The Hebrew University of Jerusalem
Computation Authority

Hudap Manual

with ***Mathematics***

and **Windows Interface**

Reuven Amar Shlomo Toledano

Jerusalem 2001, Second Edition

Reuven Amar
Computation Authority
The Hebrew University of Jerusalem
Mount Scopus, Jerusalem
Israel
E-mail: reuben@mscc.huji.ac.il
Phone: 972-2-5881088
Fax: 972-2-5828826

Shlomo Toledano
Computation Authority
The Hebrew University of Jerusalem
Mount Scopus, Jerusalem
Israel
E-mail: toledaa@mscc.huji.ac.il
Phone: 972-2-5883071
Fax: 972-2-5828826

Table of Contents

Introduction

In his paper: 'What is Not What in Statistics'[1], Guttman presents a list of the disadvantages of classical Statistics. The classical methods of statistical inference have proven themselves incapable of leading researchers to the formulation of useful, widely applicable rules, by making some frequently unrealistic assumptions, such as normal distribution, quantitative variables, etc..., by misusing the significance notion and because of requesting 'representative samples'.

In recent years eminent statisticians such as Tukey, Kruskal and others, have pointed out the limitations of statistical inference. There is increasing emphasis on the need for focusing on data analysis instead.

Contrary to classical Statistics, where packages of programs are numerous, the data analysis domain is still poor in packages. The existing data analysis programs are seen as supplementary to classical Statistics. The package HUDAP (Hebrew University Data Analysis Package) is intended to reorient researchers from classical statistics to the more efficient methods of data analysis.

HUDAP contains mainly programs based on the methods developed by Guttman, which are amply used in social science research. It also contains general purpose programs which yield descriptive statistics. Hence, HUDAP is a multi-purpose package.

HUDAP has a flexible, easy to use command language. HUDAP contains two kinds of variables: vectors and scalars. The vectors are used for holding data; scalars hold counters and intermediate results.

Following a summary of the chapters available in the HUDAP Manual.

Overview

The Overview contains examples of various manipulations performed on a sample data set. It takes the reader, step by step, through various procedures, explaining the analysis that is to be performed and describing the steps of the procedure.

No prior background or familiarity with HUDAP is necessary in order to understand the examples in the Overview. The examples serve as a useful

[1]Guttman, L. What is not what in statistics. The Statistician, 1977, 26,81-107.

introduction to HUDAP for those who are not familiar with the package; the techniques and the instructions for executing the HUDAP procedures are self-explanatory.

In addition, the Overview contains brief formal introductions to the various HUDAP data analysis jobs.

Control Language

In order to execute a HUDAP procedure (here termed section), the user must write what is called a "control file," that is, a file that contains the instructions HUDAP carry out on the data. The data analysis to be performed must be described in "control language," the language HUDAP "understands." HUDAP's control language is English-based, flexible, and easy to understand. The chapter on control language offers a formal introduction to the basics of the language.

One who wishes to perform a series of data analysis manipulations on an existing data file creates a "control file" with the appropriate control language instructions. This file is then taken as the HUDAP input file (not the input data file), and when HUDAP is instructed to execute this control file, it produces a HUDAP output file which contains the results of the analysis or error messages indicating mistakes in the input control file.

Note: Depending on the system on which you are working, (UNIX, VMS, DOS, etc.) the line instruction necessary to execute HUDAP might differ. Consult your local experts to ascertain what instruction you should use.

DATA

The HUDAP control file must contain a section which describes the data in the data file the user intend to analyse or transform. In the **DATA** section the user names and locates the variables to be read for each subject (case) in the data file, and where the data are located (pathname of the file); missing value codes as well as variable and category labels can be specified.

CODING and *COMPUTE*

Often the user may need to reorganise the data in the data file in order to check for errors, make the expected analysis more efficient, or compute new variables

with the raw data. The **CODING** and **COMPUTE** sections are powerful tools for transforming data in this way. The user can perform data-cleaning checks, correct coding errors, rescale variables, change a coding scheme, and carry out many other essential tasks to prepare data for analysis, or examine it after preliminary analysis.

The **CODING** section changes the coding scheme of an existing variable on a value-by-value basis or for ranges of values. It allows the user, for example, to change the coding order for a questionnaire's item from **0** for "**Yes**" and **1** for "**No**" to **1** for "**Yes**" and **0** for "**No**." In addition, you can change a range of values: recoding values of **1** through **10** as **1**, **11** through **20** as **2**, and so on.

The **COMPUTE** section allows the user to create new variables or transform existing variables using information from any variables in the data file. The user may wish to compute the sum or the mean of several variables, or perform a number of calculations on the data, find the minimal or maximal value of a particular variable, or find the number of valid entries for a given variable. The user might also wish to perform transformations that require conditional computations: e.g., handling a particular variable in one way if its values are below 10 and in a different way if its values are equal to or greater than 10. The **COMPUTE** section allows the user to carry out these and many other possibilities.

MATRINP

There may be circumstances in which it is advantageous to input data in the form of a matrix rather than as raw data. In such cases, the **MATRINP** section (which is analogous to the **DATA** section for raw data) allows this option. The matrix can be then processed by another section as for example **WSSA1**.

MODIFILE

It is often necessary to analyse a set of data many times. For example, preliminary analysis may indicate that data transformation is advantageous before proceeding to later analysis. It might also be desirable simply to study the results of a preliminary analysis before proceeding to later analysis.

The **MODIFILE** section allows the user to store data or the results of a preliminary analysis in HUDAP (system) File. The data is then easily retrievable. Furthermore, the HUDAP File contains the number of variables, the variable names, the scalar variable names and the flags for missing values. Thus, you need not specify this information again when you use this HUDAP File at a later stage as input, by means of **RESTORE** section.

MODIFILE section can be used also to select cases via the conditional **IF...ENDIF** feature.

SET

The **SET** section can accompany any HUDAP job, but it is optional. In this section the user sets parameters such as linesize or pagesize, gives a title to the job, or requests the **EDIT** or **CHECK** commands. (**EDIT** instructs HUDAP to inspect the control file for syntax errors. **CHECK** instructs HUDAP to check a computational error, aborts the job, and provides an error message.)

FOR...ENDFOR

At times it is necessary to perform the same operation on different variables or for different options. In such cases, the **FOR** section is a convenient feature. The **FOR...ENDFOR** facility can be used for computations involving repeated calculations or for repeated execution of different HUDAP sections (e.g. **MONCO**, **FREQ**, **CRST**, etc.). In addition, nesting **FOR...ENDFOR** blocks within each other can serve as a powerful tool for simplifying HUDAP jobs.

INFO

The **INFO** section provides information on current variables and parameters of the data file, like number of variables, number of cases, missing values, variable and category labels etc..

OUTPUT

The user may wish at times to create and save a file of data with the results of a preliminary analysis or data transformation. The **OUTPUT** section allows the user to create such a file, which can be printed out and examined or inputted into HUDAP or any other foreign program at a later stage.

FREQ

The **FREQ** section produces a table of frequency counts and percentages for values of variables. Variables are of any type: integer, real or character. One can also obtain histograms for interval variables, univariate summary statistics and percentiles.

CRST

The **CRST** section produces crosstabulations containing the joint distribution of two variables that have a limited number of distinct values. The frequency distribution of one variable is subdivided according to the values of the other variable. The combination of one value of the first variable with one value of the second variable defines a cell - the basic element of **CRST** tables.

In addition to cell counts, the user can obtain cell percentages and optional measures of association (Monotonicity coefficient **MONCO** between the two variables, Pearson correlation coefficient, Chi-square and Dependency coefficients). A matrix of these coefficients can be supplied, for example, to a **WSSA1** section as an input matrix.

MULTABS

The **MULTABS** section produces crosstabulations of one variable (columns) with a series of variables (rows). The output of **MULTABS** is an easily readable labelled table which can show at a glance the desired cross-tabulations. The output is formatted in rows and columns where the columns correspond to the categories of one variable and the rows to the categories of several variables.

If, for example, the user has data indicating the levels of education and income of a given population, and s/he wishes to examine the difference in education and income levels between men and women s/he can produce a **MULTABS** table which consists of two columns, men and women, and the levels of education and income as rows. The table may consist of the raw cell counts row percentages, column percentages, or percentages from the total frequency. (Only one of these options can be obtained in a single **MULTABS** section.)

MONCO

The **MONCO** section computes Guttman weak monotonicity coefficients for pairs of ordinal or interval variables.

This coefficient, designated as μ_2, expresses the extent to which replies to one question increase in a particular direction as the replies to the other question increase, without assuming that the increase is exactly according to a straight line. This coefficient varies between -1 and $+1$. $\mu_2 = +1$ implies a fully monotone relationship with positive or rising trend; and $\mu_2 = -1$ implies a fully monotone relationship which is of negative or descending trend. *Weak* monotonicity is meant here. Ties in one variable may be untied in the other without penalty.
μ_2 can equal $+1$ or -1 even though the marginal distributions of the two variables differ from each other. Therefore a weak monotonicity coefficient is especially appropriate in condition where marginal distributions differ from item to item, as is the case in most research studies

The resulting matrix of a **MONCO** section can be inputted in the same job to **WSSA1** section, and/or written on a raw matrix file for future process.

DISCO

When two or more populations have distributions on the same numerical variable *x*, it is of interest to know to what extent these distributions overlap. One motivation for this interest is the problem of discriminant analysis.

Guttman presented two discrimination coefficients for this purpose without making any assumptions as to how their ditributions may differ in other respects. Both coefficients express the loss due to overlap as a direct function of the variance between the arithmetic means of the distributions. One coefficient is called *Disco* for "discrimination coefficient". The other is called *Odisco*; it is more relaxed than *disco* in a certain sense of overlap. The "*O*" at the beginning of *odisco* is meant to indicate that some overlap is allowed. Each is distribution-free, avoiding unrealistic assumptions of normality of population distributions and equality of variances within the populations.

Both coefficients vary between 0 and 1. They equal 0 if there are no differences among the means. Each equals 1 if perfect discrimination holds in its sense. *Disco* equals 1 when there is no overlap between the distributions. *Odisco* equals 1 when there is overlap only between the means of the respective distributions. Such efficacy coefficients are always appropriate since consistent estimation is ensured.

The **DISCO** section computes discrimination coefficients for comparison of arithmetic means. 4 coefficients are given: *Odisco*, *Disco*, *Eta* and *F* (*F* is calculated only for traditional interest in it).

WSSA1

This is a practical technique for aiding comprehension of the structure of interrelationships among variables. The analysis may be regarded as being essentially geometric. SSA treats each variable as a point in a Euclidean space in such a way that the higher the correlation between two variables, the closer they are in the space. The space of smallest dimensionality is used that makes possible such an inverse relationship between the observed correlations and the geometric distances. The empirical data to be anlysed are not limited to coefficients of similarity. They can be also dissimilarity coefficients (distances). In such a case, the monotonicity condition becomes: the smaller dissimilarity coefficient between variables, the closer their points are in the space

Formally, given a matrix $\{R_{ij}\}$ containing pairwise similarity coefficients (correlations), among a set of n variables, $V_1, V_2, \ldots, V_n$, the **WSSA1** section enables study of the matrix in a simple yet comprehensive manner.

The principal output is a space diagram plot representing each V_i as a point in the dimensionality chosen by the user, but sought to be as small as possible. The points are located in the space in such a way that they satisfy the monotonicity condition as well as possible, that is $d_{ij} < d_{kl}$ whenever the observed data indicate that $R_{ij} > R_{kl}$, d_{ij} being the Euclidean distance between two points.

There is an option to superimpose facet elements for each variable, and produce facet diagrams. This option facilitates viewing regional correspondence between the empirical distribution of the variables and their faceted definition.

Subgroups of the population can be located in the fixed space diagram of the original variables via external variable feature.

The empirical data to be analysed are not limited to coefficients of similarity. They can be also dissimilarity coefficients (distances), say $\{D_{ij}\}$. In such a case, the monotonicity condition becomes: $D_{ij} < D_{kl} \Leftrightarrow d_{ij} < d_{kl}$.

POSAC

While SSA deals with space of variables, **POSAC** deals with space of subjects.

A partial order analysis begins with some *n* different criteria (items) for the stratification of the population, each item being ordered in a sense common to all items.

Each member of the population has an observed profile composed of *n* structs, one struct from each item. The overall partial-order is an automatic consequence of the simple orders on each item separately.

By definition, a profile is higher than another, if and only if it is higher on at least one item and not lower on any other item. Similar definition holds for "lower" relation. A profile is equal to another if both are equal on all items. When the relation between two profiles is "higher", "lower" or "equal", these profiles are said to be *comparable*. Two profiles are *incomparable* if and only if one profile is the higher on at least one item while the other profile is also the higher on at least one item.

The complete **POSAC** output provides a two-dimensional representation of the partial order of the set of profiles.

POSAC calculates a mathematically optimal pair of base axes (*x* and *y*) for the empirical partial order. These base coordinates usually have a substantive meaning for the partial order, though there need not always be items that correspond to the directions of these base coordinates.

Each profile appears as a point in the two-dimensional space. Any two profiles that are comparable will have their two points on a line with a positive slope, namely the joint direction (*x*+*y*). Two incomparable profiles have their points on a line with a negative slope or the lateral direction (*x*-*y*). All four directions in the two-dimensional space (*x*, *y*, *joint* and *lateral*) have a role in interpreting the results.

Detailed analysis of the systematic differences among the items is made in terms of *n* **POSAC** diagrams, one for each item.

The **POSAC** section provides a two-dimensional representation of a partially ordered set **P** of *N* profiles, based on *n* variables (to be referred to as "the internal variables").

POSAC tries to reduce the number of variables from *n* to 2. More precisely, to each profile $p = (p_1, p_2, \ldots, p_n)$ in **P** will correspond a profile (x_p, y_p) in $\mathbf{R}^2$ such that, through this application the partial order in **P** is preserved as well as possible. x_p and y_p are called base coordinates.

Furthermore, for specified categories of a given external variable (a variable not in the original set of the *n* variables) representing an external criterion or trait, trait-diagrams are presented depicting the proportion of subjects possessing that trait among all those sharing the same profile in *n* variables. The trait may be specified also as a combination (intersection) of response categories from different external variables.

MPOSAC

POSAC was technically limited to two dimensions, until recently when a new algorithm allowed for processing **POSAC** in any dimensionality. The related computer module is termed **MPOSAC**, namely Multidimensional **POSAC**.

PEARSON

The **PEARSON** section computes Pearson coefficients of correlation. The syntax and output results are similar to those of **MONCO** section

INTRACLASS

In some cases involving pairs of measurements there is no way to distinguish between the pairs. For example, given IQ measurements of a pair of identical twins, how can it be decided which measurement is x and which is y? In such cases a different type of correlation coefficient, called the intraclass correlation coefficient which treats the pairs of measurements symmetrically, is needed to asses the relation between them. The **INTRACLASS** section computes such coefficients.

APPENDIX A

A complete and detailed mathematical description of the methods and coefficients used in HUDAP is given at the end of the manual.

APPENDIX B

This Appendix describes the Windows version of Hudap for two analyses: WSSA1 and Disco.

In WSSA1 example, a tool for detecting regionality (Axial, Modular or Polar) is demonstrated.

Overview

We present here a summary of the salient capabilities of HUDAP, together with examples. In subsequent chapters these features, and the manner in which the HUDAP system executes them, are discussed in greater detail. For the moment, our purpose is to give the user an overview of how the system operates, and to inform him of what he can and cannot accomplish with it.

Sequencing Calculations:

HUDAP is driven through its various functions by a sequence of instructions that the user must prepare. The instructions are written in sentences that are grouped into sections. HUDAP first reads all the instructions until the end of the user command file, and next it executes the instructions in sequential order, section by section, and sentence by sentence. The process is illustrated by a chart in figure 1.

There is a control program in HUDAP whose sole function is to decode the section names and to pass control to the appropriate subroutine which decodes the sentences of this section. The function called upon by this section is then performed, and HUDAP passes control back to the control program which then processes another section, etc. This calculation sequence is carried out automatically by HUDAP, and the details of how the control program and subprograms operate need be of no concern to the user. The important point for the user to realise is that HUDAP processes instructions in sequential order. It is up to the user to arrange the instructions so that the system will perform actions in the intended order.

Nevertheless, there is an exception: Sections comprised between **`FOR`** and **`ENDFOR`** sections are performed a number of times as requested by the user, in a loop manner.

HUDAP instructions are written in a quasi natural language. But, in order to use HUDAP, it is necessary that the user learn this language. This is not as formidable a task as it may sound, since the same sentences as '**`NAMES = _____`**' exist in different sections, and all sentences have a similar format with a minimum of rules. The user is free to choose names and labels that are natural to the problem at hand.

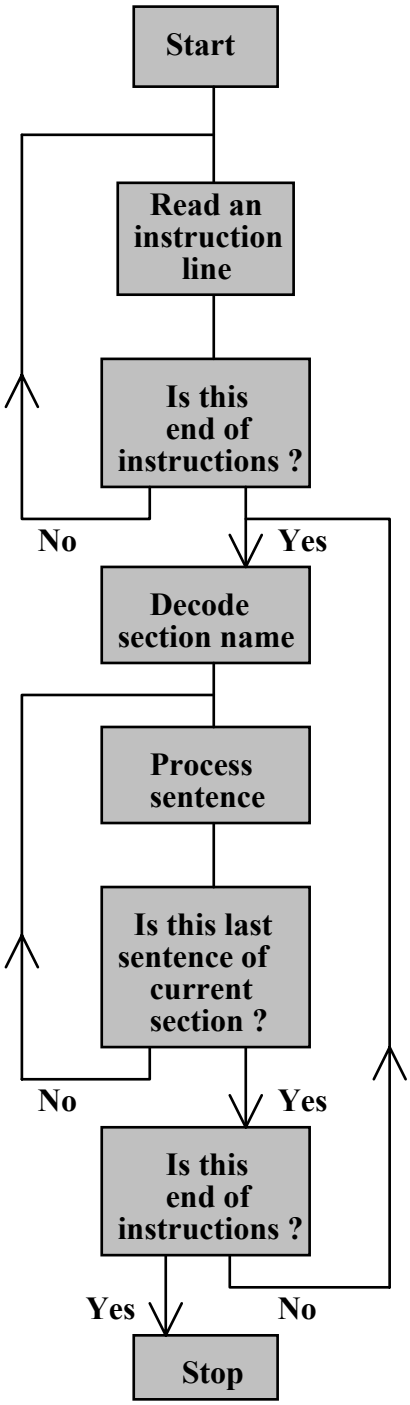


Figure 1
Program sequencing in HUDAP

Entering and processing data:

Data may be entered into HUDAP in a variety of ways. The most common way is by creating a file of data using an editor. The name of this data file must be supplied to HUDAP using the control instructions.

Some of the HUDAP control instructions define and describe the data while other types cause specific calculations to be executed. Instructions describing the data are grouped into one section called **DATA**.

We will demonstrate HUDAP using a study made in 1974 by the Israel Institute of Applied Social Research, the object of which was to explore the values of school-enrolled youths in Israel in various areas of life.

The youths were asked the following questions about "negative" commandments:

1.- To what extent is the Israeli society harmed by "receiving bribes from a public institution"?
2.- To what extent is the Israeli society harmed by "damaging school property"?
3.- To what extent is the Israeli society harmed by the "use of drugs"?
4.- To what extent is the Israeli society harmed by "not listening to a policeman"?
5.- To what extent is the Israeli society harmed by "emigrating from Israel"?
6.- To what extent is the Israeli society harmed by "copying in exams"?
7.- To what extent is the Israeli society harmed by "stealing from the rich"?
8.- To what extent is the Israeli society harmed by "supporting a Palestinian state in Israel"?
9.- To what extent is the Israeli society harmed by "stealing exams"?
10.- To what extent is the Israeli society harmed by "conversion out of the Jewish religion"?

For every question asked, the pupil had to choose between one of the following answers:

- extremely harmful.
- very harmful.
- slightly harmful.
- not harmful at all.

the design of the above questions was made in accordance with the following sentence, called a "mapping sentence":

The extent to which pupil $\{x\}^{X}$ sees an action as $\left\{\begin{matrix} a1 & \text{against} \\ a2 & \text{not against} \end{matrix}\right\}^{A}$

the law in the matter of the $\left\{\begin{matrix} b1 & \text{primary} \\ b2 & \text{secondary} \end{matrix}\right\}^{B}$ environment of

Jewish Israelis $\left\{\begin{matrix} c1 & \text{inside} \\ c2 & \text{outside} \end{matrix}\right\}^{C}$ the school as $\left\{\begin{matrix} \text{from} & \text{extremely harmful} \\ \text{to} & \text{not harmful at all} \end{matrix}\right\}^{R}$

with the aim of maintaining the values of the Israeli society.

The labels **X**, **A**, **B**, **C** indicate "facets". **R** indicates the common response range to the questions.

The above mapping sentence defines 8 types of possible items. The number of possible types is the multiplication of the number of elements in the various facets: 2*2*2 = 8. For each type it is possible to ask many questions. In fact, only a sample of 10 questions were asked. Every question contains one component of each of the three facets.

The data gathered was as follows:

```
   1  3212331411
   2  2211121311
   3  2212333222
   4  2113231230
   5  1111144141
   6  3212420313
   7  4323232423
   8  2323242321
   9  1212321112
  10  3312233032
 ..............
 ..............
 ..............
4607  3333333333
4608  3211132231
4609  3333221423
4610  3211231223
4611  4241142443
4612  3112441341
```

The data was prepared by means of a program of editing (using the editor) and was stored in a data file called **VALUES.DAT** . It was written on 4612 lines where each line contains the answers of each pupil. The data was prepared with fixed format so that every answer to each question could be found at the same place in every line. The number of the pupil was placed in columns 1-4, the first

question in column 7, the second in column 8, the third in column 9 etc... and the tenth in column 16.

First step

Let us assume that we wish to extract the following information from the above data:

1 The percentage of those who answered "extremely harmful" to each of the ten questions.
2 The percentage of those who answered "very harmful" or "extremely harmful" to each of the ten questions.
and so on.

We can extract this information by preparing a "command file" for HUDAP, that is, we shall create a file composed of instructions (in HUDAP's control language) that directs HUDAP to carry out the analysis we want. We must use the editor to write this file, of course. Let's assume that this command file is to be stored under the name **VALUES.HUD** (as opposed to the data file which was stored under the name **VALUES.DAT**).

The file **VALUES.HUD** is as follows:

```
$SET  NAME='Values of youth in Israel - 1974';
$DATA NAMES = ID 1-4 HARM1 TO HARM10 7-16 ;
      FILE = 'VALUES.DAT'  ;
      MISSINGS = HARM1 TO HARM10 0;
      VARLABS=   HARM1 'Receiving bribes'
                 HARM2 'Damaging school property'
                 HARM3 'Use of drugs'
                 HARM4 'Not listening to police'
                 HARM5 'Emigrating from Israel'
                 HARM6 'copying in exams'
                 HARM7 'Stealing from the rich'
                 HARM8 'Supporting Palest. state'
                 HARM9 'Stealing exams'
                 HARM10 'Conversion'  ;
      CATLABS= HARM1 TO HARM10
                      1 'extremely harmful'
                      2 'very harmful'
                      3 'slightly harmful'
                      4 'not harmful at all';
$FREQ NAMES = HARM1 TO HARM10;
```

The above file contains 3 sections called: **SET**, **DATA**, **FREQ**. The character $ separates each section.

1 The section **SET** gives a title to the work. The title is:
"**Values of youth in Israel - 1974**"

2 The section **DATA** describes the data in the terminology of the HUDAP software. This example uses the 5 following sentences: **NAMES**, **FILE**, **MISSINGS, VARLABS, CATLABS**.

The sentence **NAMES** provides names and locations for the variables. In this example, the first variable, **ID**, indicates the numbers used to identify the students (the number in the first four columns of each line in the data file **VALUES.DAT**.) The variable **HARM1** indicates the set of answers which were given to the first question ("To what extent is the Israeli society harmed by the use of drugs?") and so on for **HARM2** through **HARM10**.

The next sentence in the **DATA** section, **FILE**, indicates the name of the file containing the data ("**VALUES.DAT**" in this case).

The third statement: **MISSINGS** , specifies that **0** is a missing value for the 10 variables **HARM1 to HARM10**.

The fourth statement: **VARLABS** , specifies, between quotation marks, a label for each of the 10 variables **HARM1 to HARM10**.

The fifth statement: **CATLABS** , specifies, also between quotation marks, a label for each possible category of the above variables.

3 The section **FREQ** requests HUDAP to yield frequency table for each of the 10 variables **HARM1** through **HARM10**.

Now, when HUDAP is instructed to execute this control file, frequency tables, such as the one below (for the variable **HARM1**) will be created.

```
  Frequency table for HARM1   : Receiving bribes
  ---------------------------
Category                  Category              Cum
  Name                     Value   Freq   Pct   Pct

extremely harmful              1    671  15.13  15.13
very harmful                   2   1404  31.65  46.78
slightly harmful               3   1673  37.71  84.49
not harmful at all             4    688  15.51 100.00
                                   ---- ------ ------
  Sum                              4436 100.00 100.00


    Statistics for HARM1
    ----------------------
N of valid cases= 4436     N of missing cases=  176
```

The preceding table contains 5 columns : **Category Name**, **Category Value**, frequencies (**Freq**), percentages (**Pct**), cumulative percentages (**Cum Pct**).

In the first column called "**Category Name**", the label of each category appears. This label was supplied in the input file **VALUES.HUD** .

In the second column called "**Category Value**", the different values of the variable appear. In our case, the variable **HARM1** accepts 4 different values: **1**, **2**, **3**, **4**. Their meaning appears in the label printed at the left corner of each category.

In the third column called "**Freq**", the number of pupils who chose the corresponding value appears in each line. **671** pupils answered **1** to the first question, **1404** answered **2**, **1673** answered **3**, and **688** answered **4**. At the end of this column, the total number of pupils who answered this question, appears, i.e. **4436**.

In the fourth column called "**Pct**" the percentages of pupils who chose the corresponding value (**15.13** for value **2**, **37.71** for value **4**) appear.

In the fifth column the cumulative percentages appear. For example, **46.78%** answered that the harm is extremely severe (value **1**) or severe (value **2**), etc...

In the following table, we will summarise the ten tables produced by the section **FREQ**. There is a list of actions in order of the percentage of those who answered "extremely harmful" or "very harmful" to each one:

To what extent is the Israeli society harmed by each of the following	Percentage of those who answered "extremely harmful" or "very harmful"
Use of drugs	89%
Conversion	81%
Not listening to a policeman	79%
Stealing from the rich	73%
Stealing exams	69%
Damaging school property	69%
Emigrating from Israel	65%
Supporting Palestinian state	50%
Receiving bribes	47%
Copying in exams	34%

The data in the preceding table indicate that the majority of the actions presented, are considered by the majority of the respondents as extremely harmful, or as very harmful to Israeli society, i.e. it is very important not to do the majority of these actions.

Actions considered to be the most harmful to the Israeli society, are those related to a person's primitive interior environment, which is expressed here by his self

identification. These actions are: use of drugs and conversion out of the Jewish religion. (The majority of pupils - 70 respectively - responded that these actions are extremely harmful. Concerning the other actions, the percentage of respondents considering the harm to be very severe, did not exceed 38 use of drugs, there is a kind of fleeing from oneself and from society, and it is the same concerning conversion out of the Jewish religion, which is an action of changing one's self identification and abandoning the Jewish society. The severity by which the conversion is considered, indicates that the Israeli society is looked upon as Jewish, and that there is no conception which discriminates between nation and religion in Judaism.

The actions considered the least harmful, are related to "frauds" (copying in exams, receiving bribes from a public institution). This means that there is an inclination not to judge actions which are not completely straight with extreme severity when they are connected with schools or any other public institution.

The other actions are located at intermediate levels of "it is important not to do", between these two poles.

The response "not harmful at all" doesn't mean that it is important to do the action. Not to agree that something is negative, does not mean to agree that it is positive. The same follows concerning actions which contradict the law.

Further discussion of the task **FREQ** may be found in the chapter ***FREQ***.

Second step

The next step in our data analysis, is to discover the reciprocal relationship between the different actions. Is one who thinks that a particular action is harmful inclined also to think that another action is harmful too? Is there a strong relationship between action estimators, or is there no relationship between them at all?

In order to answer this question, let's request from HUDAP to compute a table of correlation coefficients between the ten actions. The best correlation between our ordinal variables must be a correlation which takes into account mainly the rank of the values of each variable. **MONCO** coefficient is a Guttman coefficient which measures the monotonicity between two variables, that is, how much the two variables vary in the same sense.

Following is a HUDAP job (control file) which requests computation of a **MONCO** matrix:

```
$SET  NAME='Values of youngs in Israel - 1974';
      LINESIZE = 80 ;
$DATA NAMES = ID 1-4 HARM1 TO HARM10 7-16 ;
      FILE = 'VALUES.DAT' ;
      MISSINGS = HARM1 TO HARM10 0 ;
      VARLABS= HARM1 'Receiving bribes'
               HARM2 'Damaging school property'
               HARM3 'Use of drogues'
               HARM4 'Not listening to police'
               HARM5 'Emigrating from Israel'
               HARM6 'copying in exams'
               HARM7 'Stealing from the rich'
               HARM8 'Supporting Palest. state'
               HARM9 'Stealing exams'
               HARM10 'Conversion' ;
      CATLABS= HARM1 TO HARM10
                    1 'extremely harmful'
                    2 'very harmful'
                    3 'slightly harmful'
                    4 'not harmful at all';
$MONCO NAMES = HARM1 TO HARM10 ;
```

Sections **SET** and **DATA** were explained above. The section **MONCO** instructs HUDAP to compute a table of monotonicity coefficients between the ten variables comprised between **HARM1** and **HARM10**. As in the section **FREQ** previously supplied, the section **MONCO** uses the sentence **NAMES** to specify the variables on which the analysis must be performed.

The output of the above job appears in figure 2.

Note that the variable **HARM1** ("**receiving bribes**") has very small coefficients. This means that **HARM1** is "far away" from the other variables. **HARM1** cannot help to predict the other variables. Therefore, **HARM1** will be excluded from future processes.

All the coefficients between the 9 variables **HARM2** to **HARM10** are positive and extend from **0.16** to **0.75**, e.g. in general, one pupil who is more inclined than another pupil, to judge a specified action as harmful, is also inclined to judge all other actions with more severity than the other pupil. It is interesting that this result was obtained for all the 9 actions, those that are against the law and those that are not. For example, "use of drugs" which is against the law, is related to actions such as supporting "Palestinian state in the Land of Israel" (coefficient = 0.38), and as "Emigrating from Israel" (coefficient = 0.41), which are not against the law, but relate to specific opinions. In other words, one who is not inclined to consider emigration and supporting a Palestinian state in the Land of Israel, as harmful to the Israeli society, is also inclined to consider actions which are against the law with less severity. There is a phenomenon of association between values: "one sin causes another".

Further discussion of the task **MONCO** may be found in the chapter ***MONCO***.

Third step

The higher correlation in our example, is 0.75 between "copying in exams" and "stealing exams", but the lower correlation is 0.16 between "Damaging school property" and "Emigrating from Israel".

In order to facilitate the observation of the table of monotonicity coefficients and to find out which kind of actions are near to each other, and which kinds are far away, let's describe the table graphically. According to the method of the Smallest Space Analysis (SSA), -- for which the HUDAP control language (**WSSA1** section) will be explained later --, we will find that it is possible to describe the table of coefficients by the two-dimensional space of figure 3.

```
   Matrix of weak monotonicity coefficients (Decimal point omitted)

HARM

HARM2      2 I     28    100     65     59     16     57     47
             I  (4405) (4475) (4454) (4449) (4431) (4443) (4397)
             I
HARM3      3 I      2     65    100     70     41     44     56

HARM

HARM

HARM
             I  (4407) (4443) (4457) (4460) (4441) (4481) (4409)

HARM7      7 I      4     47     56     51     20     56    100
             I  (4360) (4397) (4408) (4412) (4391) (4409) (4432)

HARM

HARM9      9 I      5     54     56     52     20     75     66
             I  (4395) (4432) (4444) (4447) (4427) (4444) (4394)
             I
HARM

HARM

HARM9      9 I     30    100     43
             I  (4346) (4471) (4455)
             I
HARM10    10 I     47     43    100
             I  (4358) (4455) (4489)
```

Figure 2

Each of the 9 actions is represented in the graph by its number enclosed in parentheses. We can consider that number as a point. Two points are relatively near one another in the graph, if the correlation between the two actions is high. For example, two close points in figure 3 are "copying in exams" and "stealing exams", in accordance with the high correlation 0.75 between them. On the other hand, the point representing "Damaging school property" is far away from the point representing "Emigrating from Israel", in accordance with the low correlation between them (0.16).

The graph shows clearly the different directions of behaviour in a manner which takes account of all mutual connections. We can divide the graph into two zones, where each zone relates to the legality of each action (facet A in the mapping sentence discussed above). The zones are separated by the vertical line. All the actions which are against the law are located in the right-hand zone: use of drugs, not listening to a policeman, damaging school property, copying in exams, stealing exams and stealing from the rich. Actions which are not against the law are located in the left-hand zone: conversion out of Jewish religion, emigrating from Israel, and supporting Palestinian state in the Land of Israel.

These two zones can be divided into sub-zones which branch out from the origin, according to the place where the action is attached (facet C): inside school or outside school. This partition is represented in the graph by the lines of "+". Points appearing in the area between these two lines refer to actions that take place inside the school.

The partition of the graph into regions which branch out from the origin, creates a circular order of variables, which can appear directly from the monotonicity coefficients. The interpretation of the circularity is as follows: If two variables are found at equal distance from the origin, but in different zones, then the closer the regions in which they are found, the higher the correlation between them. In the table of coefficients, this is expressed by the fact that the high coefficients tend to be found in the neighbourhood of the principal diagonal, decrease gradually as we withdraw from the diagonal, then increase gradually again. Figure 4 expresses the circular order.

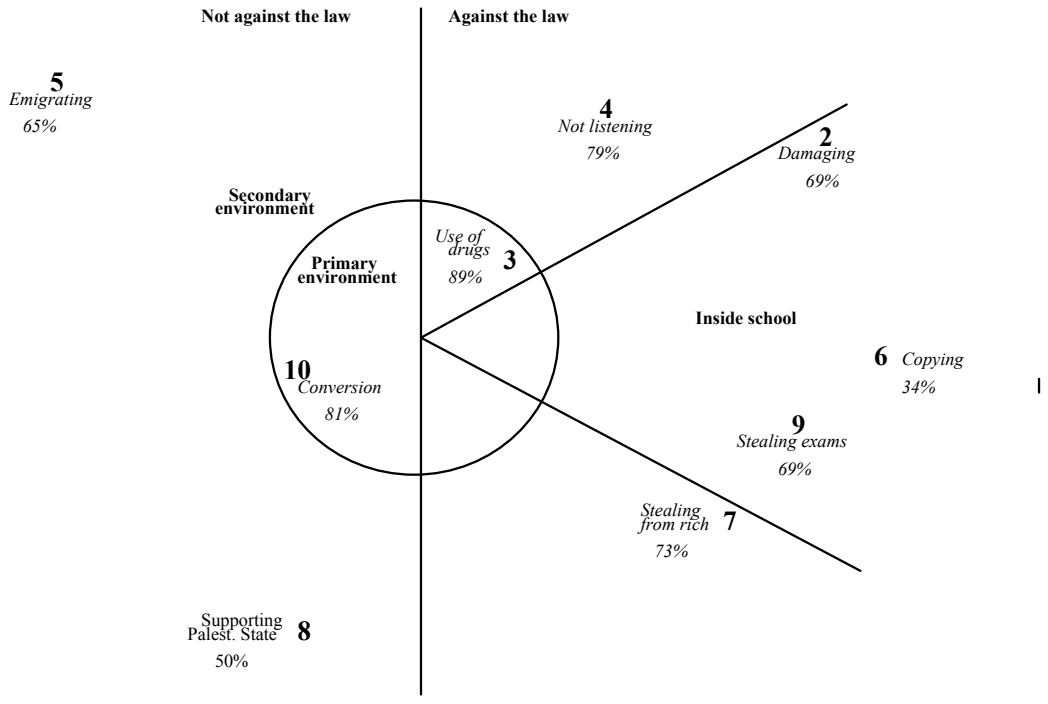


Figure 3
Graphic description of relations between 9 harmful actions

	4	2	9	7	8	5	
4	--	59	52	51	26	40	Not listening to a policeman
2	59	--	54	47	26	16	Damaging school property
9	52	54	--	66	30	20	Stealing exams
7	51	47	66	--	32	20	Stealing from the rich
8	26	26	30	32	--	28	Supporting Palest. state
5	40	16	20	20	28	--	Emigrating from Israel

Figure 4
Showing circular order of monotonicity coefficients

There is a circular order of zones, where the directions around the circle correspond to the kind of action and to the place to which the action is related as explained in the following. Let's begin with the upper part of the circle, and go clockwise:

1 Behaviour against the law, outside school: Not listening to a policeman (variable 4).
2 Behaviour against the law, inside school: Damaging school property, stealing exams (variables 2, 9).
3 Behaviour against the law, outside school: Stealing from the rich (variable 7).
4 Behaviour not against the law, outside school: Supporting Palestinian state, emigrating from Israel (variables 8, 5).
5 Behaviour against the law, outside school: (variable 4).

Therefore, facets **A** and **B** of the mapping sentence (kind of action and place to which it is attached) serve as polarising facets. Knowing the regions of the variables around the circle is not sufficient to restore the correlation coefficients between them. It is necessary to know also the distance from the origin. We can then ask: Can we also classify the content of the variables according to their distance from the origin? It appears that facet **B** - the environment frame - gives such a classification.

The origin zone (surrounded by a circle in figure 3) includes the variables "use of drugs" and "conversion out of the Jewish religion". As noted, these two actions relate to the primary environment of people, in the sense that they relate to their identification. They are both attached to the trial to change the self identification. The remainder of the actions relate to their secondary environment (school, the society in general).

Technically, the centrality, has a double meaning:

(1) the variables in the origin zone will be close to one another, even though their direction in the map is different, because they are in a circle with a small diameter. In fact, the correlation between these two variables is large enough - 0.60, even though their directions are different: one is against the law (use of drugs), and the other is not.

(2) for each other variable, there is at least one variable which is further away from it than the two central variables are. Indeed, the two central variables have a correlation of at least 0.38 with any other variable, whereas the correlation coefficients between the peripheral variables get as small as 0.16.

The partition of the graph into two circles - interior and exterior - according to the environmental frame - can illustrate aspects of the student's process of evaluating how harmful an action is. The percentage of those who consider the actions related to a person's self-identification (the interior circle) as extremely harmful or severely harmful to the Israeli society, is greater than the percentage of those who consider the remainder of the actions related to a person's secondary environment as extremely, or severely harmful. We have therefore, an interesting and important phenomenon of strong empirical association between the position of the actions in the graph and the percentage of those which think that it is important not to do them. This phenomenon exists, even though the graph does not result from the above percentages but from correlations between variables.

The graph of Figure 3 is produced by the section **WSSA1** of the HUDAP package. **WSSA1** accepts as input a matrix of proximity or distance coefficients between the variables. In the following job **WSSA1** uses the monotonicity coefficient matrix produced by the HUDAP section **MONCO** (That matrix appears in Figure 2):

```
$SET  NAME='Values of youth in Israel - 1974';
$DATA NAMES = ID 1-4 HARM1 TO HARM10 7-16 ;
      FILE = 'VALUES.DAT' ;
      MISSINGS = HARM1 TO HARM10 0 ;
      VARLABS= HARM1 'Receiving bribes'
               HARM2 'Damaging school property'
               HARM3 'Use of drogues'
               HARM4 'Not listening to police'
               HARM5 'Emigrating from Israel'
               HARM6 'copying in exams'
               HARM7 'Stealing from the rich'
               HARM8 'Supporting Palest. state'
               HARM9 'Stealing exams'
               HARM10 'Conversion' ;
      CATLABS= HARM1 TO HARM10
                   1 'extremely harmful'
                   2 'very harmful'
                   3 'slightly harmful'
                   4 'not harmful at all';
$MONCO NAMES = HARM1 TO HARM10 ;
       OUTPUT MEMORY ;
$WSSA1 NAMES = HARM2 TO HARM10 ;
       MAXDIM = 2 ;
```

The command **MEMORY** of paragraph **OUTPUT** in the above section **MONCO**, requests insertion of the monotonicity matrix into memory, in order to be used by **WSSA1**. **WSSA1** has to describe the table of monotonicity coefficients by a two-dimensional space as requested by the sentence **MAXDIM=2**. A more detailed explanation of **WSSA1** may be found in the chapter ***WSSA1***.

Control Language

Instructions are given in hierarchical form. One elementary instruction is called 'sentence'. Sentences are grouped in paragraphs, and paragraphs are grouped in sections. For example all the instructions defining data are grouped into the section **DATA**. Different sections can have the same paragraph names or/and the same sentence names. That homogeneity makes easy learning of the HUDAP language.

Data

Data files consist of values recorded in 'variables' on a set of 'cases'. The input data file for HUDAP must be a rectangular case-ordered file. That means that for every case, a value is recorded for each variable - always in the same order. A single case can be entered on one or more records. If the cases require more than one record, each case must contain the same number of records, which must be in the same order within each case. Data format can be fixed or free, as described in the ***DATA*** chapter.

HUDAP brings all data from the data file into the memory. This is required by the non-metric programs to which belongs Guttman methods, as such as for clustering methods. This fact exists also in SPSS package in the computation of SPEARMAN or KENDALL correlation by the NONPAR CORR procedure. For big data, it is recommended to work on a computer having virtual memory system.

Nevertheless, there are many advantages in bringing data into memory. Data are read only once, not by each procedure as in other packages. In addition, HUDAP is able to supply some functions as variable mean, variable minimum, variable standard deviation etc... -functions necessitating presence of data in memory. It is also possible to use results of one procedure as input for an ulterior procedure, by requesting HUDAP to bring results in memory, at the place where the following procedure expects to find its input. For example, a **MONCO** coefficients matrix can be supplied into memory to the procedure **WSSA1** (see chapter on **WSSA1** section).

Missing values

The **MISSINGS** sentence within the section **DATA**, declares the missing values for certain variables in your file. The values defined as missing values are never changed on the data; they are simply flagged in the dictionary of the data area and, if the data area is saved as a system file, in the dictionary of the system file. The HUDAP procedures and transformation sections recognise this flag, and those cases that contain a user-defined missing value are handled differently.

User-missing values defined on the **MISSINGS** sentence are distinguished from the system-missing value. HUDAP assigns the system-missing value when a blank field in data file is read for numerical variable or when a value resulting from a transformation command like **COMPUTE** is undefined.

Missing values are handled by HUDAP in such a manner that no information is lost. Statistical methods as regression analysis, factor analysis, discriminant analysis and others, require a case to be deleted, if any of the N variables has missing value in that case. This is called "listwise" deletion. Guttman methods do not need the deletion of that case.

Recoding data

Very often, you need to change values of your data, either to perform simple data-cleaning checks, to correct coding errors, to group some continuous variables prior to request their frequencies, or to rescale several variables prior to analysis. The **CODING** section provides these and many other possibilities. **CODING** section also allows conversion of multicolumn alphanumeric values to their numeric equivalents.

Variable transformations

Often, you want to create a new variable or transform an existing variable using information from other variables on your file. The **COMPUTE** section generates a variable on your data area that is constructed on a case-by-case basis as an arithmetic transformation (or logical expression) of existing variables and/or constants.

If a value is missing in any of the variables (for a certain case) used in an expression when the computation is executed, HUDAP nearly always returns the system-missing value since the operation is indeterminate.

While numeric expressions are commonly used with the **COMPUTE** command, they can be used as part of a logical expression in the **IF ... THEN ... ELSE ... ENDIF** feature.

The facilities for numeric expressions are:

1 Arithmetic Functions - enable you to truncate a variable, use the square root or the log of a variable, and so on.
2 Statistical Functions - enable you to compute statistics such as the mean or standard deviation of a variable.
3 Missing-Value Functions - used to control the number of missing values in a variable or to test whether a specific cell in the data matrix is missing or not.

Numeric expressions may also include arithmetic operations and numeric constants.

Scalar variables

Data are codes representing characteristics (e.g., sex, eye colour), values of measurements (e.g., height, weight) or responses to questions. Each characteristic, measurement or response is called a regular variable or simply a variable. This kind of variable, the more commonly used, is valued over all the cases. HUDAP allows the use of another kind of variable: the scalar variable, which is a single-valued variable.

A scalar variable can be used to store into it the value of a specific computation, the value of a specific cell in the rectangular data matrix, or special system scalar variables.

Loops

Often, a user needs to perform the same job sequence on several of the variables in the file or for various options. Instead of preparing a separate sequence for each variable or for each option to be processed, the user can reduce the amount of control-line preparation necessary by utilising the **FOR..ENDFOR** facility.

The **FOR..ENDFOR** facility has no limitations on the kind of the option to be repeated. For example, you can change data file from one step to the next in the loop. You can even change the procedure name or any element of the HUDAP language.

Selecting data

The user is able to select cases for any procedure. He has to use the **MODIFILE** section. Cases are permanently selected. To select cases in a temporary manner, you can use the facility **IF...THEN...ELSE** into the **MODIFILE** section supplying **FILE** sentence, and then give different **RESTORE** sections as you need (see ***MODIFILE*** chapter).

Retrieval data from the system

All data input into the HUDAP system, as well as recoded variables, new variables created by transformation, and file changes accomplished by selection of cases or of variables may be saved on a HUDAP system file (**MODIFILE** section) and restored later (by **RESTORE** section) for further processing.

The HUDAP system file can be processed only by HUDAP itself.

The user is also able to output the above information in a raw form, which can be dealed by any other program (see the **OUTPUT** section). The correlation procedure (**MONCO**) and the crossfrequency procedure (**CRST**) permit the user to output matrices of data which can be supplied in input to other HUDAP procedures (**WSSA1** for example). These matrices can be outputted on files, or directly brought in memory for later use.

Data Analysis Procedures in HUDAP

The **FREQ** procedure produces a table of frequency counts and percentages for values of individual variables. Variables can be of any type: integer, real or character. Optionally, one can obtain histograms for interval variables, univariate summary statistics and percentiles. One can prevent the printing of the frequency table, in the case of interval-level data, and request only statistics.

The **CRST** procedure produces tables containing joint distribution of two variables that have a limited number of distinct values. The frequency distribution of one variable is subdivided according to the values of the other variable. The combination of one value of the first variable with one value of the second value, defines a cell - the basic element of **CRST** tables. In addition to the tables of cell counts, you can obtain tables of cell percentages and optional measures of association (Monotonicity coefficient **MONCO** between the two variables, Pearson correlation coefficient, Chi-square which can be considered as a distance coefficient, Dependency coefficient which can be also considered as a

distance, with that difference that this is a normalised distance comprised between 0 and 1). A matrix of these coefficients can be produced on a file or in memory for later use. That matrix can be supplied, for example, to a **WSSA1** section as input matrix. **CRST** can handle integer, real and alphanumeric variables.

Users are often called upon to produce a variety of reports for business, schools, hospitals, and other organisations. These reports may contain tables of descriptive statistics like frequencies or joint distributions. The **MULTABS** procedure is a flexible formatter to get cross-tabulations between one variable (columns) and a series of variables (rows)

The **MONCO** procedure computes Guttman weak monotonicity coefficients for pairs of ordinal or interval variables. That is a kind of correlation. **MONCO** computes a reciprocal relationship between 2 different variables.

When a trait is observed for individual members of each of m populations, there are many features on which the m distributions can differ. The **DISCO** procedure is concerned with problems related to ascertaining how much the populations differ on the arithmetic mean on a numerical trait, without making any special assumptions as to how their distributions may differ in other respects. The m populations may be sub populations of an overall population, distinguished by a one-way, two-way, or any k-way classification scheme. Thus, our results hold in particular for the kinds of data traditionally treated by so-called analysis of variance of experimental designs. **DISCO** has to be used in place of the classical analysis of variance method (ANOVA).

The **WSSA1** procedure provides a graphic presentation of pairwise interrelationships among a set of n objects. The produced Space Diagram enables to find out which kind of variables are near to each other, and which kinds are far away. **WSSA1** is intended to replace the classical FACTOR analysis method.

The **POSAC** procedure provides a detailed study of the similarities and differences among structuples (profiles) by viewing them in the space of smallest dimensionality that can preserve the partial-order between the structuples. **POSAC** is intended to replace the classical DISCRIMINANT analysis method.

The **PEARSON** procedure computes Pearson correlation coefficients for pairs of interval variables. This section was introduced into HUDAP package in order to allow comparison between different researches, since Pearson coefficients are more utilized than any other coefficient of correlation. However, the researcher has to be aware that Pearson coefficient is a coefficient of **linear** correlation based on regression lines, while the concept of correlation does not necessarily depend on the concept of regression.

In some cases involving pairs of measurements there is no way to distinguish between the pairs. For example, given IQ measurements of a pair of identical twins, how can it be decided which measurement is *x* and which is *y*? In such cases a different type of correlation coefficient, called the intraclass correlation

coefficient which treats the pairs of measurements symmetrically, is needed to asses the relation between them. This is the purpose of **INTRACLASS** procedure.

Control Language

The data analysis which the user wishes to perform must be described in an English-based Control Language. Control language instructions are read and interpreted by HUDAP system, and are used to

1 name ,locate, and state missing value codes for the variables.
2 specify file pathnames for input or output.
3 specify the analysis such as Weighted Smallest Space Analysis.
4 repeat a sequence (loop) or perform a conditional task (if ...then).
5 specify some characteristics for the job, such as title, number of lines per page, the width of the page,...

The following example from **MONCO** -- Weak Monotonicity Coefficients -- demonstrates some Control Language instructions.

```
$SET NAME = 'This is an example';
$DATA  NAMES = VAR1 TO VAR10 5-14 / VAR11 TO VAR17 7-13 ;
       FILE = '\DATABANK\MONCO.DAT';
$MONCO NAMES = VAR1 TO VAR17 ;
```

The above HUDAP Control Language statements specify a title (This is an example) for the job; the names and location of variables in the data ; the data file pathname. Then in the **MONCO** section a matrix of monotonicity coefficients is requested for all the variables

Syntax and Punctuation

There are 3 levels in the Control Language instructions :

1 section, composed by paragraphs
2 paragraph, composed by sentences
3 sentence, which is a command or an assignment

In the above example **SET**, **DATA** and **MONCO** are sections. In the **DATA** section, **NAMES**, and **FILE** are paragraphs , each composed by a single sentence.

Sections are separated by a dollar sign (**$**), paragraphs by a semicolon (**;**), and sentences by a slash (**/**). Note that most of the paragraphs are composed by a single sentence. Therefore these sentences end with a semicolon and not a slash.

Values or names in a list are separated by blanks or comma.

Each word or value must be separated from the following word or value either by one or more blanks or by the appropriate punctuation (dollar sign, slash, semicolon or equal sign).

Control Language instructions can be typed continuously. An arbitrary number of blanks can be used between words, sentences, paragraphs or sections to make the Control Language easier to read.

Sections

The section name must be the first word in the section and begin with a dollar sign (**$**).

If a section is specified more than once, any repeated sentence overrides the former one, but a non repeated sentence is preserved.

Paragraphs

For some sections, it occurs that more than one sentence is necessary to define a common process. In this case these sentences are grouped into a paragraph. Within the paragraph, sentences are separated by a slash (**/**).

Sentences

Paragraphs are composed of sentences that are either commands (e.g., **RPCT ;**) or assignments (e.g., **MAXDIM=2;**). Sentences can be typed in any order within a section. Each sentence is terminated by a semicolon in most cases, since there is usually one sentence per paragraph.

An assignment sentence is used to assign a list of numbers, variable names, etc. to a HUDAP keyword. The general form of an assignment sentence is

HUDAP **keyword** { = / IS / ARE } value(s) assigned to the specific word.

