Performance of the Barter, the Differential Evolution and the Simulated Annealing Methods of Global Optimization on Some New and Some Old Test Functions

Mishra, SK

01. November 2006

Performance of the Barter, the Differential Evolution and the Simulated Annealing Methods of Global Optimization on Some New and Some Old Test Functions

SK Mishra
Department of Economics
North-Eastern Hill University
Shillong, Meghalaya (India)

Introduction: The objective of this paper is to assess the performance of the Barter method, a newly introduced population-based (stochastic) heuristic to search the global optimum of a (continuous) multi-modal function, vis-à-vis that of two other well-established and very powerful methods, namely, the Simulated Annealing (SA) and the Differential Evolution (DE) methods of global optimization.

First we will describe the three methods briefly and then present the findings. The computer programs for DE and Barter methods are appended. The SA program (SIMANN) developed by William Goffe has been used to optimize the functions by the Simulated Annealing method.

The Simulated Annealing Method: The simulated annealing method (Kirkpatrick et al., 1983; Cerny, 1985) mimics the annealing process in metallurgy. In an annealing process a metal in the molten state (at a very high temperature) is slowly cooled so that the system at any time is approximately in thermodynamic equilibrium. As cooling proceeds, the system becomes more ordered – the liquid freezes or the metal re-crystallizes – attaining the ground state at T=0. This process is simulated through the Monte Carlo experiment (Metropolis et al. 1953). If the initial temperature of the melt is too low or cooling is done unduly fast the metal may become 'quenched' due to being trapped in a local minimum energy state (meta-stable state) forming defects or freezing out. The simulated annealing method of optimization makes very few assumptions regarding the function to be optimized, and therefore, it is quite robust with respect to irregular surfaces. In this method, the mathematical system describing the problem mimics the thermodynamic system. The current solution to the problem mimics the current state of the thermodynamic system, the objective function mimics the energy equation for the thermodynamic system, and the global minimum mimics the ground state. However, nothing in the numerical optimization problem directly mimics the temperature, T, in the thermodynamic system underlying the metallurgical process of annealing. Therefore, a complex abstraction mimics it. An arbitrary choice of initial value of a variable called 'temperature', how many iterations are performed at each 'temperature', the step length at which the decision variables are adjusted, and the rate of fall of 'temperature' at each step as 'cooling' proceeds, together make an 'annealing schedule'. This schedule mimics the cooling process. At a high 'temperature' the step lengths at which the decision variables are adjusted are larger than those at a lower 'temperature'. Whether the system is trapped into local minima (quenching takes place) or it attains the global minimum (faultless crystallization) is dependent on the said annealing schedule. A wrong choice of the initial 'temperature', or the rate of fall in the 'temperature' leads to quenching or entrapment of the solution in the local minima. The method does not provide any clear

guideline as to the choice of the 'annealing schedule' and often requires judgment or trial and error. If the schedule is properly chosen, the process attains the global minimum. It is said that using this method is an art and requires a lot of experience and judgment.

The Differential Evolution Method: The method of Differential Evolution (DE) grew out of Kenneth Price's attempts to solve the Chebychev Polynomial fitting Problem that had been posed to him by Rainer Storn. A breakthrough happened (1996), when Price came up with the idea of using vector differences for perturbing the vector population. The crucial idea behind DE is a scheme for generating trial parameter vectors. A population of points (p in d-dimensional space) is generated and evaluated (i.e. f(p) is obtained) for their fitness. Then, in one of the schemes, for each point (p_i) three different points (p_a, p_b and p_c) are randomly chosen from the population. A new point (p_z) is constructed from those three points by adding the weighted difference between two points ($w(p_b-p_c)$) to the third point (p_a). Then this new point (p_z) is subjected to a crossover with the current point (p_i) with a probability of crossover (c_r), yielding a candidate point, say p_u. This point, p_u, is evaluated and if found better than p_i then it replaces p_i else p_i remains. Thus we obtain a new vector in which all points are either better than or as good as the current points. This new vector is used for the next iteration. In another (newly recommended) scheme, only two points are randomly chosen from the population and the third point is the p_i itself. Details set apart,, this process makes the differential evaluation scheme completely self-organizing and effective in locating the global optimum.

The Barter Method: This method is based on the well-known proposition in welfare economics that competitive equilibria, under fairly general conditions, tend to be Pareto optimal [Takayama, 1974, pp. 185-201]. In its simplest version, implementation of this proposition may be outlined as follows [Mishra, 2006 (g)]:

Let there be n (fairly large number of) individuals in a population and let each individual, i, own (or draw from the environment) an m-element real vector of resources, $x_i = (x_{i1}, x_{i2}, ..., x_{im})$. For every x_i there is a (single-valued) function $f(x_i)$ that may be used as a measure of the worth of x_i that the individual would like to optimize. The optimand function $f(.)$ is unique and common to all the individuals. Now, let the individuals in the (given) population enter into a barter of their resources with the condition that (i) $\beta(x_{ij}, x_{k\ell} : i \neq k;\ j \neq l)$ or a transaction is feasible across different persons and different resources only, and (ii) the resources will change hands (materialize) only if such a transaction is beneficial to (more desired by) both the parties (in the barter). The choice of the individuals, (i, k) and the resources, (j, ℓ) in every transaction and the quantum of transaction would be stochastic in nature. If such transactions are allowed for a large number of times, then at the end of the session: (a) every individual would be better off than what he was at the initial position, and (b) at least one individual would reach the global optimum.

Table 1: A Summary of Success/Failure of DE, BARTER and SA Methods on Test Functions									
Test Function	**Dim (M)**	**Success**			**Test Function**	**Dim (M)**	**Success**		
		DE	**BA**	**SA**			**DE**	**BA**	**SA**
Ackley Fn	10	yes	yes	yes	Mishra_2	8	no	no	no
Anns XOR Fn	9	yes	no	no	Mod RCos Fn	2	yes	yes	yes
Beale Fn	2	yes	yes	yes	Multimod Fn	2	yes	yes	yes
Bird Fn	2	yes	yes	yes	Needle-eye Fn	10	yes	no	yes
Bohachevsky#1 Fn	2	yes	yes	yes	New AM=GM Fn	10	yes	no	no
Bohachevsky#2 Fn	2	yes	yes	yes	New Decanomial Fn	2	yes	yes	yes
Bohachevsky#3 Fn	2	yes	yes	yes	New Dodecal Fn	3	yes	yes	no
Booth Fn	2	yes	yes	yes	New Factorial Fn	4	yes	no	no
Branin#1 Fn	2	yes	yes	no	New Fn #1	2	yes	no	no
Branin#2 Fn	2	yes	yes	yes	New Fn #2	2	yes	no	no
Bukin-6 Fn	2	no	no	no	New Fn #3	2	yes	yes	yes
Carrom Table Fn	2	yes	yes	yes	New Fn #4	2	yes	yes	yes
Chichinadze Fn	2	yes	yes	yes	New Fn #8	2	yes	yes	yes
Colville Fn.	4	yes	yes	yes	New sum=prod Fn	2	yes	yes	yes
Corana Fn	4	yes	yes	yes	Nonlin Fn	10	yes	yes	yes
Cross Fn	2	yes	yes	nc	Paviani Fn	10	yes	yes	yes
Cross-in-Tray Fn	2	yes	yes	no	Pen-Holder Fn	2	yes	yes	yes
Crosslegged Table Fn	2	yes	yes	no	Perm #1 Fn	4	yes	nc	no
Crowned Cross Fn	2	yes	yes	no	Perm #2 Fn	5	no	no	no
DCS Fn	4	yes	no	no	Powell Fn	8	yes	yes	yes
Dixon-Price Fn	10	yes	no	no	Power-Sum Fn	4	no	yes	no
Easom Fn	2	yes	yes	yes	Quadratic Fn	2	yes	yes	yes
Egg-Holder Fn	2	yes	yes	no	Quintic Fn	10	yes	no	yes
Fenton-Eason Fn	2	yes	yes	yes	Rastrigin Fn	10	yes	yes	no
Fletcher-Powell Fn (1)	5	uns	uns	no	Rosenbrock Fn	10	yes	yes	nc
Fletcher-Powell Fn (2)	5	uns	uns	no	Schwefel Fn	10	yes	nc	no
Fletcher-Powell Fn (o)	5	yes	no	no	Shekel Fn	4	yes	yes	no
Freud-Roth Fn	2	yes	yes	no	Shubert Fn	2	yes	yes	yes
Giunta Fn	2	yes	yes	yes	Trid Fn	10	yes	nc	yes
Glankwahmdee Fn	5	yes	yes	yes	Trigon Fn	10	yes	yes	yes
Goldstein-Price Fn	2	yes	yes	no	TT-Holder Fn	2	yes	yes	nc
Greiwank Fn	10	yes	nc	no	Weierstrass Fn	10	yes	yes	yes
Hartmann Fn	3	yes	yes	yes	Wood's Fn	4	yes	yes	no
Himmelblau Fn	2	yes	yes	yes	Yao-Liu#12 Fn	10	yes	yes	no
Holder Table Fn	2	yes	yes	yes	Yao-Liu#13 Fn	10	yes	yes	yes
Hougen Fn	5	yes	no	no	Yao-Liu#14 Fn	2	yes	no	no
Hump Fn	2	yes	yes	yes	Yao-Liu#15 Fn	4	yes	nc	no
Judge Fn	2	yes	yes	yes	Yao-Liu#2 Fn	10	yes	yes	yes
Levy#3 Fn	2	yes	yes	no	Yao-Liu#3 Fn	10	yes	yes	yes
Levy#5 Fn	2	yes	yes	no	Yao-Liu#4 Fn	10	yes	nc	yes
Levy#8 Fn	3	yes	yes	yes	Yao-Liu#6 Fn	10	yes	yes	yes
Matyas Fn	2	yes	yes	yes	Yao-Liu#7 Fn	10	uns	uns	no
McCormick Fn	2	yes	yes	yes	Zakharov Fn	8	yes	yes	yes
Michalewicz Fn	10	yes	nc	nc	Zero-sum Fn	10	yes	no	yes
Mishra_1 Fn	8	yes	yes	yes	**No. of Failure in 89 trials**		7	26	38

Note: For differently set adjustable parameters, these methods may perform better or worse than reported here.
UNS = unstable; NC = No convergence, but improving over iterations, No = Convergence to local optimum

Computer Programs: A computer program (FORTRAN) that works out the global optimum of the test functions by the two methods (DE and Barter) is appended. It incorporated 87 benchmark functions of varied types, some well-known and others new (proposed by the present author). SIMANN, a global optimization algorithm using simulated annealing by William Goffe and others, is a very effective program written in FORTRAN-77. It may be downloaded from http://www.netlib.no/netlib/opt/simann.f absolutely free of cost.

The Findings: In all, benchmark functions have been optimized 89 times. As presented in table-1, the DE succeeds in 82 cases, the Barter succeeds in 63 cases, while the Simulated Annealing method succeeds for a modest number of 51 cases. The DE as well as Barter methods are unstable for stochastic functions (Yao-Liu#7 and Fletcher-Powell functions). In particular, Bukin-6, Perm-2 and Mishra-2 functions have been hard for all the three methods.

Seen as such, the barter method is much inferior to the DE, but it performs better than SA. A comparison of the Barter method with the Repulsive Particle Swarm method has indicated that they are more or less comparable [Mishra, 2006 (g)].

The convergence rate of the Barter method is slower than the DE as well as the SA. This is because of the difficulty of 'double coincidence' in bartering. Barter activity takes place successfully in less than one percent trials.

It may be noted that the DE and the SA have a longer history behind them and they have been improved many times. In the present exercise, the DE version used here employs the latest (available) schemes of crossover, mutation and recombination. In comparison to this, the Barter method is a nascent one. We need a thorough investigation into the nature and performance of the Barter method. We have found that when the DE optimizes, the terminal population is homogenous while in case of the Barter method it is not so. This property of the Barter method has several implications with respect to the Agent-Based Computational Economics [Tesfatsion, 2002].

Bibliography

- Bauer, J.M.: "Harnessing the Swarm: Communication Policy in an Era of Ubiquitous Networks and Disruptive Technologies", *Communications and Strategies*, 45, 2002.
- Box, M.J.: "A New Method of Constrained Optimization and a Comparison with Other Methods". *Comp. J.* 8, pp. 42-52, 1965.
- Bukin, A. D.: *New Minimization Strategy For Non-Smooth Functions*, Budker Institute of Nuclear Physics preprint BUDKER-INP-1997-79, Novosibirsk 1997.
- Cerny, V.: "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm", *J. Opt. Theory Appl.*, 45, 1, 41-51, 1985.
- Eberhart R.C. and Kennedy J.: "A New Optimizer using Particle Swarm Theory", *Proceedings Sixth Symposium on Micro Machine and Human Science*, pp. 39–43. IEEE Service Center, Piscataway, NJ, 1995.
- Fleischer, M.: "Foundations of Swarm Intelligence: From Principles to Practice", Swarming Network Enabled C4ISR, arXiv:nlin.AO/0502003 v1 2 Feb 2005.

- G.E.P. Box, "Evolutionary operation: A Method for Increasing Industrial Productivity", *Applied Statistics*, 6 , pp. 81-101, 1957.
- Glover F.," Future paths for Integer Programming and Links to Artificial Intelligence", *Computers and Operations Research*, 5:533-549, 1986.
- Hayek, F.A.: *The Road to Serfdom*, Univ. of Chicago Press, Chicago, 1944.
- Holland, J.: *Adaptation in Natural and Artificial Systems,* Univ. of Michigan Press, Ann Arbor, 1975.
- Karush, W. *Minima of Functions of Several Variables with Inequalities as Side Constraints*. M.Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago, Chicago, Illinois, 1939.
- Kirkpatrick, S., Gelatt, C.D. Jr., and Vecchi, M.P.: "Optimization by Simulated Annealing", *Science*, 220, 4598, 671-680, 1983.
- Kuhn, H.W. and Tucker, A.W.: "Nonlinear Programming", in Neymann, J. (ed) *Proceedings of Second Berkeley Symposium on Mathematical Statistics and Probability,* Univ. of California Press, Berkrley, Calif. pp. 481-492, 1951.
- Metropolis, N. The Beginning of the Monte Carlo Method. *Los Alamos Science*, No. 15, Special Issue, pp. 125-130, 1987.
- Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E.: "Equation of State Calculations by Fast Computing Machines", *J. Chem. Phys.*,21, 6, 1087-1092, 1953.
- Mishra, S.K.: "Some Experiments on Fitting of Gielis Curves by Simulated Annealing and Particle Swarm Methods of Global Optimization", *Social Science Research Network* (SSRN): http://ssrn.com/abstract=913667, Working Papers Series, 2006 (a).
- Mishra, S.K.: "Least Squares Fitting of Chacón-Gielis Curves by the Particle Swarm Method of Optimization", *Social Science Research Network* (SSRN), Working Papers Series, http://ssrn.com/abstract=917762 , 2006 (b).
- Mishra, S.K.: "Performance of Repulsive Particle Swarm Method in Global Optimization of Some Important Test Functions: A Fortran Program" , *Social Science Research Network* (SSRN), Working Papers Series, http://ssrn.com/abstract=924339 , 2006 (c).
- Mishra, S.K.: "Some New Test Functions for Global Optimization and Performance of Repulsive Particle Swarm Method", *Social Science Research Network* (SSRN) Working Papers Series, http://ssrn.com/abstract=927134, 2006 (d).
- Mishra, S.K.: "Repulsive Particle Swarm Method on Some Difficult Test Problems of Global Optimization" ,SSRN: http://ssrn.com/abstract=928538 , 2006 (e).
- Mishra, SK.: "Global Optimization by Differential Evolution and Particle Swarm Methods: Evaluation on Some Benchmark Functions" SSRN: http://ssrn.com/abstract=933827 ,2006 (f)
- Mishra, SK.: " The Barter Method: A New Heuristic for Global Optimization and its Comparison with the Particle Swarm and the Differential Evolution Methods" SSRN: http://ssrn.com/abstract=939180 , 2006(g)
- Nagendra, S.: *Catalogue of Test Problems for Optimization Algorithm Verification*, Technical Report 97-CRD-110, General Electric Company, 1997.
- Nelder, J.A. and Mead, R.: "A Simplex Method for Function Minimization" *Computer Journal*, 7: pp. 308-313, 1964.
- Parsopoulos, K.E. and Vrahatis, M.N., "Recent Approaches to Global Optimization Problems Through Particle Swarm Optimization", *Natural Computing*, 1 (2-3), pp. 235- 306, 2002.
- Prigogine, I. and Strengers, I.: *Order Out of Chaos: Man's New Dialogue with Nature*, Bantam Books, Inc. NY, 1984.
- Silagadge, Z.K.: "Finding Two-Dimensional Peaks", Working Paper, Budkar Insttute of Nuclear Physics, Novosibirsk, Russia, arXive:physics/0402085 V3 11 Mar 2004.
- Simon, H.A.: *Models of Bounded Rationality*, Cambridge Univ. Press, Cambridge, MA, 1982.
- Smith, A.: *The Theory of the Moral Sentiments*, The Adam Smith Institute (2001 e-version), 1759.
- Storn, R. and Price, K: "Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces" : Technical Report, International Computer Science Institute, Berkley, 1995.
- Sumper, D.J.T.: "The Principles of Collective Animal Behaviour", *Phil. Trans. R. Soc. B*. 361, pp. 5-22, 2006.

