MPRA
Munich Personal RePEc Archive

The most representative composite rank ordering of multi-attribute objects by the particle swarm optimization

Mishra, SK
North-Eastern Hill University, Shillong (India)

12. January 2009

The Most Representative Composite Rank Ordering of Multi-Attribute Objects by the Particle Swarm Optimization

SK Mishra
Department of Economics
North-Eastern Hill University
Shillong (India)
Contact: mishrasknehu@yahoo.com

I. Introduction: Consider n objects each with $m < n$ common attributes. Suppose that these attributes cannot be measured, yet the objects can be rank ordered according to each attribute. More often than not, different evaluators would rank order the objects differently on the basis of each attribute or criterion (or even a particular evaluator may rank order the objects differently in different sessions of evaluation). There may be a good deal of concordance among the ranking scores obtained by the objects on the different criteria and the different sessions, but, in general, the concordance would not be perfect. There will be a need to summarize the ranking scores obtained on varied individual attributes (criteria). The summary will be given by a single array of overall ordinal ranking scores, which would represent the detailed attribute-(criterion-) wise ordinal ranking scores.

II. Criterion of Representation: Among the many possible criteria to summarize the imperfectly concordant arrays of individual measures $(x_{ij} \in X_{(n,m)};\ i=1,2,...,n;\ \ j=1,2,...,m)$ into a single array $(z_i \in Z_{(n)}; i=1,2,...,n),$ the one is to obtain Z such that the sum of squared (product moment) coefficients of correlation between the composite array of ranking scores, $Z,$ with the individual arrays of ranking scores, $x_j \in X,$ is maximum. Or, in other words, $\sum_{j=1}^{m} r^2(Z, x_j)$ is maximum. It may be noted that this criterion also minimizes the (Euclidean) distance between Z and X such that Z passes through the center of the swarm of points in X. The product moment coefficient of correlation incorporates Spearman's coefficient of rank correlation as a special case.

III. The Conventional Principal Component Analytic Approach: However, as a matter of practice, Z is seldom found out so as to maximize $\sum_{j=1}^{m} r^2(Z, x_j).$ Instead, $Y = Xw$ that maximizes $\sum_{j=1}^{m} r^2(Y, x_j)$ is found out and, consequently, Y is rank ordered to obtain $Z = \Re(Y),$ where $\Re(Y)$ is the rule of rank ordering Y. In order to do this, the Principal Components Analysis (Hotelling, 1936; Kendall and Stuart, 1968) is used which essentially runs into five steps: (i) standardization of x_j to $u_j = (x_j - \bar{x}_j)/s_j\ \forall\ j=1,2,...,m$ where $\bar{x}_j$ and s_j are the arithmetic mean and the standard deviation of x_j respectively; (ii) obtaining $R = (1/n)U'U : u_j \in U$; (iii) obtaining the largest eigenvalue, $\lambda,$ and the associated eigenvector, $\omega,$ of R; (iv) normalizing ω such that $v_i = \omega_i / \kappa,$ where $\kappa = \left(\sum_{j=1}^{m} \omega_j^2\right)^{1/2}$ and finally, (v) obtaining $\Upsilon = Uv.$ Now, since the rank ordering of $Y = Xw$ that maximizes $\sum_{j=1}^{m} r^2(Y, x_j)$ is identical to the rank ordering obtained by $\Upsilon = Uv$, that is, $\Re(Y)$ and $\Re(\Upsilon)$ are identical, it is generally believed that rank ordering of objects on the basis of $Y = Xw$ or $\Upsilon = Uv$ best represents X. It may be shown, nevertheless, that $Z = \Re(Y)$ or

$Z=\Re(\Upsilon)$ does not necessarily maximize $\sum_{j=1}^{m} r^2(Z,x_j)$ and, thus, rank ordering based on the principal component analysis as described above is often sub-optimal (Mishra, 2008-b). This is obvious in view of the fact the $Z=\Re(Y)$ or $Z=\Re(\Upsilon)$ is not a linear function and consequently, Z may not inherit or preserve the optimality of Y (or Υ).

IV. The Ordinal Principal Component Approach: Korhonen (1984) and Korhonen and Siljamaki (1998) were perhaps the first attempts to directly obtain the overall ordinal ranking scores vector, Z, that maximizes $\sum_{j=1}^{m} r^2(Z,x_j)$. The authors named their method as the 'ordinal principal component analysis'. In so doing, they used the constrained integer programming as a method of optimization (Li and Li, 2004). It is obvious that their approach to obtain the solution (rank ordering) may fail or become inordinately arduous if the scheme of rank ordering (Wikipedia, 2008-a) is standard competition ranking (1-2-2-4 rule), modified competition ranking (1-3-3-4 rule), dense ranking (1-2-2-3 rule) or fractional ranking (1-2.5-2.5-4 rule). This is so because the formulation of constraints in the integer programming problem with any ranking scheme other than the ordinal ranking (1-2-3-4 rule) would be extremely difficult or impracticable.

V. Objectives of the Present Work: In this paper we propose a new method to obtain Z that maximizes $\|r(Z,x)\|_L$ irrespective of the choice of rank ordering scheme; it may be standard competition, modified competition, dense, fractional or ordinal. The matrix X may incorporate ordinal or cardinally measured variables. The norm, $\|r(Z,x)\|_L$ could be absolute (L=1, maximizing $\sum_{j=1}^{m} |r(Z,x_j)|$), Euclidean (L=2, maximizing $\left[\sum_{j=1}^{m} r^2(Z,x_j)\right]^{1/2}$ or, by implication its square, $\sum_{j=1}^{m} r^2(Z,x_j)$) or maximin (L=$\infty$, $\max(\min_j(|r(Z,x_j)|))$) or any other Minkowsky's norm. The coefficient of correlation may be computed by Karl Pearson's formula (of which the Spearman's formula is only a special case) or Bradley's formula of absolute correlation (Bradley, 1985). Different measures of norm as well as correlation may have different implications as well as applications.

VI. The Method of Optimization: A choice of method of optimization depends much on the objective function (and the constraints, if any). We have proposed a very general objective function that may be smooth, kinky or even abruptly changing, depending on the norm chosen. Further, nonlinear functions such as computation of correlation coefficient and rank ordering are imbedded in the objective function. In view of these complications, we have chosen the (Repulsive) Particle Swarm method of optimization.

VI(i). The Particle Swarm Optimizer: The Particle Swarm method of optimization (Eberhart and Kennedy, 1995) is a population-based, stochastic search method that does not require derivatives of the optimand function to be computed. This method is also an instance of a successful application of the philosophy of decentralized decision-making and *bounded rationality* to solve the global optimization problems (Hayek, 1948, 1952; Simon, 1982; Bauer, 2002; Fleischer, 2005). It is observed that a swarm of birds or insects or a school of fish searches for food, protection, etc. in a very typical manner. If one of the members of the swarm sees a desirable path to go, the others in the swarm will follow it. Every member of the swarm searches for the best in its locality - learns from its own experience. Additionally, each member learns from the others, typically from the best performer among them. Even human beings show a tendency to learn from their own experience, their immediate neighbours and the ideal performers.

The Particle Swarm method of optimization mimics the said behaviour (Wikipedia, 2008-c). Every individual of the swarm is considered as a particle in a multidimensional space that has a position and a velocity. These particles fly through hyperspace and remember the best position that they have seen. Members of a swarm communicate good positions to each other and adjust their own position and velocity based on these good positions. There are two main ways this communication is done: (i) "swarm best" that is known to all (ii) "local bests" are known in neighborhoods of particles. Updating the position and velocity is done at each iteration as follows:

$$v_{i+1} = \omega v_i + c_1 r_1 (\hat{x}_i - x_i) + c_2 r_2 (\hat{x}_{gi} - x_i)$$
$$x_{i+1} = x_i + v_{i+1}$$

where,

- x is the position and v is the velocity of the individual particle. The subscripts i and $i+1$ stand for the recent and the next (future) iterations, respectively.
- ω is the inertial constant. Good values are usually slightly less than 1.
- c_1 and c_2 are constants that say how much the particle is directed towards good positions. Good values are usually right around 1.
- r_1 and r_2 are random values in the range [0,1].
- $\hat{x}$ is the best that the particle has seen.
- $\hat{x}_g$ is the global best seen by the swarm. This can be replaced by $\hat{x}_L$, the local best, if neighborhoods are being used.

The Particle Swarm method has many variants. The Repulsive Particle Swarm (RPS) method of optimization (Urfalioglu, 2004), the one of such variants, is particularly effective in finding out the global optimum in very complex search spaces (although it may be slower on certain types of optimization problems). Other variants use a dynamic scheme (Liang and Suganthan, 2005). In the traditional RPS the future velocity, v_{i+1} of a particle at position with a recent velocity, v_i, and the position of the particle are calculated by:

$$v_{i+1} = \omega v_i + \alpha r_1 (\hat{x}_i - x_i) + \omega \beta r_2 (\hat{x}_{hi} - x_i) + \omega \gamma r_3 z$$
$$x_{i+1} = x_i + v_{i+1}$$

where,

- x is the position and v is the velocity of the individual particle. The subscripts i and $i+1$ stand for the recent and the next (future) iterations, respectively.
- r_1, r_2 r_3 are random numbers, [0,1]
- ω is inertia weight, [0.01,0.7]
- $\hat{x}$ is the best position of a particle
- x_h is best position of a randomly chosen other particle from within the swarm
- z is a random velocity vector

- $\alpha, \ \beta, \ \gamma$ are constants

Occasionally, when the process is caught in a local optimum, some *chaotic* perturbation in position as well as velocity of some particle(s) may be needed.

VI(ii). Memetic Modifications in the RPS Method: The traditional RPS gives little scope of local search to the particles. They are guided by their past experience and the communication received from the others in the swarm. We have modified the traditional RPS method by endowing stronger (wider) local search ability to each particle. Each particle flies in its local surrounding and searches for a better solution. The domain of its search is controlled by a new parameter. This local search has no preference to gradients in any direction and resembles closely to tunneling. This added exploration capability of the particles brings the RPS method closer to what we observe in real life. However, in some cases moderately wide search works better. This local search capability endowed to the individual members of the swarm makes the RPS somewhat memetic (in the sense of Dawkins, 1976 and Ong et al., 2006).

It has been said that each particle learns from its 'chosen' inmates in the swarm. Now, at the one extreme is to learn from the best performer in the entire swarm. This is how the particles in the original PS method learn. However, such learning is not natural. How can we expect the individuals to know as to the best performer and interact with all others in the swarm? We believe in limited interaction and limited knowledge that any individual can possess and acquire. So, our particles do not know the 'best' in the swarm. Nevertheless, they interact with some chosen inmates that belong to the swarm. Now, the issue is: how does the particle choose its inmates? One of the possibilities is that it chooses the inmates closer (at lesser distance) to it. But, since our particle explores the locality by itself, it is likely that it would not benefit much from the inmates closer to it. Other relevant topologies are : (the celebrated) *ring topology*, ring topology hybridized with random topology, star topology, von Neumann topology, etc.

Let us visualize the possibilities of choosing (a predetermined number of) inmates randomly from among the members of the swarm. This is much closer to reality in the human world. When we are exposed to the mass media, we experience this. Alternatively, we may visualize our particles visiting a public place (e.g. railway platform, church, etc) where it (he) meets people coming from different places. Here, geographical distance of an individual from the others is not important. Important is how the experiences of others are communicated to us. There are large many sources of such information, each one being selective in what it broadcasts and each of us selective in what we attend to and, therefore, receive. This selectiveness at both ends transcends the geographical boundaries and each one of us is practically exposed to randomized information. Of course, two individuals may have a few common sources of information. We have used these arguments in the scheme of dissemination of others' experiences to each individual particle. Presently, we have assumed that each particle chooses a pre-assigned number of inmates (randomly) from among the members of the swarm. However, this number may be randomized to lie between two pre-assigned limits.

VII. A Formal Description of the Problem: Now we formally describe our problem of rank ordering the individuals characterized by X as follows:

Maximize,

$f_1 = |r(Z,x_1)| + |r(Z,x_2)| + \ ... \ + |r(Z,x_m)|$ or

$f_2 = r^2(Z,x_1) + r^2(Z,x_2) + \ ... \ + r^2(Z,x_m)$ or

$f_s = \max[\min |r(Z,x_1)|, |r(Z,x_2)|, \ ... \ , |r(Z,x_m)|]$,

whichever the choice may be, such that Z is an array of ranking scores obtained by the individuals described by X, following a suitable scheme of rank ordering (such as the standard competition ranking, the dense ranking, or the ordinal ranking, etc) and the correlation function, $r(Z,x_j)$, is computed by a suitable formula (Karl Pearson's product moment or Bradley's absolute correlation). It is obvious that the optimand objective function f_1, f_2 or f_s is defined in terms of two procedures: (i) $Z \leftarrow X$, and (ii) $r \leftarrow (Z,x_j)$. In this sense, the optimand function is unusual and involves logico-arithmetic operations rather than simple arithmetic operations. This is unlike the formulation by Korhonen and Siljamaki (1998) who, by means of imposing constraints on the elements of Z, could convert the problem of optimization into a purely arithmetic procedure.

