HOW TO USE SMACOF - I

A PROGRAM FOR METRIC MULTIDIMENSIONAL SCALING

WILLEM HEISER

Department of Psychology

&

JAN de LEEUW

Department of Datatheory

University of Leiden
The Netherlands

AUGUST 1977

CONTENTS

SMACOF-I

Scaling by Maximizing a Convex Function -
Metric Euclidian version.

1.0. Introduction.

In multidimensional scaling (MDS) problems the data consist of m nonnegative square matrixes $\Delta_1, \Delta_2, \ldots, \Delta_m$ of order n, whose elements are interpreted as measures of dissimilarity between the n objects $o_1, o_2, \ldots, o_n$, measured at m replications $r_1, r_2, \ldots, r_m$. Thus δ_{ijk} is the dissimilarity between objects o_i and o_j at replication r_k.
In a psychological context the objects are often called stimuli, and the replications are defined by the dissimilarity judgments of different subjects or at different occasions.
Furthermore we assume that m nonnegative square matrixes $W_1, W_2, \ldots, W_m$ of order n are given, whose elements are interpreted as weights, i.e. w_{ijk} indicates the relative importance or precision of measurement δ_{ijk}.

Multidimensional scaling techniques represent the objects $o_1, o_2, \ldots, o_n$ as points $x_1, x_2, \ldots, x_n$ in a metric space $\langle \Omega, d \rangle$ in such a way that the distances $d(x_i, x_j)$ are approximately equal to the dissimilarities δ_{ijk}. We sometimes write d_{ij} or $d_{ij}(X)$ for $d(x_i, x_j)$. In the problem handled by SMACOF-I the representation is in the space of all p-tuples of real numbers, in which the metric is defined by the euclidian norm. Thus a representation of $o_1, o_2, \ldots, o_n$ is the n x p configuration matrix X, with elements x_{is} $(i = 1, \ldots, n;\ s = 1, \ldots, p)$, assumed to be centered, and the $d_{ij}(X)$ are euclidian distances, defined on the rows of X by

$$d_{ij}(X) = \left[\sum_{s=1}^{p} (x_{is} - x_{js})^2 \right]^{\frac{1}{2}} . \qquad (1)$$

To evaluate the badness-of-fit of a particular configuration X we can use the loss function

$$\sigma(X) = \sum_{k=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ijk} \, (\delta_{ijk} - d_{ij}(X))^2 . \qquad (2)$$

Clearly $\sigma(X) \geqslant 0$, and $\sigma(X) = 0$ if and only if $d_{ij}(X) = \delta_{ijk}$ for all i,j,k with $w_{ijk} \neq 0$. If $w_{ijk} = 0$ then the value of $\sigma(X)$ does not depend on δ_{ijk}. This provides us with a simple device for handling missing data: if the observation corresponding with the triple i,j,k is missing, then we can choose δ_{ijk} arbitrarily, and set $w_{ijk} = 0$.

1.1. Other work.

For a general discussion of MDS, which includes the nonmetric case, in which the dissimilarities are only partially known, the generalization to Minkovski metrics and a convergence proof of the SMACOF algorithm, see de Leeuw (1977). A less technical general overview of MDS techniques is given by Young (1972). For a discussion of different loss functions, see Kruskal and Carroll (1969) and de Leeuw and Heiser (1977), in which also generalizations to individual differences scaling are treated. Classical metric scaling methods apply both squaring and double centering to the dissimilarities, and then define the loss on the scalar products (Torgerson 1958). The first nonmetric method is due to Shepard (1962) who defines a similar loss function but does not explicitly minimize it. Algorithms for the minimization of $\sigma(X)$ have been proposed by Kruskal (1964 a,b) and Guttman (1968). A detailed discussion and comparison of the algorithms and the corresponding computer programs is available in Lingoes and Roskam (1973). Interesting geometrical and mechanical interpretations of these algorithms have been discussed by Kruskal and Hart (1965), Mc Gee (1966) and Gleason (1967). A metric version of this approach is treated in Roskam (1972). In this paper we only give detailed algebraic derivations which are relevant for the SMACOF-I program.

1.2. Problem reduction: partitioning of loss.

We can reduce the MDS problem to a more simple form by partitioning the loss function into additive components. First we split off a component which measures loss due to differences between replications. For this purpose we define

$$\bar{w}_{ij} = \frac{1}{m} \sum_{k=1}^{m} w_{ijk} , \qquad (3)$$

$$\underline{\delta}_{ij} = \frac{\sum_{k=1}^{m} w_{ijk}\,\delta_{ijk}}{m\underline{w}_{ij}}\,, \tag{4}$$

and the partitioning of $\sigma(X)$ becomes:

$$\sigma(X) = \sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ijk}\,(\delta_{ijk} - d_{ij}(X))^2 =$$

$$\sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ijk}\,(\delta_{ijk} - \underline{\delta}_{ij} + \underline{\delta}_{ij} - d_{ij}(X))^2 =$$

$$\sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ijk}\,(\delta_{ij} - \underline{\delta}_{ij})^2 + m\sum_{i=1}^{n}\sum_{j=1}^{n} \underline{w}_{ij}\,(\underline{\delta}_{ij} - d_{ij}(X))^2, \tag{5}$$

where the cross product vanishes. The partitioning can be refined by splitting up the second component. We define

$$w_{ij} = \frac{1}{2}\,(\underline{w}_{ij} + \underline{w}_{ji})\,, \tag{6}$$

$$\delta_{ij} = \frac{\underline{w}_{ij}\underline{\delta}_{ij} + \underline{w}_{ji}\underline{\delta}_{ji}}{\underline{w}_{ij} + \underline{w}_{ji}}\,. \tag{7}$$

Now we get:

$$m\sum_{i=1}^{n}\sum_{j=1}^{n} \underline{w}_{ij}\,(\underline{\delta}_{ij} - d_{ij}(X))^2 =$$

$$\frac{1}{2}\,m\sum_{i=1}^{n}\sum_{j=1}^{n} \left[\, \underline{w}_{ij}\,(\underline{\delta}_{ij} - d_{ij}(X))^2 + \underline{w}_{ji}\,(\underline{\delta}_{ji} - d_{ij}(X))^2 \,\right] =$$

$$\frac{1}{2}\,m\sum_{i=1}^{n}\sum_{j=1}^{n} \left[\, \underline{w}_{ij}\underline{\delta}_{ij}^2 - 2\underline{w}_{ij}\underline{\delta}_{ij}d_{ij}(X) + \underline{w}_{ij}d_{ij}^2(X) +\right.$$

$$\left. + \underline{w}_{ji}\underline{\delta}_{ji}^2 - 2\underline{w}_{ji}\underline{\delta}_{ji}d_{ij}(X) + \underline{w}_{ji}d_{ij}^2(X) \,\right] =$$

$$\frac{1}{2} m \sum_{i=1}^{n} \sum_{j=1}^{n} \left[\underline{w}_{ij} \underline{\delta}_{ij}^2 + \underline{w}_{ji} \underline{\delta}_{ji}^2 - 2 w_{ij} \delta_{ij}^2 \right] +$$

$$+ \frac{1}{2} m \sum_{i=1}^{n} \sum_{j=1}^{n} \left[2 w_{ij} \delta_{ij}^2 - 2 (\underline{w}_{ij} \underline{\delta}_{ij} + \underline{w}_{ji} \underline{\delta}_{ji}) d_{ij}(X) + (\underline{w}_{ij} + \underline{w}_{ji}) d_{ij}^2(X) \right] =$$

$$m \sum_{i=1}^{n} \sum_{j=1}^{n} (\underline{w}_{ij} \underline{\delta}_{ij}^2 - w_{ij} \delta_{ij}^2) + m \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (\delta_{ij} - d_{ij}(X))^2 . \quad (8)$$

As is customary in the analysis of variance, we collect the components of the partitioning in a table:

SOURCE	LOSS COMPONENT
Proper loss	$m \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (\delta_{ij} - d_{ij}(X))^2$
Symmetry	$m \sum_{i=1}^{n} \sum_{j=1}^{n} (\underline{w}_{ij} \underline{\delta}_{ij}^2 - w_{ij} \delta_{ij}^2)$
Individual differences	$\sum_{k=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ijk} (\delta_{ijk} - \underline{\delta}_{ij})^2$
Total loss	$\sum_{k=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ijk} (\delta_{ijk} - d_{ij}(X))^2$

It is obvious that we minimize the total loss if we minimize the proper loss, and that the proper loss is more simple. In fact in defining the proper loss we can suppose without loss of generality that both the weights and the dissimilarities are symmetric. The only assumption we make about the δ_{ij} is that they are nonnegative. The weights are also assumed to be nonnegative, but we make the additional nondegeneracy assumption of irreducibility: we suppose that there is no partitioning of $\{1,2,\ldots,n\}$ such that $w_{ij} = 0$ whenever i and j belong to different members of the partition. Again this assumption causes no real loss of generality, because if all between-subset weights are zero the MDS problem separates into a number of smaller problems corresponding with each of the subsets.

1.3. Problem manipulation: use of homogeneity.

As the next step in the derivation of the algorithm we use the homogeneity of the distance function $d_{ij}(X)$.
Because $d_{ij}(\beta X) = \beta d_{ij}(X)$ for all $\beta \geq 0$, we can minimize the proper loss by minimizing

$$\sigma_o(X) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}(\delta_{ij} - \beta d_{ij}(X))^2 \qquad (9)$$

over $\beta \geq 0$ and over all normalized X. We call a centered configuration matrix X normalized if $\eta(X) = 1$, where

$$\eta^2(X) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} d^2_{ij}(X) \ . \qquad (10)$$

For convenience we also assume that the dissimilarities $\Delta = \left\{ \delta_{ij} \right\}$ are normalized in the sense that $\eta(\Delta) = 1$, where

$$\eta^2(\Delta) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \delta^2_{ij} \ . \qquad (11)$$

Expanding (9), we see that

$$\sigma_o(X) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \delta^2_{ij} + \beta^2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} d^2_{ij}(X) - 2\beta \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \delta_{ij} d_{ij}(X)$$

$$= 1 + \beta^2 - 2\beta \rho(X), \qquad (12)$$

using both normalizations and defining

$$\rho(X) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \delta_{ij} d_{ij}(X) \ . \qquad (13)$$

The minimum of (12) over $\beta \geq 0$ for a fixed normalized X is $1-\rho^2(X)$, which is attained for $\hat{\beta} = \rho(X)$. Thus we can minimize $\sigma_o(X)$ by maximizing $\rho(X)$ over all normalized configurations. If $\underline{X}$ solves this maximization problem, then $\rho(\underline{X}).\underline{X}$ minimizes $\sigma(X)$.

The situation is illustrated in figure 1. Here the two matrices $\{\delta_{ij}\}$ and $\{d_{ij}(X)\}$ are depicted as two vectors δ and $d(X)$ on the unit circle.

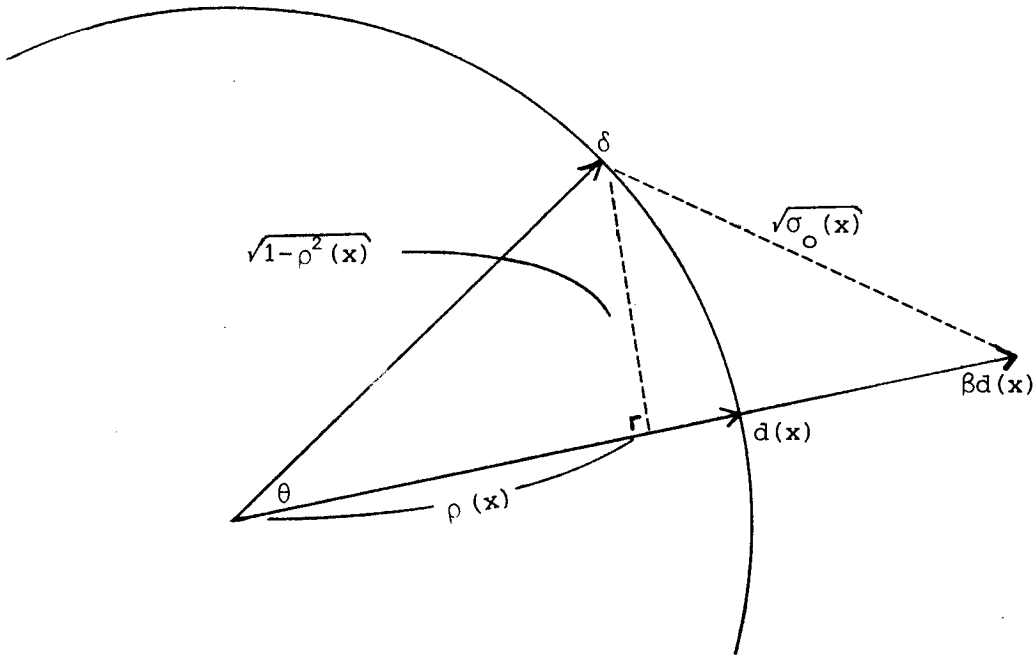


figure 1. The relation between $\rho(X)$ and $\sigma_0(X)$.

