The Barter Method: A New Heuristic for Global Optimization and its Comparison with the Particle Swarm and the Differential Evolution Methods

Mishra, SK

21 October 2006

The Barter Method: A New Heuristic for Global Optimization and its Comparison with the Particle Swarm and the Differential Evolution Methods

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

Introduction: The objective of this paper is to introduce a new population-based (stochastic) heuristic to search the global optimum of a (continuous) multi-modal function and to assess its performance (on a fairly large number of benchmark functions) vis-à-vis that of two other well-established and very powerful methods, namely, the Particle Swarm (PS) and the Differential Evolution (DE) methods of global optimization. We will call this new method the **Barter Method** of global optimization.

For the purpose of brevity we would not present here any introductory note on the Particle Swarm (or the Modified Repulsive Particle Swarm, MRPS, variant that we have used in this study) or the DE method. Such a note is available elsewhere [Mishra, 2006 (d) and (f)]. Additionally, there is a large literature on these methods.

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:

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, $(j,\ \ell)$ 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.

A Computer Program: A computer program (FORTRAN) that works out the global optimum of the test functions by the three methods (MRPS, DE and Barter) is appended. It incorporated 75 benchmark functions of varied types, some well-known and others new (proposed by the present author).

The Findings: In all, benchmark functions have been optimized 77 times. As presented in table-1, the DE succeeds in 70 cases, the RPS succeeds in 60 cases, while the Barter 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 eight cases, the Barter method could not converge in 10000 iterations (due to slow convergence rate), while in 4 cases the MRPS could not converge.

Table 1: A Summary of Success/Failure of DE, BARTER and RPS Methods on Test Functions

Test Function	Dim (M)	Success			Test Function	Dim (M)	Success		
		DE	BA	PS			BA	DE	PS
New Fn #1	2	yes	no	nc	Fenton-Eason Fn	2	yes	yes	yes
New Fn #2	2	yes	no	yes	Hougen Fn	5	yes	no	no
New Fn #3	2	yes	yes	yes	Giunta Fn	2	yes	yes	yes
New Fn #4	2	yes	yes	yes	Egg-Holder Fn	2	yes	yes	yes
New Fn #8	2	yes	yes	yes	Trid Fn	10	yes	nc	yes
Quintic Fn	10	yes	no	no	Greiwank Fn	10	yes	nc	nc
Needle-eye Fn	10	yes	no	no	Weierstrass Fn	10	yes	yes	no
Fletcher-Powell Fn (o)	5	yes	no	yes	Levy#3 Fn	2	yes	yes	yes
Fletcher-Powell Fn (1)	5	uns	uns	yes	Levy#5 Fn	2	yes	yes	yes
Fletcher-Powell Fn (2)	5	uns	uns	yes	Levy#8 Fn	3	yes	yes	yes
Powell Fn	8	yes	yes	yes	Colville Fn.	4	yes	yes	yes
Glankwahmdee Fn	5	yes	yes	yes	Hartmann Fn	3	yes	yes	yes
Zero-sum Fn	10	yes	no	no	Rastrigin Fn	10	yes	yes	no
Corana Fn	4	yes	yes	yes	Ackley Fn	10	yes	yes	yes
Mod RCos Fn	2	yes	yes	yes	Michalewicz Fn	10	yes	nc	yes
Freud-Roth Fn	2	yes	yes	yes	Schwefel Fn	10	yes	nc	yes
Anns XOR Fn	9	yes	no	no	Shubert Fn	2	yes	yes	yes
Perm #1 Fn	4	yes	nc	yes	Dixon-Price Fn	10	no	no	no
Perm #2 Fn	5	no	no	no	Shekel Fn	4	yes	no	yes
Power-Sum Fn	4	no	yes	yes	Paviani Fn	10	yes	yes	yes
Goldstein-Price Fn	2	yes	yes	yes	Branin#1 Fn	2	yes	yes	yes
Bukin-6 Fn	2	no	no	no	Branin#2 Fn	2	yes	yes	yes
DCS Fn	4	yes	no	yes	Bohachevsky#1 Fn	2	yes	yes	yes
New Factorial Fn	4	yes	no	yes	Bohachevsky#2 Fn	2	yes	yes	yes
New Decanomial Fn	2	yes	yes	yes	Bohachevsky#3 Fn	2	yes	yes	yes
Judge Fn	2	yes	yes	yes	Easom Fn	2	yes	yes	yes
New Dodecal Fn	3	yes	yes	yes	Rosenbrock Fn	10	yes	nc	yes
New sum=prod Fn	2	yes	yes	no	Crosslegged Table Fn	2	yes	yes	no
New AM=GM Fn	10	yes	no	yes	Cross Fn	2	yes	yes	yes
Yao-Liu#2 Fn	10	yes	yes	nc	Cross-in-Tray Fn	2	yes	yes	yes
Yao-Liu#3 Fn	10	yes	yes	yes	Crowned Cross Fn	2	yes	yes	no
Yao-Liu#4 Fn	10	yes	nc	nc	TT-Holder Fn	2	yes	yes	yes
Yao-Liu#6 Fn	10	yes	yes	yes	Holder Table Fn	2	yes	yes	yes
Yao-Liu#7 Fn	10	uns	uns	yes	Carrom Table Fn	2	yes	yes	yes
Yao-Liu#12 Fn	10	yes	yes	yes	Pen-Holder Fn	2	yes	yes	yes
Yao-Liu#13 Fn	10	yes	yes	yes	Bird Fn	2	yes	yes	yes
Yao-Liu#14 Fn	2	yes	yes	yes	Chichinadze Fn	2	yes	yes	yes
Yao-Liu#15 Fn	4	yes	nc	yes	McCormick Fn	2	yes	yes	yes
Wood's Fn	4	yes	yes	yes	**No. of Failure in 77 trials**		7	26	17