VIII. A Computer Program: We have developed a computer program (in FORTRAN) to solve the problem. It consists of a main program and 13 subroutines. The subroutines RPS, LSRCH, NEIGHBOR, FSELECT, RANDOM and FUNC are used for the purpose of optimization. The subroutine GINI computes the degree of diversity in the swarm population on reaching the optimal solution by some members of the swarm. Other subroutines relate to rank ordering (DORANK) and computing the coefficient of correlation. In particular, the subroutine CORA computes Bradley's absolute correlation (Bradley, 1985; Mishra, 2008-a). The parameters NOB and MVAR (no. of observations, n, and no. of variables, m, in $X_{(n,m)}$) need to be specified in the main program as well as in the subroutine CORD. In the subroutine DORANK the scheme of rank ordering should be specified (whether rank ordering is to be done by 1-2-3-4 rule, 1-2-2-4 rule, 1-3-3-4 rule, 1-2-2-3 rule or 1-2.5-2.5-4 rule). Presently, it is set to NRL=0 for the ordinal (1-2-3-4 ranking) rule. Parameters in other programs usually do not need re-specification. However, necessary comments have been given to change them if so needed in very special conditions.

IX. Three Examples of Sub-optimality of the PCA-based Rank ordering: In order to illustrate the method of the most representative composite rank ordering suggested by us, the program developed for the same purpose, and the superiority of our method to the PCA-based rank ordering, we present three examples. All the three examples are simulated by us. Notation-wise, we use $Y = Xw$ and $Z_1 = \Re(Y)$ obtained by maximization of $\sum_{j=1}^{m} r^2(Y,x_j)$ resulting into the PCA-based ranking scores, Z_1. Analogously, we use $Y' = Xv$ and $Z_2 = \Re(Y')$ obtained by maximization of $\sum_{j=1}^{m} r^2(Z_2,x_j)$ resulting into the most optimal ranking scores, Z_2, proposed by us in this paper. These examples clearly demonstrate that the PCA-based Z_1 is sub-optimal.

IX(i). Example-1: The simulated dataset (X) on ranking scores of 30 candidates awarded by 7 evaluators, the results obtained by running the principal component algorithm (PCA) and the overall rankings based on the same (Y and Z_1) and the results of rank order optimization exercise based on our method (Y' and Z_2) are presented in Table-1.1. In table-1.2 are presented the inter-correlation matrix, R_1, for the variables $[Z_1\ x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7]$. The last two rows of Table-1.2 are the weight (w) vector used to obtain $Y = Xw$ and component loadings, that is, $r(Y, x_j)$. The sum of squared component loadings (S_1) = 4.352171. The measure of representativeness of Z_1 that is $F_1 = \sum_{j=1}^{7} r^2(Z_1, x_j)$ =4.287558. All these results pertain to the standard PCA, obtained by direct optimization. These results compare perfectly with those obtained by STATISTICA, a standard statistical software, that uses the conventional singular value decomposition method to obtain the PCA-based component scores.

In Table-1.3 we have presented the inter-correlation matrix, R_2, for variables $[Z_2\ x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7]$, weights and the component loadings when the same dataset (as mentioned above) is subjected to the direct maximization of $\sum_{j=1}^{7} r^2(Z_2, x_j)$. The weights and the component loadings relate to $Y' = Xv$ and $r(Z_2, x_j)$. The sum of squared component loadings (S_2)= 4.287902 and the measure of representativeness of Z_2 that is $F_2 = \sum_{j=1}^{7} r^2(Z_2, x_j)$ also is 4.287902. Since $F_2 > F_1$, the sub-optimality of the PC-based F_1 for this dataset is demonstrated. Notably, the candidates #8, #20, #21 and #26 are rank ordered differently by the two methods. It may be noted that the changed rank ordering may mean a lot to the candidates.

IX(ii). Example-2: The simulated data and Y, Y', Z_1 and Z_2 for this dataset are presented in Table-2.1. The inter-correlation matrices, R_1 and R_2 and the associated weights and factor loadings also are presented in Tables-2.2 and 2.3. The values of F_1 and F_2 for this dataset are 2.610741 and 2.610967 respectively. This also shows the sub-optimality of the PC-based F_1. The candidates #2, #5, #12, #13, #14 and #30 are rank ordered differently by the two methods.

IX(iii). Example-3: One more simulated dataset and Y, Y', Z_1 and Z_2 for this dataset are presented in Table-3.1. The inter-correlation matrices, R_1 and R_2 and the associated weights and factor loadings also are presented in Tables-3.2 and 3.3. The values of F_1 and F_2 for this dataset are 4.476465 and 4.476555 respectively. Once again, it is demonstrated that the PC-based F_1 is sub-optimal. The candidates #22 and #26 are rank ordered differently by the two methods.

X. Two Examples of Overall Rank ordering by Maximization of the Absolute Norm: Earlier it has been mentioned that an overall composite rankings may also be obtained by maximization of $f_1 = |r(Z_2, x_1)| + |r(Z_2, x_2)| + \ldots + |r(Z_2, x_m)|$ which is only an analogous version of maximization of $f_2 = r^2(Z_2, x_1) + r^2(Z_2, x_2) + \ldots + r^2(Z_2, x_m)$. Similarly, analogous to the principal component based rank ordering scores $Z_1 = \Re(Y);\ Y = Xw$ obtained by maximization of $\sum_{j=1}^{m} r^2(Y, x_j)$, one may also obtain $Z_1'' = \Re(Y'');\ Y'' = X\upsilon$ by maximization of $\sum_{j=1}^{m} |r(Y'', x_j)|$. This exercise has been done here and two examples have been presented. Results of examples 4 and 5 are presented in the Tables 4.1 through 5.3. The solutions exhibit some robustness to large variations of scores obtained by different individuals. We also find that AF_1 (=$\sum_{j=1}^{m} |r(Y'', x_j)|$ yielding $Z_1'' = \Re(Y'');\ Y'' = X\upsilon$) in the Table 4.2 (and 5.2) and AF_2 (=$\sum_{j=1}^{m} |r(Z_2'', x_j)|$ yielding Z_2'', in which $Y''' = X\omega$ is instrumental to obtain $Z_2'' = \Re(Y''')$) in the Table 4.3 (and 5.3) are equal, although Z_1'' and Z_2'' rank order the objects differently (see objects #11 and #12 in Table 4.1 and objects #9 and #16 in Table 5.1). This equality suggests that maximization of the absolute norm yields multiple solutions. Absolute norm estimators often exhibit this property of multiple solutions (Wikipedia, 2008-b). In the sense of sum of squared component loadings (F_1 and F_2), Z_2'' performs better than Z_1'' in example 4, but worse in example 5, although this is a different matter altogether. Obviously, under such conditions, no clear conclusion can be drawn.

XI. An Example of Overall Rank ordering by Maximin Absolute Correlation Criterion: In Tables 6.1 through 6.3 we present the results of an exercise to obtaining the composite rank ordering on the basis of maximin (absolute) correlation. Such maximin correlation signifies the *floor* (lowest absolute) correlation that the individual ranking scores (X) may have with the overall composite ranking score. In table 6.1, Z_1^* is obtained by max(min($|r(Y^*, x_j)|$)) while Z_2^* is obtained by max(min($|r(Z_2^*, x_j)|$)). The maximin correlation for Z_1^* is 0.671190, smaller than the maximin correlation (0.673860) for Z_2^*. Once again, sub-optimality of Z_1^* is demonstrated. Representation of X by the composite ranking scores has been presented in Fig.-1. It may also be reported that in obtaining the overall rankings by maximin correlation, the optimization method (the RPS) is often caught in the local optimum trap and, hence, the program was run several times with different seeds for generating random numbers.

XII. Concluding Remarks: Rank-ordering of individuals or objects on multiple criteria has many important practical applications. A reasonably representative composite rank ordering of multi-attribute objects/individuals or multi-dimensional points is often obtained by the Principal Component Analysis, although much inferior but computationally convenient methods also are frequently used. However, such rank ordering – even the one based on the Principal Component Analysis – may not be optimal. This has been demonstrated by several numerical examples. To solve this problem, the Ordinal Principal Component Analysis was suggested some time back. However, this approach cannot deal with various types of alternative schemes of rank ordering, mainly due to its dependence on the method of solution by the constrained integer programming. In this paper we propose an alternative method of solution, namely by the Particle Swarm Optimization. A computer program in FORTRAN to solve the problem has also been provided. The suggested method is notably versatile and can take care of various schemes of rank ordering, norms and types or measures of correlation. The versatility of the method and its capability to obtain the most representative composite rank ordering of multi-attribute objects or multi-dimensional points have been demonstrated by several numerical examples. It has also been found that rank ordering based on maximization of the sum of absolute values of the correlation coefficients of composite rank scores with its constituent variables has robustness, but it may have multiple optimal solutions. Thus, while it solves the one problem, it gives rise to the other problem. On this consideration, rank ordering by optimization of the absolute norm cannot be readily prescribed. The overall ranking of objects by maximin correlation principle performs better if the composite rank scores are directly obtained by maximization of min($|r(Z_2^*, x_j)|$) rather than min($|r(Y^*, x_j)|$).

References

Bauer, J.M. (2002): “Harnessing the Swarm: Communication Policy in an Era of Ubiquitous Networks and Disruptive Technologies”, *Communications and Strategies*, 45.

Bradley, C. (1985) “The Absolute Correlation”, *The Mathematical Gazette*, 69(447): 12-17.

Dawkins, R. (1976) *The Selfish Gene*. Oxford University Press, Oxford.

Eberhart R.C. and Kennedy J. (1995): “A New Optimizer using Particle Swarm Theory”, *Proceedings Sixth Symposium on Micro Machine and Human Science:* 39–43. IEEE Service Center, Piscataway, NJ.

Fleischer, M. (2005): “Foundations of Swarm Intelligence: From Principles to Practice”, Swarming Network Enabled C4ISR, arXiv:nlin.AO/0502003 v1.

Hayek, F. A. (1948) *Individualism and Economic Order*, The University of Chicago Press, Chicago.

Hayek, F. A. (1952) *The Sensory Order: An Inquiry into the Foundations of Theoretical Psychology*, University of Chicago Press, Chicago.

Hotelling, H. (1936) “Relations Between Two Sets of Variates”, *Biometrica*, 28: 321-377.

Kendall, M.G. and Stuart, A. (1968): *The Advanced Theory of Statistics*, vol. 3, Charles Griffin & Co. London.

Korhonen, P. (1984) Ordinal Principal Component Analysis, *HSE Working Papers*, Helsinki School of Economics, Helsinki, Finland.

Korhonen, P. and Siljamaki, A. (1998) Ordinal Principal Component Analysis. Theory and an Application”, *Computational Statistics & Data Analysis,* 26(4): 411-424.

Li, J. and Li, Y. (2004) Multivariate Mathematical Morphology based on Principal Component Analysis: Initial Results in Building Extraction”, http://www.cartesia.org/geodoc/isprs2004/comm7/papers/223.pdf

Liang, J.J. and Suganthan, P.N. (2005) “Dynamic Multi-Swarm Particle Swarm Optimizer”, *International Swarm Intelligence Symposium*, IEEE # 0-7803-8916-6/05/$20.00. : 124-129.

Mishra, S.K. (2006) “Global Optimization by Differential Evolution and Particle Swarm Methods: Evaluation on Some Benchmark Functions”, available at SSRN: http://ssrn.com/abstract=933827

Mishra, S. K. (2008-a) “On Construction of Robust Composite Indices by Linear Aggregation”, available at SSRN: http://ssrn.com/abstract=1147964

Mishra, S. K. (2008-b) “A Note on the Sub-Optimality of Rank Ordering of Objects on the Basis of the Leading Principal Component Factor Scores”, available at http://ssrn.com/abstract=1321369

Ong Y. S., Lim M. H., Zhu N. and Wong K. W. (2006). "Classification of Adaptive Memetic Algorithms: A Comparative Study". *IEEE Transactions on Systems Man and Cybernetics -- Part B.* **36** (1): 141-152.

Shevlyakov, G.L. (1997) “On Robust Estimation of a Correlation Coefficient”, *Journal of Mathematical Sciences*, 83(3): 434-438.

Simon, H.A.(1982): *Models of Bounded Rationality*, Cambridge Univ. Press, Cambridge, MA.

Spearman, C. (1904) "The Proof and Measurement of Association between Two Things", *American. Journal of Psychology*, 15: 88-93.

Urfalioglu, O. (2004) "Robust Estimation of Camera Rotation, Translation and Focal Length at High Outlier Rates", *Proceedings of the 1st Canadian Conference on Computer and Robot Vision*, IEEE Computer Society Washington, DC, USA: 464 – 471.

