Yadolah Dodge · Joe Whittaker
Editors

Computational Statistics

Volume 2

Proceedings of the
10th Symposium on
Computational Statistics

COMPSTAT
Neuchâtel, Switzerland, August 1992

With 97 Figures

Physica-Verlag
A Springer-Verlag Company

Editorial Interface in Statistical Computing and Related Areas

S. Mustonen

Department of Statistics, University of Helsinki, Aleksanterinkatu 7, 00100 Helsinki, Finland

Abstract

A general environment for statistical computing and other related areas is described. The functions within this framework are based on an editorial interface. It permits the statistician to control all stages of the work by a general text editor. Even statistical data sets can be written in the edit field which is a visible work sheet on the screen. For large data bases, special data files are provided. Information from other sources, like text files, is easily imported.

The ideas presented are realized in the Survo system. This system covers besides statistical computing and analysis various functions in general data management, graphics, spreadsheet computing, matrix operations, text processing, and desktop publishing. The newest extension to the editorial interface is a special macro language. This language is used for making various teaching and expert applications.

1. INTRODUCTION

The idea of statistical computing based on text editing emerged over 12 years ago from the conviction that statistical applications are not alone in this world. In modern computing, these applications have many connections to other activities.

When working with a computer, a statistician needs not only tools for statistical computations but many other services in order to carry out jobs encountered in a normal research process. It is valuable in that case if all the tasks could be performed within the same framework without a necessity to jump from one program and environment to another.

On the other hand, many non-professionals in statistics, pursuing mainly text processing and speadsheet computing, require statistical tools every now and then. For them, statistical services offered on the same platform would be more suitable than pure statistical packages.

In certain integrated environments, these diverse demands are now satisfied, at least to some extent. One is then able to control all phases of the job in a similar style. In our approach, implemented in practice as the Survo system, the editorial interface provides facilities for the above mentioned purposes.

This paper is not a comparative study between various forms of interactivity in statistical computing. Rather we try to present ideas behind the editorial approach mostly through examples and illustrations taken from the current version (3.34) of the SURVO 84C system running on standard PC's. Although descriptions on paper do not reveal every aspect of dynamic working methods, we hope that they will clarify the main concepts.

All the actions in Survo are based on text editing. The user types text and commands into an *edit field* which is an area in the memory and always partially visible on the screen. The text editor links the various parts of the system by calling program modules according to the user's activations. All the operations and commands are written in the edit field as well and the user activates them by the ESC key. Likewise, the results are printed into the edit field in places indicated by the user. Larger results are automatically saved in matrix and text files. Any result or text file can be loaded to the edit field and used as an input for new operations.

The editorial interface supports various representations of statistical data sets and bases. Small data sets can be written in the edit field as lists or tables. Larger data sets are saved in data files. Maximum number of variables in one data file can be several thousands and there are no limits for the number of observations except the general restrictions of the operating system for the file size. Many operations are provided for general data management, data input, editing, screening, etc. The variables can be transformed by (conditional) rules defined by the user.

In statistical operations, the variables can be selected in different ways. Also the scale types of the variables can be indicated in data files and the statistical operations will observe whether the scales of selected variables are valid in current analysis. Similarly, observations can be processed conditionally. All the computations are performed in double precision and the intermediate results are saved in matrix files for subsequent studies. A general matrix interpreter and various techniques (*touch mode* and *editorial computing*) related to spreadsheet computing are readily available for subtler analysis of the results.

An essential part of the editorial interface is an on-line help facility accessible as a hypertext. By using the tools of this help system, other hypertext applications can be produced on any application area as well.

As a collection of programs, Survo is open for additional modules made by experienced users. The rules and different tools for making modules in C are described by the author (1989). After a new module has been programmed and compiled, the commands and operations defined in it can be used as any standard Survo operation.

In many ways Survo exceeds the limits of typical statistical packages. One of the main goals is to give the user an opportunity to do most of the things belonging to a statistician's work with the same tool.

In fact, two different main approaches to working with the system can be seen. In the first one, the user creates a series of work schemes (setups in the edit field) to accomplish a typical chain of statistical analyses including graphics, simulation, etc. In the second model, the main target is to produce a multipage printed report with text, tables, and graphical illustrations (like this paper) on the basis of the current data more or less automatically. In more demanding expert applications, usually created as *sucros*, both working models are present.

A sucro is a recorded Survo session with conditional operations and prompts for the user. It is originally constructed by using Survo under the *tutorial mode* which enables saving of all user interventions in a selected file. The file can be edited later. This technique permits making of expert applications based on the existing operations. It also gives good possibilities for using the editorial interface in teaching. In fact, several sets of tutorials have been made for teaching various activities in the editorial interface, statistical methods, etc.

2. EDITORIAL COMPUTING

Arithmetic expressions and are typed in the edit field according to normal mathematical notation. For example, to calculate the arithmetic mean of numbers 12, 17, and 25 we enter:[1]

```
 22  1 SURVO 84C EDITOR Sat Jan 03 17:08:54 1987          D:\COMP\ 100 100 0
  1 *
  2 *
  3 *         (12+17+25)/3=_
  4 *
  5 *
```

Now, when the cursor is blinking immediately after =, we press the activation key [ESC]. Because there is no command on the current line, the editor studies the charater just before the cursor position. If it is = as in this case, the editor assumes that the user wants to calculate something and calls the editorial computing module. It analyzes the current expression, computes its value and writes the result in the edit field. Finally the control is transferred back to the editor and we may continue the work.

In this case, the following display is obtained:

```
 22  1 SURVO 84C EDITOR Sat Jan 03 17:08:54 1987          D:\COMP\ 100 100 0
  1 *
  2 *
  3 *         (12+17+25)/3=18
  4 *
  5 *
```

Since after each activation we are back in editorial mode, it is easy to round and edit the results, change the numbers, expressions and activate again.