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

Seen as such, the barter method is inferior to the other two methods. Additionally, the convergence rate of the Barter method is slower than the DE as well as the MRPS. However, the DE and the RPS have a history of a full decade behind them and they have been improved many times. In the present exercise, the RPS is a modified version (MRPS) that has an extra ability for local search. The DE version used here uses 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.

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)
- 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.
- 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
      DIMENSION XX(3,50),KKF(3),MM(3),FMINN(3)
      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(*,*)'====================      WARNING     =============== '
      METHOD(1)=' : DIFFERENTIAL EVALUATION'
      METHOD(2)=' : BARTER ALGORITHM'
      METHOD(3)=' : REPULSIVE PARTICLE SWARM'
      DO I=1,3

      IF(I.EQ.1) THEN
      WRITE(*,*)'====== WELCOME TO DE, RPS, BARTER GO PROGRAMM ======='
      WRITE(*,*)'TO PROCEED TYPE ANY CHARACTER AND STRIKE ENTER'
      READ(*,*) PROCEED
      CALL DE(M,X,FMINDE) ! CALLS DE AND RETURNS OPTIMAL X AND FMIN
      FMIN=FMINDE
      ENDIF
C     ------------------------------------------------------------------
      IF(I.EQ.2) THEN
      WRITE(*,*)' '
      WRITE(*,*)' '
      WRITE(*,*)'==========BARTER ALGORITHM PROGRAM =========='
      WRITE(*,*)'TO PROCEED TYPE ANY CHARACTER AND STRIKE ENTER'
C      READ(*,*) PROCEED
      CALL BARTER(M,X,FMINEXC)! CALLS RPS AND RETURNS OPTIMAL X AND FMIN
      FMIN=FMINEXC
      ENDIF
C     -----------------------------------------------------------------
      IF(I.EQ.3) 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) ! CALLS RPS AND RETURNS OPTIMAL X AND FMIN
      FMIN=FMINRPS
      ENDIF

C     -----------------------------------------------------------------
      DO J=1,M
      XX(I,J)=X(J)
      ENDDO
      KKF(I)=KF
      MM(I)=M
      FMINN(I)=FMIN
      ENDDO
      WRITE(*,*)' '
      WRITE(*,*)' '
      WRITE(*,*)'---------------------- FINAL RESULTS=================='
      DO I=1,3
      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(*,*)'PROGRAM ENDED'
      END
C     -----------------------------------------------------------------
      SUBROUTINE DE(M,A,FBEST)
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=0.5, RX2=5.0) ! to be adjusted suitably, if needed
C     RX1 AND RX2 CONTROL THE SCHEME OF CROSSOVER. WHEN RX1=RX2=0, ONLY
C     SCHEME3 IS USED. WHEN RX1=RX2=0.5, 50% CASES SCHEME 1 AND REST
C     SCHEME 2 IS USED AND SO ON.
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
      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)
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
      CALL RANDOM(RAND)
      X(I,J)=(RAND-.5D00)*2000
C     RANDOM NUMBERS BETWEEN -RRANGE AND +RRANGE (BOTH EXCLUSIVE)
      ENDDO
      ENDDO
      WRITE(*,*)'COMPUTING --- PLEASE WAIT '
```

```fortran
      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
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
           ENDIF
           IF(IRI.EQ.2.AND.IRJ.NE.I.AND.IRJ.NE.IR(1)) THEN
           IR(IRI)=IRJ
           ENDIF
       IF(IRI.EQ.3.AND.IRJ.NE.I.AND.IRJ.NE.IR(1).AND.IRJ.NE.IR(2)) THEN
            IR(IRI)=IRJ
            ENDIF
            ENDDO   ! IRI LOOP ENDS
C     CHECK IF ALL THE THREE IR ARE POSITIVE (INTEGERS)
          DO IX=1,3
          IF(IR(IX).LE.0) THEN
          GOTO 20   ! IF NOT THEN REGENERATE
          ENDIF
          ENDDO
C     THREE RANDOMLY CHOSEN INDIVIDUALS DIFFERENT FROM I AND DIFFERENT
C     FROM EACH OTHER ARE IR(1),IR(2) AND IR(3)
C     ===================== randomization of ncross ===================
C     RANDOMIZES NCROSS
      NCROSS=0
      CALL RANDOM(RAND)
      IF(RAND.GT.RX1) NCROSS=1
      IF(RAND.GT.RX2) NCROSS=2

C     ----------------------------------------------------------------
C      NO CROSS OVER, ONLY REPLACEMENT THAT IS PROBABILISTIC
          IF(NCROSS.LE.0) THEN
          DO J=1,M      ! J LOOP BEGINS
          CALL RANDOM(RAND)
```