Wikipedia (2008-a) "Ranking", available at Wikipedia http://en.wikipedia.org/wiki/Rank_order

Wikipedia (2008-b) "Least absolute deviations": http://en.wikipedia.org/wiki/Least_absolute_deviations

Wikipedia (2008-c) "Particle Swarm Optimization", available at Wikipedia http://en.**wikipedia**.org/wiki/Particle_**swarm**_optimization

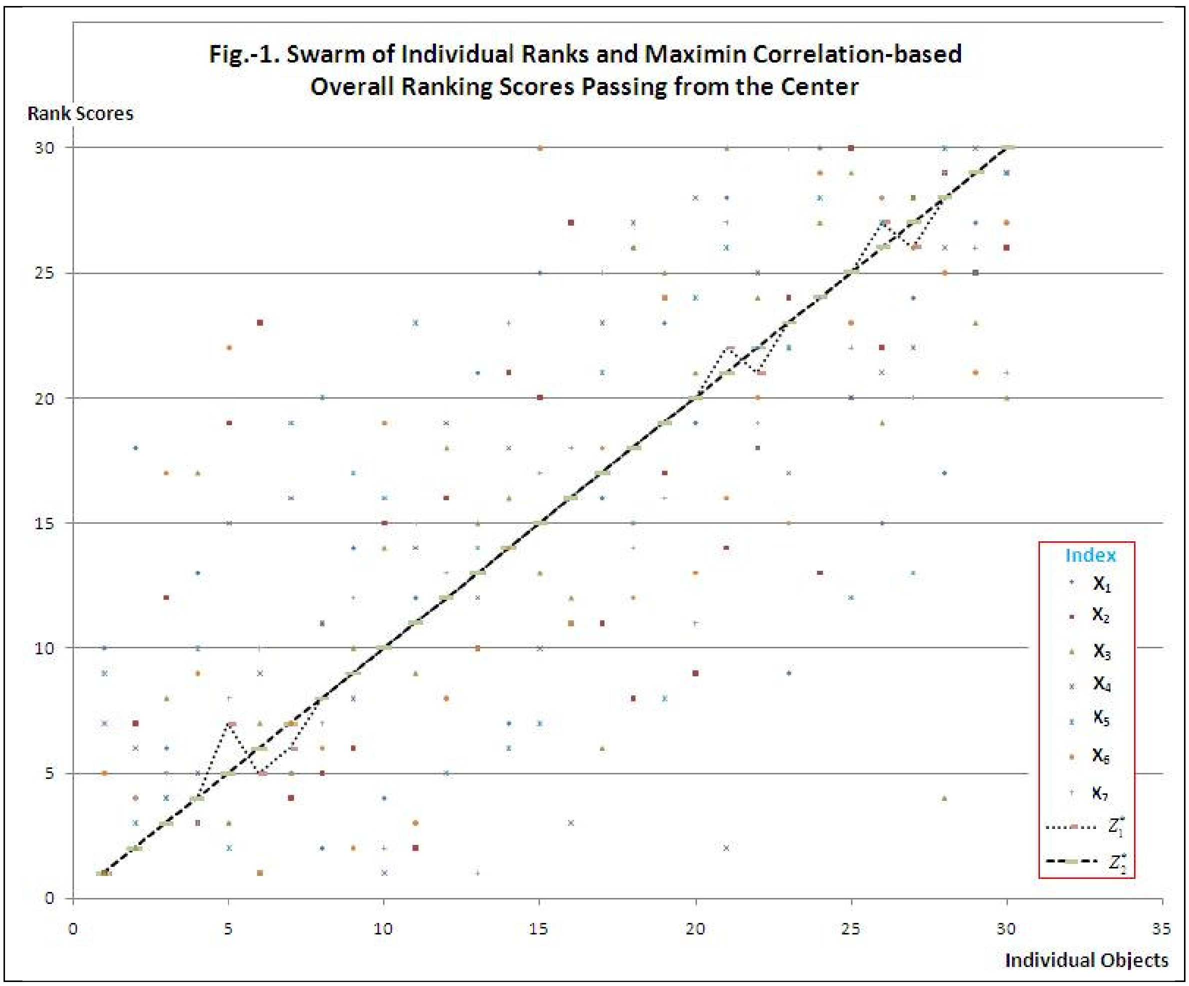

Table-1.1: Dataset Relating to Example-1 Showing Sub-optimality of PC-based Rank-ordering of Objects											
Sl. No.	Ranking Scores of 30 candidates awarded by Seven Evaluators							Composite Score (Y) Optimized Results		Rank-Order (Z_2) Optimized Results	
	X_1	X_2	X_3	X_4	X_5	X_6	X_7	Y	Z_1	Y'	Z_2
1	1	10	3	1	1	6	8	11.22449	3	9.94513	3
2	4	9	12	14	11	5	1	21.21789	5	22.02743	5
3	28	18	20	25	27	15	30	61.89772	26	60.65756	26
4	23	29	15	18	30	17	29	60.44523	25	57.66703	25
5	11	19	18	26	20	23	26	54.19009	22	52.54342	22
6	26	27	28	24	29	28	18	67.29577	28	65.48724	28
7	18	25	30	21	16	18	24	57.61014	24	56.18663	24
8	8	16	9	15	15	27	12	37.83847	12	35.71465	11
9	5	21	26	23	23	9	15	46.22725	19	46.03365	19
10	16	17	11	16	14	20	19	42.55687	16	40.82902	16
11	22	15	21	20	17	19	13	47.86438	20	47.36776	20
12	25	12	22	22	19	30	21	56.74538	23	54.95989	23
13	15	23	16	27	10	8	14	43.60463	17	44.80203	17
14	21	4	25	9	22	16	16	42.06405	15	40.18355	15
15	24	26	27	28	13	29	25	65.25869	27	63.94859	27
16	29	24	29	30	21	24	28	70.25945	30	69.24504	30
17	3	8	13	8	3	13	17	24.69311	7	23.04507	7
18	12	30	14	12	12	14	11	39.33113	14	37.96445	14
19	14	1	5	3	2	1	9	13.52052	4	13.35966	4
20	17	5	7	17	8	26	20	37.82411	11	36.15267	12
21	2	3	1	2	4	12	4	10.12458	2	8.83153	1
22	20	28	8	13	25	21	23	51.38818	21	48.22206	21
23	10	7	6	7	9	22	5	24.16536	6	22.50172	6
24	9	14	17	5	18	2	2	24.73648	8	24.29566	8
25	30	20	23	19	6	11	6	43.85996	18	45.06863	18
26	6	2	2	4	7	3	3	10.08922	1	9.92610	2
27	13	13	10	11	28	7	22	38.94103	13	36.86430	13
28	19	6	19	6	5	10	7	27.10137	10	26.67865	10
29	27	22	24	29	26	25	27	68.06338	29	66.67034	29
30	7	11	4	10	24	4	10	26.03013	9	25.10108	9

Table-1.2: Inter-Correlation Matrix, Weights and Component Loadings of Composite Score Optimized Overall Ranking Scores for the Dataset in Example-1 (F_1=4.287558; S_1= 4.352171)								
	Z_1	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_1	1.000000	**0.805117**	**0.781980**	**0.801112**	**0.891880**	**0.690768**	**0.658287**	**0.824694**
X_1	**0.805117**	1.000000	0.474082	0.666741	0.645384	0.451390	0.537709	0.597330
X_2	**0.781980**	0.474082	1.000000	0.554616	0.688543	0.552392	0.409121	0.569299
X_3	**0.801112**	0.666741	0.554616	1.000000	0.731257	0.438487	0.426919	0.491880
X_4	**0.891880**	0.645384	0.688543	0.731257	1.000000	0.526140	0.606229	0.708120
X_5	**0.690768**	0.451390	0.552392	0.438487	0.526140	1.000000	0.324583	0.630256
X_6	**0.658287**	0.537709	0.409121	0.426919	0.606229	0.324583	1.000000	0.608009
X_7	**0.824694**	0.597330	0.569299	0.491880	0.708120	0.630256	0.608009	1.000000
Weights		0.381529	0.369988	0.377232	0.430731	0.337587	0.337887	0.401968
Loadings		0.796004	0.771847	0.786981	0.898610	0.704269	0.704900	0.838500

Table-1.3: Inter-Correlation Matrix, Weights and Component Loadings of Rank Order Optimized Overall Ranking Scores for the Dataset in Example-1 (F_2=4.287902; S_2=4.287902)								
	Z_2	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_2	1.000000	**0.810901**	**0.776641**	**0.800667**	**0.893660**	**0.688988**	**0.653838**	**0.827809**
X_1	**0.810901**	1.000000	0.474082	0.666741	0.645384	0.451390	0.537709	0.597330
X_2	**0.776641**	0.474082	1.000000	0.554616	0.688543	0.552392	0.409121	0.569299
X_3	**0.800667**	0.666741	0.554616	1.000000	0.731257	0.438487	0.426919	0.491880
X_4	**0.893660**	0.645384	0.688543	0.731257	1.000000	0.526140	0.606229	0.708120
X_5	**0.688988**	0.451390	0.552392	0.438487	0.526140	1.000000	0.324583	0.630256
X_6	**0.653838**	0.537709	0.409121	0.426919	0.606229	0.324583	1.000000	0.608009
X_7	**0.827809**	0.597330	0.569299	0.491880	0.708120	0.630256	0.608009	1.000000
Weights		0.406915	0.347111	0.382829	0.561247	0.295479	0.250909	0.319554
Loadings		0.810901	0.776641	0.800667	0.89366	0.688988	0.653838	0.827809

Table-2.1: Dataset Relating to Example-2 Showing Sub-optimality of PC-based Rank-ordering of Objects											
Sl. No.	Ranking Scores of 30 candidates awarded by Seven Evaluators							Composite Score (Y) Optimized Results		Rank-Order (Z_2) Optimized Results	
	X_1	X_2	X_3	X_4	X_5	X_6	X_7	Y	Z_1	Y'	Z_2
1	6	9	3	12	1	3	11	17.70085	2	17.40745	2
2	25	17	19	23	18	30	7	49.33984	20	50.79072	21
3	1	11	6	15	11	29	21	32.42039	13	32.71282	13
4	12	15	15	27	9	20	20	44.63941	18	44.92521	18
5	20	26	27	17	15	28	18	53.54956	24	54.44562	25
6	8	7	12	4	22	9	14	27.13537	6	26.65443	6
7	21	10	24	19	13	19	22	48.77574	19	49.11107	19
8	27	14	22	25	28	15	30	61.45677	29	60.82086	29
9	24	6	14	10	8	17	13	34.06021	14	34.68700	14
10	4	1	11	14	10	8	3	19.95753	4	20.30134	4
11	29	25	10	21	7	26	28	52.68741	23	52.95060	23
12	13	28	17	29	20	10	26	53.88032	25	53.03821	24
13	23	23	1	9	3	25	10	30.50765	11	31.42658	12
14	22	5	28	18	5	23	27	49.81567	21	50.44069	20
15	10	21	26	26	17	11	25	52.34680	22	51.87400	22
16	16	24	21	28	14	16	29	56.14449	26	55.76413	26
17	28	22	30	30	30	5	16	62.14490	30	61.80191	30
18	3	27	4	6	12	21	15	27.59234	8	27.59011	8
19	11	3	16	5	19	24	4	27.70509	9	28.74193	9
20	17	13	23	8	29	13	2	36.86148	15	37.34405	15
21	19	20	25	22	27	27	23	58.98898	28	59.27033	28
22	9	29	13	2	6	4	17	27.57817	7	26.98883	7
23	5	30	2	20	26	22	24	43.75418	17	43.00494	17
24	18	16	5	11	16	14	6	29.43173	10	29.71901	10
25	30	12	29	24	25	7	19	57.21300	27	56.99428	27
26	2	19	7	1	4	2	12	15.97747	1	15.43028	1
27	14	18	20	13	23	18	8	39.98784	16	40.46898	16
28	7	8	8	16	2	6	1	18.71262	3	19.19911	3
29	15	2	18	7	21	1	5	26.77338	5	26.62086	5
30	26	4	9	3	24	12	9	30.68758	12	30.74115	11

Table-2.2: Inter-Correlation Matrix, Weights and Component Loadings of Composite Score Optimized Overall Ranking Scores for the Dataset in Example-2 (F_1=2.610741; S_1= 2.656011)								
	Z_1	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_1	1.000000	**0.649833**	**0.389989**	**0.703226**	**0.812236**	**0.510567**	**0.384650**	**0.688098**
X_1	**0.649833**	1.000000	-0.014905	0.519021	0.355729	0.327697	0.209344	0.205339
X_2	**0.389989**	-0.014905	1.000000	-0.039822	0.314794	0.054060	0.216463	0.471858
X_3	**0.703226**	0.519021	-0.039822	1.000000	0.511902	0.477642	0.022914	0.305451
X_4	**0.812236**	0.355729	0.314794	0.511902	1.000000	0.253393	0.209789	0.607119
X_5	**0.510567**	0.327697	0.054060	0.477642	0.253393	1.000000	0.009121	0.070078
X_6	**0.384650**	0.209344	0.216463	0.022914	0.209789	0.009121	1.000000	0.277419
X_7	**0.688098**	0.205339	0.471858	0.305451	0.607119	0.070078	0.277419	1.000000
Weights		0.389271	0.245369	0.444693	0.503508	0.309999	0.225894	0.435731
Loadings		0.634388	0.399833	0.724691	0.820595	0.505245	0.368176	0.710153