When the same numbers are used for several computations or when more general expressions are wanted, we may also use symbolic notation and type

```
 31  1 SURVO 84C EDITOR Sat Jan 03 17:45:43 1987          D:\COMP\ 100 100 0
  1 *
  2 *         X=12 Y=17 Z=25
  3 * Arithmetic mean is (X+Y+Z)/3=_
  4 * Geometric mean is (X*Y*Z)^(1/3)=
  5 *
```

After activating both expressions we get the following display:

```
 34  1 SURVO 84C EDITOR Sat Jan 03 17:45:43 1987          D:\COMP\ 100 100 0
  1 *
  2 *         X=12 Y=17 Z=25
  3 * Arithmetic mean is (X+Y+Z)/3=18
  4 * Geometric mean is (X*Y*Z)^(1/3)=17.2130062073
  5 *
```

Various mathematical and statistical basic functions are readily available. More functions can be defined temporarily for the applications in the current edit field.

---

[1] Normally 23 or 48 consecutive lines of the edit field are shown in a window on the screen. To save space, the pertinent lines only are shown in these examples.

```
 17  1 SURVO 84C EDITOR Sun Mar 01 10:58:10 1992          D:\COMP\ 100 100 0
  1 *SAVE LOGIT
  2 *
  3 *Logit function is defined as a temporary function:
  4 *   logit(p):=log(p/(1-p)) .
  5 *
  6 *Probit function is readily available as a crude approximation
  7 *suitable e.g. for simulation. A more accurate alternative is
  8 *the inverse normal distribution function N.G(0,1,p) .
  9 *
 10 *Examples:
 11 *    p=0.95  u=rnd(0)
 12 *      logit(p).=
 13 *     probit(p).=
 14 *    N.G(0,1,p).=
 15 *      logit(u).=
 16 *     probit(u).=_
 17 *
```

In this example, the logit function is defined as a temporary function on line 4. Use of this and certain other related functions is tested by examples on lines 12-16. Since each of these expressions is tailed by .= (instead of mere =), activation of one of them implies an automatic multiple activation of the remaining ones, too. Two last expressions are dependent on u=rnd(0) which means a random deviate from the uniform distribution on (0,1). Thus when we activate this scheme (here from line 16), the display will be altered to:

```
 17  1 SURVO 84C EDITOR Sun Mar 01 11:06:08 1992          D:\COMP\ 100 100 0
  1 *SAVE LOGIT
  2 *
  3 *Logit function is defined as a temporary function:
  4 *   logit(p):=log(p/(1-p)) .
  5 *
  6 *Probit function is readily available as a crude approximation
  7 *suitable e.g. for simulation. A more accurate alternative is
  8 *the inverse normal distribution function N.G(0,1,p) .
  9 *
 10 *Examples:
 11 *    p=0.95  u=rnd(0)
 12 *      logit(p).=2.9444389791664
 13 *     probit(p).=1.6452114401438
 14 *    N.G(0,1,p).=1.6448536269515
 15 *      logit(u).=-0.14342985984947
 16 *     probit(u).=-0.08963034471472
 17 *
```

If this scheme is activated repeatedly, the values of the two last expressions will vary from one activation to another. The last one gives random deviates from the standard normal distribution.

Computation schemes

It is always up to the user how to organize the computations in the editorial interface. In many applications all the formulas and operations needed for reaching a specific goal can be expressed as a **computation scheme** which to some extent resembles a computer program. One clear distinction is, however, that in a computation scheme there is no specific order of statements.

Each activation in a work scheme leads always to a search process where the editor and other programs called for help are looking for the information needed for carrying out the task.

In fact, all the previous examples of editorial computing have been computation schemes in a modest sense. A true computation scheme, however, usually contains instructions and comments to help the user in applying the scheme.

For example, testing of a correlation coefficient by using Fisher's z transformation could be represented as a computation scheme as follows:

```
 33  1 SURVO 84C EDITOR Sun Jan 11 12:22:39 1987          D:\COMP\ 100 100 0
 47 *  Testing the correlation coefficient
 48 *  The sample correlation coefficient is r and the sample size n.
 49 *  To test the hypothesis that in the population the unknown
 50 *  correlation coefficient rho is r0 against the alternative rho>r0,
 51 *  we form the test statistic
 52 *         U=sqrt(n-3)*(Fisher(r)-Fisher(r0))
 53 *  where
 54 *         Fisher(r):=0.5*log((1+r)/(1-r))
 55 *  is Fisher's transformation of the correlation coefficient.
 56 *
 57 *  If the null hypothesis is true, U is approximately N(0,1).
 58 *  Hence we reject the hypothesis if P=1-N.F(0,1,U)
 59 *  is less than the risk level (say 0.05).
 60 *
 61 *  Assume now that n=25,  r=0.85 and r0=0.7
 62 *
 63 *  Then U.=1.8238788825   and P.=0.03408519244644
 64 *
 65 *  Thus if P<0.05, we reject the hypothesis that correlation
 66 *  coefficient in the population is r0.=0.7
 67 *.....................................................................
 68 *  Instructions: Insert your own values on line 61 and
 69 *                activate P.= on line 63.
```

Above we have moved the cursor to the line 63 to a place after = in P.= and pressed the ESC key. In order to get a value for P (and U because this is a multiple activation) the system has to find the definitions and constants given as **specifications** in the current subfield limited by dotted lines (like 67). The specifications are recognized by looking for strings containing '=' characters.

The free setting of commands, specifications and textual comments offers exciting possibilities for creative working both in research and teaching. It is easy to edit any detail in the computation scheme and activate again.

