FORTRAN IV program for relational regression using the gradient method*

W. K. GRASSMANN and B. G. ARCHIBALD

*University of Saskatchewan
Saskatoon, Saskatchewan, Canada*

Often one wishes to find which factors determine selections or ranking orders of certain objects. This is what this program does. The objects are assumed to be selected or ranked according to their performance or ability. The performance of the j^{th} element, g_j is a linear function of certain known independent variables, x_{ij} , and an error term, u_j . Thus, one has

$$g_j = \alpha_1 x_{j1} + \alpha_2 x_{j2} + \cdots + \alpha_m x_{jm} + u_j$$

The program estimates the α_i and g_j , subject to the restriction that the α_i sum up to one. The estimation is done by linear programming in such a way that the predicted rankings or selections coincide as close as possible with the observed ones. The large number of restrictions (400 and above) required a special linear programming algorithm, namely the gradient method.

Input. Card 1 contains an alphanumeric problem description. Card 2 contains a sentinel for selection or ranking-order application, then values for the number of cases, the number of variables, an epsilon, and an iteration counter. Card 3 contains descriptions of the variables, giving each variable a name. Card 4 contains values for the number of observations for each case, plus the number selected per case. Of course, the number selected per case is a dummy value in a ranking-order application. Succeeding cards contain an identifier for each observation plus values for the independent variables (x_{ij}).

Output. The output consists of a problem description, the values of the α_i parameters, and a table containing the x_{ij} and comparisons of the actual and predicted performances.

Computer and Language. The computer used in running the program was an IBM 370-155. The programming language used was FORTRAN IV. The execution times never exceeded a few seconds.

Restrictions. The program handles a maximum of 4 variables and 400 comparisons. The core requirement is 56K for the aforementioned maximum.

Availability. A source deck, program listing, and instructions for the program's use are available free of charge from W. K. Grassmann, Department of Computational Science, University of Saskatchewan, Saskatoon, Saskatchewan, Canada S7n 0W0.

REFERENCES

W. K. Grassman. Relational regression. *Journal of Business Administration*, in press.

*The development of this program was made possible by the Youth Employment Program of the Province of Saskatchewan.

Two special-purpose programs that perform nonmetric multidimensional scaling*

FORREST W. YOUNG and JAY R. LEVINSOHN

*Psychometric Laboratory, University of North Carolina
Chapel Hill, North Carolina 27514*

SCALAR is a multidimensional scaling (MDS) program that performs the most common type of analysis on the most common type of data: a nonmetric MDS in Euclidean space of a symmetric matrix of dissimilarities that has no diagonal and no missing data. Either the Guttman-Lingoes (1968) rank image transformation, the Kruskal (1964) least squares transformation, or the Young (1974) weighted average transformation may be chosen.

SCALAR is restricted in the above ways so that an algorithm could be written which is: (a) very easy to use, and (b) efficient in storage requirements. The program is most suitable for the user who wishes to perform an MDS of an average number of stimuli (25) on a small machine or of a large number of stimuli (250) on a large machine.

Software. SCALAR has been written in two versions: SCALAR-1130 was designed to run on an IBM 1130 (8K 1 disk drive) and is coded in FORTRAN as implemented on the IBM 1130. SCALAR-360 was designed to run on the IBM 360/370 series, is coded in FORTRAN IV, and compiles and executes under both the G and H levels of the IBM FORTRAN. The major difference between the two versions of the program is the size of the problem they handle. SCALAR-1130 will accept a problem of up to 25 points to be scaled in up to five dimensions; SCALAR-360 will handle a variable number of points and dimensions, constrained by the amount of computer memory available to the user.