The HUDAP words are described for each specific section. Examples are **NAMES**, **FILE** and **MISSINGS**. Note that in the chapters below, the definitions

of the HUDAP words are enclosed within boxes. Each assignment sentence must end with a semicolon (or a slash, when the paragraph contains more than one sentence). **IS** and **ARE** are special words that are interchangeable with the equal sign (**=**). They can be used only in this context unless they are enclosed in apostrophes. The values can be numbers, names, titles, etc., according to the definition of the particular word. In our definitions that assign values to HUDAP words (options) we use one of the following forms (examples follow):

HUDAP **keyword =**

- **<word1 | word2 | ... | wordn>** ; one specific word among the n choices.
- **<value>** ; one number.
  <val list> ; one or more numbers.
- **<variable>** ; one variable name not exceeding 8 characters.
  <var list> ; one or more variable names not exceeding 8 characters.
- **'<char>'** ; file name, title or label requiring apostrophes.

1 Examples of assignment sentences that require one specific word:

```
PROCESS = PARTIAL ;   (in POSAC section)
DATA = DISSIM ;       (in WSSA1 section)
```

These sentences are described as:

```
PROCESS = <COMPLETE | PARTIAL> ;
DATA = <PROXIM | DISSIM> ;
```

2 Examples of assignment sentences that require one or more numbers (but not names) are:

```
MAXDIM = 4 ;               (in WSSA1 section)
TOP = 7 ;                  (in MULTABS section)
MISSINGS = 0 9.99 -1 ;     (in MATRINP section)
```

In our section descriptions we write the definition of **MAXDIM** as:

```
MAXDIM = <value> ;
```

to indicate that only one value (a number) is permissible.
In the definition of **MISSINGS** we write:

```
MISSINGS = <val list> ;
```

to indicate that more than one number can be assigned.

3 Examples of assignment sentences that require one or more names are:

```
FREQ = Count ;          (in POSAC section)
NAMES = Id Age Height Weight ;(in various sections)
```

When only one name is possible as for **FREQ**, the definition is written as:

```
FREQ = <variable> ;
```

When one or more names are possible, we specify:

```
NAMES = <var list> ;
```

Each name is limited to a maximum of eight characters, and different names are separated by blanks or commas.

4 Examples of sentences that contain a file name, or title are:

```
FILE = 'PUPILS.DAT' ; (in DATA section)
TITLE = 'SSA on blood chemistry data' ; (in WSSA1)
```

In our section descriptions we write the definition of **FILE** as:

```
FILE = '<char>' ;
```

Default Values

Does the user need to assign values to all HUDAP words? No. Very few specifications are necessary to process a section. Many sections only require that you specify the names of the variables. If not specified, many sentences are given default values that are appropriate for most problems. The default value is the value used for an option when you do not specify its value. The definition of each HUDAP sentence includes a statement to describe the default value.

Reserved words and symbols :

IS, **ARE**, **BY** and **TO** are special words in the HUDAP instruction language explained below. Therefore they should not be used except in their special context. If they are used in any other way, they must be enclosed in apostrophes (**'**). The apostrophe (**'**) is a reserved symbol and should not be otherwise used.

Number notation :

Numbers can be either integers or real numbers (numbers with a decimal point). In addition, numbers can be in E-notation (scientific notation). That is, .000218 can also be written as 21.8E-5, or as .218E-3. And 218000 can be written as 218000.0 or as 2.18E5.

Blanks :

Blanks can be used freely to space sentences and lists, but cannot be used in the middle of numbers or names (unless the name is enclosed in apostrophes). We use blanks to space sentences and sections so they are easy to read.

```
$DATA NAMES = VAR1 TO VAR5 1-5 ;
      FILE = 'PUPILS.DAT' ;
      RECORDS = 2 ;
```

is easier to read than

```
$DATA NAMES=VAR1 TO VAR5 1-5;FILE='PUPILS.DAT';RECORDS=2;
```

but both are correct.

Manual format for definitions and summary of instructions

In this manual the following format is used to define a HUDAP section and sentences :

XXX section menu

Sentence or **Command;**
=============> (definition)
=============> (restriction if any)
=============> (defaults if any)

..... ;
.....
.....

YYY paragraph (if any)
=============> (general definition)

Sentence or **Command /**
=============> (definition)
=============> (restriction if any)
=============> (defaults if any)

..... /
.....
.....

For example, summary of **DISCO** section:

DISCO section menu

NAMES = <var list> ;

Names of the variables for which discriminant coefficients are computed. There are two kinds of Disco (Odisco) coefficients:

- If **GROUPS** sentence is submitted the coefficient is computed on each variable in "**var list**" throughout the grouping variable.
- If **GROUPS** sentence is not submitted the coefficient is computed on each pair of variables in "**var list**". The resultant matrix can be outputted to the memory, by mean of **OUTPUT** paragraph, in order to use it as an input of another section (for example **WSSA1**).

..... ;
.....
.....

OUTPUT paragraph

Paragraph defining the output of the **DISCO** Matrix (only when **GROUPS** sentence is omitted).

FILE = '<char>' /

"**char**" is the matrix file pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

..... /

.....

.....

Convenience Features for the HUDAP Instructions

The Keyword TO in naming variables

We will see later that the definition of the sentence **NAMES** in **DATA** section (without location specifications, for free format) is :

NAMES = <var list> ;

For example :

```
NAMES = Age Sex Income Occup Educat ;
```

But this variable-naming convention can become tedious if a large number of similar variables must be declared which do not lend themselves to useful, distinct mnemonics. If these variables are located on the raw-input-data file in such a way that they can be read in sequential order, an alternate method can be employed which requires much less instruction preparation. This convention identifies variables by means of a user-specified alphabetic prefix with associated variable sequence numbers. A large string of such variables can be named and defined by the following type of inclusive variable list :

```
ALPHAnnn TO ALPHAmmm
```

where **ALPHA** represents a user-specified alphabetic prefix, **nnn** and **mmm** represent integer numbers without leading zeroes, and **mmm** must be greater than **nnn**. The prefix **ALPHA** must begin and end with alphabetic characters, but can vary in length from one to seven characters as long as the prefix **ALPHA** and the associated number range do not exceed eight characters in length. The keyword **TO** is placed between the two variable sequence identifiers and must be separated from each by at least one blank. The **ALPHA** prefixes to the left and right of **TO** must be identical. Under these conditions, a string of variables will be implicitly

defined equal to the number of integers between **nnn** and **mmm**. Using this inclusive-naming convention, the variables **Item1**, **Item2**, **Item3**, ..., **Item100** would all be defined by the following **NAMES** specification :

```
NAMES = Item1 TO Item100;
```

The implied intervening variable names(in this case **Item2** to **Item99**) will be automatically generated by the system and associated with the proper variables in the same way as alphabetic mnemonics. It is clear that this notation results in a considerable economy when a large number of variables must be declared. However, some of the mnemonic quality of the variable names may be lost. The two naming conventions may be freely intermixed, producing a variable list for the file which contains some individually named variables and some variable names of the **ALPHAnnn** type. The variable sequence identifiers associated with a given prefix are arbitrary numbers and do not necessarily correspond to the actual sequential position of the variable in the file. For example :

```
NAMES = Nissim Ephraim Yaacov Tsioun1 TO Tsioun4
        Shemouel Miryam Reuven Rama5 TO Rama7
        Yehudite Ytshak;
```

Note that the sentence

```
Item001 TO Item010
```

is illegal because of the presence of leading zeroes.

The Keyword TO when referencing Adjacent Variables

When three or more adjacent variables are to be processed, the variable names may be specified by a notation of the following type:

```
Ephraim TO Yehudite
```

where **Ephraim** and **Yehudite** are variables previously defined, and **Ephraim** precedes **Yehudite** in **NAMES** specification of **DATA** section. The keyword **TO** must be separated from each identifier by at least one blank. This notation enables the user to process simultaneously large numbers of adjacent variables. Note that the effect of the keyword **TO** in an inclusive list differs from the effect of the keyword **TO** in an **ALPHAnnn TO ALPHAmmm** list previously explained. If the specification in sentence **NAMES** of **DATA** section is **... V2 V3 V1 V5 V6 V4 ...,** entering **V2 TO V4** in an inclusive list will reference the six variables **V2 V3 V1 V5 V6** and **V4.** Entering **V1 TO V6** in an inclusive list will reference only the three variables **V1 V5** and **V6.**

Comments in the HUDAP language

HUDAP Language accepts user comments anywhere in the job. In order to clarify the HUDAP input file, the user can insert comment blocks. A comment block is a string inserted between brackets. For example :

```
{ This is a comment }
```

This feature can be also used to suppress the execution of a part of a job, without deleting it. For example :

```
$DATA NAMES = ID 1-4 (A) Item1 TO Item20 5-24 ;
    { MISSINGS = Item1 Item2  0 TO 4  ; }
      FILE = 'MONCO.DAT'  ;
$MONCO NAMES = Item1 TO Item6 ;
```

Nested comment blocks are permitted without any restriction on the number of the blocks. This is useful when one wants to suppress a part of a job which is already commented. For example :

```
$DATA  { Naming-locating the variables and
         Specifying datafile name  }
        NAMES = Id1 Id2 1-8 (A) Murder_m Rape Assault
               Robbery Burglary Larceny Auto_The 9-15 ;
        FILE = '\DATABANK\CRIME.DAT' ;
{ $MONCO  { Processing MONCO on POSAC variables }
        NAMES =  Murder_m  TO  Auto_The ;  }
$POSAC  { Processing POSAC }
       NAMES = Murder_m  TO  Auto_The ;
       LABEL = Id1 Id2 ;
```

In this example, the execution of **MONCO** is suppressed. This "commented" part contains 2 nested comment blocks.

The *DATA* Section

The **DATA** section is required for any problem (analysis) or data transformation. This section allows the user to name his variables, specify how they are organised (their format), and where the data are located (pathname of the file). In this section missing value codes as well as variable and category labels can be specified.

The **DATA** section must at least contain the sentences **NAMES** and **FILE**.

However, the user can generate his own data by the **COMPUTE** section. In such a case the section **DATA** is not needed at all.

The sentence NAMES : naming and locating the variables

Variables often have short names that are easy to remember such as **SEX**, **AGE**, **HEIGHT**, **WEIGHT**, **ID**, etc. Your results can be interpreted more easily if these names are printed in the output. For example

```
NAMES  =  ID  AGE HEIGHT WEIGHT INCOME ;
```

The first name is that of the first variable, the second that of the second variable, etc. Variable names are restricted to eight characters. They must begin with a letter.

Names do not need to be specified for all of the variables. The keyword **TO** can be used to shorten lists (see above in HUDAP features).

Specifying location and data format

The example given above is for a free formatted data. When the data are fixed formatted on the data file you must specify for each variable it's location within the case. The case or observation can contain more than one record. By record, we mean a line in the data file.

Variable location

A user variable is located by two specifications :

_the record number

_it's position within the record

Two symbols can control the record number:

#n moves the pointer to record number "n".
/ moves the pointer to next record.

If there are more than one record to skip, you can use either

m**(/)** or **//**.../ (m times)

which means skip "m" records.

A variable within a record is located by specifying the beginning column, a dash, and the ending column as in:

```
INCOME 16-21
```

If the width of the variable is one, the beginning column is sufficient. Example:

```
SEX 5
```

If several variables are recorded in adjacent columns on the same record and have the same width and format type, HUDAP equally divides among the variables the total number of columns specified as in:

```
TEST10 TO TEST20 10-20
```

If the number of columns does not divide equally, an error message is issued.

Variable format type

HUDAP can read 3 types of variables:

_Integers
_Reals with decimal point.
_Alphanumerics

If the variable is integer, you just specify column locations after variable name as in:

```
INCOME 16-21
```

If the variable is real with "d" decimal digits, you specify **(**d**)** after the column location as in:

```
HEIGHT 7-10 (2)
```

If the variable is alphanumeric, you add **(A)** after the column location as in:

```
ID 1-3 (A)
```

Note that, since HUDAP Package is not suitable for manipulating alphanumeric variables, no more than 4 characters are allowed for this type of variables.

Let us clarify the specifications described above by some examples:

```
NAMES = ID 1-3 (A) AGE 5-6 HEIGHT 7-10 (2)
        WEIGHT 11-15 (1) INCOME 16-21;
```

ID is alphanumeric, **AGE** and **INCOME** are integers, **HEIGHT** and **WEIGHT** are reals with respectively 2 and 1 digits after decimal point.

```
NAMES = #1 ID 1-3 (A) AGE 5-6 HEIGHT 7-10 (2)
           WEIGHT 11-15 (1) INCOME 16-21
        #2 B45 TO B72 45-72 ;
```

Here we pick up variables from 2 records.

```
NAMES = ID 1-3 (A) AGE 5-6 HEIGHT 7-10 (2)
           WEIGHT 11-15 (1) INCOME 16-21
        / B45 TO B72 45-72 ;
```

In this example the reading of the variables is identical to the precedent one. Note that record number 1 (**#1**) is not requested at the beginning if the first variable is picked up from this record. A slash (**/**) is used to read variables from record number 2.

```
NAMES = ID 1-3 (A) AGE 5-6 HEIGHT 7-10 (2)
           WEIGHT 11-15 (1) INCOME 16-21
        / B45 TO B72 45-72
        #4 TEST10 TO TEST20 10-20
        #5  ;
```

Here, we skip the third record by specifying "**#4**". Note that record number 5 was specified without any variable. This is a manner to instruct HUDAP system that we have 5 records per case. However, it is also possible to use **RECORDS** sentence instead.

It is not necessary to list the variables in the order of the records. You can choose your own order which is the more convenient for your purpose. The last example can be rewritten as in:

```
NAMES = #4 TEST10 TO TEST15 10-15
        #1 AGE 5-6 INCOME 16-21
        #4 TEST16 TO TEST20 16-20
        #1 HEIGHT 7-10 (2) WEIGHT 11-15 (1)
        #2 B45 TO B72 45-72
        #1 ID 1-3 (A)
        #5  ;
```

Missing values

Very often, your data file lacks complete information on some cases for some variables. Monitoring equipment can malfunction, interviewers can forget to ask a question or record an answer, respondents can refuse to answer, data can be entered incorrectly, and so forth. Missing does not always mean the same as unknown or absent. In order to distinguish why information is missing, you can instruct HUDAP to consider more than one value as missing for each variable. For example, if you code the value **9** for "**Refused to answer**" and the value **0** for "**No answer reported**", you might want to specify both of these values as missing.

The **MISSINGS** sentence declares the missing values for certain variables in your file. The values defined as missing values are never changed on the data; they are simply flagged in memory and are saved if a system file is requested. The HUDAP procedures and transformation sections recognise this flag, and those cases that contain a user-defined missing value are handled differently.

User-missing values defined on the **MISSINGS** sentence are distinguished from the system-missing value. HUDAP assigns the system-missing value when a value resulting from a transformation command like **COMPUTE** is undefined, or when a blank was encountered in the data file for a numeric value. System missing value is also set for a numeric variable with illegal data.

The MISSINGS sentence

The **MISSINGS** sentence consists of the keyword **MISSINGS**, followed by the equal sign, a variable name or variable list and the specified missing value or values, as in:

```
MISSINGS = COUNT 9999  COLOR 8 9  ;
```

This sentence defines 9999 as the missing value for the variable **COUNT** and 8 and **9** for the variable **COLOR**.

- You can specify missing values for any variable previously defined on a **NAMES** sentence of the **DATA** section or a transformation command.
- You can specify a maximum of ten missing intervals for each variable. An interval is defined as a single number, or as a continuous interval such as: **mm TO nn**.

For example, if you conduct a survey and ask the respondents to report their income level, some respondents may refuse to answer the question, others may indicate that they do not know, and some respondents may simply neglect to fill in an answer. In this instance, you might code **9** for "**Refused to answer**",

8 for "**Don't know**", and **0** for "**No answer**". To declare all of these values as missing data for the variable **INCOME**, specify:

```
MISSINGS = INCOME  0 8 9  ;
```

To declare a large number of values as missing, you can either specify a range or use the transformation language in the **CODING** section to change the values to a single value and declare that value missing, as in: :

```
$CODING NAMES=X ;
               REPLACE -999 TO 0 BY 0 ;
```

The **CODING** section, described in the relevant chapter, recodes in this example negative values to **0** and the **MISSINGS** sentence declares **0** as missing.

Referencing Several Variables

You can define missing values for more than one variable on a **MISSINGS** sentence either by specifying a variable list when the missing values are the same for all variables in the list or by specifying several sets of variable names and missing-value specifications when the values are different.

To define the same missing values for several variables, list all of the variables followed by the missing-value specification. Consecutive variables as defined in the **NAMES** sentence of the **DATA** section can be referenced using the keyword **TO**. For example, to declare the value **0** as missing for all the variables beginning with **DEPT9** through **SALARY82** and for the variable **AGE**, specify:

```
$DATA NAMES =  MOHIRED YRHIRED DEPT79 TO DEPT82
               SEX SALARY79 TO SALARY82
               HOURLY81 HOURLY82
               AGE RAISE82 JOBCAT ;
      MISSINGS = DEPT79 TO SALARY82 AGE 0  ;
```

To define different missing values for other variables in your file, specify the additional variables and their missing-value specifications, as in:

```
$DATA NAMES =  MOHIRED YRHIRED DEPT79 TO DEPT82
               SEX SALARY79 TO SALARY82 HOURLY81 HOURLY82
               AGE RAISE82 JOBCAT ;
      MISSINGS = DEPT79 TO SALARY82 AGE 0
                 HOURLY81 HOURLY82 RAISE82 -999
                 JOBCAT 9 ;
```

You can continue this process of specifying the missing values of variables in one or more **MISSINGS** sentences.

If you accidentally name the same variable on two **MISSINGS** sentences, the second specification will override the first.

Specifying Ranges of Missing Values

You can specify a range of values as missing for numeric variables. Use keyword **TO** to indicate an inclusive list of values. For example, **0 TO 1.5** includes the values **0** through (and including) **1.5**. The values must be separated from **TO** by at least one blank space. The sentence

```
MISSINGS = RAISE82 -10000 TO 0  ;
```

defines all negative values and **0** as missing for variable **RAISE82**.

Redefining Missing Values

You can define new missing values for all previously defined missing values using the **MISSINGS** sentence anywhere in the job.

Missing-value specifications for a variable on a **MISSINGS** sentence replace all of the previously defined missing values for that variable.

RECORDS : the number of records per case.

The number of records per case must be specified if there are more than one record per case and if there are more records than the number detected in **NAMES** sentence.

The specification for **RECORDS** is for example:

```
RECORDS  =  3 ;
```

The **RECORDS** sentence is not required if you implicitly define the number of records within **NAMES** sentence as in:

```
NAMES = ID 1-3 (A) AGE 5-6 HEIGHT 7-10 (2)
        WEIGHT 11-15 (1) INCOME 16-21  #5 ;
```

In this example variables are picked up from the first record, but the specification "**#5**" indicates that there are 5 records per case.

NCASES : the number of cases.

The number of cases need not be specified unless only a portion of the file is to be read (e.g., if a file contains 1000 cases and you want to analyse the first 50). HUDAP stops reading the data when an end-of-file indicator is found.

The specification for **NCASES** is for example:

```
NCASES  =  50 ;
```

FILE : the datafile pathname

The pathname of a file is a full name which can be handled by the operating system. The syntax of the pathname depends upon the operating system in which you are running HUDAP .

You must specify the pathname of the file which contains the data. The pathname is limited to 40 characters. Example for a DOS system:

```
FILE  =  '\MEHKAR\DATABANK\MONCO.DAT';
```

BLANK : the value of blank field

By default, when a blank field is picked up for a numerical variable (integer or real), the value of the variable in the specific case is set to system-missing value. However, the user can choose another value for blank field as in:

```
BLANK = 0 ;
```

Variable and Category Labels

Although you can construct variable names to represent what the variable actually measures, it is sometimes difficult to fully describe a variable in an eight-character name. Likewise, categories of variables sometimes have no apparent meaning by themselves. Use **VARLABS** sentence to assign labels to variables in your file, and **CATLABS** sentence to assign labels to categories of variables, (that is, the possible values that a given variable can take). HUDAP displays these variable and category labels on the output produced by the procedures.

Labels in HUDAP are specified as "literals". In the sentence

```
VARLABS = SALARY82 'Salary in 1982'  ;
```

the label for variable **SALARY82** is specified as the literal "**Salary in 1982**."

- Enclose literals within apostrophes
- Enter an apostrophe as part of a label by entering the apostrophe twice with no separation.

For example

```
VARLABS = SALARY82 'Employee''s salary in 82'  ;
```

The VARLABS sentence

Use the **VARLABS** sentence to assign an extended descriptive label to variables. Specify the variable name followed by at least one comma or blank and the associated label enclosed in apostrophes, as in:

```
VARLABS = YRHIRED 'Year of first hiring'
          DEPT82 'Dpt. of employment in 82'
          SALARY82 'Yearly salary in 1982'
```

This sentence assigns variable labels to the variables **YRHIRED**, **DEPT82**, **SALARY82**, and **JOBCAT**.

- A variable label applies to only one variable.
- The variable must have been previously defined, either in **NAMES** sentence of **DATA** section or in a transformation process.
- Each variable label can be up to 24 characters long and can include blanks and any character.

However, you can define many labels for a single variable. This is useful when 24 characters are not sufficient to label a variable. In that case, in general, each label will be printed on a single line. Thus, the general format of **VARLABS** sentence is:

```
VARLABS = <variable name 'label_1'
                         'label_2'
                         ' ......'
                         'label_j'
           variable name 'label_1'
                         'label_2'
                         '.......'
                         'label_k'
           ......................
           variable name 'label_1'
                         'label_2'
                         '.......'
                         'label_m'> ;
```

The CATLABS sentence

Use the **CATLABS** sentence to provide descriptive labels for categories. The **CATLABS** sentence contains a variable name, or variable list, followed by a list of the category values with their associated labels. The sentence:

```
CATLABS = DEPT82  0 'Not reported'
                  1 'Administrative'
                  2 'Project Directors'
                  3 'Chicago operations'
                  4 'St Louis operations'  ;
```

assigns labels to the values **0**, **1**, **2**, **3**, and **4** of **DEPT82**.

- You can assign labels to categories of any previously defined variable.
- Enclose each category label in apostrophes.
- Category labels cannot exceed 24 characters and can contain any characters including blanks.

Category labels are automatically displayed on the output of many procedures and are saved in the dictionary of a system file. It is not necessary to enter category labels for all the variables or values in your file. In some instances, the value itself is completely descriptive, such as the values for **SALARY82**.

To assign the same labels to the same categories of several variables, list all of the variables followed by the categories and associated labels. Also, additional sets of variable names and category labels can be specified on the same sentence, as in:

```
CATLABS = DEPT79 TO DEPT82  0 'Not reported'
                            1 'Administrative'
                            2 'Project Directors'
                            3 'Chicago operations'
                            4 'St Louis operations'
                       SEX  1 'Male'  2 'Female'
                    JOBCAT  1 'Officials and Managers'
                            2 'Professionals'
                            3 'Technicians'
                            4 'Office and clerical'
                            5 'Craftsmen'
                            6 'Service workers'  ;
```

A blank is sufficient to separate category labels for one variable or variable list from the next variable or variable list.

If you assign category labels to any variable that already has value labels assigned to it, the new assignment completely replaces the old assignment. Specifications of category labels are not additive.

DATA section menu

NAMES = <[r] var list [c] [t]
. . . . . .
[r] var list [c] [t]> ;

Naming, locating and defining variables. Each variable name is restricted to eight characters. Variable names are used in the HUDAP instructions to identify the variables.

Square brackets in the syntax description mean optional specification.

"**r**" : record specifications. It can be:
#n for going to record number "n".
m**(/)** or **//.../** (m times) for skipping "m" records.

"**c**" : column specifications. The form is:
start_col**-**end_col where
start_col : starting column
end_col : ending column

"**t**" : format type specifications. It can be:
(d**)** for real variable with "d" decimal digits.
(A) for alphanumeric variable.
If "**t**" is missing, the variable is assumed to be integer.

MISSINGS = <var list val list
var list val list
. . . . . .
var list val list> ;

Defining the list of values "**val list**" as missings for the corresponding variable list "**var list**".

RECORDS = <value> ;

The number of records per case.
The default is 1.

NCASES = <value> ;

The number of cases or observations to be read. Not required if data are to be read until end-of-file.

FILE = '<char>' ;

"**char**" is the datafile pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

BLANK = <value> ;

The value of blank field for numerical variable.
The default is system-missing value.

```
VARLABS = <variable name 'label_1'
                         'label_2'
                         ' ......'
                         'label_j'
          variable name 'label_1'
                         'label_2'
                         '.......'
                         'label_k'
          ......................
          variable name 'label_1'
                         'label_2'
                         '.......'
                         'label_m'> ;
```

Defining many labels for each variable. In general, for each variable name, each "**label_i**" will be printed on a single line. Each "**label_i**" can be up to 24 characters long and can include blanks and any characters.

```
CATLABS = <var list  value  'label'
                     value  'label'
                    .....   '.....'
                     value  'label'
          var list   value  'label'
                     value  'label'
                    ......  '.....'
                     value  'label'
          ......................
          var list   value  'label'
                     value  'label'
                      ..... '.....'
                     value  'label'> ;
```

Defining the label "**label**" for each category value "**value**" in "**var list**".

The *CODING* Section

The ability to transform data before you analyse it or after preliminary analysis is often as important as the analysis itself. You may want to perform simple data-cleaning checks, correct coding errors, or adjust an inconvenient coding scheme. Or you may want to construct an index from several variables or rescale several variables prior to analysis. The HUDAP transformation instructions provide these and many other possibilities.

This chapter describes the major components of the HUDAP **CODING** Section.

Introduction to Data Transformations

Two sections are at the core of data transformation: **CODING** and **COMPUTE**. The **CODING** section, documented here, changes the coding scheme of an existing variable on a value-by-value basis or for ranges of values. For example, to change the coding order for three questionnaire items from **0** for "**Agree**," **1** for "**No opinion**," and **2** for "**Disagree**" to **1, 0, and -1**, respectively, specify:

```
$CODING
       NAMES = ITEM1 ITEM2 ITEM3 ;
       REPLACE 0 BY 1 | 1 BY 0 | 2 BY -1 ;
```

The three values are recoded as required for each of the variables in the variable list.

The **COMPUTE** section, documented in the relevant chapter, computes a new variable as some combination or transformation of one or more existing variables. For example, to build a simple index averaging the three questionnaire items just recoded, specify:

```
$COMPUTE   INDEXQ = (ITEM1 + ITEM2 + ITEM3)/3 ;
```

For each case, the three items are added together, the sum is divided by **3**, and the result is stored in the new variable **INDEXQ**.

You can make execution of data transformations conditional on other information in the data via the **IF...THEN...ELSE...ENDIF** feature. For example, to establish a dichotomous (two-valued) variable indicating cities that are classified as poor based on median family income, specify:

```
$COMPUTE POOR=0 ;
         IF FAMINC LE 10000 THEN
              POOR=1 ;
         ENDIF
```

In the **COMPUTE** section, variable **POOR** is initialised to **0** for all cities, and the **IF** command changes **POOR** to **1** for cities with values for **FAMINC** less than or equal to $10,000.

The Coding Section

The most direct data transformation is the **CODING** section, which instructs HUDAP to change the code for a variable as the data are being read. The sequence

```
$CODING NAMES = X ;
        REPLACE 0 BY 9 ;
```

instructs HUDAP to change all zeros found for variable **X** to nines.

The variable or variables to be recoded must already exist and must be specified in the **NAMES** sentence. You can specify as many value specifications as needed, ending each specification by the symbol "**|**", as in:

```
$CODING NAMES = ITEM1 ;
        REPLACE 0 BY 1 | 1 BY 0 | 2 BY -1 ;
```

You can use multiple input values in a single specification but only one output value following the **BY** keyword, as in

```
$CODING NAMES = RESPONSE ;
        REPLACE 8 9 BY 1 | 4 TO 7 BY 2 | 1 2 BY 3 ;
```

The **REPLACE** command is evaluated from left to right, and the values for a case are recoded only once per **REPLACE** command. For example, if a case has an input value of **0** for variable **ITEM1**, the sequence

```
$CODING NAMES = ITEM1 ;
        REPLACE 0 BY 1 | 1 BY 0 | 2 BY -1 ;
```

recodes **ITEM1** to **1** and HUDAP then moves on to the next command. The value is not recoded back to **0** by the second value specification. Input values not mentioned in the **REPLACE** command are left unchanged.

You can name multiple variables for the same value specifications, as in:

```
$CODING NAMES = ITEM1 TO ITEM3 ;
        REPLACE 0 BY 1 | 1 BY 0 | 2 BY -1 ;
```

Only one **NAMES** sentence and one **REPLACE** command are allowed in a **CODING** section. To specify different values for different variables you must re specify **CODING** section, as in:

```
$CODING NAMES = AGE ;
        REPLACE 0 BY 9 ;
$CODING NAMES = ITEM1 TO ITEM3 ;
        REPLACE 0 BY 1 | 1 BY 0 | 2 BY -1 ;
```

Keyword TO

To recode ranges of values for numeric variables into a single value, use keyword **TO**. For example, to recode all individuals below the United States voting age to **0** and leave all other ages unchanged, specify:

```
$CODING NAMES = AGE ;
        REPLACE 1 TO 17 BY 0 ;
```

You can also use keyword **TO** to collapse variable **AGE** into gross categories, perhaps for tabular display, as in:

```
$CODING NAMES = AGE ;
        REPLACE 1 TO 20 BY 1 | 20 TO 25 BY  2 |
                25 TO 30 BY 3 | 30 TO 35 BY  4 |
                35 TO 40 BY 5 | 40 TO 45 BY  6 |
                45 TO 50 BY 7 | 50 TO 55 BY  8 |
                55 TO 60 BY 9 | 60 TO 65 BY 10 |
                65 TO 999 BY 11 ;
```

CODING section menu

NAMES = <var list> ;

Names of the variables for which codes are replaced by new ones.

REPLACE <val list BY value | val list BY value | ...| val list BY value> ;

Each "**val list**" or "**char list**" is replaced by its corresponding value in the data. This is done only for the variables defined in **NAMES**. The symbol "**|**" is a delimiter between two replacement definitions.

The *COMPUTE* Section

The ability to transform data before you analyse it or after preliminary analysis is often as important as the analysis itself. You may want to perform simple data-cleaning checks, correct coding errors, or adjust an inconvenient coding scheme. Or you may want to construct an index from several variables or rescale several variables prior to analysis. The HUDAP transformation instructions provide these and many other possibilities.

This chapter describes the major components of the **COMPUTE** Section

Introduction to Data Transformations

Two sections are at the core of data transformations: **CODING** and **COMPUTE**. The **CODING** section, documented in the relevant chapter, changes the coding scheme of an existing variable on a value-by-value basis or for ranges of values. For example, to change the coding order for three questionnaire items from **0** for "**Agree**," **1** for "**No opinion**," and **2** for "**Disagree**" to **1**, **0**, and **-1**, respectively, specify:

```
$CODING
        NAMES = ITEM1 ITEM2 ITEM3 ;
        REPLACE 0 BY 1 | 1 BY 0 | 2 BY -1 ;
```

The three values are recoded as required for each of the variables in the variable list.

The **COMPUTE** section, documented here , computes a new variable as some combination or transformation of one or more existing variables. For example, to build a simple index averaging the three questionnaire items just recoded, specify:

```
$COMPUTE   INDEXQ = (ITEM1 + ITEM2 + ITEM3)/3 ;
```

For each case, the three items are added together, the sum is divided by **3**, and the result is stored in new variable **INDEXQ**.

You can make execution of data transformations conditional on other information in the data via the **IF...THEN...ELSE...ENDIF** feature. For example, to establish a dichotomous (two-valued) variable indicating cities that are classified as poor based on median family income, specify:

```
$COMPUTE POOR=0 ;
        IF FAMINC LE 10000 THEN
           POOR=1 ;
        ENDIF
```

In the **COMPUTE** section variable **POOR** is initialised to **0** for all cities, and the **IF** command changes **POOR** to **1** for cities with values for **FAMINC** less than or equal to $10,000.

The Compute Section

Often, you want to create a new variable or transform an existing variable using information from other variables on your file. The **COMPUTE** section generates a variable that is constructed on a case-by-case basis as an arithmetic transformation (or logical expression) of existing variables and/or constants. For example, the sequence

```
$COMPUTE INCOME = WAGES + BONUS + INTEREST + OTHERINC ;
```

assigns the sum of four existing variables to variable **INCOME** for each case.

To compute a variable, specify the target variable on the left of the equals sign and the expression on the right. You can compute several target variables per **COMPUTE** section ending each assignment by a semicolon "**;**". The expression must be numeric (return a number). Facilities for computing numeric variables are described in the following :

Computing Numeric Variables

The sequence

```
$COMPUTE X=1 ;
```

assigns the value **1** to variable **X** for every case. Numeric variable **X** is the target and **1** is the numeric expression.

The target variable can be an existing variable or a new variable defined by the **COMPUTE** section itself. If the target variable already exists when HUDAP encounters the assignment, the old values are replaced. If the target variable does not exist, it is created by the **COMPUTE** section. New numeric variables are initialised to the system-missing value.

Once computed, the variable exists in memory in its new form and can be analysed, and stored on a new system file along with all other variables in memory. If it is a new variable, it is added to the end of the dictionary.

The expression to the right of the equals sign can be composed of existing variables, arithmetic operators such as **+** and **-**, arithmetic or statistical functions such as **SQRT** or **MEAN**, system scalar variables, numeric constants, scalar variables (see below), and so forth. For example, the sequence

```
$COMPUTE PCTWAGES=(WAGES/INCOME)*100 ;
```

creates **PCTWAGES** as a percentage of **INCOME** through use of the slash for division, the asterisk for multiplication, and the parentheses to clarify the order of operations . The sequence

```
$COMPUTE   _MINCOME = MEAN(INCOME) ;
```

returns in scalar variable **_MINCOME** the mean value of variable **INCOME**. Facilities for handling numeric expressions are documented in this chapter.

Missing Values

If a value is missing in any of the variables (for a certain case) used in an expression when the computation is executed, HUDAP nearly always returns the system-missing value since the operation is indeterminate. For example, in the sequence

```
$COMPUTE PAYHOURS = WORKDAYS * 7.5 ;
```

variable **PAYHOURS** cannot be computed for any case where the variable **WORKDAYS** is missing.

HUDAP also returns missing values when the expression itself is undefined. In the sequence

```
$COMPUTE PCTWAGES=(WAGES/INCOME)*100 ;
```

variable **PCTWAGES** is considered indeterminate for a case when the value for **INCOME** is **0**, since division by **0** is not defined .

If these rules do not fit your application, you should be able to specify exactly what you want using one or more of the functions described in this chapter.

Numeric Expressions

While numeric expressions are commonly used with the **COMPUTE** command, they can be used as part of a logical expression in the **IF...THEN...ELSE ...ENDIF** feature.

The facilities for numeric expressions are:

- Arithmetic Functions - enable you to truncate a variable, use the square root or the log of a variable, and so on.
- Statistical Functions - enable you to compute statistics such as the mean or standard deviation of a variable.
- Missing-Value Functions - used to control the number of missing values in a variable or to test whether a specific cell in the data matrix is missing or not.

Numeric expressions may also include arithmetic operations and numeric constants.

Arithmetic Operations

Arithmetic operators and their meanings are:

+ Addition
- Subtraction.
***** Multiplication.
/ Division.
****** Exponentiation. (See also the **SQRT** function for calculating the square root.)

Operators cannot appear consecutively. You cannot specify **VAR1+*VAR2**. In addition, you cannot use arithmetic operators implicitly. For example, you cannot specify **(VAR1)(VAR2)** in place of **VAR1*VAR2**.

The arithmetic operators and the parentheses serve as delimiters. You can insert blanks (or commas) before and after an operator to improve readability, as in:

```
$COMPUTE PCTWAGES = (WAGES / INCOME) * 100 ;
```

Numeric Constants

Constants used in numeric expressions or as arguments to functions can be integer or non integer, depending on the application or the function. You can specify as many digits in a constant as needed, as long as you understand the precision restrictions of your computer. Numeric constants can be signed (**+** or **-**) but cannot contain any other special characters such as the comma or dollar sign. You can use the alternative exponential format by specifying **E** and a signed exponent after a number, as in:

```
$COMPUTE X = Y * 5.1E+5 ;
```

This command returns the value **510,000.0** for a case with value **1** for variable **Y**. This is also known as scientific notation.

The Order of Operations

You can use variables, constants, and functions with arithmetic operators to form complex expressions. The order in which HUDAP executes the operations of a **COMPUTE** sequence when the data are read and the target variable is constructed is (1) functions; (2) exponentiations; (3) multiplication, division, and unary -; and (4) addition and subtraction. Thus, in the sequence

```
$COMPUTE X = SQRT(Y1) / SQRT(Y2) + SQRT(Y3) ;
```

the square root operations are executed first, then the division, and then the addition.

You can control the order of operations by enclosing in parentheses the operation you want to execute first. The sequence

```
$COMPUTE X = SQRT(Y1) / (SQRT(Y2) +SQRT(Y3)) ;
```

returns a different value than the previous example since the square roots of **Y2** and **Y3** are summed before the division is performed.

The order of execution for operations at the same level unspecified by parentheses is from left to right. Thus, the sequence

```
$COMPUTE TESTVAR = (X/Y*Z) + 1 ;
```

returns **3** for a case with a value of **2** for **X**, **Y**, and **Z** since 2 (variable **X**) divided by **2**(variable **Y**) is **1**, times **2** (variable **Z**) is **2**, plus **1** is **3**. However, the sequence

```
$COMPUTE TESTVAR = (X/(Y*Z)) + 1 ;
```

returns **1.5** for the same case since the expression **(2/(2*2))** returns .**5** when the parentheses alter the order of execution.

If you are ever unsure of the order of execution, use parentheses to make the order explicit - even if you specify the order HUDAP would use anyway.

Numeric Functions

You can use the functions described below in any numeric expression with the **IF** feature in the **COMPUTE** section. Numeric functions always return numbers (or the system-missing value whenever the result is indeterminate). The expression to be transformed by a function is called the "argument". Most functions have one or two variable names as arguments. For example, to generate the square root of variable **X**, specify variable **X** as the argument to the **SQRT** function, as in **SQRT(X)**. Enclose arguments in parentheses, as in

```
$COMPUTE INCOME = INT(INCOME) ;
```

where the **INT** function returns the integer portion of variable **INCOME**. Separate two arguments with commas or blanks, as in

```
$COMPUTE _N = VALID(INCOME,_I) ;
```

where the **VALID** function returns the value **0** (if **INCOME** has missing value in case **_I**), or 1.

These functions, their arguments, their applications, and how they handle missing values are discussed below.

Arithmetic Functions

ABS(arg)	Absolute value. **ABS(-4.7)** is **4.7**; **ABS (4.7)** is **4.7**.
INT(arg)	Truncate to an integer. **INT(-4.7)** is **-4**.
SIGN(arg)	**-1** if **arg<0** ; **0** if **arg=0** ; **1 if arg>0**
SQRT(arg)	Square root.
EXP(arg)	Exponential. **e** is raised to the power of the argument.
LOG10(arg)	Base 10 logarithm.
LOG(arg)	Natural or Naperian logarithm (base **e**).
ASIN(arg)	Arcsine. The result is given in radians.
ACOS(arg)	Arccosine. The result is given in radians.
ATAN(arg)	Arctangent. The result is given in radians.
SIN (arg)	Sine. The argument must be specified in radians.
COS(arg)	Cosine. The argument must be specified in radians.
TAN(arg)	Tangent. The argument must be specified in radians.

All arithmetic functions have single arguments.

Arguments cannot be numeric expressions as in **INT(A**2/B)**. You can get the desired result by

```
$COMPUTE  INTAB = A**/B ;  INTAB = INT(INTAB) ;
```

Statistical Functions

MEAN(arg)	Mean of the values in the variable.
STDV(arg)	Standard deviation of the variable.

Missing-Value Functions

VALID(arg)	Count of the number of valid values in the variable "**arg**".
VALID(arg1,_arg2)	returns **0** if variable "**arg1**" has missing value in case number "**_arg2**", **1** otherwise.

"**arg1**" is a regular variable while "**_arg2**" is a constant or a scalar variable.

Scalar variables

Data are codes representing characteristics (e.g., sex, eye colour), values of measurements(e.g., height, weight) or responses to questions. Each characteristic, measurement or response is called a regular variable or simply a variable. This kind of variable, the more commonly used, is valued over all the cases. HUDAP allows the use of another kind of variable : the scalar variable, which is a single-valued variable. There are 3 kinds of scalar variables.

"Independent" scalar variable

Sometimes the user may save and store the value of a specific computation result in order to use it later. He can do it by means of scalar variables. The names of these variables must begin by an underline sign ("_"). For example :

```
_N = VALID(SALARY) ;
```

the number of non-missing values in **SALARY** is returned into **_N**

Data cell scalar variable

The second kind of scalar variable is one which is related to a regular variable, namely a specific cell in the rectangular data matrix. For this kind of scalar variable the syntax is different and is expressed as **var[c]**, where "**var**" is the name of a regular variable and "**c**" a case number (constant or independent scalar variable). For example :

```
NEWVAR[_I] = _I**2 ;
```

System scalar variables

Special system scalar variables are used in data transformations to determine the number of cases read by the system (or set by the user), the current number of variables or the system-missing value. As for the "independent" scalar variables, the names of these variables begin with an underline sign. You can modify a system variable and you can use it anywhere in the transformation language. The reserved names for the system scalar variables are :

_MISS	System-missing value. The variable is initialised by HUDAP to 1.E+38
_NVARS	Current number of variables.
_NCASES	Number of (read) cases.

Note that the alteration of **_NCASES** leads to the destruction of the data. In fact, the assignment of a number to **_NCASES** is useful only when ones wants to generate his own data instead of reading the data from an input file.

The alteration of **_NVARS** has another effect. Suppose **n1** is the current number of variables, if you modify the number of variables by

```
$COMPUTE  _NVARS = n2 ;
```

where **n2<n1** , the effect is that the last (**n1-n2**) variables are deleted from the memory. This is useful to suppress temporary variables.

Conditional computations

In certain situations, you may want to construct or alter variables in one way for one subset of cases and in other ways for other subsets. You instruct HUDAP to execute data computations conditionally via the **IF...THEN...ELSE... ENDIF** structure. The syntax is:

```
IF  l_expression  THEN
    target variable_1 = expression_1 ;
    target variable_2 = expression_2 ;
      .............   =   .........  ;
    target variable_k = expression_k ;
                ELSE
         ............ = .......... ;
         ............ = .......... ;
ENDIF
```

where **l_expression** is a logical expression. A logical expression consists of one or more sets of relations. A relation, in turn, is composed of an algebraic comparison of two quantities. A relation is formed by joining two arithmetic expressions with a relational operator. Its form is then

```
(arithmetic_expression_1) relational operator
(arithmetic_expression_2)
```

An arithmetic expression may be composed of variable names, scalar variable names, functions, constants, and arithmetic operators. No undefined variable may be used in the expression.

The two arithmetic expressions must be linked by one and only one of the six relational operators. The following table indicates the relation, keyword, and definition of each of these operators.

Relation	Keyword	Definition
Greater than or equal to	**GE**	If expression 1 is greater than or equal to expression 2 then the value of the relation is true; otherwise it is false
Less than or equal to	**LE**	If expression 1 is less than or equal to expression 2, the value of the relation is true.
Greater than	**GT**	If expression 1 is greater than expression 2, the value of the relation is true.
Less than	**LT**	If expression 1 is less than expression 2, the value of the relation is true.
Equal to	**EQ**	If expression 1 is exactly equal to expression 2, the value of the relation is true.
Not equal to	**NE**	If expression 1 is not exactly equal to expression 2, the value of the relation is true.

The six relational operators are keywords, and as such, must be spelled correctly and separated from the arithmetic expressions by one or more common delimiters. Some examples of relations follow.

```
Vara EQ 1
```

In this simple case, **Vara** is the first arithmetic expression, **1** is the second, and **EQ** is the relational operator indicating equality. This relation will be evaluated as true when and only when **Vara** has a value of precisely **1**. A more complex relation might be

```
Vara*Varb GT MEAN(Vara)*MEAN(Varb)
```

In this case, the first arithmetic expression consists of multiplying **Vara** by **Varb**; the second expression is the mean value of **Vara** multiplied by the mean value of **Varb**; and the relational operator is **GT**, which indicates that for the relation to be evaluated as true, the value of the first expression must be greater than that of the second. To relate this example to the **IF...ENDIF** structure, the researcher might wish, for example, to create a new variable from these two variables, which will have two values: if the first expression is greater than the second, **1**; otherwise, **2**. This could be accomplished by using the following sequence:

```
$
```

While a logical expression can consist of a single relation, it can also consist of several relations joined to each other by logical operators which may be either **AND** or **OR**. When this is the case, a logical expression has the following format:

```
relation   logical operator   relation
```

Both **AND** and **OR** are keywords and must be separated from other elements by one or more common delimiters. They must also conform to all other rules governing the use of keywords. **AND** and **OR** are used to combine relations, and while no more than one logical operator may be used to combine two relations, many relations may be combined into a larger logical expression by means of their use. No matter how many relations are used to build a logical expression, the result of the evaluation of this expression must be a single value - true or false. Logical operators combine the values of relations according to the following rules:

1 **AND**
the resulting logical expression will have the value true if and only if the relations directly preceding and following the operator have the value true.

2 **OR**
the resulting logical expression will be true if either or both of the constituent relations have the value true.

COMPUTE section menu

```
target (scalar) variable_1 = expression_1 ;
target (scalar) variable_2 = expression_2 ;
  ......................   =   .........  ;
target (scalar) variable_k = expression_k ;
```

variable_i to the left side of the assignment may be scalar or regular. A scalar variable is a single-valued variable while a a regular variable is a variable valued over all the cases. There are 3 kinds of scalar variables :
1) an "independent" scalar variable which must begin with the underline symbol (for example **_MEANV**)
2) a scalar variable related to a regular one, namely a specific cell in the rectangular data matrix. The syntax is then : **var[c]** where "**var**" is the name of a regular variable and "**c**" a case number (for example : **INCOME[112]**)
3) system scalar variables : **_MISS**, **_NVARS**, **_NCASES** variable_i can be an existing variable or a new variable. If it already exists the old value is replaced. If it does not exist, it is created by **COMPUTE**.

expression_i to the right side of the assignment can be composed of existing variables, arithmetic operators such as **+** and **-**, arithmetic functions such as **SQRT** or **LOG**, statistical functions as **MEAN** or **STDV**, system variables such as **_MISS** or **_NCASES**, numeric constants, and so forth.
Computations can be made conditional on other information in the data via the **IF...THEN...ELSE...ENDIF** feature :

```
IF  l_expression  THEN
    target variable_1 = expression_1 ;
    target variable_2 = expression_2 ;
      .............   =   .........  ;
    target variable_k = expression_k ;
                 ELSE
         ............ = .......... ;
         ............ = .......... ;
ENDIF
```

where **l_expression** is a logical expression as for example :

```
INCOME - (FAMSIZE*600) LE 3000
```

or

```
SEX EQ 1 AND RACE EQ 6
```

The *MATRINP* Section

There may be circumstances in which you wish to input data in the form of a matrix. It might be necessary, for example, to perform an analysis (e.g. **WSSA1**) on a matrix that represents the results of some earlier analysis performed on raw data. In such a case, rather than repeating the analysis on the raw data in order to obtain the matrix, it is possible to store the matrix itself and recall it later using the **MATRINP** section.

The **MATRINP** section is analogous to the **DATA** section, with input in matrix form rather than as raw data. The sentences **NAMES FILE** and **VARLABS** carry the same meaning here as in the **DATA** section, although in the **MATRINP** section the reading refers to a single row of the matrix rather than a single case, as in a data file. Refer to **DATA** section for a detailed description of **NAMES** sentence

The sentence **NAMES** apply only in the case of a square matrix. **NAMES** specifies the names of the variables along each side of the matrix.

The **MISSINGS** sentence in the **MATRINP** section, in contrast to the **DATA** section, defines the missing values for any matrix entry. That is, rather than defining missing values for specific variables, you define the missing values that might appear in any position in the matrix, with the sentence:

```
MISSINGS = <value list> ;
```

Sometimes, the user may have to reverse the sign of one or several variables in the matrix (rows/columns). This can be performed by the sentence:

```
CHNSIGN = <var list> ;
```

By default, when a blank field is picked up for a specific cell of the matrix, the value of this cell is set to system-missing value. However, the user can choose another value for blank field by means of the sentence:

```
BLANK = <value> ;
```

MATRINP section menu

NAMES = <[r] var list [c] [t]
.....
[r] var list [c] [t]> ;

Naming, locating and defining variables for a square matrix. Each variable name is restricted to eight characters. Variable names are used in the HUDAP instructions to identify the variables.

Square brackets in the syntax description mean optional specification.

"**r**" : record specifications. It can be:
#n for going to record number "n".
m**(/)** or **//.../** (m times) for skipping "m" records.

"**c**" : column specifications. The form is:
start_col**-**end_col where
start_col : starting column
end_col : ending column

"**t**" : format type specifications. It can be:
(d**)** for real variable with "d" decimal digits.
(A) for alphanumeric variable.
If "**t**" is missing, the variable is assumed to be integer.

FILE = '<char>' ;

"**char**" is the matrix file pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

MISSINGS = <val list> ;

Numbers representing missing values in the matrix.

CHNSIGN = <var list> ;

To change the sign of each variable listed in "**var list**".

BLANK = <value> ;

The value of blank field.
The default is system-missing value.

```
VARLABS = <variable name 'label_1'
                         'label_2'
                         ' ......'
                         'label_j'
          variable name 'label_1'
                         'label_2'
                         '.......'
                         'label_k'
          ......................
          variable name 'label_1'
                         'label_2'
                         '.......'
                         'label_m'> ;
```

The *MODIFILE* and *RESTORE* Sections

Introduction

A set of data is usually analysed many times by HUDAP jobs. For example, the data may first be examined for extreme values (outliers) and for fixing errors in data file; then necessary transformations can be performed, or relationships between the variables studied. The results of an analysis may suggest that further analyses are needed.

The **MODIFILE** section allows you to store your data or results from an analysis, using the HUDAP (System) File, so that you can reuse them in other HUDAP jobs, using **RESTORE** section. There are several advantages to storing data in a system file:

- The system file contains the number of variables, the variable names, the variable and category labels (if any),the scalar variable names and flags for missing values. Therefore, when you use a system file as input (by **RESTORE**), you do not specify this information unless you want to change it.
- Data are read efficiently from a system file; the cost of reading a large amount of data from a system file is substantially less than when an ordinary reading is used (the system file is unformatted).
- Data are stored in the system file after transformation and case selection have been performed.

Using a system file shortens and simplifies the Control Language instructions needed for subsequent analyses.

MODIFILE section can be used also to select cases via the conditional **IF...ENDIF** feature and/or select variables via the **NAMES** sentence.

Let us detail the various functions of **MODIFILE** section:

HUDAP system file

Datafile can be saved as system file , that is a binary file containig tha data, the variable names, the missing values ..for example :

```
$DATA NAMES = V1 TO V10 1-10 ;
      FILE = 'MYDATA' ;
$MODIFILE
           FILE = 'SAVDAT' ;
           SAVE ;
```

This job saves all variables and all cases on system file '**SAVDAT**'. Thereafter, in another HUDAP job, you can for example get frequencies by :

```
$RESTORE FILE = 'SAVDAT' ;
$FREQ NAMES = V1 TO V10 ;
```

The **RESTORE** section retrieves all the information saved in the previous job.