- Takayama, A.: *Mathematical Economics*, The Dryden Press, Hinsdale, Illinois, 1974.
- Tesfatsion, L.: "Agent-Based Computational Economics: Growing Economies from the Bottom Up", SSRN Working Papers Series, http://ssrn.com/abstract=305080, 2002.
- Törn, A.A and Viitanen, S.: "Topographical Global Optimization using Presampled Points", *J. of Global Optimization*, 5, pp. 267-276, 1994.
- Törn, A.A.: "A search Clustering Approach to Global Optimization" , in Dixon, LCW and Szegö, G.P. (Eds) *Towards Global Optimization – 2*, North Holland, Amsterdam, 1978.
- Tsallis, C. and Stariolo, D.A.: "Generalized Simulated Annealing", *ArXive condmat/9501047 v1 12 Jan,* 1995.
- Valentine, R.H.: *Travel Time Curves in Oblique Structures*, Ph.D. Dissertation, MIT, Mass, 1937.
- Veblen, T.B.: "Why is Economics Not an Evolutionary Science" *The Quarterly Journal of Economics*, 12, 1898.
- Veblen, T.B.: *The Theory of the Leisure Class*, The New American library, NY. (Reprint, 1953), 1899.
- Vesterstrøm, J. and Thomsen, R.: "A comparative Study of Differential Evolution, Particle Swarm Optimization, and Evolutionary Algorithms on Numerical Benchmark Problems", *Congress on Evolutionary Computation, 2004. CEC2004***,** 2, pp. 1980-1987, 2004.
- Whitley, D., Mathias, K., Rana, S. and Dzubera, J.: "Evaluating Evolutionary Algorithms", *Artificial Intelligence*, 85, pp. 245-276, 1996.
- Yao, X. and Liu, Y.: "Fast Evolutionary Programming", in Fogel, LJ, Angeline, PJ and Bäck, T (eds) *Proc. 5th Annual Conf. on Evolutionary programming*, pp. 451-460, MIT Press, Mass, 1996.

```
C     MAIN PROGRAM : PROVIDES TO USE BARTER METHOD, REPULSIVE PARTICLE
C     SWARM METHOD AND DIFFERENTIAL EVOLUTION METHOD.
C     ----------------------------------------------------------------
C     ADJUST THE PARAMETERS SUITABLY IN SUBROUTINES DE, RPS AND BARTER
C     WHEN THE PROGRAM ASKS FOR PARAMETERS, FEED THEM SUITABLY
C     ----------------------------------------------------------------
      PROGRAM DERPSBART
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON /KFF/KF,NFCALL,FTIT ! FUNCTION CODE, NO. OF CALLS & TITLE
      CHARACTER *30  METHOD(3)
      CHARACTER *1 PROCEED
      CHARACTER *70 FTIT
      COMMON /XBASE/XBAS
      COMMON /RNDM/IU,IV ! RANDOM NUMBER GENERATION (IU = 4-DIGIT SEED)
      INTEGER IU,IV
      DIMENSION XX(3,50),KKF(3),MM(3),FMINN(3),XBAS(500,50)
      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 DE OR RPS
      WRITE(*,*)'====================     WARNING    =============== '
      WRITE(*,*)'ADJUST PARAMETERS IN SUBROUTINES DE, RPS & BARTER'
      WRITE(*,*)'FOR BARTER METHOD, DIMENSION MUST BE GREATER THAN ONE'
      WRITE(*,*)'====================     WARNING    =============== '
      METHOD(1)=' : DIFFERENTIAL EVOLUTION'
      METHOD(2)=' : REPULSIVE PARTICLE SWARM'
      METHOD(3)=' : BARTER ALGORITHM'
C     INITIALIZATION. THIS XBAS WILL BE USED IN ALL THREE PROGRAMS TO
C     INITIALIZE THE POPULATION.
      WRITE(*,*)' '
      WRITE(*,*)'FEED RANDOM NUMBER SEED [4-DIGIT ODD INTEGER] TO BEGIN'
      READ(*,*) IU
C     THIS XBAS WILL BE USED IN ALL THE THREE METHODS AS INITIAL X
      DO I=1,500
      DO J=1,50
      CALL RANDOM(RAND)
      XBAS(I,J)=(RAND-0.5D00)*2000 ! RANDOM NUMBER BETWEEN (-1000, 1000)
      ENDDO
      ENDDO
      WRITE(*,*)' ******************************************************'
C     ----------------------------------------------------------------
      DO I=1,3

      IF(I.EQ.1) THEN
      WRITE(*,*)'====== WELCOME TO DE, RPS, BARTER GO PROGRAM ========'
      WRITE(*,*)'TO PROCEED TYPE ANY CHARACTER AND STRIKE ENTER'
      READ(*,*) PROCEED
      CALL DE(M,X,FMINDE,Q0,Q1) ! CALLS DE AND RETURNS OPTIMAL X AND FMIN
      FMIN=FMINDE
      ENDIF
C     ----------------------------------------------------------------
      IF(I.EQ.2) THEN
      WRITE(*,*)' '
      WRITE(*,*)' '
      WRITE(*,*)'==========REPULSIVE PARTICLE SWARM PROGRAM =========='
      WRITE(*,*)'TO PROCEED TYPE ANY CHARACTER AND STRIKE ENTER'
C      READ(*,*) PROCEED
      CALL RPS(M,X,FMINRPS,H0,H1) ! CALLS RPS AND RETURNS OPTIMAL X AND FMIN
      FMIN=FMINRPS
      ENDIF
C     ----------------------------------------------------------------
      IF(I.EQ.3) THEN
      WRITE(*,*)' '
      WRITE(*,*)' '
      WRITE(*,*)'==========BARTER ALGORITHM PROGRAM =========='
      WRITE(*,*)'TO PROCEED TYPE ANY CHARACTER AND STRIKE ENTER'
C      READ(*,*) PROCEED
      CALL BARTER(M,X,FMINEXC,G0,G1)! CALLS RPS;RETURNS OPTIMAL X & FMIN
```

```
      FMIN=FMINEXC
      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=1,3
      WRITE(*,*)'FUNCT CODE=',KKF(I),'  FMIN=',FMINN(I),' : DIM=',MM(I)
      WRITE(*,*)'OPTIMAL DECISION VARIABLES : ',METHOD(I)
      WRITE(*,*)(XX(I,J),J=1,M)
      WRITE(*,*)'//////////////////////////////////////////////////////'
      ENDDO
      WRITE(*,*)'OPTIMIZATION PROGRAM ENDED'
      WRITE(*,*)'******************************************************'
      WRITE(*,*)'MEASURE OF EQUALITY/INEQUALITY'
      WRITE(*,*)'DE: BEFORE AND AFTER OPTIMIZATION = ',Q0,Q1
      WRITE(*,*)'RPS: BEFORE AND AFTER OPTIMIZATION = ',H0,H1
      WRITE(*,*)'BARTER: BEFORE AND AFTER OPTIMIZATION = ',G0,G1
      END
C     ------------------------------------------------------------------
      SUBROUTINE DE(M,A,FBEST,G0,G1)
C     PROGRAM: "DIFFERENTIAL EVOLUTION ALGORITHM" OF GLOBAL OPTIMIZATION
C     THIS METHOD WAS PROPOSED BY R. STORN AND K. PRICE IN 1995. REF --
C     "DIFFERENTIAL EVOLUTION - A SIMPLE AND EFFICIENT ADAPTIVE SCHEME
C     FOR GLOBAL OPTIMIZATION OVER CONTINUOUS SPACES" : TECHNICAL REPORT
C     INTERNATIONAL COMPUTER SCIENCE INSTITUTE, BERKLEY, 1995.
C     PROGRAM BY SK MISHRA, DEPT. OF ECONOMICS, NEHU, SHILLONG (INDIA)
C     ------------------------------------------------------------------
C     PROGRAM DE
      IMPLICIT DOUBLE PRECISION (A-H, O-Z) ! TYPE DECLARATION
      PARAMETER(NMAX=500,MMAX=50) ! MAXIMUM DIMENSION PARAMETERS
      PARAMETER (RX1=0.0, RX2=0.5) ! TO BE ADJUSTED SUITABLY, IF NEEDED
C     RX1 AND RX2 CONTROL THE SCHEME OF CROSSOVER. (0 <= RX1 <= RX2) <=1
C     RX1 DETERMINES THE UPPER LIMIT OF SCHEME 1 (AND LOWER LIMIT OF
C     SCHEME 2; RX2 IS THE UPPER LIMIT OF SCHEME 2 AND LOWER LIMIT OF
C     SCHEME 3. THUS RX1 = .2 AND RX2 = .8 MEANS 0-20% SCHEME1, 20 TO 80
C     PERCENT SCHEME 2 AND THE REST (80 TO 100 %) SCHEME 3.
C     PARAMETER(NCROSS=2) ! CROSS-OVER SCHEME (NCROSS <=0 OR =1 OR =>2)
      PARAMETER(IPRINT=500,EPS=1.D-08)!FOR WATCHING INTERMEDIATE RESULTS
C     IT PRINTS THE INTERMEDIATE RESULTS AFTER EACH IPRINT ITERATION AND
C     EPS DETERMINES ACCURACY FOR TERMINATION. IF EPS= 0, ALL ITERATIONS
C     WOULD BE UNDERGONE EVEN IF NO IMPROVEMENT IN RESULTS IS THERE.
C     ULTIMATELY "DID NOT CONVERGE" IS REOPORTED.
      COMMON /RNDM/IU,IV ! RANDOM NUMBER GENERATION (IU = 4-DIGIT SEED)
      INTEGER IU,IV      ! FOR RANDOM NUMBER GENERATION
      COMMON /KFF/KF,NFCALL,FTIT ! FUNCTION CODE, NO. OF CALLS * TITLE
      COMMON /XBASE/XBAS
      CHARACTER *70 FTIT ! TITLE OF THE FUNCTION
C     ------------------------------------------------------------------
C     THE PROGRAM REQUIRES INPUTS FROM THE USER ON THE FOLLOWING ------
C     (1) FUNCTION CODE (KF), (2) NO. OF VARIABLES IN THE FUNCTION (M);
C     (3) N=POPULATION SIZE (SUGGESTED 10 TIMES OF NO. OF VARIABLES, M,
C         FOR SMALLER PROBLEMS N=100 WORKS VERY WELL);
C     (4) PCROS = PROB. OF CROSS-OVER (SUGGESTED : ABOUT 0.85 TO .99);
C     (5) FACT = SCALE (SUGGESTED 0.5 TO .95 OR  1, ETC);
C     (6) ITER = MAXIMUM NUMBER OF ITERATIONS PERMITTED (5000 OR MORE)
C     (7) RANDOM NUMBER SEED (4 DIGITS INTEGER)
C     ------------------------------------------------------------------
      DIMENSION X(NMAX,MMAX),Y(NMAX,MMAX),A(MMAX),FV(NMAX)
```

```
      DIMENSION IR(3),XBAS(500,50)
C     ------------------------------------------------------------------
C     ------- SELECT THE FUNCTION TO MINIMIZE AND ITS DIMENSION -------
      CALL FSELECT(KF,M,FTIT)
C     SPECIFY OTHER PARAMETERS ----------------------------------------
      WRITE(*,*)'POPULATION SIZE [N] AND NO. OF ITERATIONS [ITER] ?'
      WRITE(*,*)'SUGGESTED : N => 100 OR =>10.M; ITER 10000 OR SO'
      READ(*,*) N,ITER
      WRITE(*,*)'CROSSOVER PROBABILITY [PCROS] AND SCALE [FACT] ?'
      WRITE(*,*)'SUGGESTED : PCROS ABOUT 0.9; FACT=.5 OR LARGER BUT <=1'
      READ(*,*) PCROS,FACT
      WRITE(*,*)'RANDOM NUMBER SEED ?'
      WRITE(*,*)'A FOUR-DIGIT POSITIVE ODD INTEGER, SAY, 1171'
      READ(*,*) IU

      NFCALL=0 ! INITIALIZE COUNTER FOR FUNCTION CALLS
      GBEST=1.D30 ! TO BE USED FOR TERMINATION CRITERION
C     INITIALIZATION : GENERATE X(N,M) RANDOMLY
      DO I=1,N
      DO J=1,M
C      CALL RANDOM(RAND) ! GENERATES INITION X WITHIN
C      X(I,J)=(RAND-.5D00)*2000 ! GENERATES INITION X WITHIN
C     RANDOM NUMBERS BETWEEN -RRANGE AND +RRANGE (BOTH EXCLUSIVE)
      X(I,J)=XBAS(I,J)! TAKES THESE NUMBERS FROM THE MAIN PROGRAM
      ENDDO
      ENDDO
      WRITE(*,*)'COMPUTING --- PLEASE WAIT '
      IPCOUNT=0
      DO 100 ITR=1,ITER  ! ITERATION BEGINS
C     ------------------------------------------------------------------
C     EVALUATE ALL X FOR THE GIVEN FUNCTION
      DO I=1,N
      DO J=1,M
      A(J)=X(I,J)
      ENDDO
      CALL FUNC(A,M,F)
C     STORE FUNCTION VALUES IN FV VECTOR
      FV(I)=F
      ENDDO
      IF(ITR.EQ.1) CALL GINI(FV,N,G0)
C     ------------------------------------------------------------------
C     FIND THE FITTEST (BEST) INDIVIDUAL AT THIS ITERATION
                FBEST=FV(1)
                KB=1
                DO IB=2,N
                     IF(FV(IB).LT.FBEST) THEN
                     FBEST=FV(IB)
                     KB=IB
                     ENDIF
                ENDDO
C     BEST FITNESS VALUE = FBEST : INDIVIDUAL X(KB)
C     ------------------------------------------------------------------
C     GENERATE OFFSPRINGS
      DO I=1,N    ! I LOOP BEGINS
C     INITIALIZE CHILDREN IDENTICAL TO PARENTS; THEY WILL CHANGE LATER
           DO J=1,M
           Y(I,J)=X(I,J)
           ENDDO
C     SELECT RANDOMLY THREE OTHER INDIVIDUALS
   20      DO IRI=1,3  ! IRI LOOP BEGINS
           IR(IRI)=0

           CALL RANDOM(RAND)
            IRJ=INT(RAND*N)+1
C     CHECK THAT THESE THREE INDIVIDUALS ARE DISTICT AND OTHER THAN I
           IF(IRI.EQ.1.AND.IRJ.NE.I) THEN
           IR(IRI)=IRJ
```

```
202:             ENDIF
203:             IF(IRI.EQ.2.AND.IRJ.NE.I.AND.IRJ.NE.IR(1)) THEN
204:             IR(IRI)=IRJ
205:             ENDIF
206:          IF(IRI.EQ.3.AND.IRJ.NE.I.AND.IRJ.NE.IR(1).AND.IRJ.NE.IR(2)) THEN
207:              IR(IRI)=IRJ
208:             ENDIF
209:             ENDDO    ! IRI LOOP ENDS
210: C      CHECK IF ALL THE THREE IR ARE POSITIVE (INTEGERS)
211:           DO IX=1,3
212:           IF(IR(IX).LE.0) THEN
213:           GOTO 20  ! IF NOT THEN REGENERATE
214:           ENDIF
215:           ENDDO
216: C      THREE RANDOMLY CHOSEN INDIVIDUALS DIFFERENT FROM I AND DIFFERENT
217: C      FROM EACH OTHER ARE IR(1),IR(2) AND IR(3)
218: C      ==================== RANDOMIZATION OF NCROSS ==================
219: C      RANDOMIZES NCROSS
220:        NCROSS=0
221:        CALL RANDOM(RAND)
222:        IF(RAND.GT.RX1) NCROSS=1 ! IF RX1=>1, SCHEME 2 NEVER IMPLEMENTED
223:        IF(RAND.GT.RX2) NCROSS=2 ! IF RX2=>1, SCHEME 3 NEVER IMPLEMENTED
224:
225: C      ---------------------- SCHEME 1 ---------------------------------
226: C       NO CROSS OVER, ONLY REPLACEMENT THAT IS PROBABILISTIC
227:           IF(NCROSS.LE.0) THEN
228:           DO J=1,M       ! J LOOP BEGINS
229:           CALL RANDOM(RAND)
230:           IF(RAND.LE.PCROS) THEN ! REPLACE IF RAND < PCROS
231:           A(J)=X(IR(1),J)+(X(IR(2),J)-X(IR(3),J))*FACT ! CANDIDATE CHILD
232:           ENDIF
233:           ENDDO  ! J LOOP ENDS
234:           ENDIF
235:
236: C      ---------------------- SCHEME 2 ---------------------------------
237: C      THE STANDARD CROSSOVER SCHEME
238: C      CROSSOVER SCHEME (EXPONENTIAL) SUGGESTED BY KENNETH PRICE IN HIS
239: C      PERSONAL LETTER TO THE AUTHOR (DATED SEPTEMBER 29, 2006)
240:        IF(NCROSS.EQ.1) THEN
241:          CALL RANDOM(RAND)
242:     1    JR=INT(RAND*M)+1
243:          J=JR
244:     2    A(J)=X(IR(1),J)+FACT*(X(IR(2),J)-X(IR(3),J))
245:     3    J=J+1
246:          IF(J.GT.M) J=1
247:     4    IF(J.EQ.JR) GOTO 10
248:     5    CALL RANDOM(RAND)
249:          IF(PCROS.LE.RAND) GOTO 2
250:     6    A(J)=X(I,J)
251:     7    J=J+1
252:          IF(J.GT.M) J=1
253:     8    IF (J.EQ.JR) GOTO 10
254:     9    GOTO 6
255:    10    CONTINUE
256:        ENDIF
257: C      ---------------------- SCHEME 3 ---------------------------------
258: C      ESPECIALLY SUITABLE TO NON-DECOMPOSABLE (NON-SEPERABLE) FUNCTIONS
259: C      CROSSOVER SCHEME (NEW) SUGGESTED BY KENNETH PRICE IN HIS
260: C      PERSONAL LETTER TO THE AUTHOR (DATED OCTOBER 18, 2006)
261:        IF(NCROSS.GE.2) THEN
262:           CALL RANDOM(RAND)
263:           IF(RAND.LE.PCROS) THEN
264:              CALL NORMAL(RN)
265:              DO J=1,M
266:              A(J)=X(I,J)+(X(IR(1),J)+ X(IR(2),J)-2*X(I,J))*RN
267:              ENDDO
268:            ELSE
```

```
269:              DO J=1,M
270:              A(J)=X(I,J)+(X(IR(1),J)- X(IR(2),J))! FACT ASSUMED TO BE 1
271:              ENDDO
272:             ENDIF
273:         ENDIF
274: C       ---------------------------------------------------------------
275:             CALL FUNC(A,M,F) ! EVALUATE THE OFFSPRING
276:             IF(F.LT.FV(I)) THEN ! IF BETTER, REPLACE PARENTS BY THE CHILD
277:             FV(I)=F
278:             DO J=1,M
279:             Y(I,J)=A(J)
280:             ENDDO
281:             ENDIF
282:         ENDDO    ! I LOOP ENDS
283:         DO I=1,N
284:         DO J=1,M
285:         X(I,J)=Y(I,J) ! NEW GENERATION IS A MIX OF BETTER PARENTS AND
286: C                       BETTER CHILDREN
287:         ENDDO
288:         ENDDO
289:         IPCOUNT=IPCOUNT+1
290:         IF(IPCOUNT.EQ.IPRINT) THEN
291:         DO J=1,M
292:         A(J)=X(KB,J)
293:         ENDDO
294:         WRITE(*,*)(X(KB,J),J=1,M),'  FBEST UPTO NOW = ',FBEST
295:         WRITE(*,*)'TOTAL NUMBER OF FUNCTION CALLS =',NFCALL
296:              IF(DABS(FBEST-GBEST).LT.EPS) THEN
297:              WRITE(*,*) FTIT
298:              WRITE(*,*)'COMPUTATION OVER'
299:              GOTO 999
300:              ELSE
301:              GBEST=FBEST
302:              ENDIF
303:         IPCOUNT=0
304:         ENDIF
305: C       ---------------------------------------------------------------
306:   100 ENDDO    ! ITERATION ENDS : GO FOR NEXT ITERATION, IF APPLICABLE
307: C       ---------------------------------------------------------------
308:         WRITE(*,*)'DID NOT CONVERGE. REDUCE EPS OR RAISE ITER OR DO BOTH'
309:         WRITE(*,*)'INCREASE N, PCROS, OR SCALE FACTOR (FACT)'
310:   999 CALL GINI(FV,N,G1)
311:         RETURN
312:         END
313: C       ---------------------------------------------------------------
314:         SUBROUTINE NORMAL(R)
315: C       PROGRAM TO GENERATE N(0,1) FROM RECTANGULAR RANDOM NUMBERS
316: C       IT USES BOX-MULLER VARIATE TRANSFORMATION FOR THIS PURPOSE.
317: C       ---------------------------------------------------------------
318: C       ----- BOX-MULLER METHOD BY GEP BOX AND ME MULLER (1958) ---------
319: C       BOX, G. E. P. AND MULLER, M. E. "A NOTE ON THE GENERATION OF
320: C       RANDOM NORMAL DEVIATES." ANN. MATH. STAT. 29, 610-611, 1958.
321: C       IF U1 AND U2 ARE UNIFORMLY DISTRIBUTED RANDOM NUMBERS (0,1),
322: C       THEN X=[(-2*LN(U1))**.5]*(COS(2*PI*U2) IS N(0,1)
323: C       ALSO,  X=[(-2*LN(U1))**.5]*(SIN(2*PI*U2) IS N(0,1)
324: C       PI = 4*ARCTAN(1.0)= 3.14159265358979323846264338327795
325: C       2*PI = 6.28318530717958647692528676655 9
326: C       ---------------------------------------------------------------
327:         IMPLICIT DOUBLE PRECISION (A-H,O-Z)
328:         COMMON /RNDM/IU,IV
329:         INTEGER IU,IV
330: C       ---------------------------------------------------------------
331:
332: C       ---------------------------------------------------------------
333:         CALL RANDOM(RAND) ! INVOKES RANDOM TO GENERATE UNIFORM RAND [0, 1]
334:         U1=RAND ! U1 IS UNIFORMLY DISTRIBUTED [0, 1]
335:         CALL RANDOM(RAND) ! INVOKES RANDOM TO GENERATE UNIFORM RAND [0, 1]
```

```