```
           IF(RAND.LE.PCROS) THEN ! REPLACE IF RAND < PCROS
           A(J)=X(IR(1),J)+(X(IR(2),J)-X(IR(3),J))*FACT ! CANDIDATE CHILD
           ENDIF
           ENDDO  ! J LOOP ENDS
           ENDIF

C      -----------------------------------------------------------------
C      CROSSOVER SCHEME (EXPONENTIAL) SUGGESTED BY KENNETH PRICE IN HIS
C      PERSONAL LETTER TO THE AUTHOR (DATED SEPTEMBER 29, 2006)
       IF(NCROSS.EQ.1) THEN
         CALL RANDOM(RAND)
    1    JR=INT(RAND*M)+1
         J=JR
    2    A(J)=X(IR(1),J)+FACT*(X(IR(2),J)-X(IR(3),J))
    3    J=J+1
         IF(J.GT.M) J=1
    4    IF(J.EQ.JR) GOTO 10
    5    CALL RANDOM(RAND)
         IF(PCROS.LE.RAND) GOTO 2
    6    A(J)=X(I,J)
    7    J=J+1
         IF(J.GT.M) J=1
    8    IF (J.EQ.JR) GOTO 10
    9    GOTO 6
   10    CONTINUE
       ENDIF
C      -----------------------------------------------------------------
C      CROSSOVER SCHEME (NEW) SUGGESTED BY KENNETH PRICE IN HIS
C      PERSONAL LETTER TO THE AUTHOR (DATED OCTOBER 18, 2006)
       IF(NCROSS.GE.2) THEN
           CALL RANDOM(RAND)
           IF(RAND.LE.PCROS) THEN
              CALL NORMAL(RN)
              DO J=1,M
              A(J)=X(I,J)+(X(IR(1),J)+ X(IR(2),J)-2*X(I,J))*RN
              ENDDO
            ELSE
             DO J=1,M
             A(J)=X(I,J)+(X(IR(1),J)- X(IR(2),J))! FACT ASSUMED TO BE 1
             ENDDO
            ENDIF
       ENDIF
C      -----------------------------------------------------------------
           CALL FUNC(A,M,F) ! EVALUATE THE OFFSPRING
           IF(F.LT.FV(I)) THEN ! IF BETTER, REPLACE PARENTS BY THE CHILD
           FV(I)=F
           DO J=1,M
           Y(I,J)=A(J)
           ENDDO
           ENDIF
       ENDDO   ! I LOOP ENDS
       DO I=1,N
       DO J=1,M
       X(I,J)=Y(I,J) ! NEW GENERATION IS A MIX OF BETTER PARENTS AND
C                      BETTER CHILDREN
       ENDDO
       ENDDO
       IPCOUNT=IPCOUNT+1
       IF(IPCOUNT.EQ.IPRINT) THEN
       DO J=1,M
       A(J)=X(KB,J)
       ENDDO
       WRITE(*,*)(X(KB,J),J=1,M),'  FBEST UPTO NOW = ',FBEST
       WRITE(*,*)'TOTAL NUMBER OF FUNCTION CALLS =',NFCALL
            IF(DABS(FBEST-GBEST).LT.EPS) THEN
            WRITE(*,*) FTIT
            WRITE(*,*)'COMPUTATION OVER'
```

```
              RETURN
              ELSE
              GBEST=FBEST
              ENDIF
         IPCOUNT=0
         ENDIF
C        ---------------------------------------------------------------
     100 ENDDO    ! ITERATION ENDS : GO FOR NEXT ITERATION, IF APPLICABLE
C        ---------------------------------------------------------------
         WRITE(*,*)'DID NOT CONVERGE. REDUCE EPS OR RAISE ITER OR DO BOTH'
         WRITE(*,*)'INCREASE N, PCROS, OR SCALE FACTOR (FACT)'
         RETURN
         END
C        ----------------------------------------------------------------
         SUBROUTINE NORMAL(R)
C        Program to generate N(0,1) from Rectangular Random Numbers
C        It uses Variate Transformation for this purpose.
C        ----------------------------------------------------------------
C        IF U1 AND U2 ARE UNIFORMLY DISTRIBUTED RANDOM NUMBERS (0,1),
C        THEN X=[(-2*ln(U1))**.5]*(COS(2*PI*U2) IS N(0,1)
C        PI = 4*ARCTAN(1.0)= 3.1415926535897932384626433832795
C        2*PI = 6.283185307179586476925286766559
C        ----------------------------------------------------------------
         IMPLICIT DOUBLE PRECISION (A-H,O-Z)
         COMMON /RNDM/IU,IV
         INTEGER IU,IV
C        ----------------------------------------------------------------
         CALL RANDOM(RAND) ! INVOKES RANDOM TO GENERATE UNIFORM RAND [0, 1]
         U1=RAND ! U1 IS UNIFORMLY DISTRIBUTED [0, 1]
         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*6.283185307179586476925286766559D00)
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)
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
      INTEGER IU,IV
      CHARACTER *70 FTIT
      DIMENSION X(N,MX),V(N,MX),A(MX),VI(MX)
      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)
C      CALL FSELECT(KF,M,FTIT)
C     -----------------------------------------------------------------
      GGBEST=1.D30 ! TO BE USED FOR TERMINATION CRITERION
      LCOUNT=0
      NFCALL=0
      WRITE(*,*)'4-DIGITS SEED FOR RANDOM NUMBER GENERATION'
      WRITE(*,*)'A FOUR-DIGIT POSITIVE ODD INTEGER, SAY, 1171'
C      READ(*,*) IU
      IU=1111
      FMIN=1.0E30
C     GENERATE N-SIZE POPULATION OF M-TUPLE PARAMETERS X(I,J) RANDOMLY
      DO I=1,N
        DO J=1,M
        CALL RANDOM(RAND)
         X(I,J)=(RAND-0.5D00)*2000
C     WE GENERATE RANDOM(-5,5). HERE MULTIPLIER IS 10. TINKERING IN SOME
C     CASES MAY BE NEEDED
         ENDDO
        F(I)=1.0D30
      ENDDO
C     INITIALISE VELOCITIES V(I) FOR EACH INDIVIDUAL IN THE POPULATION
      DO I=1,N
      DO J=1,M
      CALL RANDOM(RAND)
       V(I,J)=(RAND-0.5D+00)
C       V(I,J)=RAND
      ENDDO
      ENDDO
      DO 100 ITER=1,ITRN
C     LET EACH INDIVIDUAL SEARCH FOR THE BEST IN ITS NEIGHBOURHOOD
        DO I=1,N
           DO J=1,M
           A(J)=X(I,J)
           VI(J)=V(I,J)
           ENDDO
           CALL LSRCH(A,M,VI,NSTEP,FI)
           IF(FI.LT.F(I)) THEN
            F(I)=FI
```