3. COMPARISON OF SAMPLES

A general tool for testing small samples is the COMPARE operation. It provides means for comparing independent samples and for pairwise comparisons. As in all statistical operations, the data set can be written in the edit field as a list or a table of values or it can be saved in a data file and called by its name when used in any statistical operation.

Assume now that we have written in the edit field: (In this and forthcoming examples, the activated commands are highlighted just for illustration.)

```
 21  1 SURVO 84C EDITOR Sat Feb 29 12:50:56 1992          D:\COMP\ 100 100 0
  1 *SAVE BOYS
  2 *
  3 *Example from
  4 *Conover: Practical Nonparametric Statistics, 2nd Edition, p.218
  5 *
  6 *The senior class in a particular high school had 48 boys. Twelve boys
  7 *lived on farms and the other 36 lived in town. A test was devised to see
  8 *if farm boys in general were more physically fit than the town boys.
  9 *Each boy in the class was given a physical fitness test in which a low
 10 *score indicates poor physical condition. The scores of the farm boys
 11 *and the town boys are as follows.
 12 *
 13 *DATA FARM: 14.8 7.3 5.6 6.3 9.0 4.2 10.6 12.5 12.9 16.1 11.4 2.7 END
 14 *DATA TOWN: 12.7 14.2 12.6 2.1 17.7 11.8 16.9 7.9 16.0 10.6 5.6 5.6
 15 *           7.6 11.3 8.3 6.7 3.6 1.0 2.4 6.4 9.1 6.7 18.6 3.2
 16 *           6.2 6.1 15.3 10.6 1.8 5.9 9.9 10.6 14.8 5.0 2.6 4.0 END
 17 *
 18 *COMPARE FARM,TOWN,19
 19 *
```

In this display, two samples with some explanatory text are typed. The whole setup can be saved as an edit file BOYS by activating the SAVE command (typed on the first line).

On line 18 appears a COMPARE command referring to samples FARM and TOWN. The last parameter (19) tells the first line intended for the results of the operation.

If now the COMPARE command is activated (by moving the cursor to line 18 and by pressing the ESC key), the editor calls another program, the COMPARE module of Survo, to carry out comparisons between samples. The COMPARE module is called as a child process of the editor program. It means that the editor stays resident in the memory and COMPARE is loaded to the remaining part of the memory.

COMPARE starts the job by computing certain basic statistics and keeps a temporary display of them in the form:

```
 21  1 SURVO 84C EDITOR Sat Feb 29 12:51:11 1992          D:\COMP\ 100 100 0

Independent samples            FARM                 TOWN
Sample size                    12                   36
Mean                            9.450000             8.650000
Standard deviation              4.283902             4.914265
Student's t=0.503 df=46 (P=0.6913 one-sided test)
Sum of ranks (R)               321                  855
Mann-Whitney (U)               243                  189
(P=0.7398 one-sided Mann-Whitney, normal approximation)
Critical levels by simulation:
               Mean    R or U
Critical level 0.69737 0.74124  N=13700
Standard error 0.00392 0.00374

To interrupt simulation, press any key!
```

By default, it is assumed that samples to be compared are independent and we want to test the hypothesis that they have identical distribution functions against shift alternatives. Besides the standard statistics, the COMPARE module tries also to calculate true critical levels of

Mann-Whitney test based on ranks and of a corresponding test based on the sample means by the Fisher's randomization principle. In both cases, the joint sample of 12+36=48 observations is split randomly into two subsamples of 12 and 36 observations. For these subsamples, the rank sums (Mann-Whitney) and sums of original scores are computed and compared with corresponding sums in the true FARM and TOWN samples. The randomization process will be repeated indefinitely and relative frequencies for replicates where the sums exceed the true sums are counted. The counts are updated on the screen after each 100th replicate. The relative frequencies thus formed will tend to critical values of one-tailed tests. To see the accuracy of the critical level estimates, their standard errors are displayed, too.

The user may limit the number of replicates by a SIMUMAX specification written anywhere in the edit field. Default is SIMUMAX=100000. Below, the results after this automatic interrupt are displayed. The COMPARE module writes the results from the line 19 onwards and terminates. Then we are back in the editorial mode again and by scrolling the visible part of the field we can see:

```
 21  1 SURVO 84C EDITOR Sat Feb 29 12:52:57 1992          D:\COMP\ 100 100 0
 17 *
 18 *COMPARE FARM,TOWN,19_
 19 *Independent samples          FARM                TOWN
 20 *Sample size                  12                  36
 21 *Mean                          9.450000            8.650000
 22 *Standard deviation            4.283902            4.914265
 23 *Student's t=0.503 df=46 (P=0.6913 one-sided test)
 24 *Sum of ranks (R)             321                 855
 25 *Mann-Whitney (U)             243                 189
 26 *(P=0.7398 one-sided Mann-Whitney, normal approximation)
 27 *Critical levels by simulation:
 28 *              Mean    R or U
 29 *Critical level 0.69690 0.74115  N=100000
 30 *Standard error 0.00145 0.00139
 31 *
```

Because the operations (commands) are written into the text like any other information, it is easy to edit them and activate again. Thus the commands do not disappear from the the screen (edit field) after they have been completed. The user handles commands and their results like the text of his own.

4. FITTING A UNIVARIATE DISTRIBUTION

We shall study a mixture of two normal distribution of the form

$$p*\mathrm{N}(m_1, s_1^2) + (1-p)*\mathrm{N}(m_2, s_2^2) .$$

We generate 10000 observations with parameters p=0.7, m_1=0, s_1=1, m_2=2, s_2=0.5, and try to re-estimate them from starting values INIT=0.5,-1,1.5,3,1.