      U2=RAND ! U1 IS UNIFORMLY DISTRIBUTED [0, 1]
      R=DSQRT(-2.D0*DLOG(U1))
      R=R*DCOS(U2*6.28318530717958647692528676655900D00)
C     R=R*DCOS(U2*6.28318530718D00)
      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
       RAND=REAL(RAND1)
       IV=IU*65539
       IF(IV.LT.0) THEN
       IV=IV+2147483647+1
       ENDIF
       RAND=IV
       IU=IV
       RAND=RAND*0.4656613E-09
       RAND1= RAND
       RETURN
       END
C     ======================================================================
C     ======================================================================
      SUBROUTINE RPS(M,BST,FMINIM,H0,H1)
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=50,MX=50,NSTEP=11,ITRN=10000,NSIGMA=1,ITOP=3)
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
      COMMON /XBASE/XBAS
      INTEGER IU,IV
      CHARACTER *70 FTIT
      DIMENSION X(N,MX),V(N,MX),A(MX),VI(MX),XBAS(500,50)
      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,EPSI /.5D0,.5D0,5.D-04,.5D00,1.D-03,1.D-08/
C     ------------------------------------------------------------------
C     CALL SUBROUTINE FOR CHOOSING FUNCTION (KF) AND ITS DIMENSION (M)
```

```
403: C       CALL FSELECT(KF,M,FTIT)
404: C      ---------------------------------------------------------------------
405:        GGBEST=1.D30 ! TO BE USED FOR TERMINATION CRITERION
406:        LCOUNT=0
407:        NFCALL=0
408:        WRITE(*,*)'4-DIGITS SEED FOR RANDOM NUMBER GENERATION'
409:        WRITE(*,*)'A FOUR-DIGIT POSITIVE ODD INTEGER, SAY, 1171'
410: C       READ(*,*) IU
411:        IU=1111 ! COULD BE ANY OTHER 4 OR 5 DIGIT ODD INTEGER
412:        FMIN=1.0E30
413: C      GENERATE N-SIZE POPULATION OF M-TUPLE PARAMETERS X(I,J) RANDOMLY
414:        DO I=1,N
415:          DO J=1,M
416: C          CALL RANDOM(RAND)  !GENERATE X WITHIN
417: C           X(I,J)=(RAND-0.5D00)*2000   ! GENERATE X WITHIN
418: C      WE GENERATE RANDOM(-1000,1000). HERE MULTIPLIER IS 200.
419: C      TINKERING IN SOME CASES MAY BE NEEDED
420:        X(I,J)=XBAS(I,J)  ! GET X FROM OUTSIDE
421:           ENDDO
422:          F(I)=1.0D30
423:        ENDDO
424: C      INITIALISE VELOCITIES V(I) FOR EACH INDIVIDUAL IN THE POPULATION
425:        DO I=1,N
426:        DO J=1,M
427:        CALL RANDOM(RAND)
428:         V(I,J)=(RAND-0.5D+00)
429: C         V(I,J)=RAND
430:        ENDDO
431:        ENDDO
432:        DO 100 ITER=1,ITRN
433: C      LET EACH INDIVIDUAL SEARCH FOR THE BEST IN ITS NEIGHBOURHOOD
434:          DO I=1,N
435:             DO J=1,M
436:             A(J)=X(I,J)
437:             VI(J)=V(I,J)
438:             ENDDO
439:             CALL LSRCH(A,M,VI,NSTEP,FI)
440:             IF(FI.LT.F(I)) THEN
441:              F(I)=FI
442:              DO IN=1,M
443:              BST(IN)=A(IN)
444:              ENDDO
445: C      F(I) CONTAINS THE LOCAL BEST VALUE OF FUNCTION FOR ITH INDIVIDUAL
446: C      XX(I,J) IS THE M-TUPLE VALUE OF X ASSOCIATED WITH LOCAL BEST F(I)
447:                DO J=1,M
448:                XX(I,J)=A(J)
449:                ENDDO
450:                ENDIF
451:          ENDDO
452:        IF(ITER.EQ.1) CALL GINI(F,N,H0)
453: C       NOW LET EVERY INDIVIDUAL RANDOMLY COSULT NN(<<N) COLLEAGUES AND
454: C       FIND THE BEST AMONG THEM
455:        DO I=1,N
456: C      ---------------------------------------------------------------------
457:        IF(ITOP.GE.3) THEN
458: C      RANDOM TOPOLOGY ****************************************
459: C      CHOOSE NN COLLEAGUES RANDOMLY AND FIND THE BEST AMONG THEM
460:            BEST=1.0D30
461:             DO II=1,NN
462:                   CALL RANDOM(RAND)
463:                NF=INT(RAND*N)+1
464:                 IF(BEST.GT.F(NF)) THEN
465:                  BEST=F(NF)
466:                 NFBEST=NF
467:                  ENDIF
468:              ENDDO
469:        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.100) 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
      IF(DABS(FMIN-GGBEST).LT.EPSI) THEN
           WRITE(*,*)'COMPUTATION OVER'
           FMINIM=FMIN
           GOTO 999
           ELSE
           GGBEST=FMIN
           ENDIF
      ENDIF
      LCOUNT=LCOUNT+1
  100 CONTINUE
      WRITE(*,*)'COMPUTATION OVER:',FTIT
      FMINIM=FMIN
  999 CALL GINI(F,N,H1)
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE LSRCH(A,M,VI,NSTEP,FI)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      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
```

```
604: C     EITHER PURE OR HYBRIDIZED
605:       SUBROUTINE NEIGHBOR(I,N,J,K)
606:       IF(I-1.GE.1 .AND. I.LT.N) THEN
607:       J=I-1
608:       K=I+1
609:       ELSE
610:       IF(I-1.LT.1) THEN
611:       J=N-I+1
612:       K=I+1
613:       ENDIF
614:       IF(I.EQ.N) THEN
615:       J=I-1
616:       K=1
617:       ENDIF
618:       ENDIF
619:       RETURN
620:       END
621: C     ---------------------------------------------------------------
622: C     BARTER ALGORITHM
623: C     ---------------------------------------------------------------
624: C     *************** THIS METHOD IS PROPOSED BY SK MISHRA ************
625: C     PROGRAM BY SK MISHRA, DEPT. OF ECONOMICS, NEHU, SHILLONG (INDIA)
626: C     ---------------------------------------------------------------
627:       SUBROUTINE BARTER(M,BEST,FBEST,G0,G1)
628:       IMPLICIT DOUBLE PRECISION (A-H, O-Z) ! TYPE DECLARATION
629:       PARAMETER(IPRINT=500, EPS=1.D-08)
630:       COMMON /RNDM/IU,IV ! RANDOM NUMBER GENERATION (IU = 4-DIGIT SEED)
631:       COMMON /XBASE/XBAS
632:       INTEGER IU,IV      ! FOR RANDOM NUMBER GENERATION
633:       COMMON /KFF/KF,NFCALL,FTIT ! FUNCTION CODE,NO. OF CALLS & TITLE
634:       CHARACTER *70 FTIT ! TITLE OF THE FUNCTION
635: C     ---------------------------------------------------------------
636: C     ---------------------- VERY IMPORTANT -------------------------
637: C     FOR THE BARTER METHOD THE DIMENSION M MUST EXCEED UNITY : OR M =>2
638: C     ---------------------------------------------------------------
639:       DIMENSION X(500,50),FV(500),A(50),B(50),BEST(50),XBAS(500,50)
640: C     ---------------------------------------------------------------
641: C     ------- SELECT THE FUNCTION TO MINIMIZE AND ITS DIMENSION -------
642: C     CALL FSELECT(KF,M,FTIT)
643: C     SPECIFY OTHER PARAMETERS ----------------------------------------
644:       WRITE(*,*)'POPULATION SIZE [N] AND NO. OF ITERATIONS [ITER] ?'
645: C     READ(*,*) N,ITER
646: C     ---------------------------------------------------------------
647:       NBART=0 ! NO. OF TIMES  THE BARTER WAS SUCCESSFUL
648:       NTRY=0  ! NO. OF TIMES  ATTEMPT WAS MADE TO BARTER
649:       N=M*10
650:       IF(N.LT.100) N=100
651:       ITER=10000
652:       WRITE(*,*)'RANDOM NUMBER SEED ?'
653: C      READ(*,*) IU
654:       IU=7113 ! COULD BE ANY OTHER 4 OR 5 DIGIT ODD INTEGER
655: C     ---------------------------------------------------------------
656:       NFCALL=0 ! INITIALIZE COUNTER FOR FUNCTION CALLS
657:       GBEST=1.D30 ! TO BE USED FOR TERMINATION CRITERION
658: C     INITIALIZATION : GENERATE X(N,M) RANDOMLY
659:       DO I=1,N
660:       DO J=1,M
661: C      CALL RANDOM(RAND)    ! GENERATE INITIAL X WITHIN
662: C      X(I,J)=(RAND-.5D00)*2000  ! GENERATE INITIAL X WITHIN
663: C     RANDOM NUMBERS BETWEEN -RRANGE AND +RRANGE (BOTH EXCLUSIVE)
664:       X(I,J)=XBAS(I,J) ! : GET X FROM OUTSIDE
665:       ENDDO
666:       ENDDO
667:       WRITE(*,*)'COMPUTING --- PLEASE WAIT '
668:       IPCOUNT=0
669:       DO 100 ITR=1,ITER  ! ITERATION BEGINS
670:
```

```
671: C     EVALUATE ALL X FOR THE GIVEN FUNCTION
672:       DO I=1,N
673:         DO J=1,M
674:         A(J)=X(I,J)
675:         ENDDO
676:       CALL FUNC(A,M,FA)
677:       FV(I)=FA
678: C     ------------------------------------------
679:        CALL SEARCH(A,M,FI)
680: C     STORE FUNCTION VALUES IN FV VECTOR
681:          IF(FI.LT.FV(I)) THEN
682:            DO J=1,M
683:            X(I,J)=A(J)
684:            ENDDO
685:          FV(I)=FI
686:          ENDIF
687:       ENDDO
688:       IF(ITR.EQ.1) CALL GINI(FV,N,G0)
689: C     ------------------------------------------------------------------
690:       DO I=1,N
691:       NTRY=NTRY+1
692: C     CHOOSE IB TH INDIVIDUAL RANDOMLY
693:       CALL RANDOM(RAND)
694:       IB=INT(RAND*N)+1 !THE RANDOM INDIVIDUAL
695: C     IB=2*M+1
696: C     STORE ITH IN A AND RANDOMLY SELECTED INDIVIDUAL IN B
697:          DO J=1,M
698:          A(J)=X(I,J)
699:          B(J)=X(IB,J) ! OF THE INDIVIDUAL RANDOMLY SELECTED
700:          ENDDO
701: C     CHOSE AN INDEX BETWEEN 1 AND M RANDOMLY
702:            CALL RANDOM(RAND)
703:           JA=INT(RAND*M)+1
704: C     CHOOSE ANOTHER INDEX RANDOMLY : MUST BE DIFFERENT FROM JA
705:    1      CALL RANDOM(RAND)
706:           JB=INT(RAND*M)+1
707:           IF(JA.EQ.JB) GOTO 1
708: C     EXCHANGE A(JA) WITH B(JB)
709:            TEMP1=A(JA)
710:            TEMP2=B(JB)
711:              CALL NORMAL(RN)! OBTAIN STANDARD NORMAL  RANDOM NUMBER
712:            A(JB)=A(JB)+RN*TEMP2
713:            B(JB)=B(JB)-RN*TEMP2
714:            A(JA)=A(JA)-RN*TEMP1
715:            B(JA)=B(JA)+RN*TEMP1
716: C     EVALUATE A AND B VECTORS
717:             CALL FUNC(A,M,FA)
718:             CALL FUNC(B,M,FB)
719: C     CHECK IF FA < FV(I) AND FB < FV(IB)
720:            IF(FA.LT.FV(I) .AND. FB.LT.FV(IB)) THEN
721:            NBART=NBART+1
722:            FV(I)=FA
723:            FV(IB)=FB
724:            DO J=1,M
725:            X(I,J)=A(J)
726:            X(IB,J)=B(J)
727:            ENDDO
728:            ENDIF
729:       ENDDO
730: C     ------------------------------------------------------------------
731: C     FIND THE  BEST
732:       FBEST=1.D30
733:       DO I=1,N
734:       IF(FV(I).LT.FBEST) THEN
735:       FBEST=FV(I)
736:       KB=I
737:       ENDIF
```

```
738:       ENDDO
739:       DO J=1,M
740:       BEST(J)=X(KB,J)
741:       ENDDO
742: C     ------------------------------------------------------------------
743:       IPCOUNT=IPCOUNT+1
744:       IF(IPCOUNT.EQ.IPRINT) THEN
745:       WRITE(*,*)(BEST(J),J=1,M),'  FBEST UPTO NOW = ',FBEST
746:       WRITE(*,*)'TOTAL NUMBER OF FUNCTION CALLS =',NFCALL
747:            IF(DABS(FBEST-GBEST).LT.EPS) THEN
748:            WRITE(*,*) FTIT
749:            WRITE(*,*)'COMPUTATION OVER'
750:            GOTO 999
751:            ELSE
752:            GBEST=FBEST
753:            ENDIF
754:       IPCOUNT=0
755:       ENDIF
756: C     ------------------------------------------------------------------
757:   100 ENDDO   ! ITERATION ENDS : GO FOR NEXT ITERATION, IF APPLICABLE
758: C     ------------------------------------------------------------------
759:       WRITE(*,*)'DID NOT CONVERGE. REDUCE EPS  -- MAY MAKE IT ZERO  --'
760:       WRITE(*,*)'RAISE ITER OR DO BOTH OR INCREASE N, POPULATION SIZE'
761:   999 CALL GINI(FV,N,G1)
762:       PSUCCESS=NBART/DFLOAT(NTRY)*100.0
763:       WRITE(*,*) 'NO. OF TIMES AN ATTEPT TO BARTER WAS MADE = ', NTRY
764:       WRITE(*,*) 'NO. OF TIMES BARTER WAS SUCCESSFUL =' , NBART
765:       WRITE(*,*) 'PERCENTAGE SUCCESS OF BARTER ATTEMPTS = ', PSUCCESS
766:       RETURN
767:       END
768: C     ------------------------------------------------------------------
769:       SUBROUTINE SEARCH(A,M,FI)
770:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
771:       COMMON /RNDM/IU,IV
772:       INTEGER IU,IV
773:       DIMENSION A(*),B(100)
774:       NSTEP=11
775:       AMN=1.0D30
776:       DO J=1,NSTEP
777:          DO JJ=1,M
778:          CALL RANDOM(RAND)
779:          B(JJ)=A(JJ)+(J-(NSTEP/2)-1)*RAND*0.0001D00
780:          ENDDO
781:       CALL FUNC(B,M,FI)
782:         IF(FI.LT.AMN) THEN
783:         AMN=FI
784:         DO JJ=1,M
785:         A(JJ)=B(JJ)
786:         ENDDO
787:         ENDIF
788:       ENDDO
789:       FI=AMN
790:       RETURN
791:       END
792: C     ------------------------------------------------------------------
793:       SUBROUTINE GINI(F,N,G)
794:       PARAMETER (K=1) !K=1 GINI COEFFICENT; K=2 COEFFICIENT OF VARIATION
795: C     THIS PROGRAM COMPUTES MEASURE OF INEQUALITY
796: C     IF K =1 GET THE GINI COEFFICIENT. IF K=2 GET COEFF OF VARIATIONE
797:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
798:       DIMENSION F(*)
799:       S=0.D0
800:       DO I=1,N
801:       S=S+F(I)
802:       ENDDO
803:       S=S/N
804:       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
      WRITE(*,*)'-----------------------------------------------------'
      DATA TIT(1)/'KF=1 NEW FUNCTION(N#1) 2-VARIABLES M=2'/
      DATA TIT(2)/'KF=2 NEW FUNCTION(N#2) 2-VARIABLES M=2'/
      DATA TIT(3)/'KF=3 NEW FUNCTION(N#3) 2-VARIABLES M=2'/
      DATA TIT(4)/'KF=4 NEW FUNCTION(N#4) 2-VARIABLES M=2'/
      DATA TIT(5)/'KF=5 NEW QUINTIC FUNCTION M-VARIABLES M=?'/
      DATA TIT(6)/'KF=6 NEW NEEDLE-EYE FUNCTION (N#6) M-VARIABLES M=?'/
      DATA TIT(7)/'KF=7 NEW ZERO-SUM FUNCTION (N#7) M-VARIABLES M=?'/
      DATA TIT(8)/'KF=8 CORANA FUNCTION 4-VARIABLES M=4'/
      DATA TIT(9)/'KF=9 MODIFIED RCOS FUNCTION 2-VARIABLES M=2'/
      DATA TIT(10)/'KF=10 FREUDENSTEIN ROTH FUNCTION 2-VARIABLES M=2'/
      DATA TIT(11)/'KF=11 ANNS XOR FUNCTION 9-VARIABLES M=9'/
      DATA TIT(12)/'KF=12 PERM FUNCTION #1 (SET BETA) 4-VARIABLES M=4'/
      DATA TIT(13)/'KF=13 PERM FUNCTION #2 (SET BETA) M-VARIABLES M=?'/
      DATA TIT(14)/'KF=14 POWER-SUM FUNCTION 4-VARIABLES M=4'/
      DATA TIT(15)/'KF=15 GOLDSTEIN PRICE FUNCTION 2-VARIABLES M=2'/
      DATA TIT(16)/'KF=16 BUKIN 6TH FUNCTION 2-VARIABLES M=2'/
      DATA TIT(17)/'KF=17 NEW FUNCTION (N#8) 2-VARIABLES M=2'/
      DATA TIT(18)/'KF=18 DEFL CORRUG SPRING FUNCTION M-VARIABLES M=?'/
      DATA TIT(19)/'KF=19 NEW FACTORIAL FUNCTION M-VARIABLES M=?'/
      DATA TIT(20)/'KF=20 NEW DECANOMIAL FUNCTION 2-VARIABLES M=2'/
      DATA TIT(21)/'KF=21 JUDGE FUNCTION 2-VARIABLES M=2'/
      DATA TIT(22)/'KF=22 NEW DODECAL FUNCTION 3-VARIABLES M=3'/
      DATA TIT(23)/'KF=23 NEW SUM-EQ-PROD FUNCTION 2-VARIABLES M=2'/
      DATA TIT(24)/'KF=24 NEW AM-EQ-GM FUNCTION M-VARIABLES M=?'/
      DATA TIT(25)/'KF=25 YAO-LIU FUNCTION#2 M-VARIABLES M=?'/
      DATA TIT(26)/'KF=26 YAO-LIU FUNCTION#3 M-VARIABLES M=?'/
      DATA TIT(27)/'KF=27 YAO-LIU FUNCTION#4 M-VARIABLES M=?'/
      DATA TIT(28)/'KF=28 YAO-LIU FUNCTION#6 M-VARIABLES M=?'/
      DATA TIT(29)/'KF=29 YAO-LIU FUNCTION#7 M-VARIABLES M=?'/
      DATA TIT(30)/'KF=30 YAO-LIU FUNCTION#12 M-VARIABLES M=?'/
      DATA TIT(31)/'KF=31 YAO-LIU FUNCTION#13 M-VARIABLES M=?'/
      DATA TIT(32)/'KF=32 YAO-LIU FUNCTION#14 2-VARIABLES M=2'/
      DATA TIT(33)/'KF=33 YAO-LIU FUNCTION#15 4-VARIABLES M=4'/
      DATA TIT(34)/'KF=34 WOOD FUNCTION : 4-VARIABLES M=4'/
      DATA TIT(35)/'KF=35 FENTON-EASON FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(36)/'KF=36 HOUGEN FUNCTION : 5-VARIABLES M=5'/
      DATA TIT(37)/'KF=37 GIUNTA FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(38)/'KF=38 EGGHOLDER FUNCTION : M-VARIABLES M=?'/
      DATA TIT(39)/'KF=39 TRID FUNCTION : M-VARIABLES M=?'/
      DATA TIT(40)/'KF=40 GRIEWANK FUNCTION : M-VARIABLES M=?'/
      DATA TIT(41)/'KF=41 WEIERSTRASS FUNCTION : M-VARIABLES M=?'/
      DATA TIT(42)/'KF=42 LEVY-3 FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(43)/'KF=43 LEVY-5 FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(44)/'KF=44 LEVY-8 FUNCTION : 3-VARIABLES M=3'/
      DATA TIT(45)/'KF=45 RASTRIGIN FUNCTION : M-VARIABLES M=?'/
      DATA TIT(46)/'KF=46 ACKLEY FUNCTION : M-VARIABLES M=?'/
      DATA TIT(47)/'KF=47 MICHALEWICZ FUNCTION : M-VARIABLES M=?'/
      DATA TIT(48)/'KF=48 SCHWEFEL FUNCTION : M-VARIABLES M=?'/
      DATA TIT(49)/'KF=49 SHUBERT FUNCTION : 2-VARIABLES M=2'/
```

```
872:       DATA TIT(50)/'KF=50 DIXON-PRICE FUNCTION : M-VARIABLES M=?'/
873:       DATA TIT(51)/'KF=51 SHEKEL FUNCTION : 4-VARIABLES M=4'/
874:       DATA TIT(52)/'KF=52 PAVIANI FUNCTION : 10-VARIABLES M=10'/
875:       DATA TIT(53)/'KF=53 BRANIN FUNCTION#1 : 2-VARIABLES M=2'/
876:       DATA TIT(54)/'KF=54 BRANIN FUNCTION#2 : 2-VARIABLES M=2'/
877:       DATA TIT(55)/'KF=55 BOHACHEVSKY FUNCTION#1 : 2-VARIABLES M=2'/
878:       DATA TIT(56)/'KF=56 BOHACHEVSKY FUNCTION#2 : 2-VARIABLES M=2'/
879:       DATA TIT(57)/'KF=57 BOHACHEVSKY FUNCTION#3 : 2-VARIABLES M=2'/
880:       DATA TIT(58)/'KF=58 EASOM FUNCTION : 2-VARIABLES M=2'/
881:       DATA TIT(59)/'KF=59 ROSENBROCK FUNCTION : M-VARIABLES M=?'/