Selecting cases on system file

The **MODIFILE** section can be used also to select cases via the **IF...ENDIF** feature. For example :

```
$DATA NAMES = V1 TO V10 1-10 ;
      FILE = 'MYDATA' ;
$MODIFILE IF V1 EQ 2 THEN
             FILE = 'SAVDAT' ;
             SAVE ;
          ENDIF
```

Now, the system file '**SAVDAT**' contains only cases where **V1=2**. A second HUDAP job reffering to '**SAVDAT**' can be for example :

```
$RESTORE FILE = 'SAVDAT' ;
$CRST ROWNAMES = V5 ; COLNAMES = V6 ;
```

Selecting variables on system file

The **MODIFILE** section can be used also to select specific variables. For example::

```
$DATA NAMES = V1 TO V10 1-10 ;
      FILE = 'MYDATA' ;
$MODIFILE  NAMES = V3 V5 TO V9 ;
           FILE = 'SAVDAT' ;
           SAVE ;
```

Selecting cases in memory

Until now we used **MODIFILE** with **FILE** sentence. When this sentence is omitted, the data are saved into the memory instead on a system file. For example:

```
$DATA NAMES = V1 TO V10 1-10 ;
      FILE = 'MYDATA' ;
$MODIFILE IF V1 EQ 2 THEN
            SAVE ;
          ENDIF
$MONCO NAMES = V1 TO V10 ;
```

Note that after the **MODIFILE** section any procedure (here **MONCO**) refers only to cases where **V1=2**, because in the memory of HUDAP system, the original data where replaced by data where **V1=2**.

Processing subgroups.

Now suppose that **V1** has **5** categories and that you want to process **MONCO** for each category of **V1**, the job will be:

```
$DATA NAMES = V1 TO V10 1-10 ;
      FILE = 'MYDATA' ;
$FOR CAT = 1 TO 5 ;
     FF = F1 TO F5 ;
     $MODIFILE IF V1 EQ CAT THEN
                  FILE = 'FF' ;
                  SAVE ;
               ENDIF
$ENDFOR
$FOR FF = F1 TO F5 ;
     $RESTORE FILE = 'FF' ;
     $MONCO NAMES = V1 TO V10 ;
$ENDFOR
```

In the first **FOR...ENDFOR** block, five system files (**F1**, **F2**, **F3**, **F4** and **F5**) are saved, each corresponding to a subpopulation (for **V1=1**, **2**, **3**, **4** and **5**). In the second **FOR...ENDFOR** block, **MONCO** is processed on each restored system file.

Note that, in **MODIFILE** section, the specifications must end with the command **SAVE** for execution.

MODIFILE section menu

NAMES = <var list> ;

Names of the variables to be saved. If this sentence is omitted, all the variables are saved.

FILE = '<char>' ;

"**char**" is the HUDAP outfile pathname not exceeding 40 characters in length. It must be enclosed in apostrophes. If sentence **FILE** is omitted, the selected variables and cases are saved into the memory.

SAVE ;

Command keyword to save cases and variables specified before. You can save several system files in the same **MODIFILE** section. The variables are selected by the **NAMES** sentence, while the cases are selected by an **IF** ... **THEN** ... **ELSE** ... **ENDIF** feature.

RESTORE section menu

FILE = '<char>' ;

"**char**" is the HUDAP infile pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

The *SET* Section

The **SET** section allows you to set various parameters that are relevant to the execution of a HUDAP job (e.g. linesize, pagesize), to request a preliminary editing for syntax errors (without actually processing the job), to check computational errors, and to give a title to your job.

This section can accompany any HUDAP job, but its inclusion is entirely optional.

Below is a brief description of each of the options in the **SET** section.

EDIT

If you would like to edit a HUDAP input job without actually processing it, in order to detect syntax errors, simply include the command "**EDIT**" in the **SET** section.

To cancel a previous **EDIT** command insert the command "**NOEDIT**". "**NOEDIT**" is the default option. If you do not specify anything, HUDAP will not edit the job but will simply try to process it.

NAME

The sentence

```
NAME = 'title' ;
```

will assign the title specified to your job.

PAGESIZE

You can select what is to be the maximum number of printed lines per page with the sentence

PAGESIZE = <value> ;

where "**value**" is the chosen maximum number.

LINESIZE

It is possible to select the width of the printed output. For a small (narrow) output, let

LINESIZE = 80 ;

The default value, for large (wide) output is **132**.

CHECK

You may wish to instruct HUDAP to abort a job in the event of a computational error (attempted division by zero, inappropriate argument in a mathematical function) and to output a suitable error message. This can be done by including the command **CHECK** in the **SET** section. In order to cancel a previous **CHECK** command, insert the sentence **NOCHECK** at the appropriate place. **NOCHECK** is the default: when a computational error occurs, the result is set to the system missing value.

SET section menu

NAME = '<char>' ;

To give the title "**char**" to the job.

EDIT ;

To edit a HUDAP input job in order to detect syntax errors without processing.

NOEDIT ;

To cancel a previous **EDIT** command.
This is the default.

PAGESIZE = <value> ;

Maximum number of printed lines per page.

LINESIZE = <value> ;

Small line if value = 80 , large line otherwise.
The default is 132.

CHECK ;

To check a computational error as division by 0, wrong argument in a mathematical function etc... The job is aborted and a suitable error message is outputted.

NOCHECK ;

To cancel a previous **CHECK** command. When a computational error occurs, the result is set to the system missing value.
This is the default.

The *FOR...ENDFOR* Utility

Often a user needs to perform the same operations on several of the variables in the file or for various options. Instead of preparing a separate sequence for each variable or for each option to be processed, the user can reduce the amount of control-line preparation necessary by utilising the **FOR...ENDFOR** facility. Following a **FOR** keyword the user places one replica of the desired sequence followed by an **ENDFOR** keyword. The job sequence is prepared using stand-in names in place of the effective names or constants on which the process is actually to be applied. The stand-in names come after the **FOR** keyword, each with the associated list of the actual names or constants which it represents.

The general format of the **FOR...ENDFOR** block is

```
$FOR  stand-in name_1 = list_1 ;
      stand-in name_2 = list_2 ;
      ............... = ...... ;
      stand-in name_k = list_k ;
            ......
            ...... sequence of HUDAP sections
            ...... including FOR...ENDFOR Blocks
            ......
$ENDFOR
```

The above represents the most general form of a **FOR...ENDFOR** sequence. The processing of a **FOR...ENDFOR** block is done by replacing each stand-in name with an actual name or constant from its associated list as follows: in the first processing of the sequence HUDAP will replace each stand-in name with the first name or constant from its list (That is, **stand-in name_1** = first item in list **1**, **... stand-in name_k** = first item in list **k**.) With these values, the sequence of HUDAP sections is then carried out. Then the stand-in names take on the values of the second items in each list, and the HUDAP sections are processed with these values, etc. Each list must therefore contain the same number of items.

A stand-in name must be different from all existing variable names in the file or HUDAP keywords and from all other stand-in names coded within the same **FOR...ENDFOR** block. Each stand-in name is followed by an equal sign, followed by the list of actual names or constants it represents; each list is followed by a semicolon. After the block has been processed the stand-in names are "forgotten" by the system; these names can be used again later in the job as new variable names or stand-in names.

Stand-in names can represent variables already existing in the file, or they can represent new variables. When the stand-in variable represents existing variables, the associated variable list may be of the usual **Vara Varb TO Vard Vare**

form; and when the stand-in variable represents new variables to be processed, the variable list may use the implicit **TO** convention: **X Y1 TO Y10 Z**. Example:

```
$FOR X = OYEHUDA TO ODAVID ;
     Y = NYEHUDA TO NDAVID ;
     Z = Z1 TO Z50 ;
         $COMPUTE
            Z = X - Y ;
$ENDFOR
```

Here **X**, **Y**, and **Z** are stand-in names, **OYEHUDA TO ODAVID** and **NYEHUDA TO NDAVID** are existing variables, while **Z1 TO Z50** are new variables. Note that from **OYEHUDA** to **ODAVID** and from **NYEHUDA** to **NDAVID** there must be **50** variables. The first computation will yield **Z1=OYEHUDA-NYEHUDA**, and the final computation will be **Z50=ODAVID-NDAVID**.

We saw above that a stand-in name can represent an existing or new variable. In fact, HUDAP allows various kinds of lists. A stand-in name can represent almost everything: constants, scalar variables, file names, HUDAP functions, HUDAP keywords, HUDAP section names, arithmetic operators, logical operators, and system scalar variables. This turns out the **FOR..ENDFOR** utility to be a powerful feature. Example:

```
$FOR V1 = ITEM1 ITEM3 ITEM5 ;
     V2 = ITEM2 ITEM4 ITEM6 ;
     SECTION = MONCO FREQ MONCO ;
         $SECTION  NAMES = V1 V2 ;
$ENDFOR
```

In this example **SECTION** is a stand-in name, and the above is equivalent to:

```
$MONCO  NAMES = ITEM1 ITEM2 ;
$FREQ   NAMES = ITEM3 ITEM4 ;
$MONCO  NAMES = ITEM5 ITEM6 ;
```

Sometimes one wants to vary an index between two constant bounds. For example:

```
$FOR  I = 1 TO 30 ;
```

But, suppose the constant "**30**" comes from the computation of a scalar variable. In that case, the user can specify the value of a scalar variable instead of the constant "**30**". The symbol "**~**" is used to specify the value of a scalar variable. Thus the value of **_A** is represented by **~A**. If **_A=30**, the above example becomes:

```
$FOR  I = 1 TO ~A ;
```

Note that the list **1 TO _A** is incorrect; it is not possible to mix different kinds of items (here constant with scalar variable). Note also that the list **_X TO _Z** is different from the list **~X TO ~Z**. In the first case the list is referenced by name,

that is, in the order that the scalar variables appear in the user job. In the second case the list is referenced by value, that is, **~Z-~X+1** serial values are requested: **~X**, **~X+1**, **~X+2**, ..., **~Z**.

FOR...ENDFOR structures can be nested within other **FOR...ENDFOR** structures up to 8 levels.

Let us give an example of data generation using 2 nested **FOR...ENDFOR** blocks:

```
$COMPUTE  _NCASES = 50 ;
$FOR  I = 1 TO ~NCASES ;
      $FOR VAR  = VAR1 TO VAR4 ;
           FUNC = SIN COS LOG EXP ;
           $COMPUTE  VAR[I] = FUNC(I) ;
      $ENDFOR
$ENDFOR
```

At line 1, the system scalar variable **_NCASES** (number of cases) is set to **50**. The external **FOR** at line 2, has one list of constant kind. At line 3, begins an internal **FOR** with two lists: the stand-in name "**VAR**" represents new variables, while the stand-in name "**FUNC**" represents mathematical functions. The line 5 is the "heart of the body". This **COMPUTE** will be processed 50*4 = 200 times.

In the first case (**I=1**) we will get **VAR1[1]=SIN(1)**, **VAR2[1]=COS(1)**, **VAR3[1]=LOG(1)**, and **VAR4[1]=EXP(1)**. The 50th case will yield **VAR1[50]** through **VAR4[50]**. Here the square brackets are used to index the variables. (Without this facility only the final values computed for **VAR1** through **VAR4** would be preserved, that is, the case **I=50**. Each of the preceding values that **VAR1** through **VAR4** had taken on would have been systematically replaced by the successive computation.) If we want to use these variables in later analyses or computations, or file them away (using the **OUTPUT** section) it is not necessary to use the square brackets.

The square brackets were only necessary for the stage of computation.

FOR...ENDFOR Utility

```
$FOR  stand-in name_1 = list_1 ;
      stand-in name_2 = list_2 ;
      ..................... = ...... ;
      stand-in name_k = list_k ;
           ......
           ...... sequence of HUDAP sections
           ...... including FOR...ENDFOR Blocks
           ......
$ENDFOR
```

name_i to the left side of the assignment is a stand-in index which can represent

_ constants
_ existing scalar variables
_ new scalar variables
_ values of existing scalar variables
_ existing (regular) variables
_ new (regular) variables
_ file names
_ HUDAP functions
_ HUDAP keywords
_ HUDAP section names
_ arithmetic operators
_ logical operators
_ system scalar variables

list_i is a list of the above kinds of items.

In the same **FOR** the different lists must have the same length.

The symbol "**~**" is used to specify the value of a scalar variable. Thus, the value of **_A** is **~A** Note that the list : **_A TO _B** is different from the list : **~A TO ~B**. In the first case the list is referenced by name, that is, in the order that the scalar variables appear in the job. In the second case the list is referenced by value, that is, (**~B-~A+1**) serial values are requested : **~A, ~A+1, ~A+2, ..., ~B**

The *INFO* Section

This section gives information on variables and parameters of the current data stored in HUDAP memory, like number of variables, number of cases, missing values, scalar variables, variable and category labels.

In particular, the information given by **INFO** section is helpful when one have to know the content of a HUDAP system file in order to use it correctly.

INFO section contains two commands. **REVERSE** command is useful when labels have to be reversed (hebrew for example). **NOREVERSE** command, which is the default, disables the previous command.

Follows an example of HUDAP job using **INFO** section.

```
$DATA
     NAMES = #2  BB6 6-7  B62 62  B63 63
                 B67 67  B72 72 ;
     FILE = 'C233.DAT' ;
     MISSINGS = BB6 TO B72  0 ;
     VARLABS = BB6 'Reading newspaper' B62 'Sex'
               B63 'Education (study years)'
               B67 'Age'
               B72 'Place of birth'
                   'Respondent/his father' ;
     CATLABS = BB6  4  'Yediot ahronot'
                    5  'Ma''ariv'
                    6  'Unknown'
               B62  1  'Men'  2  'Women'
               B63  1  '8 or less'  2  '9 to 11'
                    3  'Twelve'  4  '13 and above'
               B67  1  '20 to 29'  2  '30 to 39'
                    3  '40 to 49'  4  '50 to 59'
                    5  '60 and above'
               B72  1  'Israel/Israel'
                    2  'Israel/Asia-Africa'
                    3  'Israel/Europe-US'
                    4  'Both Asia-Africa'
                    5  'Both Europe-US' ;
$INFO
```

Output listing of the above job:

```
                  Information on current Data and Parameters
                  ------------------------------------------

Number of Variables .........    5
Number of Scalar Variables ..    3
Number of cases ............. 1199


Variable Name      Missing Values

BB6                  .0 SMV
B62                  .0 SMV
B63                  .0 SMV
B67                  .0 SMV
B72                  .0 SMV

**Note** 'SMV' stands for System Missing Value


List of scalar variables with their values

  _MISS      _NVARS     _NCASES
   .10E+39    .50E+01    .12E+04


Variable Name
              Variable Label(s)
                                        Category  Category Label

BB6
              Reading newspaper
                                             4.0  Yediot ahronot
                                             5.0  Ma'ariv
                                             6.0  Unknown

B62
              Sex
                                             1.0  Men
                                             2.0  Women

B63
              Education (study years)
                                             1.0  8 or less
                                             2.0  9 to 11
                                             3.0  Twelve
                                             4.0  13 and above

B67
              Age
                                             1.0  20 to 29
                                             2.0  30 to 39
                                             3.0  40 to 49
                                             4.0  50 to 59
                                             5.0  60 and above

B72
              Place of birth
              Respondent/his father
                                             1.0  Israel/Israel
                                             2.0  Israel/Asia-Africa
                                             3.0  Israel/Europe-US
                                             4.0  Both Asia-Africa
                                             5.0  Both Europe-US
```

INFO section menu

REVERSE ;
To reverse text of variable and category labels in the printout.

NOREVERSE ;
Disables the **REVERSE** command.
This is the default.

The *OUTPUT* Section

Often it is advantageous to store the results of a preliminary analysis or data transformation in such a way that will allow easy access for later analyses. The section **OUTPUT** allows you to create and save files containing such preliminary results.

Note: The section **OUTPUT** should not be confused with the paragraph **OUTPUT**, which appears within several sections as **MONCO**, **WSSA1**, **CRST** ....

The sentences **NAMES**, and **FILE**, are used for describing output data in this section in the same manner in which they are used for describing input data in the **DATA** section.

The following example illustrates a simple procedure in which data is inputted, recoded (using the **CODING** section), and outputted onto an external file using the **OUTPUT** section:

```
$DATA
     NAMES = A1 TO A8 1-8 ;
     FILE = 'DATA.1' ;
$CODING
     NAMES = A2 TO A5 ;
     REPLACE 0 TO 2 BY 1 | 3 TO 6 BY 2 | 7 TO 9 BY 3 ;
$OUTPUT
     NAMES = A1 TO A8 1-8 ;
     FILE = 'DATA.2' ;
```

In the above example variables **A1** through **A8** were inputted from the file **DATA.1**. Variables **A2** through **A5** are recoded (values of **0** through **2** are replaced by **1**, etc.) and the revised record as a whole (that is, the original values of **A1**, **A6**, **A7**, **A8** as well as the new values for **A2 - A5**) is written into a file named **DATA.2**.

OUTPUT section menu

NAMES = <[r] var list [c] [t]
.....
[r] var list [c] [t]> ;

The variables referenced in this sentence will be written on the output file according to the supplied specifications.
Square brackets in the syntax description mean optional specification.
"**r**" : record specifications. It can be:
#n for going to record number "n".
m**(/)** or **//.../** (m times) for skipping "m" records.
"**c**" : column specifications. The form is:
start_col**-**end_col where
start_col : starting column
end_col : ending column
"**t**" : format type specifications. It can be:
(d**)** for real variable with "d" decimal digits.
(A) for alphanumeric variable.
If "**t**" is missing, the variable is assumed to be integer.

FILE = '<char>' ;

"**char**" is the outfile pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

The *FREQ* Section

Overview

The **FREQ** section produces a table of frequency counts and percentages for values of individual variables. Variables can be of any type: integer, real or character. Optionally, one can obtain histograms for interval variables, univariate summary statistics and percentiles. One can prevent the printing of the frequency table, in the case of interval-level data, and request only statistics. Frequencies can be obtained form by means of **TOGETHER** command.

Operation

The procedure **FREQ** is invoked by the section name **FREQ**. Let's recall that **FREQ** has to be preceded by the section delimiter "**$**". 7 sentences (**NAMES**, **NOFR**, **HIST**, **MAXCAT**, **TOGETHER**, **NOPERC**, and **NOCUMUL**) and 1 paragraph (**STATS**) are available in this section. **NAMES** and **MAXCAT** sentences are assignment sentences. They are followed by an equal sign and specifications. **NOFR HIST TOGETHER NOPERC** and **NOCUMUL** sentences are command sentences. They are not followed by anything.

The above sentences and the paragraph **STATS** can be named in any order and are separated from each other by a semicolon. Each one can be used only once for **FREQ** section. The only required sentence is **NAMES** which specifies the variables being analysed. All other sentences such as the paragraph **STATS** are optional.

The NAMES sentence

The **NAMES** sentence names the variables to be analysed. It is the only required sentence. Following is a very simple HUDAP input command file requesting frequencies for 2 variables **SEX** and **EDUC** (education):

```
$DATA NAMES =   ID 6-8 SEX 9
             #2 ORIGIN 16 EDUC 18 V1 TO V7 19-25
                INCOME 26-31 (2) ;
      FILE = 'FDATA'   ;
      CATLABS = SEX  1 'Male' 2 'Female';
$FREQ NAMES = SEX EDUC;
```

Following is the table of **SEX** produced by the above command file.

```
  Frequency table for SEX     :
  ----------------------------

Category                  Category              Cum
  Name                     Value   Freq   Pct   Pct

Male                           1    338  50.37  50.37
Female                         2    333  49.63 100.00
                                  -- ------ ------
  Sums                           671 100.00 100.00


    Statistics for SEX
    ----------------------

N of valid cases=  671     N of missing cases=    0

  Frequency table for EDUC    :
  ----------------------------

Category                  Category              Cum
  Name                     Value   Freq   Pct   Pct

                               0      5   0.75   0.75
                               1      1   0.15   0.89
                               2     20   2.98   3.87
                               3     53   7.90  11.77
                               4    178  26.53  38.30
                               5    143  21.31  59.61
                               6     70  10.43  70.04
                               7     65   9.69  79.73
                               8    136  20.27 100.00
                                  -- ------ ------
  Sums                           671 100.00 100.00


    Statistics for EDUC
    ----------------------

N of valid cases=  671     N of missing cases=    0
```

The value labels are printed at the left of the table. Value labels were specified by the **CATLABS** sentence of the **DATA** section.

The **CATLABS** sentence is optional and can be omitted. In such a case, the left corner of the frequency table is empty.

The value labels are followed by the value and the number of cases that have the value. The percentage is based on all the observations, and the valid and cumulative percentages are based on those cases that have valid values. The number of valid and missing observations is also provided. The variable list

given in the **NAMES** sentences must contain, only existing variables named in the **DATA** section or computed by the **COMPUTE** section (or generated by **POSAC**).

NOFR command

This command, if given, disables printing of frequency table for all specified variables. If that command is not supplied, frequency tables are printed.

It is recommended to supply the **NOFR** sentence, if you are only interested by histograms or by statistics. This can be the case of interval-level variables.

HIST command

Print histogram for each variable specified in the **NAMES** sentence. The histogram can be printed on a width of 132 characters or on a width of 80 characters. The width of 80 characters is requested by giving the "**LINESIZE=80**' sentence in the **SET** section.

Example:

```
$SET  NAME IS 'THIS IS AN EXAMPLE';
      LINESIZE=80;
$DATA NAMES = ID ORIGIN SEX EDUC V1 TO V7 INCOME ;
$FREQ NAMES = INCOME ;
              NOFR ; HIST;
```

INCOME is in the above example an interval-level variable. The frequency table has no meaning. Then **NOFR** sentence is supplied. Each star in the histogram represents one or more cases, according to the linesize and to the maximal frequency.

If the linesize is equal to 132 and the maximal frequency is less or equal to 100, each star represents one case. If the maximal frequency is greater than 100 and less or equal 200, each star represents 2 cases, and so forth.

If the linesize is equal to 80 and the maximal frequency is less or equal to 60, each star represents one case. If the maximal frequency is greater than 60 and less or equal to 120, each star represents 2 cases, and so forth.

The MAXCAT sentence

In general frequencies are requested for variables with few categories. Therefore, in order to avoid mistakes, the number of categories for each variable was limited to 50. However, this limit can be modified by means of **MAXCAT** sentence.

For example :

```
MAXCAT = 100 ;
```

The TOGETHER command

This command allows the user to get frequencies as well as percentages, cumulative percentages and statistics in column form, one column per variable. In conjunction with this command there are two related commands: **NOPERC** and **NOCUMUL** to suppress printing of percentage or cumulative percentage tables.

Example:

```
$SET  LINESIZE=80 ;
$DATA NAMES =  A11 TO A20 11-20  #5 ;
      MISSINGS =  A11 TO A20  0   ;
      FILE = '\HUDAP\DATA\NOARB.504' ;
$FREQ NAMES=  A11 TO A18  ;
      TOGETHER ;
      STATS ALL ;
```

The paragraph STATS

Requests summary statistics for each of the variables named by the **NAMES** sentence. Each statistic is requested by a command such as: **MEAN**, **VAR**, **MED** etc... The commands are separated by slash.

Example:

```
$FREQ NAMES = V1 TO V5 INCOME ;
              NOFR ; HIST ;
      STATS MEAN/SD/VAR/MODE/SUM;
```

- **SD** to get standard deviation.
- **VAR** to get variance.
- **MODE** to get the mode.

The coefficient of variation **CV** is computed by: **CV**=**100*STDDEV/MEAN**. If the mean is zero, **CV** is set to **999.0**. The above **STATS** paragraph produces means, standard deviations, variances, modes and sums for the variables **V1 to V5** and **INCOME**.

Note that **MEAN**, **SD**, **VAR**, **MIN**, **MAX**, **SUM** and **CV** (coefficient of variation) are unavailable for character type.

STATS ALL produces mean, standard deviation, variance, median, mode, minimum, maximum, sum and coefficient of variation for integer and real variables. **STATS ALL** produces only median and mode for character type variables.

The PERCENTILES sentence:

This is a sentence of the **STATS** paragraph. It is the only one which is an assignment sentence and not a command. The keyword **PERCENTILES** or **PER** is followed by an equal sign and by a list of numbers between **0** and **1**. These numbers represent percentages of cases to be taken for each percentile. For example: **0.5** requests computing of median and **0.25** requests computing of the first quartile .

The percentile numbers have to be separated by commas or by blanks.

Example:

```
$FREQ NAMES = V1 TO V5 INCOME;
      STATS MODE/PER=0.1 0.25 0.5 0.75 0.9;
```

The above **STATS** paragraph requests the mode and the 5 percentiles defined respectively by 10, 50 and 90 the cases from the lowest value of each variable named in the **NAMES** sentence of the **FREQ** section.

Note that **STATS ALL** does not include the **PER**centile sentence.

Up to 10 percentile numbers may be requested in the **PER**centile sentence.

Percentiles can be requested for character type variables.

Printed output from section FREQ

```
------------------------------------------------------------
_  1_   $SET NAME='Values of youngs in Israel - 1974';
_  2_        LINESIZE = 80 ;
_  3_   $DATA NAMES=HARM1 TO HARM10 1-10 ;
_  4_         FILE='\DATABANK\NOAR.DAT';
_  5_         MISSINGS=HARM1 TO HARM10 0;
_  6_         VARLABS= HARM1 'Receiving bribes'
_  7_                  HARM2 'Damaging school property'
_  8_                  HARM3 'Use of drogues'
_  9_                  HARM4 'Not listening to police'
_ 10_                  HARM5 'Emigrating from Is
_ 11_                  HARM6 'copying in exams'
_ 12_                  HARM7 'Stealing from the rich'
_ 13_                  HARM8 'Supporting Palest. state'
_ 14_                  HARM9 'Stealing exams'
_ 15_                  HARM10 'Conversion'  ;
_ 16_         CATLABS= HARM1 TO HARM10
_ 17_                   1 'Extremely harmful'
_ 18_                   2 'very harmful'
_ 19_                   3 'slightly harmful'
_ 20_                   4 'not harmful at all';
_ 21_   $FREQ NAMES=HARM1 TO HARM10;
_ 22_         STATS ALL  /
_ 23_               PERCENTILES = 0.1 0.25 0.5 0.75 0.9  ;
------------------------------------------------------------


     ****************************

     ****************************



Number of variables ..........  10
Number of cases ..............  4612


  Frequency table for HARM1   : Receiving bribes
  ---------------------------

Category                 Category              Cum
  Name                    Value   Freq   Pct   Pct

Extremely harmful            1    671  15.13  15.13
very harmful                 2   1404  31.65  46.78
slightly harmful             3   1673  37.71  84.49
not harmful at all           4    688  15.51 100.00
                                   -- ------ ------
  Sums                          4436 100.00 100.00


    Statistics for HARM1
    ----------------------

N of valid cases= 4436     N of missing cases=  176
Mean=     2.53607   Variance=     0.86141   C.V.= 36.60
Standard Deviation=     0.92812   Sum= 11250.00000
Median=     3.00000   Min=     1.00000   Max=     4.00000
Modes=     3.00000
Percentiles=   1.00000   2.00000   3.00000   3.00000   4.00000
```

```
  Frequency table for HARM2   : Damaging school property
  ---------------------------

Category                  Category              Cum
  Name                     Value   Freq   Pct   Pct

Extremely harmful             1   1205  26.93  26.93
very harmful                  2   1888  42.19  69.12
sl
not harmful at all            4    247   5.52 100.00
                                  -- ------ ------
  Sums                           4475 100.00 100.00


    Statistics for HARM2
    ----------------------

N of valid cases= 4475     N of missing cases=  137
Mean=     2.09475   Variance=     0.73471   C.V.= 40.92
Standard Deviation=     0.85715   Sum=  9374.00000
Median=     2.00000   Min=     1.00000   Max=     4.00000
Modes=     2.00000
Percentiles=   1.00000   1.00000   2.00000   3.00000   3.00000

  Frequency table for HARM3   : Use of drogues
  ---------------------------

Category                  Category              Cum
  Name                     Value   Freq   Pct   Pct

Extremely harmful             1   3155  70.25  70.25
very harmful                  2    854  19.02  89.27
slightly harmful              3    309   6.88  96.15
not harmful at all            4    173   3.85 100.00
                                  -- ------ ------
  Sums                           4491 100.00 100.00


    Statistics for HARM3
    ----------------------

N of valid cases= 4491     N of missing cases=  121
       ..............................
       ..............................
```

Printed output from section FREQ with TOGETHER

```
-------------------------------------------------------------------------
_  1_     $set  linesize=80 ;
_  2_     $data names =  a11 to a20 11-20  #5 ;
_  3_           missings =  a11 to a20  0   ;
_  4_           file = '\databank\noarb.504' ;
_  5_     $freq names=  a11 to a18  ;
_  6_           together ;
_  7_           stats all ;
-------------------------------------------------------------------------
     *********************************
          F R E Q U E N C I E S
     *********************************
```

```
Number of variables .........   8
Number of cases .............  100

                                Frequencies

                a       a       a       a       a       a       a       a
                1       1       1       1       1       1       1       1
                1       2       3       4       5       6       7       8
Ca
       1 |      6      10      13      23      35      33      58       9
       2 |     27      48      32      48      46      35      33      26
       3 |     46      32      32      21      10      20       4      49
       4 |     18       5      16       5       5       5              10
         +-------------------------------------------------------------
   Total       97      95      93      97      96      93      95      94

                               Percentages

                a       a       a       a       a       a       a       a
                1       1       1       1       1       1       1       1
                1       2       3       4       5       6       7       8
Category +-------------------------------------------------------------
       1 |   6.19  10.53  13.98  23.71  36.46  35.48  61.05   9.57
       2 |  27.84  50.53  34.41  49.48  47.92  37.63  34.74  27.66
       3 |  47.42  33.68  34.41  21.65  10.42  21.51   4.21  52.13
       4 |  18.56   5.26  17.20   5.15   5.21   5.38         10.64
         +-------------------------------------------------------------
   Total  100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00

                          Cumulative Percentages

                a       a       a       a       a       a       a       a
                1       1       1       1       1       1       1       1
                1       2       3       4       5       6       7       8
Category +-------------------------------------------------------------
       1 |   6.19  10.53  13.98  23.71  36.46  35.48  61.05   9.57
       2 |  34.02  61.05  48.39  73.20  84.38  73.12  95.79  37.23
       3 |  81.44  94.74  82.80  94.85  94.79  94.62 100.00  89.36
       4 | 100.00 100.00 100.00 100.00 100.00 100.00        100.00
         +-------------------------------------------------------------


                               Statistics

                a       a       a       a       a       a       a       a
                1       1       1       1       1       1       1       1
                1       2       3       4       5       6       7       8
         +-------------------------------------------------------------
    Mean |   2.78   2.34   2.55   2.08   1.84   1.97   1.43   2.64
 Std Dev |    .82    .74    .94    .81    .81    .89    .58    .80
Variance |    .67    .54    .88    .66    .66    .79    .33    .64
 Minimum |      1      1      1      1      1      1      1      1
 Maximum |      4      4      4      4      4      4      3      4
  Median |   3.00   2.00   3.00   2.00   2.00   2.00   1.00   3.00
    Mode |      3      2      2      2      2      2      1      3
     Sum |    270    222    237    202    177    183    136    248
         +-------------------------------------------------------------
```

FREQ section menu

NAMES = <var list> ;
Names of the variables for which frequencies and/or statistics are computed.

NOFR ;
Disables printing of frequency table for all specified variables.

HIST ;
Print histogram for each variable specified in **NAMES** sentence.

MAXCAT = <value> ;
Maximal number of categories allowed for all variables listed in **NAMES** sentence. If for some variable the number of categories exceeds "**value**", the frequency table for this variable is not printed.
The default for "**value**" is 50.

TOGETHER ;
To get frequencies, percentages, cumulative percentages and statistics in a condensed form.

NOPERC ;
To suppress percents table when **TOGETHER** is used.

NOCUMUL ;
To suppress cumulative percents table when **TOGETHER** is used.

STATS paragraph

Paragraph defining various statistics.

MEAN /
Print mean for each variable specified in **NAMES** sentence. Unavailable for character type.

SD /
Print standard deviation for each variable specified in **NAMES** sentence. Unavailable for character type.

VAR /
Print variance for each variable specified in **NAMES** sentence. Unavailable for character type.

MED /
Print median for each variable specified in **NAMES** sentence.

MODE /
Print modes for each variable specified in **NAMES** sentence.

MIN /
Print minimum for each variable specified in **NAMES** sentence. Unavailable for character type.

MAX /
Print maximum for each variable specified in **NAMES** sentence. Unavailable for character type.

SUM /
Print sum for each variable specified in **NAMES** sentence. Unavailable for character type.

CV /
Print Coefficient of Variation for each variable as above. Unavailable for character type.

ALL /
Print all the above statistics.

PERcentiles = <val list> /
Print percentiles for each variable specified in **NAMES** sentence. "**val list**" are numbers between 0 and 1 representing percentages of cases to be taken for each percentile. For example: 0.5 requests computing of median, 0.25 requests computing of the first quartile ... Up to 10 percentiles may be requested.

The *CRST* Section

Introduction

The **CRST** section produces tables containing joint distribution of two variables that have a limited number of distinct values. The frequency distribution of one variable is subdivided according to the values of the other variable. The combination of one value of the first variable with one value of the second value, defines a cell - the basic element of **CRST** tables.

In addition to the tables of cell counts, you can obtain tables of cell percentages and optional measures of association between the two variables, (Monotonicity coefficient, Pearson correlation coefficient, Chi-square which can be considered as a distance coefficient, Dependency coefficient which can be also considered as a distance, with that difference that this is a normalised distance comprised between 0 and 1). A matrix of these coefficients can be produced on a file or in memory for later use. That matrix can be supplied, for example, to a **WSSA1** section as input matrix.

CRST can handle integer, real and alphanumeric variables. But **CRST** cannot compute Monotonicity and Pearson coefficients between alpha variables.

Operation

The procedure **CRST** is invoked by the section name **CRST**. Let's recall that **CRST** has to be preceded by the section delimiter "**$**". 8 sentences (**COLNAMES**, **ROWNAMES**, **NAMES**, **NOFR**, **RPCT**, **CPCT**, **TPCT, MAXCAT**) and 2 paragraphs (**STATS** and **OUTPUT**) are available in this section. **COLNAMES**, **ROWNAMES**, **NAMES** and **MAXCAT** sentences are assignment sentences. They are followed by an equal sign and specifications. **NOFR**, **RPCT**, **CPCT**, and **TPCT** sentences are command sentences. They are not followed by anything.

These above sentences and the paragraphs **STATS** and **OUTPUT** can be named in any order and are separated from each other by a semicolon. Each one can be used only once for **CRST** section. The only required sentences are either **NAMES** or **COLNAMES** and **ROWNAMES** which specifies the variables being analysed. All other sentences such as the paragraph **STATS** are optional.

The COLNAMES sentence

The **COLNAMES** sentence names the variables whose categories will define the columns in the cross table. The **COLNAMES** sentence must be given together with the **ROWNAMES** sentence. The variables given after the **COLNAMES** sentence are crossed with all variables defined in **ROWNAMES** sentence. The syntax is :

COLNAMES = <var list> ;

Example :

```
COLNAMES = Race Weight ;
```

The ROWNAMES sentence

In this sentence a variable list can be specified. The categories of each variable of the list will constitute the rows of the output. For each of these variables we obtain one table, where the categories in columns are supplied by each variable of the list given after the **COLNAMES** sentence. The syntax is :

ROWNAMES = <var list> ;

Example :

```
ROWNAMES = Sex Income Education Age Origin Item5 TO Item10 ;
```

The NAMES sentence

The **NAMES** sentence specifies a variable list for which one table will be produced for each pair of variables.

For example:

```
NAMES = SEX INCOME EDUCATION AGE ORIGIN ;
```

One cross table will be produced between **SEX** and **INCOME**, one between **SEX** and **EDUCATION**, one between **SEX** and **AGE**, one between **SEX** and **ORIGIN**, one between **INCOME** and **EDUCATION**, one between **INCOME** and **AGE**, one between **INCOME** and **ORIGIN**, one between **EDUCATION** and **AGE**, one between **EDUCATION** and **ORIGIN**, and the last between **AGE** and **ORIGIN**.

If sentence **NAMES** is given, the couple **COLNAMES** and **ROWNAMES** can't be given, and inversely. We have, then, 2 possible ways of running **CRST** section. The way of **NAMES** sentence is needed if you want to output matrices of coefficients (**CHISQ**, **MONCO**, **DEPCO**, **PEARSON**) between the variables, by means the **OUTPUT** paragraph.

Following is a simple input command file requesting cross frequencies for the variables **SEX** and **ORIGIN** and for the variables **SEX** and **EDUC** (education) :

```
$DATA NAMES = ID 1-4 SEX ORIGIN EDUC 5-10
              V1 TO V7 11-17 INCOME 23-30 (2) ;
      FILE='DATAFILE';
$CRST COLNAMES = SEX ; ROWNAMES = ORIGIN EDUC ;
```

If we change the last line by the following:

```
$CRST NAMES = SEX ORIGIN EDUC ;
```

then, 3 tables will be produced.

NOFR command

This command, if given, disables printing of frequency table for all specified variables. If that command is not supplied, frequency tables are printed. It is recommended to supply the **NOFR** sentence, if you are only interested by histograms or by statistics. This can be the case of interval-level variables.

RPCT command

This command requests to print tables of percent of the row totals. Such a table is produced for each couple of variables which have to be processed by **CRST**.

In the following example:

```
$CRST NAMES = SEX ORIGIN EDUC ;  NOFR;  RPCT;
```

three tables of row percentages are produced, one for each pair of variables. Tables of frequencies are not printed.

CPCT command

This command requests to print tables of percent of the column totals. Such a table is produced for each couple of variables which have to be processed by **CRST**.

In the following example:

```
$CRST NAMES = SEX ORIGIN EDUC ;  NOFR;  CPCT;
```

three tables of column percentages are produced, one for each pair of variables. Tables of frequencies are not printed.

TPCT command

This command requests to print tables of percent of the total frequency. Such a table is produced for each couple of variables which have to be processed by **CRST**.

In the following example:

```
$CRST NAMES = SEX ORIGIN EDUC ;  NOFR;  RPCT;  TPCT;
```

one table of row percents and one table of percents of the total frequency are produced for each pair of variables. Since there are 3 couples of variables in our example, then 6 tables were produced. Tables of frequencies are not printed.

The MAXCAT sentence

In general crosstabulations are requested for variables with few categories. Therefore, in order to avoid mistakes, the number of categories for each variable was limited to 50. However, this limit can be modified by means of **MAXCAT** sentence.

For example :

```
MAXCAT = 100 ;
```

The paragraph STATS

This paragraph requests coefficients of association for each couple of variables named by the **NAMES** sentence or by the pair of sentences: **COLNAMES** and **ROWNAMES**. The coefficients of association available are: Weak monotonicity, Chi square, Dependency coefficient, Pearson correlation. Each coefficient is requested by a command such as: **MONCO**, **CHISQ**, **DEPCO**, **PEARSON**. The commands are separated by a slash.

Example:

```
$CRST NAMES = SEX ORIGIN EDUC ; TPCT ;
      STATS CHISQ / DEPCO ;
```

One table of frequencies followed by one table of percents of the total frequency are produced for each pair of variables. Then, for each such couple of variables, a Chi-square and two dependency coefficients (variable1/variable2 and variable2/variable1) are produced. The significance for Chi-square is not computed in order to prevent the common misinterpretation and misuse of significance tests. Chi-square itself is here produced for historical reasons only. But it can be considered as a distance between the 2 variables and supplied in a matrix form to the **WSSA1** section (see below the **OUTPUT** paragraph).

The paragraph OUTPUT

This paragraph is available only if **NAMES** sentence is used (instead of **COLNAMES** and **ROWNAMES** sentences). In such a case, a squared matrix of monotonicity, pearson or dependency coefficients (**MATRIX** sentence) can be written either on a file (**FILE** sentence) or into memory (**MEMORY** command). Follow the different sentences and commands:

MATRIX sentence

This sentence specifies what matrix to output, coefficients of monotonicity, pearson or dependency. Only one matrix can be chosen. The syntax is:

```
MATRIX = <MONCO | PEARSON | DEPCO> /
```

FILE sentence

If this sentence is submitted, the matrix specified in MATRIX sentence is written on a file. The syntax is:

```
FILE = '<char>' /
```

"**char**" is the file pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

FORMAT sentence

This sentence is optional and meaningful if **FILE** sentence is submitted. The syntax is:

```
FORMAT = '<char>' /
```

The format "**char**" specifies the layout (in fixed fields) of the output in a single row. The usual FORTRAN rules for the format are applicable, but only the F specification is allowed. "**char**" cannot exceed 80 characters. The default is (13F6.2).

MEMORY command

The resulting effect of this command is a transfer of the matrix into an area in the memory, together with the variable names for which **CRST** (with **NAMES** sentence) was evoked. This means that any matrix processing procedure (as **WSSA1** for example) in the same job will be able to access the matrix and recognise the variable names. The syntax is:

```
MEMORY /
```

PRINT command

This command allows the printing of the matrix on the output listing. The syntax is:

```
PRINT /
```

CRST section menu

COLNAMES = <var list> ;
Names of variables that define the column categories.

ROWNAMES = <var list> ;
Names of variables that define the row categories.

NAMES = <var list> ;
One table is produced for each pair of variables. A matrix is produced which gives the value of the connection (by the **STATS** paragraph) between each variable and the rest of the variables in the list. The above sentences are 2 possible ways of running **CRST** section. One way is to give **COLNAMES** and **ROWNAMES** sentences together. In such a case **OUTPUT** paragraph can't be supplied. Another way is to give the **NAMES** sentence, and in such a case, output of matrices of coefficients (**CHISQ**, **MONCO**, **PEARSON**) can be requested by means of the **OUTPUT** paragraph. But, one and only one of them is necessary.

NOFR ;
Disables printing of frequency table for all variables specified

RPCT ;
Tables of percents of the row totals are produced.

CPCT ;
Tables of percents of the column totals are produced.

TPCT ;
Tables of percents of the total frequency in the tables are produced.

MAXCAT = <value> ;
Maximal number of categories allowed for all variables listed in **COLNAMES**, **ROWNAMES** or **NAMES** sentences. If for some variable the number of categories exceeds "**value**", the crossfrequency table for this variable with any other is not printed.
The default for "**value**" is 50.

STATS paragraph

Paragraph defining various statistics.

MONCO /
Print monco between each column variable and the common row variable.

CHISQ /
Print chi-square between each column variable and the common row variable.

DEPCO /
Print Dependency coefficient which expressed how much the column variable is dependent on the row variable.

PEARSON /
Print Pearson correlation between each column variable and the common row variable.

ALL /
Print all the above statistics.

OUTPUT paragraph

Paragraph defining the output of the **MONCO**, **PEARSON** or **DEPCO** coefficients matrix.

MATRIX = <MONCO | PEARSON | DEPCO> /
Specifies one of the 3 matrices: **MONCO**, **PEARSON** or **DEPCO**

FILE = '<char>' /
"**char**" is the matrix file pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

FORMAT = '<char>' /
The format **char** specifies the layout (in fixed fields) of the matrix in a single row . The usual FORTRAN rules for the format are applicable, but only the F specification is allowed. "**char**" cannot exceed 80 characters.
The default is (8F10.3).

MEMORY /
The resulting matrix is written into the memory, in order to use it as input of another section (as **WSSA1** for example) in the same job.

PRINT /
The resulting matrix is printed on the output file.

The *MULTABS* Section

Introduction

Users are often called upon to produce a variety of reports for business, schools, hospitals, and other organisations. These reports may contain tables of descriptive statistics like frequencies or joint distributions. The **MULTABS** section is a flexible formatter to get cross-tabulations between one variable (columns) and a series of variables (rows).

A **MULTABS** output has a basic structure that you can modify with a variety of sentences or commands. The body of the report is formatted in rows and columns where the columns correspond to the categories of one variable and the rows to categories of many variables.

To format the output, **MULTABS** provides full default specifications, but allows you to control:

_ page lengths and margins
_ labels for variables and category variables
_ printing direction (English or Hebrew)

Some of the above control specifications must be supplied in other sections like **SET** for page length or **DATA** for labels. Indeed, these specifications are also common to other procedures.

Specifying variables

The user specifies the desired tables by means of **COLNAMES** and **ROWNAMES** sentences. The variables referenced in these sentences may be previously defined in **NAMES** sentence of **DATA** section, or computed in **COMPUTE** section.

The COLNAMES sentence

In the **COLNAMES** sentence the user specifies the variables whose categories will define the columns in the table. The **COLNAMES** sentence must be given together

with the **ROWNAMES** sentence. The variables given after the **COLNAMES** sentence are crossed with all variables defined in **ROWNAMES** sentence. The syntax is:

```
COLNAMES = <var list> ;
```

Example:

```
COLNAMES = Race Weight ;
```

The ROWNAMES sentence

In this sentence a variable list can be specified. The categories of each variable of the list will constitute the rows of the output. The syntax is :

```
ROWNAMES = <var list> ;
```

Example:

```
ROWNAMES = Sex Income Education Age Origin Item5 TO Item10 ;
```

Different kinds of tables

MULTABS can print the frequencies (joint distribution) themselves, but also the frequencies as row percentages, column percentages or percentages of the total frequency. However, only one of these options can be obtained in a single **MULTABS** section.

In order to understand the content of the different tables, let us give some computational definitions for one table (i.e. crosstabulation of the column variable with one row variable) with *n* columns and *m* rows:

Frequency in cell (*i*,*j*): f_{ij}

Row marginal frequency: $r_i = \sum_{j=1}^{n} f_{ij} \qquad i = 1,\ldots,m$

Column marginal frequency: $c_j = \sum_{i=1}^{m} f_{ij} \qquad j = 1,\ldots,n$

Total frequency: $F = \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij}$

The FREQ command

This command requests to print the observed frequencies:

$$f_{ij}$$

This is the cross-tabulations between the column variable and the row variables. The marginal frequencies on the column and row variables are also given (see Figure 1.). **FREQ** is the default of **MULTABS** procedure.

The RPCT command

This command requests to print the percentages of the total row marginal frequencies:

$$\frac{100 f_{ij}}{r_i}$$

However, the marginal frequencies of row variables are also given (see Figure 2.)

The CPCT command

This command requests to print the percentages of the total column frequencies:

$$\frac{100 f_{ij}}{c_j}$$

However, the marginal frequencies of column variable are also given (see Figure 3)

The TPCT command

This command requests to print the percentages of the total frequency:

$$\frac{100 f_{ij}}{F}$$

However, the marginal frequencies of column and row variables are also given (see Figure 4)

Note: In the following figures, the character "f" represents a frequency while "p" represents a percentage.

	Column variable label				
	Cat. label	Cat. label	Cat. label	Cat. label	Total
Total	f	f	f	f	f
Row variable label					
Category label	f	f	f	f	f
Category label	f	f	f	f	f
Category label	f	f	f	f	f
To					
Row variable label					
Category label	f	f	f	f	f
Category label	f	f	f	f	f
Category label	f	f	f	f	f
Total	f	f	f	f	f
Row variable label					
Category label	f	f	f	f	f
Category label	f	f	f	f	f
Category label	f	f	f	f	f
Total	f	f	f	f	f

Figure 1
Page Layout of **FREQ** option

	Column variable label					
	Cat. label	Cat. label	Cat. label	Cat. label	Total	N
Total	p	p	p	p	100	f
Row variable label						
Category label	p	p	p	p	100	f
Category label	p	p	p	p	100	f
Category label	p	p	p	p	100	f
Row variable label						
Category label	p	p	p	p	100	f
Category label	p	p	p	p	100	f
Category label	p	p	p	p	100	f
Row variable label						
Category label	p	p	p	p	100	f
Category label	p	p	p	p	100	f
Category label	p	p	p	p	100	f

Figure 2
Page Layout of **RPCT** option

```
                         Column variable label
____________________________________________________________________
                           Cat.   Cat.   Cat.   Cat.
                           label  label  label  label     Total
N                              f      f      f      f         f

Row variable label
Category label                 p      p      p      p         p
Category label                 p      p      p      p         p
Category label                 p      p      p      p         p
Category label             ____p  ____p  ____p  ____p     ____p
Total                        100    100    100    100       100
N                              f      f      f      f         f

Row variable label
Category label                 p      p      p      p         p
Category label                 p      p      p      p         p
Category label                 p      p      p      p         p
Category label             ____p  ____p  ____p  ____p     ____p
Total                        100    100    100    100       100
N                              f      f      f      f         f

Row variable label
Category label                 p      p      p      p         p
Category label                 p      p      p      p         p
Category label                 p      p      p      p         p
Category label             ____p  ____p  ____p  ____p     ____p
Total                        100    100    100    100       100
N                              f      f      f      f         f
____________________________________________________________________
```

Figure 3
Page Layout of **CPCT** option

	Column variable label					
	Cat. label	Cat. label	Cat. label	Cat. label	Total	N
Total	p	p	p	p	100	f
Row variable label						
Category label	p	p	p	p	p	f
Category label	p	p	p	p	p	f
Category label	p	p	p	p	p	f
Total	p	p	p	p	100	f
N	f	f	f	f	f	
Row						
Category label	p	p	p	p	p	f
Category label	p	p	p	p	p	f
Cate				p	p	f
Total	p	p	p	p	100	f
N	f	f	f	f	f	
Row						
Category label	p	p	p	p	p	f
Category label	p	p	p	p	p	f
Cate						f
Total	p	p	p	p	100	f
N						

Figure 4
Page Layout of **TPCT** option

Example of MULTABS job with output

Input user directives :

```
$DATA
     NAMES = #2  BB6 6-7  B62 62  B63 63
                 B67 67  B68 68  B72 72 ;
     FILE = 'C233.DAT' ;
     MISSINGS = BB6 TO B72  0 ;
     VARLABS = BB6 'Reading newspaper' B62 'Sex'
               B63 'Education (study years)'
               B67 'Age' B68 'Income level'
               B72 'Place of birth'
                   'Respondent/his father' ;
     CATLABS = BB6  4  'Yediot ahronot'
                    5  'Ma''ariv'
                    6  'Unknown'
               B62  1  'Men'  2  'Women'
               B63  1  '8 or less'  2  '9 to 11'
                    3  'Twelve'  4  '13 and above'
               B67  1  '20 to 29'  2  '30 to 39'
                    3  '40 to 49'  4  '50 to 59'
                    5  '60 and above'
               B68  1  'Low'  2  'Average'  3  'High'
               B72  1  'Israel/Israel'
                    2  'Israel/Asia-Africa'
                    3  'Israel/Europe-US'
                    4  'Both Asia-Africa'
                    5  'Both Europe-US' ;
$CODING  NAMES = BB6  ;
         REPLACE 0 TO 3 6 TO 10 BY 6 ;
$CODING  NAMES = B63 ;
         REPLACE 1 2 3 BY 1 | 4 5 BY 2 |
                 6 BY 3 | 7 8 BY 4 ;
$CODING  NAMES = B67 ;
         REPLACE 1 2 BY 1 | 3 4 BY 2 | 5 6 BY 3 |
                 7 8 BY 4 | 9 BY 5 ;
$CODING  NAMES = B68 ;
         REPLACE 1 2 3 BY 1 | 4 5 6 BY 2 |
                 7 8 9 BY 3 ;
$CODING  NAMES = B72 ;
         REPLACE 6 BY 0 ;
$MULTABS COLNAMES =   BB6 ;
         ROWNAMES =   B62 TO B72 ;
         RPCT ;
```

In the **DATA** section, beside the sentences describing the variables (**NAMES**, **FILE**, **MISSINGS**), there are 2 sentences related with **MULTABS** section: **VARLABS** and **CATLABS**. These sentences attribute labels to the variables and their categories (see chapter on **DATA** section).

The various **CODING** sections come to recode some of the variables.

Finally, **MULTABS** section is requested with **RPCT** option (see Figure 2.).

Printed output :

	Yediot ahronot	Ma'ariv	Unknown	Total	N
Tota					
Sex					
Men					
Wome					
Educ					
8 or					
9 to 11	49	13	37	100	247
Twelve	45	18	37	100	362
13 and above	33	25	42	100	374
Age					
20 to 29	52	11	37	100	303
30 to 39	50	19	31	100	326
40 to 49	39	23	38	100	185
50 to 59	33	20	48	100	163
60 and above	23	20	57	100	218
Inco					
Low	36	12	52	100	368
Average	47	19	34	100	443
High	40	25	35	100	213
Place of birth					
Respondent/his father					
Israel/Israel	54	21	25	100	80
Israel/Asia-Africa	60	10	31	100	218
Israel/Europe-US	43	23	34	100	216
Both Asia-Africa	41	11	47	100	276
Both Europe-US	29	23	48	100	392

More aspects on MULTABS formatting

The basic layout for each of the 4 options (**FREQ**, **RPCT**, **CPCT**, **TPCT**) is as indicated in Figures 1 to 4. However, some details are needed to understand the format of the printed output in order to control it.