For simplicity, the weights are omitted. For fixed $d(X)$, the minimum of $\sigma_0(X)$ over β clearly is the projection of δ on $d(X)$ or $\cos\theta = \rho(X)$. So instead of trying to find a $d(X)$ which has least distance to δ we can try to find a $d(X)$ on which the projection of δ is greatest.

1.4. Algorithm.

We now describe the algorithm using matrix notation.
The normalization (10) can be written as

$$\eta^2(X) = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} \sum_{s=1}^{p} (x_{is} - x_{js})^2$$

$$= \sum_{s=1}^{p}\sum_{i=1}^{n}\sum_{j\neq i}^{n} w_{ij}(x_{is} - x_{js})^2$$

$$= \sum_{s=1}^{p}\sum_{i=1}^{n}\sum_{j\neq i}^{n} w_{ij}x_{is}^2 + \sum_{s=1}^{p}\sum_{i=1}^{n}\sum_{j\neq i}^{n} w_{ij}x_{js}^2 - 2\sum_{s=1}^{p}\sum_{i=1}^{n}\sum_{j\neq i}^{n} w_{ij}x_{is}x_{js}$$

$$= \sum_{s=1}^{p} \sum_{i=1}^{n} x_{is}^2 \sum_{k\neq i}^{n} w_{ik} + \sum_{s=1}^{p} \sum_{j=1}^{n} x_{js}^2 \sum_{k\neq j}^{n} w_{kj} - 2 \sum_{s=1}^{p} \sum_{i\neq j}^{n} \sum_{j}^{n} w_{ij} x_{is} x_{js} . \quad (14)$$

We now define a matrix V as

$$v_{ij} = -2w_{ij} \quad \text{if} \quad i \neq j, \quad (15a)$$

$$v_{ii} = \sum_{k\neq i}^{n} 2w_{ik} , \quad (15b)$$

and we can write (14) as

$$\eta^2(X) = \sum_{s=1}^{p} \sum_{i=j}^{n} \sum^{n} v_{ij} x_{is} x_{js} + \sum_{s=1}^{p} \sum_{i\neq j}^{n} \sum^{n} v_{ij} x_{is} x_{js}$$

$$= \sum_{s=1}^{p} \sum_{i=1}^{n} \sum_{j=1}^{n} v_{is} x_{is} x_{js}$$

$$= \text{tr } X'VX . \quad (16)$$

In a similar way we can derive

$$\rho(X) = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{w_{ij}\delta_{ij}}{d_{ij}(X)} d_{ij}^2(X)$$

$$= \sum_{s=1}^{p} \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij}(X) x_{is} x_{js}$$

$$= \text{tr } X'B(X)X , \quad (17)$$

where the matrix B(X) is defined by

$$b_{ij}(X) = \frac{-2w_{ij}\delta_{ij}}{d_{ij}(X)} \quad \text{if} \quad i\neq j , \quad (18a)$$

$$b_{ii}(X) = \sum_{k\neq i}^{n} \frac{2w_{ik}\delta_{ik}}{d_{ik}(X)} , \quad (18b)$$

$$b_{ij}(X) = 0 \quad \text{if} \quad d_{ij}(X) = 0 . \quad (18c)$$

Both B(X) and V are real symmetric matrices with nonpositive off-diagonal and nonnegative diagonal elements, whose rows and columns sum to zero. By a familiar matrix theorem they are consequently both positive semi-definite of rank not exceeding n-1. Because V is irreducible by assumption we have in fact rank(V) = n-1, and the null space of V is the set of all vectors with constant elements. If e is the n-vector with all elements equal to one, then the Moore-Penrose inverse of V is

$$V^+ = (V + \frac{1}{n} ee')^{-1} - \frac{1}{n} ee' \quad . \tag{19a}$$

In fact it can be shown that

$$VV^+ = I - \frac{1}{n} ee' \quad , \tag{19b}$$

from which the Moore-Penrose conditions are easily verified. Moreover, for an arbitrary centered matrix V

$$VV^+X = (I - \frac{1}{n} ee')X = X \quad , \tag{19c}$$

a fact which we will use in the sequel.

The fundamental inequality, derived in de Leeuw (1977), on which the algorithm is based can be written as

$$\rho(X) \geqslant \text{tr } X'B(Y)Y \quad . \tag{20}$$

This inequality is true for all pairs of configuration matrices X,Y; it shows that $\rho(X)$ majorizes a family of linear functions, and the algorithm derives from this fact. Suppose X_μ is our current best normalized solution, where μ is the subscript for iterations. We first compute

$$Y_\mu = V^+B(X_\mu)X_\mu \quad , \tag{21}$$

and then we compute $X_{\mu+1}$ by normalizing Y_μ:

$$X_{\mu+1} = \frac{Y_\mu}{\eta(Y_\mu)} \quad . \tag{22}$$

We will now show that the following inequalities are true:

$$\rho(X_\mu) \leqslant \eta(Y_\mu) \ , \tag{23}$$

$$\eta(Y_\mu) \leqslant \rho(X_{\mu+1}) \ . \tag{24}$$

To see (23), notice that

$$\begin{aligned} \text{tr } X_\mu' V Y_\mu &= \text{tr } X_\mu' V V^+ B(X_\mu) X_\mu \\ &= \text{tr } X_\mu' B(X_\mu) X_\mu \\ &= \rho(X_\mu) \ , \end{aligned} \tag{25}$$

by using (21) and (17). Furthermore, the Cauchy-Schwartz inequality implies

$$\begin{aligned} \text{tr } X_\mu' V Y_\mu &\leqslant \left[\text{tr } X_\mu' V X_\mu . \text{tr } Y_\mu' V Y_\mu \right]^{\frac{1}{2}} \\ &\leqslant \left[\text{tr } Y_\mu' V Y_\mu \right]^{\frac{1}{2}} \\ &\leqslant \eta(Y_\mu) \ , \end{aligned} \tag{26}$$

because X_μ is normalized and using (16). Taken together, (25) and (26) imply (23). To verify (24), we can write

$$\begin{aligned} \eta(Y_\mu) &= \frac{1}{\eta(Y_\mu)} \text{tr } Y_\mu' V Y_\mu \\ &= \text{tr } X_{\mu+1}' V Y_\mu \\ &= \text{tr } X_{\mu+1}' V V^+ B(X_\mu) X_\mu \\ &= \text{tr } X_{\mu+1}' B(X_\mu) X_\mu \\ &\leqslant \rho(X_{\mu+1}) \ , \end{aligned} \tag{27}$$

using (16), (22), (21) and (20) respectively. Now, taking (23) and (24) together, we get

$$\rho(X_\mu) \leqslant \eta(Y_\mu) \leqslant \rho(X_{\mu+1}) \ , \tag{28}$$

which means that both $\rho(X_\mu)$ and $\eta(Y_\mu)$ are bounded increasing sequences, converging to the same limit. If we define

$$\zeta(X_\mu, X_{\mu+1}) = \frac{1}{2} \text{ tr } (X_\mu - X_{\mu+1})'V(X_\mu - X_{\mu+1}) \ , \tag{29}$$

then, because both X_μ and $X_{\mu+1}$ are normalized and using (22) and (25)

$$\begin{aligned} \zeta(X_\mu, X_{\mu+1}) &= \frac{1}{2} \text{ tr } X_\mu'VX_\mu - \text{tr } X_\mu'VX_{\mu+1} + \frac{1}{2} \text{ tr } X_{\mu+1}'VX_{\mu+1} \\ &= 1 - \text{tr } X_\mu'VX_{\mu+1} \\ &= 1 - \frac{\text{tr } X_\mu'VY_\mu}{\eta(Y_\mu)} \\ &= 1 - \frac{\rho(X_\mu)}{\eta(Y_\mu)} \ . \end{aligned} \tag{30}$$

So $\zeta(X_\mu, X_{\mu+1})$ converges to zero. For a more complete account of the convergence proof, see de Leeuw and Heiser (1977).

1.5. Specifics of the algorithm.

To illustrate the way in which the algorithm works we reformulate the basic iteration scheme (21) for the unweighted case. We get the following results:

$$W = ee' \tag{31}$$

$$V = 2n(I - \frac{1}{n} ee') \tag{32}$$

$$V^+ = \frac{1}{2n} (I - \frac{1}{n} ee') \tag{33}$$

We now omit the subscript for iterations and write Y for the update of X:

$$\begin{aligned} Y &= V^+B(X)X \\ &= \frac{1}{2n} (I - \frac{1}{n} ee')B(X)X \\ &= \frac{1}{2n} B(X)X \ . \end{aligned} \tag{34}$$

To derive a formula for the displacement of a particular point x_{is} to y_{is} we rewrite (34) as

$$y_{is} = \frac{1}{2n} \sum_{j=1}^{n} b_{ij}(X) x_{js}$$

$$= \frac{1}{2n} \left[\sum_{k \neq i}^{n} \frac{2\delta_{ik}}{d_{ik}} x_{is} - \sum_{j \neq i}^{n} \frac{2\delta_{ij}}{d_{ij}} x_{js} \right] . \tag{35}$$

Notice that because X is centered we have

$$x_{is} = - \sum_{j \neq i}^{n} x_{js} . \tag{36}$$

Furthermore we define the total shift t_i and the specific shift s_{ij} as follows

$$t_i = \sum_{k \neq i}^{n} \left(1 - \frac{\delta_{ik}}{d_{ik}}\right) \tag{37}$$

$$s_{ij} = 1 - \frac{\delta_{ij}}{d_{ij}} \tag{38}$$

Now we get

$$y_{is} = \frac{1}{n} \sum_{k \neq i}^{n} \frac{\delta_{ik}}{d_{ik}} x_{is} - \frac{1}{n} \sum_{j \neq i}^{n} \frac{\delta_{ij}}{d_{ij}} x_{js}$$

$$= x_{is} - \frac{1}{n} (t_i + 1) x_{is} - \frac{1}{n} \sum_{j \neq i}^{n} \frac{\delta_{ij}}{d_{ij}} x_{js}$$

$$= x_{is} + \frac{1}{n} \sum_{j \neq i}^{n} (t_i + 1) x_{js} - \frac{1}{n} \sum_{j \neq i}^{n} \frac{\delta_{ij}}{d_{ij}} x_{js}$$

$$= x_{is} + \frac{1}{n} \sum_{j \neq i}^{n} (t_i + s_{ij}) x_{js} . \tag{39}$$

We see that the displacement $y_{is} - x_{is}$ is determined by a weighted sum of "forces" towards or away from the other points. If the point x_{is} is "too close" to the others, i.e. if in general its distances to the other points are smaller than the corresponding dissimilarities, then t_i will be negative and will cause a total shift away from the other points.

Moreover, there will be a specific shift towards a specific point x_{js} if s_{ij} is positive (if $\delta_{ij} < d_{ij}$) or away from it if s_{ij} is negative (if $\delta_{ij} > d_{ij}$). If $\delta_{ij} = d_{ij}$ for all j, then $s_{ij} = 0$ for all j and $t_i = 0$, so the point is not moved. If there is only one point, say ℓ, for which $d_{i\ell} \neq \delta_{i\ell}$, then $t_i = s_{i\ell}$ and there is not only a shift in the direction of ℓ but also towards or away from the centroid.

A special case of both theoretical and practical interest is metric one-dimensional scaling. From (35) we get

$$y_i = \frac{1}{n} \sum_{j \neq i}^{n} \frac{\delta_{ij}}{d_{ij}} (x_i - x_j)$$

$$= \frac{1}{n} \sum_{j \neq i}^{n} \delta_{ij} \operatorname{sign} (x_i - x_j) \; . \qquad (40)$$

This means that y, the update of x, only depends on the rank order of the x_i. Because there are only a finite number of possible rank orders, and because the algorithm cannot repeat any given rank order, this implies that we reach convergence of (40) in a finite number of steps. This sounds nice, but unfortunately it only reflects the fact that the one-dimensional metric MDS problem is a combinatorial optimization problem. If we do not know the optimal order of the points on the dimension then it is almost impossible to find the global optimum (if n is at all large) with the finite algorithm (40). If the order of the points on the dimension is known (or fixed), then the problem can be shown to be a monotone regression problem.