882:       DATA TIT(60)/'KF=60 CROSS-LEGGED TABLE FUNCTION:2-VARIABLES M=2'/
883:       DATA TIT(61)/'KF=61 CROSS FUNCTION : 2-VARIABLES M=2'/
884:       DATA TIT(62)/'KF=62 CROSS-IN-TRAY FUNCTION : 2-VARIABLES M=2'/
885:       DATA TIT(63)/'KF=63 CROWNED CROSS FUNCTION : 2-VARIABLES M=2'/
886:       DATA TIT(64)/'KF=64 TT-HOLDER FUNCTION : 2-VARIABLES M=2'/
887:       DATA TIT(65)/'KF=65 HOLDER-TABLE FUNCTION : 2-VARIABLES M=2'/
888:       DATA TIT(66)/'KF=66 CARROM-TABLE FUNCTION : 2-VARIABLES M=2'/
889:       DATA TIT(67)/'KF=67 PENHOLDER FUNCTION : 2-VARIABLES M=2'/
890:       DATA TIT(68)/'KF=68 BIRD FUNCTION : 2-VARIABLES M=2'/
891:       DATA TIT(69)/'KF=69 CHICHINADZE FUNCTION : 2-VARIABLES M=2'/
892:       DATA TIT(70)/'KF=70 MCCORMICK FUNCTION : 2-VARIABLES M=2'/
893:       DATA TIT(71)/'KF=71 GLANKWAHMDEE FUNCTION : 5-VARIABLES M=5'/
894:       DATA TIT(72)/'KF=72 FLETCHER-POWELL FUNCTION : M-VARIABLES M=?'/
895:       DATA TIT(73)/'KF=73 POWELL FUNCTION: M-VARIABLES M (MULT OF 4)=?'/
896:       DATA TIT(74)/'KF=74 HARTMANN FUNCTION: 3-VARIABLES M=3'/
897:       DATA TIT(75)/'KF=75 COLVILLE FUNCTION: 4-VARIABLES M=4'/
898:       DATA TIT(76)/'KF=76 HIMMELBLAU FUNCTION: 2-VARIABLES M=2'/
899:       DATA TIT(77)/'KF=77 BEALE FUNCTION: 2-VARIABLES M=2'/
900:       DATA TIT(78)/'KF=78 BOOTH FUNCTION: 2-VARIABLES M=2'/
901:       DATA TIT(79)/'KF=79 HUMP FUNCTION: 2-VARIABLES M=2'/
902:       DATA TIT(80)/'KF=80 MATYAS FUNCTION: 2-VARIABLES M=2'/
903:       DATA TIT(81)/'KF=81 MISHRA_1 FUNCTION: M-VARIABLES M=?'/
904:       DATA TIT(82)/'KF=82 MISHRA_2 FUNCTION: M-VARIABLES M=?'/
905:       DATA TIT(83)/'KF=83 ZAKHAROV FUNCTION: M-VARIABLES M=?'/
906:       DATA TIT(84)/'KF=84 MULTIMOD FUNCTION: 2-VARIABLES M=2'/
907:       DATA TIT(85)/'KF=85 NONLINEAR FUNCTION: M-VARIABLES M=?'/
908:       DATA TIT(86)/'KF=86 QUADRATIC FUNCTION: 2-VARIABLES M=2'/
909:       DATA TIT(87)/'KF=87 TRIGON FUNCTION: M-VARIABLES M=?'/
910:       DATA TIT(88)/'KF=88 COMPOUND FUNCTION: 4-VARIABLES M=4'/
911: C     ------------------------------------------------------------------
912:       DO I=1,88
913:       WRITE(*,*)TIT(I)
914:       ENDDO
915:       WRITE(*,*)'-------------------------------------------------'
916:       WRITE(*,*)'FUNCTION CODE [KF] AND NO. OF VARIABLES [M] ?'
917:       READ(*,*) KF,M
918:       FTIT=TIT(KF) ! STORE THE NAME OF THE CHOSEN FUNCTION IN FTIT
919:       RETURN
920:       END
921: C     ------------------------------------------------------------------
922:       SUBROUTINE FUNC(X,M,F)
923: C     TEST FUNCTIONS FOR GLOBAL OPTIMIZATION PROGRAM
924:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
925:       COMMON /RNDM/IU,IV
926:       COMMON /KFF/KF,NFCALL,FTIT
927:       INTEGER IU,IV
928:       DIMENSION X(*)
929:       CHARACTER *70 FTIT
930:       PI=4.D+00*DATAN(1.D+00)! DEFINING THE VALUE OF PI
931:       NFCALL=NFCALL+1 ! INCREMENT TO NUMBER OF FUNCTION CALLS
932: C     KF IS THE CODE OF THE TEST FUNCTION
933: C     ------------------------------------------------------------------
934:       IF(KF.EQ.1) THEN
935: C     FUNCTION #1 MIN AT -0.18467 APPROX AT (-8.4666, -10) APPROX
936:       F=0.D00
937:       DO I=1,M
938:       IF(DABS(X(I)).GT.10.D00) THEN
```

```
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F=DABS(DCOS(DSQRT(DABS(X(1)**2+X(2)))))**0.5 +0.01*X(1)+.01*X(2)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.2) THEN
C     FUNCTION #2 MIN = -0.199409 APPROX AT (-9.94112, -10) APPROX
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F=DABS(DSIN(DSQRT(DABS(X(1)**2+X(2)))))**0.5 +0.01*X(1)+.01*X(2)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.3) THEN
C     FUNCTION #3 MIN = -1.01983 APPROX AT (-1.98682, -10.00000) APPROX
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F1=DSIN(( DCOS(X(1))+DCOS(X(2)) )**2)**2
      F2=DCOS(( DSIN(X(1))+DSIN(X(2)) )**2)**2
      F=(F1+F2+X(1))**2 ! IS MULTIMODAL
      F=F+ 0.01*X(1)+0.1*X(2) ! MAKES UNIMODAL
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.4) THEN
C     FUNCTION  #4 MIN = -2.28395 APPROX AT (2.88631, 1.82326) APPROX
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F1=DSIN((DCOS(X(1))+DCOS(X(2)))**2)**2
      F2=DCOS((DSIN(X(1))+DSIN(X(2)))**2)**2
      F3=-DLOG((F1-F2+X(1))**2 )
      F=F3+0.1D00*(X(1)-1.D00)**2+0.1D00*(X(2)-1.D00)**2
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.5) THEN
C     QUINTIC FUNCTION:GLOBAL MINIMA,EXTREMELY DIFFICULT TO OPTIMIZE
C     MIN VALUE = 0 AT PERMUTATION OF (2, 2,..., 2, -1, -1, ..., -1,
C     -0.402627941) GIVES MIN F = 0.
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      CALL QUINTIC(M,F,X)
      RETURN
      ENDIF
```

```
1006: C     ------------------------------------------------------------------
1007:       IF(KF.EQ.6) THEN
1008: C     NEEDLE-EYE FUNCTION M=>1;
1009: C     MIN = 1 IF ALL ABS(X) ARE SMALLER THAN THE EYE
1010: C     SMALLER THE VALUE OF ZZ, MORE DIFFICULT TO ENTER THE EYE
1011: C     LARGER THE VALUE OF M, MORE DIFFICULT TO FIND THE OPTIMUM
1012:       F=0.D00
1013:       EYE=0.000001D00
1014:       FP=0.D00
1015:       DO I=1,M
1016:       IF(DABS(X(I)).GT.EYE) THEN
1017:       FP=1.D00
1018:        F=F+100.D00+DABS(X(I))
1019:       ELSE
1020:       F=F+1.D00
1021:       ENDIF
1022:       ENDDO
1023:       IF(FP.EQ.0.D00) F=F/M
1024:       RETURN
1025:       ENDIF
1026: C     ------------------------------------------------------------------
1027:       IF(KF.EQ.7) THEN
1028: C     ZERO SUM FUNCTION : MIN  = 0 AT SUM(X(I))=0
1029:       F=0.D00
1030:       DO I=1,M
1031:       IF(DABS(X(I)).GT.10.D00) THEN
1032:       CALL RANDOM(RAND)
1033:       X(I)=(RAND-0.5D00)*20
1034:       ENDIF
1035:       ENDDO
1036:       SUM=0.D00
1037:       DO I=1,M
1038:       SUM=SUM+X(I)
1039:       ENDDO
1040:       IF(SUM.NE.0.D00) F=1.D00+(10000*DABS(SUM))**0.5
1041:       RETURN
1042:       ENDIF
1043: C     ------------------------------------------------------------------
1044:       IF(KF.EQ.8) THEN
1045: C     CORANA FUNCTION : MIN = 0 AT (0, 0, 0, 0) APPROX
1046:       F=0.D00
1047:       DO I=1,M
1048:       IF(DABS(X(I)).GT.1000.D00) THEN
1049:       CALL RANDOM(RAND)
1050:       X(I)=(RAND-0.5D00)*2000
1051:       ENDIF
1052:       ENDDO
1053:       DO J=1,M
1054:         IF(J.EQ.1) DJ=1.D00
1055:         IF(J.EQ.2) DJ=1000.D00
1056:         IF(J.EQ.3) DJ=10.D00
1057:         IF(J.EQ.4) DJ=100.D00
1058:             ISGNXJ=1
1059:             IF(X(J).LT.0.D00) ISGNXJ=-1
1060:             ZJ=(DABS(X(J)/0.2D00)+0.49999)*ISGNXJ*0.2D00
1061:             ISGNZJ=1
1062:             IF(ZJ.LT.0.D00) ISGNZJ=-1
1063:             IF(DABS(X(J)-ZJ).LT.0.05D00) THEN
1064:             F=F+0.15D00*(ZJ-0.05D00*ISGNZJ)**2 * DJ
1065:             ELSE
1066:             F=F+DJ*X(J)**2
1067:             ENDIF
1068:       ENDDO
1069:       RETURN
1070:       ENDIF
1071: C     ------------------------------------------------------------------
1072:       IF(KF.EQ.9) THEN
```

```
1073: C     MODIFIED RCOS FUNCTION MIN=-0.179891 AT (-3.196989, 12.52626)APPRX
1074:       F=0.D00
1075:       IF(X(1).LT.-5.D00 .OR. X(1).GT.10.D00) THEN
1076:       CALL RANDOM(RAND)
1077:       X(1)=RAND*15.D00 -5.D00
1078:       ENDIF
1079:       IF(X(2).LT.0.D00 .OR. X(2).GT.15.D00) THEN
1080:       CALL RANDOM(RAND)
1081:       X(2)=RAND*15.D00
1082:       ENDIF
1083:       CA=1.D00
1084:       CB=5.1/(4*PI**2)
1085:       CC=5.D00/PI
1086:       CD=6.D00
1087:       CE=10.D00
1088:       CF=1.0/(8*PI)
1089:       F1=CA*(X(2)-CB*X(1)**2+CC*X(1)-CD)**2
1090:       F2=CE*(1.D00-CF)*DCOS(X(1))*DCOS(X(2))
1091:       F3=DLOG(X(1)**2+X(2)**2+1.D00)
1092:       F=-1.0/(F1+F2+F3+CE)
1093:       RETURN
1094:       ENDIF
1095: C     ----------------------------------------------------------------
1096:       IF(KF.EQ.10) THEN
1097: C     FREUDENSTEIN ROTH  FUNCTION : MIN  = 0 AT (5, 4)
1098:       F=0.D00
1099:       DO I=1,M
1100:       IF(DABS(X(I)).GT.10.D00) THEN
1101:       CALL RANDOM(RAND)
1102:       X(I)=(RAND-0.5D00)*20
1103:       ENDIF
1104:       ENDDO
1105:       F1=(-13.D00+X(1)+((5.D00-X(2))*X(2)-2)*X(2))**2
1106:       F2=(-29.D00+X(1)+((X(2)+1.D00)*X(2)-14.D00)*X(2))**2
1107:       F=F1+F2
1108:       RETURN
1109:       ENDIF
1110: C     ----------------------------------------------------------------
1111:       IF(KF.EQ.11) THEN
1112: C     ANNS XOR FUNCTION (PARSOPOULOS, KE, PLAGIANAKOS, VP, MAGOULAS, GD
1113: C     AND VRAHATIS, MN "STRETCHING TECHNIQUE FOR OBTAINING GLOBAL
1114: C     MINIMIZERS THROUGH PARTICLE SWARM OPTIMIZATION")
1115: C     MIN=0.9597588 FOR X=(1, -1, 1, -1, -1, 1, 1, -1, 0.421134) APPROX
1116: C     OBTAINED BY DIFFERENTIAL EVOLUTION PROGRAM
1117:       F=0.D00
1118:       DO I=1,M
1119:       IF(DABS(X(I)).GT.1.D00) THEN
1120:       CALL RANDOM(RAND)
1121:       X(I)=(RAND-0.5D00)*2
1122:       ENDIF
1123:       ENDDO
1124:       F11=X(7)/(1.D00+DEXP(-X(1)-X(2)-X(5)))
1125:       F12=X(8)/(1.D00+DEXP(-X(3)-X(4)-X(6)))
1126:       F1=(1.D00+DEXP(-F11-F12-X(9)))**(-2)
1127:       F21=X(7)/(1.D00+DEXP(-X(5)))
1128:       F22=X(8)/(1.D00+DEXP(-X(6)))
1129:       F2=(1.D00+DEXP(-F21-F22-X(9)))**(-2)
1130:       F31=X(7)/(1.D00+DEXP(-X(1)-X(5)))
1131:       F32=X(8)/(1.D00+DEXP(-X(3)-X(6)))
1132:       F3=(1.D00-(1.D00+DEXP(-F31-F32-X(9)))**(-1))**2
1133:       F41=X(7)/(1.D00+DEXP(-X(2)-X(5)))
1134:       F42=X(8)/(1.D00+DEXP(-X(4)-X(6)))
1135:       F4=(1.D00-(1.D00+DEXP(-F41-F42-X(9)))**(-1))**2
1136:       F=F1+F2+F3+F4
1137:       RETURN
1138:       ENDIF
1139: C     ----------------------------------------------------------------
```

```
      IF(KF.EQ.12) THEN
C     PERM FUNCTION #1 MIN = 0 AT (1, 2, 3, 4)
C     BETA => 0. CHANGE IF NEEDED. SMALLER BETA RAISES DIFFICULY
C     FOR BETA=0, EVERY PERMUTED SOLUTION IS A GLOBAL MINIMUM
      BETA=50.D00
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.M) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*2*M
      ENDIF
      ENDDO
      DO K=1,M
      SUM=0.D00
      DO I=1,M
      SUM=SUM+(I**K+BETA)*((X(I)/I)**K-1.D00)
      ENDDO
      F=F+SUM**2
      ENDDO
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.13) THEN
C     PERM FUNCTION #2 MIN = 0 AT (1/1, 1/2, 1/3, 1/4,..., 1/M)
C     BETA => 0. CHANGE IF NEEDED. SMALLER BETA RAISES DIFFICULY
C     FOR BETA=0, EVERY PERMUTED SOLUTION IS A GLOBAL MINIMUM
      BETA=10.D00
      DO I=1,M
      IF(DABS(X(I)).GT.1.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*2
      ENDIF
      SGN=X(I)/DABS(X(I))
      ENDDO
      F=0.D00
      DO K=1,M
      SUM=0.D00
      DO I=1,M
      SUM=SUM+(I+BETA)*(X(I)**K-(1.D00/I)**K)
      ENDDO
      F=F+SUM**2
      ENDDO
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.14) THEN
C     POWER SUM FUNCTION; MIN = 0 AT PERM(1,2,2,3) FOR B=(8,18,44,114)
C     0 =< X <=4
      F=0.D00
      DO I=1,M
C     ANY PERMUTATION OF (1,2,2,3) WILL GIVE MIN = ZERO
      IF(X(I).LT.0.D00 .OR. X(I).GT.4.D00) THEN
      CALL RANDOM(RAND)
      X(I)=RAND*4
      ENDIF
      ENDDO
      DO K=1,M
      SUM=0.D00
      DO I=1,M
      SUM=SUM+X(I)**K
      ENDDO
      IF(K.EQ.1) B=8.D00
      IF(K.EQ.2) B=18.D00
      IF(K.EQ.3) B=44.D00
      IF(K.EQ.4) B=114.D00
      F=F+(SUM-B)**2
      ENDDO
```

```
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.15) THEN
C     GOLDSTEIN PRICE  FUNCTION  : MIN VALUE = 3 AT (0, -1)
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*20
      ENDIF
      ENDDO
      F11=(X(1)+X(2)+1.D00)**2
      F12=(19.D00-14*X(1)+ 3*X(1)**2-14*X(2)+ 6*X(1)*X(2)+ 3*X(2)**2)
      F1=1.00+F11*F12
      F21=(2*X(1)-3*X(2))**2
      F22=(18.D00-32*X(1)+12*X(1)**2+48*X(2)-36*X(1)*X(2)+27*X(2)**2)
      F2=30.D00+F21*F22
      F= (F1*F2)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.16) THEN
C     BUKIN'S 6TH FUNCTION  MIN = 0 FOR (-10, 1)
C     -15. LE. X(1) .LE. -5 AND -3 .LE. X(2) .LE. 3
        IF(X(1).LT. -15.D00 .OR. X(1) .GT. -5.D00) THEN
        CALL RANDOM(RAND)
        X(1)=-(RAND*10+5.D00)
        ENDIF
        IF(DABS(X(2)).GT.3.D00) THEN
        CALL RANDOM(RAND)
        X(2)=(RAND-.5D00)*6
        ENDIF
      F=100.D0*DSQRT(DABS(X(2)-0.01D0*X(1)**2))+ 0.01D0*DABS(X(1)+10.D0)
       RETURN
       ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.17) THEN
C     NEW N#8 FUNCTION (MULTIPLE GLOBAL MINIMA)
C     MIN VALUE = -1 AT (AROUND .7 AROUND, 0.785 APPROX)
      F=0.D00
      DO I=1,M
      IF(X(I).LT.0.5D00 .OR. X(I).GT.1.D00) THEN
      CALL RANDOM(RAND)
      X(I)=RAND/2.D00
      ENDIF
      ENDDO
      F=-DEXP(-DABS(DLOG(.001D00+DABS((DSIN(X(1)+X(2))+DSIN(X(1)-X(2))+
     & (DCOS(X(1)+X(2))*DCOS(X(1)-X(2))+.001))**2)+
     & .01D00*(X(2)-X(1))**2)))
       RETURN
       ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.18) THEN
C     DEFLECTED CORRUGATED SPRING FUNCTION
C     MIN VALUE = -1 AT (5, 5, ..., 5) FOR ANY K AND ALPHA=5; M VARIABLE
      CALL DCS(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.19) THEN
C     FACTORIAL FUNCTION, MIN =0 AT X=(1,2,3,...,M)
      CALL FACTOR1(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.20) THEN
```

```
1274: C     DECANOMIAL FUNCTION, MIN =0 AT X=(2, -3)
1275:         DO I=1,M
1276:         IF(DABS(X(I)).GT.4.D00) THEN
1277:         CALL RANDOM(RAND)
1278:         X(I)= (RAND-0.5D00)*8
1279:         ENDIF
1280:         ENDDO
1281:       CALL DECANOM(M,F,X)
1282:       RETURN
1283:       ENDIF
1284: C     ------------------------------------------------------------------
1285:       IF(KF.EQ.21) THEN
1286: C     JUDGE'S FUNCTION  F(0.864, 1.23) = 16.0817; M=2
1287:       CALL JUDGE(M,X,F)
1288:       RETURN
1289:       ENDIF
1290: C     ------------------------------------------------------------------
1291:       IF(KF.EQ.22) THEN
1292: C     DODECAL FUNCTION
1293:       CALL DODECAL(M,F,X)
1294:       RETURN
1295:       ENDIF
1296: C     ------------------------------------------------------------------
1297:       IF(KF.EQ.23) THEN
1298: C     WHEN X(1)*X(2)=X(1)*X(2)  ? M=2
1299:       CALL SEQP(M,F,X)
1300:       RETURN
1301:       ENDIF
1302: C     ------------------------------------------------------------------
1303:       IF(KF.EQ.24) THEN
1304: C     WHEN ARITHMETIC MEAN = GEOMETRIC MEAN ? : M =>1
1305:       CALL AMGM(M,F,X)
1306:       RETURN
1307:       ENDIF
1308: C     ------------------------------------------------------------------
1309:       IF(KF.EQ.25) THEN
1310: C     M =>2
1311:       CALL FUNCT2(M,F,X)
1312:       RETURN
1313:       ENDIF
1314: C     ------------------------------------------------------------------
1315:       IF(KF.EQ.26) THEN
1316: C     M =>2
1317:       CALL FUNCT3(M,F,X)
1318:       RETURN
1319:       ENDIF
1320: C     ------------------------------------------------------------------
1321:       IF(KF.EQ.27) THEN
1322: C     M =>2
1323:       CALL FUNCT4(M,F,X)
1324:       RETURN
1325:       ENDIF
1326: C     ------------------------------------------------------------------
1327:       IF(KF.EQ.28) THEN
1328: C     M =>2
1329:       CALL FUNCT6(M,F,X)
1330:       RETURN
1331:       ENDIF
1332: C     ------------------------------------------------------------------
1333:       IF(KF.EQ.29) THEN
1334: C     M =>2
1335:       CALL FUNCT7(M,F,X)
1336:       RETURN
1337:       ENDIF
1338: C     ------------------------------------------------------------------
1339:       IF(KF.EQ.30) THEN
1340: C     M =>2
```

```
1341:       CALL FUNCT12(M,F,X)
1342:       RETURN
1343:       ENDIF
1344: C     ------------------------------------------------------------------
1345:       IF(KF.EQ.31) THEN
1346: C     M =>2
1347:       CALL FUNCT13(M,F,X)
1348:       RETURN
1349:       ENDIF
1350: C     ------------------------------------------------------------------
1351:       IF(KF.EQ.32) THEN
1352: C     M =2
1353:       CALL FUNCT14(M,F,X)
1354:       RETURN
1355:       ENDIF
1356: C     ------------------------------------------------------------------
1357:       IF(KF.EQ.33) THEN
1358: C     M =4
1359:       CALL FUNCT15(M,F,X)
1360:       RETURN
1361:       ENDIF
1362: C     ------------------------------------------------------------------
1363:       IF(KF.EQ.34) THEN
1364: C     WOOD FUNCTION : F MIN  : M=4
1365:       CALL WOOD(M,X,F)
1366:       RETURN
1367:       ENDIF
1368: C     ------------------------------------------------------------------
1369:       IF(KF.EQ.35) THEN
1370: C     FENTON & EASON FUNCTION :   : M=2
1371:       CALL FENTONEASON(M,X,F)