The BLANK sentence

MULTABS breaks the category labels of the column variable into words and prints separately each word on a different line. Two words are separated at least by one blank. In the above example, the category **4** of variable **BB6** was assigned label **Yediot ahronot**. On the printout this label is printed on 2 lines.

The wide of the label category column is computed according to the category labels of the column variable. Namely, it is the length of the maximal word

among all the words contained in all category labels of the column variable. In the above example, the maximal word among **Yediot**, **ahronot**, **Ma'ariv** and **Unknown** is **ahronot** (also **Ma'ariv** or **Unknown**); therefore, the wide of the label category column is set to 7. However, the minimal wide of the column is 5 even if all the words are less than 5.

We saw, that blank is a separator for **MULTABS** between two words in order to break the category labels of the column variable and print each word on a different line. But sometimes the words are small and the user may want to print two or more words on the same line. This can be done by inserting the default blank character "**&**" between two words. This character will be printed as a blank. For example, in the above job, suppose that instead of **Unknown** in category **6** of **BB6**, we have **My good News**, and we want to print **My good** on the same line. The job may appear:

```
$DATA

     ..............
     CATLABS = BB6  4  'Yediot ahronot'
                    5  'Ma''ariv'
                    6  'My&good News'
     ..............
     ..............
```

Corresponding printed output :

```
                          Reading newspaper
_____________________________________________________________________
                            Yediot           My good
                            ahronot Ma'ariv  News     Total     N
                                 42      18       41     100   1199

                       ...................
```

If the default blank character "**&**" is part of the user labels, the user can define its own blank character by means of **BLANK** sentence. in **MULTABS** section as in:

```
$DATA
     ..............
     ..............
     CATLABS = BB6  4  'Yediot ahronot'
                    5  'Ma''ariv'
                    6  'My~good News'
     ..............
     ..............
$MULTABS COLNAMES =   BB6 ;
         ROWNAMES =   B62 TO B72 ;
         RPCT ;  BLANK = ~ ;
```

The blank character can be used also to centre titles. For example, in our job, the following **CATLABS**:

```
     CATLABS = BB6  4  '&Yediot ahronot'
                    5  'Ma''ariv'
                    6  'My&good &&News'
```

will produce the following printout:

```
                                 Reading newspaper
_______________________________________________________________________________
                                     Yediot         My good
                                    ahronot Ma'ariv  News    Total      N
Total                                    42      18       41    100   1199

                              ...................
                              ...................
```

The TOP sentence

A table of MULTABS is printed at the beginning of the physical page. However, the user can choose his own margin in number of lines from the top of the physical page. The syntax is:

`TOP = <value> ;`

The BOTTOM sentence

Like **`TOP`** sentence, the **`BOTTOM`** sentence defines the number of lines to be left at the bottom of the page. The syntax is:

`BOTTOM = <value> ;`

The SKIP sentence

This sentence defines the number of lines to be skipped before printing label of next row variable. The syntax is:

`SKIP = <value> ;`

The default is 1.

The CONCAT command

The fields corresponding to the categories of the column variables are separated by a space (default). One can use the **`CONCAT`** command to suppress this space. In such a way, the category labels are concatenated. This is useful for building labels of different levels from the basic category labels.

The HEBREW command

By default, the printout of the tables is as shown in Figures 1,2,3 and 4. This is good for English labels. When the labels are in Hebrew, the **HEBREW** command allows a right to left printing direction, such that the labels of the row variables appear at the right hand of the tables.

The NOHEAD command

When specified, **NOHEAD** command suppress printing of **MULTABS** header and "Number of cases" line, such that the printout begins "cleanly" with the tables. This is useful when section **MULTABS** has to be repeated for different parameters via the **FOR..ENDFOR** feature.

The DECIMAL sentence

By default, percentages are rounded to integer values.. However, the user can decide to get the percentages with a specific number of digits after the decimal point. The syntax is:

```
DECIMAL = <value> ;
```

where "**value**" must lie between 0 and 3.

The MARGIN sentence

The syntax is:

```
MARGIN = <value> ;
```

This sentence allows the shifting of the output to the right by "**value**" spaces.

MULTABS section menu

COLNAMES = <var list> ;
Names of variables that define the column categories.

ROWNAMES = <var list> ;
Names of variables that define the row categories.

FREQ ;
Tables of frequencies are produced.
This is the default.

RPCT ;
Tables of percents of the row totals are produced.

CPCT ;
Tables of percents of the column totals are produced.

TPCT ;
Tables of percents of the total frequency in the tables are produced.

TOP = <value> ;
Top margin in number of lines from the top of the physical page.
The default is 0.

BOTTOM = <value> ;
Bottom margin in number of lines from the bottom of the physical page.
The default is 0.

BLANK = <char> ;
"**char**" indicates a character in the user labels which will be printed as blank.
This is useful for control formatting of the labels, in particular for words which must not be broken.
The default blank character is **&**.

SKIP = <value> ;
Number of lines to skip before printing label of row variable.
The default is 1.

CONCAT ;
Concatenation of the category labels of the column variable.
The default is no concatenation.

HEBREW ;
Right to left printing of the output.
The default is left to right printing.

NOHEAD ;
Suppression of the job header of **MULTABS**. This option is useful when section **MULTABS** is inside a **FOR** ... **ENDFOR** block.

DECIMAL = <value> ;
Percents are printed with "**value**" digits after the decimal point. "**value**" must lie between 0 and 3.
The default for "**value**" is 0.

MARGIN = <value> ;
To shift the output to the right by "**value**" spaces.

NON ;
Suppresion of marginal frequency.

N ;
Printing of marginal frequency.
This is the default

NOUNDERL ;
Suppresion of underlines in tables.

UNDERL ;
Printing of underlines in tables.
This is the default

The *MONCO* Section

Introduction

The **MONCO** Section computes Guttman weak monotonicity coefficients for pairs of ordinal or interval variables.

Definition

Given two numerical variables x and y, the weak coefficient of monotonicity μ_2 tells us how much the two variables vary in the same sense. In other words, when x increases does y increases or not. See Appendix A for a mathematical description.

μ_2 as a correlation coefficient

The weak monotonicity coefficient between two variables x and y is in some sense a correlation coefficient between these variables. Indeed, the concept of correlation does not necessarily depend on the concept of regression. For the weak monotonicity coefficient varies between **-1** and **+1**, reaching these extreme values for perfect monotonicity, without knowledge of the function relating the two variables nor use of the means of the variables. The coefficient μ_2 belongs to a large family of monotonicity coefficients called 'Regression-free coefficients of monotonicity'. There are coefficients of strong, semi-strong (semi-weak) and weak monotonicity. Goodman-Kruskal's gamma, Kendall's tau and Spearman's rho are special cases of this family.

The regression-free coefficients of monotonicity are useful in data analysis of time-series, and are also the basis for non metric data analysis programs, like **WSSA1** (Weighted Smallest Space Analysis).

Advantages of the Weak Monotonicity Coefficient

The usefulness of the coefficient μ_2 appears clearly when μ_2 is compared with the Pearson correlation coefficient. Indeed, the Pearson correlation coefficient for dichotomies depends, to a large extent, on the marginal distributions. This is also

true, to some extent, when the variables have three or four categories. Whereas the Pearson correlation coefficient specifies a linear regression, the weak monotonicity coefficient μ_2 doesn't assume anything about the exact nature of the regression function. The purpose of μ_2 is to indicate to what extent values in one variable increase (decrease) monotonously with increases in other variables.

When Pearson's correlation coefficient equals **+1** or **-1** , the weak monotonicity coefficient μ_2 for the same data will have the same value. In other cases, the absolute value of μ_2 will be higher than that of Pearson's coefficient.

Operation

The procedure **MONCO** is invoked by the section name **MONCO**. 4 sentences (**NAMES**, **COLNAMES**, **ROWNAMES**, **MATRIX**) and one paragraph (**OUTPUT**) are available in this section.

These above sentences and the paragraph **OUTPUT** can be named in any order and are separated from each other by a semicolon. Each one can be used only once for **MONCO** section. The only required sentences are either **NAMES** or **COLNAMES** and **ROWNAMES** which specifies the variables being analysed.

The NAMES sentence

The **NAMES** sentence specifies a variable list for which coefficients are produced for each pair of variables in a squared matrix form.

For example:

```
NAMES = Item1 to Item20 ;
```

If sentence **NAMES** is given, the couple **COLNAMES** and **ROWNAMES** can't be given, and inversely. We have, then, 2 possible ways of running **MONCO** section. The way of **NAMES** sentence is needed if you want to output a squared matrix of coefficients between the variables.

Following is a simple input command file requesting monotonicity coefficients between variables **EDUC**, **V1** to **V7** and **INCOME**:

```
$DATA NAMES = ID 1-4 SEX ORIGIN EDUC 5-10
              V1 TO V7 11-17 INCOME 23-30 (2) ;
      FILE='DATAFILE';
$MONCO NAMES = EDUC TO INCOME ;
```

If we change the last line by the following:

```
$MONCO COLNAMES = V1 TO V7 ;
       ROWNAMES = EDUC INCOME ;
```

then, a rectangular matrix will be produced.

The COLNAMES and ROWNAMES sentences

These sentences are useful when a rectangular matrix of coefficients is required. The **COLNAMES** sentence names the variables defining the columns in the matrix, while the variables listed in **ROWNAMES** sentence define the rows. The **COLNAMES** sentence must be given together with the **ROWNAMES** sentence. The variables given after the **COLNAMES** sentence are crossed with all variables defined in **ROWNAMES** sentence. The syntax is :

```
COLNAMES = <var list> ;
ROWNAMES = <var list> ;
```

Example :

```
COLNAMES = Score1 to Score10 ;
ROWNAMES = Income Age ;
```

The MATRIX sentence

The procedure **MONCO** gives the monotonicity coefficients in a matrix form (squared or rectangular). Each cell in this matrix is composed by two numbers : the first one is the coefficient value, the second (inserted between parentheses) is the number of cases for which this coefficient was computed. This number is in general different from the number of read cases, because of missing values (if there is). Sometimes, the user wants to get a coefficient matrix without the number of retained cases, for example in order to publish it. In this case he can use the following sentence :

```
MATRIX = SEPARATE  ;
```

This option will output two matrices, one for the coefficients, the other for the number of cases.

The paragraph OUTPUT

This paragraph allows the user to output the monotonicity matrix. There are two ways of using this paragraph : writing the matrix on an external file, in order to use this file later, or writing the matrix into the memory for subsequent use in the same run (for example processing **WSSA1** after **MONCO**).

In the first case, the user may specify the following sentences:

```
OUTPUT
      FILE = 'matrix_file_name' /
      FORMAT = '(F-format)' ;
```

where :

matrix_file_name is a user supplied name for the matrix
F-format is a fortran format in **F**-specification.
The default of the format is **(13F6.2)**.

In the second case, the user may only specify :

```
OUTPUT  MEMORY ;
```

The resulting effect of this command is a transfer of the matrix into an area in the memory, together with the variable names for which **MONCO** was evoked. This means that any matrix processing program in the same run will be able to access the matrix and recognise the variable names. Let us give a simple example using **WSSA1** after **MONCO** with **OUTPUT** paragraph :

```
$
      NAMES = ID 1-4 (A) ITEM1 TO ITEM20 5-24  ;
      FILE = 'MYDATA';
$MONCO
      NAMES = ITEM4 TO ITEM18 ;
      OUTPUT

$WSSA1
      NAMES = ITEM5 TO ITEM15  ;
```

Coefficient of distribution uniformity

When a variable is almost constant over the observations, the coefficient of monotonicity between this variable and any other variable becomes unsteady and therefore cannot be used. To detect such variables, a coefficient, named distribution uniformity coefficient, was found. This coefficient tells us to what extent the variable is uniform. Its value varies between **0** and **1**. **0** occurs when the variable is constant (with one category), **1** is obtained when there are at least 2 categories and when the frequencies of these categories are equal. It was found, empirically, that a value of less than 0.2 is "bad".

The **MONCO** section gives the distribution uniformity coefficient on each variable specified in **NAMES**, **ROWNAMES** or **COLNAMES** sentences

Printed output from section MONCO

```
-----------------------------------------------------------------------
_  1_ $SET  LINESIZE = 80 ;
_  2_ $DATA NAMES = A31 TO A37 31-37 #2  ;
_  3_       FILE = 'W222.DAT' ;
_  4_       MISSINGS = A31 TO A37 0  ;
_  5_ {
_  6_    Squared matrix of monotonicity coefficients between
_  7_    all variables                                        }
_  8_ $MONCO NAMES = A31 TO A37  ;
-----------------------------------------------------------------------

     *********************************
     * WEAK MONOTONICITY COEFFICIENTS *
     *             MONCO              *
     *********************************



Number of Variables .........    7
Number of cases .............  593


    Coefficients of distribution uniformity on each variable

      A   A   A   A   A   A   A
      3   3   3   3   3   3   3
      1   2   3   4   5   6   7
  +----------------------------
  |
  |  73  50  31  50  21  63  61


   Matrix of weak monotonicity coefficients (Decimal point omitted)
              and numbers of cases (N) in computing them


                     1      2      3      4      5      6      7
             +-----------------------------------------------------
             |
A31        1 |    100     70     55     46     51     53     38
             |  ( 586) ( 585) ( 585) ( 584) ( 582) ( 580) ( 581)
             |
A32        2 |     70    100     75     70     70     46     46
             |  ( 585) ( 585) ( 584) ( 583) ( 582) ( 580) ( 581)
             |
A33        3 |     55     75    100     53     78     32     42
             |  ( 585) ( 584) ( 585) ( 583) ( 581) ( 579) ( 580)
             |
A34        4 |     46     70     53    100     79     38     40
             |  ( 584) ( 583) ( 583) ( 584) ( 581) ( 580) ( 581)
             |
A35        5 |     51     70     78     79    100     49     50
             |  ( 582) ( 582) ( 581) ( 581) ( 582) ( 578) ( 579)
             |
A36        6 |     53     46     32     38     49    100     74
             |  ( 580) ( 580) ( 579) ( 580) ( 578) ( 580) ( 578)
             |
A37        7 |     38     46     42     40     50     74    100
             |  ( 581) ( 581) ( 580) ( 581) ( 579) ( 578) ( 581)
```

MONCO section menu

NAMES = <var list> ;

Names of the variables for which matrix of weak monotonicity coefficients is computed.

COLNAMES = <var list> ;

This sentence is used together with **ROWNAMES** sentence to get a rectangular table of coefficients rather than a squared matrix (computed by **NAMES** sentence). "**var list**" appear as column variables.

ROWNAMES = <var list> ;

"**var list**" appear as row variables. Refer to **COLNAMES** sentence.

MATRIX = <UNIFIED | SEPARATE> ;

The way in which results are obtained. **UNIFIED** gives one matrix of coefficients together with number of retained cases. If **SEPARATE** is chosen, two matrices are outputted, one for coefficients, the other for number of cases. This last option is useful when the matrix of coefficients is published.
The default is **UNIFIED**.

OUTPUT paragraph

Paragraph defining the output of the **MONCO** Matrix.

FORMAT = '<char>' /

The format "**char**" specifies the layout (in fixed fields) of the matrix in a single row . The usual FORTRAN rules for the format are applicable, but only the F specification is allowed. "**char**" cannot exceed 80 characters.
The default is (13F6.2).

FILE = '<char>' /

"**char**" is the matrix file pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

MEMORY /

The resulting matrix is written into the memory, in order to use it as input of another section (as **WSSA1** for example) in the same job.

References

Guttman, L. Coefficients of polytonicity and monotonicity. Encyclopedia of Statistical Sciences, N.Y.: John Wiley & Sons, Inc., 1986, 7, 80-87.

Guttman, L. What is not what in statistics. The Statistician, 1977, 26, 81-107.*

* To be found also in Levy, S. (Ed.) Louis Guttman on Theory and Methodology: Selected Writings, Aldershot: Dartmouth, 1994.

The *DISCO* Section

Introduction

When two or more populations have distributions on the same numerical variable *x*, it is of interest to know to what extent these distributions overlap. One motivation for this interest is the problem of discriminant analysis. Suppose an individual has a known value of *x*, but the population is unknown. To which population should the individual be assessed to belong, with minimal expected error?

Two coefficients are computed by **`DISCO`** section. They express the loss due to overlap as a direct function of the variance between the arithmetic means of the distributions. One is called *disco* for "discrimination coefficient". The other is called *odisco*; it is more relaxed than *disco* in a certain sense of overlap. The "*o*" at the begining of *odisco* is meant to indicate that some overlap is allowed. Each is distribution-free, avoiding traditional assumptions of normality of population distributions and equality of variances within the populations. Such conventional assumptions are unrealistic and misleading in many cases.

Definition

There are two kinds of *disco* or *odisco* coefficients. These coefficients can be computed for one variable on *m* populations or for two (or more) variables on one population.

For one variable on m populations

Given a numerical variable *x* and *m* populations P_k $(k = 1,2,\ldots,m)$. For each pair of populations P_a and P_b, let $\bar{x}_a$ and $\bar{x}_b$ be the arithmetic means of x, let the largest value of *x* for P_b be denoted by $\max_{x \in P_b}(x)$, and let the smallest value of *x* for P_a be denoted by $\min_{x \in P_a}(x)$.

disco asks whether or not

$$\max_{x \in P_b}(x) \le \min_{x \in P_a}(x) \quad \text{for} \quad \bar{x}_b < \bar{x}_a \qquad \text{(}\textit{disco}\text{ condition)}$$

In contrast *odisco* asks whether or not two inequalities hold:

$$\max_{x \in P_b} \leq \bar{x}_a \quad \text{and} \quad \min_{x \in P_a} \geq \bar{x}_b \quad \text{for} \quad \bar{x}_b < \bar{x}_a \qquad (\textit{odisco} \text{ condition})$$

When the *disco* condition holds, then there is no overlap between the distributions. When the *odisco* condition holds, no members of population P_b, has an *x*-value above $\bar{x}_a$ and no member of population P_a has an *x*-value below $\bar{x}_b$: there may be overlap in the interval between the two means, but no overlap in the two intervals outside the means.

disco and *odisco* vary between 0 and 1. They equal 0 if there is no difference among the means, *disco* equals 1 for perfect discrimination (non-overlap) (see Figure 1), *odisco* equals 1 for perfect discrimination in the intervals outside the means (see Figure 2). We have always $0 \leq disco \leq odisco \leq 1$.

For a mathematical formulation of *disco* and *odisco* see the Appendix A.

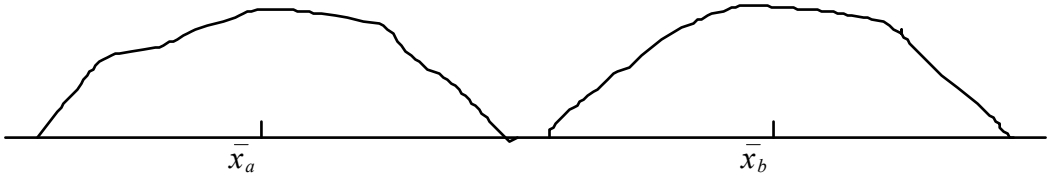


Figure 1

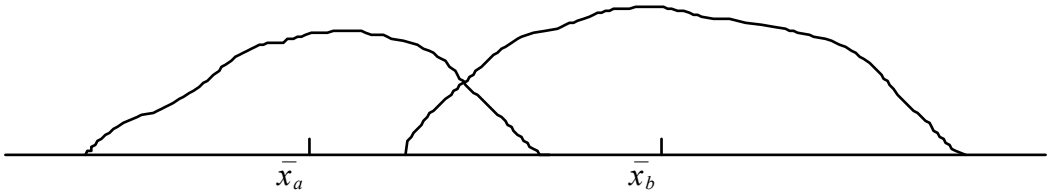


Figure 2

For two variables on one population

Given 2 numerical variables *x* and *y* on one population *P*, it is possible to define *disco* and *odisco* between these variables. The *disco* condition becomes:

$$\max_{y \in P}(y) \leq \min_{x \in P}(x) \quad \text{for} \quad \bar{y} < \bar{x}$$

The *odisco* condition becomes:

$$\max_{y \in P} \leq \bar{x} \quad \text{and} \quad \min_{x \in P} \geq \bar{y} \quad \text{for} \quad \bar{y} < \bar{x}$$

These conditions have a meaning only if variables *x* and *y* have the same scale.

The attention was focused on two variables, since this is the more practical case. But the above conditions can be extended to more than two variables.

Operation

The procedure **DISCO** is invoked by the section name **DISCO**. 3 sentences (**NAMES**, **GROUPS**, **CONTRAST**) and one paragraph (**OUTPUT**) are available in this section.

These above sentences and the paragraph **OUTPUT** can be named in any order and are separated from each other by a semicolon. Each one can be used only once for **DISCO** section.

The NAMES sentence

As stated above there are two kinds of *disco* and *odisco* coefficients.

If **GROUPS** sentence is specified (see later), coefficients are computed for each variable listed in **NAMES** sentence, according to populations defined in **GROUPS** sentence.

If **GROUPS** sentence is omitted coefficients are computed for each pair of variables listed in **NAMES** sentence.

GROUPS sentence

In order to compute the coefficients, **DISCO** needs the different groups (populations) to be defined through one grouping variable. This is done by the **GROUPS** sentence:

```
GROUPS = <variable val list | ... | val list> ;
```

"**variable**" is the grouping variable. Each "**val list**" defines a group. The symbol "**|**" is a delimiter between two group definitions. For example :

```
GROUPS = Race  1 2 3 | 4 | 5 TO 9  ;
```

Here 3 groups are defined. The first group is composed by members who have categories **1**, **2** or **3** in variable **Race**; the second group is defined only by category **4**; all the members who have categories **5**, **6**, **7**, **8** or **9** define the third group.

CONTRAST command

As described above *disco* or *odisco* coefficients are computed for *m* groups. When *m* > 2, these coefficients cannot give information about the couples of

groups, unless *disco* = 0 or 1. The command **CONTRAST** allows the user to get the discrimination coefficients for each couple of groups, in adding

```
CONTRAST ;
```

PLOT command

If **GROUPS** sentence is specified, the distribution of the variable for each group is plotted. The plotting is repeated for each variable listed in **NAMES** sentence.

If **GROUPS** sentence is omitted, the distribution of each variable listed in **NAMES** sentence is plotted.

To get these plots, the user adds the command:

```
PLOT ;
```

The paragraph OUTPUT

This paragraph is available only for the second kind of *disco*, that is for pairs of variables on one population. It allows the user to output the *disco* matrix between all pairs of variables listed in **NAMES** sentence. There are two ways of using this paragraph: writing the matrix on an external file, in order to use this file later, or writing the matrix into the memory for subsequent use in the same job (for example processing **WSSA1** after **DISCO**).

In the first case, the user may specify the following sentences :

```
OUTPUT
      FILE = 'matrix_file_name' /
      FORMAT = '(F-format)' ;
```

where :

matrix_file_name is a user supplied name for the matrix
F-format is a fortran format in **F**-specification.
The default of the format is **(13F6.2)**.

In the second case, the user may only specify :

```
OUTPUT  MEMORY ;
```

The resulting effect of this command is a transfer of the matrix into an area in the memory, together with the variable names for which **DISCO** was evoked. This means that any matrix processing program in the same run will be able to access the matrix and recognise the variable names. Let us give a simple example using **WSSA1** after **DISCO** with **OUTPUT** paragraph :

```
$DATA
      NAMES = A31 TO A41 31-41 ;
      FILE = 'W222.DAT' ;
      MISSINGS = A31 TO A41 0  ;
$DISCO
       NAMES = A31 TO A41 ;
       OUTPUT  MEMORY ;
$WSSA1
       NAMES = A31 TO A41 ;
       DATA = DISSIM ;
```

Note that disco or odisco are dissimilarity (distance) coefficients. Therefore, WSSA1 has to be informed to handle the matrix correctly. This is the purpose of the sentence:

```
DATA = DISSIM ;
```

Printed output from section DISCO

```
-----------------------------------------------------------------
_  1_ $DATA NAMES = A31 TO A41 31-41 / RELIGION 69  ;
_  2_       FILE = 'W222.DAT' ;
_  3_       MISSINGS = A31 TO A41 0  ;
_  4_       VARLABS = A31 'J. history of 3000 years'
_  5_                 A32 'Recent J. world history'
_  6_                 A33 'Holocaust'
_  7_                 A34 'Recent J. history in Isr'
_  8_                 A35 'Establish. of the State'
_  9_                 A36 'Jewish religion'
_ 10_                 A37 'Jewish moralty'
_ 11_                 A38 'Mutual assistance'
_ 12_                 A39 'Jewish tradition'
_ 13_                 A40 'Education at home'
_ 14_                 A41 'Attitude of non-Jews'  ;
_ 15_ $DISCO NAMES = A31 TO A41 ;
_ 16_        GROUPS  = RELIGION  1 2 | 3 | 4 ;
_ 17_        CONTRAST ;
-----------------------------------------------------------------


               ***************
               *  D I S C O  *
               ***************

        Discrimination Coefficients
     For Comparison of Arithmetic Means
             ("Effective ANOVA")




Number of Variables ..........   11
Number of Cases ..............  593



                           Notes on F
                           ----------

1) F is  calculated  here only for traditional interest in it.
2) When there are only two groups, F = t**2 (for t-test).
3) Probabilities are not printed here for  significance  of  F
   because of the following reasons :
   a) The   traditional  calculations  of  probabilities   are
      incorrect for  simultaneous  null  hypotheses  such   as
      tested when more than one F is calculated (as is done by
      this program and by traditional ANOVA programs).
   b) The use  of  different levels of significance simultane-
      ously, ( as traditionally done by use of starring :   *,
      **, ***  )  contradicts  the  Neyman-Pearson  theory  of
      testing hypotheses,  and  vitiates  any  possibility  of
      correct calculation  of probabilities.  (Such use is but
      a poor  way  of  trying  to  estimate   size   of   mean
      differences or  efficacy.   Much better is to use direct
      efficacy coefficients such as disco odisco and eta.)
   c) Even were correct calculation of probabilities  possible
      for rejection-acceptance  of null hypotheses, they would
      say nothing  whatsoever   about   the   probability   of
      rejection-acceptance in a replication.
   d) F estimates  no  population parameter (it is merely used
      to test  the  null  hypothesis   that   the   population
```

```
      parameter of  eta  is  zero,  that is, the hypothesis of
      non- effectiveness).
4) Efficacy coefficients such as disco  odisco  and  eta,  are
   consistent   estimates   of   their   respective   population
   parameters, are consistent estimates of what will happen in
   a
   r
5) There can  be  perfect discrimination (non-overlap) between
   the group distributions, yet F can never reveal  this  (nor
   can   eta).   The   Disco   value   equals   1   for    perfect
   d
   in the intervals outside the means.


Group  1 : RELIGION =  [ 1.0, 1.0] , [ 2.0, 2.0]
Group  2 : RELIGION =  [ 3.0, 3.0]
Grou


Variable  Group     N      Mean        SD  * Odisco  Disco    Eta
-------------------------------------------------------------------
J. h


        Group  3  150     2.507      .988  *    .49    .39    .23
      -------------------------------------------------------------
      | Group  1  177     1.881      .984  *
      | Group  2  254     2.220      .961  *    .41    .31    .17
      -------------------------------------------------------------
      | Group  1  177     1.881      .984  *
      | Group  3  150     2.507      .988  *    .64    .52    .30
      -------------------------------------------------------------
      | Group  2  254     2.220      .961  *
      | Group  3  150     2.507      .988  *    .33    .26    .14
-------------------------------------------------------------------
Recent J. world history                      *
        Group  1  177     1.706      .764  *
        Group  2  253     1.897      .780  *
        Group  3  150     1.920      .879  *    .30    .23    .11
      -------------------------------------------------------------
      | Group  1  177     1.706      .764  *
      | Group  2  253     1.897      .780  *    .30    .24    .12
      -------------------------------------------------------------
      | Group  1  177     1.706      .764  *
      | Group  3  150     1.920      .879  *    .33    .25    .13
      -------------------------------------------------------------
      | Group  2  253     1.897      .780  *
      | Group  3  150     1.920      .879  *    .04    .03    .01
-------------------------------------------------------------------
Holocaust                                    *
        Group  1  176     1.392      .667  *
        Group  2  254     1.492      .715  *
        Group  3  150     1.507      .730  *    .17    .16    .07
      -------------------------------------------------------------
      | Group  1  176     1.392      .667  *
      | Group  2  254     1.492      .715  *    .17    .16    .07
      -------------------------------------------------------------
      | Group  1  176     1.392      .667  *
      | Group  3  150     1.507      .730  *    .19    .18    .08
-------------------------------------------------------------------
     ........................................
     ........................................
```

DISCO section menu

NAMES = <var list> ;

Names of the variables for which discriminant coefficients are computed. There are two kinds of Disco (Odisco) coefficients:

- If **GROUPS** sentence is submitted the coefficient is computed on each variable in "**var list**" throughout the grouping variable.
- If **GROUPS** sentence is not submitted the coefficient is computed on each pair of variables in "**var list**". The resultant matrix can be outputted to the memory, by mean of **OUTPUT** paragraph, in order to use it as an input of another section (for example **WSSA1**).

GROUPS = <variable val list | ... | val list> ;

Definition of groups through a grouping variable. Each "**val list**" defines a group. The symbol "**|**" is a delimiter between two group definitions.

CONTRAST ;

This option allows the processing of all couples of groups if there is more than 2 groups.

PLOT ;

To plot the distribution of the variable for each group. If **GROUPS** sentence is not submitted the distribution of each variable is plotted.

OUTPUT paragraph

Paragraph defining the output of the **DISCO** Matrix (only when **GROUPS** sentence is omitted).

FORMAT = '<char>' /

The format "**char**" specifies the layout (in fixed fields) of the matrix in a single row . The usual FORTRAN rules for the format are applicable, but only the F specification is allowed. "**char**" cannot exceed 80 characters.
The default is **(13F6.2)**.

FILE = '<char>' /

"**char**" is the matrix file pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

MEMORY /

The resulting matrix is written into the memory, in order to use it as input of another section (as **WSSA1** for example) in the same job.

References

Guttman, L. Efficacy coefficients for differences among averages. in I. Borg (Ed.) Multidimensional Data Representations: When and Why. Ann Arbor, MI: Mathesis Press, 1981, pp. 1-10.

Guttman, L. Eta, Disco, Odisco and F. Psychometrika, 1988, 53, 393-405.

Guttman, R. and Zohar, A. Spouse similarities in personality items: Changes over years of marriage. Behavior Genetics, 1987, 17, 179-189.

Guttman, R. Zohar, A., Willerman, L., and Kahneman, I. Spouse similarities in personality traits for intra and interethnic marriages in Israel. Personality and Individual Differences, 1988, 9, 763-770.

Levy, S. Discriminant analysis of recognition of works of art among school students. In R. Gutierrez & M.J. Valderrama (Eds.) Applied Stochastic Models and Data Analysis, Singapore: world Scientific, 1991, pp. 409-432.

The *WSSA1* Section

Purpose

The name **WSSA1** is for: **W**eighted **S**mallest **S**pace **A**nalysis. "**1**" indicates that the process is for symmetric matrix.

Given a matrix $\{R_{ij}\}$ containing pairwise similarity coefficients (correlations), among a set of *n* variables, $V_1, V_2, \ldots, V_n$, the **WSSA1** section enables study of the matrix in a simple yet comprehensive manner.

The principal output is a space diagram plot representing each V_i as a point in the dimensionality chosen by the user, but sought to be as small as possible. The points are located in the space in such a way that they satisfy the monotonicity condition as well as possible, that is $d_{ij} < d_{kl}$ whenever the observed data indicate that $R_{ij} > R_{kl}$, d_{ij} being the Euclidean distance between two points.

There is an option to superimpose facet elements for each variable, and produce facet diagrams. This option facilitates viewing regional correspondence between the empirical distribution of the variables and their faceted definition.

Subgroups of the population can be located in the fixed space diagram of the original variables via external variable feature.

The empirical data to be analysed are not limited to coefficients of similarity. They can be also dissimilarity coefficients (distances), say $\{D_{ij}\}$. In such a case, the monotonicity condition becomes: $D_{ij} < D_{kl} \Leftrightarrow d_{ij} < d_{kl}$.

A detailed description of **WSSA1** algorithm is given in Appendix A.

Input to WSSA1

The use of **WSSA1** is not limited to a specific type of items or numbers. The items may be various objects (e.g. people, bodies, growths, social structures, etc.) or variables of different sorts (e.g. age, sex, ethnicity, space, volume, etc.) or categories of variables (specific ages, specific attitudes, or specific quantities).

The coefficients in the matrix will express in each case a relationship between two items. The relationship may be either of similarity or of dissimilarity.

The coefficients themselves may be of various kinds: correlation coefficients (a kind of similarity), geographical distances (a kind of dissimilarity), similarity between people, various proportions, or any numerical or other "graded" system expressing similarity or dissimilarity.

Examples of Matrices Appropriate to WSSA1

We shall now look at a number of matrices appropriate as input for the **WSSA1** procedure.

Example 1

```
-----------------------------------------------------------------
Variable                 1    2    3    4    5    6    7    8    9
-----------------------------------------------------------------
1. Dominant           1.00  .56  .24 -.26 -.27 -.30  .01 -.06  .39
2. Hostile-Rebellious  .56 1.00  .60 -.06 -.04  .06 -.38 -.49  .00
3. Suspicious          .24  .60 1.00  .27  .28  .21 -.35 -.50 -.16
4. Inhibited          -.26 -.06  .27 1.00  .49  .19 -.27 -.51 -.54
5. Abasive            -.27 -.04  .28  .49 1.00  .50 -.10 -.14 -.26
6. Passive-Dependent  -.30  .06  .21  .19  .50 1.00 -.32 -.17 -.19
7. Nurturant           .01 -.38 -.35 -.27 -.10 -.32 1.00  .61  .29
8. A
9. S
----
```

Table 1.
Correlation Coefficients among 9 Social Behaviour Variables From Lorr & McNair, An Interpersonal Behaviour Circle. Jour. of Abnormal and Social Psychology, 1963, 67: 68-75.

Table 1 is a matrix representing correlation coefficients among 9 social behaviour characteristics. The correlation coefficients are based on a sample of 420 people who were scored on each of these characteristics.

The correlation coefficients indicate the relative similarity between the items (variables) of the matrix. The higher the correlation the greater the similarity. Thus the two most similar variables in the table are variables 7 and 8: their intercorrelation is .61. Variables 1 and 7 have practically a zero correlation, .01, and hence variable 1 is more dissimilar from variable 7 than variable 8 is from variable 7. Being "**Nurturant**" is more similar to being "**Affiliative**" than it is to being "**Dominant**".

A negative correlation indicates even less similarity than does a zero correlation. Thus the variable most dissimilar to "**Nurturant**" is variable 2 "**Hostile-Rebellious**" since it has the most negative correlation (-.38), in row 7 and column 2.

There often arises the question of what sign to give a correlation coefficient. Reversing the wording of a variable will reverse the sign of its correlation with all other variables. Decision on what direction to give a variable depends on the substantive content of the theory involved, and is outside the scope of **WSSA1**. This decision must be made on theoretical grounds before the input is submitted.

There is an implicit social theory in the directions of scoring of the nine variables leading to Table 1. An interesting pattern that results is apparent in the matrix, namely, a circular ordering among the variables, in the order listed. Two variables are more similar as they are closer together in the order, and less similar as they get farther apart until about half-way down the list when they begin to increase again. Thus variable 1 is close to variable 2, is less so to variable 3 and is least similar to variable 6, becoming similar again to variable 9. The pattern here is so simple that its essential circularity can be seen without a formal treatment by **WSSA1**. A matrix with a circular structure like this is called a circumplex.

Example 2.

Settlement	1	2	3	4	5	6	7	8	9	10
1. Metula	0	87	85	48	65	106	96	107	95	119
2. R										
3. Naharia	85	11	0	52	67	93	56	66	10	31
4. Zfat	48	54	52	0	36	77	58	70	52	74
5. Tveria	65	75	67	36	0	41	32	43	57	70
6. Beit She'an	106	104	93	77	41	0	40	27	84	69
7. Nazeret	96	66	56	58	32	40	0	12	47	38
8. Afula	107	77	66	70	43	27	12	0	57	42
9. Acco	95	21	10	52	57	84	47	57	0	22
10.Haifa	119	42	31	74	70	69	38	42	22	0

Table 2
Geographical Distances Between Settlements in the Northern Part of Israel (in Kilometres)

Table 2 is a distance matrix, in kilometres, between settlements in northern Israel.

Distance coefficients are an example of dissimilarity coefficients. They are ranked in the opposite direction from similarity coefficients: the smaller the coefficient the greater the similarity between the things being compared. "Similarity" for Example 2 is in the sense of geographical location. Two places which are more similar (closer) geographically have a smaller coefficient in Table 2.

Both Tables 1 and 2 can be given a Smallest Space Analysis portrayal in two dimensions, and it is not difficult to do this by hand. Therefore the computer analysis will not be given here; the hand analysis will be left as exercises.

As already discussed, the input coefficients may be that of similarity or dissimilarity. There is a place in the **WSSA1** section - to be discussed below - to

indicate which type of coefficient is being analysed. Improper results will be obtained by confusing the two types of coefficients.

Example 3.

Table 3 is an example of a symmetrical matrix of input coefficients, which was analysed by the **WSSA1** program. The matrix consists of correlation coefficients of lengths of boys' limbs, i.e. correlation coefficients of the interrelationship among four variables. These are similarity coefficients, since a higher correlation coefficient means greater similarity.

```
-----------------------------------------------

-----------------------------------------------
1. Leg            1.00    .75    .69    .63
2. Arm             .75   1.00    .72    .72
3. F
4. H
----
```

Table 3.
Correlation Coefficients Between Lengths of Boy's Limbs
Source: R. Guttman & L. Guttman, A New Approach to theAnalysis of Growth Patterns. Growth, 1965, 29:219-231.

Table 4 is a printout of the same input coefficients and is part of the **WSSA1** output. The printout consists of the full matrix although it is a symmetric matrix. Printing of 100 in the diagonal indicates that we have a matrix of similarity. For a dissimilarity matrix, 0's are printed on the diagonal. The variables in the printout are given a serial number according to their order of appearance in the matrix. The variable in the extreme left column (hence also the first row) is coded "1", the following variable is coded "2", and so on. This coding accompanies the variables throughout the printout.

For the purpose of printing, the input coefficients are multiplied or divided by a power of 10 in order to express them into integer numbers of at most 3 digits. This is a nice presentation for publication purposes.

```
            I N P U T   M A T R I X *


                1   2   3   4
            +---------------
            I
Leg       1 I 100  75  69  63
            I
Arm       2 I  75 100  72  72
            I
Foot      3 I  69  72 100  75

Hand


* The original coefficients were multiplied by  100
  and rounded into integer numbers
```

Table 4.
Printout of the input coefficients by **WSSA1**

Running MONCO and WSSA1 in the same job

A matrix of correlation coefficients can be supplied, for example, to **WSSA1**, by a previous process of **MONCO**, which uses itself raw data entered by **DATA** section. Follows a command file of such a kind of job:

```
$DATA
      NAMES = ID 1-4 (A) Item1 to Item20 5-24  ;
      FILE = 'MYDATA';
$MONCO
      NAMES = Item1 to Item20 ;
      OUTPUT
            MEMORY  ;
$WSSA1
      NAMES = Item1 TO Item20  ;
```

The paragraph **OUTPUT** with **MEMORY** option in **MONCO** section, transfers the resulting matrix of monotonicity coefficients into an area in HUDAP memory. Then, **WSSA1** program takes this matrix as input.

Running WSSA1 directly on an external matrix

Another way to run **WSSA1** is to enter directly, into HUDAP, a matrix by means of **MATRINP** section. This matrix can be of course, a matrix of similarity or dissimilarity coefficients. Follows an example:

```
$MATRINP
        NAMES =   V1  to V16 1-80 (2)
                / V17 to V30 1-70 (2) ;
        FILE  = 'Mat30' ;
$WSSA1
        NAMES = V1 TO V30 ;
```

The output results of WSSA1 procedure

The main part of the **WSSA1** output is the space diagram. In this diagram, each variable is represented by a point. For every pair of variables the program calculates a value which is the distance between the points which represent them in the space: the larger the similarity coefficient between two variables, the smaller the distance between their points in the space. (The opposite is true with respect to distance coefficients). More precisely, the following rule applies:

If the similarity coefficient between A and B is larger than the similarity coefficient between E and F, then the distance between A and B is smaller than that between E and F.

It should be noted that A, B, E, F, refer both to the variables and the points representing them in the space. It should also be noted that the above rule specifies what is called semi-strong monotonicity. The rule leaves open what to do with tied input coefficients: the program is allowed to yield unequal distances for tied input if this will simplify the data analysis. However, the **WSSA1** algorithm looks for an optimal order of the tied input coefficients to obtain the better fit (see below).

Dimensionality

The space presenting the points has a specific number of dimensions. A space may be unidimensional (a straight line), two-dimensional (a surface represented by a length and a width), three-dimensional (a body represented by a length, width and height), or four-dimensional or more (in which case a direct physical presentation is impossible). The **WSSA1** program is designed to finding a space with the smallest number of dimensions which enables a faithful presentation of the input coefficients matrix in the sense that a perfectly faithful representation of any symmetric matrix of n variables is always mathematically possible in a space of n-2 dimensions. For n=4 the Guttman theorem implies that two dimensions must always suffice. Thus, since Table 3 has only four variables, it must be representable in exactly two dimensions.

For both technical and substantive reasons it is desirable to fit a space of smaller dimensionality than n-2 to empirical data. The smaller the dimensionality relative to n the greater the stability of the computed solution.

More important from the scientific point of view, small dimensionality by itself is an indication of empirical lawfulness and facilitates seeing more specific types of lawfulness in the data.

Two sentences allow the user to choose the dimensionalities of the analysis:

MINDIM = <value> ;
where "**value**" is the minimal dimensionality from which **WSSA1** is processed. It must be a positive integer number.The default of **MINDIM** is 2.

MAXDIM = <value> ;
where "**value**" is the maximal dimensionality at which **WSSA1** is processed. It must be a positive integer number. The default of **MAXDIM** is 3.

Example of WSSA1 output

Let us return to Tables 3 and 4. Since only four variables are involved, a two-dimensional space must give a perfect fit, according to the Guttman Theorem. Let us first examine computer output of the two-dimensional space diagram (Fig.1) and then look at the Table accompanying the diagram.

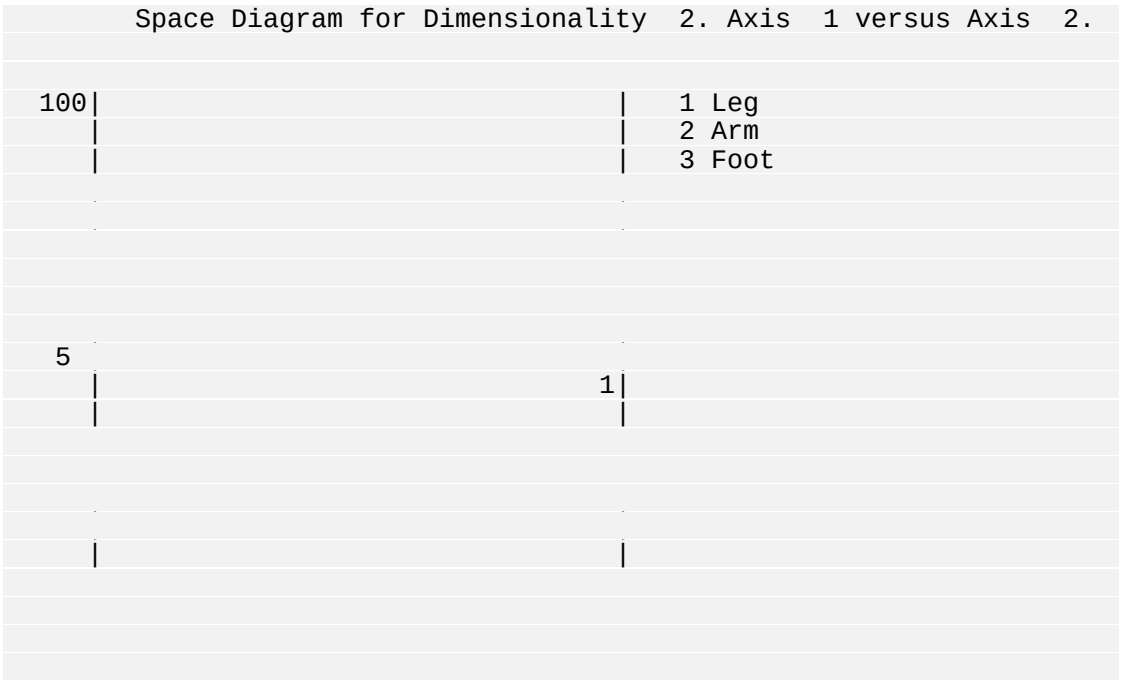


Figure 1
A two-dimensional space diagram.

At the top of the page presenting the diagram appears a title, which indicates the dimensionality and the 2-dimensional projection represented by the diagram. Axis 1 (the horizontal axis x) denotes dimension 1, Axis 2 (the vertical axis y) denotes dimension 2. Such axes are presented only for convenience and are not

essential to the analysis, (More will be said about interpretation of axes later; the major message is: do not try to interpret axes in general.)

Four points appear in the diagram -- coded from 1 to 4 -- which represent the respective four variables in the matrix of input coefficients. At the right hand of the plot appear the variable names given by the user. If variable labels are also supplied (via **DATA** section), they are printed instead of the variable names.

Verifying the Fit

Let us verify the fitness of the diagram to the matrix. From Table 5. we can compute the Euclidean distances between each of the 4 variables. The results are grouped in Table 3 bis, in the form of normalised distance matrix (values between 0. and 1.). The distance between variables 1 and 2 (0.52) is less than that between 3 and 4 (0.55), although, the correlation coefficients between these two pairs are identical (0.75). This is still correct because of the semi-strong monotonicity which allows to tied input values to have unequal corresponding distances. Variables 1 and 2 are closer to each other than 2 and 3 (0.52 < 0.80), and indeed, the correlation coefficient between 1 and 2 (0.75) is higher than the correlation coefficient between 2 and 3 (0.72). The lowest correlation coefficient in the matrix is that of between variables 1 and 4 (0.63), and indeed, they are the farthest away from each other in the diagram (1.00).

```
----
1. L
2. A
3. F
4. Hand           1.00   0.64   0.55   0.00
-----------------------------------------------
```

Table 3 bis
Matrix of computed distances corresponding to correlation coefficients of Table 3.

It should be noted that this examination refers only to the relative distances of the points from each other, and not to the absolute distance values. There is no absolute implication to a specific distance between two points, but rather to the extent of their closeness, in relation to the closeness of other points. Note also that tied values in the input were not untied in the output. Perfect fit was possible in this case without untying ties. The solution maintains strong monotonicity in this example, and not merely semi-strong. More usually, tied input coefficients will correspond to untied distances in the output.

Coding of the Space Diagram

In order to facilitate the use of the diagram and enable the numerical location of the points (which is especially necessary when the input matrix is large), the

coordinates x and y in the space were assigned values. These values run from 0 to 100. Such coordinates for each point in the diagram is a pair of values which describe the exact location of the point in the space.

The hypothetical point at the lower left corner of the diagram, (called "the origin") is assigned the pair of values (0,0). The point at the upper left corner of the diagram has the coordinates (0, 100), where 0 is the value of the first dimension and 100 that of the second dimension. The point located at the upper right corner has the coordinates (100, 100).

Table 5 is a table giving the coordinates calculated and used for the space diagram. That the present example is two dimensional is indicated in its title:

D I M E N S I O N A L I T Y 2

Rank Numb Coef	Ser Num		
1	.00000	100.00	44.61
2	.00000	67.31	.00
3	.00000	12.66	64.83
4	.00000	.00	7.95

Table 5
The numerical table accompanying the 2 dimensional space

Locating Points in The Space Diagram

In the body of Table 5 are given for each variable: its code (from 1 to 4), a number which is the distance of this point from the centroid (= gravity centre) and the values assigned to that point in the first (horizontal) and second (vertical) dimensions.

In principle, the information contained in the table (the coordinate values of all the points) is identical to the information provided graphically by the diagram. Actually, the tabular data are a bit more precise than those of the diagram. Owing to technical limitations, the computer printer can print only a fixed number of signs along the vertical and horizontal axes. It is not always possible to print the points in the exact place they should appear according the calculations.

As stated above, the numbers appearing in the second column of the table represent the distances of the points from the centroid. Variable 1 is farthest away form the centroid (57.09). Variable 2 is closest to it (36.87).

The great advantage of the diagram over the table lies in the simplicity of observation. The diagram enables viewing the interrelations among all variables simultaneously. The same information is in the Table, but not so easily digestible.

The Shepard Diagram

Figure 2 is a printout of another important feature of **WSSA1**, called the Shepard Diagram. Here, the vertical axis (y) represents the input coefficient values. In this example the y values run from 63 to 75. The horizontal axis (x) represents the distances between points in the Space Diagram. In our example the x values run from 0.811 to 1.563.

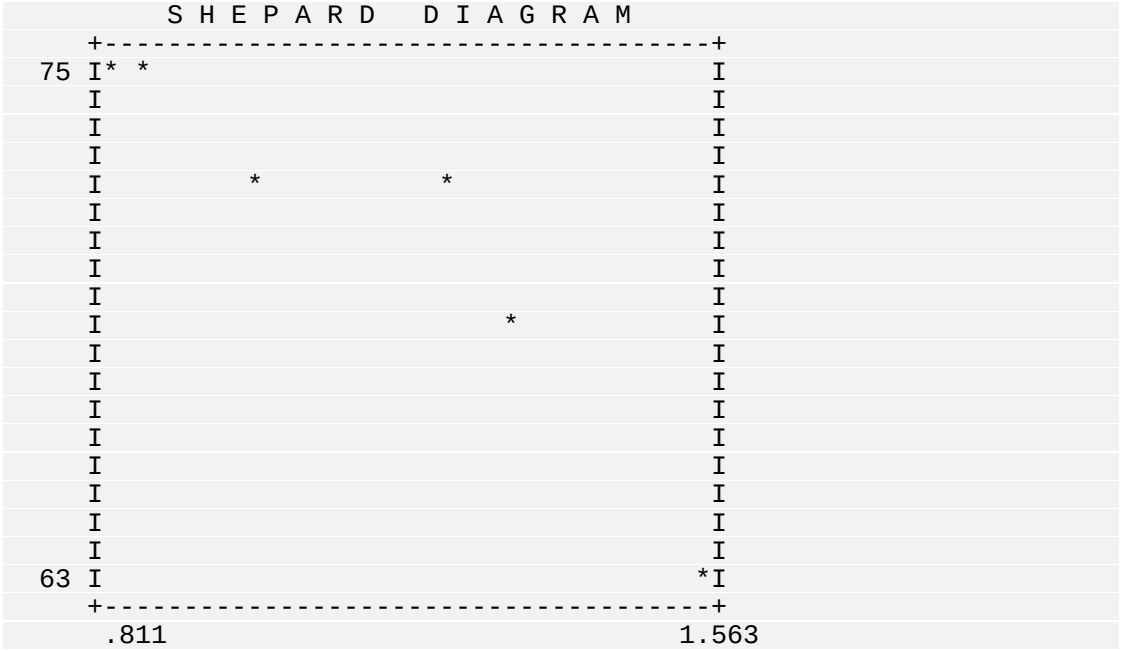


Figure 2.
A Shepard diagram.

In the Shepard Diagram there are more points than in the Space Diagram. The number of points equals the number of pairs of variables, i.e. the number of coefficients in the input matrix. Thus, if n is the number of variables, then the number of pairs of variables is n(n-1)/2. Every similarity coefficient between two variables is the y coordinate of a specific point, while the distance between these two variables in the space is the x coordinate of the same point. The point itself represents the pair of variables, in contrast to the Space Diagram in which each point represents a single variable.