A file SIMUDATA is first created for 10000 observations of variable X and values from the mixture are then computed by the VAR scheme on lines 8-11.

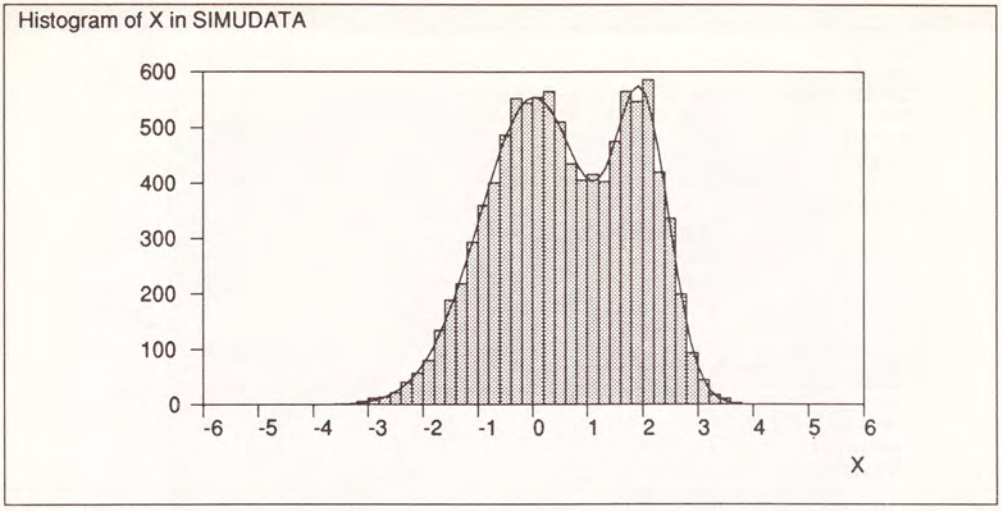
To estimate the parameters from the sample, a frequency distribution of X is formed by a HISTO operation on line 17. The FIT=MIXNORM specification on line 19 implies HISTO to fit the MIXNORM distribution defined on lines 14-16 to SIMUDATA. The results of estimation are displayed on lines 22-35.

```
 20  1 SURVO 84C EDITOR Fri Mar 01 16:21:58 1992          D:\COMP\ 100 100 0
  1 *
  2 *FILE CREATE SIMUDATA,4,1,64,7,10000
  3 *  Sample (N=10000) from a mixture of two normal distributions
  4 *FIELDS:
  5 *1 N 4 X
  6 *END
  7 *
  8 *VAR X TO SIMUDATA
  9 *  X=if(rnd(1)<0.7)then(X1)else(X2)
 10 *  X1=probit(rnd(1))
 11 *  X2=0.5*probit(rnd(1))+2
 12 *.....................................................................
 13 *
 14 *DENSITY MIXNORM(p,m1,s1,m2,s2)
 15 *y(x)=c*(p/s1*exp(-0.5*((x-m1)/s1)^2)+(1-p)/s2*exp(-0.5*((x-m2)/s2)^2))
 16 *     c=0.39894226
 17 *HISTO SIMUDATA,X,22
 18 *X=-6(0.2)6 XSCALE=-6(1)6 YSCALE=0(100)600
 19 *FIT=MIXNORM INIT=0.5,-1,1.5,3,1
 20 *SIZE=1600,800 DEVICE=PS
 21 *.....................................................................
 22 *HISTO: Estimated parameters of MIXNORM:
 23 *p=0.7003 (0.0125)
 24 *m1=0.0301 (0.0296)
 25 *s1=1.0077 (0.0193)
 26 *m2=2.0095 (0.0180)
 27 *s2=0.4906 (0.0137)
 28 *logL=16043.335886  # of function evaluations =311
 29 *Correlations:
 30 *                p      m1      s1      m2      s2
 31 * p          1.000   0.845   0.796   0.728  -0.707
 32 * m1         0.845   1.000   0.802   0.673  -0.661
 33 * s1         0.796   0.802   1.000   0.572  -0.591
 34 * m2         0.728   0.673   0.572   1.000  -0.700
 35 * s2        -0.707  -0.661  -0.591  -0.700   1.000
```

HISTO also lists the frequency distribution and various related statistics as follows

```
 20  1 SURVO 84C EDITOR Fri Mar 01 16:22:10 1992          D:\COMP\ 100 100 0
 36 *Frequency distribution of X in SIMUDATA: N=10000
 37 *
 38 *Class midpoint  f     %   Sum     %     e      e     f     X^2
 39 *      <=-3.6    0   0.0     0   0.0   1.1
 40 *        -3.5    2   0.0     2   0.0   1.2
 41 *        -3.3    1   0.0     3   0.0   2.4
 42 *        -3.1    6   0.1     9   0.1   4.5    9.3    9     0.0
 43 *        -2.9   12   0.1    21   0.2   8.2    8.2   12     1.7
 ------------
 74 *         3.3   18   0.2  9984  99.8  19.2   19.2   18     0.1
 75 *         3.5   11   0.1  9995 100.0   6.8
 76 *         3.7    4   0.0  9999 100.0   2.2
 77 *         3.9    1   0.0 10000 100.0   0.7
 78 *       > 4.0    0   0.0 10000 100.0   0.4   10.0   16     3.6
 79 *Mean=0.623320 Std.dev.=1.267161
 80 *Fitted by MIXNORM(0.7003,0.0301,1.0077,2.0095,0.4906) distribution
 81 *Chi-square=27.48 df=28 P=0.4921
```

and plots the graph:



5. INFLUENCE CURVES FOR THE CORRELATION COEFFICIENT

The following work scheme is intended for plotting a scatter diagram with appropriate contour curves describing the robustness of the correlation coefficient. Actually these influence curves will appear as contours of a raster image of the influence function.