2.0. SMACOF - program description.

The algorithm is implemented in the FORTRAN-program SMACOF-I (Scaling by Maximizing a Convex Function). The logical steps of the program can be summarized as follows:

Start: set p equal to p_{max}
set μ equal to o
compute or read an initial configuration Y_o

Step 1: compute $d_{ij}(Y_\mu)$ and $\eta(Y_\mu)$

Step 2: given Y_μ and $\eta(Y_\mu)$, compute $X_{\mu+1}$, $d_{ij}(X_{\mu+1})$ and $\rho(X_{\mu+1})$

Step 3: if $\rho(X_{\mu+1}) - \rho(X_\mu) <$ criterion go to step 6

else: go to step 4

Step 4: compute $B(X_{\mu+1})$ and $Y_{\mu+1}$

Step 5: set μ equal to $\mu+1$

if $\mu > \mu_{max}$ go to step 6

else: go to step 1

Step 6: if $p_{min} = p$ stop

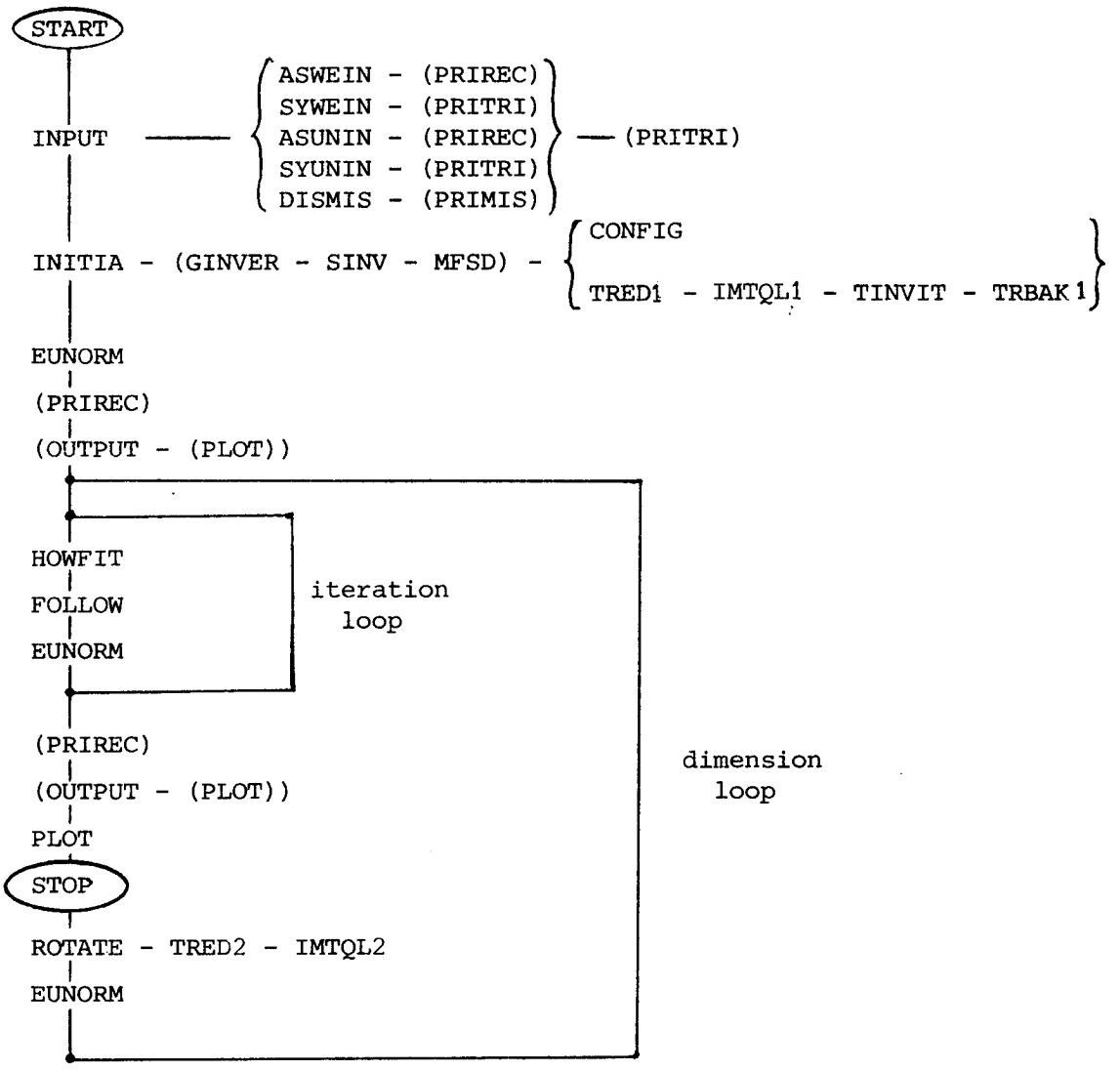
else: form Y_μ by rotating $X_{\mu+1}$ to its principal axes

set p equal to p-1

set μ equal to 1

go to step 1

In terms of its subroutines, the program is structured as follows:



The successive subroutine calls of the main program are given from top to bottom, whereas within subroutines further subroutine calls and -returns are pictured horizontally. As many computational details are commented upon in the source program, we only give a rough picture of the main features here.

In section 1.2. we reduced the more general MDS problem in several ways. Nevertheless the program will accept on input datamatrices according to the cells of table 1. with any number of replications, return the appropriate partitioning of loss components on output, and then work with the

	weighted	unweighted
asymmetric	ASWEIN	ASUNIN
symmetric	SYWEIN	SYUNIN

Table 1. Special purpose input routines for four different cases.

reduced problem. If missing values are present, the subroutine DISMIS will take care of the symmetric, unweighted case; other cases should be specified in weight matrices explicitly (with reasonable values in the missing entries).
The reason that several special purpose input subroutines were written instead of one general one derives from the fact that input/output operations tend to affect execution time considerably. For the same reason three printroutines are used: PRIREC, PRITRI and PRIMIS for rectangular, triangular and missing value matrices respectively. In the design of the "printplot" subroutine PLOT the major concern was minimizing the required space; this was done by plotting line after line instead of setting up a whole plottable.

Preliminary computations are performed by INITIA; in the weighted case it calls GINVER (which calls SINV which calls MFSD) to compute the generalized inverse V^+ (from 19). Upon the user's request, CONFIG reads an initial

configuration, otherwise INITIA constructs one by Torgersons (1958) method, i.e. by double centering the matrix of squared dissimilarities and then taking the p eigenvectors associated with the p largest eigenvalues, scaled to these eigenvalues.

Because the Torgerson procedure does not allow for missing values, DISMIS computes "reasonable" estimates by the following procedure. As the dissimilarities are assumed to be distance-like numbers, they should satisfy the triangle inequality. Thus

$$\delta_{ij} \leqslant \delta_{ik} + \delta_{jk} \tag{41}$$

for all i, j and k. This implies

$$\delta_{ij} \leqslant \min_{k} (\delta_{ik} + \delta_{jk}) \ . \tag{42}$$

On the other hand, the triangle inequality gives us

$$\delta_{ij} \geqslant |\delta_{ik} - \delta_{jk}| \ , \tag{43}$$

for all i, j and k, which implies

$$\delta_{ij} \geqslant \max_{k} |\delta_{ik} - \delta_{jk}| \ . \tag{44}$$

Now if δ_{ij} is missing, we use the mean of this greatest lower bound and the least upper bound as an estimate:

$$\hat{\delta}_{ij} = \frac{1}{2} (\min_{k}(\delta_{ik} + \delta_{jk}) + \max_{k} |\delta_{ik} - \delta_{jk}|) \ , \tag{45}$$

where k is taken over all non-missing entries in a row.

The eigenvalue-eigenvector decomposition is performed by four subroutines adapted from the EISPACK subroutine package. TRED1 reduces the double centered matrix to tridiagonal form using orthogonal similarity transformations,

IMTQL1 then determines all eigenvalues using the implicit QL-method, TINVIT determines only p eigenvectors of the tridiagonal matrix using inverse iteration and TRBAK1 transforms these back to eigenvectors of the original matrix. Especially for large problems (say n > 20 where p << n) this turns out to be considerably faster than a complete eigenvalue - eigenvector decomposition such as performed by TRED2/IMTQL2 (also from EISPACK; we use them for the rotation problem).

The three subroutines which constitute the SMACOF iteration scheme perform three of the logical steps described above. EUNORM computes "step 1", HOWFIT computes "step 2" and FOLLOW "step 4". Timing experiments indicated that FOLLOW dominates the speed of the program, so care has been taken to make it as efficient as possible (for instance by not explicitly computing $B(X_{\mu+1})$). The other steps are done in MAIN; if the criterion has been reached or the maximum number of iterations has been exceeded, it checks whether an analysis in a lower dimensionality is desired; if so, it calls ROTATE, which rotates $X_{\mu+1}$ to its principal axes (by computing the complete eigenvalue-eigenvector decomposition of $X'_{\mu+1}X_{\mu+1}$ and postmuliplication of $X_{\mu+1}$ with the normalized eigenvectors), and takes the first p-1 columns as initial configuration for the next analysis.

2.1. Size and speed.

The program is written according to the FORTRAN-ANS conventions. To reduce overhead in passing the parameters between program units, nearly all arrays and many control parameters are organized in labeled COMMON blocks. The limits set for the most important size determining parameters are 60 for the number of points (n) and 9 for the number of dimensions (p); the number of replications is unlimited. The required array area on the IBM-370 is 69 K; the object code from the FORTRAN-H-EXTENDED optimizing compiler requires 36 K (1572 source statements).

To get an idea of the relation between execution time and the size of a problem in terms of n and p, we analyzed several sets of uniformly distributed random data, both weighted and unweighted (on the IBM-370). In table 2 and figures 2 and 3 the results are given for p = 2 and p = 3 with n varying.

In table 3 and figure 4 results are given for n = 15 and n = 45 with p varying. In all cases, the minimum of I/O options was used.

		number of points											
		5	10	15	20	25	30	35	40	45	50	55	60
unweighted	p = 2	0.32	0.52	1.03	1.27	1.89	2.81	4.38	6.75	8.80	10.59	10.02	14.35
		(19)	(23)	(34)	(23)	(24)	(26)	(33)	(41)	(43)	(41)	(30)	(38)
	p = 3	0.30	0.47	0.80	1.73	2.44	4.12	4.62	7.96	8.26	15.87	18.05	22.48
		(10)	(16)	(18)	(33)	(41)	(40)	(30)	(43)	(34)	(57)	(52)	(55)
weighted	p = 2	0.34	0.54	1.14	2.10	2.65	7.91	6.63	12.28	14.20	19.24	21.10	27.99
		(22)	(16)	(26)	(32)	(25)	(66)	(37)	(58)	(52)	(57)	(51)	(57)
	p = 3	0.34	0.58	1.95	3.64	5.37	5.58	10.58	15.75	18.10	22.60	36.18	41.08
		(12)	(17)	(48)	(55)	(53)	(36)	(55)	(63)	(57)	(56)	(79)	(75)

Table 2. Execution times in seconds for random data (means over three different sets, mean number of iterations in parentheses).

As can be seen from figures 2 and 3, execution time increases quadratically with number of points; departures from this trend are due to irregularities in the number of iterations needed to reach the stopcriterion (some smoothing of the curves has already been obtained by taking the mean times over three different datasets). Furthermore, there is a large difference in time between the weighted and unweighted cases, reflecting the dominance of the updating routine where we compute $V^{+}B(X)X$ for the weighted and $B(X)X$ for the unweighted case.