For the example of Figure 2, n=4, so the number of points in the Shepard Diagram is 4(4-1)/2 = 6. In general practice, n is much larger than 4 and n(n-1)/2 is far larger than n. The number of points in the Shepard Diagram is usually very much larger than in the Space Diagram.

There are two kinds of information portrayed by the Shepard Diagram: (1) The shape of the curve relating the input to the output, and (2) the goodness-of-fit of this curve.

We shall first discuss shape of curve in the context of the present example. Discussion of goodness-of-fit will be postponed to the next example which will illustrate the case of imperfect fit.

Shape of the Curve

When there is perfect fit, as in the example of Figures 1 and 2, this is indicated in the Shepard Diagram by a descending monotone series of points. The further to the right a point lies (i.e. the greater the distance between the two variables that the point represents) the lower will it be. This is an expression of a perfect monotone relation between the input and the output coefficients. For semi-strong monotonicity, two adjacent points may remain at the same level: they have tied inputs, but untied outputs (distances).

The descending curve is for the case of similarity coefficients as input. The monotone curve will be ascending when the input is of dissimilarity coefficients.

The monotone descending curve is clear in Figure 2. At the upper left corner of the diagram in Fig. 2 there are two points on the same level - these have as y-cooordinate the two highest coefficients in the input matrix (.75), i.e. the coefficients between variables 1 and 2, and between variables 3 and 4.

To the right and below these two points there is another pair of points whose y-coordinate is .72. These are the two tied coefficients between variables 2 and 3 and between variables 2 and 4. To the right and below these is located a point with y-coordinate .69 for the pair of variables 1 and 3, while at the bottom of the diagram lies the point with y-coordinate .63 for the pair of variables 1 and 4. Thus, the points are arranged in a descending monotone order, where the point with the largest input (y) value has the smallest output distance (x).

Analysis of an empirical example

The previous example exhibited a perfect fit between the two-dimensional space diagram and the input matrix. We shall now look at an example in which the fit is not perfect. This is in fact the case when empirical data are analysed. In practice, it is generally not possible to fit a small space perfectly.

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	+															
	I															
1	I															
	I															
2	I	54	100	49	28	18	14	21	19	0	32	23	19	13	19	23
	I															
3	I	44	49	100	29	23	19	26	27	6	45	29	23	21	23	30
	I															
4	I															
	I															
5	I															
	I															
6	I	14	14	19	15	54	100	24	23	17	24	20	17	12	18	24
	I															
7	I	22	21	26	23	25	24	100	33	13	35	27	25	25	27	34
	I															
8	I	22	19	27	23	26	23	33	100	21	37	32	40	30	40	50
	I															
9	I	5	0	6	6	18	17	13	21	100	17	17	9	12	14	26
	I															
10	I															
	I															
11	I															
	I															
12	I															
	I															
13	I															
	I															
14	I	24	19	23	21	18	18	27	40	14	32	25	31	48	100	50
	I															
15	I	28	23	30	24	28	24	34	50	26	45	36	32	38	50	100

Table 6.
Interrelationships (Pearson Coefficients) among fifteen variables of satisfaction with life in the United States.

1 City as place to live
2 Neighbourhood
3 Housing
4 Life in the U.S.
5 Amount of education
6 Useful education
7 Job
8 Spending of spare time
9 Health
10 Standard of living
11 Savings and investments
12 Friendships
13 Marriage
14 Family life
15 Life in general

Table 6 is a matrix of input coefficients of similarity among variables of satisfaction with life in the United States, taken from a study carried out at the Israel Institute of Applied Social Research. This matrix comes from a special

analysis of data gathered for the study "Quality of Life" by the University of Michigan Survey Research Centre on a national U.S. sample of 2164 respondents, in the summer of 1971. We shall show how goodness-of-fit is assessed using the Coefficient of Alienation.

The Coefficient of Alienation varies between 0 and 1. Perfect fit is represented by the value 0, and the worst possible fit is given by the value 1. Intermediate values of the coefficient represent intermediate of degrees of goodness-of-fit.

Goodness-of-Fit: The Coefficient of Alienation

Table 7 presents coordinates for the 15 variables of Table 6 in the best-fitting two-dimensional space obtained by **WSSA1**. Inside the table we find printed:

```
Coefficient of Alienation ..............  .14040
```

The number 0.14 expresses the extent of alienation or badness-of-fit, of the two-dimensional space to the input matrix. In other words, it expresses the extent to which some distances between pairs of points in the two-dimensional space do not adhere to the rule regarding the monotone relationship between input coefficients and output distances.

```
                     D I M E N S I O N A L I T Y   2
                     --------------------------------

 Rank image transformations .............   8
 Number of iterations ...................  14
 Coefficient of Alienation ..............  .14040

 Serial    Item coeff. of   Plotted Coordinates
 Number    Alienation          1        2
 ------------------------------------------------
    1        .09546           8.50    41.25
    2        .08284           5.57    50.57
    3        .10188          19.64    43.46
    4        .12786            .00    29.01
    5        .10158          55.98    63.50
    6        .12433          60.44    68.11
    7        .18447          52.23    34.98

    9        .12147         100.00    34.16
   10        .11495          36.12    38.56
   1
   12        .17676          28.05     8.56
   13        .15433          53.64      .00
   1
   1
```

Table 7
Coordinates for the 15 variables of Table 6. in a 2 dimensional space

What is a Good Fit?

How small should the Coefficient of Alienation be for the fit to be satisfactory? This is a question to which there is no absolute answer. Indeed in all of empirical science, for any kind of coefficient of fit, there can be no absolute answer; it depends on the purpose of the study. What may be exact enough for one purpose will not be exact enough for another. Without stating a particular purpose beyond that of a "blind" fit there is no way of deciding what is a satisfactory size for a coefficient.

Regardless, many people would like to have an absolute answer, without realising that this is equivalent to asking: "Precisely how inexact are we allowed to be?" If one is willing to tolerate inexactness in fit, one should also be willing to tolerate inexactness in the size of coefficient which should be regarded as acceptable.
The purpose of analysing empirical data such as correlation matrices is usually to find some empirical lawfulness with respect to the content of the variables.

The present **WSSA1** program do not take account of content. Indeed all existing computer programs of a similar nature have no content input. In the near future we hope to build content into the programs by use of facet theory. Currently, relating **WSSA1** output to content remains a separate operation, performed by hand after the computer finishes its presently assigned job. However the user can get facet diagram using **FACETS** paragraph in **WSSA1** section, when facet structuples are supplied for each variable. We will see such facet diagram related to the above example, and how they are related to the mapping sentence.

As a rule-of-thumb, an Alienation Coefficient of less than .15 is considered a good candidate for being "satisfactory". It has very often been found that subsequent inspection of content has revealed empirical lawfulness. Contrary cases have also occurred: systematic relations of the Space Diagram with the content of the variables have been observed, even when the Coefficient of Alienation is equal to .20 or more. Conversely, fits with coefficients less than .15 do not necessarily lead to a lawful relation with content.

Another rule-of-thumb is to see how the Coefficient of Alienation is reduced by adding dimensionality. Does adding a dimension reduce the coefficient "substantially"? (Again the temptation is to ask: "Precisely what is substantially?"). In the present example, the Alienation Coefficient is less than .15.

Input directives of the above example

Let us show now the entire input user directives for the **WSSA1** process of the matrix in Table 6.

```
$SET  LINESIZE = 80 ;
$MATRINP NAMES =   V1 TO V15  1-45 ;
         VARLABS =
                V1   'City as place to live'
                V2   'Neighborhood'
                V3   'Housing'
                V4   'Life in the U.S.'
                V5   'Amount of education'
                V6   'Useful education'
                V7   'Job'
                V8   'Spending of spare time'
                V9   'Health'
               V10   'Standard of living'
               V11   'Savings and investments'
               V12   'Friendships'
               V13   'Marriage'
               V14   'Family life'
               V15   'Life in general' ;
         FILE = 'STATES.MAT' ;
$WSSA1 NAMES =   V1 TO V15   ;
       MINDIM = 2 ;
       MAXDIM = 2 ;
       TITLE =
    'SSA on satisfaction with life in the United States' ;
       FACETS
             NFACETS = 2 /
             PROFILES =
                       V1    2 3
                       V2    2 3
                       V3    2 3
                       V4    2 3
                       V5    2 1
                       V6    2 1
                       V7    1 7
                       V8    1 4
                       V9    2 6
                       V10   1 2
                       V11   2 2
                       V12   2 4
                       V13   2 5
                       V14   1 5
                       V15   1 8   /
             DIAGRAMS =  2 1 1 2  &  2 2 1 2 ;
```

The **SET** section defines the size of the output line.

The main body of the job begins from **MATRINP** section. This section defines the names of the variables with their locations (**V1 TO V15 1-45**), the filename of the data matrix (**STATES.MAT**).

The **WSSA1** section invites HUDAP to process the smallest space analysis on all variables listed in the **MATRINP** section. In the paragraph **FACETS** a structuple of **2** facets are defined for each variable in **PROFILES** sentence, then **2** facet diagrams are requested by means of **DIAGRAMS** sentence. The first facet diagram is defined by the sequence "**2 1 1 2**" whose meaning is:

2 : dimensionality **2**
1 : first facet
1 2 : projection axis **1** versus axis **2**

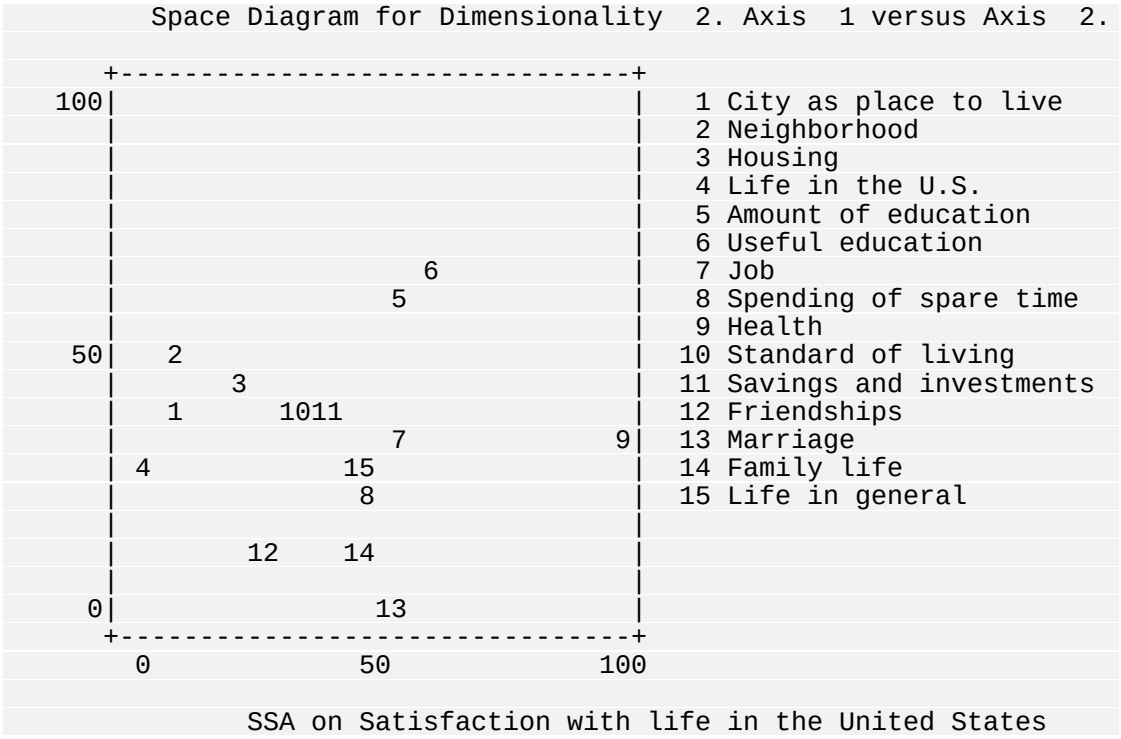


Figure 3
Two-dimensional space diagram. for matrix of Table 6.

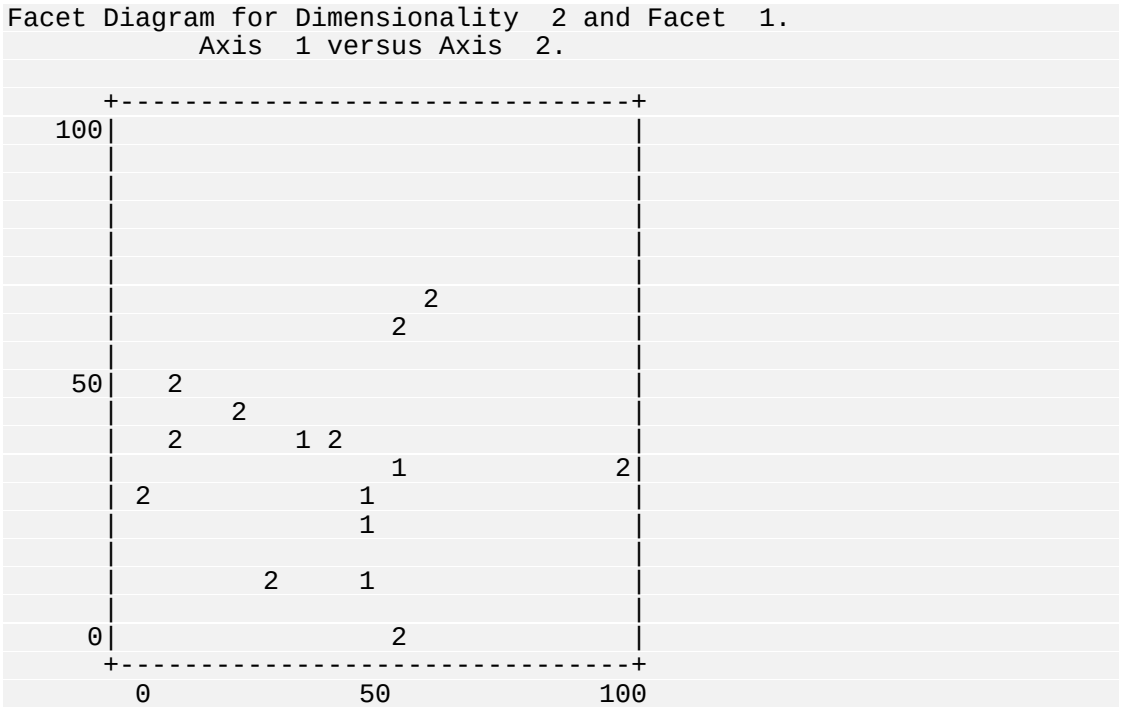


Figure 4
Facet diagram for facet no. 1 of example in Table 6.

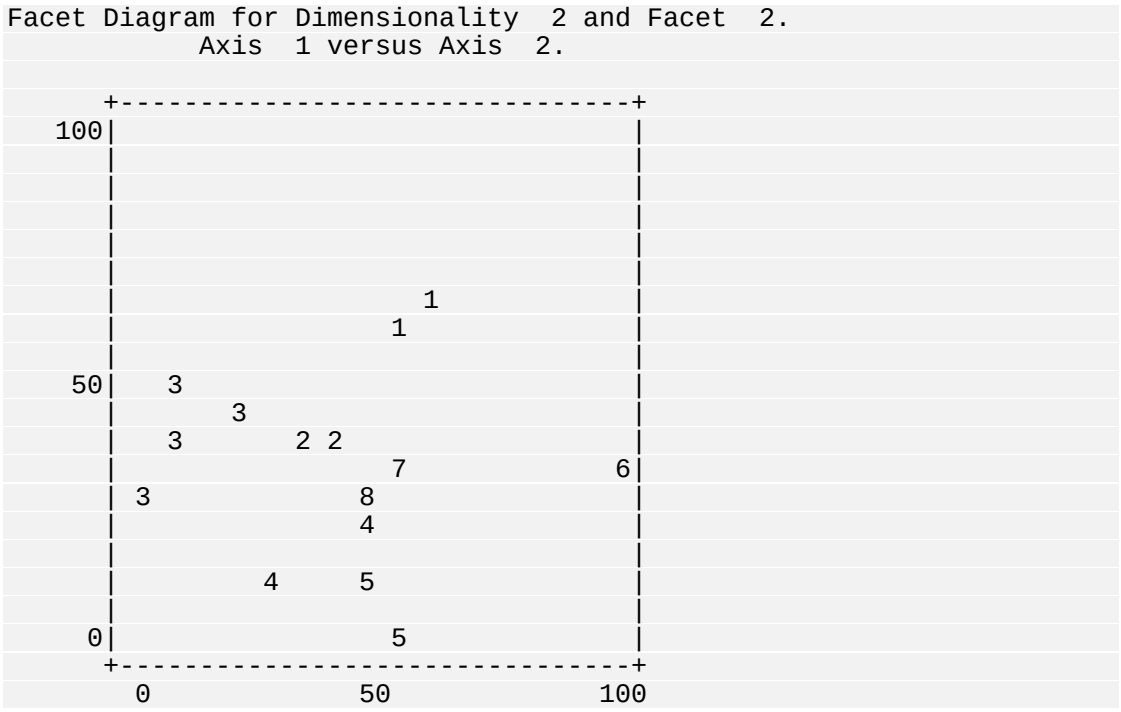


Figure 5
Facet diagram for facet no. 2 of example in Table 6.

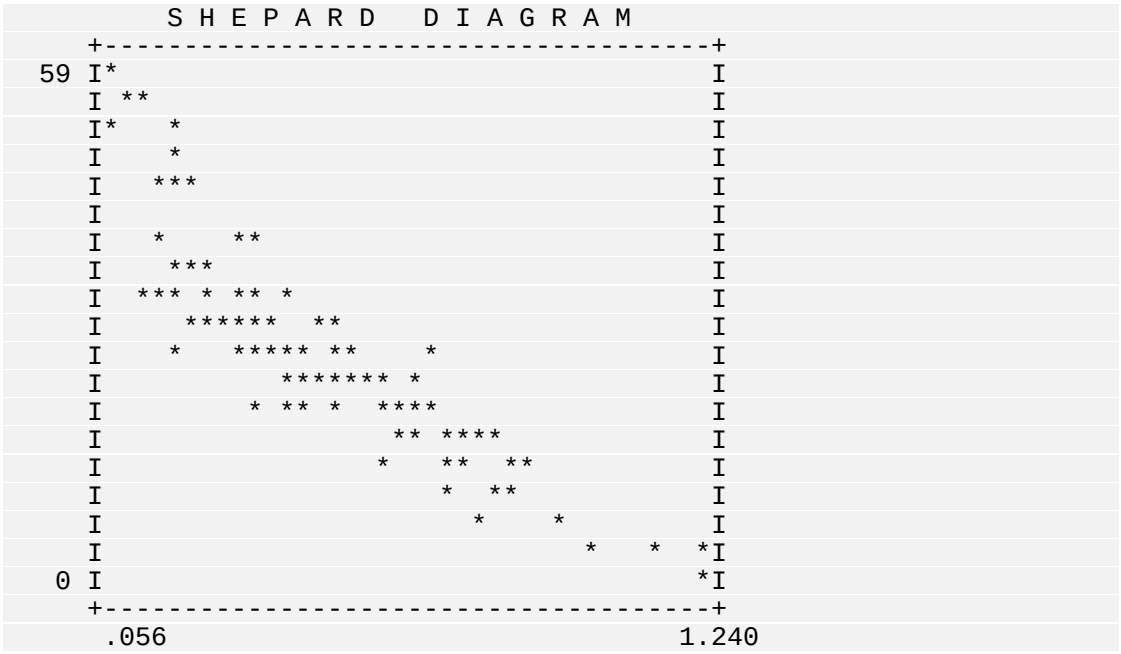


Figure 6
Shepard diagram for example in Table 6.

A similar study on certain aspects of quality of life was also conducted independently in Israel.

	1	2	3	4	5	6	7	8	9	10
1	100	50	17	17	35	65	38	38	49	42
2	50	100	-8	11	21	37	14	10	29	31
3	17	-8	100	49	57	39	27	26	58	26
4	17	11	49	100	57	35	20	29	31	33
5	35	21	57	57	100	55	35	34	57	51
6	65	37	39	35	55	100	47	42	66	49
7	38	14	27	20	35	47	100	83	49	42
8	38	10	26	29	34	42	83	100	48	42
9	49	29	58	31	57	66	49	48	100	51
10	42	31	26	33	51	49	42	42	51	100

Table 8.
Interrelationships (weak monotonicity coefficients)
among ten variables of satisfaction with life areas in Israel*.

1 Income
2 Housing
3 Health
4 Nervousness
5 Mood
6 General Situation
7 Job
8 Place of work
9 Personal life
10 Spending of spare time

*Special analysis of data from the Continuing Survey for March-April 1971 of the Israel Institute of Applied Social Research and the Communications Institute of the Hebrew University, on a sample of 1,620 Jewish urban adults.

Figure 7.
Two dimensional space diagram. for matrix of Table 8.

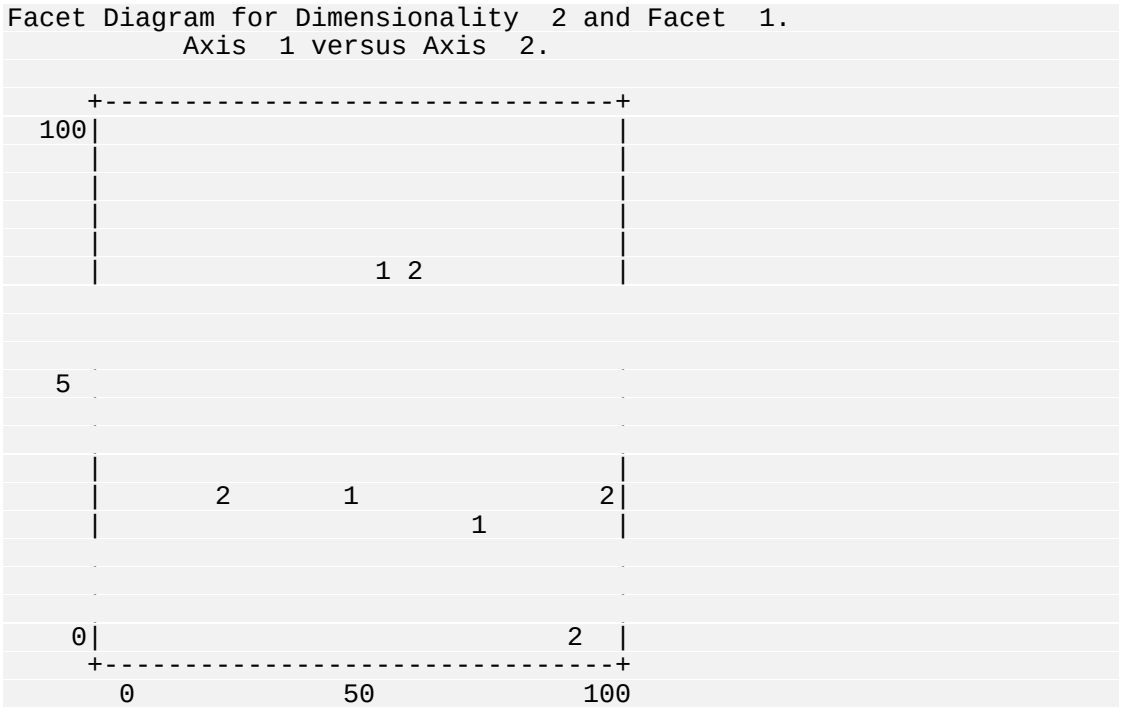


Figure 8.
Facet diagram for facet no. 1 of example in Table 8.

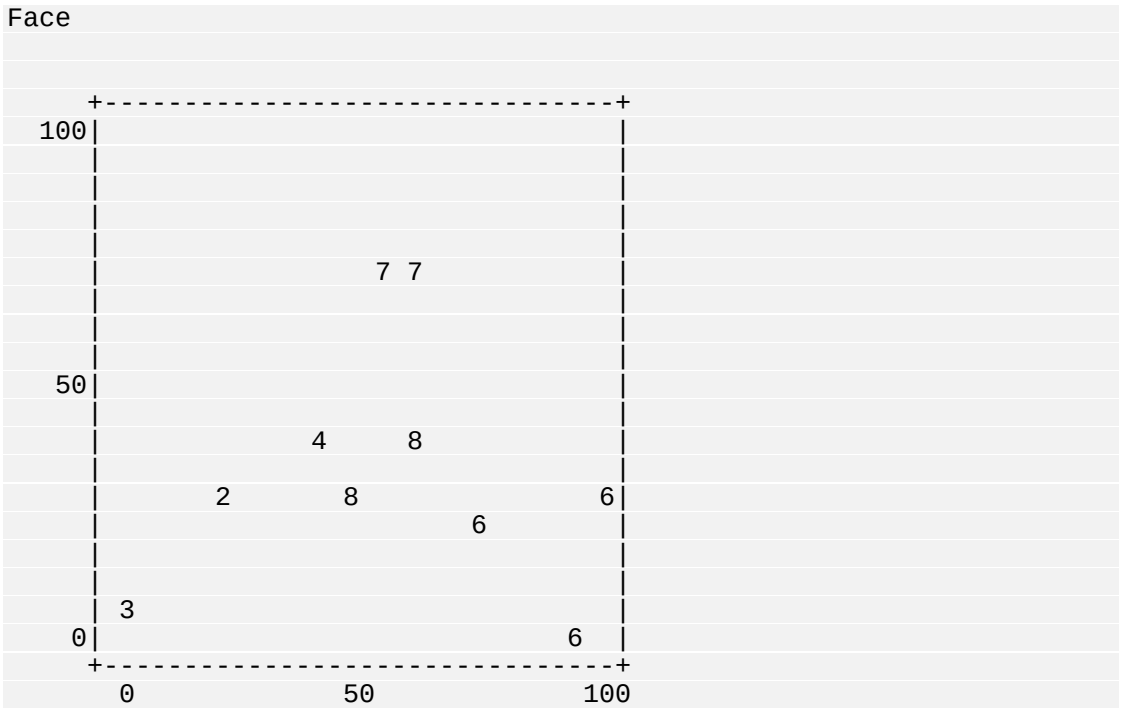


Figure 9.
Facet diagram for facet no. 2 of example in Table 8.

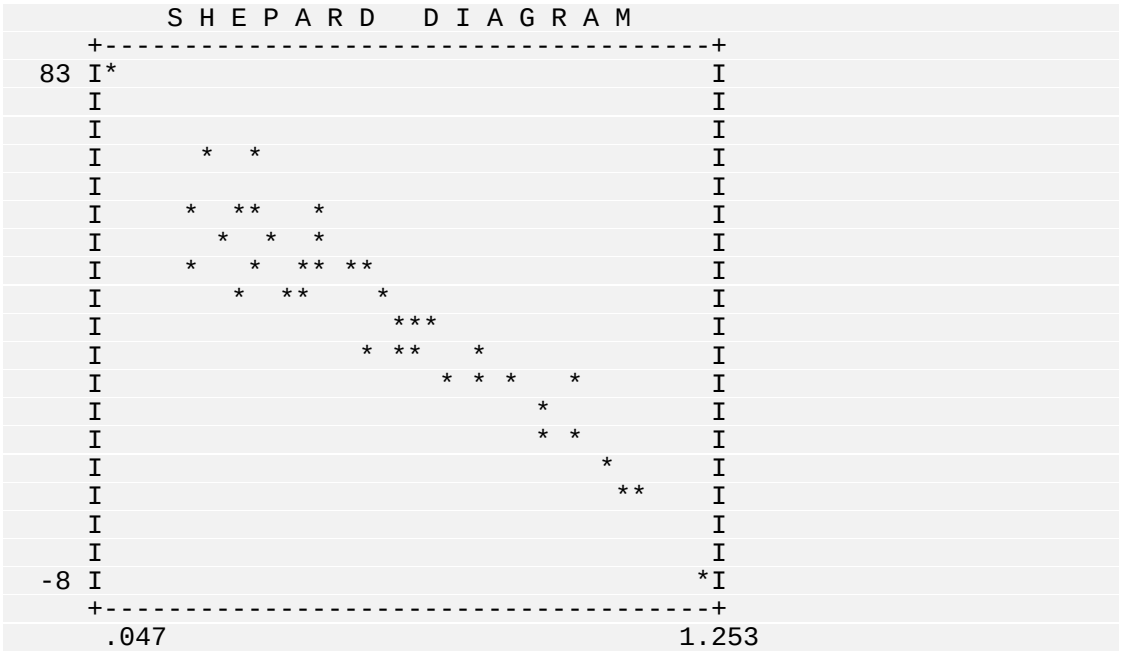


Figure 10
Shepard diagram for example in Table 8.

Use of the mapping sentence for coordinating theory and research: a cross-cultural example

Introduction

Formalization of the research process may be worthwhile only within the more general context of the formalization of substantive theory. Otherwise, the formalization - and the research itself - may become mere busywork. The need for joint formalization of theory and research is emphasised in Guttman's definition of theory: "A theory is an hypothesis of a correspondence between a definitional system for a universe of observations and an aspect of the empirical structure of those observations, together with a rationale for such an hypotheses" (in Gratch1, 1973, p. 35). Such a definition not only puts sharp focus on the necessity of defining the universe of observations to be researched, but also implies that the definitional system should be in a form that facilitates perceiving correspondences with aspects of the empirical data. Guttman's mapping sentence idea is intended to promote these two purposes (as well as many more): (a)

[1]Gratch, H. (Ed.) Twenty-Five Years of Social Research in Israel, Jerusalem: Jerusalem Academic Press, 1973.

definition of the universe of observations and (b) in a form that aids perception of systematic relationships with the data.

In effect, use of mapping sentences is a basic technique of facet theory (see David Canter[2], Facet Theory). Facet theory provides general strategies for developing fruitful specific theories in the above sense of "theory". In this study, We shall present but one simple example of the use of facets and the power of the mapping sentence approach. For some previously published examples see Aranya[3] et al. (1976), Elizur[4] (1970), Guttman (1959[5], 1970[6], 1971[7]), Kernberg[8] et al. (1972), Levy and Guttman (1974[9],1975[10]), Schlesinger and Guttman[11] (1969), Yalan[12], et al. (1972).

The Mapping Sentence

Two surveys on certain aspects of the quality of life were conducted independently of each other, and about the same time (spring and summer of 1971), one in the U.S. and one in Israel. Each contained some items asking about satisfaction with different aspects of one's life. The U.S. questionnaire contained fifteen items of this variety, as listed in Table 6. The Israeli questionnaire had ten such items, as listed in Table 8.

---

[2]Canter, D. (Ed.) Facet Theory:.Approaches to Social Research, New York: Springer-Verlag, 1985.

[3]Aranya, N., Jacobson, D. and Shye, S. Organizational and occupational commitment: A facet theoretical analysis of empirical data. Nederlands Tijdschrift voor de psychologie, 1976

[4]Elizur, D. Adaptation to Innovation, Jerusalem: Jerusalem Academic Press, 1970.

[5]Guttman, L. "A structural theory for intergroup beliefs and action," American Sociological Review, 1959, 24, 318-328.

[6]Guttman, L. Integration of test design and analysis. In Toward a Theory of Achievement Measurement, Proceedings of the 1969 Invitational Conference on Testing Problems, New Jersey: Princeton 1970, pp. 53-65.

[7]Guttman, L. "Social problem indicators," The Annals of the American Academy of Political and Social Science, 1971, 393, 40-46.

[8]Kernberg, O. et al. "Psychotherapy and Psychoanalysis: Final report of the Menninger Foundation's Psychotherapy Research Project," Bulletin of the Menninger Clinic, 1972, 36, 517-528.

[9]Levy, S. and Guttman, L. Values and Attitudes of Israeli High School Youth, Jerusalem: IIASR (in Hebrew, with English Introduction and Summary), 1974.

[10]Levy, S. and Guttman, L. Structure and dynamics of worries. Sociometry, 1975, 38, 445-473.

[11]Schlesinger, I.M. and Guttman, L. Smallest space analysis of intelligence and achievement tests. Psychological Bulletin, 1969, 71, 95-100.

[12]Yalan, E., Finkel, C., Guttman, L. and Jacobsen, C. The Modernisation of the Traditional Agricultural Village - Minority Villages in Israel. Rehovot: Settlement Study Centre, 1972.

The aspect of the empirical data to be discussed here is the matrix of intercorrelations among the variables. What is the structure of this matrix for the U.S., and what is the structure for Israel? To what extent are the two structures similar to each other?

Inspection of the two lists of items shows some overlap of content, according to their brief titles: housing, job, health, spending of spare time. Otherwise, these and the other items have different wordings in the respective surveys. Clearly, to compare the two structures requires establishing a common definitional framework which will transcend the particular wordings. Indeed, such a framework is needed to help study the structure for each country separately. A suitable framework is proposed in terms of the following mapping sentence:

The extent of satisfaction of respondent $\{x\}^{X}$ with the

$$\begin{Bmatrix} 1 \text{ state of} \\ 2 \text{ resources for} \end{Bmatrix}^{A} \text{ his activities in area of life } \begin{Bmatrix} 1 \text{ education} \\ 2 \text{ economics} \\ 3 \text{ residence} \\ 4 \text{ spare time} \\ 5 \text{ family} \\ 6 \text{ health} \\ 7 \text{ work} \\ 8 \text{ general} \end{Bmatrix}^{B}$$

$$\longrightarrow \begin{Bmatrix} \text{very positive} \\ \text{to} \\ \text{very negative} \end{Bmatrix}^{R} \text{ satisfaction with life}$$

This particular sentence has four explicit facets. The first facet - symbolised by "X" - designates the population of respondents being researched. The next two facets - labelled "A" and "B" - are for classifying the content of the items. Since the first of these content facets has two elements and the other has eight elements, together they suffice to define 16 (=2*8) varieties of items of satisfaction. The fourth and final facet - labelled "R" - expresses a common range for the response categories of the universe of items. This is also the range of the mapping sentence. Elements of the four facets are designated by small letters: *x*, *a*, *b*, and *r* respectively.

Abstraction and Substance

The research design expressed by the mapping sentence as a whole calls for assigning to each respondent (*x*) a value of the range (*r*) for each item (*ab*) classified by the two content facets of the domain. Abstractly, this set of assignments can be expressed by the mapping:

$$\mathbf{XAB} \longrightarrow \mathbf{R}$$

The left member, or domain, of the mapping is the Cartesian set **XAB**, giving all possible combinations (structuples) of the form *xab*, that is, of each member of the population **X** with each of the 16 content varieties of the Cartesian set **AB**. The arrow indicates the mapping into the set **R** of possible responses: each structuple *xab* of the domain has one and only one response in **R**.

While such an abstract formula for a mapping may suffice for mathematical purposes, it is insufficient for substantive theory and for empirical research design. Flesh is needed to cover this skeletal structure for purposes of substantive thinking and practical usage. Guttman's proposal is to add verbal connectives and further literary additions to make the mapping readable in ordinary language. Such a substantive elaboration of a mapping is what is called a Mapping sentence. It is actually a set of sentences of ordinary speech which have common connectives and which differ according to their facet elements.

Strategies of Modification

The emphasis of the present study is methodological, although we shall present an actual substantive theory. The latter can be improved on, especially with respect to facet **A**. Indeed, in related work, instead of dichotomous facet **A** we have used a facet "environment (internal, social, resource) and secondary environment (neighbourhood, town, state, world). The present facet **A** is a collapsing of this, since we have too few variables in the present data to document a theory about the more complete facet design. Let us just remark here that suggesting strategies for extension and intension of theories is an important feature of facet theory. Fruitful strategies are made possible by use of mapping sentences, since the latter lend themselves easily to correction, deletion, extension, and intension.

A Common Attitudinal Object: The First Law of Attitude

The fifteen U.S. and the ten Israel satisfaction items are easily classified by the two content facets **A** and **B.** Neither country's questionnaire contains all sixteen possible varieties of variables. Before going on to discuss the details of this domain classification and their implications, let us first focus on a central feature of the design: the common direction of "satisfaction" for the responses in **R**.

While the wording of response categories may differ from item to item, the categories of each item can be ranked from "very positive" to "very negative" expression of satisfaction. Actually each item is an attitudinal item: each conforms to the definition of attitudinal items as formulated by Guttman (Gratch, 1973, p. 36):

"An item belongs to the universe of attitude items if and only if

its domain asks about behaviour in a $\left\{\begin{matrix}\text{cognitive}\\\text{affective}\\\text{instrumental}\end{matrix}\right\}$

modality toward an object, and its range is ordered from

$\left\{\begin{matrix}\text{very positive}\\\text{to}\\\text{very negative}\end{matrix}\right\}$ toward that object."

This raises an interesting further question: do all these satisfaction items have a common object? If they do, then Guttman's First Law of Attitude should hold: the regression of any item on any other item should be monotone, and no correlation should have a negative sign (Gratch, 1973, p. 36).

Inspection of the correlation matrices of Tables 6 and 8 indeed shows that all correlations are positive or zero (the one slightly negative correlation may be regarded as a sampling error). From the wording of the domain of the mapping sentence, the immediate attitudinal object of each item is a particular area of life. However, all the items may be regarded as having a general object: life itself. According to such an interpretation, all the items do have a single attitudinal object in common as required by the First Law. This specification of a single common object is indicated by the phrase "satisfaction with life" following facet **R** in the above mapping sentence. Hence, we have a rationale for the First Law of Attitude to hold in this case for both the U.S. and Israel.

The Radex Theory of Satisfaction

Whether or not the First Law is appropriate, there is an apparent systematic correspondence between the two content facets and the empirical structures of the correlation matrices. Facets **A** and **B** help indicate which items should be more highly intercorrelated and which should have lower intercorrelations. This lawfulness is brought out by viewing the matrices through the eyes of Smallest Space Analysis, in particular **`WSSA1`**.

For each matrix, a two-dimensional geometrical space proves to give a rather good fit. The best fitting 2-spaces for the U.S. and Israel are shown in Figures 3 and 7 respectively. In each diagram, each item appears as a point, and two points tend to be closer together as the correlation increases between the two items involved. The spread of points in the diagrams shows a clear relationship to the classification of the items by facets **A** and **B**.

It turns out that the dichotomous facet **A** - of "state" versus "resources" - serves as a modulating facet, or corresponds to distance from an origin in the **`WSSA1`** space. The items closest to the origin, within the inner circle, assess the

satisfaction with the state of activities, while the outer band of items assess satisfaction with resources for activities. Facet diagrams in Figures 4 and 8 can help to detect this modulating structure.

Content facet **B**, of eight areas of life, serves as polarising facet. Its elements correspond to regions in the **WSSA1** space emanating from the origin and radiating outward, each in its own direction. Since these directions are all contained within a two-dimensional space, they have a circular ordering amongst themselves. This circular ordering turns out to be the same for the U.S. and Israel. (We have found the facet "areas of life" to play a polarising role in many other contexts as well, as reported at the 1973 annual meeting of the Israel Sociological Association.). Polarising structure can be seen in Figures 5 and 9.

Having two facets correspond to modulating and polarising partitions of an **WSSA1** space is a form of lawfulness called that of a Radex (Guttman, 1954; Schlesinger and Guttman, 1969; Levy and Guttman, 1974). In sum, then, the content facets of the mapping sentence have the same form of correspondence with the correlation matrices for the U.S. and for Israel, namely that of a radex.

Thus we hope to have illustrated how the mapping sentence device has facilitated seeing lawfulness which would otherwise be difficult to ascertain. We have in effect provided evidence for a radex theory for satisfaction with life. Having such a toehold can be a basis for systematic extensions and intensions and other fruitful modifications of the theory.

Some other features of WSSA1 procedure

DATA sentence

The main use of **WSSA1** is analysis of correlation matrix. A correlation coefficient is one kind of a more general family of relations: proximity or similarity. However, **WSSA1** can also process a matrix of distances or dissimilarities. In such a case, the user has to submit the following sentence:

```
DATA = DISSIM ;
```

The syntax of the sentence is:

DATA = <PROXIM | DISSIM> ;

where **PROXIM** is for similarities (e.g. correlations) **DISSIM** for dissimilarities (e.g. distances). The default is **PROXIM**.

TYING sentence

The algorithm of **WSSA1** is based on the following monotonicity condition:

$R_{ij} > R_{kl} \Leftrightarrow d_{ij} < d_{kl}$ for each quadruplet *(i,j,k,l)*.

where R_{ij} is the correlation (similarity) coefficient between two variables (v_i, v_j) and d_{ij} is the corresponding computed Euclidean distance between the two points representing v_i and v_j in the space. Note that the above monotonicity condition is for similarities. When the coefficients are dissimilarities (say D_{ij} instead of R_{ij}), the condition becomes:

$D_{ij} < D_{kl} \Leftrightarrow d_{ij} < d_{kl}$ for each quadruplet *(i,j,k,l)*.

In any case (similarity or dissimilarity), the monotonicity condition is based on a strict order of the input coefficients. The condition does not tell us what to do when there are tied values, that is $R_{ij} = R_{kl}$ or $D_{ij} = D_{kl}$ for some quadruplet(s) *(i,j,k,l)*. A set of tied matrix cells for the same value is named a tied class in **WSSA1** (by analogy of the mathematical concept of "equivalence class").

WSSA1 algorithm has chosen the following strategy to handle tied values: within each tied class in the matrix, the cells (R_{ij}) are ordered according to the order of the corresponding computed distances (d_{ij}) of the last iteration.

In practice, the user can decide that two coefficients are "equal" if they are "close" within a given tolerance. More precisely, if T is the tolerance, R_{ij} and R_{kl} are considered to be tied values (in the same class) if $|R_{ij} - R_{kl}| \leq T$. The T value can be submitted by the following sentence:

```
TYING = <value> ;
```

The default is T=0 ($R_{ij} = R_{kl}$).

The problem of tied values is critical when, there are big tied classes in the input matrix, for example a matrix of only 0's and 1's. Without a good treatment, the solution can be very bad.

WEIGHT sentence

The user can weight the input coefficients in order to emphasise a local or a global fit , where "local" means that more weight is given to error of fit for distances which should be relatively small, and "global" means more weight is given to error for distances which should be relatively large (e.g. between hypothesised clusters).

The parameter defining the weight is submitted by the following sentence:

WEIGHT = <value> ;

"**value**" must be an integer number (positive or negative). Let **value**=*p*, the weight w_{ij} assigned to the input cell D_{ij} is defined by:

$$w_{ij} = \left(\frac{1}{D_{ij}} \right)^p$$

TITLE sentence

The user can supply a title to be printed under the space diagram, by means of the following sentence:

TITLE = '<char>' ;

where "**char**" is a string up to 104 characters. See examples in Figures 3 and 7.

EXTNAMES sentence

For a detailed explanation and use of external variables in a Smallest Space Analysis, see later in this chapter. Meantime, the syntax is:

EXTNAMES = <var list> ;

where "**var list**" is the list of external variables to be located into the SSA space of the original variables defined in **NAMES** sentence.

OUTPUT paragraph

This paragraph is useful for transferring some results from HUDAP to other software like database or graphic programs. Besides the command(s), the user has to supply a name and a format (optional) of an external file for the storage of the requested information. Follow the differenr commands and sentences:

COORD command

Syntax:

COORD /

To output the computed variable coordinates

FACETS command

Syntax:

`FACETS /`

To output the user facets defined in **`FACETS`** paragraph.

DIST command

Syntax:

`DIST /`

To output the computed matrix of distances. These distances are normalised numbers between 0. and 1.

FORMAT sentence

Syntax:

`FORMAT = '<char>' /`

The format "**`char`**" specifies the layout (in fixed fields) of the output in a single row. The usual FORTRAN rules for the format are applicable, but only the F specification is allowed. "**`char`**" cannot exceed 80 characters. The default is (13F6.2).

FILE sentence

Syntax:

`FILE = '<char>' /`

"**`char`**" is the file pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

External variables

Definition

Let us define what is an external variable in a Smallest Space Analysis. We already know what is an SSA processed on a symmetric matrix of similarity coefficients (correlations) among a set of *n* variables, $V_1, V_2, \ldots, V_n$, called original

variables. The result is a graphic representation of these variables in a Euclidean space of a suitable dimensionality.

Let E be a variable, called external variable, which has a vector of similarity coefficients $\{R_j\}_{j=1,2,\ldots,n}$ with the original ones. The purpose is to represent the variable E as a point in the fixed configuration of the original space. This point is located in such a way that it satisfies the monotonicity condition as well as possible, that is $d_j < d_k$ whenever the observed data indicate that $R_j > R_k$, d_j being the Euclidean distance between two points representing E and V_j.

When the variable E has dissimilarity coefficients with the original variables, $\{D_j\}_{j=1,2,\ldots,n}$ instead of $\{R_j\}_{j=1,2,\ldots,n}$, the monotonicity condition becomes: $D_j < D_k \Leftrightarrow d_j < d_k$.

If there are several external variables, they are located one by one in the original space, thus, the interrelationships between external variables are not considered.

External variables can be of any kind, but the most interesting use is that of dummy variables which represent subgroups of the total population with different criteria.

Empirical example using external variables

The example presented here to illustrate the use of external variables is from a doctoral thesis on Jewish Identity and Identification submitted to the Hebrew University of Jerusalem by Shlomit Levy. The data were collected in the framework of the Continuing Survey of the Louis Guttman Institute of Applied Social Research in November 1992. The sample consisted of 593 urban Jewish adults. The issue analysed here is that of influence of Jewish Identity components on the Jewish identification of the respondents

The respondents were asked: "Some things have a lot influence and others only a little on your feeling that you are, or want to be, a part of the Jewish people. To what extent does each of the following influence your feeling in this respect". The respondents were asked to asses the extent of each of the components according to 4 scale answer categories: "influence a lot"; "influence"; "not so influence"; "not at all influence". Following is the list of the 11 identity components:

1 Jewish history of 3000 years
2 Recent Jewish world history
3 Holocaust
4 Recent Jewish history in Israel
5 Establishment of the State of Israel
6 Jewish religion
7 Jewish morality
8 Mutual assistance among Jews
9 Jewish tradition

10 Education at home
11 Attitude of non-Jews toward Jews

The content of these components can be classified according to 2 facets. One facet distinguishes among 3 types of identity components:

1) Process (history)
2) Ideology (i.e., religion, ethics)
3) Applying Ideology (mutual assistance, attitude of non-Jews)

The second facet specifies whether each type of component is of a general or selective nature.

These 2 facets can be incorporated in the following mapping sentence:

$$\text{The extent to which Israeli Jew } \{x\}^{X} \text{ attributes influence on the Jewish identification}$$

$$\text{of self to a } \left\{\begin{matrix}1\ \text{general}\\ 2\ \text{selective}\end{matrix}\right\}^{A} \text{ component of the type } \left\{\begin{matrix}1\ \text{process}\\ 2\ \text{ideology}\\ 3\ \text{applying ideology}\end{matrix}\right\}^{B}$$

$$\text{of the Jewish identity} \longrightarrow \left\{\begin{matrix}\text{very high}\\ \text{to}\\ \text{very low}\end{matrix}\right\}^{R} \text{ influence on Jewish identification.}$$

The eleven components can now be formally classified by the two content facets **A** and **B** of the above mapping sentence:

Item	Facet profile
1	$a_1\ b_1$
2	$a_2\ b_1$
3	$a_2\ b_1$
4	$a_2\ b_1$
5	$a_2\ b_1$
6	$a_1\ b_2$
7	$a_1\ b_2$
8	$a_2\ b_3$
9	$a_1\ b_2$
10	$a_1\ b_2$
11	$a_2\ b_3$

The mapping sentence includes also the population of the respondents symbolised by x and the range facet **R**, namely the set of response categories. The research design expressed by the above mapping sentence calls for assigning for each respondent x a value r of the range for each item (a b).

The eleven components were analysed by SSA. There is great distinction between the internal consistency problem (the structure of interrelations among the components) and the external prediction problem such as background traits. The new feature in SSA enables to integrate in the analysis "external" variables, while at the same time the internal structure remains unchanged. The trait chosen for external prediction in this example is ethnicity. The respondents were asked to specify the country of origin of self and his father. The response categories (respondent/his father) were:

1 Israel/Israel
2 Israel/East
3 Israel/West
4 East/East
5 West/West
6 any other combination

Let us describe the different steps to proceed in the HUDAP job (see Figure 11) in order to get an SSA with external variables.

Entering and describing the data

As usual, we begin with **DATA** section to enter the data and define missings and variable labels. 0 was chosen to code missing information. It was defined as missing value for all variables. Note that 1 and 6 were defined missing values for variable **ETHNIC** in order to reject them, even if they are valid information. The reason will be explained later.

Building external variables

From variable **ETHNIC** (ethnicity), we build the external variables (dummy variables). It was found that among the 6 categories of **ETHNIC**, 1 and 6 had relatively low frequency, therefore they were rejected, and 4 dummy variables **G2**, **G3**, **G4**, **G5**, were built from **ETHNIC** , corresponding to categories 2, 3, 4 and 5. For example, to built variable **G2**, first, **ETHNIC** is copied into **G2** by:

```
$COMPUTE G2=ETHNIC ;
```

thereafter, 2 is replaced by 1 ("Yes" Israel-East), and 3, 4, 5 are replaced by 2 ("Not" Israel-East) using:

```
$CODING NAMES = G2 ; REPLACE 2 BY 1 | 3 4 5 BY 2 ;
```

Note that the relevant category 2 in **ETHNIC** is coded by 1 in **G2**, while the other categories, 3, 4, 5, are coded by 2. The choice of this direction (1 for "Yes" 2 for "Not") is in accordance with the response categories of the original variables (1 for "A lot of influence" to 4 for "No influence at all"). As a rule "Yes" in the external variable corresponds to "High" in the original item.

In order to assign labels to the new variables, section **DATA** is called again, but only with **VARLABS** sentence.

Computing matrix of monotonicity coefficients

Once dummy variables are computed, a matrix of monotonicity coefficients among all variables (original and external) is computed and directed into memory.

Processing SSA

The **WSSA1** section takes the matrix from memory. The **NAMES** sentence defines the original (internal) variables, while the **EXTNAMES** sentence specifies the external variables from the whole matrix.

The dimensionality was restricted to 2 (**MAXDIM = 2**).