The final graph is produced by a series of 4 different Survo operations (CORR, PLOT scatter diagram, PLOT contours, and PRINT) given in the next display.

The CORR DECA,4 command computes the means, standard deviations, and correlations of the active variables of a data set DECA (on disk) and prints the results from the line 4 onwards. Thus, in this case, lines 4-11 are output from the CORR operation. The set of active variables has been limited to Height and Weight by the specification VARS.

The user has copied the basic statistics obtained by CORR in an abbreviated form to line 13. The PLOT scheme needed for making a contour plot of the influence surface is located on lines 13-23. The actual PLOT command on line 15 plots a function z(x,y) of two variables x,y as a contour plot (specified by TYPE=CONTOUR on line 19). The expression defining z(x,y) depends on the current value of the correlation coefficient r and on three auxiliary functions u,v, and w which in turn depend on parameters n,mx,my,sx, and sy. These functions are defined as specifications on lines 16-18.

Our function z(x,y) gives the change in the value of correlation coefficient r when a new observation x,y is obtained.

When making the raster image, the values of the function z are mapped continuously to various shades of gray in such a way that 0 corresponds to 'black' and 1 corresponds to 'white'.

If the function value exceeds 1, the shading is selected 'modulo 1'. In this case, the original function values are multiplied by 20 (by ZSCALING=20,0) which gives a complete cycle of shadings when the function value changes by 1/20 = 0.05 . Thus, the final graph will depict contours of r with increments of 0.05 . The SCREEN=NEG specification (on line 20) simply reverses the shadings.

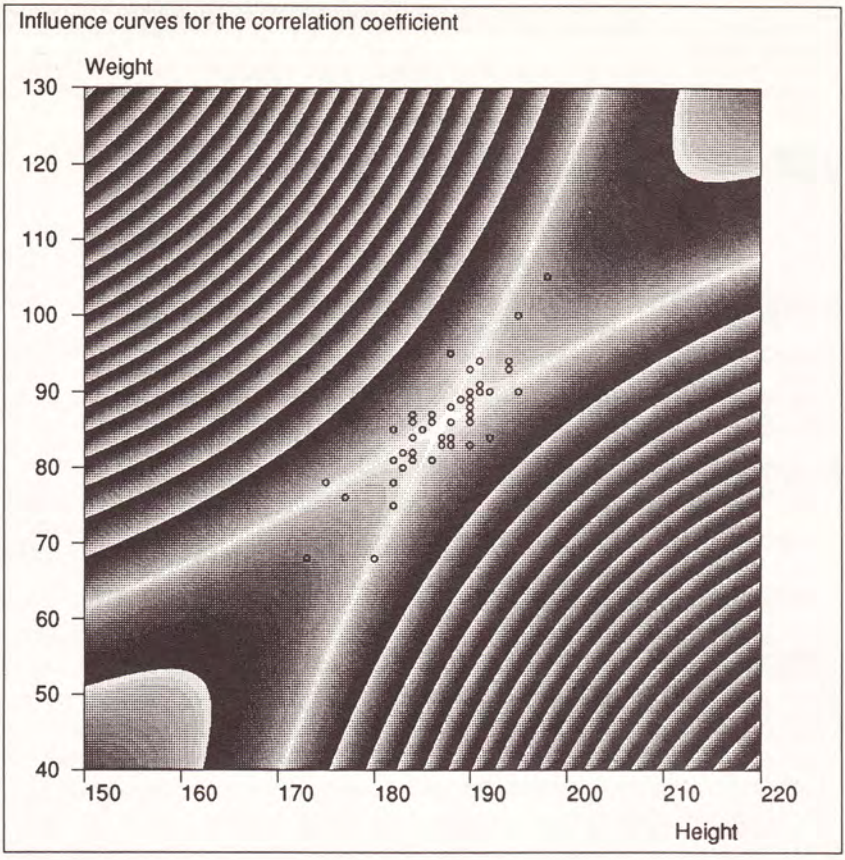
```
 12  1 SURVO 84C EDITOR Tue Mar 03 13:40:25 1992          D:\COMP\ 100 100 0
  1 *SAVE INF
  2 *
  3 *CORR DECA,4 / VARS=Height,Weight
  4 *Means, std.devs and correlations of DECA  N=48
  5 *Variable  Mean        Std.dev.
  6 *Height    186.9583    5.090493
  7 *Weight    85.56250    6.847600
  8 *Correlations:
  9 *             Height  Weight
 10 * Height       1.0000  0.8522
 11 * Weight       0.8522  1.0000
 12 *.....................................................................
 13 *r=0.85 mx=186.96 my=85.56 sx=5.09 sy=6.85 n=48
 14 *HEADER=Influence curves for the correlation coefficient
 15 *PLOT z(x,y)=abs(r*(1-w)+u*v)/w
 16 *  u=sqrt(n/(n*n-1))*(x-mx)/sx
 17 *  v=sqrt(n/(n*n-1))*(y-my)/sy
 18 *  w=sqrt((1+u*u)*(1+v*v))
 19 *TYPE=CONTOUR  ZSCALING=20,0      (1/0.05=20)
 20 *             SCREEN=NEG
 21 *XSCALE=150(10)220 YSCALE=40(10)130 SIZE=1350,1350
 22 *x=150,220,0.1   y=40,130,0.1
 23 *DEVICE=PS,INF.PS
 24 *.....................................................................
 25 *PLOT DECA,Height,Weight
 26 *XSCALE=150(10)220 YSCALE=40(10)130 SIZE=1350,1350
 27 *HEADER=
 28 *DEVICE=PS,DECA.PS
 29 *.....................................................................
 30 *PRINT 31,33
 31 % 1650
 32 - picture INF.PS,*,*
 33 - picture DECA.PS,*,*
 34 *
```

Some plotting parameters, regulating ranges of variables and the size of the graph are given on lines 21-22.