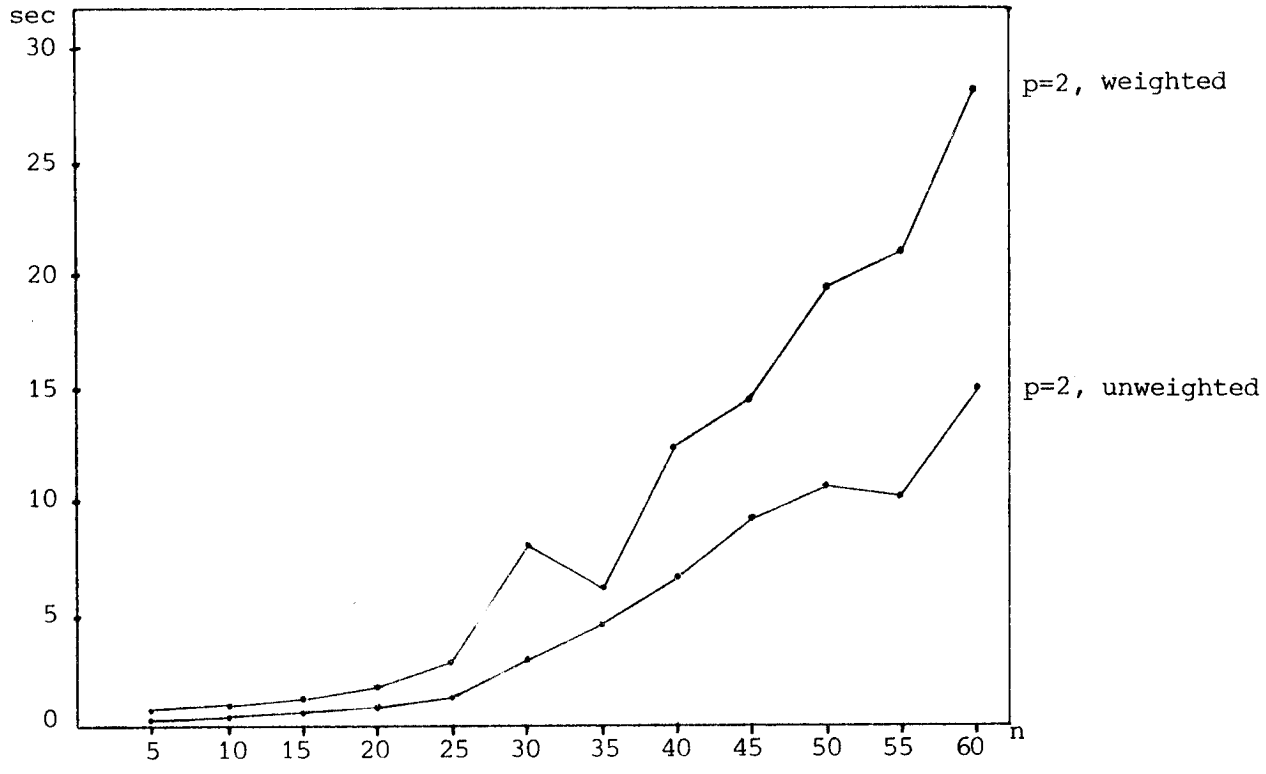


Figure 2. Plot of execution time versus number of points, 2 dimensions (see table 2).

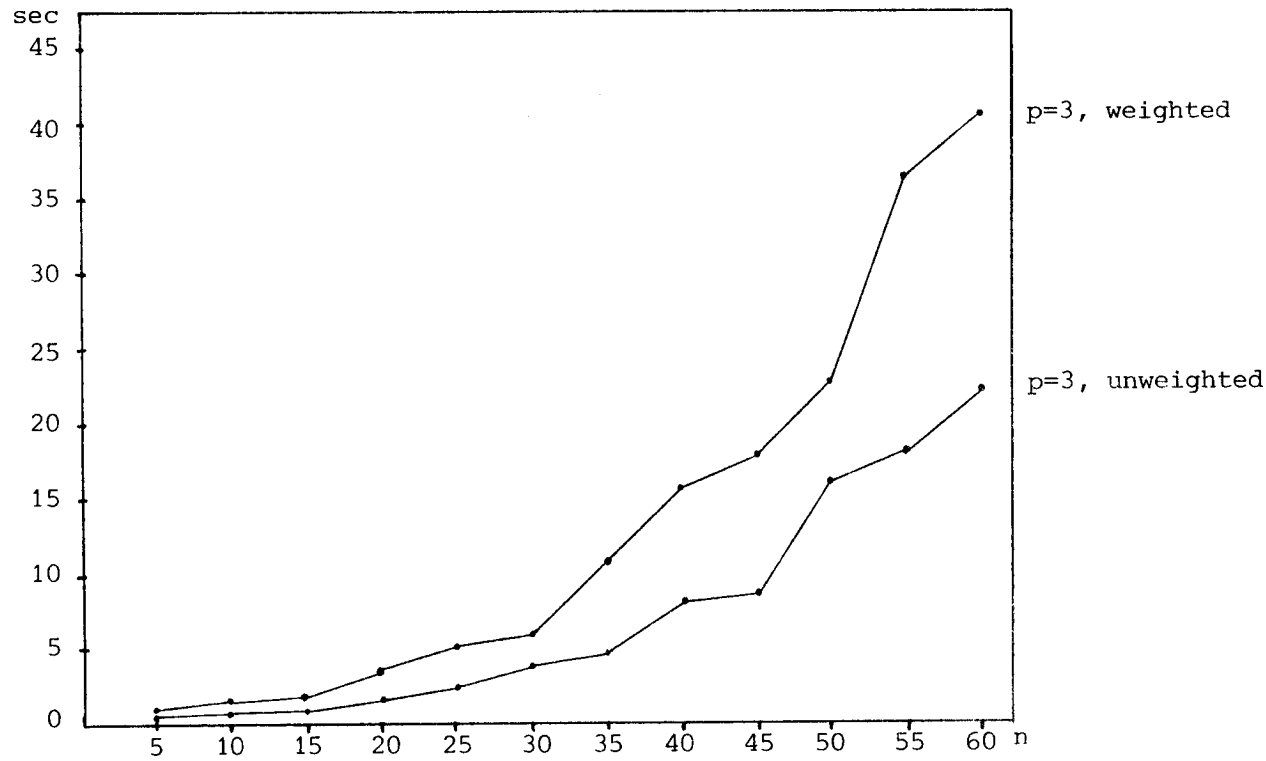


Figure 3. Plot of execution time versus number of points, 3 dimensions (see table 2).

		number of dimensions								
		1	2	3	4	5	6	7	8	9
unweighted	n = 15	0.46 (5)	1.03 (34)	0.80 (18)	1.16 (32)	1.07 (24)	1.12 (23)	1.35 (26)	1.48 (27)	1.55 (28)
	n = 45	2.95 (9)	8.80 (43)	8.26 (34)	9.58 (35)	10.18 (33)	10.35 (31)	8.91 (24)	10.01 (25)	11.33 (27)
weighted	n = 15	0.58 (5)	1.14 (26)	1.95 (48)	1.71 (35)	1.61 (28)	1.73 (27)	2.04 (29)	2.27 (30)	2.33 (30)
	n = 45	4.35 (9)	14.20 (52)	18.10 (57)	21.51 (61)	27.60 (69)	24.62 (55)	23.90 (48)	23.34 (42)	25.12 (41)

Table 3. Execution times in seconds for random data (means over three different sets, mean number of iterations in parentheses).

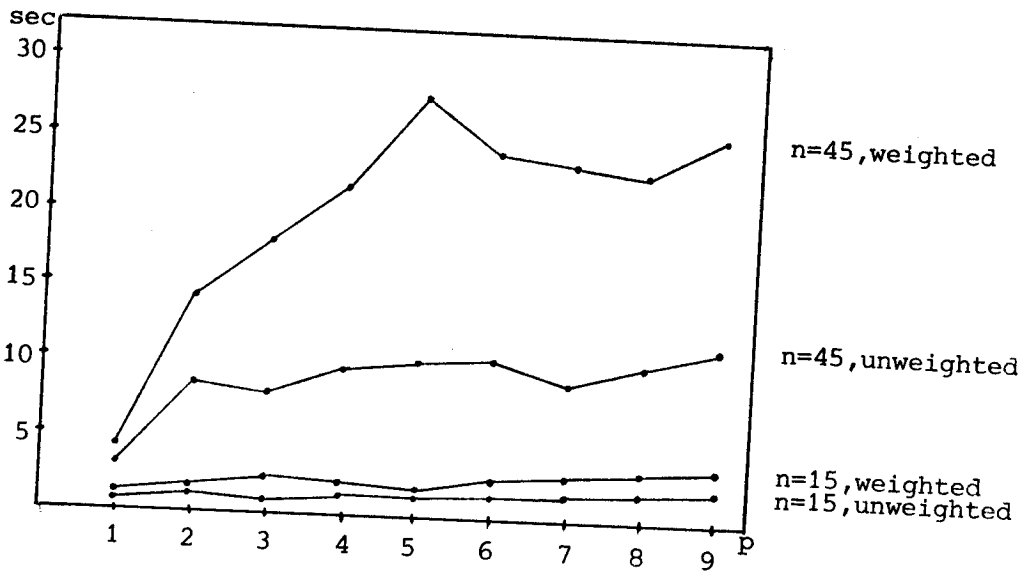


Figure 4. Execution time in seconds for n=15 and n=45, random data (see tables).

Figure 4 shows the small increase in time for different dimensionalities; again we have some dips and peaks due to irregular numbers of iterations. It is clear that the influence of p increases as n grows, although the curves also may seem to be asymptotic; this effect however is entirely due to the pattern in the number of iterations. If we analyse all cases of table 3 with the number of iterations fixed (making the stopcriterion extremely small) we get the results of table 4, pictured in figure 5.

		number of dimensions								
		1	2	3	4	5	6	7	8	9
u.	n = 15	0.44 (4)	0.88	1.00	1.08	1.15	1.24	1.35	1.41	1.56
	n = 45	2.76 (8)	6.58	7.24	7.96	8.76	9.70	10.45	11.30	12.13
w.	n = 15	0.61 (7)	1.21	1.33	1.49	1.60	1.74	1.92	2.08	2.32
	n = 45	4.35 (9)	9.66	10.97	11.97	13.72	15.01	16.05	17.35	18.74

Table 4. Execution times in seconds for random data (except for p = 1, number of iterations 30).

The dependence of execution time on p turns out to be linear now, if we discard p = 1, for which it typically is impossible to get as many iterations as for the other values of p.

In all analysis reported here, for each case a separate job was run with one fixed value of p. Of course, as the program permits to do a series of analyses with decreasing values of p, it is possible to reduce execution time for such a job of related problems, because then all kinds of preliminary computations need not to be performed. We did not investigate the influence of this procedure on convergence behaviour.

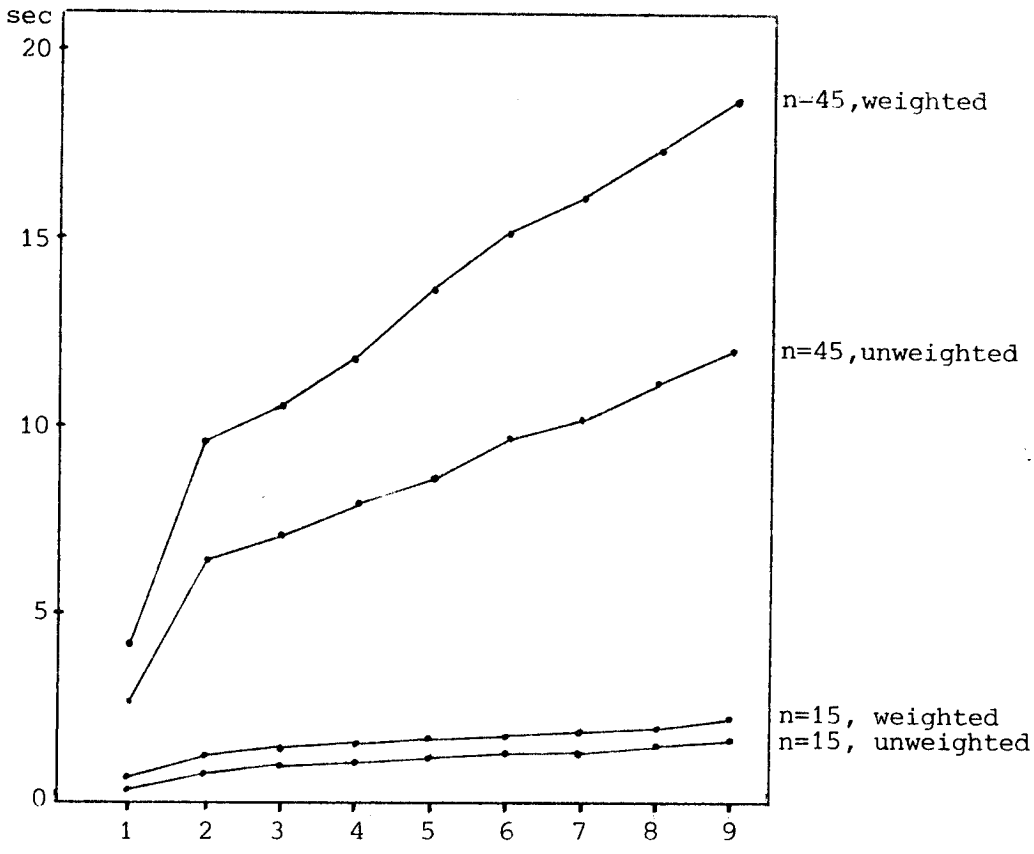


Figure 5. Execution time in seconds for n=15 and n=45, random data, number of iterations fixed (see table 4).

2.2. An example.

If we choose a particular configuration of points, compute the euclidian interpoint distances and then feed the program with these "perfect" data, it will recover the target configuration perfectly (up to a rotation of axes), as it should do. We will illustrate the program with a more "difficult" example, viz. a set of dissimilarities between ethnic groups from a study of Funk, Horowitz, Lipshitz and Young (1974), which is discussed extensively in the POLYCON user's guide (Young, 1973). The labels of the stimuli are given in table 5.