Table-2.3: Inter-Correlation Matrix, Weights and Component Loadings of Rank Order Optimized Overall Ranking Scores for the Dataset in Example-2 (F_2=2.610967; S_2=2.610967)								
	Z_2	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_2	1.000000	**0.652948**	**0.402892**	**0.700111**	**0.811791**	**0.504783**	**0.401557**	**0.676085**
X_1	**0.652948**	1.000000	-0.014905	0.519021	0.355729	0.327697	0.209344	0.205339
X_2	**0.402892**	-0.014905	1.000000	-0.039822	0.314794	0.054060	0.216463	0.471858
X_3	**0.700111**	0.519021	-0.039822	1.000000	0.511902	0.477642	0.022914	0.305451
X_4	**0.811791**	0.355729	0.314794	0.511902	1.000000	0.253393	0.209789	0.607119
X_5	**0.504783**	0.327697	0.054060	0.477642	0.253393	1.000000	0.009121	0.070078
X_6	**0.401557**	0.209344	0.216463	0.022914	0.209789	0.009121	1.000000	0.277419
X_7	**0.676085**	0.205339	0.471858	0.305451	0.607119	0.070078	0.277419	1.000000
Weights		0.401917	0.239038	0.466325	0.507664	0.283960	0.277857	0.385103
Loadings		0.652948	0.402892	0.700111	0.811791	0.504783	0.401557	0.676085

Table-3.1: Dataset Relating to Example-3 Showing Sub-optimality of PC-based Rank-ordering of Objects											
Sl. No.	Ranking Scores of 30 candidates awarded by Seven Evaluators							Composite Score (Y) Optimized Results		Rank-Order (Z_2) Optimized Results	
	X_1	X_2	X_3	X_4	X_5	X_6	X_7	Y	Z_1	Y’	Z_2
1	19	16	14	20	15	1	18	39.79121	15	39.47904	15
2	27	18	12	19	10	21	4	41.68016	18	41.95384	18
3	21	23	20	21	26	27	29	62.78029	25	62.64734	25
4	18	17	13	15	9	13	17	38.58541	14	38.72100	14
5	9	9	25	10	20	14	22	41.09776	17	41.26278	17
6	20	30	18	23	24	23	20	59.43680	23	59.13327	23
7	11	5	6	6	3	7	9	17.78085	4	18.01278	4
8	26	27	22	30	25	25	28	69.21167	28	68.86556	28
9	23	28	29	28	21	29	30	70.78323	30	70.76889	30
10	7	21	8	5	6	4	2	19.83920	6	19.95443	6
11	17	15	7	11	13	5	16	32.08370	12	32.03662	12
12	22	24	24	13	12	28	12	50.14432	20	50.90558	20
13	16	7	1	2	14	3	6	18.66655	5	18.81742	5
14	10	1	3	12	22	18	1	25.00260	8	24.48451	8

15	13	12	11	22	23	6	13	38.42074	13	37.58991	13
16	14	19	19	14	19	11	19	43.54547	19	43.50900	19
17	28	29	30	16	18	12	11	54.47542	22	55.15631	22
18	3	2	2	9	4	10	14	16.58316	3	16.29347	3
19	24	22	28	25	29	30	27	69.59663	29	69.55277	29
20	4	3	17	18	5	8	3	22.13969	7	21.83190	7
21	25	25	26	24	28	26	25	67.44970	27	67.41736	27
22	1	4	15	1	7	9	5	15.47967	1	15.80932	2
23	5	20	5	8	11	17	7	26.89428	11	26.71896	11
24	29	26	23	26	27	20	26	67.11005	26	66.98988	26
25	8	6	4	4	16	22	10	25.65787	10	25.67207	10
26	6	11	9	3	2	2	8	15.50943	2	15.78531	1
27	30	13	21	27	30	15	23	60.76023	24	60.44323	24
28	12	10	27	29	17	16	24	51.46019	21	50.94166	21
29	2	8	16	7	1	19	15	25.03077	9	25.36557	9
30	15	14	10	17	8	24	21	40.77648	16	40.76065	16

Table-3.2: Inter-Correlation Matrix, Weights and Component Loadings of Composite Score Optimized Overall Ranking Scores for the Dataset in Example-3 (F_1=4.476465; S_1= 4.558674)

	Z_1	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_1	1.000000	**0.824249**	**0.779755**	**0.798443**	**0.860289**	**0.797998**	**0.716574**	**0.813126**
X_1	**0.824249**	1.000000	0.720133	0.555951	0.710790	0.694327	0.453170	0.553281
X_2	**0.779755**	0.720133	1.000000	0.600445	0.557286	0.506563	0.494994	0.529255
X_3	**0.798443**	0.555951	0.600445	1.000000	0.675640	0.538598	0.509232	0.662291
X_4	**0.860289**	0.710790	0.557286	0.675640	1.000000	0.706785	0.517241	0.727697
X_5	**0.797998**	0.694327	0.506563	0.538598	0.706785	1.000000	0.492325	0.640489
X_6	**0.716574**	0.453170	0.494994	0.509232	0.517241	0.492325	1.000000	0.551947
X_7	**0.813126**	0.553281	0.529255	0.662291	0.727697	0.640489	0.551947	1.000000
Weights		0.391189	0.364617	0.377348	0.409665	0.381840	0.327169	0.388544
Loadings		0.835132	0.778517	0.805647	0.874764	0.815356	0.698526	0.829528

Table-3.3: Inter-Correlation Matrix, Weights and Component Loadings of Rank Order Optimized Overall Ranking Scores for the Dataset in Example-3 (F_2=4.476555; S_2=4.476555)

	Z_2	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_2	1.000000	**0.822024**	**0.776641**	**0.801112**	**0.859399**	**0.800222**	**0.719689**	**0.811791**
X_1	**0.822024**	1.000000	0.720133	0.555951	0.710790	0.694327	0.453170	0.553281
X_2	**0.776641**	0.720133	1.000000	0.600445	0.557286	0.506563	0.494994	0.529255
X_3	**0.801112**	0.555951	0.600445	1.000000	0.675640	0.538598	0.509232	0.662291
X_4	**0.859399**	0.710790	0.557286	0.675640	1.000000	0.706785	0.517241	0.727697
X_5	**0.800222**	0.694327	0.506563	0.538598	0.706785	1.000000	0.492325	0.640489
X_6	**0.719689**	0.453170	0.494994	0.509232	0.517241	0.492325	1.000000	0.551947
X_7	**0.811791**	0.553281	0.529255	0.662291	0.727697	0.640489	0.551947	1.000000
Weights		0.424073	0.364980	0.405664	0.360753	0.357984	0.337184	0.387815
Loadings		0.822024	0.776641	0.801112	0.859399	0.800222	0.719689	0.811791

Table-4.1: Dataset Relating to Example-4 Showing Rank-ordering by Maximization of the Absolute Norm

Sl. No.	Ranking Scores of 30 candidates awarded by Seven Evaluators							Composite Score (Y”) Optimized Results		Rank-Order (Z_2”) Optimized Results	
	X_1	X_2	X_3	X_4	X_5	X_6	X_7	Y”	Z_1”	Y”’	Z_2”
1	18	18	5	3	13	22	4	31.371434	10	31.62440	10
2	20	22	30	27	7	13	13	49.890877	22	50.03646	22
3	3	29	20	20	21	11	18	46.111595	19	46.88164	19
4	10	27	7	13	25	8	23	42.710587	18	43.15021	18
5	9	11	10	7	5	24	15	30.614628	8	30.17925	8
6	12	8	26	17	16	16	14	41.197770	15	40.99650	15
7	7	14	9	8	28	26	1	35.150913	13	35.71881	13
8	6	13	12	14	15	28	24	42.331403	17	41.78964	17
9	27	17	27	22	19	18	28	59.718366	24	59.15019	24
10	15	28	21	19	29	27	29	63.498033	29	63.56338	29
11	14	25	23	11	17	19	17	47.623564	20	47.99190	21
12	17	19	19	12	22	7	30	47.623968	21	47.44355	20
13	4	1	8	5	20	12	7	21.544108	5	21.47765	5
14	16	12	11	4	9	14	8	27.969578	7	27.95407	7
15	25	24	28	16	23	20	25	60.852521	26	60.81207	26
16	2	6	18	23	8	21	11	33.637782	11	33.29539	11
17	29	20	24	18	27	29	20	63.120288	28	62.91071	28
18	22	9	3	10	1	3	19	25.323880	6	24.59793	6
19	24	23	22	15	24	30	26	61.986236	27	61.71055	27
20	19	5	17	24	14	5	9	35.150701	12	34.87592	12
21	5	2	1	6	3	6	22	17.008182	3	16.16041	3
22	8	10	13	1	2	1	5	15.118738	2	15.36358	2
23	26	26	25	25	30	17	10	60.096859	25	60.72466	25
24	13	4	4	9	10	9	6	20.788207	4	20.54303	4
25	21	21	16	30	18	23	27	58.962115	23	58.44723	23
26	30	15	14	21	12	2	16	41.576617	16	41.30021	16
27	11	16	15	29	4	15	12	38.551624	14	38.39481	14
28	23	7	6	26	6	10	3	30.615030	9	30.24845	9
29	1	3	2	2	11	4	2	9.449315	1	9.62243	1
30	28	30	29	28	26	25	21	70.679438	30	70.88402	30

Table-4.2: Inter-Correlation Matrix, Weights and Component Loadings of Composite Score Optimized Overall Ranking Scores for the Dataset in Example-4

(AF_1=4.894105; AS_1= 4.93740575; F_1=3.487339)

	Z_1	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_1	1.000000	**0.594661**	**0.846496**	**0.820690**	**0.624027**	**0.738821**	**0.598220**	**0.671190**
X_1	**0.594661**	1.000000	0.426474	0.480311	0.478977	0.277864	0.164850	0.321913
X_2	**0.846496**	0.426474	1.000000	0.629366	0.410456	0.638265	0.441602	0.532369
X_3	**0.820690**	0.480311	0.629366	1.000000	0.596885	0.490545	0.422469	0.441602
X_4	**0.624027**	0.478977	0.410456	0.596885	1.000000	0.213793	0.220022	0.309900
X_5	**0.738821**	0.277864	0.638265	0.490545	0.213793	1.000000	0.519911	0.369077
X_6	**0.598220**	0.164850	0.441602	0.422469	0.220022	0.519911	1.000000	0.302336
X_7	**0.671190**	0.321913	0.532369	0.441602	0.309900	0.369077	0.302336	1.000000
Weights		0.377992	0.377978	0.377951	0.377940	0.377995	0.377938	0.377957
Loadings		0.638074	0.826052	0.822529	0.654188	0.710798	0.622017	0.663747

Table-4.3: Inter-Correlation Matrix, Weights and Component Loadings of Rank Order Optimized Overall Ranking Scores for the Dataset in Example-4 (AF_2=4.894105; AS_2=4.894105; F_2=3.488051)								
	Z_2	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_2	1.000000	**0.593326**	**0.849166**	**0.822469**	**0.623582**	**0.736596**	**0.603560**	**0.665406**
X_1	**0.593326**	1.000000	0.426474	0.480311	0.478977	0.277864	0.164850	0.321913
X_2	**0.849166**	0.426474	1.000000	0.629366	0.410456	0.638265	0.441602	0.532369
X_3	**0.822469**	0.480311	0.629366	1.000000	0.596885	0.490545	0.422469	0.441602
X_4	**0.623582**	0.478977	0.410456	0.596885	1.000000	0.213793	0.220022	0.309900
X_5	**0.736596**	0.277864	0.638265	0.490545	0.213793	1.000000	0.519911	0.369077
X_6	**0.603560**	0.164850	0.441602	0.422469	0.220022	0.519911	1.000000	0.302336
X_7	**0.665406**	0.321913	0.532369	0.441602	0.309900	0.369077	0.302336	1.000000
Weights		0.361685	0.420167	0.387054	0.370485	0.395085	0.363555	0.342504
Loadings		0.593326	0.849166	0.822469	0.623582	0.736596	0.603560	0.665406

Table-5.1: Dataset Relating to Example-5 Showing Rank-ordering by Maximization of the Absolute Norm											
Sl. No.	Ranking Scores of 30 candidates awarded by Seven Evaluators							Composite Score (Y”) Optimized Results		Rank-Order (Z_2”) Optimized Results	
	X_1	X_2	X_3	X_4	X_5	X_6	X_7	Y”	Z_1”	Y”’	Z_2”
1	20	30	18	14	12	9	14	44.22174	18	44.08011	18
2	1	12	4	17	3	5	6	18.14242	1	17.92374	1
3	26	28	26	7	27	30	29	65.38783	29	65.07686	29
4	27	19	6	10	11	16	20	41.19832	17	41.17978	17
5	3	14	9	3	4	15	1	18.52015	2	18.71988	2
6	7	17	2	5	2	19	2	20.41032	4	20.59862	4
7	4	11	27	16	28	18	26	49.13530	21	48.30841	21
8	30	25	28	28	19	10	30	64.25320	28	64.26996	28
9	17	4	1	25	6	6	18	29.10329	9	29.06091	8
10	25	8	24	21	18	26	19	53.29222	22	53.89539	22
11	6	9	7	4	8	13	12	22.29989	5	22.18983	5
12	11	18	30	20	25	27	27	59.71790	26	59.45911	26
13	14	20	16	6	13	12	16	36.66241	13	36.48242	13
14	8	29	20	30	30	23	17	59.34161	25	57.99207	25
15	22	23	10	29	17	24	25	56.69511	24	56.32758	24
16	16	15	5	11	1	14	15	29.10285	8	29.54333	9
17	5	6	12	9	10	8	4	20.41002	3	20.32964	3
18	2	1	8	15	5	28	5	24.18937	7	24.78023	7
19	12	24	21	18	29	21	24	56.31741	23	55.20101	23
20	23	16	3	12	9	22	11	36.28498	12	36.44815	12
21	24	7	11	22	16	25	21	47.62348	19	47.80678	19
22	28	26	13	19	26	20	28	60.47520	27	59.57479	27
23	13	22	17	8	7	4	9	30.23671	10	30.32081	10
24	21	13	19	24	21	17	13	48.37962	20	48.19456	20
25	9	21	23	2	20	3	22	37.79632	14	36.99930	14
26	19	10	25	13	22	11	8	40.82002	16	40.72036	16
27	18	2	15	1	23	1	3	23.81236	6	23.19690	6
28	10	5	22	27	24	7	7	38.55266	15	37.96188	15
29	15	3	14	26	14	2	10	31.74894	11	31.58722	11
30	29	27	29	23	15	29	23	66.14273	30	66.91077	30

