_1, A_2\}$	is recoded as	$(1, 1, 0, 0, \ldots, 0)$
$\{A_1, A_3\}$	is recoded as	$(1, 0, 1, 0, \ldots, 0)$
$\vdots$	$\vdots$	$\vdots$
$\vdots$	$\vdots$	$\vdots$
$\{A_1, A_2, A_3, A_4, \ldots, A_m\}$	is recoded as	$(1, 1, 1, 1 \ldots, 1)$

Original observation

S	Z
s_1	A_k, A_l

Recoding observation

S	Z
s_1	$(0, \ldots, 0, 1, 0, \ldots, 0, 1, 0, \ldots, 0)$

The $m - dimensional$ value $(0, \ldots, 0, 1, 0, \ldots, 0)$ defined in the last table, place the first value 1 at the$k - th$ range and the second one at $l - th$ range.

- **Probabilistic variable**

The Probabilistic variable is a super generalization of the multinominal variable. The recoding model for a probabilistic variable is defined through an example :

Original observation

S	Z
s_1	(A 10%, B 3%, C 11% D 42% E 34%)

Recoding observation

S	Z
s_1	$(0.10, 0.03, 0.11, 0.42, 0.34)$

1.3 The weights Matrix

In the next section, we will see that each unit accepts inputs data. We first multiply each of them by a corresponding weight and we then apply the sum of the weighted inputs to a transfert function to produce an input data.

We give a matrix $\mathcal{W}$ of weights wich dimension depends on the structure (described in section 2) of the neuronal network (number of inputs, number of units, number of layers).

For a layer l composed of $n^{(l)}$ units, we have the following matrix of weights $\mathcal{W}^{(l)}$:

$$\mathcal{W}^{(l)} = \begin{pmatrix} w_{11}^{(l)} & w_{12}^{(l)} & \cdots & w_{1n^{(l-1)}}^{(l)} \\ w_{21}^{(l)} & w_{22}^{(l)} & \cdots & w_{2n^{(l-1)}}^{(l)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n^{(l)}1}^{(l)} & w_{n^{(l)}2}^{(l)} & \cdots & w_{n^{(l)}n^{(l-1)}}^{(l)} \end{pmatrix}$$

We form the global weights matrix $\mathcal{W}$ by connecting all the matrix $\mathcal{W}^{(l)}$ for $l = 1, \ldots, L$ where L is the number of layers :

$$\mathcal{W} = (\mathcal{W}^{(1)}, \mathcal{W}^{(2)}, \ldots, \mathcal{W}^{(L)})$$

Many methods exist to initialize the weights w_{ij}. We give here two of these methods :

- The weights w_{ij} are chosen at random, according to the distribution of an uniform variable in interval $[-v_{max}, v_{max}]$ for each i and j.
- The weights w_{ij} are chosen at random, according to the distribution of a normal random variable (with $mean = 0$ and $variance = 1/(n^{(l-1)} - 1)$) for each i and j.

2 Background

2.1 The Multi-Layer Perceptron

The neural network is statistic analysis tool used to construct behaviour model from a set of examples.

A Multi-Layer Perception (MLP)is a type of neural network.It is composed of a number of active units. It can be used for classification or prediction of some behaviours.

We construct the result of a MLP by considering networks having successive layers of processing units, with connections running from every unit in one layer to every unit in the next layer.

A basic structure of a Multi-Layer Perception is :

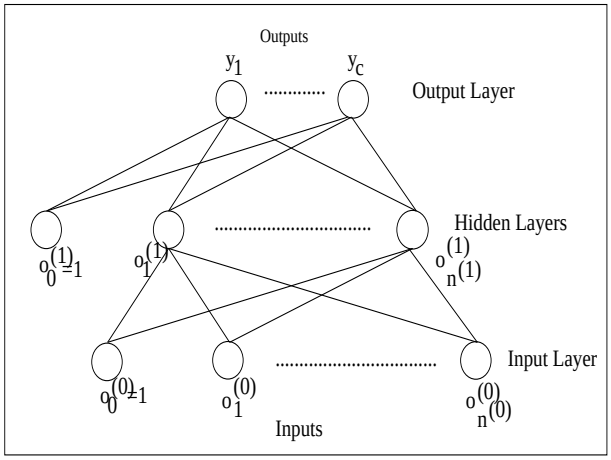


Figure 1: Example of a Multi-Layer perceptron structure, with 2 Layers

We construct here a network, with single-layers networks based on a linear combination of the input variables which is transformed by a non-linear activation function.