1372:       RETURN
1373:       ENDIF
1374: C     ------------------------------------------------------------------
1375:       IF(KF.EQ.36) THEN
1376: C     HOUGEN FUNCTION 5 VARIABLES : M =3
1377:       CALL HOUGEN(X,M,F)
1378:       RETURN
1379:       ENDIF
1380: C     ------------------------------------------------------------------
1381:       IF(KF.EQ.37) THEN
1382: C     GIUNTA FUNCTION 2 VARIABLES  :M =2
1383:       CALL GIUNTA(M,X,F)
1384:       RETURN
1385:       ENDIF
1386: C     ------------------------------------------------------------------
1387:       IF(KF.EQ.38) THEN
1388: C     EGGHOLDER FUNCTION M VARIABLES
1389:       CALL EGGHOLD(M,X,F)
1390:       RETURN
1391:       ENDIF
1392: C     ------------------------------------------------------------------
1393:       IF(KF.EQ.39) THEN
1394: C     TRID FUNCTION M VARIABLES
1395:       CALL TRID(M,X,F)
1396:       RETURN
1397:       ENDIF
1398: C     ------------------------------------------------------------------
1399:       IF(KF.EQ.40) THEN
1400: C     GRIEWANK FUNCTION M VARIABLES
1401:       CALL GRIEWANK(M,X,F)
1402:       RETURN
1403:       ENDIF
1404: C     ------------------------------------------------------------------
1405:       IF(KF.EQ.41) THEN
1406: C     WEIERSTRASS FUNCTION M VARIABLES
1407:       CALL WEIERSTRASS(M,X,F)
```

```
1408:       RETURN
1409:       ENDIF
1410: C     ------------------------------------------------------------------
1411:       IF(KF.EQ.42) THEN
1412: C     LEVY-3 FUNCTION 2 VARIABLES
1413:       CALL LEVY3(M,X,F)
1414:       RETURN
1415:       ENDIF
1416: C     ------------------------------------------------------------------
1417:       IF(KF.EQ.43) THEN
1418: C     LEVY-5 FUNCTION 2 VARIABLES
1419:       CALL LEVY5(M,X,F)
1420:       RETURN
1421:       ENDIF
1422: C     ------------------------------------------------------------------
1423:       IF(KF.EQ.44) THEN
1424: C     LEVY-8 FUNCTION 3 VARIABLES
1425:       CALL LEVY8(M,X,F)
1426:       RETURN
1427:       ENDIF
1428: C     ------------------------------------------------------------------
1429:       IF(KF.EQ.45) THEN
1430: C     RASTRIGIN FUNCTION M VARIABLES
1431:       CALL RASTRIGIN(M,X,F)
1432:       RETURN
1433:       ENDIF
1434: C     ------------------------------------------------------------------
1435:       IF(KF.EQ.46) THEN
1436: C     ACKLEY FUNCTION M VARIABLES
1437:       CALL ACKLEY(M,X,F)
1438:       RETURN
1439:       ENDIF
1440: C     ------------------------------------------------------------------
1441:       IF(KF.EQ.47) THEN
1442: C     MICHALEWICZ FUNCTION M VARIABLES
1443:       CALL MICHALEWICZ(M,X,F)
1444:       RETURN
1445:       ENDIF
1446: C     ------------------------------------------------------------------
1447:       IF(KF.EQ.48) THEN
1448: C     SCHWEFEL FUNCTION M VARIABLES
1449:       CALL SCHWEFEL(M,X,F)
1450:       RETURN
1451:       ENDIF
1452: C     ------------------------------------------------------------------
1453:       IF(KF.EQ.49) THEN
1454: C     SHUBERT FUNCTION 2 VARIABLES
1455:       CALL SHUBERT(M,X,F)
1456:       RETURN
1457:       ENDIF
1458: C     ------------------------------------------------------------------
1459:       IF(KF.EQ.50) THEN
1460: C     DIXON AND PRICE FUNCTION M VARIABLES
1461:       CALL DIXPRICE(M,X,F)
1462:       RETURN
1463:       ENDIF
1464: C     ------------------------------------------------------------------
1465:       IF(KF.EQ.51) THEN
1466: C     SHEKEL FUNCTION 4 VARIABLES
1467:       CALL SHEKEL(M,X,F)
1468:       RETURN
1469:       ENDIF
1470: C     ------------------------------------------------------------------
1471:       IF(KF.EQ.52) THEN
1472: C     PAVIANI FUNCTION 10 VARIABLES
1473:       CALL PAVIANI(M,X,F)
1474:       RETURN
```

```
1475:       ENDIF
1476: C     ------------------------------------------------------------------
1477:       IF(KF.EQ.53) THEN
1478: C     BRANIN FUNCTION#1 2 VARIABLES
1479:       CALL BRANIN1(M,X,F)
1480:       RETURN
1481:       ENDIF
1482: C     ------------------------------------------------------------------
1483:       IF(KF.EQ.54) THEN
1484: C     BRANIN FUNCTION#2 2 VARIABLES
1485:       CALL BRANIN2(M,X,F)
1486:       RETURN
1487:       ENDIF
1488: C     ------------------------------------------------------------------
1489:       IF(KF.EQ.55) THEN
1490: C     BOHACHEVSKY FUNCTION#1 2 VARIABLES
1491:       CALL BOHACHEVSKY1(M,X,F)
1492:       RETURN
1493:       ENDIF
1494: C     ------------------------------------------------------------------
1495:       IF(KF.EQ.56) THEN
1496: C     BOHACHEVSKY FUNCTION#2 2 VARIABLES
1497:       CALL BOHACHEVSKY2(M,X,F)
1498:       RETURN
1499:       ENDIF
1500: C     ------------------------------------------------------------------
1501:       IF(KF.EQ.57) THEN
1502: C     BOHACHEVSKY FUNCTION#3 2 VARIABLES
1503:       CALL BOHACHEVSKY3(M,X,F)
1504:       RETURN
1505:       ENDIF
1506: C     ------------------------------------------------------------------
1507:       IF(KF.EQ.58) THEN
1508: C     EASOM FUNCTION#3 2 VARIABLES
1509:       CALL EASOM(M,X,F)
1510:       RETURN
1511:       ENDIF
1512: C     ------------------------------------------------------------------
1513:       IF(KF.EQ.59) THEN
1514: C     ROSENBROCK FUNCTION  M VARIABLES
1515:       CALL ROSENBROCK(M,X,F)
1516:       RETURN
1517:       ENDIF
1518: C     ------------------------------------------------------------------
1519:       IF(KF.EQ.60) THEN
1520: C     CROSS-LEGGED TABLE FUNCTION : 2 VARIABLES
1521:       CALL CROSSLEG(M,X,F)
1522:       RETURN
1523:       ENDIF
1524: C     ------------------------------------------------------------------
1525:       IF(KF.EQ.61) THEN
1526: C     CROSS FUNCTION : 2 VARIABLES
1527:       CALL CROSS(M,X,F)
1528:       RETURN
1529:       ENDIF
1530: C     ------------------------------------------------------------------
1531:       IF(KF.EQ.62) THEN
1532: C     CROSS-IN-TRAY FUNCTION : 2 VARIABLES
1533:       CALL CROSSINTRAY(M,X,F)
1534:       RETURN
1535:       ENDIF
1536: C     ------------------------------------------------------------------
1537:       IF(KF.EQ.63) THEN
1538: C     CROWNED-CROSS FUNCTION : 2 VARIABLES
1539:       CALL CROWNEDCROSS(M,X,F)
1540:       RETURN
1541:       ENDIF
```

```
C     ------------------------------------------------------------------
      IF(KF.EQ.64) THEN
C     TT-HOLDER FUNCTION : 2 VARIABLES, MIN F([+/-]1.5706, 0)= -10.8723
      CALL TTHOLDER(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.65) THEN
C     HOLDER-TABLE FUNCTION : 2 VARIABLES
C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -26.92034 APPROX
      CALL HOLDERTABLE(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.66) THEN
C     CARROM-TABLE FUNCTION : 2 VARIABLES
C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -24.15682 APPROX
      CALL CARROMTABLE(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.67) THEN
C     PEN-HOLDER FUNCTION : 2 VARIABLES
C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -0.963535 APPROX
      CALL PENHOLDER(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.68) THEN
C     BIRD FUNCTION : 2 VARIABLES
C     MIN F (4.70104, 3.15294) APPROX = -106.764537 APPROX  OR
C     MIN F (-1.58214, -3.13024) APPROX = -106.764537 APPROX
      CALL BIRD(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.69) THEN
C     CHICHINADZE FUNCTION :  -30 <=X(I)<= 30;  M=2
C     MIN F (5.901329, 0.5) = -43.3158621
      CALL CHICHINADZE(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.70) THEN
C     MCCORMICK FUNCTION : -1.5<= X(1)<=4; -3<=X(2)<=4 ; M=2
C     MIN F (-0.54719755, -1.54719755) = -1.913223 APPROX
      CALL MCCORMICK(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.71) THEN
C     GLANKWAHMDEE FUNCTION:
      CALL GLANKWAHMDEE(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.72) THEN
C     FLETCHER-POWELL FUNCTION
      CALL FLETCHER(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.73) THEN
C     POWELL FUNCTION
      CALL POWELL(M,X,F)
      RETURN
      ENDIF
```

```
1609: C     ------------------------------------------------------------------
1610:       IF(KF.EQ.74) THEN
1611: C     HARTMANN FUNCTION
1612:       CALL HARTMANN(M,X,F)
1613:       RETURN
1614:       ENDIF
1615: C     ------------------------------------------------------------------
1616:       IF(KF.EQ.75) THEN
1617: C     COVILLE FUNCTION
1618:       CALL COLVILLE(M,X,F)
1619:       RETURN
1620:       ENDIF
1621: C     ------------------------------------------------------------------
1622:       IF(KF.EQ.76) THEN
1623: C     HIMMELBLAU FUNCTION
1624:       CALL HIMMELBLAU(M,X,F)
1625:       RETURN
1626:       ENDIF
1627: C     ------------------------------------------------------------------
1628:       IF(KF.EQ.77) THEN
1629: C     BEALE FUNCTION
1630:       CALL BEALE(M,X,F)
1631:       RETURN
1632:       ENDIF
1633: C     ------------------------------------------------------------------
1634:       IF(KF.EQ.78) THEN
1635: C     BOOTH FUNCTION
1636:       CALL BOOTH(M,X,F)
1637:       RETURN
1638:       ENDIF
1639: C     ------------------------------------------------------------------
1640:       IF(KF.EQ.79) THEN
1641: C     HUMP FUNCTION
1642:       CALL HUMP(M,X,F)
1643:       RETURN
1644:       ENDIF
1645: C     ------------------------------------------------------------------
1646:       IF(KF.EQ.80) THEN
1647: C     MATYAS FUNCTION
1648:       CALL MATYAS(M,X,F)
1649:       RETURN
1650:       ENDIF
1651: C     ------------------------------------------------------------------
1652:       IF(KF.EQ.81) THEN
1653: C     MISHRA_1 FUNCTION
1654:       CALL MISHRA_1(M,X,F)
1655:       RETURN
1656:       ENDIF
1657: C     ------------------------------------------------------------------
1658:       IF(KF.EQ.82) THEN
1659: C     MISHRA_2 FUNCTION
1660:       CALL MISHRA_2(M,X,F)
1661:       RETURN
1662:       ENDIF
1663: C     ------------------------------------------------------------------
1664:       IF(KF.EQ.83) THEN
1665: C     ZAKHAROV FUNCTION
1666:       CALL ZAKHAROV(M,X,F)
1667:       RETURN
1668:       ENDIF
1669: C     ------------------------------------------------------------------
1670:       IF(KF.EQ.84) THEN
1671: C     MULTOMOD FUNCTION
1672:       CALL MULTIMOD(M,X,F)
1673:       RETURN
1674:       ENDIF
1675: C     ------------------------------------------------------------------
```

```
      IF(KF.EQ.85) THEN
C     NONLINEAR  FUNCTION
      CALL NONLIN(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.86) THEN
C     QUADRATIC  FUNCTION
      CALL QUADRATIC(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.87) THEN
C     TRIGON FUNCTION
      CALL TRIGON(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.88) THEN
C     COMPOUND FUNCTION
      CALL COMPOUND(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
C      IF(KF.EQ.99) THEN
C     *****  FUNCTION
C      CALL SUBROUTINE(M,X,F)
C      RETURN
C      ENDIF
C     ==================================================================
      WRITE(*,*)'FUNCTION NOT DEFINED. PROGRAM ABORTED'
      STOP
      END
C     >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
      SUBROUTINE DCS(M,F,X)
C     FOR DEFLECTED CORRUGATED SPRING FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*),C(100)
C     OPTIMAL VALUES OF (ALL) X ARE ALPHA , AND K IS ONLY FOR SCALING
C     OPTIMAL VALUE OF F IS -1. DIFFICULT TO OPTIMIZE FOR LARGER M.
      DATA K,ALPHA/5,5.D00/ ! K AND ALPHA COULD TAKE ON ANY OTHER VALUES
      R2=0.D00
      DO I=1,M
      C(I)=ALPHA
      R2=R2+(X(I)-C(I))**2
      ENDDO
      R=DSQRT(R2)
      F=-DCOS(K*R)+0.1D00*R2
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE QUINTIC(M,F,X)
C     QUINTIC FUNCTION: GLOBAL MINIMA,EXTREMELY DIFFICULT TO OPTIMIZE
C     MIN VALUE = 0 AT PERM( -1, -0.402627941, 2)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      DO I=1,M
      F=F+DABS(X(I)**5-3*X(I)**4+4*X(I)**3+2*X(I)**2-10*X(I)-4.D00)
      ENDDO
      F=1000*F
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE FACTOR1(M,F,X)
C     FACTORIAL FUNCTION; MIN (1, 2, 3, ...., M) = 0
C     FACT = FACTORIAL(M) = 1 X 2 X 3 X 4 X .... X M
```

```
1743: C     FIND X(I), I=1,2,...,M SUCH THAT THEIR PRODUCT IS EQUAL TO FACT.
1744: C     LARGER THE VALUE OF M (=>8) OR SO, HARDER IS THE PROBLEM
1745:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1746:       DIMENSION X(*)
1747:       F=0.D00
1748:       FACT=1.D00
1749:       P=1.D00
1750:       DO I=1,M
1751:       FACT=FACT*I
1752:       P=P*X(I)
1753:       F=F+DABS(P-FACT)**2
1754:       ENDDO
1755:       RETURN
1756:       END
1757: C     ----------------------------------------------------------------------
1758:       SUBROUTINE DECANOM(M,F,X)
1759: C     DECANOMIAL FUNCTION; MIN (2, -3) = 0
1760:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1761:       DIMENSION X(*)
1762:       F1= DABS(X(1)**10-20*X(1)**9+180*X(1)**8-960*X(1)**7+
1763:      & 3360*X(1)**6-8064*X(1)**5+13340*X(1)**4-15360*X(1)**3+
1764:      & 11520*X(1)**2-5120*X(1)+2624.D00)
1765:       F2= DABS(X(2)**4+12*X(2)**3+54*X(2)**2+108*X(2)+81.D00)
1766:       F=0.001D00*(F1+F2)**2
1767:       RETURN
1768:       END
1769: C     ----------------------------------------------------------------------
1770:       SUBROUTINE JUDGE(M,X,F)
1771:       PARAMETER (N=20)
1772: C     THIS SUBROUTINE IS FROM THE EXAMPLE IN JUDGE ET AL., THE THEORY
1773: C     AND PRACTICE OF ECONOMETRICS, 2ND ED., PP. 956-7. THERE ARE TWO
1774: C     OPTIMA: F(0.86479,1.2357)=16.0817307 (WHICH IS THE GLOBAL MINUMUM)
1775: C     AND F(2.35,-0.319)=20.9805 (WHICH IS LOCAL). ADAPTED FROM BILL
1776: C     GOFFE'S SIMMAN (SIMULATED ANNEALING) PROGRAM
1777:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1778:       DIMENSION Y(N), X2(N), X3(N), X(*)
1779:        DATA (Y(I),I=1,N)/4.284,4.149,3.877,0.533,2.211,2.389,2.145,
1780:      & 3.231,1.998,1.379,2.106,1.428,1.011,2.179,2.858,1.388,1.651,
1781:      & 1.593,1.046,2.152/
1782:        DATA (X2(I),I=1,N)/.286,.973,.384,.276,.973,.543,.957,.948,.543,
1783:      & .797,.936,.889,.006,.828,.399,.617,.939,.784,.072,.889/
1784:       DATA (X3(I),I=1,N)/.645,.585,.310,.058,.455,.779,.259,.202,.028,
1785:      & .099,.142,.296,.175,.180,.842,.039,.103,.620,.158,.704/
1786:
1787:       F=0.D00
1788:       DO I=1,N
1789:       F=F+(X(1) + X(2)*X2(I) + (X(2)**2)*X3(I) - Y(I))**2
1790:       ENDDO
1791:       RETURN
1792:       END
1793: C     ----------------------------------------------------------------------
1794:       SUBROUTINE DODECAL(M,F,X)
1795:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1796:       DIMENSION X(*)
1797: C     DODECAL POLYNOMIAL MIN F(1,2,3)=0
1798:       DO I=1,M
1799:       IF(DABS(X(I)).GT.5.D0) THEN
1800:       CALL RANDOM(RAND)
1801:       X(I)=(RAND-0.5D00)*10
1802:       ENDIF
1803:       ENDDO
1804:       F=0.D00
1805:       F1=2*X(1)**3+5*X(1)*X(2)+4*X(3)-2*X(1)**2*X(3)-18.D00
1806:       F2=X(1)+X(2)**3+X(1)*X(2)**2+X(1)*X(3)**2-22.D00
1807:       F3=8*X(1)**2+2*X(2)*X(3)+2*X(2)**2+3*X(2)**3-52.D00
1808:       F=(F1*F3*F2**2+F1*F2*F3**2+F2**2+(X(1)+X(2)-X(3))**2)**2
1809:       RETURN
```

```
      END
C     ------------------------------------------------------------------
      SUBROUTINE SEQP(M,F,X)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     FOR WHAT VALUES X(1)+X(2)=X(1)*X(2) ? ANSWER: FOR (0,0) AND (2,2)
C     WHILE X(1), X(2) ARE INTEGERS.
      X(1)=INT(X(1)) ! X(1) CONVERTED TO INTEGER
      X(2)=INT(X(2)) ! X(2) CONVERTED TO INTEGER

      F1=X(1)+X(2)
      F2=X(1)*X(2)
      F=(F1-F2)**2     ! TURN ALIVE THIS  XOR
C      F=DABS(F1-F2)    ! TURN ALIVE THIS  - BUT NOT BOTH --------------
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE AMGM(M,F,X)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     FOR WHAT VALUES ARITHMETIC MEAN = GEOMETRIC MEAN ? THE ANSWER IS:
C     IF X(1)=X(2)=....=X(M) AND ALL X ARE NON-NEGATIVE
C     TAKE ONLY THE ABSOLUTE VALUES OF X
      SUM=0.D00
      DO I=1,M
      X(I)=DABS(X(I))
      ENDDO
C     SET SUM = SOME POSITIVE NUMBER. THIS MAKES THE FUNCTION UNIMODAL
      SUM= 100.D00 ! TURNED  ALIVE FOR UNIQUE MINIMUM AND SET SUM TO
C     SOME POSITIVE NUMBER. HERE IT IS 100; IT COULD BE ANYTHING ELSE.
      F1=0.D00
      F2=1.D00
      DO I=1,M
      F1=F1+X(I)
      F2=F2*X(I)
      ENDDO