1.	anglo
2.	black
3.	chinese
4.	german
5.	indian
6.	irish
7.	italian
8.	japanese
9.	jewish
10.	mexican
11.	negro
12.	polish
13.	puerto-rican

Table 5. Labels of the stimuli used in the Funk e.a. (1974) study.

The data represent the dissimilarity between these ethnic subgroups of the american culture, as perceived by forty-nine psychology students. Funk e.a. also gathered rating scale data on a number of attributes and compared several different kinds of analysis. We will focus on the "fully metric" MDS solution, where their "quasi-nonmetric" and "nonmetric" solutions were very similar. The principal results of the POLYCON analysis are presented in figures 6 and 7. The stress for this solution was 0.3918, a very poor value, but if a three-dimensional solution was tried, again essentially three clusters emerged with small and medium data values becoming small withincluster distances and large data values becoming large betweencluster distances.

The output of the SMACOF analysis of these data is presented on the next few pages. After heading, summary of options and print of the data-matrix we get the results of the eigenvector-eigenvalue decomposition which is used as an initial configuration. Notice that the first two dimensions of this initial configuration have a very big resemblance to the POLYCON two-dimensional solution. The history of computation shows however, that we can

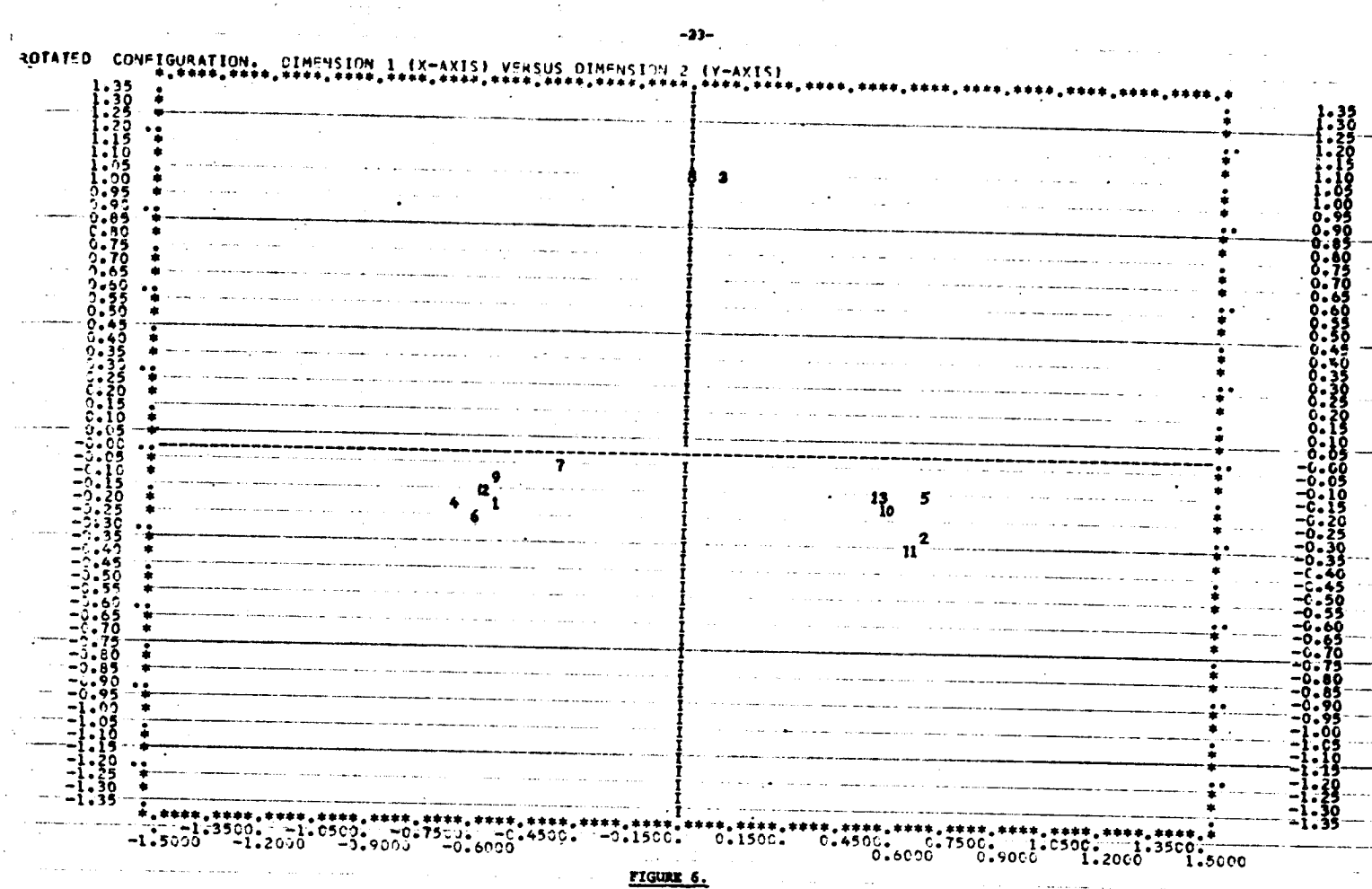


FIGURE 6.

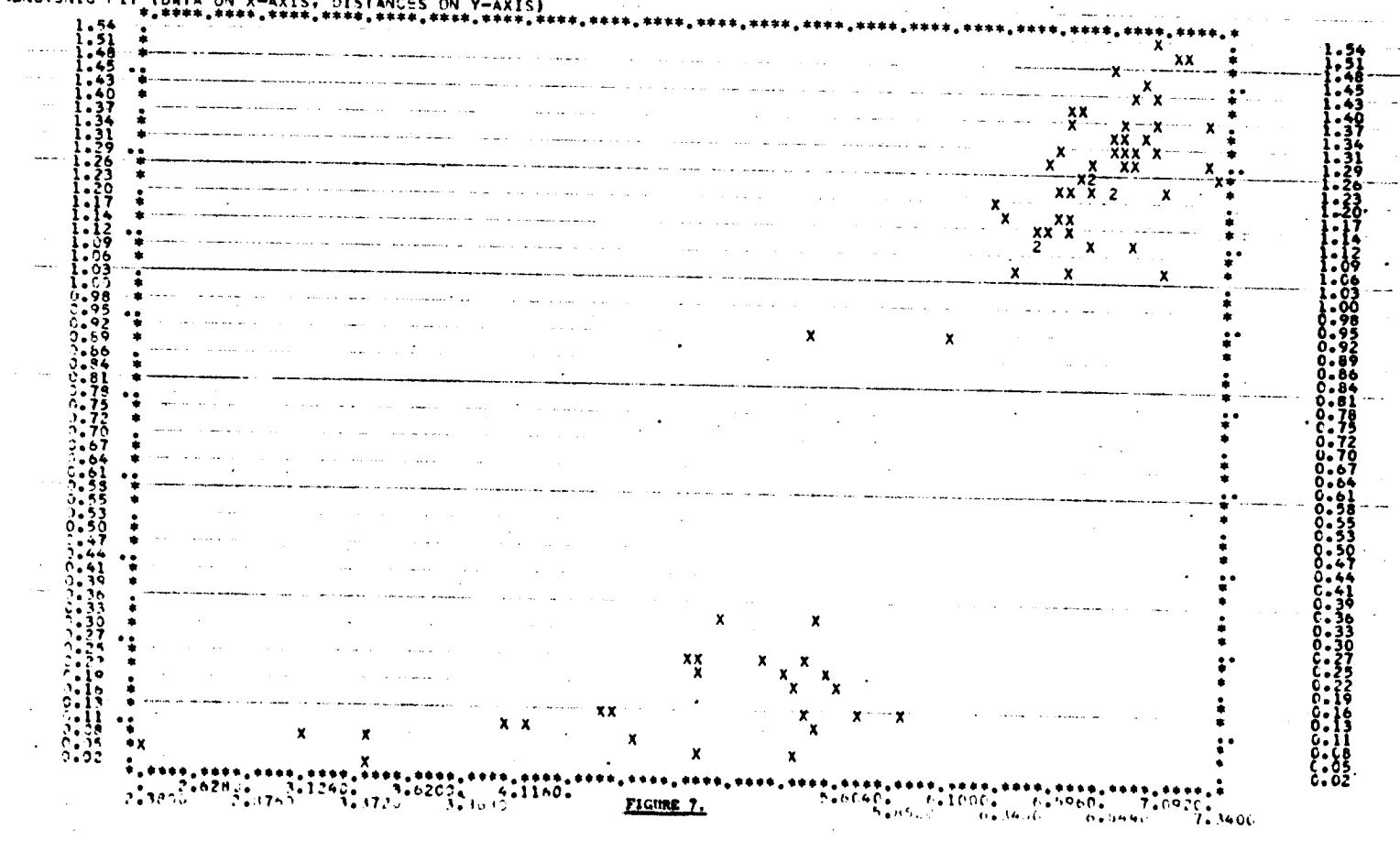


FIGURE 7.

S M A C O F - I
VERSION - 6
AUGUST 1976

METHIC MULTIDIMENSIONAL SCALING

WILLEM HEISER
DEPT. OF PSYCHOLOGY
&
JAN DE LEEUW
DEPT. OF DATATHEORY
THE UNIVERSITY OF LEIDEN

JOB TITLE: ***** ETNIC-DATA1UNWEIGHTED3,2 *****

ECHO OF PARAMETER CARDS:

```
13   1   0   0   0   0.0
 2   3   2   1   1   0   0   0
 3   2   0  00.0
```

DATA SPECIFICATIONS:

THE NUMBER OF POINTS IS 13
THE NUMBER OF REPLICATIONS IS 1
THE INPUT IS SYMMETRIC
NOT WEIGHTED

ANALYSIS SPECIFICATIONS:

THE MAXIMUM NUMBER OF DIMENSIONS IS 3
THE MINIMUM NUMBER OF DIMENSIONS IS 2
THE MAXIMUM NUMBER OF ITERATIONS IS 50
THE CONVERGENCE CRITERION IS 0.10E-04

ORIGINAL DATA MATRIX OF REPLICATION NO 1

	1	2	3	4	5	6	7	8	9	10	11	12
1	*											
2	6.610	*										
3	6.680	6.830	*									
4	4.550	7.000	7.040	*								
5	6.710	5.410	6.910	7.300	*							
6	3.400	6.910	7.010	4.080	7.040	*						
7	5.270	5.640	6.340	5.060	7.110	5.520	*					
8	6.630	7.170	3.130	6.630	7.050	6.960	6.510	*				
9	4.640	6.850	6.930	5.910	7.340	5.710	5.360	6.960	*			
10	6.490	5.620	6.720	6.630	4.500	6.590	6.100	6.820	6.920	*		
11	6.260	2.380	7.140	6.900	5.450	6.720	6.420	7.050	7.090	5.550	*	
12	5.410	6.740	6.870	4.190	7.270	5.490	4.970	6.860	4.970	6.650	6.830	*
13	6.520	4.970	6.540	6.750	5.470	6.660	5.460	6.620	6.740	3.430	4.910	6.540