Our network have successive layers of processing units, with connections running from any unit in the next layer.

The output of the hidden unit is obtained by first forming a weight linear combination of the $d+1$ input values.

$$v_j^{(1)} = \sum_{i=0}^{i=n^{(0)}} w_{ji}^{(1)} o_i^{(0)} \tag{1}$$

where $w_{ji}^{(1)}$ denotes a weight in the first layer, going from input i to hidden unit j.

The activation of hidden unit j is :

$$o_j^{(1)} = T_j^{(1)}(v_j^{(1)}) \tag{2}$$

with $T_j^{(1)}(.)$ an activation function.

For each output unit j, a linear combination of the output of the hidden units give :

$$v_j^{(2)} = \sum_{i=0}^{i=n^{(1)}} w_{ji}^{(2)} o_i^{(1)} \tag{3}$$

where $w_{ji}^{(2)}$ denotes a weight in the hidden layer, going from input $o_i^{(1)}$ to output unit j.

The activation of jth output is :

$$y_j = T_j^{(2)}(v_j^{(2)}) \tag{4}$$

or

$$o_j^{(2)} = y_j = T_j^{(2)}(\sum_{k=0}^{k=n^{(1)}} w_{jk}^{(2)} T_k^{(1)}(\sum_{i=0}^{i=n^{(0)}} w_{ki}^{(1)} o_i^{(0)})) \tag{5}$$

Here $T_j^{(2)}$ is the activation function of the output unit.

2.2 Forward propagation

Consider the layer l of a Multi-Layer Perceptron. For each unit j of the layer l, we first compute a weighted sum of its inputs.

And then we transforme this sum, by using an activation function $T_j^{(l)}$. So the forward propagation formulas are :

$$v_j^{(l)} = \sum_{i=0}^{n^{(l-1)}} w_{ji}^{(l)} o_i^{(l-1)} \tag{6}$$

$$o_j^{(l)} = T_j^{(l)}(v_j^{(l)}) \tag{7}$$

where $o_i^{(l-1)}$ is the activation of a unit, which sends a connection to unit j; $w_{ji}^{(l)}$ is the weight associated with that connection and $n^{(l-1)}$ is the number of units in the layer $l-1$.

2.3 Activation Functions

The multi-layer perceptron use sigmoid activation functions.

- The logistic sigmoid activation function

  It is often used for the hidden units of a multi-layer perceptron :

$$T(x) = \frac{1}{1 + exp(-x)} \tag{8}$$

- The 'tanh' activation function

$$T(x) = \frac{exp(x) - exp(-x)}{exp(x) + exp(-x)} \tag{9}$$

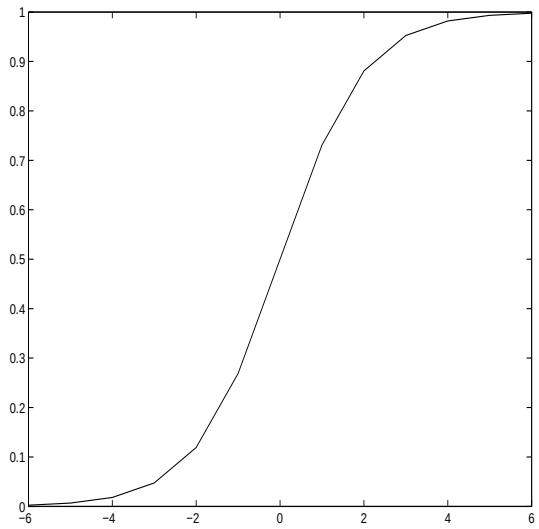


Figure 2: The logistic sigmoid activation function

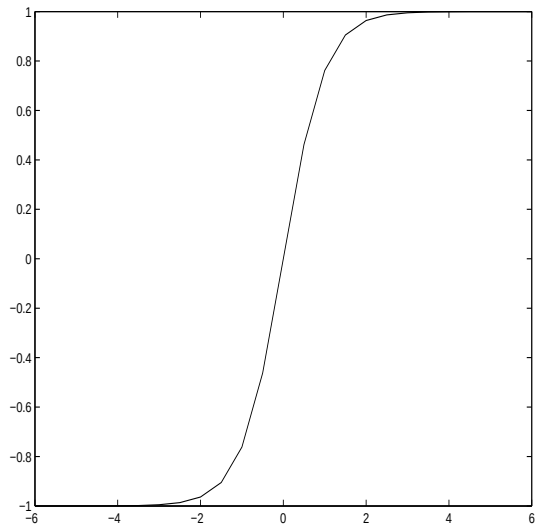


Figure 3: The logistic sigmoid activation function

2.4 Learning Method

Consider a set of N objects described by $(x^{(i)}, y^{(i)})1 \leq iN$. Let $x \mapsto F(x, w)$ be a behaviour model for this dataset.