      XSUM=F1
      F1=F1/M ! SUM DIVIDED BY M = ARITHMETIC MEAN
      F2=F2**(1.D00/M) ! MTH ROOT OF THE PRODUCT = GEOMETRIC MEAN
      F=(F1-F2)**2
      IF(SUM.GT.0.D00) F=F+(SUM-XSUM)**2
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE FUNCT2(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     IN FOGEL, L.J., ANGELIN, P.J. AND BACK, T. (ED) PROCEEDINGS OF THE
C     FIFTH ANNUAL CONFERENCE ON EVOLUTIONARY PROGRAMMING, PP. 451-460,
C     MIT PRESS, CAMBRIDGE, MASS.
C     MIN F (0, 0, ..., 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      F1=1.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*20
      ENDIF
      ENDDO
      DO I=1,M
      F=F+DABS(X(I))
      F1=F1*DABS(X(I))
      ENDDO
      F=F+F1
      RETURN
      END
```

```
1877: C     ------------------------------------------------------------------
1878:       SUBROUTINE FUNCT3(M,F,X)
1879: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1880: C     MIN F (0, 0, ... , 0) = 0
1881:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1882:       DIMENSION X(*)
1883:       F=0.D00
1884:       F1=0.D00
1885:       DO I=1,M
1886:       IF(DABS(X(I)).GT.100.D00) THEN
1887:       CALL RANDOM(RAND)
1888:       X(I)=(RAND-.5D00)*200
1889:       ENDIF
1890:       ENDDO
1891:       DO I=1,M
1892:       F1=0.D00
1893:       DO J=1,I
1894:       F1=F1+X(J)**2
1895:       ENDDO
1896:       F=F+F1
1897:       ENDDO
1898:       RETURN
1899:       END
1900: C     ------------------------------------------------------------------
1901:       SUBROUTINE FUNCT4(M,F,X)
1902: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1903: C     MIN F (0, 0, ..., 0) = 0
1904:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1905:       DIMENSION X(*)
1906:       F=0.D00
1907:       DO I=1,M
1908:       IF(X(I).LT.0.D00 .OR. X(I).GE.M) THEN
1909:       CALL RANDOM(RAND)
1910:       X(I)=RAND*2*M
1911:       ENDIF
1912:       ENDDO
1913: C     FIND MAX(X(I))=MAX(ABS(X(I))) NOTE: HERE X(I) CAN BE ONLY POSITIVE
1914:       XMAX=X(1)
1915:       DO I=1,M
1916:       IF(XMAX.LT.X(I)) XMAX=X(I)
1917:       ENDDO
1918:       F=XMAX
1919:       RETURN
1920:       END
1921: C     ------------------------------------------------------------------
1922:       SUBROUTINE FUNCT6(M,F,X)
1923: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1924: C     MIN F (-.5, -.5, ..., -.5) = 0
1925:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1926:       DIMENSION X(*)
1927:       F=0.D00
1928:       DO I=1,M
1929:       IF(DABS(X(I)).GT.100.D00) THEN
1930:       CALL RANDOM(RAND)
1931:       X(I)=(RAND-.5D00)*200
1932:       ENDIF
1933:       ENDDO
1934:       DO I=1,M
1935:        F=F+(X(I)+0.5D00)**2
1936:       ENDDO
1937:       RETURN
1938:       END
1939: C     ------------------------------------------------------------------
1940:       SUBROUTINE FUNCT7(M,F,X)
1941: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1942: C     MIN F(0, 0, ..., 0) = 0
1943:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
```

```
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
      F=0.D00
        DO I=1,M
        IF(DABS(X(I)).GT.1.28D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*2.56D00
        ENDIF
        ENDDO
      DO I=1,M
      CALL RANDOM(RAND)
      F=F+(I*X(I)**4)
      ENDDO
       CALL RANDOM(RAND)
       F=F+RAND
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE FUNCT12(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(100),Y(100)
      DATA A,B,C /10.D00,100.D00,4.D00/
      PI=4.D00*DATAN(1.D00)
      F=0.D00
C     MIN F (-1, -1, -1, ..., -1) = 0
C       X(I)=-1.D00  ! TO CHECK, TURN IT ALIVE
       DO I=1,M
        IF(DABS(X(I)).GT.50.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*100.D00
        ENDIF
        ENDDO
      F1=0.D00
      DO I=1,M
      XX=DABS(X(I))
      U=0.D00
      IF(XX.GT.A) U=B*(XX-A)**C
      F1=F1+U
      ENDDO
      F2=0.D00
      DO I=1,M-1
      Y(I)=1.D00+.25D00*(X(I)+1.D00)
      F2=F2+ (Y(I)-1.D00)**2 * (1.D00+10.D00*(DSIN(PI*X(I+1))**2))
      ENDDO
      Y(M)=1.D00+.25D00*(X(M)+1.D00)
      F3=(Y(M)-1.D00)**2
      Y(1)=1.D00+.25D00*(X(1)+1.D00)
      F4=10.D00*(DSIN(PI*Y(1)))**2
      F=(PI/M)*(F4+F2+F3)+F1
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE FUNCT13(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(100)
      DATA A,B,C /5.D00,100.D00,4.D00/
      PI=4*DATAN(1.D00)
      F=0.D00
C     MIN F (1, 1, 1, ..., 4.7544 APPROX) = -1.15044 APPROX
C        X(I)=1.D00  ! TO CHECK, TURN IT ALIVE
C        X(M)=-4.7544  ! TO CHECK, TURN IT ALIVE
      DO I=1,M
      IF(DABS(X(I)).GT.50.D00) THEN
      CALL RANDOM(RAND)
```

```
2011:       X(I)=(RAND-.5D00)*100.D00
2012:       ENDIF
2013:       ENDDO
2014:       F1=0.D00
2015:       DO I=1,M
2016:       XX=DABS(X(I))
2017:       U=0.D00
2018:       IF(XX.GT.A) U=B*(XX-A)**C
2019:       F1=F1+U
2020:       ENDDO
2021:       F2=0.D00
2022:       DO I=1,M-1
2023:       F2=F2+ (X(I)-1.D00)**2 * (1.D00+(DSIN(3*PI*X(I+1))**2))
2024:       ENDDO
2025:       F3=(X(M)-1.D00)*  (1.D00+(DSIN(2*PI*X(M)))**2)
2026:       F4=(DSIN(3*PI*X(1)))**2
2027:       F=0.1*(F4+F2+F3)+F1
2028:       RETURN
2029:       END
2030: C     ------------------------------------------------------------------
2031:       SUBROUTINE FUNCT14(M,F,X)
2032: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
2033: C     MIN F (-31.98, 31.98) = 0.998
2034:       PARAMETER (N=25,NN=2)
2035:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2036:       DIMENSION X(2), A(NN,N)
2037:       DATA (A(1,J),J=1,N) /-32.D00,-16.D00,0.D00,16.D00,32.D00,-32.D00,
2038:      & -16.D00,0.D00,16.D00,32.D00,-32.D00,-16.D00,0.D00,16.D00,32.D00,
2039:      & -32.D0,-16.D0,0.D0,16.D0,32.D0,-32.D0,-16.D0,0.D0,16.D0,32.D0/
2040:       DATA (A(2,J),J=1,N) /-32.D00,-32.D00,-32.D00,-32.D00,-32.D00,
2041:      & -16.D00,-16.D00,-16.D00,-16.D00,-16.D00,0.D00,0.D00,0.D00,0.D00,
2042:      & 0.D00,16.D00,16.D00,16.D00,16.D00,16.D00,32.D00,32.D00,
2043:      & 32.D00,32.D00,32.D00/
2044:
2045:       F=0.D00
2046:       DO I=1,M
2047:       IF(DABS(X(I)).GT.100.D00) THEN
2048:       CALL RANDOM(RAND)
2049:       X(I)=(RAND-.5D00)*200.D00
2050:       ENDIF
2051:       ENDDO
2052:       F1=0.D00
2053:       DO J=1,N
2054:       F2=0.D00
2055:       DO I=1,2
2056:       F2=F2+(X(I)-A(I,J))**6
2057:       ENDDO
2058:       F2=1.D00/(J+F2)
2059:       F1=F1+F2
2060:       ENDDO
2061:       F=1.D00/(0.002D00+F1)
2062:       RETURN
2063:       END
2064: C     ------------------------------------------------------------------
2065:       SUBROUTINE FUNCT15(M,F,X)
2066: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
2067: C     MIN F(.19, .19, .12, .14) = 0.3075
2068:       PARAMETER (N=11)
2069:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2070:       DIMENSION X(*), A(N),B(N)
2071:       DATA (A(I),I=1,N) /.1957D00,.1947D00,.1735D00,.16D00,.0844D00,
2072:      & .0627D00,.0456D00,.0342D00,.0323D00,.0235D00,.0246D00/
2073:       DATA (B(I),I=1,N) /0.25D00,0.5D00,1.D00,2.D00,4.D00,6.D00,8.D00,
2074:      & 10.D00,12.D00,14.D00,16.D00/
2075:       DO I=1,N
2076:       B(I)=1.D00/B(I)
2077:       ENDDO
```

```
2078:       F=0.D00
2079:       DO I=1,M
2080:       IF(DABS(X(I)).GT.5.D00) THEN
2081:       CALL RANDOM(RAND)
2082:       X(I)=(RAND-.5D00)*10.D00
2083:       ENDIF
2084:       ENDDO
2085:       DO I=1,N
2086:       F1=X(1)*(B(I)**2+B(I)*X(2))
2087:       F2=B(I)**2+B(I)*X(3)+X(4)
2088:       F=F+(A(I)-F1/F2)**2
2089:       ENDDO
2090:       F=F*1000
2091:       RETURN
2092:       END
2093: C     ----------------------------------------------------------------
2094:       SUBROUTINE LINPROG1(M,F,X)
2095: C     LINEAR PROGRAMMING : MINIMIZATION PROBLEM
2096: C     IN THIS PROBLEM : M = NO. OF DECISION VARIABLES = 2
2097: C     MIN F (2.390, 2.033) = -19.7253 APPROX
2098: C     MIN F = OVER J=1, M : DO SUM(A(1,J)*X(J))  SUBJECT TO CONSTRAINTS
2099: C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=2
2100: C     . . .         . . .          . . .        . . .
2101: C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=N
2102: C     ALL X(I) => 0
2103:       PARAMETER (N=3) ! N IS THE NO. OF CONSTRAINTS + 1
2104:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2105:       DIMENSION X(*),A(20,10),C(20),FF(20)
2106:       DATA (A(1,J),J=1,2),C(1)/4.D0,5.D0,0.0D0/!COEFF OF OBJ FUNCTION
2107:       DATA (A(2,J),J=1,2),C(2)/10.D0,3.D0,30D0/!COEFF OF 1ST CONSTRAINT
2108:       DATA (A(3,J),J=1,2),C(3)/6.D0,20.D0,55.D0/!COEFF OF 2ND CONSTRAINT
2109: C     ----------------------------------------------------------------
2110: C     USING ONLY NON-NEGATIVE VALUES OF X(I)
2111:       DO I=1,M
2112:       X(I)=DABS(X(I))
2113:       ENDDO
2114: C     EVALUATION OF OBJ FUNCTION AND CONSTRAINTS
2115:       DO I=1,N
2116:       FF(I)=0.D00
2117:       DO J=1,M
2118:       FF(I)=FF(I)+A(I,J)*X(J)
2119:       ENDDO
2120:       ENDDO
2121:       F=-FF(1) ! CHANGE OF SIGN FOR MINIMIZATION
2122: C     CHECK FOR SATISFYING OR VIOLATING THE CONSTRAINTS
2123:       DO I=2,N
2124:       FF(I)=FF(I)-C(I) ! SLACK
2125: C     PENALTY FOR CROSSING LIMITS
2126:       IF(FF(I).GT.0) F=F+(10+FF(I))**2
2127:       ENDDO
2128:       RETURN
2129:       END
2130: C     ----------------------------------------------------------------
2131:       SUBROUTINE LINPROG2(M,F,X)
2132: C     LINEAR PROGRAMMING : MINIMIZATION PROBLEM
2133: C     IN THIS PROBLEM : M = NO. OF DECISION VARIABLES = 3
2134: C     MIN F (250, 625, 0) = -3250
2135: C     MIN F = OVER J=1, M : DO SUM(A(1,J)*X(J))   SUBJECT TO CONSTRAINTS
2136: C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=2
2137: C     . . .         . . .          . . .        . . .
2138: C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=N
2139: C     ALL X(I) => 0
2140:       PARAMETER (N=4) ! N IS THE NO. OF CONSTRAINTS + 1
2141:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2142:       DIMENSION X(*),A(20,10),C(20),FF(20)
2143:       DATA (A(1,J),J=1,3),C(1)/30.D0,40.D0,20.D0,0.0D0/! COEFF OF OBJ FUNCTION
2144:       DATA (A(2,J),J=1,3),C(2)/10.D0,12.D0,7.D0,10000.0D0/!COEFF OF 1ST CONSTRAINT
```

```
2145:       DATA (A(3,J),J=1,3),C(3)/7.D0,10.D0,8.D0,8000.D0/! COEFF OF 2ND CONSTRAINT
2146:       DATA (A(4,J),J=1,3),C(4)/1.D0,1.D0,1.D0,1000.D0/! COEFF OF 3RD CONSTRAINT
2147: C     ------------------------------------------------------------------
2148: C     USING ONLY NON-NEGATIVE VALUES OF X(I)
2149:       DO I=1,M
2150:       X(I)=DABS(X(I))
2151:       ENDDO
2152: C     EVALUATION OF OBJ FUNCTION AND CONSTRAINTS
2153:       DO I=1,N
2154:       FF(I)=0.D00
2155:       DO J=1,M
2156:       FF(I)=FF(I)+A(I,J)*X(J)
2157:       ENDDO
2158:       ENDDO
2159:       F=-FF(1) ! CHANGE OF SIGN FOR MINIMIZATION
2160: C     CHECK FOR SATISFYING OR VIOLATING THE CONSTRAINTS
2161:       DO I=2,N
2162:       FF(I)=FF(I)-C(I) ! SLACK
2163: C     PENALTY FOR CROSSING LIMITS
2164:       IF(FF(I).GT.0.D00) F=F+(100.D00+FF(I))**2
2165:       ENDDO
2166:       RETURN
2167:       END
2168: C     ------------------------------------------------------------------
2169:       SUBROUTINE HOUGEN(A,M,F)
2170:       PARAMETER(N=13,K=3)
2171:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2172:       DIMENSION X(N,K),RATE(N),A(*)
2173: C     ------------------------------------------------------------------
2174: C     HOUGEN FUNCTION (HOUGEN-WATSON MODEL FOR REACTION KINATICS)
2175: C     NO. OF PARAMETERS (A) TO ESTIMATE = 5 = M
2176:
2177: C     BEST RESULTS ARE:
2178: C     A(1)=1.253031; A(2)=1.190943; A(3)=0.062798; A(4)=0.040063
2179: C     A(5)=0.112453  ARE BEST ESTIMATES OBTAINED BY ROSENBROCK &
2180: C     QUASI-NEWTON METHOD WITH SUM OF SQUARES OF DEVIATION =0.298900994
2181: C     AND R=0.99945.
2182:
2183: C     THE NEXT BEST RESULTS GIVEN BY HOOKE-JEEVES & QUASI-NEWTON
2184: C     A(1)=2.475221;A(2)=0.599177; A(3)=0.124172; A(4)=0.083517
2185: C     A(5)=0.217886; SUM OF SQUARES OF DEVIATION = 0.318593458
2186: C     R=0.99941
2187: C     MOST OF THE OTHER METHODS DO NOT PERFORM WELL
2188: C     ------------------------------------------------------------------
2189:       DATA X(1,1),X(1,2),X(1,3),RATE(1) /470,300,10,8.55/
2190:       DATA X(2,1),X(2,2),X(2,3),RATE(2) /285,80,10,3.79/
2191:       DATA X(3,1),X(3,2),X(3,3),RATE(3) /470,300,120,4.82/
2192:       DATA X(4,1),X(4,2),X(4,3),RATE(4) /470,80,120,0.02/
2193:       DATA X(5,1),X(5,2),X(5,3),RATE(5) /470,80,10,2.75/
2194:       DATA X(6,1),X(6,2),X(6,3),RATE(6) /100,190,10,14.39/
2195:       DATA X(7,1),X(7,2),X(7,3),RATE(7) /100,80,65,2.54/
2196:       DATA X(8,1),X(8,2),X(8,3),RATE(8) /470,190,65,4.35/
2197:       DATA X(9,1),X(9,2),X(9,3),RATE(9) /100,300,54,13/
2198:       DATA X(10,1),X(10,2),X(10,3),RATE(10) /100,300,120,8.5/
2199:       DATA X(11,1),X(11,2),X(11,3),RATE(11) /100,80,120,0.05/
2200:       DATA X(12,1),X(12,2),X(12,3),RATE(12) /285,300,10,11.32/
2201:       DATA X(13,1),X(13,2),X(13,3),RATE(13) /285,190,120,3.13/
2202: C     WRITE(*,1)((X(I,J),J=1,K),RATE(I),I=1,N)
2203: C    1 FORMAT(4F8.2)
2204:       DO J=1,M
2205:       IF(DABS(A(J)).GT.5.D00) THEN
2206:       CALL RANDOM(RAND)
2207:       A(J)=(RAND-0.5D00)*10.D00
2208:       ENDIF
2209:       ENDDO
2210:       F=0.D00
2211:       DO I=1,N
```

```
2212:       D=1.D00
2213:          DO J=1,K
2214:          D=D+A(J+1)*X(I,J)
2215:          ENDDO
2216:       FX=(A(1)*X(I,2)-X(I,3)/A(M))/D
2217: C     FX=(A(1)*X(I,2)-X(I,3)/A(5))/(1.D00+A(2)*X(I,1)+A(3)*X(I,2)+
2218: C     A(4)*X(I,3))
2219:       F=F+(RATE(I)-FX)**2
2220:       ENDDO
2221:       RETURN
2222:       END
2223: C     ------------------------------------------------------------------
2224:       SUBROUTINE GIUNTA(M,X,F)
2225:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2226:       DIMENSION X(*)
2227: C     GIUNTA FUNCTION
2228: C     X(I) = -1 TO 1;  M=2
2229:         DO I=1,M
2230:         IF(DABS(X(I)).GT.1.D00) THEN
2231:         CALL RANDOM(RAND)
2232:         X(I)=(RAND-0.5D00)*2.D00
2233:         ENDIF
2234:         ENDDO
2235:       C=16.D00/15.D00
2236:       F=DSIN(C*X(1)-1.D0)+DSIN(C*X(1)-1.D0)**2+DSIN(4*(C*X(1)-1.D0))/50+
2237:      &DSIN(C*X(2)-1.D0)+DSIN(C*X(2)-1.D0)**2+DSIN(4*(C*X(2)-1.D0))/50+.6
2238:       RETURN
2239:       END
2240: C     ------------------------------------------------------------------
2241:       SUBROUTINE EGGHOLD(M,X,F)
2242: C     EGG HOLDER FUNCTION
2243:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2244:       DIMENSION X(*)
2245:       DO I=1,M
2246:        IF(DABS(X(I)).GT.512.D00) THEN
2247:         CALL RANDOM(RAND)
2248:         X(I)=(RAND-0.5D00)*1024.D00
2249:        ENDIF
2250:        ENDDO
2251:        F=0.D00
2252:       DO I=1,M-1
2253:       F1=-(X(I+1)+47.D00)
2254:       F2=DSIN(  DSQRT(  DABS(  X(I+1)+X(I)/2+47.D00  )  )  )
2255:       F3=DSIN(  DSQRT(  DABS(  X(I)-(X(I+1)+47.D00) ) ))
2256:       F4=-X(I)
2257:       F=F+ F1*F2+F3*F4
2258:       ENDDO
2259:       RETURN
2260:       END
2261: C     ------------------------------------------------------------------
2262:       SUBROUTINE TRID(M,X,F)
2263: C     TRID FUNCTION
2264:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2265:       DIMENSION X(*)
2266:       F1=0.D00
2267:       F2=0.D00
2268:       DO I=1, M
2269:       F1=F1+(X(I)-1.D00)**2
2270:       ENDDO
2271:       DO I=2, M
2272:       F2=F2+X(I)*X(I-1)
2273:       ENDDO
2274:       F=F1-F2
2275:       RETURN
2276:       END
2277: C     ------------------------------------------------------------------
2278:       SUBROUTINE GRIEWANK(M,X,F)
```

```
2279:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2280:       DIMENSION X(*)
2281: C     GRIEWANK FUNCTION
2282:       F1=0.D00
2283:       F2=1.0D00
2284:       DO I=1,M
2285:       F1=F1+X(I)**2
2286:       FI=DFLOAT(I)
2287:       F2=F2*DCOS(X(I)/DSQRT(FI))
2288:       ENDDO
2289:       F=F1/4000.D00-F2+1.0D00
2290:       RETURN
2291:       END
2292: C     ------------------------------------------------------------------
2293:       SUBROUTINE WEIERSTRASS(M,X,F)
2294:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2295:       DIMENSION X(*)
2296:       PI=4*DATAN(1.D00)
2297: C     WEIERSTRASS FUNCTION ----------------------------------
2298:       DATA AX,BX,KMAX/0.5D00,3.D00,20/
2299:       F1=0.D00
2300:       F2=0.D00
2301:
2302:         DO I=1,M
2303:         IF(DABS(X(I)).GT.0.5D00) THEN