	13
13	*

INITIAL CONFIGURATION ***** ETNIC-DATA1UNWEIGHTED3.2 *****

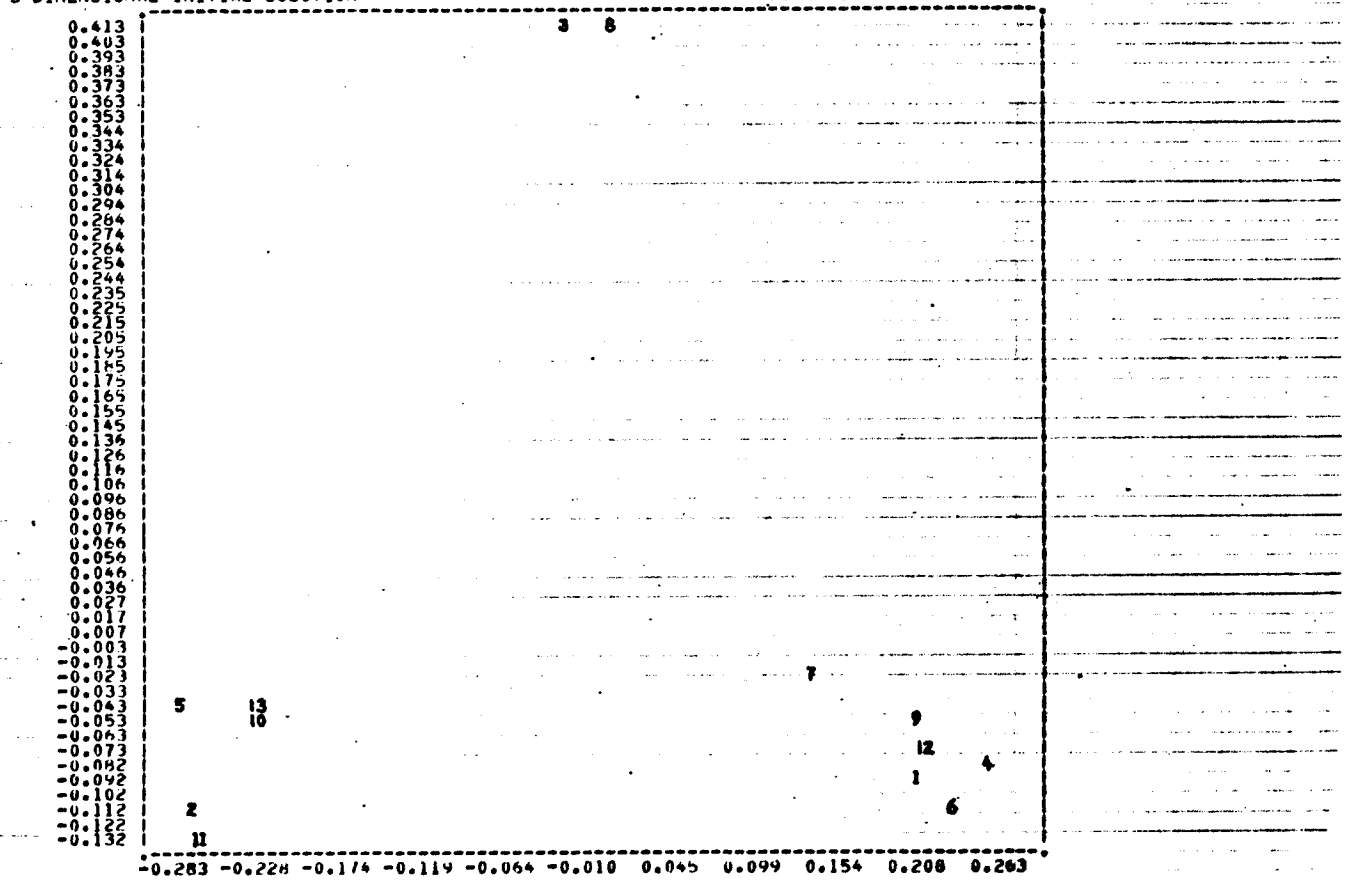
	1	2	3
1	0.211	-0.090	-0.061
2	-0.272	-0.114	-0.224
3	-0.018	0.412	-0.041
4	0.261	-0.031	0.024
5	-0.280	-0.042	0.047
6	0.232	-0.111	-0.016
7	0.139	-0.022	0.095
8	0.008	0.413	-0.041
9	0.215	-0.056	-0.034
10	-0.224	-0.046	0.250
11	-0.264	-0.133	-0.219
12	0.223	-0.072	0.014
13	-0.226	-0.047	0.157

THE EIGENVALUES ARE:

0.60710 0.41422 0.21563 0.19788 0.16426 0.13625 0.08690 0.05493 0.04215 0.03753
0.03153 0.01161 -0.00000

3-DIMENSIONAL INITIAL SOLUTION

DIMENSION 1 (X-AXIS) VERSUS DIMENSION 2 (Y-AXIS)



HISTORY OF COMPUTATION FOR ANALYSIS IN 3 DIMENSION(S)

***** ETNIC-DATA:UNWEIGHTED3.2 *****

ITERATION NO.	RHO	ETA	DEL	EPS	STRESS
1	0.952486	0.967496	0.952486	0.176152	0.304583
2	0.974542	0.978198	0.022056	0.086469	0.224207
3	0.980342	0.981708	0.005800	0.052761	0.197308
4	0.982624	0.983261	0.002283	0.036014	0.185606
5	0.983721	0.984061	0.001096	0.026313	0.179705
6	0.984325	0.984536	0.000605	0.020716	0.176362
7	0.984714	0.984864	0.000389	0.017469	0.174179
8	0.984996	0.985111	0.000282	0.015317	0.172577
9	0.985214	0.985304	0.000218	0.013567	0.171330
10	0.985385	0.985455	0.000172	0.011960	0.170340
11	0.985520	0.985577	0.000135	0.010786	0.169558
12	0.985631	0.985679	0.000111	0.009863	0.168912
13	0.985726	0.985768	0.000094	0.009316	0.168360
14	0.985811	0.985851	0.000085	0.009056	0.167861
15	0.985891	0.985930	0.000081	0.008897	0.167386
16	0.985970	0.986009	0.000079	0.008897	0.166921
17	0.986048	0.986087	0.000078	0.008843	0.166460
18	0.986127	0.986164	0.000079	0.008735	0.165991
19	0.986204	0.986240	0.000076	0.008680	0.165537
20	0.986278	0.986313	0.000075	0.008513	0.165093
21	0.986350	0.986383	0.000072	0.008229	0.164663
22	0.986416	0.986447	0.000066	0.007994	0.164265
23	0.986478	0.986506	0.000061	0.007564	0.163895
24	0.986534	0.986558	0.000056	0.007176	0.163559
25	0.986583	0.986605	0.000050	0.006766	0.163259
26	0.986628	0.986646	0.000045	0.006176	0.162989
27	0.986666	0.986683	0.000038	0.005859	0.162756
28	0.986701	0.986715	0.000034	0.005437	0.162548
29	0.986731	0.986743	0.000030	0.005167	0.162365
30	0.986758	0.986769	0.000027	0.004784	0.162201
31	0.986782	0.986791	0.000024	0.004475	0.162054
32	0.986803	0.986811	0.000021	0.004257	0.161926
33	0.986822	0.986830	0.000019	0.004143	0.161811
34	0.986840	0.986847	0.000018	0.003906	0.161702
35	0.986856	0.986862	0.000017	0.003654	0.161601
36	0.986872	0.986878	0.000015	0.003521	0.161506
37	0.986885	0.986892	0.000014	0.003906	0.161423
38	0.986901	0.986906	0.000015	0.003521	0.161330
39	0.986914	0.986920	0.000014	0.003782	0.161247
40	0.986927	0.986933	0.000013	0.003654	0.161165
41	0.986941	0.986947	0.000014	0.003383	0.161080
42	0.986955	0.986960	0.000013	0.003239	0.160998
43	0.986967	0.986972	0.000013	0.003383	0.160920
44	0.986980	0.986985	0.000012	0.003383	0.160845
45	0.986992	0.986996	0.000012	0.003239	0.160772
46	0.987003	0.987008	0.000012	0.002930	0.160699
47	0.987014	0.987018	0.000010	0.003088	0.160635
48	0.987025	0.987028	0.000011	0.002762	0.160566

NO FURTHER IMPROVEMENT OF RHO: THE FINAL VALUES ARE:

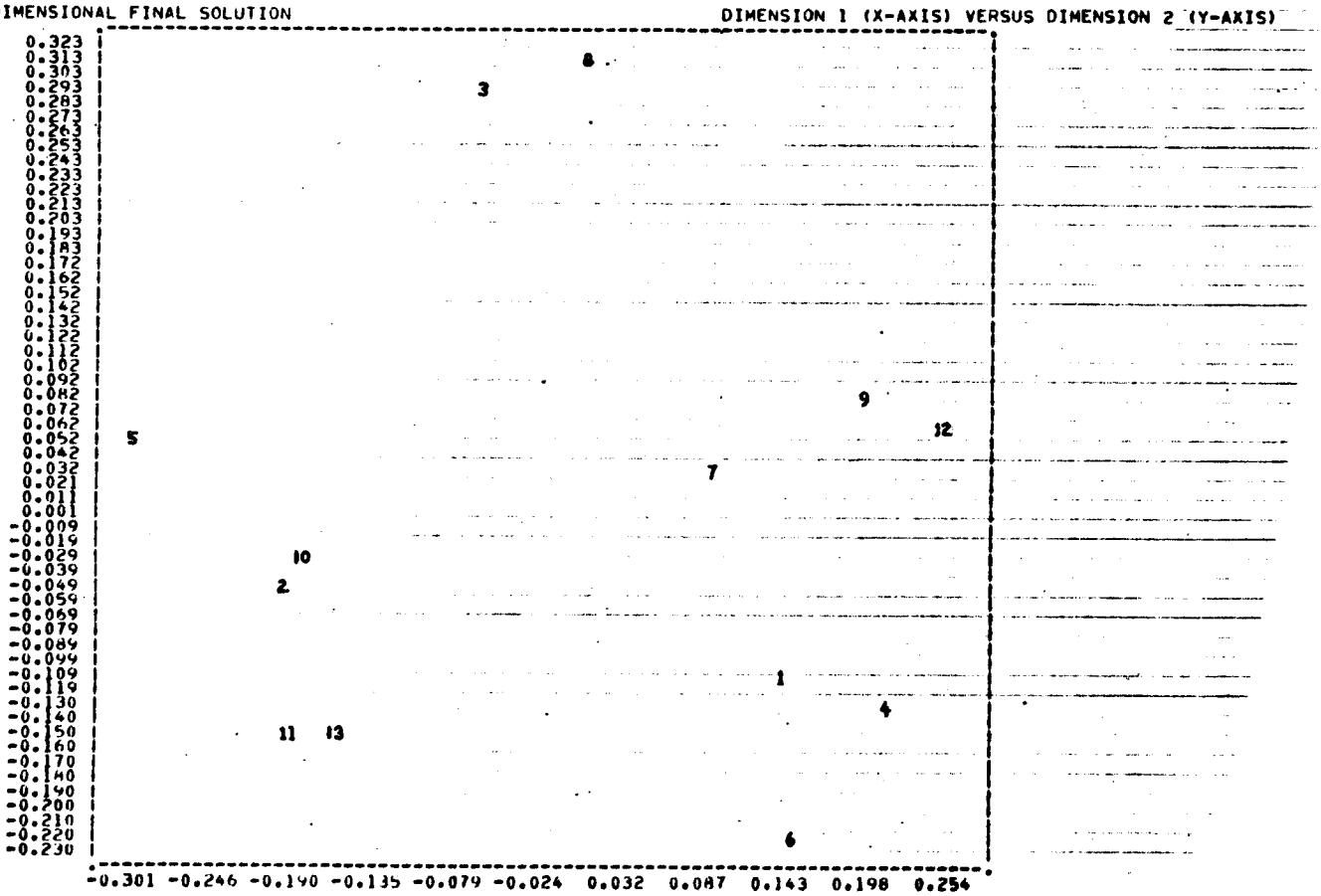
49	0.987034	0.987038	0.000009	0.002930	0.160512

FINAL 3-DIMENSIONAL CONFIGURATION FOR ***** ETNIC-DATA(UNWEIGHTED)3.2 *****

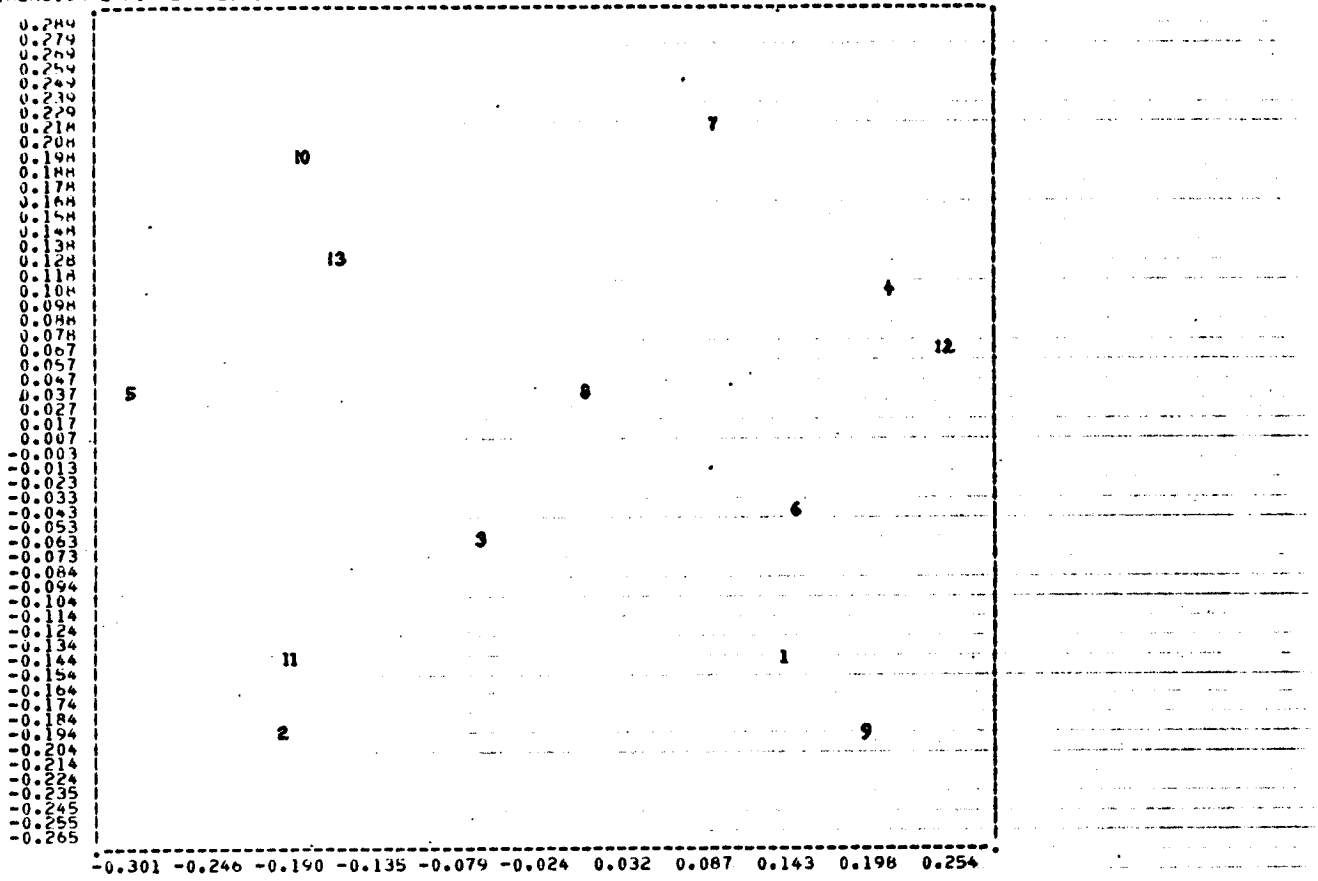
	1	2	3
1	0.150	-0.106	-0.147
2	-0.194	-0.053	-0.193
3	-0.063	0.246	-0.067
4	0.216	-0.130	0.110
5	-0.301	0.054	0.036
6	0.157	-0.221	-0.041
7	0.049	0.043	0.217
8	0.010	0.313	0.033
9	0.144	0.081	-0.194
10	-0.178	-0.032	0.197
11	-0.140	-0.146	-0.143
12	0.254	0.071	0.068
13	-0.148	-0.141	0.124