Table-5.2: Inter-Correlation Matrix, Weights and Component Loadings of Composite Score Optimized Overall Ranking Scores for the Dataset in Example-5 (AF_1=4.648943; AS_1= 4.66456597; F_1=3.175427)								
	Z_1	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_1	1.000000	**0.644049**	**0.603560**	**0.703226**	**0.544383**	**0.739711**	**0.560845**	**0.853170**
X_1	**0.644049**	1.000000	0.362403	0.233370	0.296552	0.260512	0.261846	0.548832
X_2	**0.603560**	0.362403	1.000000	0.333482	0.057175	0.278309	0.305451	0.555506
X_3	**0.703226**	0.233370	0.333482	1.000000	0.240044	0.749944	0.185762	0.497664
X_4	**0.544383**	0.296552	0.057175	0.240044	1.000000	0.338821	0.241824	0.390434
X_5	**0.739711**	0.260512	0.278309	0.749944	0.338821	1.000000	0.236930	0.562625
X_6	**0.560845**	0.261846	0.305451	0.185762	0.241824	0.236930	1.000000	0.441602
X_7	**0.853170**	0.548832	0.555506	0.497664	0.390434	0.562625	0.441602	1.000000
Weights		0.377957	0.377988	0.377893	0.377968	0.378045	0.377960	0.377942
Loadings		0.635321	0.620066	0.694647	0.549860	0.734729	0.573131	0.856811

Table-5.3: Inter-Correlation Matrix, Weights and Component Loadings of Rank Order Optimized Overall Ranking Scores for the Dataset in Example-5 (AF_2=4.648943; AS_2=4.648943; F_2=3.159146)								
	Z_2	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_2	1.000000	**0.643604**	**0.608454**	**0.705006**	**0.538154**	**0.737486**	**0.564405**	**0.851835**
X_1	**0.643604**	1.000000	0.362403	0.233370	0.296552	0.260512	0.261846	0.548832
X_2	**0.608454**	0.362403	1.000000	0.333482	0.057175	0.278309	0.305451	0.555506
X_3	**0.705006**	0.233370	0.333482	1.000000	0.240044	0.749944	0.185762	0.497664
X_4	**0.538154**	0.296552	0.057175	0.240044	1.000000	0.338821	0.241824	0.390434
X_5	**0.737486**	0.260512	0.278309	0.749944	0.338821	1.000000	0.236930	0.562625
X_6	**0.564405**	0.261846	0.305451	0.185762	0.241824	0.236930	1.000000	0.441602
X_7	**0.851835**	0.548832	0.555506	0.497664	0.390434	0.562625	0.441602	1.000000
Weights		0.403896	0.357386	0.417812	0.377040	0.306707	0.401047	0.370822
Loadings		0.643604	0.608454	0.705006	0.538154	0.737486	0.564405	0.851835

Table-6.1: Dataset Relating to Example-6 Showing Rank-ordering by Maximization of Minimal Absolute Correlation											
Sl. No.	Ranking Scores of 30 candidates awarded by Seven Evaluators							Composite Score (Y^*) Optimized Results		Rank-Order (Z_2^*) Optimized Results	
	X_1	X_2	X_3	X_4	X_5	X_6	X_7	Y^*	Z_1^*	Y^{**}	Z_2^*
1	12	2	9	14	23	3	15	25.30634	11	25.88073	11
2	19	9	21	28	24	13	11	39.49625	20	38.73892	20
3	7	21	16	18	6	14	23	29.96293	14	31.40229	14
4	24	28	28	22	13	26	20	47.93919	27	48.81581	26
5	13	3	17	5	10	9	3	17.77573	4	17.45455	4
6	4	15	14	1	16	19	2	25.24309	10	25.82015	10
7	20	30	29	20	12	23	22	47.22933	25	48.32437	25
8	22	18	24	25	18	20	19	43.30421	22	43.75467	21
9	9	24	22	17	22	15	30	43.68190	23	45.84717	23
10	28	14	30	2	26	16	27	42.69905	21	44.95308	22
11	1	23	7	9	1	1	10	20.65061	6	21.18701	5
12	2	5	11	11	20	6	7	22.14594	8	22.18122	8
13	11	27	12	3	11	11	18	32.50658	16	34.47639	16

14	25	20	13	10	7	30	17	32.36392	15	34.27532	15
15	27	25	23	30	25	21	26	54.91396	29	55.97770	29
16	23	17	25	24	8	24	16	36.98198	19	37.19732	19
17	14	6	10	8	17	2	12	23.33585	9	23.87632	9
18	6	12	8	4	4	17	5	16.01656	3	16.75380	3
19	21	10	15	12	14	10	1	28.24829	13	27.48070	13
20	26	8	26	27	15	12	14	36.53673	18	35.78311	18
21	16	11	6	23	21	18	25	33.92366	17	35.75911	17
22	15	22	19	21	27	28	28	47.80047	26	50.10524	27
23	8	16	18	19	5	8	13	26.38564	12	26.33481	12
24	30	13	27	13	28	29	24	46.41054	24	48.29007	24
25	10	1	1	7	9	5	9	11.60210	1	12.27820	1
26	3	19	3	15	2	22	8	20.61575	5	21.48559	7
27	17	29	4	26	30	25	29	52.11630	28	54.82387	28
28	18	7	2	6	3	4	4	13.83704	2	13.95344	2
29	29	26	20	29	29	27	21	57.47630	30	58.46160	30
30	5	4	5	16	19	7	6	21.52460	7	21.39735	6

Table-6.2: Inter-Correlation Matrix, Weights and Component Loadings of Composite Score Optimized Overall Ranking Scores for the Dataset in Example-6
(Maximin Correlation=0.671190; Maximum Correlation=0.813126)

	Z_1	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_1	1.000000	**0.676085**	**0.673415**	**0.690323**	**0.684538**	**0.671190**	**0.700111**	**0.813126**
X_1	**0.676085**	1.000000	0.206229	0.636485	0.423359	0.442047	0.534594	0.451835
X_2	**0.673415**	0.206229	1.000000	0.353949	0.360178	0.070523	0.624472	0.614683
X_3	**0.690323**	0.636485	0.353949	1.000000	0.362848	0.324583	0.461624	0.454505
X_4	**0.684538**	0.423359	0.360178	0.362848	1.000000	0.403337	0.418465	0.493660
X_5	**0.671190**	0.442047	0.070523	0.324583	0.403337	1.000000	0.318354	0.552836
X_6	**0.700111**	0.534594	0.624472	0.461624	0.418465	0.318354	1.000000	0.550612
X_7	**0.813126**	0.451835	0.614683	0.454505	0.493660	0.552836	0.550612	1.000000
Weights		0.276166	0.652291	0.256872	0.269181	0.595985	0.044783	0.051025
Loadings		0.680300	0.680222	0.680243	0.680277	0.680222	0.714510	0.804249

Table-6.3: Inter-Correlation Matrix, Weights and Component Loadings of Rank Order Optimized Overall Ranking Scores for the Dataset in Example-6
(Maximin Correlation=0.673860; Maximum Correlation=0.820245)