2304:         CALL RANDOM(RAND)
2305:         X(I)=RAND-0.5D00
2306:         ENDIF
2307:         ENDDO
2308:
2309:       DO I=1,M
2310:       S=0.D00
2311:           DO KK=1,KMAX+1
2312:           K=KK-1
2313:           S=S+(AX**K*DCOS(2*PI*BX**K*(X(I)+0.5D00)))
2314:           ENDDO
2315:       F1=F1+S
2316:       ENDDO
2317:
2318:       DO KK=1,KMAX+1
2319:       K=KK-1
2320:       F2=F2+(AX**K*DCOS(2*PI*BX**K*0.5D00))
2321:       ENDDO
2322:       F=F1-M*F2
2323:       RETURN
2324:       END
2325: C     ------------------------------------------------------------------
2326:       SUBROUTINE LEVY3(M,X,F)
2327:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2328:       DIMENSION X(*)
2329: C     LEVY # 3 (LEVY ET AL. 1981) ---------------------------------
2330:          DO I=1,M
2331:          IF(DABS(X(I)).GT.10.D00) THEN
2332:          CALL RANDOM(RAND)
2333:          X(I)=(RAND-0.5D00)*20
2334:          ENDIF
2335:          ENDDO
2336:           F1=0.0D+00
2337:           F2=0.0D+00
2338:           DO I=1,5
2339:           F1=F1+(I*DCOS((I-1)*X(1)+I))
2340:           F2=F2+(I*DCOS((I+1)*X(2)+I))
2341:           ENDDO
2342:          F=F1*F2
2343:          RETURN
2344:          END
2345: C     ------------------------------------------------------------------
```

```
      SUBROUTINE LEVY5(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
            F1=0.0D+00
          F2=0.0D+00
        DO I=1,5
       F1=F1+(I*DCOS((I-1)*X(1)+I))
          F2=F2+(I*DCOS((I+1)*X(2)+I))
          ENDDO
          F3=(X(1)+1.42513D+00)**2
        F4=(X(2)+0.80032D+00)**2
        F=(F1*F2) + (F3+F4)
        RETURN
        END
C     ------------------------------------------------------------------
      SUBROUTINE LEVY8(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*),Y(3)
      PI=4*DATAN(1.D00)
C     LEVY # 8 FUNCTION ------------------------------------------------
        DO I=1,3
        Y(I)=1.D+00+(X(I)-1.D+00)/4.D+00
        ENDDO
        F1=DSIN(PI*Y(1))**2
       F3=(Y(3)-1.D+00)**2
      F2=0.D+00
       DO I=1,2
        F2=F2+((Y(I)-1.D+00)**2)*(1.D+00+10.D+00*(DSIN(PI*Y(I+1)))**2)
        ENDDO
      F=F1+F2+F3
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE RASTRIGIN(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
C     RASTRIGIN'S FUNCTION
      F=0.D00
      DO I=1,M
      F=F+ X(I)**2 -10*DCOS(2*PI*X(I)) + 10.D00
      ENDDO
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE ACKLEY(M,X,F)
C     ACKLEY FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      F=20.D00+DEXP(1.D00)
      DO I=1,M
      IF(X(I).LT. -15.D00 .OR. X(I).GT.30.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*90 -15.D00
      ENDIF
      ENDDO
      F1=0.D00
      F2=0.D00
        DO I=1,M
         F1=F1+X(I)**2
         F2=F2+DCOS(2*PI*X(I))
        ENDDO
      F1=-20*DEXP(-0.2D00*DSQRT(F1/M))
      F2=-DEXP(F2/M)
      F=F+F1+F2
      RETURN
```

```
      END
C     ------------------------------------------------------------------
      SUBROUTINE MICHALEWICZ(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     MICHALEWICZ FUNCTION   [ 0 <= X(I) <= PI ] MP IS A PARAMETER
      MP=10  ! SET IT TO THE DESIRED VALUE
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF(X(I).LT.0.D00 .OR. X(I).GT.PI)THEN
      CALL RANDOM(RAND)
      X(I)=RAND*PI
      ENDIF
      ENDDO
      F=0.D00
      DO I=1,M
      F=F-DSIN(X(I))*(DSIN(I*X(I)**2/PI))**(2*MP)
      ENDDO
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE SCHWEFEL(M,X,F)
C     SCHWEFEL FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      DO I=1,M
      IF(DABS(X(I)).GT.500) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*1000
      ENDIF
      ENDDO
      F=0.D00
      DO I=1,M
      F=F+ X(I)*DSIN(DSQRT(DABS(X(I))))
      ENDDO
      F=418.9829D00*M - F
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE SHUBERT(M,X,F)
C     SHUBERT FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
         DO I=1,M
         IF(DABS(X(I)).GT.10.D00) THEN
         CALL RANDOM(RAND)
         X(I)=(RAND-0.5D00)*20
         ENDIF
         ENDDO
      F1=0.D00
      F2=0.D00
      DO I=1,5
      F1=F1+I*DCOS((I+1.D00)*X(1)+I)
      F2=F2+I*DCOS((I+1.D00)*X(2)+I)
      ENDDO
      F=F1*F2
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE DIXPRICE(M,X,F)
C     DIXON & PRICE FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
       DO I=1,M
         IF(DABS(X(I)).GT.10.D00) THEN
         CALL RANDOM(RAND)
         X(I)=(RAND-0.5D00)*20
```

```
2480:           ENDIF
2481:          ENDDO
2482:         F=0.D00
2483:         DO I=2, M
2484:         F=F + I*(2*X(I)**2-X(I-1))**2
2485:         ENDDO
2486:         F=F+(X(1)-1.D00)**2
2487:         RETURN
2488:         END
2489: C       ------------------------------------------------------------------
2490:         SUBROUTINE SHEKEL(M,X,F)
2491: C       SHEKEL FUNCTION FOR TEST OF GLOBAL OPTIMIZATION METHODS
2492: C       MIN F (4, 4, 4, 4) = 10.02 TO 10.54 DEPENDING ON PARAMETERS
2493:         PARAMETER(NROW=10,NCOL=4, NR=9)! NR MAY BE 2 TO 10
2494:         IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2495:         DIMENSION A(NROW,NCOL),C(NROW),X(*)
2496:         DATA ((A(I,J),J=1,NCOL),I=1,NROW)/4.,4.,4.,4.,1.,1.,1.,1.,8.,8.,
2497:        & 8.,8.,6.,6.,6.,6.,3.,7.,3.,7.,2.,9.,2.,9.,5.,5.,3.,3.,8.,1.,8.,
2498:        & 1.,6.,2.,6.,2.,7.,3.6D00,7.,3.6D00/
2499:         DATA (C(I),I=1,NROW)/0.1D00,0.2D00,0.2D00,0.4D00,0.4D00,0.6D00,
2500:        & 0.3D00,0.7D00,0.5D00,0.5D00/
2501:         F=0.D00
2502:         DO I=1,NR
2503:           S=0.D00
2504:           DO J=1,M
2505:           S=S+(X(J)-A(I,J))**2
2506:           ENDDO
2507:         F=F-1.D00/(S+C(I))
2508:         ENDDO
2509:         RETURN
2510:         END
2511: C       ------------------------------------------------------------------
2512:         SUBROUTINE PAVIANI(M,X,F)
2513: C       PAVIANI FUNCTION : MIN F(9.3502,...,9.3502)=45.77847 APPROX
2514: C       IN THE DOMAIN 2<= X(I) <= 10 FOR I=1,2,...,10.
2515:         IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2516:         DIMENSION X(*)
2517:         DO I=1,M
2518:         IF(X(I).LE.2.D00.OR.X(I).GE.10.D00) THEN
2519:         CALL RANDOM(RAND)
2520:         X(I)=RAND*8+2.D00
2521:         ENDIF
2522:         ENDDO
2523:         F1=0.D00
2524:         F2=1.D00
2525:         DO I=1,M
2526:         F1=F1+ DLOG(X(I)-2.D00)**2+DLOG(10.D00-X(I))**2
2527:         F2=F2*X(I)
2528:         ENDDO
2529:         F=F1-F2**0.2
2530:         RETURN
2531:         END
2532: C       ------------------------------------------------------------------
2533:         SUBROUTINE BRANIN1(M,X,F)
2534: C       BRANIN FUNCTION #1  MIN F (1, 0) = 0
2535:         IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2536:         DIMENSION X(*)
2537:         PI=4*DATAN(1.D00)
2538:         DO I=1,M
2539:         IF(DABS(X(I)).GT.10.D00) THEN
2540:         CALL RANDOM(RAND)
2541:         X(I)=(RAND-0.5D00)*20
2542:         ENDIF
2543:         ENDDO
2544:         F=(1.D00-2*X(2)+DSIN(4*PI*X(2))/2.D00-X(1))**2+(X(2)-
2545:        & DSIN(2*PI*X(1))/2.D00)**2
2546:         RETURN
```

```
      END
C     ------------------------------------------------------------------
      SUBROUTINE BRANIN2(M,X,F)
C     BRANIN FUNCTION #2 MIN F (3.1416, 2.25)= 0.397887 APPROX
C     (-3.1416, 12.250), (9.4248, 2.25) ARE OTHER SOLUTIONS
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      IF(X(1).LT.-5.D00 .OR. X(1).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(1)=RAND*15-5.D00
      ENDIF
      IF(X(2).LT.0.D00 .OR. X(2).GT.15.D00) THEN
      CALL RANDOM(RAND)
      X(2)=RAND*15
      ENDIF
      F=(X(2)-5.D00*X(1)**2/(4*PI**2)+5*X(1)/PI-6.D00)**2 +
     &   10*(1.D00-1.D00/(8*PI))*DCOS(X(1))+10.D00
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE BOHACHEVSKY1(M,X,F)
C     BOHACHEVSKY FUNCTION #1 : MIN F (0, 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1))-0.4D00*DCOS(4*PI*X(2))
     & +0.7D00
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE BOHACHEVSKY2(M,X,F)
C     BOHACHEVSKY FUNCTION #2  : MIN F (0, 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1))*DCOS(4*PI*X(2))+0.3D00
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE BOHACHEVSKY3(M,X,F)
C     BOHACHEVSKY FUNCTION #3  : MIN F (0, 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1)+4*PI*X(2))+0.3D00
      RETURN
      END
C     ------------------------------------------------------------------
```

```
      SUBROUTINE EASOM(M,X,F)
C     EASOM FUNCTION : 2-VARIABLES, MIN F (PI, PI) = -1.
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=-DCOS(X(1))*DCOS(X(2))*DEXP(-(X(1)-PI)**2 -(X(2)-PI)**2)
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE ROSENBROCK(M,X,F)
C     ROSENBROCK FUNCTION : M VARIABLE; MIN F (1, 1,...,1)=0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      DO I=1,M
      IF(X(I).LT.-5.D00 .OR. X(I).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=RAND*15-5.D00
      ENDIF
      ENDDO
      F=0.D00
      DO I=1,M-1
      F=F+ (100.D00*(X(I+1)-X(I)**2)**2 + (X(I)-1.D00)**2)
      ENDDO
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE CROSSLEG(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     CROSS-LEGGED TABLE FUNCTION ; -10<= X(I) <=10; M=2
C     MIN F(0 , X ) OR F(X, 0) = -1.
      PI=4*DATAN(1.D00)
        DO I=1,M
        IF( DABS(X(I)).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
        ENDDO
       F=-(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
     & (X(1)**2+X(2)**2)/PI))))+1.D00)**(-.1)
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE CROSS(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     CROSS FUNCTION ; -10<= X(I) <=10; M=2;
C     MIN F(A, B)=0 APPROX; A, B=1.3494 APPROX OF EITHER SIGN (+ OR -)
      PI=4*DATAN(1.D00)
        DO I=1,M
        IF( DABS(X(I)).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
        ENDDO
       F=(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
     & (X(1)**2+X(2)**2)/PI))))+1.D00)**(-.1)
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE CROSSINTRAY(M,X,F)
```

```
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     CROSS IN TRAY FUNCTION ; -10<= X(I) <=10; M=2;
C     MIN F(A, B)=-20626.1218 APPROX; A, B=1.3494 APPROX OF EITHER SIGN
      PI=4*DATAN(1.D00)
        DO I=1,M
        IF( DABS(X(I)).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
        ENDDO
      F=-(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
     & (X(1)**2+X(2)**2)/PI))))+1.D00)**(.1)
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE CROWNEDCROSS(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     CROWNED CROSS FUNCTION ; -10<= X(I) <=10; M=2; MIN F = 1
C     at (0, -9.4082), (-6.2832, 0)  minf =1
      PI=4*DATAN(1.D00)
        DO I=1,M
        IF( DABS(X(I)).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
        ENDDO
      F=(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-
     & (DSQRT(X(1)**2+X(2)**2)/PI))))+1.D00)**(.1)
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE TTHOLDER(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     TEST-TUBE HOLDER FUNCTION ; -10<= X(I) <=10; M=2;
C     MIN F([+/-]1.5706, 0)= -10.8723
      PI=4*DATAN(1.D00)
        DO I=1,M
        IF( DABS(X(I)).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
        ENDDO
      F=-4*DABS(DSIN(X(1))*DCOS(X(2))*DEXP(DABS(DCOS((X(1)**2+X(2)**2)/
     & 200))))
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE HOLDERTABLE(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     HOLDER-TABLE FUNCTION  ; -10<= X(I) <=10; M=2;
C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -26.92034 APPROX
      PI=4*DATAN(1.D00)
        DO I=1,M
        IF( DABS(X(I)).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
        ENDDO
      F=-DABS(DCOS(X(1))*DCOS(X(2))*DEXP(DABS(1.D00-(DSQRT(X(1)**2+
     & X(2)**2)/PI))))
      RETURN
      END
C     ------------------------------------------------------------------
```

```
      SUBROUTINE CARROMTABLE(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     CARROM-TABLE FUNCTION  ; -10<= X(I) <=10; M=2;
C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -24.15682 APPROX
      PI=4*DATAN(1.D00)
        DO I=1,M
        IF( DABS(X(I)).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
        ENDDO
      F=-1.D00/30*(DCOS(X(1))*DCOS(X(2))*DEXP(DABS(1.D00-
     & (DSQRT(X(1)**2 + X(2)**2)/PI))))**2
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE PENHOLDER(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     PENHOLDER FUNCTION  ; -11<= X(I) <=11; M=2;
C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -0.963535 APPROX
      PI=4*DATAN(1.D00)
        DO I=1,M
        IF( DABS(X(I)).GT.11.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*22
        ENDIF
        ENDDO
      F=-DEXP(-(DABS(DCOS(X(1))*DCOS(X(2))*DEXP(DABS(1.D0-(DSQRT
     & (X(1)**2+X(2)**2)/PI))))**(-1)))
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE BIRD(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     BIRD FUNCTION ; -2PI<= X(I) <=2PI; M=2;
C     MIN F (4.70104, 3.15294) APPROX = -106.764537 APPROX  OR
C     MIN F (-1.58214, -3.13024) APPROX = -106.764537 APPROX
      PI=4*DATAN(1.D00)
        DO I=1,M
        IF( DABS(X(I)).GT.2*PI) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*4*PI
        ENDIF
        ENDDO
      F=(DSIN(X(1))*DEXP((1.D00-DCOS(X(2)))**2) +
     & DCOS(X(2))*DEXP((1.D00-DSIN(X(1)))**2))+(X(1)-X(2))**2
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE CHICHINADZE(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     CHICHINADZE FUNCTION :  -30 <=X(I)<= 30;  M=2
C     MIN F (5.901329, 0.5) = -43.3158621
        PI=4*DATAN(1.D00)
        DO I=1,M
        IF( DABS(X(I)).GT.30) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*60
        ENDIF
        ENDDO
      F=X(1)**2-12*X(1)+11.D00+10*DCOS(PI*X(1)/2)+8*DSIN(5*PI*X(1))-
     & (1.D00/DSQRT(5.D00))*DEXP(-(X(2)-0.5D00)**2/2)
      RETURN
```

```
2815:       END
2816: C     ------------------------------------------------------------------
2817:       SUBROUTINE MCCORMICK(M,X,F)
2818:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2819:       DIMENSION X(*)
2820: C     MCCORMICK FUNCTION : -1.5<= X(1)<=4; -3<=X(2)<=4 ; M=2
2821: C     MIN F (-0.54719755, -1.54719755) = -1.913223 APPROX
2822:         IF(X(1).LT. -1.5D00 .OR. X(1) .GT. 4.D00) THEN
2823:         CALL RANDOM(RAND)
2824:         X(1)=RAND*5.5D00-1.5D00
2825:         ENDIF
2826:         IF(X(2).LT. -3.D00 .OR. X(2) .GT. 4.D00) THEN
2827:         CALL RANDOM(RAND)
2828:         X(2)=RAND*7.D00-3.D00
2829:         ENDIF
2830:       F=DSIN(X(1)+X(2))+(X(1)-X(2))**2-1.5*X(1)+2.5*X(2)+1.D00
2831:       RETURN
2832:       END
2833: C     ------------------------------------------------------------------
2834:       SUBROUTINE FENTONEASON(M,X,F)
2835:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2836:       DIMENSION X(*)
2837: C     FENTON & EASON FUNCTION  FMIN(1.74345, -2.029695) = 1.744152
2838:       DO I=1,M
2839:       IF(DABS(X(I)).GT.100.D00) THEN
2840:       CALL RANDOM(RAND)
2841:       X(I)=(RAND-0.5D00)*200
2842:       ENDIF
2843:       ENDDO
2844:       F=1.2D00+0.1*X(1)**2 +(0.1D00+0.1*X(2)**2)/X(1)**2+
2845:      & (.1*X(1)**2*X(2)**2+10.D00)/((X(1)*X(2))**4)
2846:       RETURN
2847:       END
2848: C     ------------------------------------------------------------------
2849:       SUBROUTINE WOOD(M,X,F)
2850:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
2851:       COMMON /RNDM/IU,IV
2852:       INTEGER IU,IV
2853:       DIMENSION X(*)
2854: C     WOOD FUNCTION:FMIN(0.443546,-0.194607,1.466077,2.15115)=1.09485393
2855:       DO I=1,M
2856:       IF(DABS(X(I)).GT.5.D00) THEN
2857:       CALL RANDOM(RAND)
2858:       X(I)=(RAND-0.5D00)*10
2859:       ENDIF
2860:       ENDDO
2861:       F1=100*(X(2)+X(1)**2)**2 + (1.D0-X(1))**2 +90*(X(4)-X(3)**2)**2
2862:       F2=(1.D0-X(3))**2 +10.1*((X(2)-1.D0)**2+(X(4)-1.D0)**2)
2863:       F3=19.8*(X(2)-1.D0)*(X(4)-1.D0)
2864:       F=F1+F2+F3
2865:       RETURN
2866:       END
2867: C     ------------------------------------------------------------------
2868:       SUBROUTINE GLANKWAHMDEE(M,X,F)
2869:       PARAMETER (N=5)
2870:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
2871:       COMMON /RNDM/IU,IV
2872:       INTEGER IU,IV