`DEVICE=PS,INF.PS` (on line 23) implies the graph to be produced as a PostScript picture and saved in file `INF.PS` . The simpler `PLOT` scheme on lines 25-29 makes a scatter plot of variables `Height` and `Weight` in the data set `DECA` using the same plotting specifications and saves the picture as a PostScript file `DECA.PS` .

Finally, the `PRINT` operation processes the lines 31-33 and produces the next graph. From the graph we can see that, for example, a new observation x=190, y=40 would decrease the original r from 0.85 by 6×0.05 to 0.55 .

It should be noted that the setup above, although written for a particular case, gives a general basis for plotting of corresponding contour plots for any other data set as well. By loading the `INF` edit field, the user can modify the schemes and reactivate them. By supplying explanations within the schemes, the setup is easily made accessible for users unfamiliar with the technical details. The whole task can also be formulated as an automatic sucro with prompts for the user etc. on the basis of this scheme. The user is able to apply that sucro as a new operation of the system.



6. CLUSTER ANALYSIS

We demonstrate the behaviour of a certain clustering technique in a simulated heterogeneous data set of two bivariate normal samples.

In this experiment, a file N2 of 100 observations is created by FILE CREATE (on lines 6-10) and two samples from a bivariate normal distribution with different means are generated by a VAR operation (on lines 12-17).

Three more variables (G1,G2,G3) are created by another VAR on lines 19-20 to store three different groupings. These variables are initialised by 0's.

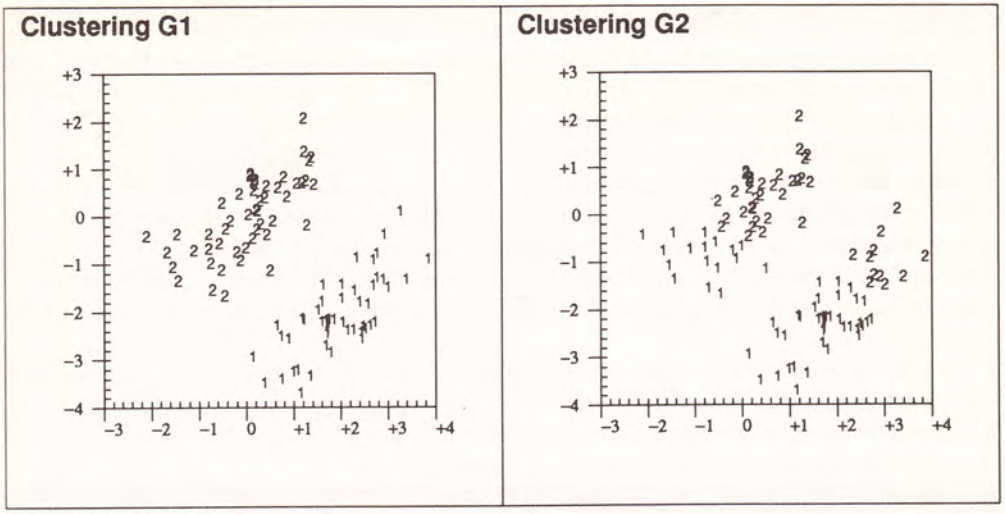
The cluster analysis is performed by the CLUSTER operation on lines 26-27 and giving the results on lines 28-34. The variables are selected and their tasks in the analysis are declared by the MASK=AAGGG specification (on line 26). It is an abbreviated form of the VARS specification and determines the roles of variables in the order they appear in the data set. Here 'A' means a variable to be analyzed and 'G' a variable to be used for storing of a clustering.

```
 13  1 SURVO 84C EDITOR Tue Mar 03 13:47:37 1989          D:\COMP\ 100 100 0
  1 *SAVE CLUSTER
  2 *  Two samples from a bivariate normal distribution
  3 *  with different means
  4 *  but the same covariance matrix are generated:
  5 *.....................................................................
  6 *FILE CREATE N2,32,10,64,7,100
  7 *FIELDS:
  8 *1 N 4 X
  9 *2 N 4 Y
 10 *END
 11 *
 12 *VAR X,Y TO N2
 13 *X=if(ORDER<51)then(X1)else(X2)
 14 *Y=if(ORDER<51)then(Y1)else(Y2)
 15 *X1=Z1      Y1=r*Z1+s*Z2       r=0.8 s=sqrt(1-r*r)
 16 *X2=Z1+2    Y2=r*Z1+s*Z2-2
 17 *Z1=probit(rnd(2)) Z2=probit(rnd(2))
 18 *.....................................................................
 19 *VAR G1:1,G2:1,G3:1 TO N2
 20 *  G1=0 G2=0 G3=0
 21 *.....................................................................
 22 *  The CLUSTER operation with 10 trials, 2 groups,
 23 *  and random number generator 2
 24 *  gives two different solutions:
 25 *
 26 *MASK=AAGGG    TRIALS=10  GROUPS=2 SEED=2
 27 *CLUSTER N2,28
 28 *Stepwise cluster analysis by Wilk's Lambda criterion
 29 *Data N2  N=100
 30 *Variables: X, Y
 31 *Best clusterings found in 10 trials are saved as follows:
 32 * Lambda     freq  Grouping var
 33 * 0.04496      6   G1
 34 * 0.14945      4   G2
 35 *
 36 *.....................................................................
 37 *The result can be checked by plotting the graph:
 38 *GPLOT N2,X,Y
 39 *HEADER=Clustering_G1
 40 *POINT=G1  (G2 gives the inferior clustering)
 41 *
```

The analysis is based on the Wilks' lambda criterion and an efficient stepwise procedure developed by Korhonen (1979) is applied. In this procedure, first a random partition of observations into 2 groups (GROUPS=2 on line 26) is selected. Thereafter the procedure tries to move observations from a group to another and if the criterion value is improved, the observation is really moved. This practice will be continued until no single move improves the criterion value.