```fortran
                 DO IN=1,M
                 BST(IN)=A(IN)
                 ENDDO
C      F(I) CONTAINS THE LOCAL BEST VALUE OF FUNCTION FOR ITH INDIVIDUAL
C      XX(I,J) IS THE M-TUPLE VALUE OF X ASSOCIATED WITH LOCAL BEST F(I)
                  DO J=1,M
                  XX(I,J)=A(J)
                  ENDDO
                  ENDIF
             ENDDO
C       NOW LET EVERY INDIVIDUAL RANDOMLY COSULT NN(<<N) COLLEAGUES AND
C       FIND THE BEST AMONG THEM
           DO I=1,N
C      ------------------------------------------------------------------
           IF(ITOP.GE.3) THEN
C      RANDOM TOPOLOGY ******************************************
C      CHOOSE NN COLLEAGUES RANDOMLY AND FIND THE BEST AMONG THEM
               BEST=1.0D30
                DO II=1,NN
                      CALL RANDOM(RAND)
                    NF=INT(RAND*N)+1
                     IF(BEST.GT.F(NF)) THEN
                      BEST=F(NF)
                     NFBEST=NF
                      ENDIF
                 ENDDO
           ENDIF
C-----------------------------------------------------------------------
           IF(ITOP.EQ.2) THEN
C      RING + RANDOM TOPOLOGY ******************************************
C      REQUIRES THAT THE SUBROUTINE NEIGHBOR IS TURNED ALIVE
            BEST=1.0D30
               CALL NEIGHBOR(I,N,I1,I3)
               DO II=1,NN
                     IF(II.EQ.1) NF=I1
                      IF(II.EQ.2) NF=I
                       IF(II.EQ.3) NF=I3
                           IF(II.GT.3) THEN
                          CALL RANDOM(RAND)
                           NF=INT(RAND*N)+1
                          ENDIF
                       IF(BEST.GT.F(NF)) THEN
                       BEST=F(NF)
                       NFBEST=NF
                      ENDIF
                  ENDDO
            ENDIF
C----------------------------------------------------------------------
           IF(ITOP.LE.1) THEN
C      RING TOPOLOGY **********************************************
C      REQUIRES THAT THE SUBROUTINE NEIGHBOR IS TURNED ALIVE
             BEST=1.0D30
                CALL NEIGHBOR(I,N,I1,I3)
                   DO II=1,3
                   IF (II.NE.I) THEN
                  IF(II.EQ.1) NF=I1
                   IF(II.EQ.3) NF=I3
                       IF(BEST.GT.F(NF)) THEN
                        BEST=F(NF)
                        NFBEST=NF
                       ENDIF
                       ENDIF
                 ENDDO
            ENDIF
C----------------------------------------------------------------------
C      IN THE LIGHT OF HIS OWN AND HIS BEST COLLEAGUES EXPERIENCE, THE
C      INDIVIDUAL I WILL MODIFY HIS MOVE AS PER THE FOLLOWING CRITERION
```