3-DIMENSIONAL FINAL SOLUTION

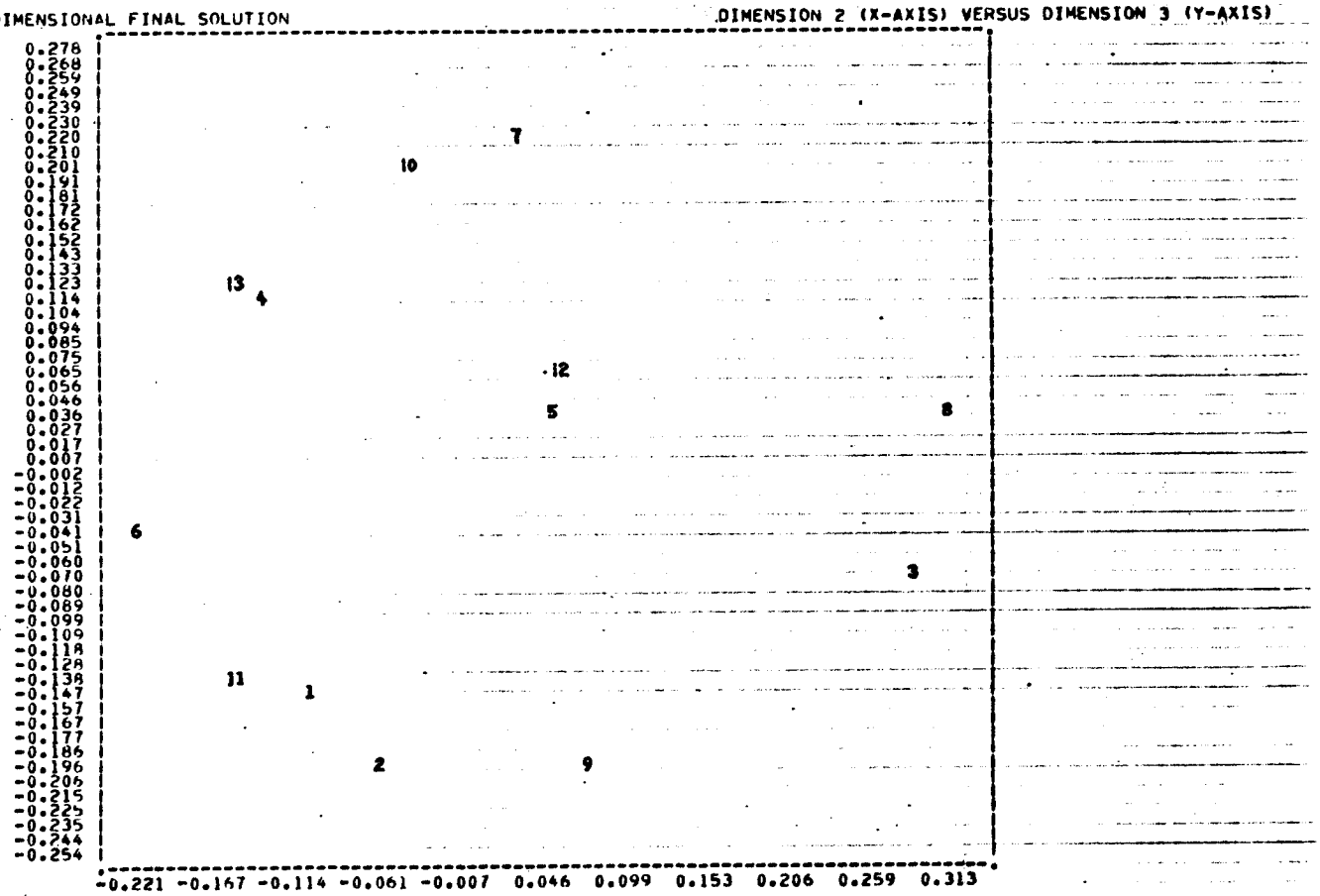
DIMENSION 1 (X-AXIS) VERSUS DIMENSION 2 (Y-AXIS)

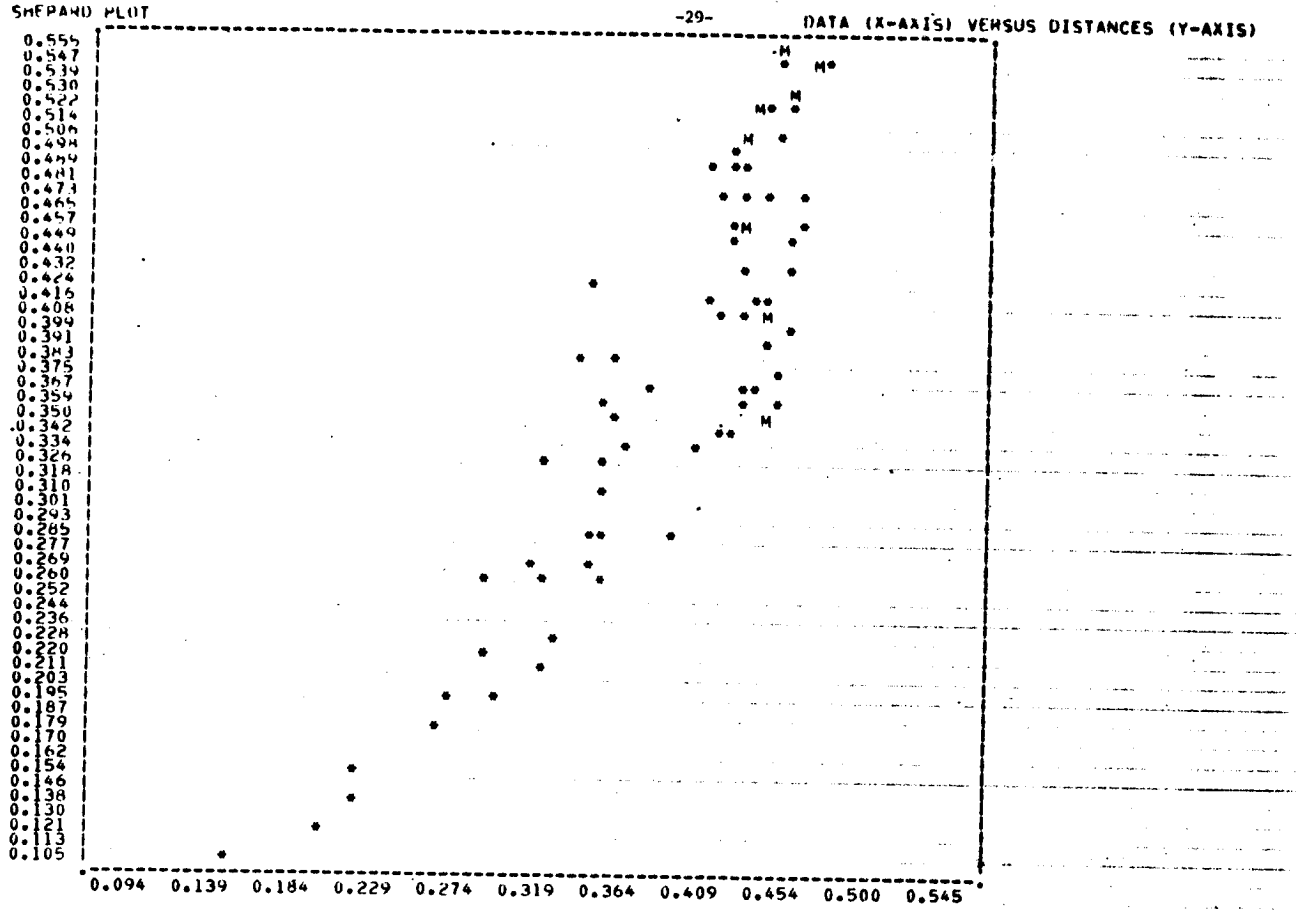


3-DIMENSIONAL FINAL SOLUTION -28- DIMENSION 1 (X-AXIS) VERSUS DIMENSION 3 (Y-AXIS)



3-DIMENSIONAL FINAL SOLUTION DIMENSION 2 (X-AXIS) VERSUS DIMENSION 3 (Y-AXIS)





HISTORY OF COMPUTATION FOR ANALYSIS IN 2 DIMENSION(S)

***** ETNIC-DATA:UNWEIGHTED3.2 *****

ITERATION NO.	RHO	ETA	DEL	EPS	STRESS
1	0.960563	0.963075	0.960563	0.072239	0.278064
2	0.964436	0.965254	0.003873	0.041190	0.264317
3	0.965787	0.966168	0.001351	0.028101	0.259337
4	0.966458	0.966690	0.000671	0.021880	0.256823
5	0.966879	0.967034	0.000421	0.017927	0.255235
6	0.967164	0.967272	0.000285	0.015002	0.254153
7	0.967365	0.967446	0.000201	0.013029	0.253388
8	0.967522	0.967592	0.000157	0.012000	0.252786
9	0.967660	0.967728	0.000138	0.011840	0.252256
10	0.967797	0.967867	0.000137	0.012000	0.251731
11	0.967938	0.968009	0.000140	0.012197	0.251191
12	0.968084	0.968156	0.000146	0.012236	0.250627
13	0.968230	0.968302	0.000146	0.012197	0.250060
14	0.968374	0.968443	0.000144	0.011960	0.249503
15	0.968511	0.968574	0.000137	0.011430	0.248971
16	0.968635	0.968691	0.000124	0.010831	0.248488
17	0.968745	0.968793	0.000110	0.010007	0.248060
18	0.968838	0.968879	0.000093	0.009213	0.247695
19	0.968916	0.968949	0.000078	0.008286	0.247389
20	0.968979	0.969006	0.000063	0.007501	0.247144
21	0.969030	0.969051	0.000052	0.006623	0.246941
22	0.969072	0.969088	0.000041	0.005940	0.246780
23	0.969105	0.969118	0.000033	0.005349	0.246649
24	0.969133	0.969143	0.000028	0.004784	0.246540
25	0.969156	0.969165	0.000023	0.004475	0.246449
26	0.969176	0.969184	0.000020	0.004143	0.246370

27	0.969193	0.969201	0.000017	0.004026	0.246301
28	0.969209	0.969215	0.000015	0.003782	0.246241
29	0.969223	0.969229	0.000015	0.003521	0.246184
30	0.969236	0.969241	0.000013	0.003239	0.246131
31	0.969248	0.969252	0.000011	0.003383	0.246087
32	0.969259	0.969263	0.000011	0.003239	0.246044
33	0.969269	0.969273	0.000010	0.003088	0.246003