Since the result depends on the initial random partition, it is good to repeat the procedure several times. In this case, 10 different random starts have been taken (TRIALS=10) and two different solutions (G1,G2) are obtained as seen on lines 33-34.

The first solution G1 having the better criterion value is, of course, the true one. The characteristics of the solutions are clearly revealed by the plots generated by the PLOT scheme on lines 38-40:



7. SUCROS

The editorial interface permits the user's actions to be saved in a selected file when working in the tutorial mode. The file can be edited later and run as a teaching program or as a user-defined task. We use the name sucro (Survo macro) for such applications. Although the sucros are most easily generated in the tutorial mode, a special sucro language has been designed for their maintenance. This language covers all the functions of Survo. It also includes statements for conditional working, user interface, and time control.

The range of sucros is wide. The small ones are like ordinary macros, but the largest sucros are general object oriented programs performing Survo operations conditionally and writing complete reports with text, tables and graphical illustrations automatically from the data at hand. By combining various Survo activities with conditional statements of the sucro language, extensive expert applications can be created.

In all categories of problems, sucros lessen the need for making actual computer programs. They are also more compact than programs. A typical sucro file for a sequence of statistical operations requires only a few kilobytes while a corresponding C program might take about 100 kilobytes.

To a great extent, the sucros are made simply by turning on the tutorial mode and starting to use Survo for solving the current problem. Then all the keystrokes will be saved in a file selected by the user. After returning to the editorial mode, the file will be closed and it is readily available for an automated repetition of the task.

When making a new sucro in the tutorial mode, one can type text and activate both Survo operations and other sucros. The possibility of using other sucros as building blocks makes this technique very powerful.

Any sucro file is edited later in Survo by TUTLOAD and TUTSAVE commands. The TUTLOAD command loads the compressed sucro file into the edit field, interprets it, and forms a readable list which could be called a sucro program. The user can edit the program, insert conditional statements and material from other sucros. After editing, the program is

compressed and saved back in the file by the TUTSAVE command.

As an example, we consider a simplified version of a sucro FACTOR for factor analysis. This sucro combines the normal steps of factor analysis. The crucial point is that the sucro selects the number of factors automatically according to the sizes of the eigenvalues of the correlation matrix. In this version we simply use value 1 as a cut off point.

Before describing the sucro program, we give an application of FACTOR for a data set DECA of 48 athletes in Decathlon. In the next display, the MASK specification (on line 2) selects the scores in 10 events of Decathlon as active variables and the /FACTOR DECA command calls the FACTOR sucro with the parameter DECA. The slash '/' before the command indicates that it is a sucro call. Everything below line 3 will be output produced by the sucro.

At first the correlations of the active variables for the active observations are computed and saved as a matrix file CORR.M (by CORR operation). Then the eigenvalues (and vectors) are computed as the spectral decomposition of CORR.M . The vector of eigenvalues D is loaded into the edit field and we have the following display:

```
 17  1 SURVO 84C EDITOR Wed Mar 04 13:42:42 1992          D:\COMP\ 120 100 0
  1 *
  2 *MASK=---AAAAAAAAAA---
  3 */FACTOR DECA
  4 *CORR DECA
  5 *MAT SPECTRAL DECOMPOSITION OF CORR.M TO S,D
  6 *MAT LOAD D,CUR+1_
  7 *MATRIX D
  8 *L(R(DECA))
  9 *///       eigenval
 10 *ev1       2.602056
 11 *ev2       2.007813
 12 *ev3       1.206620
 13 *ev4       1.067052
 14 *ev5       0.931538
 15 *ev6       0.594690
 16 *ev7       0.569080
 17 *ev8       0.538683
 18 *ev9       0.244855
 19 *ev10      0.237613
 20 *
```

The sucro FACTOR continues from this situation by 'reading' the series of eigenvalues (on lines 10-19) in order to find out a suitable cut off point. The sucros are emulating all typical functions of a normal user. However, sucros are 'blind'; they cannot 'see' the text in the edit field directly but they have to find information by seeking critical words and numbers by various search and conditional statements of the sucro language.

In this case the search will stop on line 14 (first eigenvalue less than 1). The simple decision rule sets then the number of factors to 4 and the sucro continues its work. It erases lines from 5 onwards, computes the maximum likelihood solution with 4 factors (by FACTA) and makes finally a Varimax rotation from the factor matrix FACT.M (by ROTATE). At the end of the sucro, we obtain the display:

```
 22  1 SURVO 84C EDITOR Wed Mar 04 13:42:52 1992          D:\COMP\ 120 100 0
  1 *
  2 *MASK=--AAAAAAAAAA---
  3 */FACTOR DECA
  4 *CORR DECA
  5 *FACTA CORR.M,4
  6 *ROTATE FACT.M,4,CUR+1
  7 *Rotated factor matrix
  8 *              F1     F2     F3     F4
  9 *100m       0.883  0.277  0.314 -0.036
 10 *L_jump    -0.084  0.030  0.752 -0.061
 11 *Shot_put  -0.116  0.391 -0.019  0.735
 12 *Hi_jump   -0.494  0.146 -0.037  0.095
 13 *400m       0.545 -0.401  0.262 -0.098
 14 *Hurdles    0.183  0.120  0.400  0.045
 15 *Discus    -0.140  0.532  0.043  0.684
 16 *Pole_vlt   0.037  0.041  0.003 -0.297
 17 *Javelin   -0.303  0.006  0.147  0.062
 18 *1500m      0.023 -0.854 -0.258 -0.155
 19 *
 20 *
```