- The goal of the method is to find the best w, for which, the model $x \mapsto F(x, w)$ mimic the behaviour of the data as well as possible.
- We minimize the error

$$E(w) = \sum_{i=1}^{N} d(F(x^{(i)}, w) - y^{(i)}) \tag{10}$$

  where d is an individual Error function.
  In the case of regression the error function that we minimize is :

$$E(w) = \sum_{i=1}^{N} \parallel d(F(x^{(i)}, w) - y^{(i)}) \parallel^2 \tag{11}$$

2.5 Parameter Optimization Algorithm: Gradient descent

The algorithm of the minimization of the error $E(w)$ is :

(1) Starting value of the weight vector $w^{(0)}$
(2) Stage k :

- We evaluate the direction of gradient $d^{(k)}$;
- We evaluate a step $\epsilon^{(k)}$
  (for example $\epsilon^{(k)} = constante$ or $\epsilon^{(k)} = 1/(\alpha + \beta * k)$);
- We change $w^{(k-1)}$ into :

$$w^{(k)} = w^{(k-1)} + \epsilon^{(k)} * d^{(k)}$$

(3) if $E(w^{(k)})$ gives us satisfaction then end, else we return to stage (2).

2.5.1 Simple Gradient

In this case, we iteratively update the weight vector such that, at each stage k, we move a short distance in the greatest rate of decrease of the error. The direction of the gradient is defined as :

$$d^{(k)} = \nabla E(w^{(k-1)}) \tag{12}$$

2.5.2 Conjugate Gradient : BFGS method

Here the direction of the gradient is defined by :

$$d^{(k)} = H^{(k)} \nabla E(w^{(k-1)}) \tag{13}$$

And so we change $w^{(k)}$ into $w^{(k+1)}$ according the following program :

$$\begin{cases} w^{(k+1)} = w^{(k)} + \epsilon^{(k)} * H^{(k)} \nabla E(w^{(k)}) \\ H^{(k+1)} = H^{(k)} + correction \end{cases}$$

where the correction formula is given by :

$$\begin{aligned} H^{(k+1)} &= H^{(k)} + \frac{\delta^{(k+1)} \otimes \delta^{(k+1)}}{\delta^{(k+1)}.\delta\nabla^{(k+1)}} - \frac{H^{(k)}.\delta\nabla^{(k+1)} \otimes H^{(k)}.\delta\nabla^{(k+1)}}{\delta\nabla^{(k+1)}.H^{(k)}.\delta\nabla^{(k+1)}} \\ &\quad + [\delta\nabla^{(k+1)}.H^{(k)}.\delta\nabla^{(k+1)}] u \otimes u \end{aligned} \tag{14}$$

where we have defined the following vectors :

$$\delta^{(k+1)} = w^{(k+1)} - w^{(k)} \tag{15}$$

$$\delta\nabla^{(k+1)} = \nabla E(w^{(k+1)}) - \nabla E(w^{(k)}) \tag{16}$$

$$u = \frac{\delta^{(k+1)}}{\delta^{(k+1)}.\delta\nabla^{(k+1)}} - \frac{H^{(k)}.\delta\nabla^{(k+1)}}{\delta\nabla^{(k+1)}.H^{(k)}.\delta\nabla^{(k+1)}} \tag{17}$$

2.6 Back-propagation

2.6.1 Presentation

An important point in the descent gradient method is the evaluation of the error function derivatives. The back-propagation algorithm is used to evaluate,efficiently the function error derivatives.