	Z_2	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Z_2	1.000000	**0.674750**	**0.673860**	**0.686318**	**0.676085**	**0.673860**	**0.715239**	**0.820245**
X_1	**0.674750**	1.000000	0.206229	0.636485	0.423359	0.442047	0.534594	0.451835
X_2	**0.673860**	0.206229	1.000000	0.353949	0.360178	0.070523	0.624472	0.614683
X_3	**0.686318**	0.636485	0.353949	1.000000	0.362848	0.324583	0.461624	0.454505
X_4	**0.676085**	0.423359	0.360178	0.362848	1.000000	0.403337	0.418465	0.493660
X_5	**0.673860**	0.442047	0.070523	0.324583	0.403337	1.000000	0.318354	0.552836
X_6	**0.715239**	0.534594	0.624472	0.461624	0.418465	0.318354	1.000000	0.550612
X_7	**0.820245**	0.451835	0.614683	0.454505	0.493660	0.552836	0.550612	1.000000
Weights		0.269713	0.664980	0.215650	0.208353	0.603497	0.082781	0.155176
Loadings		0.674750	0.673860	0.686318	0.676085	0.673860	0.715239	0.820245

```
C      MAIN PROGRAM : PROVIDES TO USE REPULSIVE PARTICLE SWARM METHOD TO
C                        COMPUTE COMPOSITE INDEX INDICES
C      BY MAXIMIZING SUM OF (SQUARES, OR ABSOLUTES, OR MINIMUM) OF
C      CORRELATION OF THE INDEX WITH THE CONSTITUENT VARIABLES. THE MAX
C      SUM OF SQUARES IS THE PRINCIPAL COMPONENT INDEX. IT ALSO PRIVIDES
C      TO OBTAIN MAXIMUM ENTROPY ABSOLUTE CORRELATION INDICES.
C      PRODUCT MOMENT AS WELL AS ABSOLUTE CORRELATION (BRADLEY, 1985) MAY
C      BE USED. PROGRAM BY SK MISHRA, DEPT. OF ECONOMICS, NORTH-EASTERN
C      HILL UNIVERSITY, SHILLONG (INDIA)
C      -----------------------------------------------------------------
C          ADJUST THE PARAMETERS SUITABLY IN SUBROUTINES RPS
C      WHEN THE PROGRAM ASKS FOR PARAMETERS, FEED THEM SUITABLY
C      -----------------------------------------------------------------
       PROGRAM RPSINDEX
       PARAMETER(NOB=30,MVAR=7)!CHANGE THE PARAMETERS HERE AS NEEDED.
C      -----------------------------------------------------------------
C      NOB=NO. OF CASES AND MVAR=NO. OF VARIABLES
C      TO BE ADJUSTED IN SUBROUTINE CORD(M,X,F) ALSO: STATEMENT 931
       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
       COMMON /KFF/KF,NFCALL,FTIT ! FUNCTION CODE, NO. OF CALLS & TITLE
       CHARACTER *30  METHOD(1)
       CHARACTER *70 FTIT
       CHARACTER *40 INFILE,OUTFILE
       COMMON /CORDAT/CDAT(NOB,MVAR),QIND(NOB),R(MVAR),ENTROPY,NORM,NCOR
       COMMON /XBASE/XBAS
       COMMON /RNDM/IU,IV ! RANDOM NUMBER GENERATION (IU = 4-DIGIT SEED)
       COMMON /GETRANK/MRNK
       INTEGER IU,IV
       DIMENSION XX(3,50),KKF(3),MM(3),FMINN(3),XBAS(1000,50)
       DIMENSION ZDAT(NOB,MVAR+1),FRANK(NOB),RMAT(MVAR+1,MVAR+1)
       DIMENSION X(50)! X IS THE DECISION VARIABLE X IN F(X) TO MINIMIZE
C      M IS THE DIMENSION OF THE PROBLEM, KF IS TEST FUNCTION CODE AND
C      FMIN IS THE MIN VALUE OF F(X) OBTAINED FROM RPS
       WRITE(*,*)'====================     WARNING    =============== '
       WRITE(*,*)'ADJUST PARAMETERS IN SUBROUTINES RPS IF NEEDED '
       NOPT=1 ! OPTIMIZATION BY RPS METHOD
       WRITE(*,*)'==================================================== '
       METHOD(1)=' : REPULSIVE PARTICLE SWARM OPTIMIZATION'
C      INITIALIZATION. THIS XBAS WILL BE USED TO
C      INITIALIZE THE POPULATION.
       WRITE(*,*)' '
       WRITE(*,*)'FEED RANDOM NUMBER SEED,NORM,ENTROPY,NCOR'
       WRITE(*,*)'SEED[ANY 4-DIGIT NUMBER]; NORM[1,2,3]; ENTROPY[0,1]; &
      &NCOR[0,1]'
       WRITE(*,*)' '
       WRITE(*,*)'NORM(1)=ABSOLUTE;NORM(2)=PCA-EUCLIDEAN;NORM(3)=MAXIMIN'
       WRITE(*,*)'ENTROPY(0)=MAXIMIZES NORM;ENTROPY(1)=MAXIMIZES ENTROPY'
       WRITE(*,*)'NCOR(0)=PRODUCT MOMENT; NCOR(1)=ABSOLUTE CORRELATION'
       READ(*,*) IU,NORM,ENTROPY,NCOR
       WRITE(*,*)'WANT RANK SCORE OPTIMIZATION? YES(1); NO(OTHER THAN 1)'
       READ(*,*) MRNK
       WRITE(*,*)'INPUT FILE TO READ DATA:YOUR DATA MUST BE IN THIS FILE'
       WRITE(*,*)'CASES (NOB) IN ROWS ; VARIABLES (MVAR) IN COLUMNS'
       READ(*,*) INFILE
       WRITE(*,*)'SPECIFY THE OUTPUT FILE TO STORE THE RESULTS'
       READ(*,*) OUTFILE
       OPEN(9, FILE=OUTFILE)
       OPEN(7,FILE=INFILE)
       DO I=1,NOB
       READ(7,*),CDA,(CDAT(I,J),J=1,MVAR)
       ENDDO
       CLOSE(7)
       DO I=1,NOB
       DO J=1,MVAR
       ZDAT(I,J+1)=CDAT(I,J)
       ENDDO
       ENDDO
```

```
      WRITE(*,*)'DATA HAS BEEN READ. WOULD YOU UNITIZE VARIABLES? [YES=1
     & ELSE NO UNITIZATION]'
      WRITE(*,*)'UNITIZE MEANS TRANSFORMATION FROM X(I,J) TO UNITIZED X'
      WRITE(*,*)'[X(I,J)-MIN(X(.,J))]/[MAX(X(.,J))-MIN(X(.,J))]'
      READ(*,*) NUN
      IF(NUN.EQ.1) THEN
      DO J=1,MVAR
      CMIN=CDAT(1,J)
      CMAX=CDAT(1,J)
      DO I=2,NOB
      IF(CMIN.GT.CDAT(I,J)) CMIN=CDAT(I,J)
      IF(CMAX.LT.CDAT(I,J)) CMAX=CDAT(I,J)
      ENDDO
      DO I=1,NOB
      CDAT(I,J)=(CDAT(I,J)-CMIN)/(CMAX-CMIN)
      ENDDO
      ENDDO
      ENDIF
C     -----------------------------------------------------
      WRITE(*,*)' '
      WRITE(*,*)'FEED RANDOM NUMBER SEED [4-DIGIT ODD INTEGER] TO BEGIN'
      READ(*,*) IU
C     THIS XBAS WILL BE USED AS INITIAL X
      DO I=1,1000
      DO J=1,50
      CALL RANDOM(RAND)
      XBAS(I,J)=RAND ! RANDOM NUMBER BETWEEN (0, 1)
      ENDDO
      ENDDO
C     -----------------------------------------------------------------
C     HOWEVER, THE FIRST ROW OF, THAT IS, XBAS(1,J),J=1,MVAR) MAY BE
C     SPECIFIED HERE IF THE USER KNOWS IT TO BE OPTIMAL OR NEAR-OPTIMAL
C     DATA (XBAS(1,J),J=1,MVAR) /DATA1, DATA2, ............, DATAMVAR/
C     -----------------------------------------------------------------
      WRITE(*,*)' ****************************************************'
C     -----------------------------------------------------------------
      DO I=1,NOPT
      IF(I.EQ.1) THEN
      WRITE(*,*)'==== WELCOME TO RPS PROGRAM FOR INDEX CONSTRUCTION'
      CALL RPS(M,X,FMINRPS,Q1) !CALLS RPS AND RETURNS OPTIMAL X AND FMIN
      WRITE(*,*)'RPS BRINGS THE FOLLOWING VALUES TO THE MAIN PROGRAM'
      WRITE(*,*)(X(JOPT),JOPT=1,M),' OPTIMUM FUNCTION=',FMINRPS
      IF(KF.EQ.1) THEN
      WRITE(9,*)'PARTICLE SWARM OPTIMIZATION RESULTS'
      RSUM1=0.D0
      RSUM2=0.D0
      DO J=1,MVAR
      RSUM1=RSUM1+DABS(R(J))
      RSUM2=RSUM2+DABS(R(J))**2
      ENDDO
      WRITE(9,*)'CORRELATION OF INDEX WITH CONSTITUENT VARIABLES'
      WRITE(9,*)(R(J),J=1,MVAR)
      WRITE(9,*)'SUM OF ABS (R)=',RSUM1,';   SUM OF SQUARE(R)=',RSUM2
      WRITE(9,*)'THE INDEX OR SCORE OF DIFFERENT CASES'
      DO II=1,NOB
      WRITE(9,*)QIND(II)
      FRANK(II)=QIND(II)
      ENDDO
      ENDIF
      FMIN=FMINRPS
      ENDIF
C     -----------------------------------------------------------------
      DO J=1,M
      XX(I,J)=X(J)
      ENDDO
      KKF(I)=KF
      MM(I)=M
```

```
      FMINN(I)=FMIN
      ENDDO
      WRITE(*,*)' '
      WRITE(*,*)' '
      WRITE(*,*)'---------------------- FINAL RESULTS=================='
      DO I=1,NOPT
      WRITE(*,*)'FUNCT CODE=',KKF(I),'  FMIN=',FMINN(I),' : DIM=',MM(I)
      WRITE(*,*)'OPTIMAL DECISION VARIABLES : ',METHOD(I)
      WRITE(*,*) 'WEIGHTS ARE AS FOLLOWS --------------'
      WRITE(9,*) 'WEIGHTS ARE AS FOLLOWS --------------'
      WRITE(9,*)(XX(I,J),J=1,M)
      WRITE(*,*)(XX(I,J),J=1,M)
      WRITE(*,*)'///////////////////////////////////////////////////////'
      ENDDO
      WRITE(*,*)'OPTIMIZATION PROGRAM ENDED'
      WRITE(*,*)'*******************************************************'
      WRITE(*,*)'MEASURE OF EQUALITY/INEQUALITY'
      WRITE(*,*)'RPS: BEFORE AND AFTER OPTIMIZATION = ',Q0,Q1
      WRITE(*,*)' '
      WRITE(*,*)'RESULTS STORED IN FILE= ',OUTFILE
      WRITE(*,*)'OPEN BY MSWORD OR EDIT OR ANY OTHER EDITOR'
      WRITE(*,*)' '
      WRITE(*,*)'NOTE:VECTORS OF CORRELATIONS & INDEX(BOTH TOGETHER) ARE
     & IDETERMINATE FOR SIGN &   MAY BE MULTIPLED BY (-1) IF NEEDED'
      WRITE(*,*)'THAT IS IF R(J) IS TRANSFORMED TO -R(J) FOR ALL J THEN
     &THE INDEX(I) TOO IS         TRANSFORMED TO -INDEX(I) FOR ALL I'
      WRITE(9,*)' '
      WRITE(9,*)'NOTE: VECTORS OF CORRELATIONS AND INDEX (BOTH TOGETHER)
     & ARE IDETERMINATE FOR SIGN AND MAY BE MULTIPLED BY (-1) IF NEEDED'
      WRITE(9,*)'THAT IS IF R(J) IS TRANSFORMED TO -R(J) FOR ALL J THEN
     &THE INDEX(I) TOO IS TRANSFORMED TO -INDEX(I) FOR ALL I'
      CALL DORANK(FRANK,NOB)
      DO I=1,NOB
      ZDAT(I,1)=FRANK(I)
      ENDDO
      IF(NCOR.EQ.0) THEN
      CALL CORREL(ZDAT,NOB,MVAR+1,RMAT)
      ELSE
      CALL DOCORA(ZDAT,NOB,MVAR+1,RMAT)
      ENDIF
      WRITE(9,*)'-------------------- CORRELATION MATRIX --------------'
      WRITE(*,*)'-------------------- CORRELATION MATRIX --------------'
      DO I=1,MVAR+1
      WRITE(9,1)(RMAT(I,J),J=1,MVAR+1)
      WRITE(*,1)(RMAT(I,J),J=1,MVAR+1)
      ENDDO
    1 FORMAT(8F10.6)
      WRITE(9,*)'=================================================== '
      WRITE(*,*)'=================================================== '
      WRITE(9,*)'VARIABLES: 1ST IS THE INDEX AND 2ND THE RANK OF INDEX'
      WRITE(*,*)'VARIABLES: 1ST IS THE INDEX AND 2ND THE RANK OF INDEX'
      WRITE(9,*)'=================================================== '
      WRITE(*,*)'=================================================== '
      DO I=1,NOB
      IF(MRNK.EQ.1) THEN
      QIND(I)=0.D0
      DO J=1,MVAR
      QIND(I)=QIND(I)+ZDAT(I,J+1)*XX(NOPT,J)
      ENDDO
      ENDIF
      WRITE(9,2)I,QIND(I),(ZDAT(I,J),J=1,MVAR+1)
      WRITE(*,2)I,QIND(I),(ZDAT(I,J),J=1,MVAR+1)
      ENDDO
    2 FORMAT(I4,F12.6,9F7.2)
      SR2=0.D0
      IF(NORM.LE.2) THEN
      DO J=2,MVAR+1
```

```
      SR2=SR2+DABS(RMAT(1,J))**NORM
      ENDDO
      IF(NORM.EQ.2) THEN
      WRITE(9,*)'SUM OF SQUARE OF CORRELATION R(RANK(INDEX),VAR)=',SR2
      WRITE(*,*)'SUM OF SQUARE OF CORRELATION R(RANK(INDEX),VAR)=',SR2
      ENDIF
      IF(NORM.EQ.1) THEN
      WRITE(9,*)'SUM OF ABSOLUTE OF CORRELATION R(RANK(INDEX),VAR)=',SR2
      WRITE(*,*)'SUM OF ABSOLUTE OF CORRELATION R(RANK(INDEX),VAR)=',SR2
      ENDIF
      ELSE ! GIVES MAXIMAL CORRELATION
      SR2=1.D0
      DO J=2,MVAR+1
      IF(SR2.GT.DABS(RMAT(1,J))) SR2=DABS(RMAT(1,J))
      ENDDO
      WRITE(9,*)'MAXIMIN CORRELATION R(RANK(INDEX),VAR)=',SR2
      WRITE(*,*)'MAXIMIN CORRELATION R(RANK(INDEX),VAR)=',SR2
      ENDIF
      CLOSE(9)
      WRITE(*,*) 'THE JOB IS OVER'
      END
C     -----------------------------------------------------------------
      SUBROUTINE RPS(M,ABEST,FBEST,G1)
C     PROGRAM TO FIND GLOBAL MINIMUM BY REPULSIVE PARTICLE SWARM METHOD
C     WRITTEN BY SK MISHRA, DEPT. OF ECONOMICS, NEHU, SHILLONG (INDIA)