In the FACETS paragraph, 2 facets are submitted. For each original variable a facet profile is assigned. Finally, 2 facet diagrams are requested:

2 1 1 2	In dimensionality **2**, facet number **1**, projection **1 2**.
2 2 1 2	In dimensionality **2**, facet number **2**, projection **1 2**.

```
$SET  LINESIZE = 80 ;
$DATA NAMES = A31 TO A41 31-41 / ETHNIC 72 ;
      FILE = 'W222.DAT' ;
      MISSINGS = A31 TO A41 0 ETHNIC 0 1 6 ;
      VARLABS = A31 'J. history of 3000 years'
                A32 'Recent J. world history'
                A33 'Holocaust'
                A34 'Recent J. history in Isr'
                A35 'Establish. of the State'
                A36 'Jewish religion'
                A37 'Jewish morality'
                A38 'Mutual assistance'
                A39 'Jewish tradition'
                A40 'Education at home'
                A41 'Attitude of non-Jews' ;
{  Computation of external variables (dummy variables)
  from variable ETHNIC                                }
$COMPUTE G2=ETHNIC ;
   $CODING NAMES = G2 ; REPLACE 2 BY 1 | 3 4 5 BY 2 ;
$COMPUTE G3=ETHNIC ;
   $CODING NAMES = G3 ; REPLACE 3 BY 1 | 2 4 5 BY 2 ;
$COMPUTE G4=ETHNIC ;
   $CODING NAMES = G4 ; REPLACE 4 BY 1 | 2 3 5 BY 2 ;
$COMPUTE G5=ETHNIC ;
   $CODING NAMES = G5 ; REPLACE 5 BY 1 | 2 3 4 BY 2 ;
{  Assigning labels to external variables           }
$DATA VARLABS = G2 'Israel-East'
                G3 'Israel-West'

                G5 'West-West'    ;
{   Matrix of monotonicity coefficients between all
   variables (original and external)               }
$MONCO NAMES = A31 TO A41 G2 TO G5 ;
       OUTPUT MEMORY ;
{   SSA processing with external variables and
   facet diagrams                                  }
$WSSA1 NAMES = A31 TO A41 ;
       EXTNAMES = G2 TO G5 ;
       MAXDIM = 2 ;
       FACETS
           NFACETS = 2 /
           PROFILES =  A31 1 1
                       A32 2 1
                       A33 2 1
                       A34 2 1
                       A35 2 1
                       A36 1 2
                       A37 1 2
                       A38 2 3
                       A39 1 2
                       A40 1 2
                       A41 2 3  /
           DIAGRAMS = 2 1 1 2 & 2 2 1 2 ;
```

Figure 11
HUDAP job requesting external variables in **WSSA1**

INPUT MATRIX *

		1	2	3	4	5	6	7	8	9
J. history of 3000 years	1	100	70	55	46	51	53	38	34	54
Recent J. world history	2	70	100	75	70	70	46	46	46	52
Holocaust	3	55	75	100	53	78	32	42	38	37
Recent J. history in Isr	4	46	70	53	100	79	38	40	52	50
Establish. of the State	5	51	70	78	79	100	49	50	29	47
Jewish religion	6	53	46	32	38	49	100	74	58	83
Jewish morality	7	38	46	42	40	50	74	100	76	72
Mutual assistance	8	34	46	38	52	29	58	76	100	67
Jewish tradition	9	54	52	37	50	47	83	72	67	100
Education at home	10	53	53	47	56	58	72	71	61	77
Attitude of non-Jews	11	33	36	41	45	29	24	35	55	41

		10	11
Education at home	10	100	47
Attitude of non-Jews	11	47	100

INPUT EXTERNAL MATRIX**

		1	2	3	4	5	6	7	8	9
Israel-East	12	-22	-10	-25	8	-4	15	-10	2	11
Israel-West	13	-5	9	17	24	5	-21	-6	-1	2
East-East	14	-7	-16	-31	-1	-12	28	19	13	14
West-West	15	25	14	36	-21	10	-16	-4	-12	-21

		10	11
Israel-East	12	9	-6
Israel-West	13	-9	6
East-East	14	11	-6
West-West	15	-8	5

Figure 12
Input correlation matrices

```
                     D I M E N S I O N A L I T Y    2


Rank
Numb
Coef

Seri
Numb
----
   1
   2
   3
   4
   5
   6         30.60          7.09    30.12
   7         47.78          8.51    47.35
   8         64.40         14.06    63.19
   9
  10
  11



Seri
Numb
----
  12
  13
  14
  15
```

Figure 13
Space coordinates of original and external variables

```



  10


     |                                |   4 Recent J. history in Isr
     |                                |   5 Establish. of the State
     |                                |   6 Jewish religion
     |                                |   7 Jewish morality
     |     8                 13       |   8 Mutual assistance
     |          12                    |   9 Jewish tradition
   50|   7              4             |  10 Education at home
     |14       10                     |  11 Attitude of non-Jews
     |     9                 5        |   Ext. vars
     |                                |  12 Israel-East
     |   6             2     3        |  13 Israel-West
     |                                |  14 East-East
     |                                |  15 West-West
     |                                |
     |             1                  |
```

Figure 14
Two dimensional space diagram

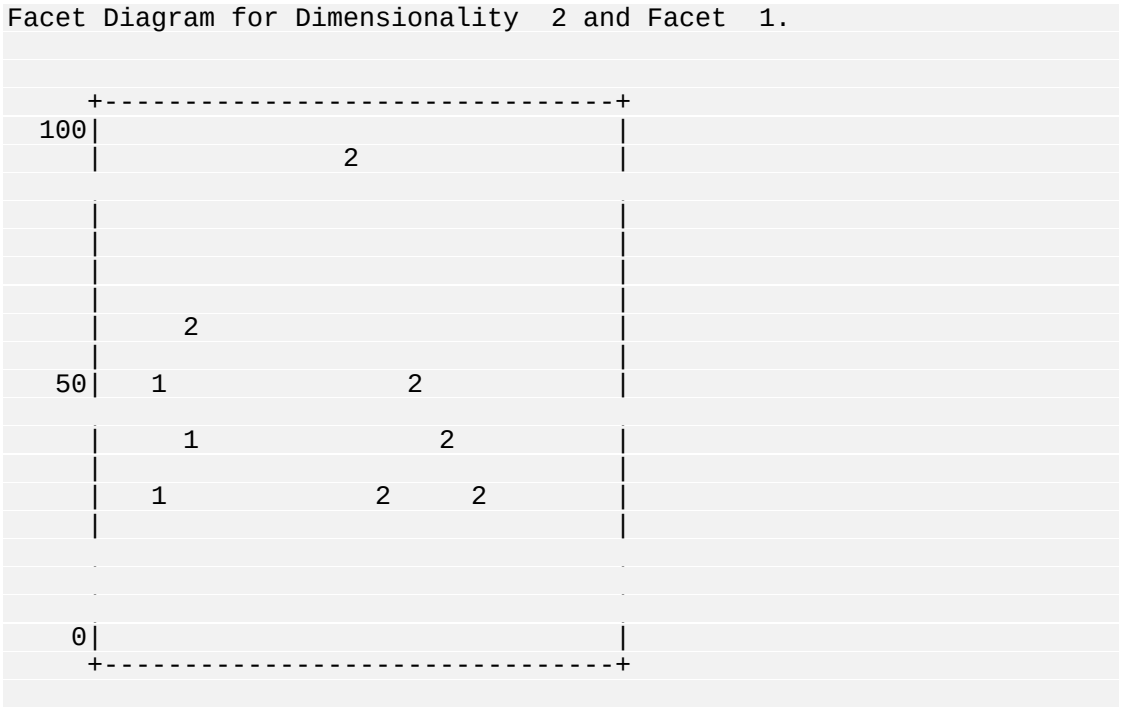


Figure 15
Facet **A** diagram

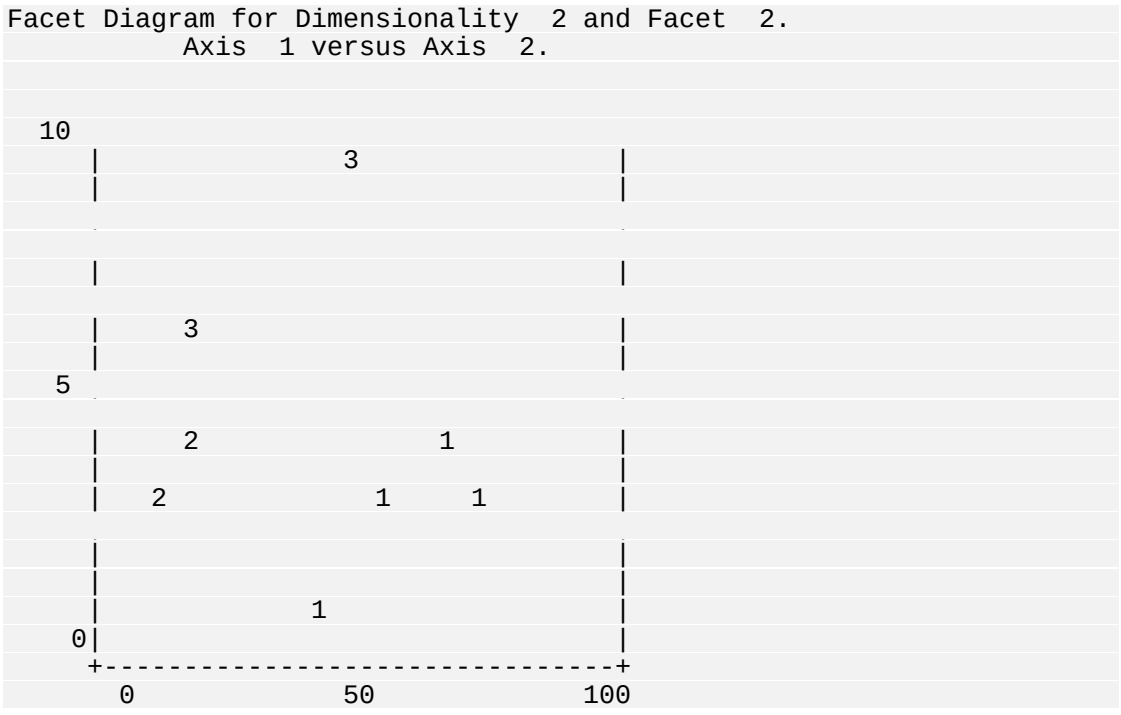


Figure 16
Facet **B** diagram

Analysis of the results

SSA showed that it is possible to present interrelations among the eleven original items in a two dimensional space with relatively good fit. The points are distributed in the space in a rectangular like shape where each direction (not to confuse with the space axes) correspond to one of the content facets. The direction from south-east to north-west is determined by the type of the identity component (facet **B**, Figure 16). Accordingly the space (Figure 14) is partitioned in this direction into 3 regions: history (items **1 2 3 4 5**), ideology (items **6 7 9 10**) and applying ideology (items **8 11**). This partition is indicated by the two lines (to be drawn in Figure 16) corresponding to this direction. The space is partitioned into two regions in the orthogonal direction, mainly from south-west to north-east according to the extent of generality of the component (facet **A**, Figure 15). The south-west region includes general components (items **1 6 7 9 10**) while the north-east region contains the components of selective nature (items **2 3 4 5 8 11**)..

Ethnicity is indicated in the space diagram (Figure.14) by 4 points (**12 13 14 15**) each of which corresponds to one variety of ethnicity. As evident from the figure, respondents of eastern origin (whether born in Israel or abroad; points **12** and **14**) are located quite closed to each other within the ideology region, namely closed to components of Jewish religion, tradition and ethics. In contrast to this the two western groups are distinct from each other. Respondents born abroad (point **15**) are located in the periphery in the south-east corner of the space. They are located at the corner of the history region close to two historical components, one of a general nature (3000 years of history) and the other of a selective nature (the Holocaust). Their Israel born offsprings however (point **13**) are located at the border of the other end of the history region close to the recent history of Israel.

Thus, while the Jewish identification of respondents of eastern origin (first and second generations) is influenced primarily by religious-traditional components, the Jewish identification of westerners is more influenced by historical components. Moreover, westerners born abroad identify almost alike with ancient history and contemporary Jewish history , while the identification of their Israeli offsprings is mainly through the recent history of the Jewish settlement in the land of Israel. In other words, the Israeli westerners are the most remote ethnic group from Jewish roots, both historical or traditional.

It should be noted however that the correlations between ethnicity and the original items are moderate or low

References

Amar, R. (2001) Mathematical Formulation of Regionality in SSA and POSAC/MPOSAC, in D. Elizur (Ed.): *Facet Theory: Integrating Theory Construction with Data Analysis*, Prague, pp. 63-74.

Amar, R. and Cohen, E.H. (2001) Hypothesized vs. Random Regionality in SSA, in D. Elizur (Ed.): *Facet Theory: Integrating Theory Construction with Data Analysis*, Prague, pp. 87-93.

Borg, I. and Lingoes, J. Multidimensional Similarity Structure Analysis, New York: Springer Verlag, 1987.

Canter, D. (Ed.) Facet Theory:.Approaches to Social Research, New York: Springer-Verlag, 1985.

Cohen, E.H. and Amar, R. (1993). External variables in SSA, including external profiles & POSAC regions. *Proceedings of the Fourth International Facet Theory Conference*, Prague, 375-385.

Cohen, E. H. and Reuven Amar (1999). External Variables in SSA and Unfolding Techniques: A Comparison. *Seventh International Facet Theory Conference, Design and Analysis.* (Ruth Meyer Schweizer Ed.). Berne, 259-279.

Guttman, L. A new approach to factor analysis: the Radex. In P.F. Lazarsfeld. (Ed.) Mathematical Thinking in the Social Sciences, Glencoe, Illinois: The Free Press, 1954, pp. 258-348..

Guttman, L. A general nonmetric technique for finding the smallest coordinate space for a configuration of points. Psychometrika, 1968, 33, 469-506.

Guttman, L. Integration of test design and analysis. In Toward a Theory of Achievement Measurement, Proceedings of the 1969 Invitational Conference on Testing Problems, New Jersey: Princeton 1970, pp. 53-65.

Guttman, L. An outline of some new methodology for social research. Public Opinion Quarterly, Winter, 1954-5, 18, 395-404.

Guttman, L. What is not what in statistics. The Statistician, 1977, 26, 81-107.*

Guttman, L. What is not what in theory construction. In R.M. Hauser, D. Mechanic and A. Haller (Eds.) Social Structure and Behaviour, New York: Academic Press, 1982, pp. 331-348.*

Guttman, L. Facet theory, smallest space analysis and factor analysis. Perceptual and Motor Skills, 1982, pp. 491-493.*

Guttman, L. What lies ahead for factor analysis? Educational and Psychological Measurement, 1958, 18, 497-515.

Guttman, L. and Guttman, R. A theory of behavioral generality and specificity during mild stress. Behavioral Science, 1976, 21, 469-477.

* To be found also in Levy, S. (Ed.) Louis Guttman on Theory and Methodology: Selected Writings, Aldershot: Dartmouth, 1994.

Levy, S. and Guttman, L. On the multivariate structure of wellbeing. Social Indicators Rsearch, 1975, 2, 361-388.

Levy, S. and Guttman, L. Structure and dynamics of worries. Sociometry, 1975, 38, 445-473.

Levy, S. (Ed.) Louis Guttman on Theory and Methodology: Selected Writings, Aldershot: Dartmouth, 1994.

Levy, S. The use of the mapping sentence for coordinating theory and research: A cross-cultural example. Quality and Quantity, 1976, 10, 117-125.

Levy, S. Lawful roles of facets in social theories. In D. Canter (Ed) Facet Theory: Approaches to Social Research, New York: Springer-Verlag, 1985, pp. 59-96.

Levy, S. The mapping sentence in cumulative theory construction: Well-being as an example. In J.J. Hox and J. De Jong-Gierveld (Eds.) Operationalization and Research Strategy, Amsterdam: Swets & Zeitlinger, 1990, pp. 155-177.

Lingoes, J.C. The Guttman-Lingoes Nonmetric Program Series, Ann Arbor, MI: Mathesis Press, 1973.

Lingoes, J.C., Roskam, E.E. and Borg, I. (Eds.) Geometric Representation of Relational Data: Readings in Multidimensional Scaling (2nd edition). Ann Arbor, MI: Mathesis Press, 1979.

Schlesinger, I.M. and Guttman, L. Smallest space analysis of intelligence and achievement tests. Psychological Bulletin, 1969, 71, 95-100.

Shapira, Z. and Zevulun, E. Performance rating in organization: A facet analysis interpretation of the multitrait multitraitor approach. Multivariate Behavioral Research, 1989, 24, 209-232.

Shye, S. (Ed.) Theory Construction and Data Analysis in the Behavioral Sciences, San Francisco: Jossey-Bass, 1978.

"Facet Theory." Applied Psychology: An International Review, 1990, 39, 4. (Special issue)

WSSA1 section menu

NAMES = <var list> ;
Names of the variables defining the input symmetric matrix.

MINDIM = <value> ;
The minimal dimensionality from which **WSSA1** is processed. "**value**" must be a positive integer number.
The default of **MINDIM** is 2.

MAXDIM = <value> ;
The maximal dimensionality at which **WSSA1** is processed. "**value**" must be a positive integer number.
The default of **MAXDIM** is 3.

DATA = <PROXIM | DISSIM> ;
Kind of matrix data . **PROXIM** for similarities (e.g. correlations) **DISSIM** for dissimilarities (e.g. distances)
The default is **PROXIM**.

TYING = <value> ; (value : T)
Tolerance **T** for tied cell values. This means that if the absolute value of the difference between two coefficients is less than **T** (or equals **T**) then the program will consider them as "equal".
The default is **T = 0**.

WEIGHT = <value> ;
Parameter allowing the user to weight the input coefficients in order to emphasise a local or a global fit , where "local" means that more weight is given to error of fit for distances which should be relatively small, and "global" means more weight is given to error for distances which should be relatively large (e.g. between hypothesised clusters). "**value**" must be an integer number (positive or negative)

EXTNAMES = <var list> ;
To locate external variables listed in "**var list**" into the SSA space of the original variables defined in **NAMES** sentence.

TITLE = '<char>' ;
"**char**" is a user supplied string up to 104 characters which will be printed under each projection of the plot configuration.

FACETS paragraph

Paragraph defining the sentences concerning information about variable facets and facet diagrams.
The following sentences have to be submitted in the order they appear here.

NFACETS = <value> / (value : k)
k is the number of facets. It must be an integer number.

```
PROFILES = <variable_1 val list
            variable_2 val list
            ..................
            variable_N val list>  /
```

variable_1, variable_2, ..., variable_N are the variable names listed in the sentence **NAMES**. All the variable names evoked in this sentence have to appear here. "**val list**" is a value list of **k** integer numbers : $n_1, n_2 \ldots, n_k$ where n_i is the category of facet no. **i**

```
DIAGRAMS = <n11 n21 n31 n41  &
            n12 n22 n32 n42  &
            ..................
            n1d n2d n3d n4d>  ;
```

For diagram number **j** the meaning of the 4 numbers is :

n1j : specific dimensionality
n2j : specific facet no.
n3j : first projection axis
n4j : second projection axis

OUTPUT paragraph

Paragraph defining the output of various objects on a file.

COORD /
To output the computed variable coordinates.

FACETS /
To output the user facets defined in **FACETS** paragraph.

DIST /
To output the computed matrix of distances. These distances are normalised numbers between 0. and 1.

FORMAT = '<char>' /
The format "**char**" specifies the layout (in fixed fields) of the output in a single row. The usual FORTRAN rules for the format are applicable, but only the F specification is allowed. "**char**" cannot exceed 80 characters.
The default is (13F6.2).

FILE = '<char>' /
"**char**" is the file pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

The *POSAC* Section

Introduction to POSAC

The name **POSAC** comes from: **P**artial **O**rder **S**calogram **A**nalysis with base **C**oordinates

The **POSAC** procedure provides a two-dimensional representation of a partial-order scalogram for a set of *N* profiles, based on *n* variables (items). The categories of these variables can be any subset of the integers from 0 to 99, and the category range may vary for different variables. However, the ranges of the variables must have the same meaning (not necessarily the phrasing) and be uniformly ordered with respect to a common content criterion (for example, category "1" may represent "low" with respect to that criterion, in all variables). If, in order to meet this requirement original categories need to be reordered or collapsed, these changes must be done by **CODING** section before running **POSAC** procedure.

A partial order on a set of profiles is defined by two relations: comparability (>, < or =) and incomparability (symbolised by "#", to be not confused with inequality symbol "$\neq$"). Let $p=(p_1,p_2,\ldots,p_n)$ and $q=(q_1,q_2,\ldots,q_n)$ be two profiles. p is said to be greater than q ($p>q$), if $p_i \geq q_i$ for all i's and there is at least one i for which $p_i > q_i$ (strictly). p is said to be incomparable to q ($p\#q$), if for some i's $p_i \geq q_i$ while for other i's (at least one) $p_i < q_i$ (strictly). Example (for profiles based on seven variables, n=7):

$$\left.\begin{matrix} p = 4343422 \\ q = 4223312 \end{matrix}\right\} \longrightarrow p > q$$

$$\left.\begin{matrix} p = 2422343 \\ q = 3432323 \end{matrix}\right\} \longrightarrow p\#q$$

Let **P** be the partially ordered set of profiles on *n* variables, **POSAC** algorithm tries to reduce the number of variables from *n* to 2.

More precisely, to each profile $p=(p_1,p_2,\ldots,p_n)$ in **P** will correspond a profile (x_p,y_p) in $\mathbf{R}^2$ (the Cartesian plane) such that, through this reduction the partial order in **P** is preserved as well as possible. *x* and *y* are called base coordinates.

A detailed description of **POSAC** algorithm is given in Appendix A.

Furthermore, for specified categories of a given external variable (a variable not in the original set of the *n* variables) representing an external criterion or trait, trait-diagrams are presented depicting the proportion of subjects possessing that trait among all those sharing the same profile in *n* variables (see **EXTMAPS** sentence). The trait may be specified also as a combination (intersection) of response categories from different external variables .

Treatment of missing values in POSAC

What happens when there are missing values in some profile? The relation between this profile and others is no longer clear. Let us describe how **POSAC** handles this problem. Assume we have two profiles *p* and *q* with missing values. The idea is to suppress from *p* and *q* the items with missing values and compare the rest. If the partial profiles are incomparable, **POSAC** states that *p* and *q* are incomparable. If the partial profiles are comparable, **POSAC** fixes the relation between *p* and *q* as unknown and bypasses the contribution of this pair from the process of finding the solution. For example:

$$\left.\begin{array}{l} p = 4343422 \\ q = 422\text{m}312 \end{array}\right\} \longrightarrow p,q : \text{the relation is unknown because} \left.\begin{array}{l} p' = 434422 \\ q' = 422312 \end{array}\right\} \longrightarrow p' > q'$$

$$\left.\begin{array}{l} p = 3432\text{m}23 \\ q = 2\text{m}22343 \end{array}\right\} \longrightarrow p \# q \text{ because} \left.\begin{array}{l} p' = 33223 \\ q' = 22243 \end{array}\right\} \longrightarrow p' \# q'$$

POSAC section

The **POSAC** procedure is called and activated by the control word **POSAC** followed by the required sentences. The variables are entered in the **NAMES** sentence in the usual manner. A simple example of a HUDAP **POSAC** job:

```
$DATA
      NAMES = Id1 Id2 1-8 (A) Murder Rape Assault
              Robbery Burglary Larceny Auto_Theft 10-16 ;
      FILE = 'CRIME.DAT' ;
$POSAC
      NAMES = Murder TO Auto_Theft ;
```

But many other options are available in **POSAC** section. The following paragraphs explain the various possibilities.

LABEL sentence

In some **POSAC** runs, the user is interested in identifying each subject. This can be done by specifying among all the variables, one or two labelling variables in the **DATA** section, and using the **LABEL** sentence in **POSAC** section. The labelling variable(s) must be alphanumeric and must not exceed 4 characters each.

Example:

```
$DATA
      NAMES = Id1 Id2 1-8 (A) Murder Rape Assault
              Robbery Burglary Larceny Auto_Theft 10-16 ;
      FILE = ..........  ;
$POSAC
      NAMES =   ........  ;
      LABEL = Id1 Id2 ;
          ...................
          ...................
```

See output example in Figure 1.

PROCESS sentence

Sometimes the user wants to see the list of the scalogram profiles before processing the partial order computations. This list can help him to see what changes must be done on the original categories, in order for example, to reduce the number of profiles. He can decide to reject a specific variable from the scalogram. To get only the list of profiles, the user may specify:

```
PROCESS = PARTIAL ;
```

FREQ sentence

Usually, the input data of **POSAC** procedure is raw data, where each case represents a subject. From this raw data, **POSAC** computes the frequencies of the different profiles. Then, on these profiles, **POSAC** processes the partial order computations to find the base coordinates. But, it can occur that the user has his profiles already computed with their frequencies, and from these profiles he wants to process **POSAC**. In this case, the user can introduce the frequency profile as a variable in **DATA** section and specify this variable in **POSAC** section by the sentence:

```
FREQ = <variable> ;
```

LOWFREQ sentence

Profiles can be rejected from the process with regard to their frequency value. If

```
LOWFREQ = <value> ;
```

is submitted, **POSAC** will retain only profiles whose frequency is greater than "**value**". This value must be a positive integer number. The default is 0.

EXTMAPS sentence

POSAC permits the inclusion of external variables, in addition to the (internal) scalogram one. The external variables are not included in structuring the scalogram itself but their relationship to the scalogram and its axes can be assessed and depicted.

External variables are typically background or demographic variables or external criteria for "validating" the phenomenon studied in the scalogram itself. They are used in the **POSAC** procedure to define an external trait. Thus, for example, the external trait "young educated female" is defined by selecting the appropriate categories from the external variables age, education, and sex. For example, this external trait can be described in HUDAP language by :

```
AGE 0 TO 20 EDUC 10 TO 15 SEX 2 ;
```

For each external trait correspond 2 external maps. See later for more details on external maps. External maps are obtained by sentence **EXTMAPS**. The syntax is:

```
EXTMAPS = <variable val list
            ...      ...
          variable val list  &
          variable val list
            ...      ...
          variable val list  &
            ...      ...
            ...      ...
          variable val list
            ...      ...
          variable val list> ;
```

The delimiter "**&**" separates the definition of two external maps. For example :

```
EXTMAPS =
          SEX 1  AGE 0 TO 40  &
          SEX 2  AGE 0 TO 40  ;
```

Here two external maps are requested for two traits, males aged 40 or less and females aged 40 or less.

NVALID sentence

When, in a case some of the items are missing, the relation of its profile with any other is ambiguous. The user can decide what is the number of non-missing items for which a case is not rejected. The syntax is:

```
NVALID = <value> ;
```

The default is 2.

Let us describe how **POSAC** treats profiles with missing items. Suppose there are two profiles p and q containing missing values. The idea is to suppress from p and q, items with missing values and to compare the remaining (partial) profiles. If the partial profiles are incomparable, **POSAC** treats p and q as incomparable. If the partial profiles are comparable, **POSAC** leaves the relation of p and q as unknown and performs a special treatment for the couple. Example:

$$\left.\begin{array}{l} p = 4343422 \\ q = 422m312 \end{array}\right\} \longrightarrow p,q : \text{unknown relation, because } \left.\begin{array}{l} p = 434422 \\ q = 422312 \end{array}\right\} \longrightarrow p > q$$

$$\left.\begin{array}{l} p = 2m22343 \\ q = 3432m23 \end{array}\right\} \longrightarrow p\#q \quad \text{because } \left.\begin{array}{l} p = 22243 \\ q = 33223 \end{array}\right\} \longrightarrow p\#q$$

In the above examples, "*m*" stands for missing element.

In all lists where profiles appear, **POSAC** prints "**-1**" instead of a missing value.

INSERT Paragraph

It is clear that the most important results of **POSAC** are the base coordinates **X** and **Y** and/or the transformed ones **J** and **L** (**J=X+Y**; **L=100+X-Y**). In some researches, the user needs these coordinates as variables among all his other variables, in order to pursue his analysis. According to the user directives, **X**, **Y**, **J** and **L**, or part of them can be added in memory as regular variables. The syntax is:

```
INSERT
       target variable = <X | Y | J | L> /
```

This sentence can be repeated for each of the 4 coordinates. "**target variable**" can be an old (existing) name or a new one.

Example of the use of **INSERT** paragraph in a HUDAP job:

```
$DATA
      NAMES = V37 TO V40 37-40 Sex 64 Origin 72;
      MISSINGS = V37 TO Origin 0 ;
      FILE = 'SH5556' ;
$CODING
      NAMES = V37 TO V40 ;
      REPLACE 4 BY 3 ;
$POSAC
      NAMES = V37 TO V40 ;
      INSERT
          VarX = X / VarY = Y / VarJ = J / VarL = L ;
$DISCO
      NAMES = VarX TO VarL ;
      GROUPS = Sex 1 | 2 ;
$DISCO
      NAMES = VarX TO VarL ;
      GROUPS = Origin 1 | 2 | 3 | 4 | 5 | 6  ;
      CONTRAST ;
```

Example of POSAC job with output

Follow a HUDAP job requesting **POSAC** on a datafile named **'CRIME.DAT'**

```
$SET LINESIZE = 80 ;
$DATA
      NAMES = Id1 Id2 1-8 (A) Murder Rape Assault
              Robbery Burglary Larceny Auto_Theft 10-16 ;
      FILE = 'CRIME.DAT' ;
$POSAC
       NAMES = Murder TO Auto_Theft ;
       LABEL = Id1 Id2  ;
```

The file **CRIME.DAT** (see the next figure) contains information on degree of severity on seven kinds of crimes in sixteen American cities.

The example presents sixteen American cities[1] with seven different kinds of crime serving as the criteria for stratification. The crime rates serve as structs for characterising the cities. Each city has a profile consisting of seven structs (crime rates), one struct for each kind of crime: murder, rape, assault, robbery, burglary, larceny and auto-theft. Each city was assigned one of four ranks on each crime item: "1" indicating the lowest rate, and "4" indicating the highest crime rate on that item. The stratification of these sixteen cities among themselves is, of course, automatically a part of the stratification of all the cities in the United States, since the partial order of a subset does not depend on the partial order of the total set; the internal ordering holds regardless of the traits of cities not in the sample.

[1] The data come from the United States Statistical Abstract, 1970, table 9.9
The hand made partial order analysis of these data was published by Shlomit Levy in 1985. Social Indicators Research 16:195-197

From the definitions of the seven crimes, it is possible to classify them into 3 distinct subsets: crimes against *person* (murder rape assault) and crimes against *property* (larceny, auto-theft) with robbery and burglary in between.

```
Atlanta  4221221
Boston   1111114
Chicago  3233112
Dallas   4343422
Denver   2422343
Detroit  3334333
Hartford 1111111
Honolulu 1111332
Houston  4223312
Kansas_C 3432323
Los_Ange 2443444
New_Orle 2332233
New_York 2234444
Portland 1221331
Tucson   1221211
Washing. 3234333
```

CRIME.DAT: Input datafile

Case identification

```
Numb

Numb
Numb
Numb

There are  17 different profiles

 Id   Profile        Sco  Freq   User Id
 --   -------        ---  ----   -------
      M R A R B L A
      u a s o u a u
      r p s b r r t
      d e a b g c o
      e   u e l e _
      r   l r a n T
          t y r y h
              y   e

 12   4 2 2 1 2 2 1   14     1  Atlanta
 16   1 1 1 1 1 1 4   10     1  Boston
 11   3 2 3 3 1 1 2   15     1  Chicago
  4   4 3 4 3 4 2 2   22     1  Dallas
  8   2 4 2 2 3 4 3   20     1  Denver
  5   3 3 3 4 3 3 3   22     1  Detroit
 17   1 1 1 1 1 1 1    7     1  Hartford
 14   1 1 1 1 3 3 2   12     1  Honolulu
 10   4 2 2 3 3 1 2   17     1  Houston
  7   3 4 3 2 3 2 3   20     1  Kansas_C
  2   2 4 4 3 4 4 4   25     1  Los_Ange
  9   2 3 3 2 2 3 3   18     1  New_Orle
  3   2 2 3 4 4 4 4   23     1  New_York
 13   1 2 2 1 3 3 1   13     1  Portland
 15   1 2 2 1 2 1 1   10     1  Tucson
  6   3 2 3 4 3 3 3   21     1  Washing.
  1   4 4 4 4 4 4 4   28     1  NOBODY
```

Figure 1

The output of **POSAC** section begins with information on some parameters . Then follows a list of all profiles, each with the user-assigned identification as requested by **LABEL** sentence (see Figure 1). This list is given only if the number of cases does not exceed 200. When **LABEL** sentence is not specified, the serial number of the case is printed instead of the User Id.

Comparability relations

```
There are  17 different profiles


 Id   Profile         Sco  Freq   Greater than profiles Smaller than p
 --   -------         ---  ----   --------------------- --------------
      M R A R B L A
      u a s o u a u
      r p s b r r t
      d e a b g c o
      e   u e l e _

          t y r y h
              y   e

  1*  4 4 4 4 4 4 4   28     1    2
  2   2 4 4 3 4 4 4   25     1    8                        1
  3   2 2 3 4 4 4 4   23     1   13                        1
  4   4 3 4 3 4 2 2   22     1   10                        1
  5   3 3 3 4 3 3 3   22     1    6                        1
  6
  7
  8
  9
 10   4 2 2 3 3 1 2   17     1   15                        4
 11   3 2 3 3 1 1 2   15     1   17                        6
 12   4 2 2 1 2 2 1   14     1   15                        4
 13   1 2 2 1 3 3 1   13     1   15                        8
 14   1 1 1 1 3 3 2   12     1   17                        8
 15   1 2 2 1 2 1 1   10     1   17                       13
 16   1 1 1 1 1 1 4   10     1   17                        3
 17   1 1 1 1 1 1 1    7     1                            15 16


   *Extreme profile added by program
```

Figure 2

In Figure 2, for each profile, the following information is given:

Id : a serial number assigned by the procedure for identification.
Profile : the response categories for a specific case or cases.
Sco : profile score which is the sum of the profile elements.
Freq : the profile frequency, number of cases having the same profile.
Greater than : list of profiles smaller than **Profile**.
Smaller than : list of profiles greater than **Profile**.

The lists given in columns "**Greater than**" or "**Smaller than**", are restricted to profiles of near level, since the comparability relation is transitive. A level is a set of profiles which have the same score. For example profile number **2** in level **25** is reported to be greater than profile number **8** of level **20**. But **8** itself is greater than **13** which is greater than **15** etc....

The maximal profile in a scalogram is a profile whose elements are the highest in all items. The minimal profile is a profile whose elements are the lowest in all items. When the maximal and/or minimal profiles are not in the input scalogram, **POSAC** adds them, and prints a star ("*****") near their Id's.

The listing of the profiles of the cities according to the crime rates is presented in Figure 2 above. The lowest profile rank on crime is "**1111111**" and characterises the city of Hartford, the highest possible profile is "**4444444**" but did not occur empirically in this sample of cities. It is marked by a star in the table. As already explained, one profile is higher (in this case on crime) if and only if it is higher on at least one criterion and not lower on any other criterion. For example consider profile "**3234333**" for Washington and "**3334333**" for Detroit. These two cities are comparable, with Detroit being higher on crime than Washington. Detroit is higher than Washington on the particular crime "rape" and both are equal (and high) on each of the rest of the crimes. To remind, two profiles are incomparable if and only if one profile is the higher on at least one struct while the other profile is also the higher on at least one struct. For example, consider the profile "**1221331**" for Portland against profile "**4221221**" for Atlanta.

Monotonicity coefficients matrix between the items

```
         COEFFICIENTS OF WEAK MONOTONICITY
                BETWEEN THE ITEMS


                    1    2    3    4    5    6    7
              +-------------------------------------
              I
Murder    1 I 1.00
              I
Rape      2 I  .53 1.00
              I
Assault   3 I  .72  .88 1.00
              I
Robbery   4 I  .71  .55  .92 1.00
              I
Burglary  5 I  .44  .79  .78  .77 1.00
              I
Lar

Auto_The  7 I -.01  .56  .53  .72  .48  .68 1.00
```

Figure 3

POSAC section gives a matrix of weak monotonicity coefficients among all scalogram variables (Figure 3). This can be a helpful information for the user.

Goodness of fit

```
Number of iterations ........  17
Time of last iteration ......  .109  seconds

Proportion of profile-pairs CORrectly REPresented
CORREP coefficient ..........  .8750  (=    119 /    136)

Proportion of comparable pairs CORrectly REPresented
CORREP1 coefficient .........  .9365  (=     59 /     63)

Proportion of incomparable pairs CORrectly REPresented
CORREP2 coefficient .........  .8219  (=     60 /     73)

SCOre--DIStance weighted coefficient
SCODIS coefficient ..........
```

Figure 4

As explained above, the aim of **POSAC** is to reduce the number of items from *n* to 2, preserving the partial order as well as possible. It is clear that for, empirical data, this reduction introduces some error in the partial order. So, we need some coefficient which measures the goodness of fit. **POSAC** gives 4 such coefficients.

For input profiles, we note:

M_C : number of comparable profile pairs

M_I : number of incomparable profile pairs

A similar notation holds for output (*x*,*y*):

m_C : number of comparable (*x*,*y*)-profile pairs

m_I : number of incomparable (*x*,*y*)-profile pairs

If *N* is the total number of profiles in the scalogram then we have:

$M_C + M_I = \frac{1}{2}N(N-1)$: number of profile pairs.

Now, we can give the exact definition of the coefficients which appear in Figure 4:

$$\text{CORREP} = \frac{m_C + m_I}{M_C + M_I}$$

$$\text{CORREP1} = \frac{m_C}{M_C}$$

$$\text{CORREP2} = \frac{m_I}{M_I}$$

There is another coefficient, SCODIS, which is related to the loss function of **POSAC** algorithm. It is less useful than the others for the user.

Base coordinates

Id	Profile							Sco	Freq	X	Y	Joint	Lateral
	Murder	Rape	Assault	Robbery	Burglary	Larceny	Auto_Theft						
1*	4	4	4	4	4	4	4	28	1	100.00	100.00	200.00	100.00
2	2	4	4	3	4	4	4	25	1	93.75	50.00	143.75	143.75
3	2	2	3	4	4	4	4	23	1	87.50	43.75	131.25	143.75
4	4	3	4	3	4	2	2	22	1	31.25	93.75	125.00	37.50
5	3	3	3	4	3	3	3	22	1	75.00	68.75	143.75	106.25
6	3	2	3	4	3	3	3	21	1	56.25	56.25	112.50	100.00
7	3	4	3	2	3	2	3	20	1	37.50	75.00	112.50	62.50
8	2	4	2	2	3	4	3	20	1	81.25	37.50	118.75	143.75
9	2	3	3	2	2	3	3	18	1	68.75	31.25	100.00	137.50
10	4	2	2	3	3	1	2	17	1	6.25	87.50	93.75	18.75
11	3	2	3	3	1	1	2	15	1	18.75	62.50	81.25	56.25
12	4	2	2	1	2	2	1	14	1	25.00	81.25	106.25	43.75
13	1	2	2	1	3	3	1	13	1	50.00	12.50	62.50	137.50
14	1	1	1	1	3	3	2	12	1	62.50	6.25	68.75	156.25
15	1	2	2	1	2	1	1	10	1	12.50	25.00	37.50	87.50
16	1	1	1	1	1	1	4	10	1	43.75	18.75	62.50	125.00
17	1	1	1	1	1	1	1	7	1	.00	.00	.00	100.00

Figure 5

X and **Y** in Figure 5 are the main results of **POSAC** section. They are called base coordinates. From these coordinates are derived the joint (**J**) and the lateral (**L**) scores (**J=X+Y** and **L=100+X-Y**).

X and **Y** are used to locate the position of the profiles in the Space Diagram (see later, Figure 7)

Monotonicity coefficients between the items and J, L, X, Y, P and Q

```
Coefficient of weak monotonicity between each observed item and the
J (i.e. X+Y) , L (i.e. X-Y) , X , Y , P (i.e. Min(X,Y)) , Q (i.e. Ma

Item name         J     L     X     Y     P     Q
---------         -     -     -     -     -     -
Murder    1    .79  -.91  -.22  1.00   .60   .86
Rape      2    .93   .00   .66   .66   .87   .85
Assault   3    .94  -.27   .53   .80   .92   .82
Robbery   4    .94  -.18   .58   .76   .91   .77
Burglary  5    .90   .23   .76   .51   .79   .88
Larceny   6    .86   .85   .99  -.10   .86   .68
Auto_The  7    .80   .61   .89   .13   .87   .55
```

Figure 6

Like **J** and **L**, **P** and **Q** are factors which are functions of the base coordinates **X** and **Y**. Precisely, **P=min(X,Y)** and **Q=max(X,Y)**.

A table of weak monotonicity coefficients between each scalogram item and each of the factors **J**, **L**, **X**, **Y**, **P** and **Q** is given by **POSAC** (see Figure 6).

Space Diagram

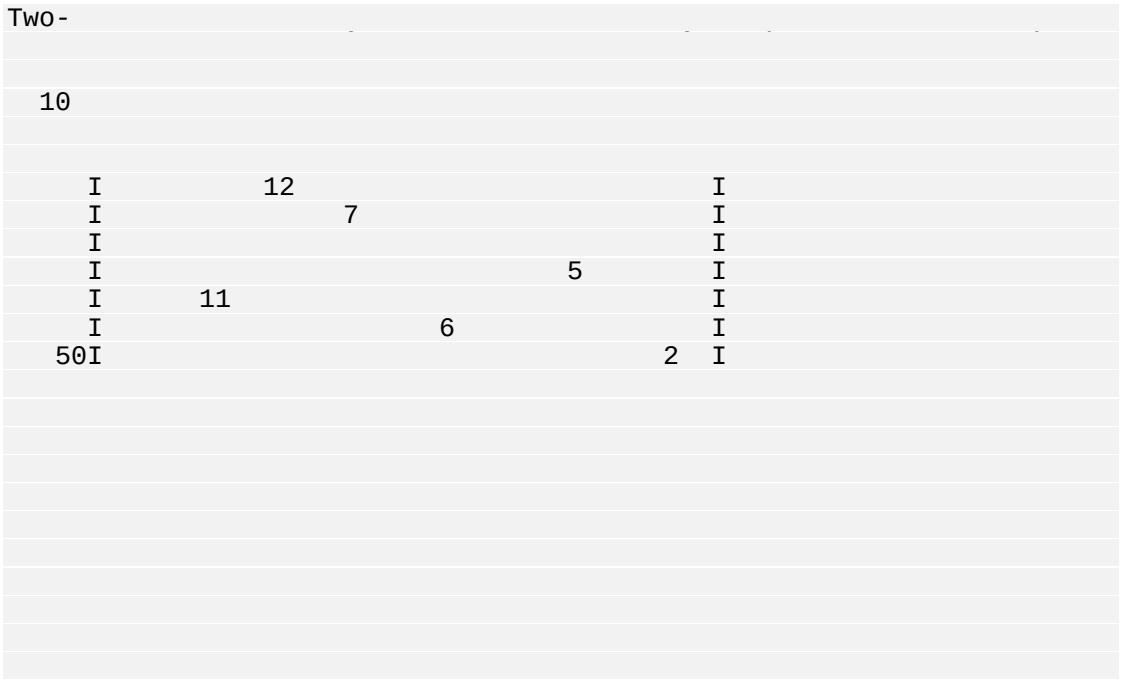


Figure 7

In Figure 7 each profile is represented by a point whose coordinates are (*x*,*y*) from table in Figure 5. The point is labelled by the Id number of the profile.

1 is the Id number of the maximal (numerically) profile, and **17** is that of the minimal profile. In every item, category **1** denotes the lowest crime rate and **4** the highest crime rate. Hence, profile **17** here represents the lowest level of crime phenomenon and profile **1** its highest level that can be created from the data. And, in general (for such interpretation of categories), the Northeast direction (i.e., the joint direction) in the **POSAC** diagram represents an increase in the studied phenomenon. Thus the level of profile **5** is higher than that of profile **6**; but the levels of profiles **7** and **9** are incomparable. The spread of profiles along the perpendicular direction (Northwest to Southeast) attests to the fact that the scalogram is one of partial order rather than complete order.

Item Diagram

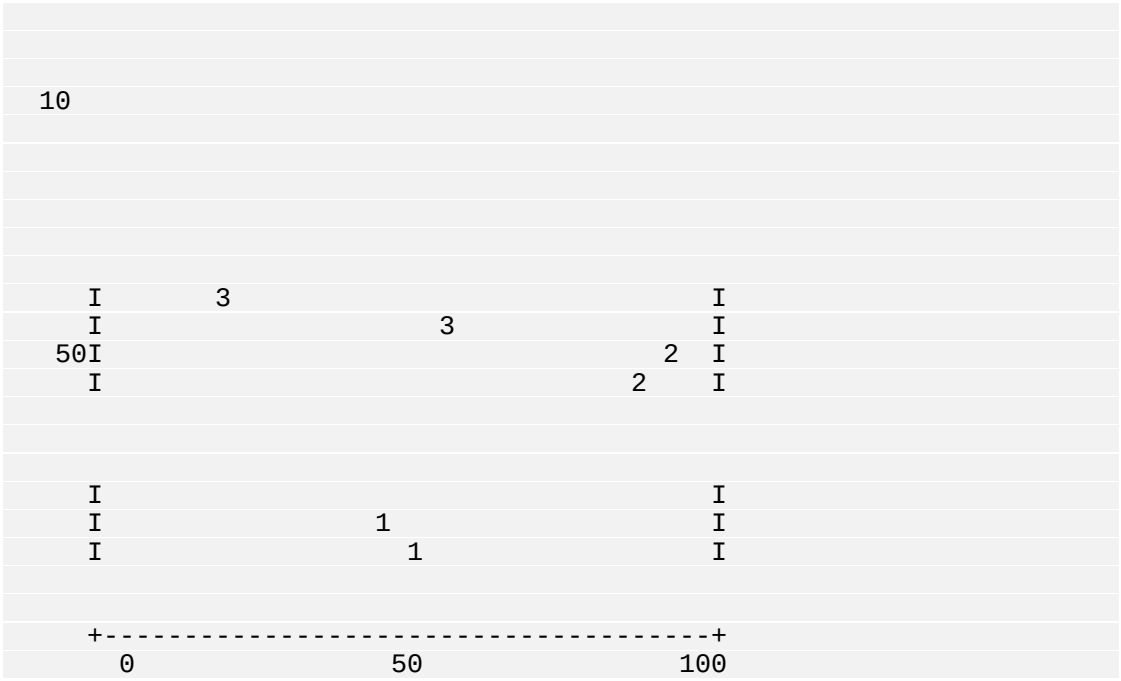


Figure 8
Horizontal partition

For each item, **POSAC** outputs an Item Diagram. The locations of the points in this diagram are as in the Space Diagram. But, instead of the profile Id number (in Space Diagram), here, is printed the category that the profile has in the specific item. For example, consider profile number **4**, "**4343422**", note its location in Figure 7. In Figure 8, at the same location, appears the number **4**, which is the value of the first item in profile **4**. In the second Item Diagram, at the same location, will be printed, the number **3**, etc...

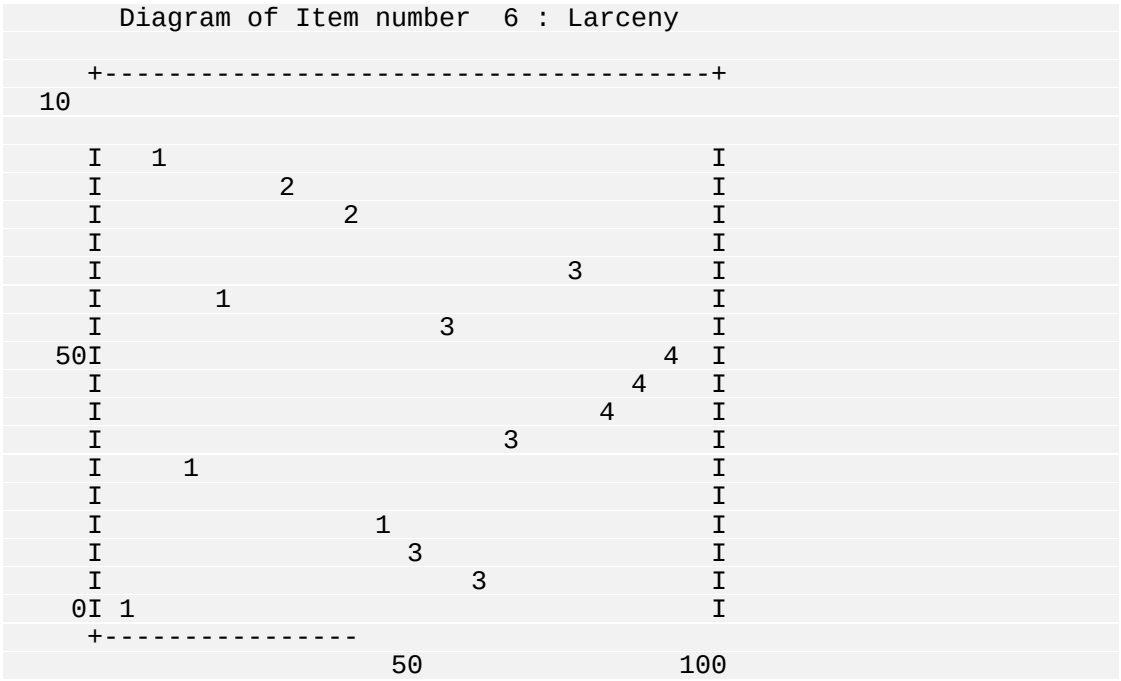


Figure 9
Vertical partition

Item Diagram helps the user to detect partitioning of the space into regions according to categories of the specific item. The partitioning may have various directions: with **X**, **Y**, **J**, **L**, **P** or **Q**. In Figure 8, **Murder** "goes" with the coordinate **Y**, while in Figure 9, **Larceny** partitions the space in the **X** direction. In fact, before viewing Item Diagrams, one can detect these directions (if any) in the table of Figure 6: a high correlation indicates a common direction. So, item 1 (**Murder**) goes with **Y** (**1.00**), while item 6 (**Larceny**) goes with **X** (**.99**).

Analysis of the results

The partial order of the sixteen American cities turns out to be essentially two-dimensional. The space diagram of the output is portrayed in Figure 7. In this figure each profile (namely city) appears as a point, the serial numbers being those in Figure 2. Thus, in the Southwest corner of the space is located the profile representing the lowest crime rate "**1111111**" which belongs to the city of Hartford. At the Northeast corner of Figure 7 is the highest crime profile "**4444444**" which belongs to "**NOBODY**". The **POSAC** procedure automatically adds an extreme profile if it does not exist in the data. All other cities are located on the space diagram according to their profiles. The diagonal direction going from the Southwest corner of the space to the Northeast corner defines the intermediate levels of crime rates and is called the *joint* direction of the partial order. In terms of base coordinates, it is **X+Y** (see also Paragraph "Base Coordinates"). *Comparable* profiles (cities) will have their points aligned with positive slope along the *joint* direction. *Incomparable* profiles have their points aligned with the negative slope, namely along the *lateral* direction (**X-Y** in terms

of the base coordinates, see also paragraph "Base Coordinates".) Detailed analysis of the systematic differences among the crime rates is made in term of seven item diagrams, one for each item. To save space only the items who partition the space in the base direction are presented. Namely, **Murder** whose categories partitions the space in the **Y** direction and **Larceny** that partitions the space in the **X** direction. These partitions are indicated by the horizontal lines in Figure 8 and vertical lines in Figure 9. This means that the partly ordered space is essentially spanned by these two crimes, one against *persons* and the other against *property*. Each of the other crimes behaves like "a partial order combination" of these two, essentially partitioning their categorical regions into finer regions. Note that there need not always be items that correspond to the directions of these basic coordinates.

The difference between levels on the base items determines the differentiation among the profiles in the lateral direction of the partial order. In the Northwest region are located cities who have lower crime rates on larceny and higher on murder (Houston, Atlanta and Dallas). In contrast, cities located towards Southeast region are higher on crimes against *property* and lower on crime against *persons* (New-York and Denver). However note that the extreme Southeast region is empty, namely in this sample of cities there are no cities that have lowest ranks on murder and highest on larceny, that means that there are cities characterised by a high rate on murder and a low rate on larceny. But, there are no cities (in this sample and may be at all) whose crime rate on larceny is very high together with a very low crime rate on murder. In other words, a high degree of larceny causes murder.

The above example shows the power of **POSAC** to stratify objects simultaneously according to *level* of the stratification criteria, *joint*, and the *kind* of the stratification criteria, *lateral*, as functions of the content meaning of the base coordinates **X** and **Y**.

It will be of interest to study the relationships of this partial order to other objective criteria of the U.S. cities, like certain demographic characteristics etc, called external variables.