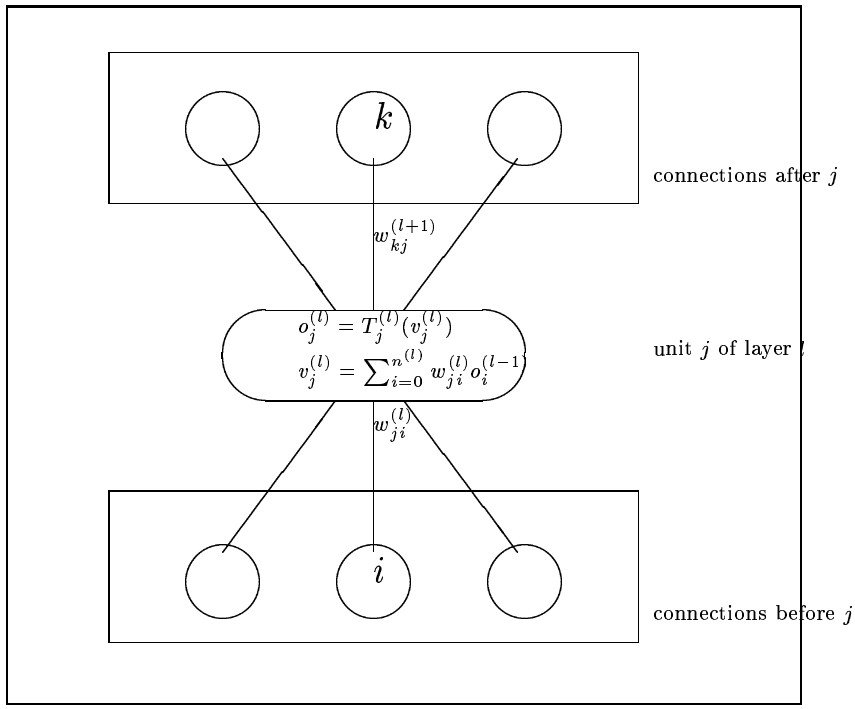


Figure 4: Presentation of a hidden unit j with its connections

To evaluate the error function (10) derivatives, we just have to know how evaluate the derivatives of the individual error function d.

Noting that the error function d depends on the weight $w_{ji}^{(l)}$ only via the summed input $v_j^{(l)}$, we derive the derivatives of d to the weight $w_{ji}^{(l)}$ by applying the chain rule for partial derivatives :

$$\frac{\partial d}{\partial w_{ji}^{(l)}} = \frac{\partial d}{\partial v_j^{(l)}} * \frac{\partial v_j^{(l)}}{\partial w_{ji}^{(l)}} \tag{18}$$

$$\frac{\partial d}{\partial w_{ji}^{(l)}} = \frac{\partial d}{\partial v_j^{(l)}} * o_i^{(l-1)} \tag{19}$$

Set :

$$\delta_j^{(l)} = \frac{\partial d}{\partial v_j^{(l)}} \tag{20}$$

For the output units the evaluation of $\delta_k^{(l)}$ is :

$$\delta_k^{(l)} = \frac{\partial d}{\partial o_k^{(l)}} * T_k^{(l)\prime}(v_k^{(l)}) \tag{21}$$

For the hidden units the evaluation of $\delta_j^{(l)}$ is :

$$\delta_j^{(l)} = T_j^{(l)\prime}(v_j^{(l)}) * \sum_k \frac{\partial d}{\partial v_k^{(l+1)}} w_{kj}^{(l)} \tag{22}$$

2.6.2 Algorithm of back-propagation

Consider a sample of N objects; a MLP described by an input layer and $L-1$ hidden layers.

For an example $(x^{(i)}, y^{(i)})$ of the dataset, we calculate the output o of the network through the forward propagation (formulas (6) and (7))

```
For each output unit, evaluate δ_k according to formula (21)
for each Layer l from L − 1 to 1
    For each unit j of this layer l
        begin
            First evaluate δ_j for each hidden unit j according to (22)
            Then evaluate the derivatives according to formula (19)
        end
end
```

3 Output

3.1 General results

The module SMLP provides some pertinent informations :

- The results of the optimization of the error function $E(\mathcal{W})$ presented in section 2.5.