```
C     FIRST, ADJUSTMENT BASED ON ONES OWN EXPERIENCE
C     AND OWN BEST EXPERIENCE IN THE PAST (XX(I))
          DO J=1,M
          CALL RANDOM(RAND)
          V1(J)=A1*RAND*(XX(I,J)-X(I,J))
C     THEN BASED ON THE OTHER COLLEAGUES BEST EXPERIENCE WITH WEIGHT W
C     HERE W IS CALLED AN INERTIA WEIGHT 0.01< W < 0.7
C     A2 IS THE CONSTANT NEAR BUT LESS THAN UNITY
          CALL RANDOM(RAND)
          V2(J)=V(I,J)
          IF(F(NFBEST).LT.F(I)) THEN
          V2(J)=A2*W*RAND*(XX(NFBEST,J)-X(I,J))
          ENDIF
C     THEN SOME RANDOMNESS AND A CONSTANT A3 CLOSE TO BUT LESS THAN UNITY
          CALL RANDOM(RAND)
          RND1=RAND
          CALL RANDOM(RAND)
           V3(J)=A3*RAND*W*RND1
C            V3(J)=A3*RAND*W
C     THEN ON PAST VELOCITY WITH INERTIA WEIGHT W
          V4(J)=W*V(I,J)
C     FINALLY A SUM OF THEM
          V(I,J)= V1(J)+V2(J)+V3(J)+V4(J)
          ENDDO
      ENDDO
C     CHANGE X
      DO I=1,N
      DO J=1,M
      RANDS=0.D00
C     ------------------------------------------------------------------
      IF(NSIGMA.EQ.1) THEN
       CALL RANDOM(RAND) ! FOR CHAOTIC PERTURBATION
       IF(DABS(RAND-.5D00).LT.SIGMA) RANDS=RAND-0.5D00
C     SIGMA CONDITIONED RANDS INTRODUCES CHAOTIC ELEMENT IN TO LOCATION
C     IN SOME CASES THIS PERTURBATION HAS WORKED VERY EFFECTIVELY WITH
C     PARAMETER (N=100,NN=15,MX=100,NSTEP=9,ITRN=100000,NSIGMA=1,ITOP=2)
      ENDIF
C     -----------------------------------------------------------------
      X(I,J)=X(I,J)+V(I,J)*(1.D00+RANDS)
      ENDDO
      ENDDO
       DO I=1,N
         IF(F(I).LT.FMIN) THEN
         FMIN=F(I)
         II=I
         DO J=1,M
         BST(J)=XX(II,J)
         ENDDO
         ENDIF
         ENDDO
      IF(LCOUNT.EQ.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
          RETURN
          ELSE
          GGBEST=FMIN
          ENDIF
      ENDIF
      LCOUNT=LCOUNT+1
  100 CONTINUE
      WRITE(*,*)'COMPUTATION OVER:',FTIT
      FMINIM=FMIN
      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
C     EITHER PURE OR HYBRIDIZED
       SUBROUTINE NEIGHBOR(I,N,J,K)
       IF(I-1.GE.1 .AND. I.LT.N) THEN
       J=I-1
       K=I+1
       ELSE
       IF(I-1.LT.1) THEN
       J=N-I+1
       K=I+1
       ENDIF
       IF(I.EQ.N) THEN
       J=I-1
       K=1
       ENDIF
       ENDIF
       RETURN
       END
c     ------------------------------------------------------------------
c     BARTER ALGORITHM
C     ------------------------------------------------------------------
C     *************** THIS METHOD IS PROPOSED BY SK MISHRA ************
C     PROGRAM BY SK MISHRA, DEPT. OF ECONOMICS, NEHU, SHILLONG (INDIA)
C     ------------------------------------------------------------------
      SUBROUTINE BARTER(M,BEST,FBEST)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z) ! TYPE DECLARATION
      PARAMETER(IPRINT=500, EPS=1.D-08)
      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
      CHARACTER *70 FTIT ! TITLE OF THE FUNCTION
C     ------------------------------------------------------------------

C     -----------------------------------------------------------------
      DIMENSION X(500,50),FV(500),A(50),B(50),BEST(50)
C     -----------------------------------------------------------------
C     ------- SELECT THE FUNCTION TO MINIMIZE AND ITS DIMENSION -------
C     CALL FSELECT(KF,M,FTIT)
C     SPECIFY OTHER PARAMETERS ----------------------------------------
      WRITE(*,*)'POPULATION SIZE [N] AND NO. OF ITERATIONS [ITER] ?'
C     READ(*,*) N,ITER
C     -----------------------------------------------------------------
```

```fortran
      N=M*10
      IF(N.LT.100) N=100
      ITER=10000
      WRITE(*,*)'RANDOM NUMBER SEED ?'
C      READ(*,*) IU
      IU=1111
C     -----------------------------------------------------------------
      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
      CALL RANDOM(RAND)
      X(I,J)=(RAND-.5D00)*2000
C     RANDOM NUMBERS BETWEEN -RRANGE AND +RRANGE (BOTH EXCLUSIVE)
      ENDDO
      ENDDO
      WRITE(*,*)'COMPUTING --- PLEASE WAIT '
      IPCOUNT=0
      DO 100 ITR=1,ITER  ! ITERATION BEGINS