Output of external maps

We already saw the definition of an external trait. Let us suppose that in the above example on crimes in American cities, we have an hypothetical variable named **Dummy**, and suppose that this external variable is dichotomous (each city get value 1 or 2 in this variable). Once, the two-dimensional partial order is processed as described above, we want to study the relationship of variable **Dummy** with this partial order. Namely, how cities having category 1 in variable **Dummy** are located in the two-dimensional space. Follow, the HUDAP job requesting this relationship:

```
$SET LINESIZE = 80 ;
$DATA
      NAMES = Id1 Id2 1-8 (A) Murder Rape Assault
              Robbery Burglary Larceny Auto_Theft 10-16
              Dummy 18 ;
      FILE = 'CRIME.DAT' ;
$POSAC
       NAMES = Murder TO Auto_Theft ;
       LABEL = Id1 Id2  ;
       EXTMAPS = Dummy 1 ;
```

With reference to this external trait (**Dummy=1**), the **POSAC** procedure provides the following information:

```
                    LEGEND ON CODES USED IN TRAIT DIAGRAMS

In the absolute trait diagrams the following codes are used for the
 percentages of subjects with the specified trait (among those with
 identical profile) .

                   O :       0%
                   A :  ] 0% TO  20%]
                   B :  ]20% TO  40%]
                   C :  ]40% TO  60%[
                        -           -
                        -           -



In the relative trait diagrams another codes are used with the
 following meaning :

        H   Proportion of trait in profile higher than
            Proportion of trait in population by more than 5 percent

        A   Proportion of trait in profile equals
            Proportion of trait in population + or - 5 percent

        L   Proportion of trait in profile lower than
            Proportion of trait in population by more than 5 percent
```

Figure 10

In Figure 10 are given the meaning of letters used in absolute and relative trait diagrams (see below)

```
                    For external trait number  1


 Id
 --


      d e a b g c o
      e   u e l e _
      r   l r a n T


  1*
  2
  3
  4   4 3 4 3 4 2 2   22     1          0          1          0
  5   3 3 3 4 3 3 3   22     1          0          1          0
  6   3 2 3 4 3 3 3   21     1          0          1          0
  7   3 4 3 2 3 2 3   20     1          0          1          0
  8   2 4 2 2 3 4 3   20     1          0          1          0
  9   2 3 3 2 2 3 3   18     1          0          1          0
 10   4 2 2 3 3 1 2   17     1          1          0        100
 11   3 2 3 3 1 1 2   15     1          1          0        100
 12   4 2 2 1 2 2 1   14     1          1          0        100
 13   1 2 2 1 3 3 1   13     1          1          0        100
 14   1 1 1 1 3 3 2   12     1          0          1          0
 15   1 2 2 1 2 1 1   10     1          1          0        100
 16   1 1 1 1 1 1 4   10     1          1          0        100
 17   1 1 1 1 1 1 1    7     1          1          0        100

Total                       16          7          9         44


MU(TRAIT,X)= -.965                  MU(TRAIT,J)= -.954
MU(TRAIT,Y)= -.289                  MU(TRAIT,L)= -.622
```

Figure 11

In Figure 11, for each profile, the frequency is divided into, **`Freq in`** (number of subjects having the trait) and **`Freq out`** (number of subjects who have not the trait). **`Percent in`** is the proportion of subjects with the specific trait among all subjects having the same profile. In our example, since the original frequency of each profile is 1, the proportion can be only 0 or 100. But in **`Total`**, proportion = 44 = 100*(7/16).

Regarding the external trait as a dichotomous variable with values 0 (indicating absence of trait) and 1 (indicating presence of trait) **`POSAC`** gives the weak monotonicity coefficients of the trait with each of the axes **`X,Y,J`** and **`L`**.

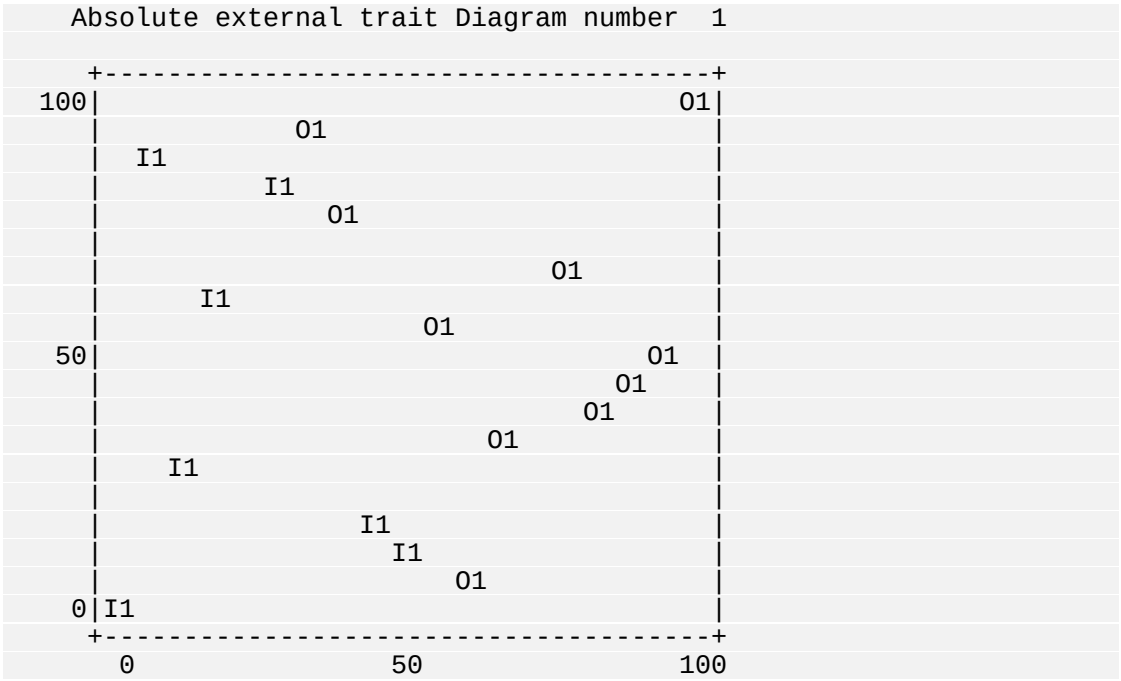


Figure 12

In Figure 12, we have the **`Absolute external trait Diagram`**. This is a reproduction of the space diagram where, in the location - and instead of - each profile Id, appear the "**`Percent in`**"(as one of the letters **`O`**, **`A`**, **`B`**, **`C`**, **`D`**, **`E`** or **`I`**, see Figure 10), followed by the frequency of the profile.

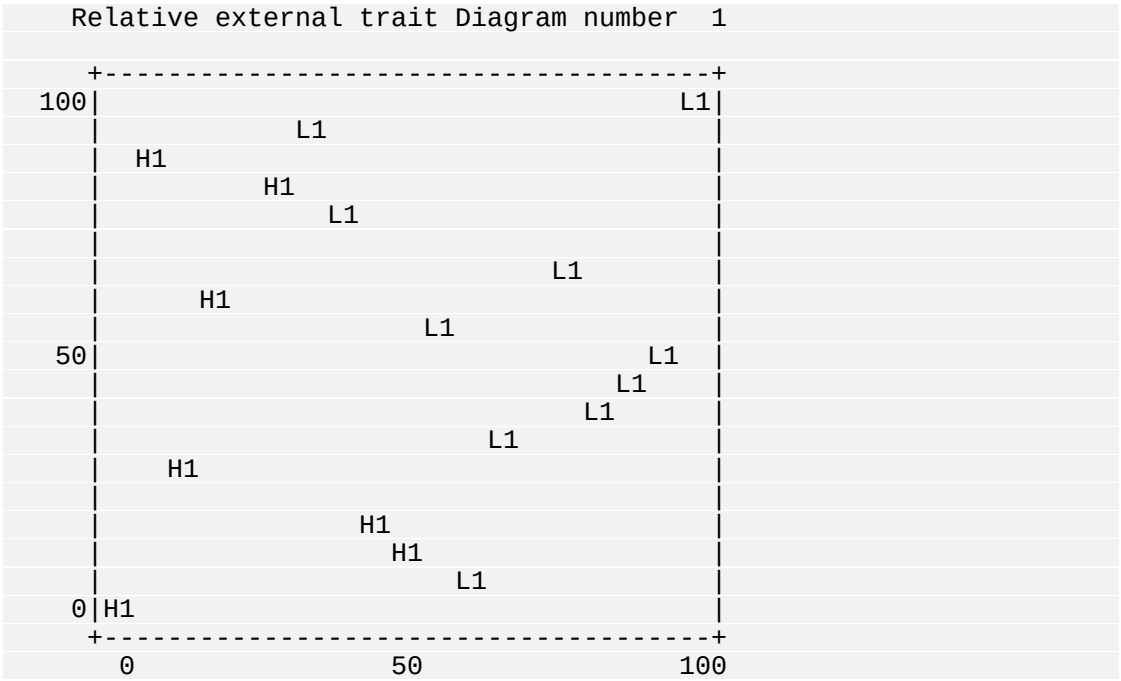


Figure 13

The **Relative external trait Diagram**, which is similar to the above except that the proportion of subjects in each profile who possess the external trait is given relative to their proportion in the entire studied sample. More precisely, let us note the **Percent in** of a specific profile by *Prct*, and the **Percent in** in the Total by *Tot*. If $Prct \le Tot - 0.05$, then the letter **L** appear at the profile location, if $Tot - 0.05 < Prct \le Tot + 0.05$, the letter **A** is printed, and if $Tot + 0.05 < Prct$, the letter **H**.

The external trait diagrams permit the identification of regions in the space diagram where the trait is widespread, and hence the study of the relationships between the structured (and interpreted) scalogram, and the external trait. For example in our hypothetical example, the trait, "**Dummy=1**" discriminates the space in the x and joint directions, as we can see in the coefficients of Figure 11, **MU(TRAIT,X)=-.965** and **MU(TRAIT,J)=-.954**.

POSAC section menu

NAMES = <var list> ;

Names of the variables from which profiles are computed and processed to give a two-dimensional configuration.

LABEL = <1 or 2 variables> ;

The variable(s) must be alphanumeric. These variables are used to identify subjects for each profile. No more than 200 subjects can be labelled.
The default is a serial number of the subjects.

PROCESS = <COMPLETE | PARTIAL> ;

Kind of **POSAC** processing. **COMPLETE** for complete processing, **PARTIAL** for partial processing, that is only computation of profiles with their case labels, scores, frequencies....
The default is **COMPLETE**.

FREQ = <variable> ;

The variable must be numerical. The value of this variable for each case will be considered by **POSAC** as the frequency of the case

LOWFREQ = <value> ;

The lower bound of frequency profile. **POSAC** will retain only profiles whose frequency is greater than this bound. This value must be a positive integer number.
The default is 0

```
EXTMAPS = <variable val list
            ...      ...
          variable val list  &
          variable val list
            ...      ...
          variable val list  &
            ...      ...
            ...      ...
          variable val list
            ...      ...
          variable val list> ;
```

To define external maps. The delimiter '**&**' separates the definition of two external maps.

NVALID = <value> ;

The number of non-missing items for which a case is not rejected.
The default is 2.

INSERT paragraph

Paragraph defining the (new) variables which will contain the base coordinates
X and **Y**, and the transformed coordinates **J** and **L** (**J** for joint, **L** for lateral).

target variable = <X | Y | J | L> /

This sentence can be repeated for each of the 4 coordinates. "**target variable**" can be an old (existing) name or a new one.

References

Brown, J.M. and Sime, J.D. Multidimensional scaling analysis of qualitative data. In E. Shepard and J.P. Watson (Eds.) Personal Meanings. John Wiley and Sons, 1982, pp. 71-90.

Guttman, L. A partial-order scalogram classification of projective techniques. In M. Hammer, K. Sazinger and S. Sutton (Eds.) Psychopathology. New York: John Wiley, 1972, pp. 481-490* .

Guttman, L. The Cornell technique for scale and intensity analysis. In Chuchman, Ackoff, and Wax (Eds.) Measurement of Consumer Interest. U. of Penna. Press, 1947, pp. 60-84.

Levy, S. Partial orders of Israeli settlements by adjustive behaviours. Israel Social Science Research, 1984, 2, 44-65.

Levy, S. Partial order analysis of crime indicators. Social Indicators Research, 1985, 16, 195-199.

Levy, S. (Ed.) Louis Guttman on Theory and Methodology: Selected Writings, Aldershot: Dartmouth, 1994.

Levy, S. and L. Guttman. The partial order of severity of thyroid cancer with the prognosis of survival. In J.F. Marcotorchino, J.M. Proth and J. Janssen (Eds.) Data Analysis in Real Life Environment: Ins and Outs of Solving Problems. Amsterdam: Elsevier Science Publisher B.V. (North-Holland), 1985, pp. 111-119*.

Levy, S. and L. Guttman. Partial order classification of microcomputers. In J. Janssen, f. Marcotorchino and J.M. Proth (Eds.) Data Analysis: The Ins and Outs of Solving Real Problems. New York: Plenum Press, 1987, pp. 255-268.

Lipshitz, G. A spatial typology of Israel on basis of partial-order analysis. Geographical Analysis, 1986, 18, 324-342.

Shye, S. and Amar R. Partial order scalogram analysis by base coordinates and lattice mapping of the items by their scalogram roles. In D. Canter (Ed.), Facet Theory: Approaches to Social Research. New York: Springer Verlag, 1985, pp. 277-298.

Shye, S. Partial order scalogram analysis In S. Shye (Ed.) Theory Construction and Data Analysis in the Behavioural Sciences. San Francisco: Jossey-Bass, 1978, pp. 265-279.

* To be found also in Levy, S. (Ed.) Louis Guttman on Theory and Methodology: Selected Writings, Aldershot: Dartmouth, 1994.

*

Shye, S. Multiple Scaling. Amsterdam: North-Holland, 1985.

Yalan E., Finkel, C., Guttman, L. and Jacobson, C. The Modernization of Traditional Agricultural Villages. Rehovot, Israel: Settlement Study Center, 1972.

The *MPOSAC* Section

Introduction to MPOSAC

The name **MPOSAC** comes from: **M**ultidimensional **P**artial **O**rder **S**calogram **A**nalysis with base **C**oordinates

The **MPOSAC** procedure provides a multidimensional representation of a partial-order scalogram for a set of *N* profiles, based on *n* variables (items). The categories of these variables can be any subset of the integers from 0 to 99, and the category range may vary for different variables. However, the ranges of the variables must have the same meaning (not necessarily the phrasing) and be uniformly ordered with respect to a common content criterion (for example, category "1" may represent "low" with respect to that criterion, in all variables). If, in order to meet this requirement original categories need to be reordered or collapsed, these changes must be done by **CODING** section before running **MPOSAC** procedure.

A partial order on a set of profiles is defined by two relations: comparability (>, < or =) and incomparability (symbolised by "#", to be not confused with inequality symbol "≠"). Let $p = (p_1, p_2, ..., p_n)$ and $q = (q_1, q_2, ..., q_n)$ be two profiles. p is said to be greater than q ($p > q$), if $p_i \geq q_i$ for all i's and there is at least one i for which $p_i > q_i$. p is said to be incomparable to q ($p \# q$), if for some i's $p_i \geq q_i$ while for other i's (at least one) $p_i < q_i$. Example (for profiles based on seven variables, n=7):

$$\left.\begin{array}{l} p = 4343422 \\ q = 4223312 \end{array}\right\} \longrightarrow p > q$$

$$\left.\begin{array}{l} p = 2422343 \\ q = 3432323 \end{array}\right\} \longrightarrow p \# q$$

Let **P** be the partially ordered set of profiles on *n* variables, **MPOSAC** algorithm tries to reduce the number of variables from *n* to *m*.

More precisely, to each profile $p = (p_1, p_2, \ldots, p_n)$ in **P** will correspond a profile $(x_1, x_2, ..., x_m)$ in $\mathbf{R}^m$ (the Cartesian hyperplane) such that, through this reduction

the partial order in **P** is preserved as well as possible. $x_1, x_2, \ldots$ and x_m are called base coordinates.

Furthermore, for specified categories of a given external variable (a variable not in the original set of the *n* variables) representing an external criterion or trait, trait-diagrams are presented depicting the proportion of subjects possessing that trait among all those sharing the same profile in *n* variables (see **EXTMAPS** sentence). The trait may be specified also as a combination (intersection) of response categories from different external variables

MPOSAC section

The **MPOSAC** procedure is called and activated by the control word **MPOSAC** followed by the required sentences. The variables are entered in the **NAMES** sentence in the usual manner. A simple example of a HUDAP **MOSAC** job:

```
$DATA
      NAMES = Id1 Id2 1-8 (A) Murder Rape Assault
              Robbery Burglary Larceny Auto_Theft 10-16 ;
      FILE = 'CRIME.DAT' ;
$MPOSAC
      NAMES = Murder TO Auto_Theft ;
      MINDIM = 2 ;
      MAXDIM = 3 ;
```

But many other options are available in **MPOSAC** section. Please refer to **POSAC** section.

Follow the new sentences specific to **MPOSAC:**

Dimensionality

Contrary to **POSAC** that is technically limited to two dimensions, **MPOSAC** can process the data in any dimensionality. A space may be two-dimensional (a surface represented by a length and a width), three-dimensional (a body represented by a length, width and height), or four-dimensional or more (in which case a direct physical presentation is impossible).

Two sentences allow the user to choose the dimensionalities of the analysis:

MINDIM = <value> ;
where "**value**" is the minimal dimensionality from which **MPOSAC** is processed. It must be a positive integer number.The default of **MINDIM** is 2.

MAXDIM = <value> ;

where "**value**" is the maximal dimensionality at which **MPOSAC** is processed. It must be a positive integer number. The default of **MAXDIM** is 2.

Item Diagrams

In **POSAC** section, the program gives automatically all item diagrams, namely, a diagram for each item, since the analysis is two-dimensional. When the dimensionality is greater than two, the number of item diagrams becomes very large, exactly: *nm(m-1)/2,* where *n* is the number of items and *m* the dimensionality. For this reason the sentence **DIAGOUT** was added to **MPOSAC** section in order to control the number of item diagrams outputted.

A coefficient of weak monotonicity between each item and each axis of the space is computed by **MPOSAC** as for example in the following table:

Item name		J*	X1	X2	X3
		-	--	--	--
Overcrowding	1	.78	-.06	1.00	.46
No inside Toilet	2	.76	1.00	-.07	.42
No Car	3	.95	.70	.88	.71
Unskilled	4	.89	.43	.92	.55
Jaundice	5	.96	.79	.63	.78
Measles	6	.89	.78	.12	.96
Scabies	7	.87	.42	.32	1.00

The syntax of **DIAGOUT** sentence is:

DIAGOUT = d ;

Where **d** is a real number between **-1.00** and **1.00**

The effect of this sentence is that if the monotonicity coefficient of an item with some axis (say Xk) is greater than **d**, then all two-dimensional item diagram projections of Xk with all other axes for this item are outputted.

The default of **d** is 0.98.

For example in the above table, item 'Overcrowding' has a coefficient of **1.00** with X2 axis. Assuming that **d=0.98**, the diagrams of this item for projections (X1,X2) and (X2,X3) will be plotted. On the other hand, the diagrams of item 'No inside Toilet' will be plotted for projections (X1,X2) and (X1,X3). No diagrams will be plotted for item 'No Car', and so on.

Note that if a user wants to output item diagrams for all items and all projections, he has just to set:

DIAGOUT = -1.00 ;

If he wants to suppress all item diagrams, he has to set:

```
DIAGOUT = 1.00 ;
```

Example of MPOSAC application

Refer to an article which appeared in Bulletin de Methodologie Sociologique (BMS) N.65 January 2000, pages 19-32.

Follow the related HUDAP job requesting **MPOSAC** on datafile named **'wards.dat'**:

```
$
      names = v1 to v7 1-7;
      file = 'wards.dat' ;
$mposac
      names = v1 to v7 ;
      maxdim = 3;
```

```
3544444
4443444
3433444
3444433
3324333
2432343
3534311
3222234
4123412
2312432
4222412
4134112
3123223
2212323
2312322
1211234
4123111
1311222
1311122
1321111
1211111
```

wards.dat: Input datafile

MPOSAC section menu

NAMES = <var list> ;

Names of the variables from which profiles are computed and processed to give a two-dimensional configuration.

LABEL = <1 or 2 variables> ;

The variable(s) must be alphanumeric. These variables are used to identify subjects for each profile. No more than 200 subjects can be labelled.
The default is a serial number of the subjects.

PROCESS = <COMPLETE | PARTIAL> ;

Kind of **MPOSAC** processing. **COMPLETE** for complete processing, **PARTIAL** for partial processing, that is only computation of profiles with their case labels, scores, frequencies....
The default is **COMPLETE**.

FREQ = <variable> ;

The variable must be numerical. The value of this variable for each case will be considered by **POSAC** as the frequency of the case

LOWFREQ = <value> ;

The lower bound of frequency profile. **MPOSAC** will retain only profiles whose frequency is greater than this bound. This value must be a positive integer number.
The default is 0

```
EXTMAPS = <variable val list
            ...      ...
          variable val list  &
          variable val list
            ...      ...
          variable val list  &
            ...      ...
            ...      ...
          variable val list
            ...      ...
          variable val list> ;
```

To define external maps. The delimiter '**&**' separates the definition of two external maps.

NVALID = <value> ;

The number of non-missing items for which a case is not rejected.
The default is 2.

MINDIM = <value> ;

The minimal dimensionality from which **MPOSAC** is processed. "**value**" must be a positive integer number.
The default of **MINDIM** is 2.

MAXDIM = <value> ;

The maximal dimensionality at which **MPOSAC** is processed. "**value**" must be a positive integer number.
The default of **MAXDIM** is 2.

DIAGOUT = <value> ;

The minimal monotonicity coefficient between any item and any axis for which a two-dimensional item diagram projection containing the specific item is printed.
The default of **DIAGOUT** is .98.

The *PEARSON* Section

Introduction

The **PEARSON** Section computes Pearson correlation coefficients for pairs of interval variables. This section was introduced into HUDAP package in order to allow comparison between different researches, since Pearson coefficients are more utilized than any other coefficient of correlation. However, the researcher has to be aware that Pearson coefficient is a coefficient of **linear** correlation based on regression lines, while the concept of correlation does not necessarily depend on the concept of regression* .

Definition

Let x and y be two numerical variables represented by N pairs (x_i, y_i). By a least square method, one can draw a regression line of variable y in regard to variable x. The same method can be used to find the regression line of variable x in regard to variable y. By definition, the coefficient of linear correlation is the square root of the two regression line slopes product. See Appendix A for a mathematical description of the coefficient.

Operation

The procedure **PEARSON** is invoked by the section name **PEARSON**. 4 sentences (**NAMES**, **COLNAMES**, **ROWNAMES**, **MATRIX**) and one paragraph (**OUTPUT**) are available in this section.

These above sentences and the paragraph **OUTPUT** can be named in any order and are separated from each other by a semicolon. Each one can be used only

* Guttman, L. What is not what in statistics. The Statistician, 1977, 26, 81-107.

once for **PEARSON** section. The only required sentences are either **NAMES** or **COLNAMES** and **ROWNAMES** which specifies the variables being analysed.

The syntax of these sentences are identical to those of **MONCO** section. Please, refer to **MONCO** section for a detailed explanation of these sentences.

PEARSON section menu

NAMES = <var list> ;

Names of the variables for which matrix of Pearson coefficients is computed.

COLNAMES = <var list> ;

This sentence is used together with **ROWNAMES** sentence to get a rectangular table of coefficients rather than a squared matrix (computed by **NAMES** sentence). "**var list**" appear as column variables.

ROWNAMES = <var list> ;

"**var list**" appear as row variables. Refer to **COLNAMES** sentence.

MATRIX = <UNIFIED | SEPARATE> ;

The way in which results are obtained. **UNIFIED** gives one matrix of coefficients together with number of retained cases. If **SEPARATE** is chosen, two matrices are output, one for coefficients, the other for number of cases. This last option is useful when the matrix of coefficients is published.
The default is **UNIFIED**.

OUTPUT paragraph

Paragraph defining the output of the **PEARSON** Matrix.

FORMAT = '<char>' /

The format "**char**" specifies the layout (in fixed fields) of the matrix in a single row . The usual FORTRAN rules for the format are applicable, but only the F specification is allowed. "**char**" cannot exceed 80 characters.
The default is **(13F6.2)**.

FILE = '<char>' /

"**char**" is the matrix file pathname not exceeding 40 characters in length. It must be enclosed in apostrophes.

MEMORY /

The resulting matrix is written into the memory, in order to use it as input of another section (as **WSSA1** for example) in the same job.

The *INTRACLASS* Section

Introduction

In some cases involving pairs of measurements there is no way to distinguish between the pairs. For example, given IQ measurements of a pair of identical twins, how can it be decided which measurement is x and which is y? In such cases a different type of correlation coefficient, called the intraclass correlation coefficient which treats the pairs of measurements symmetrically, is needed to asses the relation between them.

Moreover, this coefficient is not limited to pairs of variables. It can be defined on more than two variables. In this context, a case or observation is called a **class**. Each class, has a number of members represented by **scores**. The classes can even have unequal number of members.

Intuitive formulation

An exact mathematical formulation of the coefficient is given in Appendix A. Here we just give an intuitive definition as formulated by Haggard*

The coefficient of intraclass correlation (R) is the measure of the relative homogeneity of the scores within the classes in relation to the total variation among all the scores in the table. Thus maximal positive correlation exists when all the intraclass scores are identical and the scores differ only from class to class. As the relative heterogeneity of the intraclass scores increases, the computed value of R will decrease; maximal negative correlation exists when the heterogeneity of the intraclass scores is maximal and all the class means are the same.

The coefficient of intraclass correlation is a **similarity** coefficient which lies between **`-1`** and **`+1`**.

Follow two hypothetical examples* illustrating extreme cases:

* Haggard, E.A. Intraclass Correlation and the Analysis of Variance. New York: The Dryden Press, Inc.

	Example A			Example B		
Classes	x	y	Sum	x	y	Sum
a	1	1	2	1	5	6
b	2	2	4	2	4	6
c	3	3	6	3	3	6
d	4	4	8	4	2	6
e	5	5	10	5	1	6
Total	15	15	30	15	15	30
	R = +1			R = -1		

Operation

The Intraclass correlation procedure is invoked by the section name **INTRACLASS**. This section contains only one sentence whose syntax is:

```
NAMES = <var list_1  &
         var list_2  &
          ........
         var list_n> ;
```

where "**var list_i**" is a list of variables for which the coefficient is requested. The symbol "**&**" is a separator between 2 variable lists.

Example of input data and HUDAP job

The example is taken from the same reference* In a study of 61 high school seniors who were members of six "nonofficial" social clubs, marked consistencies were observed in the social behaviour patterns of the members within each club and marked differences between clubs. In addition, standardized measures of the intellectual ability, performance in academic school subjects, and indices of the discrepancy between these two measures were collected for each of the club members. Three boys' and three girls' clubs were studied; membership varied from 3 to 13 seniors in each club. Follow the discrepancy scores between ability and achievement for members of the six high school social clubs.

```
 28 32 23 34 28 30 28 30 31 30 30 29 40
  7 24 17 16 28 29 33 21 16 20 15 25
 34 37 37 25 30 23 29 35 38 33
 25 23 33 38 18 21 16 29 23 26 22 16 22
 27 26 15 18  7 31 26 33 15 25
  1 10 19
```

CLUBS.DAT file

Output listing containing the job and the results:

```
-------------------------------------------------------------------------
_  1_ $DATA NAMES =  V1  TO V13 1-39 ;
_  2_       FILE = 'CLUBS.DAT' ;
_  3_ $INTRACLASS  NAMES = V1 TO V13 ;
-------------------------------------------------------------------------



     *********************************
          I N T R A C
     *********************************


Number of cases .............    6

-------------------------------------------------------------------------
Variable List : V1       V2       V3       V4       V5       V6
                V7       V8       V9       V10      V11      V12
                V13
R Coefficient :   .44    Valid cases :     6
-------------------------------------------------------------------------
```

We may say that a relationship between club membership and the extent to which these children utilize their intellectual potential in terms of academic achievement is characterized by an intraclass correlation coefficient of **.44**.

Reference

Haggard, E.A. Intraclass Correlation and the Analysis of Variance. New York: The Dryden Press, Inc.

INTRACLASS section menu

```
NAMES = <var list_1  &
         var list_2  &
          ........
         var list_n> ;
```

For each "**var list_i**" a coefficient of **INTRACLASS** correlation is computed. The symbol "**&**" is a separator between 2 variable lists.

Appendix A

HUDAP Mathematics

MONCO

Definition of weak coefficient of monotonicity

Given N pairs of observations $\{(x_i, y_i); i = 1,2,..., N\}$ on two numerical variables x and y, the weak coefficient of monotonicity μ_2 between x and y is defined as follows:

$$\mu_2 = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}(x_i - x_j)(y_i - y_j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}|x_i - x_j||y_i - y_j|}$$

Intuitively, this coefficient tells us how much the two variables vary in the same sense, since the quantities $(x_i - x_j)$ and $(y_i - y_j)$ express the respective progressions of the variables x and y.

The monotonicity coefficient can be computed directly from a crossfrequency table of x and y. When there are a big number of cases with a small number of categories in x and y, the computation of μ_2 from the table of frequencies is faster.

Let $\{\xi_i; i = 1,2,..., n\}$ be the categories of variable x and let $\{\psi_k; k = 1,2,..., m\}$ be the categories of variable y. We note f_{ki} the cross-frequency of category ψ_k with category ξ_i. The table may look like:

x

y	ξ_1	ξ_2		ξ_n
ψ_1	f_{11}	f_{12}		f_{1n}
ψ_2	f_{21}	f_{22}		f_{2n}
ψ_m	f_{m1}	f_{m2}		f_{mn}

The above formula of monotonicity coefficient becomes:

$$\mu_2 = \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\sum_{k=1}^{m-1}\sum_{l=k+1}^{m}(f_{ki}f_{lj} - f_{li}f_{kj})(\xi_i - \xi_j)(\psi_k - \psi_l)}{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\sum_{k=1}^{m-1}\sum_{l=k+1}^{m}(f_{ki}f_{lj} + f_{li}f_{kj})|\xi_i - \xi_j||\psi_k - \psi_l|}$$

Definition of distribution uniformity coefficient

When a variable is almost constant over the observations, the coefficient of monotonicity between this variable and any other variable becomes unsteady and therefore cannot be used. To detect such variables, a coefficient was found. This coefficient tells us to what extent the variable is uniform. Its definition is as follows:

Let $(c_i, i = 1,2,\ldots,m)$ be the categories of a variable x, and let $(f_i, i = 1,2,\ldots,m)$ be the corresponding frequencies, the coefficient of distribution uniformity is defined by:

if $m = 1$

$$unifor = 0$$

if $m > 1$

$$unifor = 1 - \frac{\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}|f_i - f_j|}{\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}(f_i + f_j)}$$

Reference

Guttman, L. Coefficients of polytonicity and monotonicity. Encyclopedia of Statistical Sciences, N.Y.: John Wiley & Sons, Inc., 1986, 7, 80-87.

Disco and Odisco

General case of *m* populations

Notations:
P_a : a population of n_a individuals $(a = 1, 2, \ldots, m)$
x_p : a value of a numerical variable x for individual $p \in P_a$
$\bar{x}_a$: the arithmetic mean of x for P_a

$$\bar{x}_a = \frac{1}{n_a} \sum_{p \in P_a} x_p$$

disco and *odisco* are defined by the following formula :

$$disco = \frac{\sum_{a=1}^{m} \sum_{b=1}^{m} (\bar{x}_a - \bar{x}_b) \sum_{p \in P_a} \sum_{q \in P_b} (x_p - x_q)}{\sum_{a=1}^{m} \sum_{b=1}^{m} |\bar{x}_a - \bar{x}_b| \sum_{p \in P_a} \sum_{q \in P_b} |x_p - x_q|}$$

$$odisco = \frac{\sum_{a=1}^{m} \sum_{b=1}^{m} n_b (\bar{x}_a - \bar{x}_b) \sum_{p \in P_a} (x_p - \bar{x}_b)}{\sum_{a=1}^{m} \sum_{b=1}^{m} n_b |\bar{x}_a - \bar{x}_b| \sum_{p \in P_a} |x_p - \bar{x}_b|}$$

Note that even if they seem different the numerators of *disco* and *odisco* are the same. It is easy to show that they are equal to :

$$\sum_{a=1}^{m} \sum_{b=1}^{m} n_a n_b (\bar{x}_a - \bar{x}_b)^2$$

General properties :

$$0 \le disco \le odisco \le 1$$

Other properties will be given for the case $m = 2$, since this is the more practical case

The case $m = 2$

For 2 populations (P_a, P_b) the above expressions become :

$$disco(P_a, P_b) = \frac{n_a n_b |\bar{x}_a - \bar{x}_b|}{\sum_{p \in P_a} \sum_{q \in P_b} |x_p - x_q|}$$

$$odisco(P_a, P_b) = \frac{2 n_a n_b |\bar{x}_a - \bar{x}_b|}{n_b \sum_{p \in P_a} |x_p - \bar{x}_b| + n_a \sum_{q \in P_b} |x_q - \bar{x}_a|}$$

Both *disco* and *odisco* equal 0 if there is no difference among the means.

disco = 1 if there is no overlap between the two population distributions.

odisco = 1 if no member of population P_b has an *x*-value above $\bar{x}_a$, and no member of population P_a has an *x*-value below $\bar{x}_b$. In other words, there may be overlap in the interval between the two means, but no overlap in the two intervals outside the means.

Reference

Guttman, L. Eta, Disco, Odisco and F. Psychometrika, 1988, 53, 393-405.

WSSA1

Purpose of the method

To analyse a symmetric matrix of dissimilarities or similarities between *n* elements and to give a graphical representation of these *n* elements in an Euclidean space called Smallest Space.

Statement of the problem

Given a symmetric matrix of dissimilarity coefficients $\{D_{ij}\}$, D_{ij} being the coefficient between elements V_i and V_j, we want to represent the elements $(V_k; k = 1,\ldots n)$ as points in an *m*-dimensional Euclidean space such that the following monotonicity condition is fulfilled "as well as possible":

$$D_{ij} < D_{kl} \Leftrightarrow d_{ij} < d_{kl} \qquad (1)$$

for each quadruplet *(i,j,k,l),* d_{ij} being the computed Euclidean distance between points representing V_i and V_j, in the *m*-dimensional space:

$$d_{ij} = \sqrt{\sum_{a=1}^{m} (x_{ia} - x_{ja})^2} \qquad (2)$$

The monotonicity condition is fulfilled as well as possible for dimensionality *m* thought to be the smallest.

The solution is noted:

$$x = \{x_{ia}\}_{i=1,2,\ldots,n;\ a=1,2,\ldots,m}$$

Note: If the input matrix is that of similarity coefficients (as correlations for example), $\{R_{ij}\}$, the monotonicity condition becomes: $R_{ij} > R_{kl} \Leftrightarrow d_{ij} < d_{kl}$. But then, we can always find a simple monotone function which transform similarities into dissimilarities: $D_{ij} = \mathrm{M}(R_{ij})$.

The determination of a loss function

A loss function that can solve our problem would be:

$$\varphi = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(D_{ij} - d_{ij})^2}{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} D_{ij}^{\ 2}} \qquad (3)$$

It has to be pointed that the denominator is a constant. It was introduced in the loss function for normalisation considerations. The coefficients w_{ij} are weights affected to the input data D_{ij}. In particular, these weights allow a good treatment of missing information, that is, if D_{ij} is unknown then w_{ij}=0. The diagonal of the input matrix is not considered in the process, thus w_{ii}=0.

As defined in (3), the minimisation of loss function φ is in fact a "least squares" process, and there is no clear link between φ and the monotonicity condition (1) which is the main purpose of the method. However, if we "look at" D_{ij} as being images of a suitable transformation T, we can solve (1) through minimisation of (3). Therefore, it would be more exact to define φ as follows:

$$\varphi = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(\hat{D}_{ij} - d_{ij})^2}{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} \hat{D}_{ij}^{\ 2}} \qquad (4)$$

where $\hat{D}_{ij}$ are obtained by transformation T that will be defined later.

Minimisation of φ

The minimisation of φ in (4) is handled by the following iterative process of two phases:

First phase: steepest descent procedure

The iterative process is as follows:

$$x_{ia}^{t+1} = x_{ia}^{t} - \gamma^{t} \frac{\partial \varphi^{t}}{\partial x_{ia}^{t}} \qquad i = 1,2,\ldots,n;\ a = 1,2,\ldots m \qquad (5)$$

where:

t : iteration index
γ^t : a step to be found (optimised)
$\varphi^t = \varphi(x^t)$: function φ at iteration t

At iteration t we have a solution $x^t = \{x_{ia}^t\}$. On the basis of this solution, the distances and loss function are computed:

$$d_{ij}^{t} = \sqrt{\sum_{a=1}^{m} (x_{ia}^{t} - x_{ja}^{t})^2} \qquad (6)$$

$$\varphi^{t} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} (\hat{D}_{ij} - d_{ij}^{t})^2}{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} \hat{D}_{ij}^{\,2}} \qquad (7)$$

For each i (point index) and each a (axis index), the following partial derivatives are computed:

$$\frac{\partial \varphi^{t}}{\partial x_{ia}^{t}} \qquad (8)$$

From (5) we could compute x_{ia}^{t+1} if γ^t was known. In order to define γ^t, we consider it as an unknown. Consequently, x_{ia}^{t+1}, d_{ij}^{t+1} and φ^{t+1} become functions of the single variable γ^t. Since we look for a minimisation of φ (as a multivariate function of x_{ia}'s), the minimisation of φ^{t+1} (as a univariate function of γ^t) will provide us the optimal step γ^t corresponding to iteration t. For this purpose we set:

$$\frac{d\varphi^{t+1}(\gamma^{t})}{d\gamma^{t}} = 0 \qquad (9)$$

Equation (9) leads to an equation of degree 3 in γ^t:

$$A(\gamma^{t})^3 + B(\gamma^{t})^2 + C\gamma^{t} + D = 0 \qquad (10)$$

where A, B, C and D are constant expressions of $\hat{D}_{ij}$'s and x_{ia}^t's.

Among solutions of (10) we choose one that minimise $\varphi^{t+1}(\gamma^t)$.

Finally, with the knowledge of γ^t, the x_{ia}^{t+1}'s are well defined by formula (5)

Second phase: rank image transformation T

The rank image transformation is necessary in order to transform the least squares method in (3) into the monotonicity condition in (1).

Let us consider the $\hat{D}_{ij}^{s}$'s at iteration *s* (iteration index of second phase) as being a series of numbers with order r_s. Moreover, we have, from first phase at iteration *t*, the distance solutions d_{ij}^{t}. The transformation T replaces the $\hat{D}_{ij}^{s}$'s by the d_{ij}^{t}'s but in the order r_s. The result is a new series $\hat{D}_{ij}^{s+1}$. The transformation T is applied after an optimal number of iterations p_t in the first phase. The value of p_t is chosen taking into account convergence and speed of the process.

Note that a natural initial solution for the second phase can be:

$$\hat{D}_{ij}^{0} = D_{ij}$$

where D_{ij}'s are the cells of the input data matrix.

Note that the order r_s is indexed (by *s*) and therefore can vary. This will be explained in the next paragraph.

Treatment of ties

The algorithm of **WSSA1** is based on the monotonicity condition:

$D_{ij} < D_{kl} \Leftrightarrow d_{ij} < d_{kl}$ for each quadruplet *(i,j,k,l)*.

This condition suppose that there is a strict order of the input coefficients D_{ij} (referred as r_s in the rank image transformation described above). The condition does not tell us what to do when there are tied values, that is $D_{ij} = D_{kl}$ for some quadruplet(s) *(i,j,k,l)*. A set of tied matrix cells for the same value is named a tied class in **WSSA1** (by analogy of the mathematical concept of "equivalence class"). More precisely, tied class number *q* is defined by:

$$C_q = \{(i,j),(k,l) \text{ such that } D_{ij} = D_{kl}\}$$

WSSA1 algorithm has chosen the following strategy to handle tied values: the portion of r_s corresponding to a tied class C_q, $(r_s|C_q)$ is reordered according to the order of the corresponding computed distances, d_{ij}^{t} of the last iteration of the first phase. This is done after the first phase and before the rank image transformation.

In practice, the user can decide that two coefficients are "equal" if they are "close" within a given tolerance. More precisely, if *T* is the tolerance, D_{ij} and D_{kl}

are considered to be tied values (in the same class) if $|D_{ij} - D_{kl}| \leq T$. In other words, a tied class number q is redefined as:

$$C_q = \{(i,j),(k,l) \text{ such that } |D_{ij} - D_{kl}| \leq T\}$$

Note that if $T = 0$ then $\bigcap_q C_q = \varnothing$, but if $T \neq 0$ the intersection can be not empty.

Now it is clear how the original order of the D_{ij}'s, say r_0, can vary after each rank image transformation.

Coefficient of alienation and criterion of convergence

We need some criterion to stop the iterative process. At iteration t, after rank image transformation, we get a value φ^t as defined in (7). This quantity can be used as a measure for a stopping criterion, but since it is of second degree (in d_{ij}^t), it is more accurate to take a "linear " quantity (in d_{ij}^t). So, we define, at iteration t:

$$\kappa^t = \sqrt{\varphi^t}$$

κ^t is called coefficient of alienation at iteration t. Now, the iterative process will be stopped when:

$$|\kappa^{t-1} - \kappa^t| \leq \varepsilon$$

where ε is a "small" number. Suppose, the process is stopped at iteration $t = t_{final}$, the final coefficient of alienation reported is:

$$\kappa = \sqrt{\varphi^{t_{final}}}$$

Initial solution for the first phase

Formula (5) tells how to make successive corrections, but does not tell where to start.

In expanding the numerator of (4), we get among the various terms, the function:

$$u = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \hat{D}_{ij} d_{ij} \qquad (11)$$

u is in fact the main term which expresses the monotonicity between $\hat{D}_{ij}$ and d_{ij}. Therefore, we can extract an initial solution considering only this function. For computation considerations, we will take the squared distances instead of the d_{ij}'s defined in (2):

$$d_{ij} = \sum_{a=1}^{m} (x_{ia} - x_{ja})^2$$

This is possible, since the squared distance is a monotone function of the distance itself. Naturally, we set in (11):

$$\hat{D}_{ij} = D_{ij}$$

After expansion, we get an initial u:

$$u_0 = \sum_{a=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ia} x_{ja} c_{ij}$$

where $\mathbf{C} = \{c_{ij}\}$ is the symmetric matrix with elements:

$$c_{ij} = \delta_{ij} \sum_{k=1}^{n} w_{ik} D_{ik} - w_{ij} D_{ij}$$

The initial solution will be get in maximising u_0, modulo a suitable normalisation. This leads us to compute the m first latent vectors of matrix **C**.

External variables

In the first step an SSA of dimensionality m is processed on a given symmetric matrix of dissimilarity/similarity coefficients between n variables, $V_1, V_2, \ldots, V_n$, called original variables. The resulting configuration of the n points in the space is then fixed. Now, let us consider a new variable, E, called external variable, which has a vector of dissimilarity/similarity coefficients with the above original variables:

$$\{D_j\}$$

where $j = 1, \ldots, n$ are the indices of the original variables.

The purpose is to locate E among the fixed n points taking into account the vector $\{D_j\}$. The requested monotonicity condition is for each duplet (j,k):

$D_j < D_k \Leftrightarrow d_j < d_k$ as much as possible

where

$$d_j = \sqrt{\sum_{a=1}^{m} (x_a - x_{ja})^2}$$

Note that the unknowns are the x_a $(a = 1,\ldots,m)$ the coordinates of the external variable, while the x_{ja} $(j = 1,\ldots,n;\ a = 1,\ldots,m)$ are the coordinates of the original variables, and therefore are constant numbers.

The loss function to minimise is:

$$\varphi = \frac{\sum_{j=1}^{n} w_j (D_j - d_j)^2}{\sum_{j=1}^{n} w_j D_j^2}$$

The coefficients w_j are weights affected to the input data D_j. w_j=0 when D_j is unknown.

The minimisation process is similar to that described above for original SSA. We have also here two phases.

The initial solution for the first phase is completely different from the original one. Since the external variable has to be located among the fixed original variables, we choose to take as initial solution the gravity centre of the fixed configuration:

$$x_a^0 = \frac{1}{n} \sum_{j=1}^{n} x_{ja}$$

Comparison between algorithms of Original SSA and External variable SSA

	Original SSA	Ext. Vars. SSA
Input Data	symmetric matrix $\{D_{ij}\}$	vector $\{D_j\}$
Output (Solution)	$x_{ia}\ (i=1,\ldots,n;\ a=1,\ldots,m)$	$x_a\ (a=1,\ldots,m)$
Euclidean Distance (Output)	$d_{ij}=\sqrt{\sum_{a=1}^{m}(x_{ia}-x_{ja})^2}$	$d_j=\sqrt{\sum_{a=1}^{m}(x_a-x_{ja})^2}$
Monotonicity condition	$D_{ij}<D_{kl}\Leftrightarrow d_{ij}<d_{kl}$	$D_j<D_k\Leftrightarrow d_j<d_k$
Loss function	$\varphi=\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}w_{ij}(D_{ij}-d_{ij})^2}{\sum_{i=1}^{n}\sum_{j=1}^{n}w_{ij}D_{ij}^2}$	$\varphi=\frac{\sum_{j=1}^{n}w_j(D_j-d_j)^2}{\sum_{j=1}^{n}w_jD_j^2}$
Initial solution	eigenvectors of matrix $\mathbf{C}=\{c_{ij}\}$ $c_{ij}=\delta_{ij}\sum_{k=1}^{n}w_{ik}D_{ik}-w_{ij}D_{ij}$	$x_a^0=\frac{1}{n}\sum_{j=1}^{n}x_{ja}$

Reference

Guttman, L. A general nonmetric technique for finding the smallest coordinate space for a configuration of points. Psychometrika, 1968, 33, 469-506.

POSAC

Statement of the problem

Let **P** a set of *N* profiles. If $c_k \in \mathbf{P}$ we note:

$$c_k = (c_k^1, c_k^2, \ldots, c_k^n) \qquad c_k^i \in \mathbf{R}$$

Let $c_k, c_l \in \mathbf{P}$,we define a partial order on **P** as follows:

$$\begin{aligned}
&c_k = c_l \Leftrightarrow c_k^i = c_l^i \qquad i = 1,\ldots,n \\
&c_k > c_l \Leftrightarrow c_k \text{ not} = c_l \text{ and } c_k^i \geq c_l^i \qquad i = 1,\ldots,n \\
&c_k < c_l \Leftrightarrow c_k \text{ not} = c_l \text{ and } c_k^i \leq c_l^i \qquad i = 1,\ldots,n \\
&c_k \# c_l \Leftrightarrow c_k \textit{not} = c_l \text{ and } c_k \textit{not} > c_l \text{ and } c_k \textit{not} < c_l
\end{aligned}$$

When $c_k = c_l$, $c_k > c_l$ or $c_k < c_l$ the profiles c_k and c_l are said to be **comparable**. When $c_k \# c_l$, they are said to be **incomparable**. The symbol "#" was chosen for incomparability relation (do not confuse with symbol "≠" which means "not equal to")..

The purpose is to find an application :

$$\mathbf{P} \longrightarrow \mathbf{R}^2$$
$$c_k \longrightarrow z_k = (x_k, y_k)$$

such that

$$\begin{aligned}
&c_k > c_l \Leftrightarrow z_k > z_l \\
&c_k < c_l \Leftrightarrow z_k < z_l \\
&c_k \# c_l \Leftrightarrow z_k \# z_l
\end{aligned} \qquad (1)$$

"as well as possible" for every pair of profiles $c_k, c_l \in \mathbf{P}$.

In other words, we would like to reduce the number of variables from "*n*" to 2 and still conserve the partial order as well as possible.

The solution is noted:

$x = \{x_k\}_{k=1,2,\ldots,N}$ and $y = \{y_k\}_{k=1,2,\ldots,N}$

Note that in (1) equality of profiles is not handled. Indeed, without loss of generality, all profiles in **P** are supposed to be different, because equality is expressed throughout the frequency of each profile. So at each profile c_k is attached a frequency f_k.

Determination of a loss function

The problem stated above can be formulated as a minimisation of a certain function φ to be now developed.

We can express conditions (1) in term of inequalities.

For comparability :

$$\begin{aligned} c_k > c_l &\Leftrightarrow x_k > x_l \quad \text{and} \quad y_k > y_l \\ c_k < c_l &\Leftrightarrow x_k < x_l \quad \text{and} \quad y_k < y_l \end{aligned} \tag{2}$$

For incomparability :

$$c_k \# c_l \Leftrightarrow \begin{cases} x_k > x_l \quad \text{and} \quad y_k < y_l \\ \qquad\quad \text{or} \\ x_k < x_l \quad \text{and} \quad y_k > y_l \end{cases} \tag{3}$$

In fact, all what we need from the input profiles is their partial order. Therefore let us define a signature on **P** based on partial order. For each pair $c_k, c_l \in \mathbf{P}$

$$\gamma_{kl} = \begin{cases} 1 \ \text{if} \ c_k > c_l \\ -1 \ \text{if} \ c_k < c_l \\ 0 \ \text{if} \ c_k \# c_l \end{cases}$$

The two conditions on comparability in (2) can be expressed by :

$$\gamma_{kl}(x_k - x_l) \geq 0 \quad \text{and} \quad \gamma_{kl}(y_k - y_l) \geq 0 \tag{4}$$

The condition on incomparability in (3) can be expressed by :

$$(|\gamma_{kl}| - 1)(x_k - x_l)(y_k - y_l) \geq 0 \tag{5}$$

Inequalities (4) and (5) can be added together since they are mutually exclusive and the contribution of each pair (z_k, z_l) can be then measured by the following term :

$$u_{kl} = \gamma_{kl}[t_1(x_k - x_l) + t_2(y_k - y_l)] + \\ (|\gamma_{kl}| - 1)t_3(x_k - x_l)(y_k - y_l)$$

where t_1, t_2 and t_3 are positive factors to be determined in such a way that the 3 terms in u_{kl} become mutually normalised and differentiable. It was found that the optimal choice is:

$$t_1 = (y_k - y_l)^2 |x_k - x_l|$$
$$t_2 = (x_k - x_l)^2 |y_k - y_l|$$
$$t_3 = 2|x_k - x_l||y_k - y_l|$$

Finally, u_{kl} becomes:

$$u_{kl} = \gamma_{kl}[(y_k - y_l)^2 |x_k - x_l|(x_k - x_l) + \\ (x_k - x_l)^2 |y_k - y_l|(y_k - y_l)] + \\ 2(|\gamma_{kl}| - 1)|x_k - x_l||y_k - y_l|(x_k - x_l)(y_k - y_l) \qquad (6)$$

Note that in order to make u_{kl} differentiable, we used the fact that $|q|q$ is a differentiable function of q even at $q = 0$.

Using the triangular inequality and the following feature:

$$||\gamma_{kl}| - 1| = 1 - |\gamma_{kl}|$$

we can find an upper bound of $|u_{kl}|$:

$$|u_{kl}| \leq 2(x_k - x_l)^2 (y_k - y_l)^2$$

Let us note this upper bound:

$$v_{kl} = 2(x_k - x_l)^2 (y_k - y_l)^2 \qquad (7)$$

From (6) and (7), we see that each pair of profiles (c_k, c_l) contributes with a weight of $(x_k - x_l)^2 (y_k - y_l)^2$, that is, larger weight is assigned to profiles that are mapped farther apart. This is not desirable. Therefore, in order to "balance" this weight, we multiply u_{kl} and v_{kl} by weights which come from the information we have on the original profiles. These are called balancing weights.

For a profile $c_k = (c_k^1, c_k^2, \ldots, c_k^n)$ we define its score by:

$$S_k = \sum_{a=1}^{n} c_k^a$$

Given the set **P** of profiles, its maximal profile is defined by:

$$c_{k_M} = (c^1_{k_M}, c^2_{k_M}, \ldots, c^n_{k_M}) \qquad \text{where } c^a_{k_M} = \max_k(c^a_k)$$

Similarly, its minimal profile is defined by:

$$c_{k_m} = (c^1_{k_m}, c^2_{k_m}, \ldots, c^n_{k_m}) \qquad \text{where } c^a_{k_m} = \min_k(c^a_k)$$

Note that c_{k_M} or c_{k_m} may or may not be in **P**.

The balancing weights are defined by:

$$w_{kl} = \begin{cases} \dfrac{[(S_{k_M} - S_{k_m}) - |\, S_k - S_l \,|]^d}{(S_{k_M} - S_{k_m})^d} & \text{if} \quad d > 0 \\ 1 & \text{if} \quad d = 0 \end{cases}$$

It was found that $d = 4$ is a suitable value.

Besides these weights, there are another kind of weights related to the input data: the frequency of each profile. It is natural to give for a pair of profiles (c_k, c_l) a weight of $f_k f_l$.