SCALAR-1130 is organized as one main core-resident control program that orders the execution of 16 subroutines. Most of these subroutines are called into core only when needed, at other times they reside in a disk swap area. The swapping of subroutines is handled by the 1130 disk operating system's Load-on-Call (LOCAL) option. As well as swapping subroutines, SCALAR-1130 uses disk storage to hold data for temporary storage. This method allows one to maximize the use of memory by keeping all intermediate results on disk. SCALAR-1130 will process a problem of 7 points at a rate of about 12 sec per iteration, a problem of 14 points requires about 70 sec per iteration, and the maximum problem of 25 points in 5 dimensions requires 2 min per iteration. There are two concerns specific to the 1130 version that the user should be aware of:

*This research was supported, in part, by PHS research grant MH-10006 from the National Institute of Mental Health, Public Health Service, and by National Science Foundation, Science Development Grant GU-2059.

(1) data format, and (2) core requirements. Since the 1130 FORTRAN compiler has no object time format option, the user must input his data in a standard format supplied with the program or change the format of the input statement in the program. As it is written SCALAR-1130 will run on the 360/370 series with no modifications. This could be useful to some users with rather small 360/370 systems.

Computers. SCALAR-360 was designed to run on the 360/370 series computers. In general it is quite similar to the 1130 version but it has few alterations that make it more suitable to the 360/370 and more amenable to large problems. This version has all its subroutines in core and does no disk swapping of program code. Like the 1130 version, it does store data and intermediate results on disk to minimize the required core. SCALAR-360 has an object time format option and was designed so that by modifying a few specification statements in the main routine one can set the program to run as large a problem as one's machine can hold.

Algorithm. A nongradient algorithm is used which minimizes a normalized sum of squared deviations between the monotonically transformed quasiscalar products (obtained from the data matrix) and the scalar products of the Euclidean space. This algorithm is guaranteed to converge (perhaps slowly) on the global minimum of the scalar products stress function. The computational steps are outlined below: (a) The data matrix S is converted to the quasiscalar products matrix B^* using the procedure defined by Torgerson (1958). (b) The matrix B^o of integers from 1 through $[N(N+1)/2]$, where n = the number of points, which is monotonic with B^* , is obtained. This matrix is the matrix of ranking numbers representing the ordinal information contained in B^* . It is this ordinal information which we will attempt to reproduce. (c) The matrix X of scale values is obtained from B^* by

weighting the NDIM eigenvectors of B^* which correspond to the largest eigenroots by the square root of the roots. This is a matrix of NDIM-dimensional Euclidean scale values. X is normalized. (d) The matrix of scalar products $B = XX'$ is obtained. This matrix is the (linear) least squares rank NDIM approximation to B^* . (e) The matrix of monotonically transformed quasiscalar products, B^m , is obtained from B^o and B using Young's (1974) weighted average transformation technique. (f) The goodness-of-fit between B and B^m is determined using Kruskal's second stress formula defined on the scalar products. (g) (1) If the maximum number of iterations has been exceeded, we go to Step h with the current X . (2) If the goodness-of-fit has improved by more than the precision value, then we set $B^* = B^m$, and go to Step c. (3) If the goodness-of-fit has improved by less than the precision value, then we go to Step h with the current matrix X . (4) If the goodness-of-fit has worsened, we retrieve the saved X and go to Step h. In theory, this should never occur. (h) Print out the appropriate X , and compute and print its Euclidean distances, if desired.

Availability. SCALAR may be obtained by writing to the first author at the following address: Psychometric Laboratory, University of North Carolina, Chapel Hill, North Carolina 27514.

REFERENCES

- Guttman, L. A general nonmetric technique for finding the smallest coordinate space for a configuration of points. *Psychometrika*, 1968, 33, 469-506.
- Kruskal, J. B. Multidimensional scaling by optimizing goodness-of-fit to a nonmetric hypothesis. *Psychometrika*, 1964, 29, 1-28.
- Torgerson, W. S. *Theory and methods of scaling*. New York: Wiley, 1958.
- Young, F. W. On describing cardinal data with an ordinal model. L. L. Thurstone Psychometric Laboratory Report No. 130, February 1974.