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,FA)
      FV(I)=FA
      CALL SEARCH(A,M,FI)
C     STORE FUNCTION VALUES IN FV VECTOR
      IF(FI.LT.FV(I)) THEN
      DO J=1,M
      X(I,J)=A(J)
      ENDDO
      FV(I)=FI
      ENDIF
      ENDDO
C     ----------------------------------------------------------------
      DO I=1,N
C     CHOOSE IB TH INDIVIDUAL RANDOMLY
      CALL RANDOM(RAND)
      IB=INT(RAND*N)+1 !THE RANDOM INDIVIDUAL
C     IB=2*M+1
C     STORE ITH IN A AND RANDOMLY SELECTED INDIVIDUAL IN B
         DO J=1,M
         A(J)=X(I,J)
         B(J)=X(IB,J) ! of the individual randomly selected
         ENDDO
C     CHOSE AN INDEX BETWEEN 1 AND M RANDOMLY
          CALL RANDOM(RAND)
         JA=INT(RAND*M)+1
C     CHOOSE ANOTHER INDEX RANDOMLY : MUST BE DIFFERENT FROM JA
   1      CALL RANDOM(RAND)
          JB=INT(RAND*M)+1
          IF(JA.EQ.JB) GOTO 1
C     EXCHANGE A(JA) WITH B(JB)
           TEMP1=A(JA)
           TEMP2=B(JB)
            CALL NORMAL(RN)! OBTAIN STANDARD NORMAL  RANDOM NUMBER
           A(JB)=A(JB)+RN*TEMP2
           B(JB)=B(JB)-RN*TEMP2
           A(JA)=A(JA)-RN*TEMP1
           B(JA)=B(JA)+RN*TEMP1
C     EVALUATE A AND B VECTORS
            CALL FUNC(A,M,FA)
            CALL FUNC(B,M,FB)
C     CHECK IF FA < FV(I) AND FB < FV(IB)
```

```
          IF(FA.lT.FV(I) .and. FB.lt.FV(IB)) THEN
          FV(I)=FA
          FV(IB)=FB
          DO J=1,M
          X(I,J)=A(J)
          X(IB,J)=B(J)
          ENDDO
          ENDIF
      ENDDO
C     ----------------------------------------------------------------
C     FIND THE  BEST
      FBEST=1.D30
      DO I=1,N
      IF(FV(I).LT.FBEST) THEN
      FBEST=FV(I)
      KB=I
      ENDIF
      ENDDO
      DO J=1,M
      BEST(J)=X(KB,J)
      ENDDO
C     ----------------------------------------------------------------
      IPCOUNT=IPCOUNT+1
      IF(IPCOUNT.EQ.IPRINT) THEN
      WRITE(*,*)(BEST(J),J=1,M),'  FBEST UPTO NOW = ',FBEST
      WRITE(*,*)'TOTAL NUMBER OF FUNCTION CALLS =',NFCALL
          IF(DABS(FBEST-GBEST).LT.EPS) THEN
          WRITE(*,*) FTIT
          WRITE(*,*)'COMPUTATION OVER'
          RETURN
          ELSE
          GBEST=FBEST
          ENDIF
      IPCOUNT=0
      ENDIF
C     ----------------------------------------------------------------
  100 ENDDO    ! ITERATION ENDS : GO FOR NEXT ITERATION, IF APPLICABLE
C     ----------------------------------------------------------------
      WRITE(*,*)'DID NOT CONVERGE. REDUCE EPS OR RAISE ITER OR DO BOTH'
      WRITE(*,*)'INCREASE N, PCROS, OR SCALE FACTOR (FACT)'
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE SEARCH(A,M,FI)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION A(*),B(100)
      NSTEP=11
      AMN=1.0D30
      DO J=1,NSTEP
         DO JJ=1,M
         CALL RANDOM(RAND)
         B(JJ)=A(JJ)+(J-(NSTEP/2)-1)*RAND*0.0001D00
         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     -----------------------------------------------------------------
```