  For a layer l composed of $n^{(l)}$ units, we have the following matrix of weights $\mathcal{W}^{*(l)}$:

$$\mathcal{W}^{*(l)} = \begin{pmatrix} w_{11}^{*(l)} & w_{12}^{*(l)} & \cdots & w_{1n^{(l-1)}}^{*(l)} \\ w_{21}^{*(l)} & w_{22}^{*(l)} & \cdots & w_{2n^{(l-1)}}^{*(l)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n^{(l)}1}^{*(l)} & w_{n^{(l)}2}^{*(l)} & \cdots & w_{n^{(l)}n^{(l-1)}}^{*(l)} \end{pmatrix}$$

$$\mathcal{W}^* = (\mathcal{W}^{*(1)}, \mathcal{W}^{*(2)}, \ldots, \mathcal{W}^{*(L)})$$

- Let $O = (O_1, \ldots O_c)$ the output vector given by the forward propagation. We give the database where the vector $Y = (Y_1, \ldots Y_c)$ (of table 1) is replaced is here by the vector $O = (O_1, \ldots O_c)$:

S	X_1	X_2	$\ldots$	X_p	O_1	O_2	$\ldots$	O_c
s_1	$x_1^{(1)}$	$x_2^{(1)}$	$\ldots$	$x_p^{(1)}$	$o_1^{(1)}$	$o_2^{(1)}$	$\ldots$	$o_c^{(1)}$
s_2	$x_1^{(2)}$	$x_2^{(2)}$	$\ldots$	$x_p^{(2)}$	$o_1^{(2)}$	$o_2^{(2)}$	$\ldots$	$o_c^{(2)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_n	$x_1^{(n)}$	$x_2^{(n)}$	$\ldots$	$x_p^{(n)}$	$o_1^{(n)}$	$o_2^{(n)}$	$\ldots$	$o_c^{(n)}$

Table 7: Data table for n item $(s_1, \ldots, s_n)$ and variables (X, O)

- The Classifying error E_c is given by :

$$E_c = \frac{1}{n * c} \sum_{i=1}^{n} \sum_{k=1}^{c} (y_k^{(i)} - o_k^{(i)})^2 \tag{23}$$

- Confusion matrix

$$\Delta = \begin{pmatrix} \delta_{11} & \delta_{12} & \cdots & \delta_{1c} \\ \delta_{21} & \delta_{22} & \cdots & \delta_{2c} \\ \vdots & \vdots & \ddots & \vdots \\ \delta_{c1} & \delta_{c2} & \cdots & \delta_{cc} \end{pmatrix}$$

where

$$\delta_{ij} = \sum_{l=1}^{n} \delta_{ij}^{(l)} \tag{24}$$

$$\delta_{ij}^{(l)} = \begin{cases} 1 & \text{if } index \left\{ max(y_1^{(l)}, \ldots, y_c^{(l)}) \right\} = i \text{ and } index \left\{ max(o_1^{(l)}, \ldots, o_c^{(l)}) \right\} = j \\ 0 & \text{else} \end{cases}$$

3.2 Coding the output values

Our program use numerical, nominal, interval-valued, multinominal or probabilistic valued variables.

We have seen in section 1.2 that we construt numerical value variables when it is necessary by using a particular recoding method for each original variable.

At the end of the program, the we come back to the original model of each variavle by using coding method that we are going to define soon :

- **Numerical variable**
  When the variable is numeric, no coding is expected as we have seen in section 1.2.

  Output observation

S	Y
s_1	1.5060

  Coding observation

S	Y
s_1	1.5060

- **Interval-valued variable**
  The output value of an interval-valued variable is given as a 2-dimensional vector (y_1, y_2).
  The values y_1 and y_2 are respectively considered as the mean of an interval $[a, b]$ and the difference between a and b. So we obtain the minimum a and the maximum b of the side of $[a, b]$ by the following formulas :

$$a = (-y_2 + 2 * y_1)/2$$

$$b = (+y_2 + 2 * y_1)/2$$

  Output observation

S	y_1	y_2
s_1	1.50	0.6

Coding observation

S	Y
s_1	$[1.20, 1.80]$

- **Nominal variable**
  The output value of a nominal variable Y is a $m-dimensional$ vector of numerical values, where m is the modalities number of Y.
  Let denote $\{A_1, \ldots, A_m\}$ the set of the m modalities of the nominal variable Y, so the coding model is defined as follows :

$$\begin{array}{cccc}
(\mathrm{y}_1 \ldots, y_m) & \text{is coded as} & \mathrm{A}_1 & \text{if index}\{max(y_1 \ldots, y_m)\} = 1 \\
(\mathrm{y}_1 \ldots, y_m) & \text{is coded as} & \mathrm{A}_2 & \text{if index}\{max(y_1 \ldots, y_m)\} = 2 \\
\vdots & \vdots & \vdots & \vdots \\
(\mathrm{y}_1 \ldots, y_m) & \text{is coded as} & \mathrm{A}_i & \text{if index}\{max(y_1 \ldots, y_m)\} = i \\
\vdots & \vdots & \vdots & \vdots \\
(\mathrm{y}_1 \ldots, y_m) & \text{is coded as} & \mathrm{A}_m & \text{if index}\{max(y_1 \ldots, y_m)\} = m
\end{array}$$