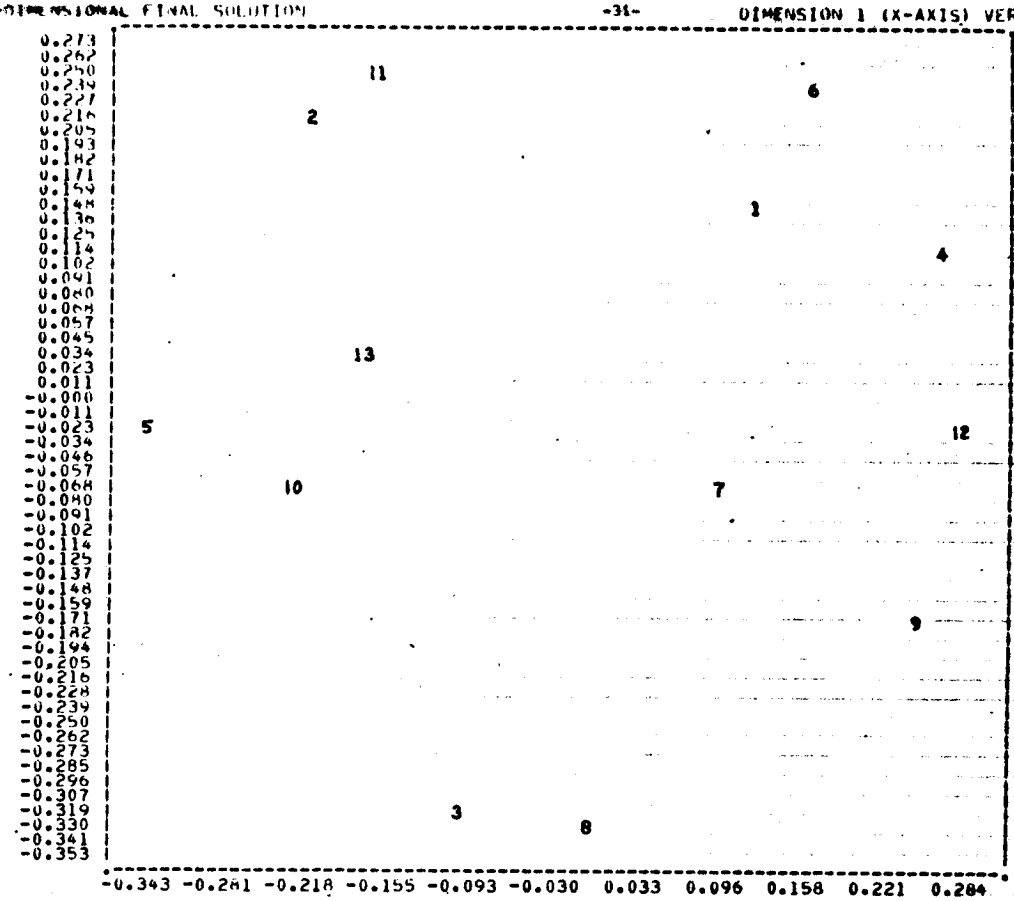
NO FURTHER IMPROVEMENT OF RHO: THE FINAL VALUES ARE:

34	0.969279	0.969282	0.000010	0.002930	0.245966

FINAL 2-DIMENSIONAL CONFIGURATION FOR ***** ETNIC-DATA:UNWEIGHTED3.2 *****

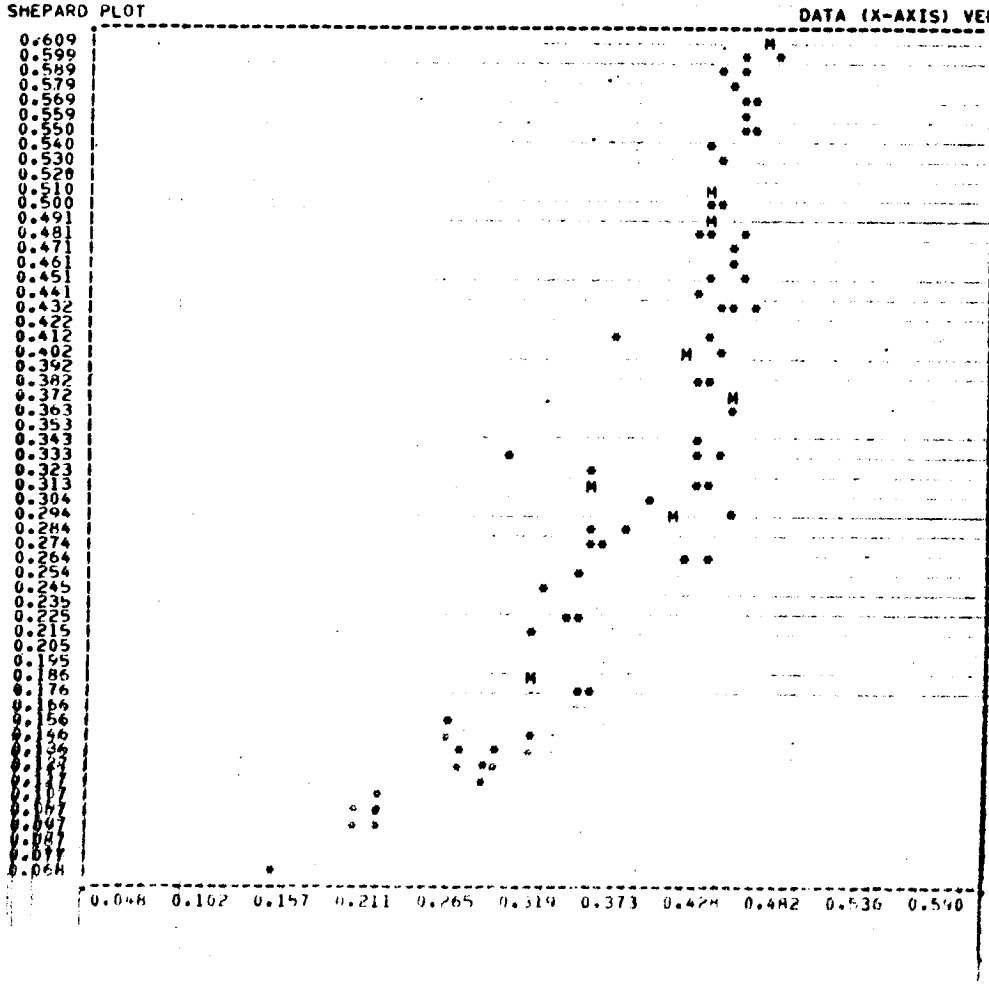
	1	2
1	0.128	0.152
2	-0.216	0.213
3	-0.094	-0.315
4	0.269	0.117
5	-0.343	-0.023
6	0.168	0.243
7	0.098	-0.074
8	0.000	-0.332
9	0.252	-0.171
10	-0.219	-0.069
11	-0.158	0.250
12	0.284	-0.024
13	-0.166	0.031

DIMENSION 1 (X-AXIS) VERSUS DIMENSION 2 (Y-AXIS)



SHEPARD PLOT

DATA (X-AXIS) VERSUS DISTANCES (Y-AXIS)



improve the fit considerably (stress going from 0.3046 to 0.1605) and that the resulting configuration does not show the "degeneracy" of the POLYCON solution. The same is true for the two-dimensional case, where a stress value of 0.2459 is obtained now.

We will not try to give an interpretation of the obtained configuration here. It is clear that a metric solution as determined by the SMACOF algorithm may improve a Torgerson-type solution considerably. The Shepard plots (of data versus distances) are much more regular than figure 7; they show a marked nonlinearity which suggest that a nonmetric program could improve the fit still further. In effect however, a nonmetric analysis starting with this configuration as input again ruins it until a three cluster solution with a two cluster Shepard plot is obtained (with a low stress value indeed).

3. Literature.

De Leeuw, J. Applications of convex analysis to multidimensional scaling. Barra, J.R. e.a. (eds), Recent Developments in Statistics. Amsterdam, North-Holland Publishing Company, 1977.

De Leeuw, J. and Heiser, W. Convergence of correction-matrix algorithms for multidimensional scaling. To appear in: Lingoes, J.C., Guttman, L. and Roskam, E. (Eds.), Geometric Representations of Relational Data, 1977 (in press).

EISPACK-guide. Matrix eigensystem Routines. Lecture Notes in Computer Science vol. 6. Berlin, Springer Verlag, 1974.

Funk, S.G., Horowitz, A.D., Lipshitz, R. and Young, F.W. The perceived structure of American ethnic groups: the use of multidimensional scaling in stereotype research. Personality and Social Psychology Bulletin, 1974, 1 (1), 66-68.

Gleason, T.C. A general model for nonmetric multidimensional scaling. Michigan mathematical psychology program, 1967, 3.

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.

Kruskal, J.B. Nonmetric multidimensional scaling: a numerical method. Psychometrika, 1964, 29, 115-129.

Kruskal, J.B. and Carroll, J.D. Geometrical models and badness of fit functions. Krishnaiah, P.R. (Ed), Multivariate analysis II. New York, Academic Press, 1969.

Kruskal, J.B. and Hart, R.E. A geometric interpretation of diagnostic data from a digital machine. Bell System Technical J., 1966, 45, 1299-1338.

Lingoes, J.C. and Roskam, E.E. A mathematical and empirical analysis of two multidimensional scaling algorithms. Psychometrika, 1973, 38, monograph supplement.

McGee, V.E. The multidimensional analysis of 'elastic' distances. Brit. J. Math. Statist. Psychol., 1966, 19, 181-196.

Roskam, E.E. Multidimensional Scaling by metric transformation of data. Ned. Tijdschrift voor de Psychologie, 1972, 27, 486-508.

Shepard, R.N. The analysis of proximities: Multidimensional Scaling with an unknown distance function I. Psychometrika, 1962, 27, 125-140.

Torgerson, W. Theory and Methods of Scaling. New York, Wiley, 1958.

Young, F.W. A model for polynomial conjoint analysis algorithms. Shepard, R.N. e.a. (Eds) Multidimensional Scaling: Theory and applications in the behavioural sciences. New York, Academic Press 1972.

Young, F.W. Conjoint Scaling: POLYCON user's manual. Chapel Hill, North Carolina: L.L. Thurstone Psychometric Laboratory Report No 118, 1973.

4. Job Setup.

CARD 1 : TITLE CARD

col.	format	information
1 - 80	20A4	any alphameric information to title the printout

CARD 2 : DATA DESCRIPTION CARD

col.	format	information
1 - 5	I5	number of points (maximum 60)
6 - 10	I5	number of replications (unlimited; default = 1)
11 - 15	I5	symmetry of data (0 = symmetric; 1 = asymmetric)
16 - 20	I5	weights (0 = no weights; 1 = weights)
21 - 25	I5	missing data (0 = no; 1 = yes)
26 - 35	F10.4	value of missing data (all datavalues less than or equal to this value are interpreted as missing; default = 0.0)

CARD 3 : OUTPUT OPTIONS

col.	format	information
1 - 5	I5	printout of data 0 = no print of data 1 = print aggregated data only 2 = print original data only 3 = print both original and aggregated data
6 - 10	I5	plotting of final configuration 0 = no plot 1 = plot first dimension only 2 = plot first two dimensions k = plot (pairwise) k dimensions

11 - 15	I5	printing and plotting of initial configuration 0 = no print or plot 1 = plot first dimension only (and print) 2 = plot first two dimensions (" ") k = plot (pairwise) k dimensions (" ")
16 - 20	I5	history of computation 0 = no 1 = yes
21 - 25	I5	plot of data versus distances (shepard plot) 0 = no 1 = yes
26 - 30	I5	punch and store options (results are written to medium specified below) 0 = no punch/store 1 = final configuration punched/stored 2 = final distances punched/stored 3 = both matrices punched/stored k = if the initial configuration should be punched/stored, add 10 to the options above
31 - 35	I5	reference number of the input medium for the data (default = 5)
36 - 40	I5	reference number of the punch/store medium (default = 6)

CARD 4 : ANALYSIS CARD

col.	format	information
1 - 5	I5	maximum number of dimensions (default = 2)
6 - 10	I5	minimum number of dimensions (default = 2)
11 - 15	I5	maximum number of iterations (default = 50)

16 - 20	I5	initial configuration 0 = no 1 = provided by user
21 - 30	F10.9	convergence criterion (default = .00001)

CARD 5 : DATA FORMAT

col.	format	information
1 - 80	20A4	FORTRAN format information for the data matrices (F-format)

CARD 6 : WEIGHTS FORMAT (optionally)

col.	format	information
1 - 80	20A4	FORTRAN format information for the weight matrices (F-format)

FOLLOWING CARDS : DATA MATRICES

The data must appear here according to the format specified on card 5. If they are symmetric, only the lower triangular part is needed. If there are weights, these should follow the data matrix; in the case of replications, each datamatrix should be alternated with its weightmatrix.

FOLLOWING CARDS (optionally)

If an initial configuration is input, it should appear here, preceded by a format card. The program expects a n x p matrix, where p is the maximum number of dimensions specified on card 4.