C     -----------------------------------------------------------------
      PARAMETER (N=100,NN=30,MX=100,NSTEP=7,ITRN=10000,NSIGMA=1,ITOP=1)
      PARAMETER (NPRN=50) ! DISPLAYS RESULTS AT EVERY 500 TH ITERATION
C     PARAMETER(N=50,NN=25,MX=100,NSTEP=9,ITRN=10000,NSIGMA=1,ITOP=3)
C     PARAMETER (N=100,NN=15,MX=100,NSTEP=9,ITRN=10000,NSIGMA=1,ITOP=3)
C     IN CERTAIN CASES THE ONE OR THE OTHER SPECIFICATION WORKS BETTER
C     DIFFERENT SPECIFICATIONS OF PARAMETERS MAY SUIT DIFFERENT TYPES
C     OF FUNCTIONS OR DIMENSIONS - ONE HAS TO DO SOME TRIAL AND ERROR
C     -----------------------------------------------------------------
C     N = POPULATION SIZE. IN MOST OF THE CASES N=30 IS OK. ITS VALUE
C     MAY BE INCREASED TO 50 OR 100 TOO. THE PARAMETER NN IS THE SIZE OF
C     RANDOMLY CHOSEN NEIGHBOURS. 15 TO 25 (BUT SUFFICIENTLY LESS THAN
C     N) IS A GOOD CHOICE. MX IS THE MAXIMAL SIZE OF DECISION VARIABLES.
C     IN F(X1, X2,...,XM) M SHOULD BE LESS THAN OR EQUAL TO MX. ITRN IS
C     THE NO. OF ITERATIONS. IT MAY DEPEND ON THE PROBLEM. 200(AT LEAST)
C     TO 500 ITERATIONS MAY BE GOOD ENOUGH. BUT FOR FUNCTIONS LIKE
C     ROSENBROCKOR GRIEWANK OF LARGE SIZE (SAY M=30) IT IS NEEDED THAT
C     ITRN IS LARGE, SAY 5000 OR EVEN 10000.
C     SIGMA INTRODUCES PERTURBATION & HELPS THE SEARCH JUMP OUT OF LOCAL
C     OPTIMA. FOR EXAMPLE : RASTRIGIN FUNCTION OF DMENSION 30 OR LARGER
C     NSTEP DOES LOCAL SEARCH BY TUNNELLING AND WORKS WELL BETWEEN 5 AND
C     15, WHICH IS MUCH ON THE HIGHER SIDE.
C     ITOP <=1 (RING); ITOP=2 (RING AND RANDOM); ITOP=>3 (RANDOM)
C     NSIGMA=0 (NO CHAOTIC PERTURBATION);NSIGMA=1 (CHAOTIC PERTURBATION)
C     NOTE THAT NSIGMA=1 NEED NOT ALWAYS WORK BETTER (OR WORSE)
C     SUBROUTINE FUNC( ) DEFINES OR CALLS THE FUNCTION TO BE OPTIMIZED.
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      COMMON /KFF/KF,NFCALL,FTIT
      INTEGER IU,IV
      CHARACTER *70 FTIT
      DIMENSION X(N,MX),V(N,MX),A(MX),VI(MX),TIT(50),ABEST(*)
      DIMENSION XX(N,MX),F(N),V1(MX),V2(MX),V3(MX),V4(MX),BST(MX)
C     A1 A2 AND A3 ARE CONSTANTS AND W IS THE INERTIA WEIGHT.
C     OCCASIONALLY, TINKERING WITH THESE VALUES, ESPECIALLY A3, MAY BE
C     NEEDED.
      DATA A1,A2,A3,W,SIGMA /.5D00,.5D00,.0005D00,.5D00,1.D-03/
      EPSILON=1.D-12 ! ACCURACY NEEDED FOR TERMINATON
C     --------------------CHOOSING THE TEST FUNCTION -------------------'
      CALL FSELECT(KF,M,FTIT)
C     -----------------------------------------------------------------
      FFMIN=1.D30
```

```
      LCOUNT=0
      NFCALL=0
      WRITE(*,*)'4-DIGITS SEED FOR RANDOM NUMBER GENERATION'
      READ(*,*) IU
      DATA FMIN /1.0E30/
C     GENERATE N-SIZE POPULATION OF M-TUPLE PARAMETERS X(I,J) RANDOMLY
      DO I=1,N
        DO J=1,M
        CALL RANDOM(RAND)
         X(I,J)=RAND
C     WE GENERATE RANDOM(-5,5). HERE MULTIPLIER IS 10. TINKERING IN SOME
C     CASES MAY BE NEEDED
         ENDDO
        F(I)=1.0D30
      ENDDO
C     INITIALISE VELOCITIES V(I) FOR EACH INDIVIDUAL IN THE POPULATION
      DO I=1,N
      DO J=1,M
      CALL RANDOM(RAND)
       V(I,J)=(RAND-0.5D+00)
C       V(I,J)=RAND
      ENDDO
      ENDDO
      DO 100 ITER=1,ITRN
      WRITE(*,*)'ITERATION=',ITER
C     LET EACH INDIVIDUAL SEARCH FOR THE BEST IN ITS NEIGHBOURHOOD
        DO I=1,N
           DO J=1,M
           A(J)=X(I,J)
           VI(J)=V(I,J)
           ENDDO
           CALL LSRCH(A,M,VI,NSTEP,FI)
           IF(FI.LT.F(I)) THEN
            F(I)=FI
            DO IN=1,M
            BST(IN)=A(IN)
            ENDDO
C     F(I) CONTAINS THE LOCAL BEST VALUE OF FUNCTION FOR ITH INDIVIDUAL
C     XX(I,J) IS THE M-TUPLE VALUE OF X ASSOCIATED WITH LOCAL BEST F(I)
             DO J=1,M
             XX(I,J)=A(J)
             ENDDO
             ENDIF
        ENDDO
C      NOW LET EVERY INDIVIDUAL RANDOMLY COSULT NN(<<N) COLLEAGUES AND
C      FIND THE BEST AMONG THEM
      DO I=1,N
C     ------------------------------------------------------------------
      IF(ITOP.GE.3) THEN
C     RANDOM TOPOLOGY ******************************************
C     CHOOSE NN COLLEAGUES RANDOMLY AND FIND THE BEST AMONG THEM
          BEST=1.0D30
           DO II=1,NN
                 CALL RANDOM(RAND)
               NF=INT(RAND*N)+1
                IF(BEST.GT.F(NF)) THEN
                 BEST=F(NF)
                NFBEST=NF
                 ENDIF
            ENDDO
      ENDIF
C----------------------------------------------------------------------
      IF(ITOP.EQ.2) THEN
C     RING + RANDOM TOPOLOGY ******************************************
C     REQUIRES THAT THE SUBROUTINE NEIGHBOR IS TURNED ALIVE
       BEST=1.0D30
          CALL NEIGHBOR(I,N,I1,I3)
```

```
          DO II=1,NN
                IF(II.EQ.1) NF=I1
                 IF(II.EQ.2) NF=I
                  IF(II.EQ.3) NF=I3
                      IF(II.GT.3) THEN
                     CALL RANDOM(RAND)
                      NF=INT(RAND*N)+1
                     ENDIF
                  IF(BEST.GT.F(NF)) THEN
                  BEST=F(NF)
                  NFBEST=NF
                 ENDIF
             ENDDO
       ENDIF
C-----------------------------------------------------------------------
      IF(ITOP.LE.1) THEN
C      RING TOPOLOGY ****************************************************
C      REQUIRES THAT THE SUBROUTINE NEIGHBOR IS TURNED ALIVE
        BEST=1.0D30
           CALL NEIGHBOR(I,N,I1,I3)
              DO II=1,3
              IF (II.NE.I) THEN
             IF(II.EQ.1) NF=I1
              IF(II.EQ.3) NF=I3
                  IF(BEST.GT.F(NF)) THEN
                   BEST=F(NF)
                   NFBEST=NF
                  ENDIF
                  ENDIF
            ENDDO
       ENDIF
C-----------------------------------------------------------------------
C      IN THE LIGHT OF HIS OWN AND HIS BEST COLLEAGUES EXPERIENCE, THE
C      INDIVIDUAL I WILL MODIFY HIS MOVE AS PER THE FOLLOWING CRITERION
C      FIRST, ADJUSTMENT BASED ON ONES OWN EXPERIENCE
C      AND OWN BEST EXPERIENCE IN THE PAST (XX(I))
           DO J=1,M
           CALL RANDOM(RAND)
           V1(J)=A1*RAND*(XX(I,J)-X(I,J))

C      THEN BASED ON THE OTHER COLLEAGUES BEST EXPERIENCE WITH WEIGHT W
C      HERE W IS CALLED AN INERTIA WEIGHT 0.01< W < 0.7
C      A2 IS THE CONSTANT NEAR BUT LESS THAN UNITY
           CALL RANDOM(RAND)
           V2(J)=V(I,J)
           IF(F(NFBEST).LT.F(I)) THEN
           V2(J)=A2*W*RAND*(XX(NFBEST,J)-X(I,J))
           ENDIF
C      THEN SOME RANDOMNESS AND A CONSTANT A3 CLOSE TO BUT LESS THAN UNITY
           CALL RANDOM(RAND)
           RND1=RAND
           CALL RANDOM(RAND)
            V3(J)=A3*RAND*W*RND1
C            V3(J)=A3*RAND*W
C      THEN ON PAST VELOCITY WITH INERTIA WEIGHT W
           V4(J)=W*V(I,J)
C      FINALLY A SUM OF THEM
           V(I,J)= V1(J)+V2(J)+V3(J)+V4(J)
           ENDDO
      ENDDO
C      CHANGE X
      DO I=1,N
      DO J=1,M
      RANDS=0.D00
C      -------------------------------------------------------------------
      IF(NSIGMA.EQ.1) THEN
       CALL RANDOM(RAND) ! FOR CHAOTIC PERTURBATION
```

```
        IF(DABS(RAND-.5D00).LT.SIGMA) RANDS=RAND-0.5D00
C      SIGMA CONDITIONED RANDS INTRODUCES CHAOTIC ELEMENT IN TO LOCATION
C      IN SOME CASES THIS PERTURBATION HAS WORKED VERY EFFECTIVELY WITH
C      PARAMETER (N=100,NN=15,MX=100,NSTEP=9,ITRN=100000,NSIGMA=1,ITOP=2)
       ENDIF
C      -----------------------------------------------------------------
       X(I,J)=X(I,J)+V(I,J)*(1.D00+RANDS)
       ENDDO
       ENDDO
        DO I=1,N
          IF(F(I).LT.FMIN) THEN
          FMIN=F(I)
          II=I
          DO J=1,M
          BST(J)=XX(II,J)
          ENDDO
          ENDIF
          ENDDO

       IF(LCOUNT.EQ.NPRN) THEN
       LCOUNT=0
       WRITE(*,*)'OPTIMAL SOLUTION UPTO THIS (FUNCTION CALLS=',NFCALL,')'
       WRITE(*,*)'X = ',(BST(J),J=1,M),' MIN F = ',FMIN
C       WRITE(*,*)'NO. OF FUNCTION CALLS = ',NFCALL
       DO J=1,M
       ABEST(J)=BST(J)
       ENDDO
       IF(DABS(FFMIN-FMIN).LT.EPSILON) GOTO 999
       FFMIN=FMIN
       ENDIF
       LCOUNT=LCOUNT+1
  100  CONTINUE
  999  WRITE(*,*)'-------------------------------------------------------'
       DO I=1,N
       IF(F(I).LT.FBEST) THEN
       FBEST=F(I)
       DO J=1,M
       ABEST(J)=XX(I,J)
       ENDDO
       ENDIF
       ENDDO
       CALL FUNC(ABEST,M,FBEST)
       CALL GINI(F,N,G1)
       WRITE(*,*)'FINAL X = ',(BST(J),J=1,M),' FINAL MIN F = ',FMIN
       WRITE(*,*)'COMPUTATION OVER:FOR ',FTIT
       WRITE(*,*)'NO. OF VARIABLES=',M,'  END.'
       RETURN
       END
C      -----------------------------------------------------------------
       SUBROUTINE LSRCH(A,M,VI,NSTEP,FI)
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       CHARACTER *70 FTIT
       COMMON /KFF/KF,NFCALL,FTIT
       COMMON /RNDM/IU,IV
       INTEGER IU,IV
       DIMENSION A(*),B(100),VI(*)
       AMN=1.0D30
       DO J=1,NSTEP
          DO JJ=1,M
          B(JJ)=A(JJ)+(J-(NSTEP/2)-1)*VI(JJ)
          ENDDO
       CALL FUNC(B,M,FI)
         IF(FI.LT.AMN) THEN
         AMN=FI
         DO JJ=1,M
         A(JJ)=B(JJ)
         ENDDO
```

```
          ENDIF
        ENDDO
        FI=AMN
        RETURN
        END
C       -----------------------------------------------------------------
C       THIS SUBROUTINE IS NEEDED IF THE NEIGHBOURHOOD HAS RING TOPOLOGY
C       EITHER PURE OR HYBRIDIZED
         SUBROUTINE NEIGHBOR(I,N,J,K)
         IF(I-1.GE.1 .AND. I.LT.N) THEN
         J=I-1
         K=I+1
         ELSE
         IF(I-1.LT.1) THEN
         J=N-I+1
         K=I+1
         ENDIF
         IF(I.EQ.N) THEN
         J=I-1
         K=1
         ENDIF
         ENDIF
         RETURN
         END
C       -----------------------------------------------------------------
C       RANDOM NUMBER GENERATOR (UNIFORM BETWEEN 0 AND 1 - BOTH EXCLUSIVE)
        SUBROUTINE RANDOM(RAND1)
         DOUBLE PRECISION  RAND1
         COMMON /RNDM/IU,IV
        INTEGER IU,IV
         IV=IU*65539
         IF(IV.LT.0) THEN
         IV=IV+2147483647+1
         ENDIF
         RAND=IV
         IU=IV
         RAND=RAND*0.4656613E-09
         RAND1= DBLE(RAND)
         RETURN
         END
C       -----------------------------------------------------------------
        SUBROUTINE GINI(F,N,G)
        PARAMETER (K=1) !K=1 GINI COEFFICENT; K=2 COEFFICIENT OF VARIATION
C       THIS PROGRAM COMPUTES MEASURE OF INEQUALITY
C       IF K =1 GET THE GINI COEFFICIENT. IF K=2 GET COEFF OF VARIATIONE