Output observation

S	Y
s_1	$(0.20, -0.35, 0.40, 0.37, 0.25)$

Coding observation

S	Y
s_1	A_3

In this example, $m = 5$ and :

$$index\ \{max(0.20, -0.35, 0.40, 0.37, 0.25)\} = 3$$

- **Multinominal variable** An output value of a be a multinominal variable Y of m dimension, is written like :

$$y = (y_1, \ldots, y_m)$$

where for each j, the value y_j is a numerical value. If we set :

$$M = \frac{1}{m}\sum_{j=1}^{m} y_j$$

So the coding model is :

$$(y_1, \ldots, y_m) \text{ is coded as } (y_1/M, \ldots, y_m/M)$$

We obtain a probabilistic data because we have :

$$\frac{y_1}{M} + \ldots + \frac{y_m}{M} = 1$$

Output observation

S	Y
s_1	$(0.2; 0.8; 0.4; 0.6)$
s_2	$(0.6; 0.3; 1.1; 2.0)$

Coding observation

S	Y
s_1	$(\frac{0.2}{2}; \frac{0.8}{2}; \frac{0.4}{2}; \frac{0.6}{2})$
s_2	$(\frac{0.6}{3}; \frac{0.3}{3}; \frac{1.1}{3}; \frac{2.0}{3})$

- **Probabilistic variable** The output data of a probabilistic variable is coded as the one of a multinominal variable.

4 Way of working

We consider a database which is composed of inputs and outputs. We want to use this base to derive an automatic learning. The goal of the method is to evaluate the optimal weights by using backpropagation.

We develop an algorithm which is composed of an input layer (of $n^{(0)}$ units), a hidden layer (of $n^{(1)}$ units) and an output layer (of c units). The general steps of the algorithm is specified as follows :

- *Input*:
  A sample S of objects $s_1, \ldots, s_n$, characterized the vector $(x_1^{(i)}, \ldots, x_p^{(i)})$ and $(y_1^{(i)}, \ldots, y_c^{(i)})$

- *Recoding data*
  We recode the original observations when it is necessary.

- *Weights initialization*
  The initial weights w_{ij} are chosen at random, according to the distribution of an uniform variable in interval $[-0.5, 0.5]$ for each i and j.

- *Iteration from 1 to n items*

  (1) Forward propagation
  For an example s_i of S, we evaluate the output vector $o = (o_1^{(i)}, \ldots, o_c^{(i)})$.

  (2) Backpropagation
  For each output unit k, we evaluate :

$$\delta_k = \frac{\partial d}{\partial o_k} * T'_k(v_k)$$

  For the hidden units j, we evaluate :

$$\delta_j = T'_j(v_j) * \sum_k \delta_k w_{kj}$$

  We evaluate $\epsilon^{(i)}$.

(3) Update the weights when $E(\mathcal{W}^{(i)})$ is acceptable
For each weight w_{ij}, we change it to : $w_j^{(i)} + \epsilon^{(i)} * \delta_i * x_j^{(i)}$

- *Iteration and stopping* The steps 1 to 3 are iterated until :
  - The number of iteration
  - The error $E(\mathcal{W})$
  - The increasing rate of the error

  satisfy respectively a given stopping criterion

- *Output*
  A MLP defined by the initial structure and the optimal matrix $\mathcal{W}^*$ of weights.

5 Example

- We consider the XOR example (a MLP of one input layer of 2 units, one hidden layer of 2 hidden units and one output layer).

S	X_1	X_2	Y
s_1	0.00	0.00	b
s_2	1.00	1.00	b
s_3	0.00	1.00	a
s_4	1.00	0.00	a

Table 8: A data table for 4 item $(s_1, \ldots, s_4)$ and 2 numerical type variables (X_1, X_2) and 1 nominal variable Y.

- After recoding this database, we obtain :

S	X_1	X_2	Y_1	Y_2
s_1	0.00	0.00	0	1
s_2	1.00	1.00	0	1
s_3	0.00	1.00	1	0
s_4	1.00	0.00	1	0

Table 9: A data table after recoding.