```
      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'/
      DATA TIT(50)/'KF=50 DIXON-PRICE FUNCTION : M-VARIABLES M=?'/
      DATA TIT(51)/'KF=51 SHEKEL FUNCTION : 4-VARIABLES M=4'/
      DATA TIT(52)/'KF=52 PAVIANI FUNCTION : 10-VARIABLES M=10'/
      DATA TIT(53)/'KF=53 BRANIN FUNCTION#1 : 2-VARIABLES M=2'/
      DATA TIT(54)/'KF=54 BRANIN FUNCTION#2 : 2-VARIABLES M=2'/
      DATA TIT(55)/'KF=55 BOHACHEVSKY FUNCTION#1 : 2-VARIABLES M=2'/
      DATA TIT(56)/'KF=56 BOHACHEVSKY FUNCTION#2 : 2-VARIABLES M=2'/
      DATA TIT(57)/'KF=57 BOHACHEVSKY FUNCTION#3 : 2-VARIABLES M=2'/
      DATA TIT(58)/'KF=58 EASOM FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(59)/'KF=59 ROSENBROCK FUNCTION : M-VARIABLES M=?'/
      DATA TIT(60)/'KF=60 CROSS-LEGGED TABLE FUNCTION:2-VARIABLES M=2'/
      DATA TIT(61)/'KF=61 CROSS FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(62)/'KF=62 CROSS-IN-TRAY FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(63)/'KF=63 CROWNED CROSS FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(64)/'KF=64 TT-HOLDER FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(65)/'KF=65 HOLDER-TABLE FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(66)/'KF=66 CARROM-TABLE FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(67)/'KF=67 PENHOLDER FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(68)/'KF=68 BIRD FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(69)/'KF=69 CHICHINADZE FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(70)/'KF=70 MCCORMICK FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(71)/'KF=71 GLANKWAHMDEE FUNCTION : 5-VARIABLES M=5'/
      DATA TIT(72)/'KF=72 FLETCHER-POWELL FUNCTION : M-VARIABLES M=?'/
      DATA TIT(73)/'KF=73 POWELL FUNCTION: M-VARIABLES M (MULT OF 4)=?'/
      DATA TIT(74)/'KF=74 HARTMANN FUNCTION: 3-VARIABLES M=3'/
      DATA TIT(75)/'KF=75 COLVILLE FUNCTION: 4-VARIABLES M=4'/
C     ----------------------------------------------------------------
      DO I=1,75
      WRITE(*,*)TIT(I)
      ENDDO
      WRITE(*,*)'----------------------------------------------------'
      WRITE(*,*)'FUNCTION CODE [KF] AND NO. OF VARIABLES [M] ?'
      READ(*,*) KF,M
      FTIT=TIT(KF) ! STORE THE NAME OF THE CHOSEN FUNCTION IN FTIT
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE FUNC(X,M,F)
C     TEST FUNCTIONS FOR GLOBAL OPTIMIZATION PROGRAM
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      COMMON /KFF/KF,NFCALL,FTIT
      INTEGER IU,IV
      DIMENSION X(*)
      CHARACTER *70 FTIT
      PI=4.D+00*DATAN(1.D+00)! DEFINING THE VALUE OF PI
      NFCALL=NFCALL+1 ! INCREMENT TO NUMBER OF FUNCTION CALLS
C     KF IS THE CODE OF THE TEST FUNCTION
C     ----------------------------------------------------------------
      IF(KF.EQ.1) THEN
C     FUNCTION #1 MIN AT -0.18467 APPROX AT (-8.4666, -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(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
C     ----------------------------------------------------------------
      IF(KF.EQ.6) THEN
C     NEEDLE-EYE FUNCTION M=>1;
C     MIN = 1 IF ALL ABS(X) ARE SMALLER THAN THE EYE
C     SMALLER THE VALUE OF ZZ, MORE DIFFICULT TO ENTER THE EYE
C     LARGER THE VALUE OF M, MORE DIFFICULT TO FIND THE OPTIMUM
      F=0.D00
      EYE=0.000001D00
      FP=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.EYE) THEN
      FP=1.D00
       F=F+100.D00+DABS(X(I))
      ELSE
      F=F+1.D00
      ENDIF
      ENDDO
      IF(FP.EQ.0.D00) F=F/M
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.7) THEN
C     ZERO SUM FUNCTION : MIN  = 0 AT SUM(X(I))=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
      SUM=0.D00
      DO I=1,M
      SUM=SUM+X(I)
      ENDDO
      IF(SUM.NE.0.D00) F=1.D00+(10000*DABS(SUM))**0.5
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.8) THEN
C     CORANA FUNCTION : MIN = 0 AT (0, 0, 0, 0) APPROX
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.1000.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*2000
      ENDIF
      ENDDO
      DO J=1,M
        IF(J.EQ.1) DJ=1.D00
        IF(J.EQ.2) DJ=1000.D00
        IF(J.EQ.3) DJ=10.D00
        IF(J.EQ.4) DJ=100.D00
            ISGNXJ=1
            IF(X(J).LT.0.D00) ISGNXJ=-1
            ZJ=(DABS(X(J)/0.2D00)+0.49999)*ISGNXJ*0.2D00
            ISGNZJ=1
            IF(ZJ.LT.0.D00) ISGNZJ=-1
            IF(DABS(X(J)-ZJ).LT.0.05D00) THEN
            F=F+0.15D00*(ZJ-0.05D00*ISGNZJ)**2 * DJ
            ELSE
            F=F+DJ*X(J)**2
            ENDIF
      ENDDO
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.9) THEN
C     MODIFIED RCOS FUNCTION MIN=-0.179891 AT (-3.196989, 12.52626)APPRX
      F=0.D00
      IF(X(1).LT.-5.D00 .OR. X(1).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(1)=RAND*15.D00 -5.D00
      ENDIF
      IF(X(2).LT.0.D00 .OR. X(2).GT.15.D00) THEN
      CALL RANDOM(RAND)
      X(2)=RAND*15.D00
      ENDIF
      CA=1.D00
      CB=5.1/(4*PI**2)
      CC=5.D00/PI
      CD=6.D00
      CE=10.D00
      CF=1.0/(8*PI)
      F1=CA*(X(2)-CB*X(1)**2+CC*X(1)-CD)**2
      F2=CE*(1.D00-CF)*DCOS(X(1))*DCOS(X(2))
      F3=DLOG(X(1)**2+X(2)**2+1.D00)
      F=-1.0/(F1+F2+F3+CE)
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.10) THEN
C     FREUDENSTEIN ROTH  FUNCTION : MIN  = 0 AT (5, 4)
      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=(-13.D00+X(1)+((5.D00-X(2))*X(2)-2)*X(2))**2
      F2=(-29.D00+X(1)+((X(2)+1.D00)*X(2)-14.D00)*X(2))**2
      F=F1+F2
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.11) THEN
C     ANNS XOR FUNCTION (PARSOPOULOS, KE, PLAGIANAKOS, VP, MAGOULAS, GD
C     AND VRAHATIS, MN "STRETCHING TECHNIQUE FOR OBTAINING GLOBAL
C     MINIMIZERS THROUGH PARTICLE SWARM OPTIMIZATION")
C     MIN=0.9597588 FOR X=(1, -1, 1, -1, -1, 1, 1, -1, 0.421134) APPROX
C     OBTAINED BY DIFFERENTIAL EVOLUTION PROGRAM
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.1.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*2
      ENDIF
      ENDDO
      F11=X(7)/(1.D00+DEXP(-X(1)-X(2)-X(5)))
      F12=X(8)/(1.D00+DEXP(-X(3)-X(4)-X(6)))
      F1=(1.D00+DEXP(-F11-F12-X(9)))**(-2)
      F21=X(7)/(1.D00+DEXP(-X(5)))
      F22=X(8)/(1.D00+DEXP(-X(6)))
      F2=(1.D00+DEXP(-F21-F22-X(9)))**(-2)
      F31=X(7)/(1.D00+DEXP(-X(1)-X(5)))
      F32=X(8)/(1.D00+DEXP(-X(3)-X(6)))
      F3=(1.D00-(1.D00+DEXP(-F31-F32-X(9)))**(-1))**2
      F41=X(7)/(1.D00+DEXP(-X(2)-X(5)))
      F42=X(8)/(1.D00+DEXP(-X(4)-X(6)))
      F4=(1.D00-(1.D00+DEXP(-F41-F42-X(9)))**(-1))**2
      F=F1+F2+F3+F4
      RETURN
      ENDIF
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
C         DECANOMIAL FUNCTION, MIN =0 AT X=(2, -3)
            DO I=1,M
            IF(DABS(X(I)).GT.4.D00) THEN
            CALL RANDOM(RAND)
            X(I)= (RAND-0.5D00)*8
            ENDIF
            ENDDO
          CALL DECANOM(M,F,X)
          RETURN
          ENDIF
C         -----------------------------------------------------------------
          IF(KF.EQ.21) THEN
C         JUDGE'S FUNCTION  F(0.864, 1.23) = 16.0817; M=2
          CALL JUDGE(M,X,F)
          RETURN
          ENDIF
C         -----------------------------------------------------------------
          IF(KF.EQ.22) THEN
C         DODECAL FUNCTION
          CALL DODECAL(M,F,X)
          RETURN
          ENDIF
C         -----------------------------------------------------------------
          IF(KF.EQ.23) THEN
C         WHEN X(1)*X(2)=X(1)*X(2) ? M=2
          CALL SEQP(M,F,X)
```