The sucro program, listed by a TUTLOAD command, is following:

```
 15  1 SURVO 84C EDITOR Wed Mar 04 14:03:07 1992          D:\COMP\ 120 100 0
 20 *
 21 *TUTLOAD FACTOR
 22 / /FACTOR <data>
 23 / makes factor analysis from active variables and observations of <data>.
 24 / Number of factors is selected automatically by examining eigenvalues
 25 / of the correlation matrix CORR.M .
 26 /
 27 / def Wdata=W1        Name of the data set
 28 / def Weigenvalue=W2  Current eigenvalue
 29 / def Wfactor=W3      Number of factors
 30 *{init}{R}
 31 *SCRATCH {act}{home}CORR {print Wdata}{act}{R}
 32 *MAT SPECTRAL DECOMPOSITION OF CORR.M TO S,D{act}{R}
 33 *{ref}MAT LOAD D,CUR+1{act}{R}
 34 *{d2}{Wfactor=-1}
 35 + Next_line: {R}
 36 *{Wfactor=Wfactor+1}{next word}{save word Weigenvalue}
 37 - if Weigenvalue >= 1 then goto Next_line
 38 *{ref}{ref}{u}SCRATCH {act}{home}FACTA CORR.M,{print Wfactor}{act}{R}
 39 *ROTATE FACT.M,{print Wfactor},CUR+1{act}{R}
 40 *{end}
 41 *
```

The first lines (with a '/' in the control column of the edit field) are comments. However, three last (27-29) of them also give proper names for locations of the sucro memory. This memory is available for control operations of the sucro. The parameters of the sucro command are stored as the first members of this memory. When the sucro is running, it can read more numbers and words from the edit field into the memory.

Plain text in the program (on lines with a '*' in the control column) is reproduced as such by the sucro, i.e. it is typed into the edit field like the text written by the user normally. Various control statements and keys appear in braces. For example, {R} is the [ENTER] key, {act} is the activation key, {d2} means pressing of the down arrow twice, and {print Wdata}

means typing of the contents of the memory location `Wdata`.

The intrinsic part of the program is the loop (lines 35-37) where the list of eigenvalues (appearing on lines 10-19 of the first display) are scanned. The statement `{Wfactor= Wfactor+1}` increases the count, `{next word}` moves the cursor (from the beginning of the current line) to the next word, and `{save word Weigenvalue}` saves the word touched by the cursor to the memory location `Weigenvalue`. On line 37 is a test whether the current eigenvalue is at least 1. `Next_line` (referred to by `goto` on line 37) is not the next line in the listing or in the edit field; it refers to the corresponding label + `Next_line` on line 35.

The `{ref}` code (corresponding to a reference key) in the first place sets a reference point, secondarily moves the cursor back to the the reference point, and thirdly releases the current reference point. This code is used for relocating the point (line 5 in our example) wherefrom the lines are erased after the preliminary examination of eigenvalues.

Although this tiny example on sucros does not uncover all essential features of sucro programming, one should observe its general nature where bigger blocks (like Survo operations) are glued together by text editing elements. In that process, the 'cursor choreography' plays a fundamental role.

Background of the system

Many of the ideas and principles appearing in Survo have been adopted from the earlier versions. The first in line was SURVO 66 originated by the author in 1966 and implemented on Elliott 803. One explanation for the name SURVO is the word "survey" since the first Survo was primarily planned for analysis of survey data. It can also be derived from the Finnish verb *"survoa"* which means *"compress"*. The SURVO 66 jobs were controlled by a simple command language. The original SURVO 66 was further developed at the University of Tampere and is now known by the name of SURVO/71.

In 1976 the first interactive version SURVO 76 was initiated by the author. It was completed in 1984 by him and his research group in the University of Helsinki. Originally SURVO 76 was made in conversational (menu-based) form. The editorial approach was introduced in 1979. SURVO 76 was implemented on the Wang 2200 minicomputer.

The work on SURVO 84 started in 1984 on the basis of SURVO 76 by using the interpretative Basic language. This was the the first microcomputer version and could be run on the Wang PC only.

The current SURVO 84C system was originated in 1985. From the user's viewpoint it is much like SURVO 84 and it is also highly compatible with SURVO 76. But the latest version is far more efficient and it allows wider applications since it is programmed in the C language. SURVO 84C can be used on MS-DOS microcomputers. From 1990 on, a reduced version SURVOS has been available. It is intended mainly for teaching applications in secondary and vocational schools.

References

P.Korhonen (1979). *A stepwise procedure for multivariate clustering*, Research Reports No.7, Computing Centre, University of Helsinki (77 pp.)

S.Mustonen (1982). *Statistical computing based on text editing*, Proceedings in Computational Statistics, 353-358, Physica-Verlag, Wien

S.Mustonen (1987). *SURVO 84C User's Guide*, Dept. of Statistics, University of Helsinki (335 pp.)

S.Mustonen (1988). *PostScript Printing in SURVO 84C*, SURVO 84C Contributions 1, Dept. of Statistics, University of Helsinki (34 pp.)

S.Mustonen (1988). *Sucros in SURVO 84C*, SURVO 84C Contributions 2, Dept. of Statistics, University of Helsinki (28 pp.)

S.Mustonen (1989). *Programming SURVO 84 in C*, SURVO 84C Contributions 3, Dept. of Statistics, University of Helsinki (80 pp.)

This paper has been produced by the PRINT operation of Survo on a PostScript laser printer.