- The initial matrice of weights is :

$$\mathcal{W}_0 = \begin{pmatrix} 1.00 & 1.00 & -0.50 & 1.00 & -1.00 \\ 1.00 & 1.00 & -1.50 & -1.00 & 1.00 \end{pmatrix}$$

- Stopping criterion
  Maximum Number of iteration (noted Max_iteration)= 4
  Minimum value of the error $E(\mathcal{W})$ (noted Min_norm)=2.5
  Minimum value of the progression rate of the error (noted Min_progression)=0.00001

- The first iteration

  For the first example $s_1 = (0.00, 0.00, 0, 1)$, it means that :

$$(x_1^{(1)}, x_2^{(1)}) = (0, 0)$$

$$(y_1^{(1)}, y_2^{(1)}) = (0, 1)$$

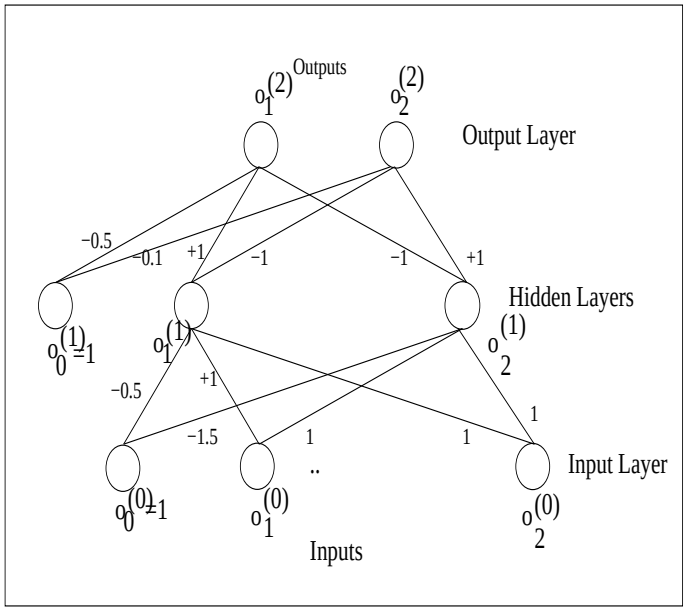


Figure 5: Example of a two-Layer perceptron for the XOR problem

- The forward propagation program (with 'tanh' sigmod activation function) gives (after a relevant number of sub-iteration) the followings results for the obsrvation s_1 :

for the output of the hidden units

$$o_1^{(1)} = -0.554730$$

$$o_2^{(1)} = -0.899843$$

for the output of the neural networks

$$o_1^{(2)} = -0.139868$$

$$o_2^{(2)} = -0.099071$$

- The value of the error at this step is :

$$E(\mathcal{W}_0) = 5.322879$$

- The update of the weights matrix is :

$$\mathcal{W}_1 = \begin{pmatrix} 1.0000 & 1.0000 & \text{-}0.6251 & 0.9965 & \text{-}1.0781 \\ 1.0000 & 1.0000 & \text{-}1.4714 & \text{-}1.0065 & 0.8491 \end{pmatrix}$$

- value of the progression rate of the error :

$$E(\mathcal{W}_0) = 5.322879$$

$$E(\mathcal{W}_1) = 5.321730$$

$$progression = \frac{E(\mathcal{W}_1) - E(\mathcal{W}_0)}{E(\mathcal{W}_0)} = 0.000216$$

Remark 5.1 *We note that the stopping criteria are not reached, indeed we have :*

$$E(\mathcal{W}_0) = 5.322879 > Min_norm = 2.5$$

$$E(\mathcal{W}_1) = 5.321730 > Min_norm = 2.5$$

$$progression = 0.000216 > Min_progression = 0.00001$$

So we can continue the iteration.