2873:       DIMENSION X(*),A(N,N), B(N)
2874: C     ------------------ GLANKWAHMDEE FUNCTION ------------------------
2875: C     GLANKWAHMDEE A, LIEBMAN JS AND HOGG GL (1979) "UNCONSTRAINED
2876: C     DISCRETE NONLINEAR PROGRAMMING. ENGINEERING OPTIMIZATION 4: 95-107
2877: C     PARSOPOULOS, KE AND VRAHATIS, MN (2002) RECENT APPROACHES TO
2878: C     GLOBAL OPTIMIZATION PROBLEMS THROUGH PARTICLE SWARM OPTIMIZATION,
2879: C     NATURAL COMPUTING 1: 235-306, 2002.  REPORT THE BEST OBTAINED
2880: C     FMIN (0,12,23,17,6)= -737 OR  MIN F(0, 11,22,16,6)= -737
2881: C     WE GET FMIN(-.232, 11.489, 22.273, 16.540, 6.115) = -739.822991
```

```
2882: C     ------------------------------------------------------------------
2883:       DATA ((A(I,J),J=1,N),I=1,N) /35,-20,-10,32,-10,-20, 40,-6,-31,32,
2884:      & -10,-6,11,-6,-10,32,-31,-6,38,-20,-10,32,-10,-20,31/
2885:       DATA (B(J),J=1,N) /-15,-27,-36,-18,-12/
2886: C     ------------------------------------------------------------------
2887:       DO I=1,M
2888:       IF(DABS(X(I)).GT.100.D00) THEN
2889:       CALL RANDOM(RAND)
2890:       X(I)=(RAND-0.5D00)*200
2891:       ENDIF
2892:       ENDDO
2893:       F=0.D0
2894:       DO J=1,M
2895:       F=F+B(J)*X(J)
2896:       ENDDO
2897:       DO I=1,M
2898:       C=0.D0
2899:       DO J=1,M
2900:       C=C+X(J)*A(J,I)
2901:       ENDDO
2902:       F=F+C*X(I)
2903:       ENDDO
2904:       RETURN
2905:       END
2906: C     ------------------------------------------------------------------
2907:       SUBROUTINE FLETCHER(M,X,F)
2908: C     FLETCHER-POWELL FUNCTION, M <= 10, ELSE IT IS VERY MUCH SLOW
2909: C     SOLUTION: MIN F = 0 FOR X=(C1, C2, C3,...,CM)
2910:       PARAMETER(N=10) ! FOR DIMENSION OF DIFFERENT MATRICES AND VECTORS
2911:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
2912:       COMMON /RNDM/IU,IV
2913:       INTEGER IU,IV
2914:       DIMENSION X(*),A(N,N),B(N,N),AA(N),BB(N),AL(N),C(N),C1(N)
2915:       PI=4*DATAN(1.D00)
2916: C     GENERATE A(I,J) AND B(I,J) BETWEEN (-100, 100) RANDOMLY.
2917: C     C(I) = BETWEEN (-PI, PI) IS EITHER GIVEN OR RANDOMLY GENERATED.
2918: C     DATA (C(I),I=1,10)/1,2,3,-3,-2,-1,0,1,2,3/ ! BETWEEN -PI AND PI
2919:       DATA (C1(I),I=1,N)/-3,-3.02,-3.01,1,1.03,1.02,1.03,-.08,.001,3/
2920: C     DATA (C1(I),I=1,N)/0,0,0,0,0,0,0,0,0,0/ ! ANOTHER EXAMPLE C1 = 0
2921:       NC=0  ! DEFINE NC HERE 0 OR 1 OR 2
2922: C     IF NC=0, C1 FROM DATA IS USED (THAT IS FIXED C);
2923: C     IF NC=1, C IS MADE FROM C1 BY ADDING RANDOM PART - THAT IS C=C1+R
2924: C     IF NC=2 THEN C IS PURELY RANDOM THAT IS C= 0 + R
2925: C     IN ANY CASE C LIES BETWEEN -PI AND PI.
2926: C     ------------------------------------------------------------------
2927: C     FIND THE MAX MAGNITUDE ELEMENT IN C1 VECTOR (UPTO M ELEMENTS)
2928:        CMAX=DABS(C1(1))
2929:        DO J=2,M
2930:        IF(DABS(C1(J)).GT.CMAX) CMAX=DABS(C1(J))
2931:        ENDDO
2932:        RANGE=PI-CMAX
2933: C     ------------------------------------------------------------------
2934:       DO J=1,M
2935:       DO I=1,M
2936:       CALL RANDOM(RAND)
2937:       A(I,J)=(RAND-0.5D00)*200.D00
2938:       CALL RANDOM(RAND)
2939:       B(I,J)=(RAND-0.5D00)*200.D00
2940:       ENDDO
2941:       IF(NC.EQ.0) AL(J)=C1(J) ! FIXED OR NON-STOCHASTIC C
2942:       IF(NC.EQ.1) THEN
2943:       CALL RANDOM(RAND)
2944:       AL(J)=C1(J)+(RAND-0.5D0)*2*RANGE ! A PART FIXED, OTHER STOCHASTIC
2945:       ENDIF
2946:       IF(NC.EQ.2) THEN
2947:       CALL RANDOM(RAND)
2948:       AL(J)=(RAND-0.5D00)*2*PI ! PURELY STOCHASTIC
```

```
2949:       ENDIF
2950:       ENDDO
2951:       DO I=1,M
2952:       AA(I)=0.D00
2953:       DO J=1,M
2954:       AA(I)=AA(I)+A(I,J)*DSIN(AL(J))+B(I,J)*DCOS(AL(J))
2955:       ENDDO
2956:       ENDDO
2957: C     ------------------------------------------------------------------
2958:       DO I=1,M
2959:       IF(DABS(X(I)).GT.PI) THEN
2960:       CALL RANDOM(RAND)
2961:       X(I)=(RAND-0.5D00)*2*PI
2962:       ENDIF
2963:       ENDDO
2964:       DO I=1,M
2965:       BB(I)=0.D00
2966:       DO J=1,M
2967:       BB(I)=BB(I)+A(I,J)*DSIN(X(J))+B(I,J)*DCOS(X(J))
2968:       ENDDO
2969:       ENDDO
2970:       F=0.D00
2971:       DO I=1,M
2972:       F=F+(AA(I)-BB(I))**2
2973:       ENDDO
2974:       RETURN
2975:       END
2976: C-----------------------------------------------------------------------
2977:       SUBROUTINE POWELL(M,X,F)
2978:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2979:       DIMENSION X(*)
2980: C     POWELL FUNCTION  ; -4<= X(I) <=5; M=A MULTIPLE OF 4;
2981: C     MIN F  =  0.0
2982:         DO I=1,M
2983:         IF(X(I).LT.-4.D00 .OR. X(I).GT.5.D00) THEN
2984:         CALL RANDOM(RAND)
2985:         X(I)=(RAND-0.5D00)*9+.5D00
2986:         ENDIF
2987:         ENDDO
2988:       M4=M/4
2989:       F=0.D00
2990:       DO I=1,M4
2991:       J=4*I
2992:       F=F+(X(J-3)+10*X(J-2))**2+5*(X(J-1)-X(J))**2+(X(J-2)-X(J-1))**4 +
2993:      & 10*(X(J-3)-X(J))**4
2994:       ENDDO
2995:       RETURN
2996:       END
2997: C-----------------------------------------------------------------------
2998:       SUBROUTINE HARTMANN(M,X,F)
2999:       PARAMETER (N=4)
3000:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
3001:       DIMENSION X(*),P(N,3),A(N,3),C(N)
3002: C     HARTMANN FUNCTION
3003: C     MIN F  =  -3.86278 APPROX  : 0 < X < 1.
3004:       DATA ((P(I,J),J=1,3),I=1,4) /0.6890,0.1170,0.2673,0.4699,0.4387,
3005:      &  0.7470,0.1091,0.8732,0.5547,0.0381,0.5743,0.8828/
3006:       DATA ((A(I,J),J=1,3),I=1,4) /3.0,10.0,30.0,0.1,10.0,35.0,3.0,
3007:      &  10.0,30.0,0.1,10.0,35.0/
3008:       DATA (C(J),J=1,4) /1.0,1.2,3.0,3.2/
3009:         DO I=1,M
3010:         IF(X(I).LE.0.D00 .OR. X(I).GE.1.D00) THEN
3011:         CALL RANDOM(RAND)
3012:         X(I)=RAND
3013:         ENDIF
3014:         ENDDO
3015:       F=0.D00
```

```
3016:       DO I=1,N
3017:       S=0.D00
3018:       DO J=1,M
3019:       S=S+A(I,J)*(X(J)-P(I,J))**2
3020:       ENDDO
3021:       F=F+C(I)*DEXP(-S)
3022:       ENDDO
3023:       F=-F
3024:       RETURN
3025:       END
3026: C-----------------------------------------------------------------------
3027:       SUBROUTINE COLVILLE(M,X,F)
3028:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
3029:       DIMENSION X(*)
3030: C     COLVILLE FUNCTION  ; -10<= X(I) <=10; M= 4;
3031: C     MINF(1,1,1,1)= 0.0
3032:         DO I=1,M
3033:         IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
3034:         CALL RANDOM(RAND)
3035:         X(I)=(RAND-0.5D00)*20
3036:         ENDIF
3037:         ENDDO
3038:       F=100*(X(1)**2-X(2))**2 + (X(1)-1.D00)**2 +(X(3)-1.D00)**2+
3039:      & 90*(X(3)**2-X(4))**2+10.1*((X(2)-1.D0)**2+(X(4)-1.D00)**2)+
3040:      & 19.8*(X(2)-1.D00)*(X(4)-1.D00)
3041:       RETURN
3042:       END
3043: C     -----------------------------------------------------------------
3044:       SUBROUTINE HIMMELBLAU(M,X,F)
3045:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
3046:       DIMENSION X(*)
3047:       NF=0    ! SET NF = 0 MULTIPLE OPTIMA NF=1 SINGLE OPTIMUM
3048:         DO I=1,M
3049:         IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
3050:         CALL RANDOM(RAND)
3051:         X(I)=(RAND-0.5D00)*20
3052:         ENDIF
3053:         ENDDO
3054: C     HIMMELBLAU FUNCTION. IT HAS MULTIPLE (4) GLOBAL OPTIMA : MINF=0
3055: C     (3, 2); (-2.8051, 3.1313); (3.5744, -1.8481); (-3.779, -3.283)
3056:       IF(NF.EQ.0) THEN
3057:       F= (X(1)**2+X(2)-11)**2+  (X(1)+X(2)**2-7)**2
3058:       RETURN
3059:       ENDIF
3060:       IF(NF.EQ.1) THEN
3061: C     MODIFIED HIMMELBLAU FUNCTION. IT HAS ONLY ONE GLOBAL OPTIMUM
3062: C     MINF=0 AT (3,2)
3063:       F= (X(1)**2+X(2)-11)**2+(X(1)+X(2)**2-7)**2+0.1D00*((X(1)-3)**2 +
3064:      1   (X(2)-2)**2)
3065:       ENDIF
3066:       RETURN
3067:       END
3068: C     -----------------------------------------------------------------
3069:       SUBROUTINE BEALE(M,X,F)
3070:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
3071:       DIMENSION X(*)
3072:         DO I=1,M
3073:         IF(X(I).LT.-4.500 .OR. X(I).GT.4.500) THEN
3074:         CALL RANDOM(RAND)
3075:         X(I)=(RAND-0.5D00)*9
3076:         ENDIF
3077:         ENDDO
3078: C     BEALE FUNCTION  : MINF =0 AT (3, 0.5)
3079:       F1=(1.5D00-X(1)+X(1)*X(2))**2
3080:       F2=(2.25D00-X(1)+X(1)*X(2)**2)**2
3081:       F3=(2.625D00-X(1)+X(1)*X(2)**3)**2
3082:       F=F1+F2+F3
```

```
3083:       RETURN
3084:       END
3085: C     ------------------------------------------------------------------
3086:       SUBROUTINE BOOTH(M,X,F)
3087:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
3088:       DIMENSION X(*)
3089:         DO I=1,M
3090:         IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
3091:         CALL RANDOM(RAND)
3092:         X(I)=(RAND-0.5D00)*20
3093:         ENDIF
3094:         ENDDO
3095: C     BOOTH FUNCTION  MINF (1,3)=0
3096:       F=(X(1)+2*X(2)-7.0D00)**2+(2*X(1)+X(2)-5.0D00)**2
3097:       RETURN
3098:       END
3099: C     ------------------------------------------------------------------
3100:       SUBROUTINE HUMP(M,X,F)
3101:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
3102:       DIMENSION X(*)
3103:         DO I=1,M
3104:         IF(X(I).LT.-5.D00 .OR. X(I).GT.5.D00) THEN
3105:         CALL RANDOM(RAND)
3106:         X(I)=(RAND-0.5D00)*10
3107:         ENDIF
3108:         ENDDO
3109: C     HUMP FUNCTION  MINF (0.0898, -0.7127) = -1.0316
3110:       F=4*X(1)**2 - 2.1D00*X(1)**4 + (X(1)**6)/3.D00 + X(1)*X(2) -
3111:      & 4*X(2)**2 + 4*X(2)**4
3112:       RETURN
3113:       END
3114: C     ------------------------------------------------------------------
3115:       SUBROUTINE MATYAS(M,X,F)
3116:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
3117:       DIMENSION X(*)
3118:         DO I=1,M
3119:         IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
3120:         CALL RANDOM(RAND)
3121:         X(I)=(RAND-0.5D00)*20
3122:         ENDIF
3123:         ENDDO
3124: C     MATYAS FUNCTION  MIN F (0, 0) = 0
3125:       F=0.26D00*(X(1)**2 + X(2)**2) - 0.48D00*X(1)*X(2)
3126:       RETURN
3127:       END
3128: C     ------------------------------------------------------------------
3129:       SUBROUTINE MISHRA_1(M,X,F)
3130:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
3131:       COMMON /RNDM/IU,IV
3132:       INTEGER IU,IV
3133:       DIMENSION X(*)
3134: C     MIN F (1, 1, ..., 1) =2
3135:       DO I=1,M
3136:       IF(X(I).LT.0.D00 .OR. X(I).GT.1.D00) THEN
3137:       CALL RANDOM(RAND)
3138:       X(I)=RAND
3139:       ENDIF
3140:       ENDDO
3141:       S=0.D00
3142:       DO I=1,M-1
3143:       S=S+X(I)
3144:       ENDDO
3145:       X(M)=(M-S)
3146:       F=(1.D00+X(M))**X(M)
3147:       RETURN
3148:       END
3149: C     ------------------------------------------------------------------
```

```
3150:       SUBROUTINE MISHRA_2(M,X,F)
3151:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
3152:       COMMON /RNDM/IU,IV
3153:       INTEGER IU,IV
3154:       DIMENSION X(*)
3155: C     MIN F (1, 1, ..., 1) =2
3156:       DO I=1,M
3157:       IF(X(I).LT.0.D00 .OR. X(I).GT.1.D00) THEN
3158:       CALL RANDOM(RAND)
3159:       X(I)=RAND
3160:       ENDIF
3161:       ENDDO
3162:       S=0.D00
3163:       DO I=1,M-1
3164:       S=S+(X(I)+X(I+1))/2.D00
3165:       ENDDO
3166:       X(M)=(M-S)
3167:       F=(1.D00+X(M))**X(M)
3168:       RETURN
3169:       END
3170: C     ----------------------------------------------------------------
3171:       SUBROUTINE ZAKHAROV(M,X,F)
3172:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
3173:       COMMON /RNDM/IU,IV
3174:       INTEGER IU,IV
3175:       DIMENSION X(*)
3176: C     ZAKHAROV FUNCTIONMIN MINF = (0, 0, ..., 0) =0
3177:       DO I=1,M
3178:       IF(X(I).LT.-5.D00 .OR. X(I).GT.10.D00) THEN
3179:       CALL RANDOM(RAND)
3180:       X(I)=(RAND-.5D0)*15 + 2.5D0
3181:       ENDIF
3182:       ENDDO
3183:       F1=0.D00
3184:       F2=0.D00
3185:       DO I=1, M
3186:       F1=F1+ X(I)**2
3187:       F2=F2 + I*X(I)/2.D00
3188:       ENDDO
3189:       F=F1+F2**2+F2**4
3190:       RETURN
3191:       END
3192: C     ----------------------------------------------------------------
3193:       SUBROUTINE MULTIMOD(M,X,F)
3194:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
3195:       COMMON /RNDM/IU,IV
3196:       INTEGER IU,IV
3197:       DIMENSION X(*)
3198: C     MULTIMODAL  FUNCTION  MINF = (0,0)= -1  : M=2
3199:       DO I=1,M
3200:       IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
3201:       CALL RANDOM(RAND)
3202:       X(I)=(RAND-.5D0)*20
3203:       ENDIF
3204:       ENDDO
3205: C     A TYPICAL MULTIMODAL FUNCTION
3206:       F=-DCOS(X(1))*DCOS(X(2))*DEXP(-(X(1)**2 + X(2)**2)**2/4.D00)
3207:       RETURN
3208:       END
3209: C     ----------------------------------------------------------------
3210:       SUBROUTINE NONLIN(M,X,F)
3211:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
3212:       COMMON /RNDM/IU,IV
3213:       INTEGER IU,IV
3214:       DIMENSION X(*)
3215: C     NONLINEAR  MULTIMODAL FUNCTION  MINF = 0
3216:       DO I=1,M
```

```
3217:       IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
3218:       CALL RANDOM(RAND)
3219:       X(I)=(RAND-.5D0)*20
3220:       ENDIF
3221:       ENDDO
3222:       F=0.D0
3223:       DO I=2,M
3224:       F=F+DCOS(DABS(X(I)-X(I-1)) / DABS(X(I-1)+X(I)))
3225:       ENDDO
3226:       F=F+(M-1.D00)
3227: C     IF 0.001*X(1) IS ADDED TO F, IT BECOMES UNIMODAL
3228: C     F=F+0.001*X(1)
3229:       RETURN
3230:       END
3231: C     ------------------------------------------------------------------
3232:       SUBROUTINE QUADRATIC(M,X,F)
3233:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
3234:       COMMON /RNDM/IU,IV
3235:       INTEGER IU,IV
3236:       DIMENSION X(*)
3237: C     QUADRATIC FUNCTION  MINF (0.19388, 0.48513) = -3873.7243 (M=2)
3238:       DO I=1,M
3239:       IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
3240:       CALL RANDOM(RAND)
3241:       X(I)=(RAND-.5D0)*200
3242:       ENDIF
3243:       ENDDO
3244:       F=-3803.84-138.08*X(1)-232.92*X(2)+128.08*X(1)**2+203.64*X(2)**2+
3245:      & 182.25*X(1)*X(2)
3246:       RETURN
3247:       END
3248: C     ------------------------------------------------------------------
3249:       SUBROUTINE TRIGON(M,X,F)
3250:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
3251:       COMMON /RNDM/IU,IV
3252:       INTEGER IU,IV
3253:       DIMENSION X(*)
3254: C     TRIGON FUNCTION F MIN (0, 0, 0,..., 0) OR (PI, 0, 0, ...,0) =0
3255:       PI=4*DATAN(1.D00)
3256:         DO I=1,M
3257:         IF(X(I).LT.0.D00 .OR. X(I).GT.PI) THEN
3258:         CALL RANDOM(RAND)
3259:         X(I)=RAND*PI
3260:         ENDIF
3261:         ENDDO
3262:       F=0.D00
3263:       DO I=2,M
3264:       F=F+(DCOS(I+0.D00)*DSIN(X(I)-X(I-1))**2 +
3265:      & (I-1.D00)*(1.D0-DCOS(X(I))))**2
3266:       ENDDO
3267:       RETURN
3268:       END
3269:
3270: C     ------------------------------------------------------------------
3271:       SUBROUTINE COMPOUND(M,X,F)
3272:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
3273:       COMMON /RNDM/IU,IV
3274:       INTEGER IU,IV
3275:       DIMENSION X(*)
3276: C     COMPOUND FUNCTION
3277:       PI=4*DATAN(1.D00)
3278:         DO I=1,M
3279:         IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
3280:         CALL RANDOM(RAND)
3281:         X(I)=(RAND-.5D0)*20
3282:         ENDIF
3283:         ENDDO
```

```
3284:       F1=DSIN(( DCOS(X(1))+DCOS(X(2)) )**2)**2
3285:       F2=DCOS(( DSIN(X(1))+DSIN(X(2)) )**2)**2
3286:       F3=DABS(DCOS(DSQRT(DABS(X(3)**2+X(4)))))**0.5 +0.01*X(3)+.01*X(4)
3287:       F=DEXP(DSIN(DCOS(F1+F3)+DCOS(F1+DCOS(F2))))
3288:       RETURN
3289:       END
3290: C     ------------------------------------------------------------------
3291: C     SUBROUTINE NAME(M,X,F)
3292: C      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
3293: C     PI=4*DATN(1.D0) ! IF NEEDED
3294: C      DIMENSION X(*)
3295: C     **** FUNCTION  ;  A<= X(I) <=B; M= MA;
3296: C     MINF( , , , )= SUCH & SUCH
3297: C        DO I=1,M
3298: C        IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
3299: C        CALL RANDOM(RAND)
3300: C        X(I)=(RAND-0.5D00)*APPROPRIATE + APPROPRATE ETC
3301: C        ENDIF
3302: C        ENDDO
3303: C      F= ETC
3304: C     RETURN
3305: C     END
```