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        DIMENSION F(*)
        S=0.D0
        DO I=1,N
        S=S+F(I)
        ENDDO
        S=S/N
        H=0.D00
        DO I=1,N-1
        DO J=I+1,N
        H=H+(DABS(F(I)-F(J)))**K
        ENDDO
        ENDDO
        H=(H/(N**2))**(1.D0/K)! FOR K=1 H IS MEAN DEVIATION;
C                               FOR K=2 H IS STANDARD DEVIATION
        WRITE(*,*)'MEASURES OF DISPERSION AND CENTRAL TENDENCY = ',G,S
        G=DEXP(-H)! G IS THE MEASURE OF EQUALITY (NOT GINI OR CV)
C       G=H/DABS(S) !IF S NOT ZERO, K=1 THEN G=GINI, K=2 G=COEFF VARIATION
        RETURN
        END
C       -----------------------------------------------------------------
        SUBROUTINE FSELECT(KF,M,FTIT)
```

```
C      THE PROGRAM REQUIRES INPUTS FROM THE USER ON THE FOLLOWING ------
C      (1) FUNCTION CODE (KF), (2) NO. OF VARIABLES IN THE FUNCTION (M);
       CHARACTER *70 TIT(100),FTIT
       NFN=1
       KF=1
       WRITE(*,*)'---------------------------------------------------'
       DATA TIT(1)/'CONSTRUCTION OF INDEX FROM M VARIABLES '/
C      -----------------------------------------------------------------
       DO I=1,NFN
       WRITE(*,*)TIT(I)
       ENDDO
       WRITE(*,*)'---------------------------------------------------'
       WRITE(*,*)'SPECIFY NO. OF VARIABLES [MVAR] HERE ALSO ?'
       READ(*,*) M
       FTIT=TIT(KF) ! STORE THE NAME OF THE CHOSEN FUNCTION IN FTIT
       RETURN
       END
C      -----------------------------------------------------------------
       SUBROUTINE FUNC(X,M,F)
C      TEST FUNCTIONS FOR GLOBAL OPTIMIZATION PROGRAM
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       COMMON /RNDM/IU,IV
       COMMON /KFF/KF,NFCALL,FTIT
       INTEGER IU,IV
       DIMENSION X(*)
       CHARACTER *70 FTIT
       NFCALL=NFCALL+1 ! INCREMENT TO NUMBER OF FUNCTION CALLS
C      KF IS THE CODE OF THE TEST FUNCTION
       IF(KF.EQ.1) THEN
       CALL CORD(M,X,F)
       RETURN
       ENDIF
C      =================================================================
       WRITE(*,*)'FUNCTION NOT DEFINED. PROGRAM ABORTED'
       STOP
       END
C      ------------------------------------------------------------------
       SUBROUTINE CORD(M,X,F)
       PARAMETER (NOB=30,MVAR=7)! CHANGE THE PARAMETERS HERE AS NEEDED.
C      ------------------------------------------------------------------
C      NOB=NO. OF OBSERVATIONS (CASES) & MVAR= NO. OF VARIABLES
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       COMMON /RNDM/IU,IV
       COMMON /CORDAT/CDAT(NOB,MVAR),QIND(NOB),R(MVAR),ENTROPY,NORM,NCOR
       COMMON /GETRANK/MRNK
       INTEGER IU,IV
       DIMENSION X(*),Z(NOB,2)
       DO I=1,M
       IF(X(I).LT.-1.0D0.OR.X(I).GT.1.0D0) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D0)*2
       ENDIF
       ENDDO
       XNORM=0.D0
       DO J=1,M
       XNORM=XNORM+X(J)**2
       ENDDO
       XNORM=DSQRT(XNORM)
       DO J=1,M
       X(J)=X(J)/XNORM
       ENDDO
C      CONSTRUCT INDEX
       DO I=1,NOB
       QIND(I)=0.D0
       DO J=1,M
       QIND(I)=QIND(I)+CDAT(I,J)*X(J)
       ENDDO
```

```fortran
      ENDDO
C     -------------------------------------------------------------------
      !FIND THE RANK OF QIND
      IF(MRNK.EQ.1) CALL DORANK(QIND,NOB)
C     -------------------------------------------------------------------
C     COMPUTE CORRELATIONS
      DO I=1,NOB
      Z(I,1)=QIND(I)
      ENDDO
      DO J=1,M
      DO I=1,NOB
      Z(I,2)=CDAT(I,J)
      ENDDO
      IF(NCOR.EQ.0) THEN
      CALL CORLN(Z,NOB,RHO)
      ELSE
      CALL CORA(Z,NOB,RHO)
      ENDIF
      R(J)=RHO
      ENDDO
      IF(ENTROPY.EQ.0.D0) THEN
C     ------------------ MAXIMIN SOLUTION ----------------------------
      IF(NORM.GT.2) THEN
       FR=DABS(R(1))
       DO J=2,M
       IF(FR.GT.DABS(R(J))) FR= DABS(R(J))
       ENDDO
       F=FR
       ENDIF
C     ------------------ FOR  NORM =1 OR 2 ---------------------------
      IF(NORM.LE.2) THEN
      F=0.D0
      DO J=1,M
      F=F+DABS(R(J))**NORM
      ENDDO
      ENDIF
C     ---------------------------------------------------------------
      ELSE
      IF(ENTROPY.NE.0.D0) THEN
C     ENTROPY MAXIMIZATION
      ENT=0.0D0
      DO J=1,M
      ENT=ENT+DABS(R(J))
      ENDDO
      F=ENT*DLOG(ENT)
      DO J=1,M
      FX=DABS(R(J))
      F=F+ (FX/ENT)*DLOG(FX/ENT)
      ENDDO
      ENDIF
      ENDIF
C     -------------------------------------------------------------------
      F=-F
      RETURN
      END
      SUBROUTINE CORLN(Z,NOB,RHO)
C     NOB = NO. OF CASES
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Z(NOB,2),AV(2),SD(2)
      DO J=1,2
      AV(J)=0.D0
      SD(J)=0.D0
      DO I=1,NOB
      AV(J)=AV(J)+Z(I,J)
      SD(J)=SD(J)+Z(I,J)**2
      ENDDO
      ENDDO
```

```
      DO J=1,2
      AV(J)=AV(J)/NOB
      SD(J)=DSQRT(SD(J)/NOB-AV(J)**2)
      ENDDO
C      WRITE(*,*)'AV AND SD ', AV(1),AV(2),SD(1),SD(2)
      RHO=0.D0
      DO I=1,NOB
      RHO=RHO+(Z(I,1)-AV(1))*(Z(I,2)-AV(2))
      ENDDO
      RHO=(RHO/NOB)/(SD(1)*SD(2))
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE CORA(Z,N,R)
C     COMPUTING BRADLEY'S ABSOLUTE CORRELATION MATRIX
C     BRADLEY, C. (1985) "THE ABSOLUTE CORRELATION", THE MATHEMATICAL
C     GAZETTE, 69(447): 12-17.
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Z(N,2),X(N),Y(N)
C     -------------------------------------------------------------------
C     PUT Z INTO X AND Y
      DO I=1,N
      X(I)=Z(I,1)
      Y(I)=Z(I,2)
      ENDDO
C     ARRANGE X ANY IN AN ASCENDING ORDER
      DO I=1,N-1
      DO II=I+1,N
      IF(X(I).GT.X(II)) THEN
      TEMP=X(I)
      X(I)=X(II)
      X(II)=TEMP
      ENDIF
      IF(Y(I).GT.Y(II)) THEN
      TEMP=Y(I)
      Y(I)=Y(II)
      Y(II)=TEMP
      ENDIF
      ENDDO
      ENDDO
C     FIND MEDIAN
      IF(INT(N/2).EQ.N/2.D0) THEN
      XMED=(X(N/2)+X(N/2+1))/2.D0
      YMED=(Y(N/2)+Y(N/2+1))/2.D0
      ENDIF
      IF(INT(N/2).NE.N/2.D0) THEN
      XMED=X(N/2+1)
      YMED=Y(N/2+1)
      ENDIF
C     SUBTRACT RESPECTIVE MEDIANS FROM X AND Y AND FIND ABS DEVIATIONS
      VX=0.D0
      VY=0.D0
      DO I=1,N
      X(I)=X(I)-XMED
      Y(I)=Y(I)-YMED
      VX=VX+DABS(X(I))
      VY=VY+DABS(Y(I))
      ENDDO
C     SCALE THE VARIABLES X AND Y SUCH THAT VX=VY
      IF(VX.EQ.0.D0.OR.VY.EQ.0.D0) THEN
      R=0.D0
      RETURN
      ENDIF
      DO I=1,N
      X(I)=X(I)*VY/VX
      ENDDO
C     COMPUTE CORRELATION COEFFICIENT
```

```
      VZ=0.D0
      R=0.D0
      DO I=1,N
      VZ=VZ+DABS(X(I))+DABS(Y(I))
      R=R+DABS(X(I)+Y(I))-DABS(X(I)-Y(I))
      ENDDO
      R=R/VZ
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE DORANK(X,N)! N IS THE NUMBER OF OBSERVATIONS
      PARAMETER (NRL=0) ! THIS VALUE IS TO BE SET BY THE USER
C                   !THE VALUE OF NRL DECIDES THE SCHEME OF RANKINGS
C     !THIS PROGRAM RETURNS RANK-ORDER OF A GIVEN VECTOR
      PARAMETER (MXD=1000)! MXD IS MAX DIMENSION FOR TEMPORARY VARIABLES
      ! THAT ARE LOCAL AND DO NOT GO TO THE INVOKING PROGRAM
      ! X IS THE VARIABLE TO BE SUBSTITUTED BY ITS RANK VALUES
C     NRULE=0 FOR ORDINAL RANKING (1-2-3-4 RULE);
C     NRULE=1 FOR DENSE RANKING (1-2-2-3 RULE);
C     NRULE=2 FOR STANDARD COMPETITION RANKING (1-2-2-4 RULE);
C     NRULE=3 FOR MODIFIED COMPETITION RANKING (1-3-3-4 RULE);
C     NRULE=4 FOR FRACTIONAL RANKING (1-2.5-2.5-4 RULE);
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION X(N),NF(MXD),NCF(MXD),RANK(MXD),ID(MXD),XX(MXD)
C     GENERATE ID(I),I=1,N
      DO I=1,N
      ID(I)=I
      NF(I)=0
      ENDDO
C     ARRANGE DATA (X) AND THE IDS IN ASCENDING ORDER
      DO I=1,N-1
      DO II=I,N
      IF(X(II).LT.X(I)) THEN
      TEMP=X(I)
      X(I)=X(II)
      X(II)=TEMP
      ITEMP=ID(I)
      ID(I)=ID(II)
      ID(II)=ITEMP
      ENDIF
      ENDDO
      ENDDO
C     MAKE DISCRETE UNGROUPED FREQUENCY TABLE
      K=0
      J=1
    1 K=K+1
      XX(K)=X(J)
      NF(K)=0
      DO I=J,N
      IF(XX(K).EQ.X(I)) THEN
      NF(K)=NF(K)+1
      ELSE
      J=I
      IF(J.LE.N) THEN
      GOTO 1
      ELSE
      GOTO 2
      ENDIF
      ENDIF
      ENDDO
    2 KK=K
      DO K=1,KK
      IF(K.EQ.1) THEN
      NCF(K)=NF(K)
      ELSE
      NCF(K)=NCF(K-1)+NF(K)
      ENDIF
```

```
      ENDDO
      DO I=1,N
      RANK(I)=1.D0
      ENDDO

      IF(NRL.GT.4) THEN
      WRITE(*,*)'RANKING RULE CODE GREATER THAN 4 NOT PERMITTED',NRL
      STOP
      ENDIF

      IF(NRL.EQ.0) THEN
      DO I=1,N
      RANK(I)=I
      ENDDO
      ENDIF
C     -----------------------------------------------------------------
      IF(NRL.GT.0) THEN
       DO K=1,KK
       IF(K.EQ.1) THEN
       K1=1
       ELSE
       K1=NCF(K-1)+1
       ENDIF
       K2=NCF(K)
       DO I=K1,K2
       SUM=0.D0
       DO II=K1,K2
       SUM=SUM+II
       ENDDO
       KX=(K2-K1+1)
       IF(NRL.EQ.1)RANK(I)=K ! DENSE RANKING (1-2-2-3 RULE)
       IF(NRL.EQ.2)RANK(I)=K1!STANDARD COMPETITION RANKING(1-2-2-4 RULE)
       IF(NRL.EQ.3)RANK(I)=K2!MODIFIED COMPETITION RANKING(1-3-3-4 RULE)
       IF(NRL.EQ.4)RANK(I)=SUM/KX !FRACTIONAL RANKING (1-2.5-2.5-4 RULE)
       ENDDO
       ENDDO
      ENDIF
C     -----------------------------------------------------------------
      DO I=1,N
      X(ID(I))=RANK(I) ! BRINGS THE DATA TO ORIGINAL SEQUENCE
      ENDDO
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE CORREL(X,N,M,RMAT)
      PARAMETER (NMX=30)!DO NOT CHANGE UNLESS NO. OF VARIABLES EXCEED 30
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION X(N,M),RMAT(M,M),AV(NMX),SD(NMX)
      DO J=1,M
      AV(J)=0.D0
      SD(J)=0.D0
      DO I=1,N
      AV(J)=AV(J)+X(I,J)
      SD(J)=SD(J)+X(I,J)**2
      ENDDO
      AV(J)=AV(J)/N
      SD(J)=DSQRT(SD(J)/N-AV(J)**2)
      ENDDO
      DO J=1,M
      DO JJ=1,M
      RMAT(J,JJ)=0.D0
      DO I=1,N
      RMAT(J,JJ)=RMAT(J,JJ)+X(I,J)*X(I,JJ)
      ENDDO
      ENDDO
      ENDDO
      DO J=1,M
```

```
      DO JJ=1,M
      RMAT(J,JJ)=RMAT(J,JJ)/N-AV(J)*AV(JJ)
      RMAT(J,JJ)=RMAT(J,JJ)/(SD(J)*SD(JJ))
      ENDDO
      ENDDO
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE DOCORA(ZDAT,N,M,RMAT)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ZDAT(N,M),RMAT(M,M),Z(N,2)
      DO J=1,M-1
      DO JJ=J+1,M
      DO I=1,N
      Z(I,1)=ZDAT(I,J)
      Z(I,2)=ZDAT(I,JJ)
      ENDDO
      CALL CORA(Z,N,R)
      RMAT(J,JJ)=R
      RMAT(JJ,J)=R
      ENDDO
      ENDDO
      DO J=1,M
      RMAT(J,J)=1.D0
      ENDDO
      RETURN
      END
```