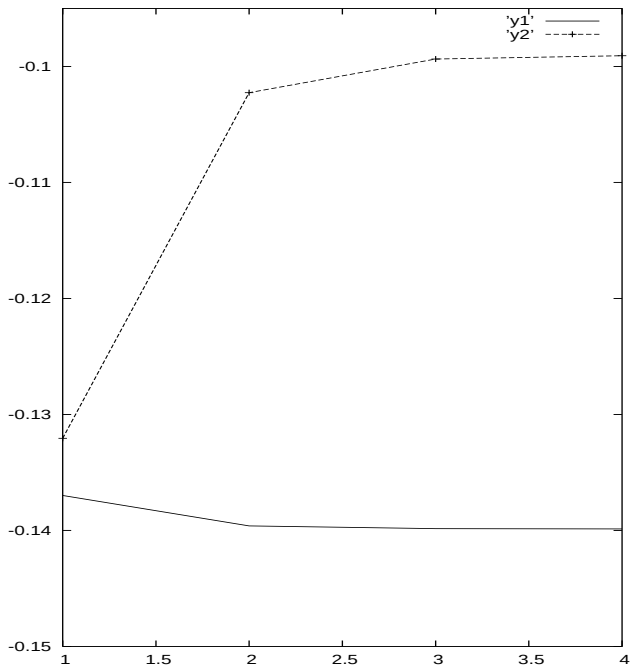


Figure 6: The evolution of y_1 and y_2 during the first iteration

- The last iteration (it's the fourth one) gives followings results :

  - The forward propagation program (with 'tanh' sigmod activation function) gives (after a relevant number of sub-iteration) the followings results for the obsrvation s_4 :

  for the output of the hidden units

$$o_1^{(1)} = 0.313477$$
$$o_2^{(1)} = -0.285036$$

  for the output of the neural networks

$$o_1^{(2)} = 0.121905$$
$$o_2^{(2)} = -0.363160$$

  - The value of the error at this step is :

$$E(\mathcal{W}_3) = 4.090423$$

  - The update of the weights matrix is :

$$\mathcal{W}_4 = \begin{pmatrix} 0.9646 & 0.9646 & \text{-0.6402} & 1.0075 & \text{-0.9283} \\ 1.0917 & 1.0917 & \text{-1.3849} & \text{-0.9998} & 0.9511 \end{pmatrix}$$

  - value of the progression rate of the error :

$$E(\mathcal{W}_3) = 4.090423$$
$$E(\mathcal{W}_4) = 3.280095$$

$$progression = \frac{E(\mathcal{W}_4) - E(\mathcal{W}_3)}{E(\mathcal{W}_3)} = 0.198104$$

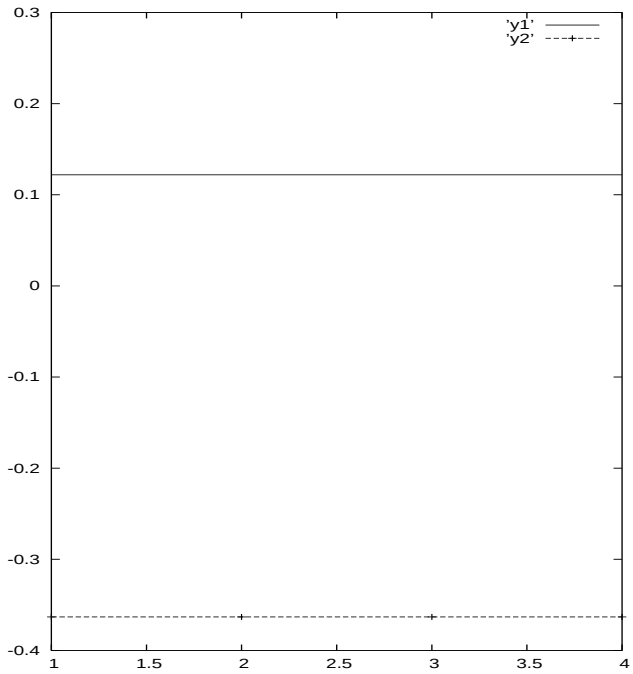


Figure 7: The evolution of y_1 and y_2 during the last iteration

- Database where the input vector $Y = (y_1, y_2)$ is replaced here by the output vector $O = (O_1, O_2)$

S	X_1	X_2	O_1	O_2
s_1	0.00	0.00	-0.166350	-0.115469
s_2	1.00	1.00	-0.276740	0.032517
s_3	0.00	1.00	0.121905	-0.363160
s_4	1.00	0.00	0.121905	-0.363160

Table 10: Data table of output before coding.

S	X_1	X_2	$Coding$
s_1	0.00	0.00	b
s_2	1.00	1.00	b
s_3	0.00	1.00	a
s_4	1.00	0.00	a

Table 11: Data table of output after coding.

- Others resutls

$$Confusion = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$

Quadratic error = 4.090423

Classifying error = 0.00000