The contribution of all pairs of profiles (c_k, c_l) in **P**, taking into account the two kinds of weights, is:

$$u = \sum_{k=1}^{N} \sum_{l=1}^{N} w_{kl} f_k f_l u_{kl}$$

where u_{kl} is as defined in (6). An upper bound of $|u|$ is the function:

$$v = \sum_{k=1}^{N} \sum_{l=1}^{N} w_{kl} f_k f_l v_{kl}$$

where v_{kl} is as defined in (7). Finally, the function to minimise is:

$$\varphi = v - u \qquad (8)$$

Note that the lower bound of φ is 0. The minimisation of φ will not yet solve our problem, because the quantities $|x_k - x_l|$ and $|y_k - y_l|$ can become smaller in the minimisation process without satisfying conditions (2) and (3). This degeneracy problem is solved by the transformation T of the second phase (see later).

Minimisation of φ

The minimisation of φ in (8) is handled by the following iterative process of two phases.

In a first step, the input profiles $\{c_k\}_{k=1,\ldots,N}$ are sorted in such a way that the index of the maximal profile k_M equals 1 and the index of the minimal profile k_m equals *N*. For these extreme profiles, the solution is obvious and fixed to:

$(x_1, y_1) = (1,1)$
$(x_N, y_N) = (0,0)$

Therefore, computations are done only for profiles of indices:

$h = 2,\ldots, N-1$

First phase: steepest descent procedure

The iterative process is as follows:

$$x_h^{t+1} = x_h^t - \lambda^t \frac{\partial \varphi^t}{\partial x_h^t} \qquad h = 2,\ldots, N-1$$
$$y_h^{t+1} = y_h^t - \lambda^t \frac{\partial \varphi^t}{\partial y_h^t} \qquad (9)$$

where:

t : iteration index
λ^t : a step to be found (optimised)
$\varphi^t = \varphi(x^t, y^t)$: function φ at iteration t

At iteration t we have a solution:

$x^t = \{x_h^t\}_{h=1,2,\ldots,N}$ and $y^t = \{y_h^t\}_{h=1,2,\ldots,N}$.

On the basis of this solution, the loss function is computed:

$\varphi^t = \varphi(x^t, y^t)$

For each h (point index: h=2,...,N-1), the following partial derivatives are computed:

$$\frac{\partial \varphi^t}{\partial x_h^t} \quad \text{and} \quad \frac{\partial \varphi^t}{\partial y_h^t} \qquad (10)$$

At iteration t+1, from (9) we could compute x_h^{t+1} and y_h^{t+1} if λ^t was known. In order to solve λ^t, we consider it as an unknown. Consequently, x^{t+1}, y^{t+1} and φ^{t+1} become functions of the single variable λ^t. Since we look for a minimisation of φ (as a multivariate function of x and y), the minimisation of φ^{t+1} (as a univariate function of λ^t) will provide us the optimal step λ^t corresponding to iteration t. In other words:

$$\varphi^{t+1} = \varphi(x^{t+1}, y^{t+1}) = \varphi(\lambda^t)$$

The iterative process in (9) is carried out until there is no improvement of φ. Then, we go to the second phase.

Second phase: uniformity transformation T

The uniformity transformation is necessary in order to avoid the degeneracy problem.

After an optimal number of iterations p_t in the first phase, we get a solution.

$$x = \{x_h\}_{h=1,2,\ldots,N} \quad \text{and} \quad y = \{y_h\}_{h=1,2,\ldots,N}$$

The transformation T respaces the x_h's and the y_h's on the interval [0,1] in an equidistant manner, keeping their order. For example:

if x = {0.57, 0.12, 0.20, 1.10, 0.80} then

T(x) = {0.50, 0.00, 0.25, 1.00, 0.75}

With this new solution we restart the first phase, and so on. The whole process may be stopped when $\varphi \le \varepsilon$ for some selected ε.

Initial approximation

Let v_a $(a = 1,\ldots,n)$ the variables on which the profiles are defined. That is if profile

$c_k = (c_k^1, c_k^2, \ldots, c_k^n)$ then c_k^a is a category value of v_a.

A matrix of monotonicity coefficients are computed between all couples (v_a, v_b). The idea is to choose 2 variables among the n variables such that they are the less monotonously correlated.

Let r_{ab} be the monotonicity coefficient between v_a and v_b.

$$r_{ab} = \frac{\sum_{k=1}^{N}\sum_{l=1}^{N} f_k f_l (c_k^a - c_l^a)(c_k^b - c_l^b)}{\sum_{k=1}^{N}\sum_{l=1}^{N} f_k f_l |c_k^a - c_l^a||c_k^b - c_l^b|}$$

Let a_1, a_2 be variable indices such that $r_{a_1 a_2} = \min_{a,b} r_{ab}$ and let $S_k = \sum_{a=1}^{n} c_k^a$ be the profile score. Then, the initial solution is defined by:

$x_k^0 = S_k + c_k^{a_1} - c_k^{a_2}$
$y_k^0 = S_k - c_k^{a_1} + c_k^{a_2}$

This initial solution has nothing to do with partial order, but it had be found to be a good initial approximation.

Reference

Shye, S. and Amar, R. Partial order scalogram analysis by base coordinates and lattice mapping of the items by their scalogram roles. In D. Canter (Ed.), Facet Theory: Approaches to Social Research. New York: Springer Verlag, 1985, pp. 277-298.

PEARSON

Fitting a regression line

Let x and y be two numerical variables represented by N pairs (x_i, y_i). By a least square method, we can draw a regression line of variable y in regard to variable x. More precisely, we have to find the parameters a and b of the line equation $y = ax + b$. Let d_i be the algebraic deviation of point (x_i, y_i) from the line:

$$d_i = y_i - (ax_i + b)$$

The best fitting line is that for which $\sum_{i=1}^{N} d_i^2$ is minimum. The computations lead to the following results:

$$a = \frac{N\sum_{i=1}^{N} x_i y_i - \left(\sum_{i=1}^{N} x_i\right)\left(\sum_{i=1}^{N} y_i\right)}{N\sum_{i=1}^{N} x_i^2 - \left(\sum_{i=1}^{N} x_i\right)^2} \qquad b = \frac{\left(\sum_{i=1}^{N} x_i^2\right)\left(\sum_{i=1}^{N} y_i\right) - \left(\sum_{i=1}^{N} x_i\right)\left(\sum_{i=1}^{N} x_i y_i\right)}{N\sum_{i=1}^{N} x_i^2 - \left(\sum_{i=1}^{N} x_i\right)^2}$$

The above expressions become more simple if we introduce the mean values of x and y:

$$a = \frac{\sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{N} (x_i - \bar{x})^2} \qquad b = \frac{\bar{y}\sum_{i=1}^{N} (x_i - \bar{x})^2 - \bar{x}\sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{N} (x_i - \bar{x})^2}$$

where $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$ $\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$

Definition

Let x and y be two numerical variables represented by N pairs (x_i, y_i). As seen in the previous paragraph, one can draw a regression line of variable y in regard to

variable x. Let its equation be $y = a_x x + b_x$. The same thing holds for the regression line of variable x in regard to variable y, $x = a_y y + b_y$. Applying the least square method, the slopes of the two lines are:

$$a_x = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{N}(x_i - \bar{x})^2} \qquad a_y = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

By definition, the Pearson coefficient of linear correlation is the square root of the two regression line slopes product.

$$r(x,y) = \sqrt{a_x a_y}$$

$$r(x,y) = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2 \sum_{i=1}^{N}(y_i - \bar{y})^2}}$$

Reference

Monjallon, A. Introduction a la methode statistique. Paris: Librairie Vuibert, 1961.

Intraclass Correlation

Terminology

The classical terms of variables and cases in Statistics have other terminology when an intraclass correlation coefficient has to be defined. A case or observation is called a **class**. Each class, has a number of **members**, and each member have a **score**. The classes can have an unequal number of members.

Definition

Suppose we have n variables v_j $(j = 1,\ldots,n)$ defined on N classes (or observations). These variables have to be the same measure of a certain item collected on n members (say, for example, IQ measured on brothers of the same family). In fact, n here is the maximal number of members among all the classes. More precisely, let x_{ij} be the value of variable v_j for class i, or, in other words, the score of member j in class i. In order to allow an unequal number of members, the following factor is defined:

$$w_{ij} = \begin{cases} 1 \text{ if } x_{ij} \text{ exists} \\ 0 \text{ otherwise} \end{cases}$$

So the number of members in class i is given by:

$$k_i = \sum_{j=1}^{n} w_{ij} \qquad \text{and it's average is} \qquad \bar{k} = \frac{1}{N-1}\left(\sum_{i=1}^{N} k_i - \frac{\sum_{i=1}^{N} k_i^2}{\sum_{i=1}^{N} k_i} \right)$$

Let $\bar{x}_i = \dfrac{\sum_{j=1}^{n} w_{ij} x_{ij}}{\sum_{j=1}^{n} w_{ij}}$ be the mean value of $\{x_{ij}\}$ within class i

and let $\bar{x} = \dfrac{\sum_{i=1}^{N}\sum_{j=1}^{n} w_{ij}x_{ij}}{\sum_{i=1}^{N}\sum_{j=1}^{n} w_{ij}}$ be the total mean value of $\{x_{ij}\}$

The intraclass correlation coefficient is then defined by:

$$R = \frac{BCMS - WMS}{BCMS + (\bar{k} - 1)WMS}$$

where:

$BCMS = \dfrac{1}{N-1}\sum_{i=1}^{N} k_i(\bar{x}_i^2 - \bar{x}^2)$ is the between classes mean square

$WMS = \dfrac{1}{\sum_{i=1}^{N}(k_i - 1)}\sum_{i=1}^{N}\sum_{j=1}^{n} w_{ij}(x_{ij}^2 - \bar{x}_i^2)$ is the within classes mean square

Reference

Haggard, E.A. Intraclass Correlation and the Analysis of Variance. New York: The Dryden Press, Inc.

Appendix B

HUDAP for Windows

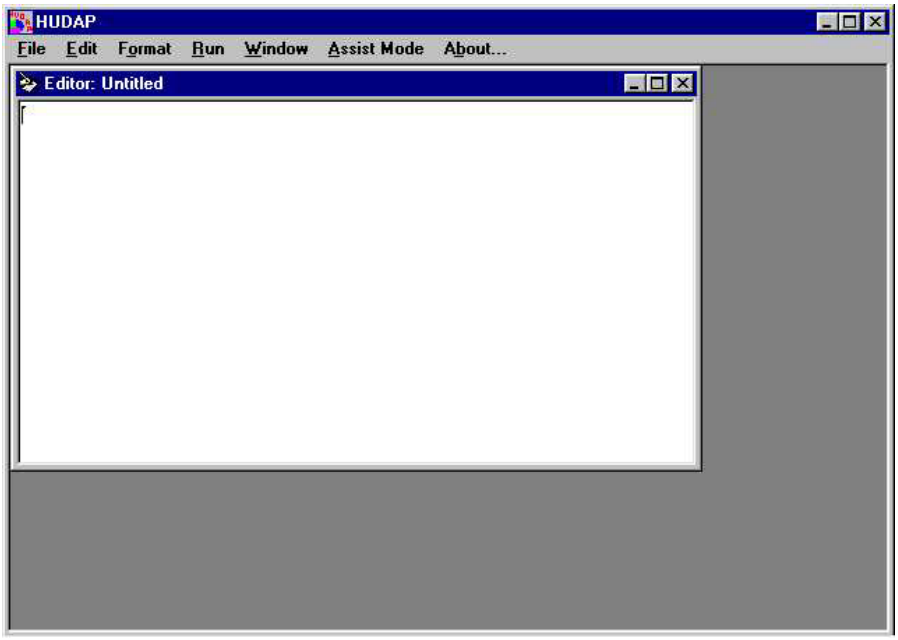
Hudap for Windows

Starting Hudap

Click the **Start** button and then click **Programs**. From the **Programs** menu choose Hudap

The Hudap Screen

When you start Hudap, the following window appears:



This is the Edit Mode, the starting mode.

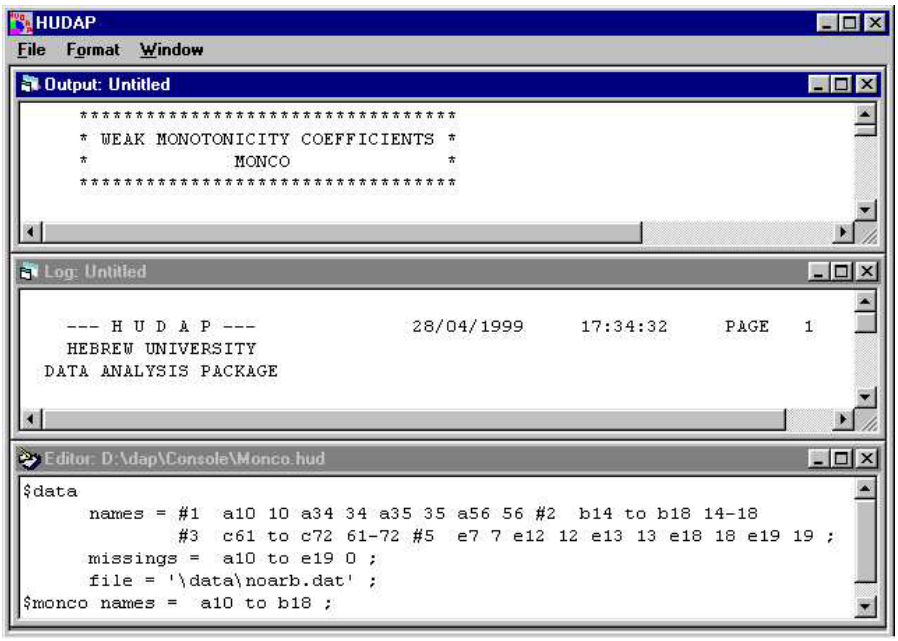
There are two modes in using Hudap for Windows:

- Edit Mode
- Assist Mode

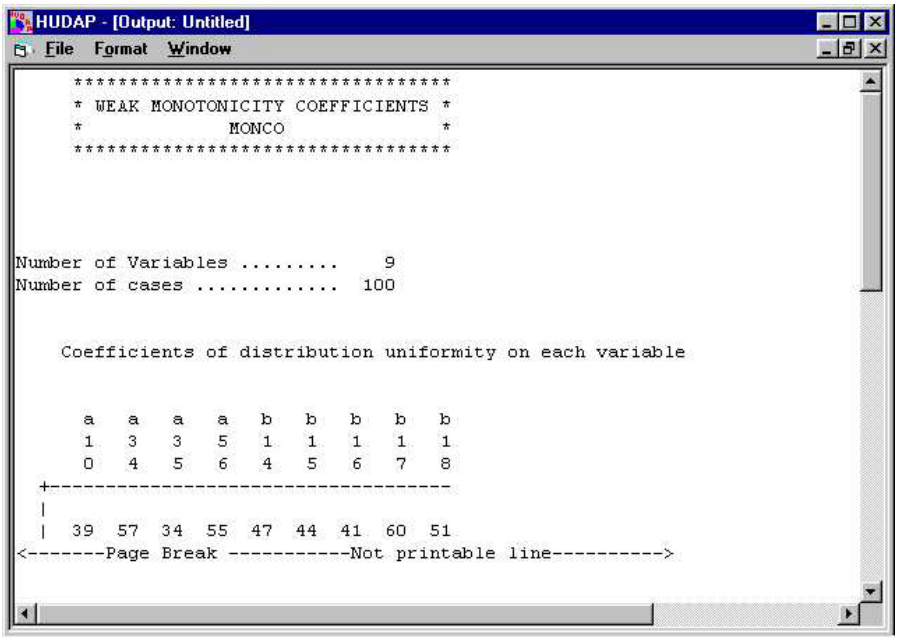
To switch to the Assist Mode, click **Assist Mode** on the menu Bar.

Edit Mode

In this mode, you can write the job in Hudap language in the area of the Editor window or load an existing job file by means of the **Open** command from the **File** menu. Thereafter you submit the job in clicking **Run** command on the menu Bar. When the process is completed, you get the following windows:



If the process is successful a window containing the Output file appears. You can maximize the Output window:



Assist Mode

The following window appears when choosing **Assist Mode** menu:

In this mode, you do not need to learn Hudap language. Instead, various menus and windows drive you to process the main Hudap analyses. In fact, according to your definitions and choices, Hudap system generates a job file in Hudap language. This job remains in background unless you choose to view it by clicking **View Job** command from **File** menu. This job is then processed if you click **Run** command on the menu Bar.

Two illustrations are presented further:

- Wssa1 Analysis in Edit Mode
- Disco Analysis in Assist Mode

WSSA1 analysis in Edit Mode

In this example we assume that a file, **`States.hud`**, was already written in Hudap language for processing a WSSA1 analysis on a matrix of correlations, **`States.mat`** file. This example is the same that was presented in WSSA1 section of Hudap manual. Here we will focus on the regionalization features of the Windows Hudap version. Following the input job file **`States.hud.`**

```
$set  linesize = 80 ;
$matrinp names =   v1 to v15 1-45 ;
         varlabs =
                v1    'City as place to live'
                v2    'Neighborhood'
                v3    'Housing'
                v4    'Life in the U.S.'
                v5    'Amount of education'
                v6    'Useful education'
                v7    'Job'
                v8    'Spending of spare time'
                v9    'Health'
                v10   'Standard of living'
                v11   'Savings and investments'
                v12   'Friendships'
                v13   'Marriage'
                v14   'Family life'
                v15   'Life in general' ;
         file = 'States.mat' ;
$wssa1 names =  v1 to v15  ;
       mindim = 2 ;
       maxdim = 2 ;
       facets
             nfacets = 2 /
             profiles =
                    v1    2 3
                    v2    2 3
                    v3    2 3
                    v4    2 3
                    v5    2 1
                    v6    2 1
                    v7    1 7
                    v8    1 4
                    v9    2 6
                    v10   1 2
                    v11   2 2
                    v12   2 4
                    v13   2 5
                    v14   1 5
                    v15   1 8  ;
```

Note that two facets are defined in this file. Facet A has 2 elements while facet B has 8 elements.

```
100 54 44 33 19 14 22 22 05 33 25 24 14 24 28
 54100 49 28 18 14 21 19 00 32 23 19 13 19 23
 44 49100 29 23 19 26 27 06 45 29 23 21 23 30
 33 28 29100 12 15 23 23 06 24 19 21 13 21 24
 19 18 23 12100 54 25 26 18 32 28 16 09 18 28
 14 14 19 15 54100 24 23 17 24 20 17 12 18 24
 22 21 26 23 25 24100 33 13 35 27 25 25 27 34
 22 19 27 23 26 23 33100 21 37 32 40 30 40 50
 05 00 06 06 18 17 13 21100 17 17 09 12 14 26
 33 32 45 24 32 24 35 37 17100 59 25 25 32 45
 25 23 29 19 28 20 27 32 17 59100 24 23 25 36
 24 19 23 21 16 17 25 40 09 25 24100 21 31 32
 14 13 21 13 09 12 25 30 12 25 23 21100 48 38
 24 19 23 21 18 18 27 40 14 32 25 31 48100 50
 28 23 30 24 28 24 34 50 26 45 36 32 38 50100
```

`States.mat:` input matrix file

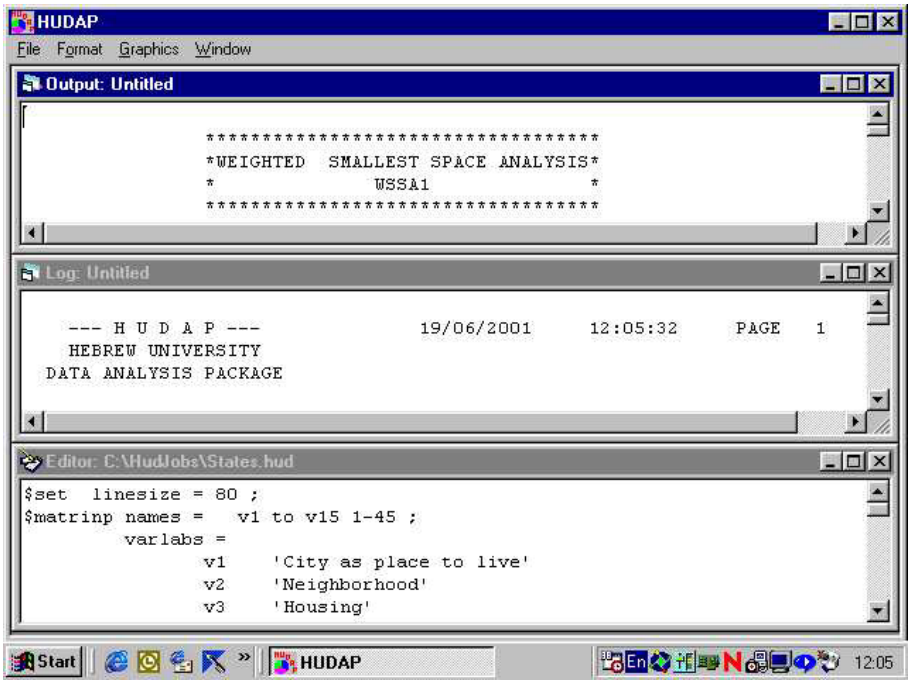
A typical sequence of operations

Let us describe a scenario of operations handled by a user in activating Hudap under Windows.

1. Start Hudap

2. Load **`States.hud`** file by means of the **Open** command from the **File** menu.

3. Click **Run** command on the menu bar. When the process is completed, you get the following window:

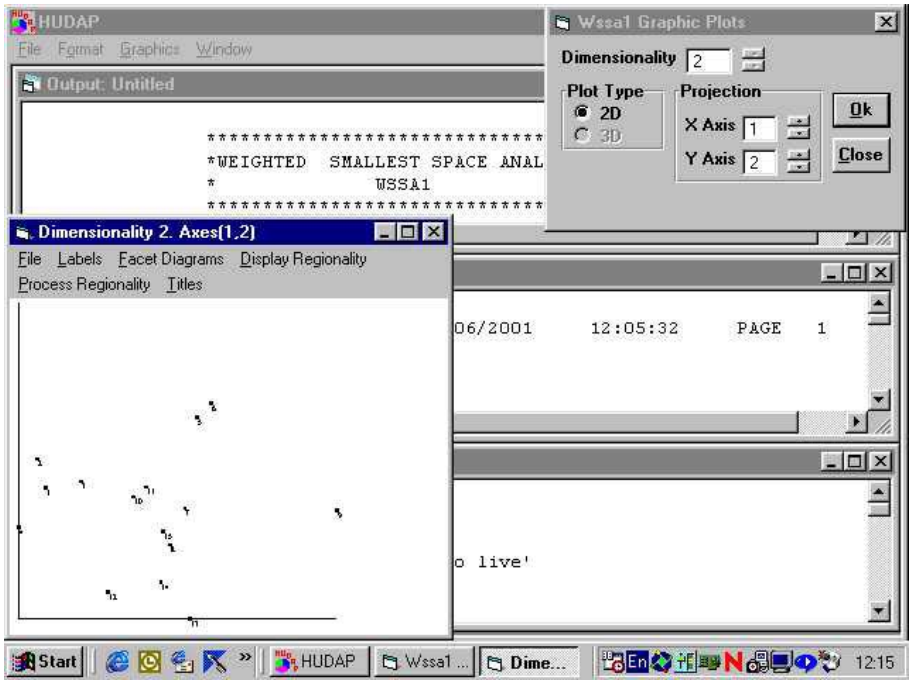


The upper window is the output where the complete process is given as a text file. You can maximize it and scroll it to see the results.

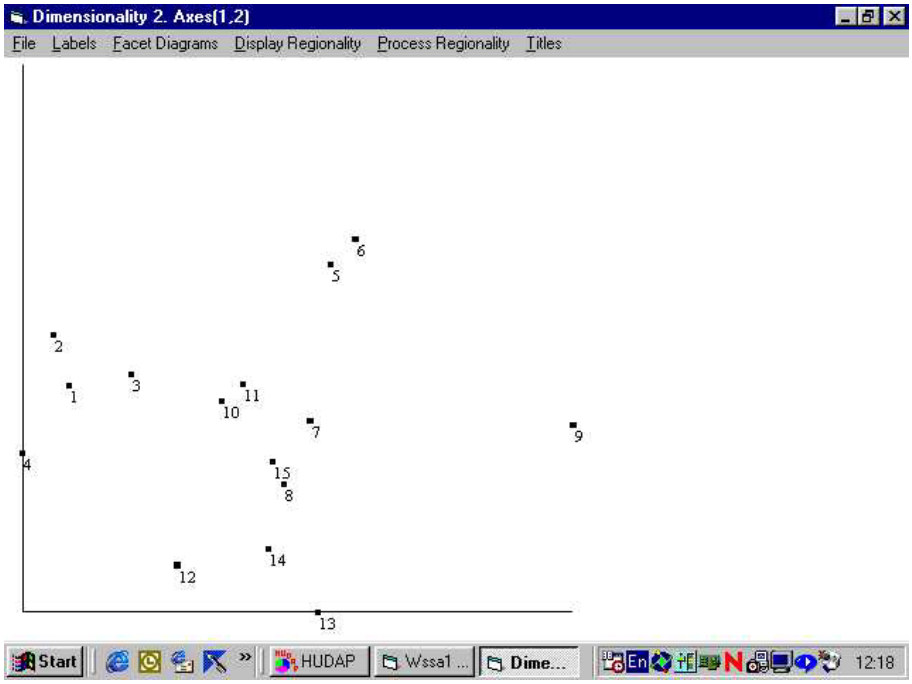
But let us focus on the graphic features of WSSA1:

4. Click on **Graphics** menu and choose **1. WSSA1** submenu (submenus are numbered because several analyses can be requested in a single job). A small window, **WSSA1 Graphics Plots**, appears at the upper right corner of the screen. Through this window you can choose the dimensionality, the plot type (2D or 3D) and the specific projection.

5. Keep the defaults and click **Ok**. A window containing the plot appears at the lower left corner of the screen.

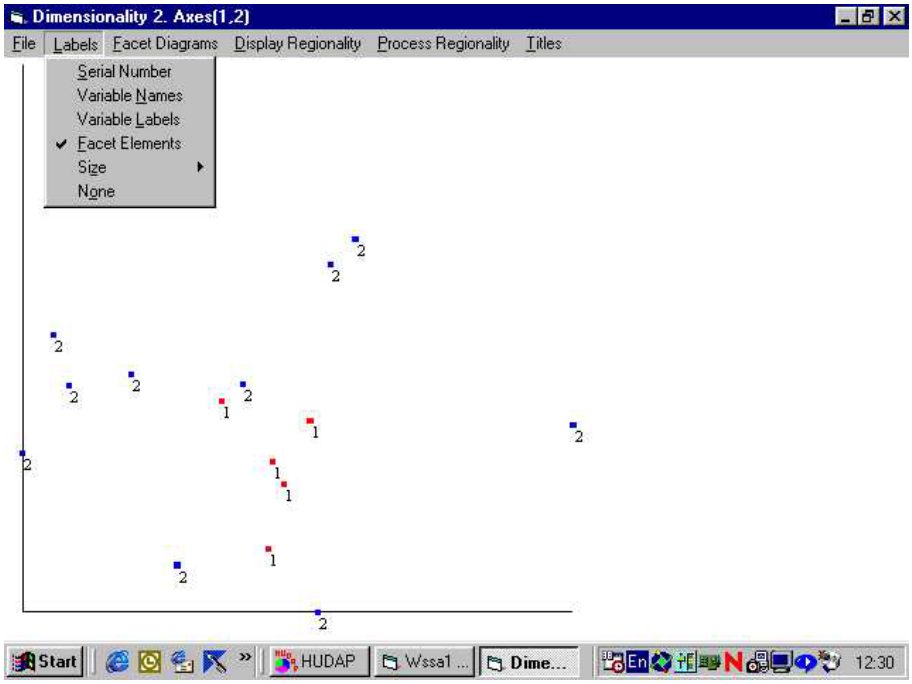


6. Maximize the plot window to allow more convenient operations. We get the following window. This window contains several menus helping the researcher to explore more efficiently the results.



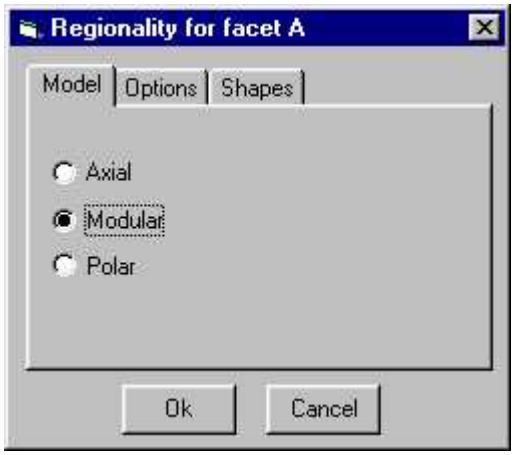
7. Click on **Facet Diagrams** menu and choose **Facet A**. Variables 7, 8, 10, 14 and 15 become red; the remaining variables become blue, according to the user definition of facet A in the Hudap job. This facet contains two elements numbered 1 and 2.

8. Click on **Labels** menu and choose **Facet Elements** submenu to display facet elements as labels instead of the serial numbers of the variables.



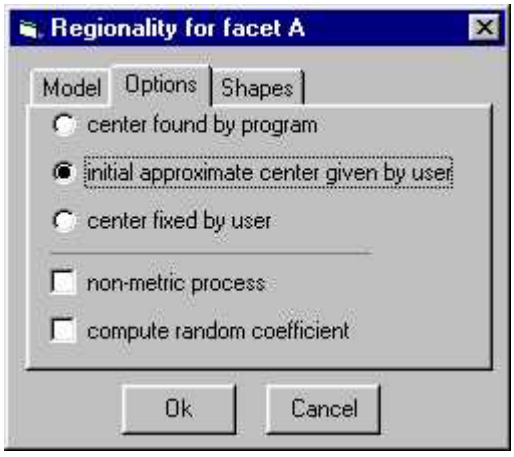
We can see on the plot that facet A has a modular role. The **Process Regionality** tool can check this.

9. Click on **Process Regionality** menu and choose **Facet A**. The following window appears.



This window contains 3 tabs: **Model**, **Options** and **Shapes**.

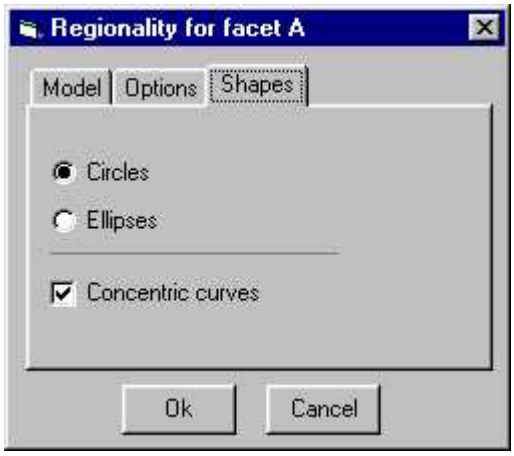
10. Click on **Modular** option of the **Model** Tab to request a modular model for facet A. At this stage you can click on **Ok** button and the regionalization process will begin. But, in order to practice useful options, click on **Options** Tab. The following options are displayed.



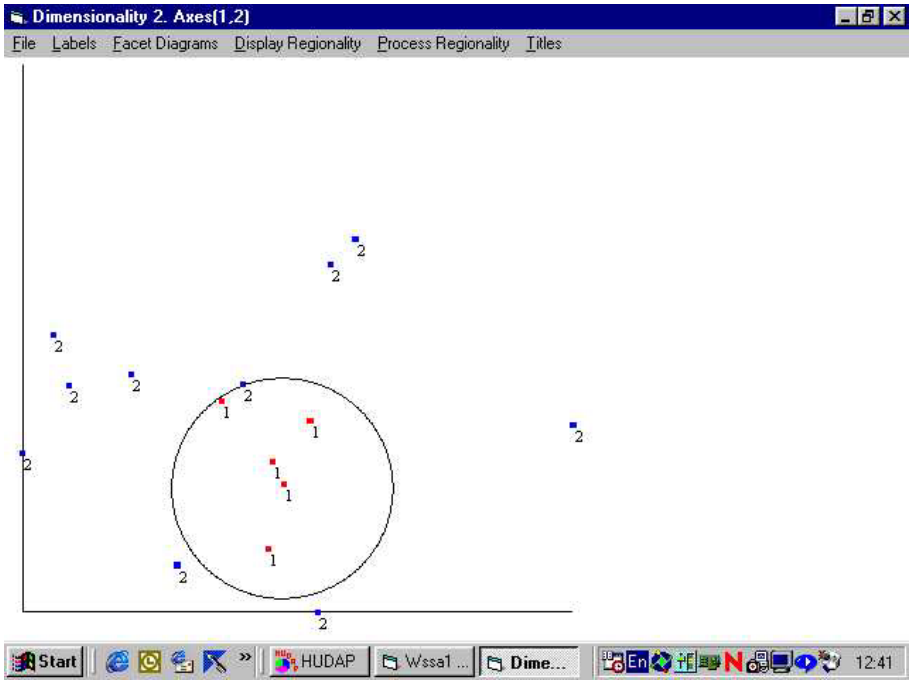
11. Choose **initial approximate center given by user** option.

The regionalization is a mathematical iterative process. Therefore it is necessary to start with an initial approximation of the solution. In the modular model, the solution is a series of nested circles (or ellipses). The circles can be concentric or not (see **Shapes** Tab). When the user chooses this second option, he chooses in fact an initial approximate center of the inner circle. In our case, facet A has only 2 elements, therefore, there is only one circle separating these elements.

12. Click on **Shapes** Tab. For modular model, the curves can be circles or ellipses, concentric or non-concentric. Keep the defaults as shown in the following window.



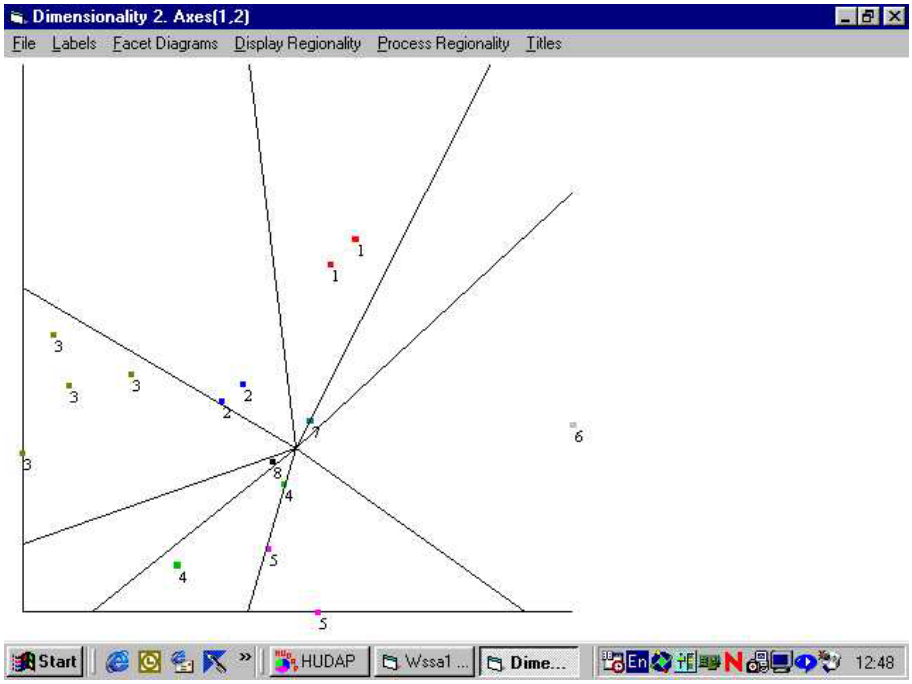
13. Click on **Ok** button to begin the regionalization process. The above window disappears and a cross hair pointer appears on the plot area. Position the cross in the region of element 1 in order to be an initial approximate center of the circle. Left click the mouse. A circle is drawn on the plot. If the solution is not satisfactory, move the cross to another position and left click again. A new circle is drawn. Repeat this operation until you get a good solution. Right click the mouse to get an arrow pointer. The regionalization for facet A is shown in the following window. In this case it is a perfect solution and the regionality coefficient equals 1.



14. Click on **Display Regionality** menu. This menu allows the user to display or hide the regionalization solution for each facet. At this stage, you will see as first submenu **Facet A Modular (concentric circles) 1.00**. This submenu is checked to tell us that the circle is displayed. Click on it. The circle disappears, but the solution is still stored in memory. You can redisplay it at any time. Since we will concentrate now on facet B, it is recommended to hide the circle meantime. Note that the submenu for facet B is disabled since no solution for this facet is yet available.

To process the regionalization for facet B:

15. Repeat the sequence of operations from step 7 to step 13, but select **Polar** model for facet B. The solution for facet B is shown in the following window. Here too, it is a perfect solution and the regionality coefficient equals 1.

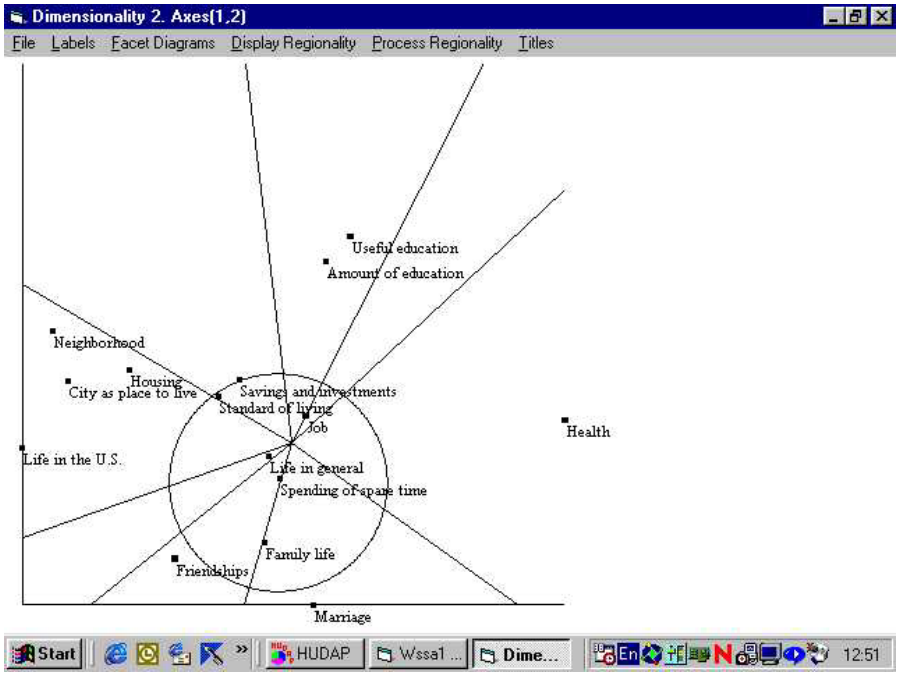


16. Click on **Display Regionality** menu. Click on the submenu of facet A. The circle reappears. The submenu of facet B, **Facet B Polar 1.00**, is already checked. So, the two solutions are displayed simultaneously.

17. Click on **Facet Diagrams** menu and choose **None**, to suppress the color of the last facet, since the regionalization is displayed for the two facets.

18. Click on **Labels** menu and choose **Variable Labels** submenu. The following window shows the final plot. This plot can be saved as a graphic BMP file by means of the **Save Graphic As…** command from the **File** menu.

Thereafter, you can insert the graphic file, for example in a document of Microsoft Word or any other word processor


Dimensionality 2. Axes(1,2)
File Labels Facet Diagrams Display Regionality Process Regionality Titles
Useful education
Amount of education
Neighborhood
Housing
City as place to live
Savings and investments
Standard of living
Job
Health
Life in the U.S.
Life in general
Spending of spare time
Family life
Friendships
Marriage
Start
HUDAP
Wssa1 ...
Dime...
12:51

Disco analysis in Assist Mode

Let us describe the different steps to process a typical Disco analysis.

Before activating Hudap, you must have on your disk a data file in ASCII format. You must note the position of each variable in this file.

Suppose you have ten numerical variables located from column 1 to column 10 in the data file, and a grouping variable “sex” located in column 12.

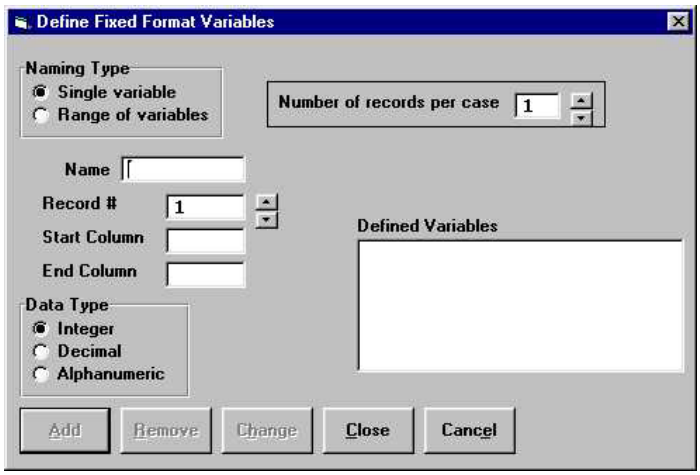
Start Hudap and click on **Assist Mode**. The above main window appears.

Data definition

The first step is to name the variables and specify their location in the data file. You have also to select the data file pathname.

To name and locate variables

From **Data** menu, choose **Raw Data**, **Define Variables**, then **Fixed Format**. The following window appears:



In the **Naming Type** frame there are two options: **Single variable** and **Range of variables**. The last option is useful when the user wants to define a sequence of adjacent variables using the "To" convention. Let us define our ten variables[1]:

1. Choose Range of variables option
2. In From Name box type `v1`
3. In To Name box type `10`
4. In Start Column box type `1`
5. In End Column box type `10`
6. Click on Add button

In **Defined Variables** box appears: **`#1 v1 to v10 1-10`**

1. Define now the variable "sex":
2. Choose **Single variable** option
3. In **Name** box type **`sex`**
4. In **Start Column** box type **`12`**
5. Click on **Add** button

In **Defined Variables** box appears: **`#1 sex 12`**

At this stage all variables are defined. Click on **Close** button.

To select the data file name

From **Data** menu, choose **Raw Data** then **Select Data File**. The following window appears:

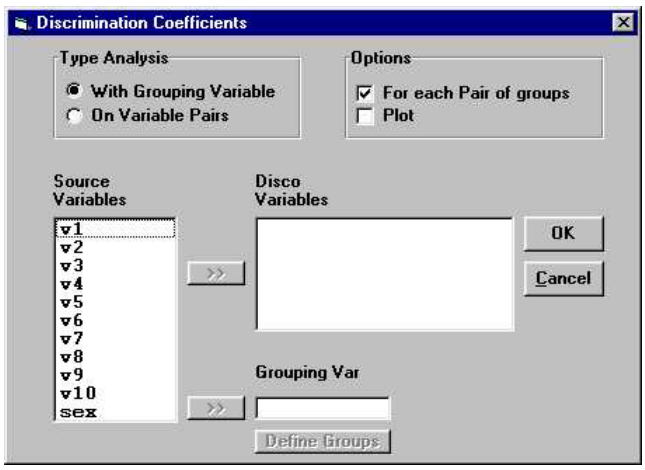
Choose the Directory and File Name requested. Click **OK**.

The Data definition step is completed.

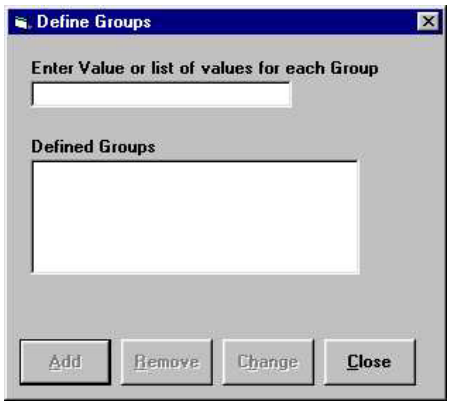
[1] Note: To skip from box to box press on Tab key.

Defining Disco Analysis

From **Analysis** menu, choose **Disco**. The following window appears:



From **Source Variables** box select variables **`v1`** to **`v10`** using the mouse
Click on the upper select button >>
From **Source Variables** box select variable **`sex`**
Click on the lower select button >>
Click on **Define Groups** button. The following window appears:



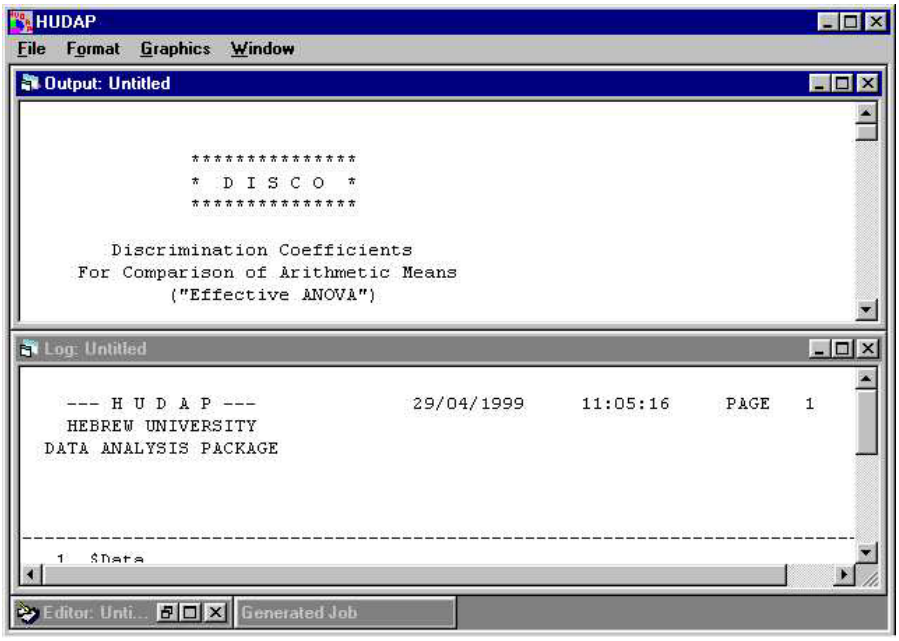
Type 1 (for males)
Click on **Add** button
Type 2 (for females)
Click on **Add** button
Click on **Close** button

Click on **OK** button of Disco window.

The Analysis definition step is completed.

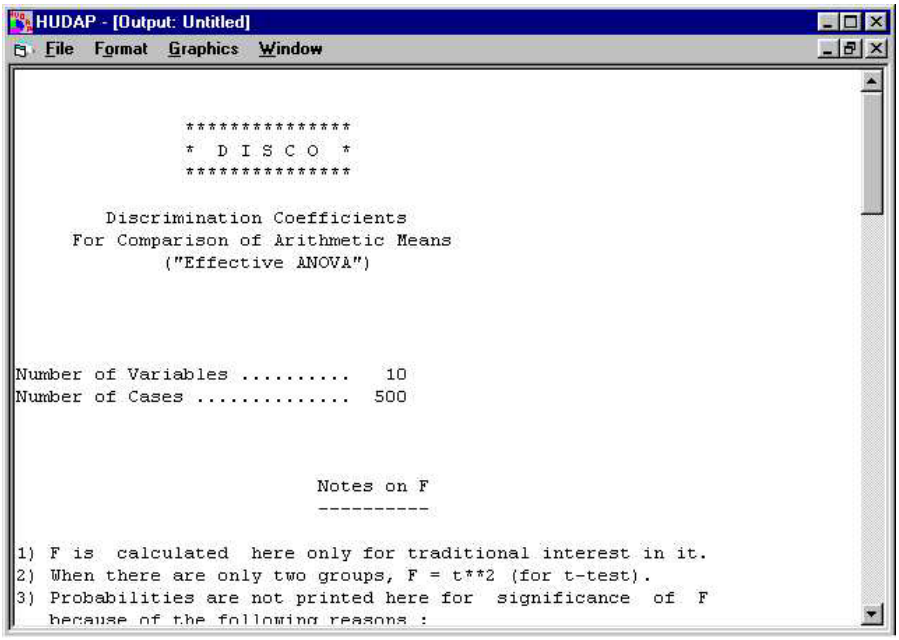
Process and results

To process the requested analysis click **Run** command on the menu Bar. When the process is completed, you get the following window:

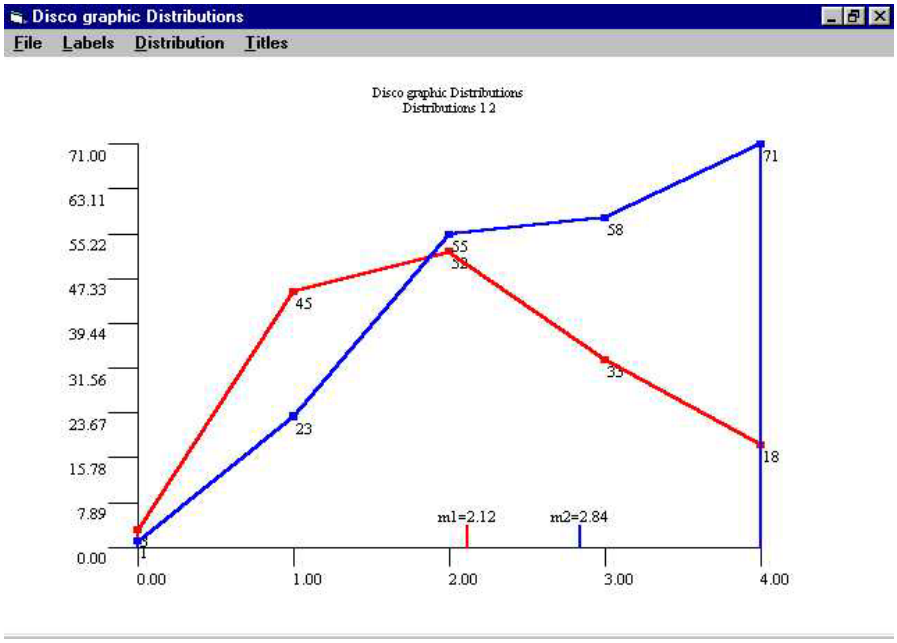


This is the window containing the Log file, and if the process is successful, the Output file.

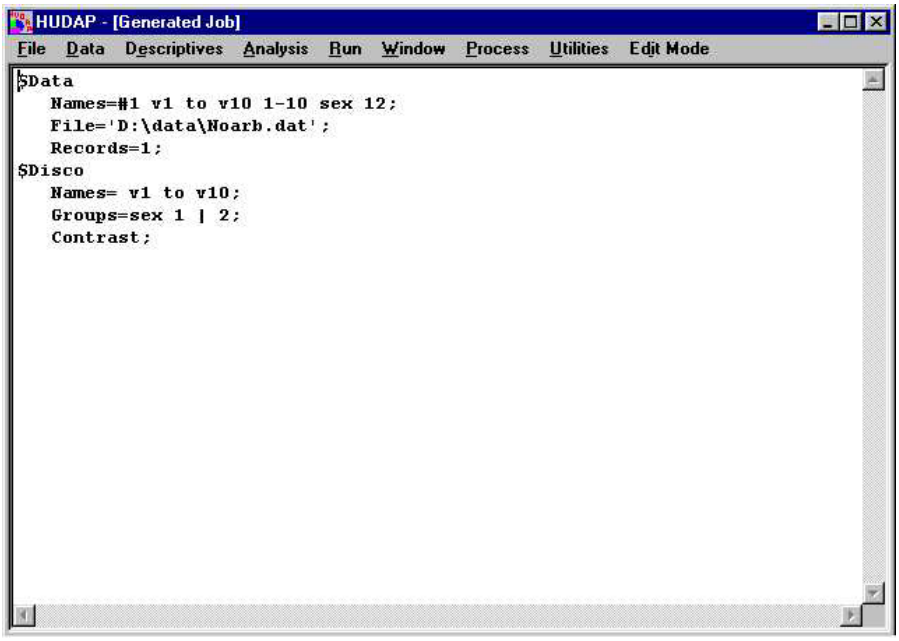
You can maximize the Output window and scroll it to see the results:



If the analysis produces a plot, **Graphics** menu appears on the Output window. Click on it and choose the specific graphic. In Disco, you may obtain, for example:



You can view the job generated by Hudap system for Disco analysis. In Output window, click on **Generated Job** from **Window** menu. The following window appears:



You cannot perform changes in this window. However, if you click on Edit mode menu, the job is written into the Editor window of Edit Mode. Here you can edit the job by means of Hudap language.

Index

P

R