```
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.24) THEN
C     WHEN ARITHMETIC MEAN = GEOMETRIC MEAN ? : M =>1
      CALL AMGM(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.25) THEN
C     M =>2
      CALL FUNCT2(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.26) THEN
C     M =>2
      CALL FUNCT3(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.27) THEN
C     M =>2
      CALL FUNCT4(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.28) THEN
C     M =>2
      CALL FUNCT6(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.29) THEN
C     M =>2
      CALL FUNCT7(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.30) THEN
C     M =>2
      CALL FUNCT12(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.31) THEN
C     M =>2
      CALL FUNCT13(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.32) THEN
C     M =2
      CALL FUNCT14(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.33) THEN
C     M =4
      CALL FUNCT15(M,F,X)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.34) THEN
C     WOOD FUNCTION : F MIN  : M=4
      CALL WOOD(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.35) THEN
C     FENTON & EASON FUNCTION :    : M=2
      CALL FENTONEASON(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.36) THEN
C     HOUGEN FUNCTION 5 VARIABLES : M =3
      CALL HOUGEN(X,M,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.37) THEN
C     GIUNTA FUNCTION 2 VARIABLES   :M =2
      CALL GIUNTA(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.38) THEN
C     EGGHOLDER FUNCTION M VARIABLES
      CALL EGGHOLD(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.39) THEN
C     TRID FUNCTION M VARIABLES
      CALL TRID(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.40) THEN
C     GRIEWANK FUNCTION M VARIABLES
      CALL GRIEWANK(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.41) THEN
C     WEIERSTRASS FUNCTION M VARIABLES
      CALL WEIERSTRASS(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.42) THEN
C     LEVY-3 FUNCTION 2 VARIABLES
      CALL LEVY3(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.43) THEN
C     LEVY-5 FUNCTION 2 VARIABLES
      CALL LEVY5(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.44) THEN
C     LEVY-8 FUNCTION 3 VARIABLES
      CALL LEVY8(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.45) THEN
C     RASTRIGIN FUNCTION M VARIABLES
      CALL RASTRIGIN(M,X,F)
      RETURN
      ENDIF
```

```
C     -----------------------------------------------------------------
      IF(KF.EQ.46) THEN
C     ACKLEY FUNCTION M VARIABLES
      CALL ACKLEY(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.47) THEN
C     MICHALEWICZ FUNCTION M VARIABLES
      CALL MICHALEWICZ(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.48) THEN
C     SCHWEFEL FUNCTION M VARIABLES
      CALL SCHWEFEL(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.49) THEN
C     SHUBERT FUNCTION 2 VARIABLES
      CALL SHUBERT(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.50) THEN
C     DIXON AND PRICE FUNCTION M VARIABLES
      CALL DIXPRICE(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.51) THEN
C     SHEKEL FUNCTION 4 VARIABLES
      CALL SHEKEL(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.52) THEN
C     PAVIANI FUNCTION 10 VARIABLES
      CALL PAVIANI(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.53) THEN
C     BRANIN FUNCTION#1 2 VARIABLES
      CALL BRANIN1(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.54) THEN
C     BRANIN FUNCTION#2 2 VARIABLES
      CALL BRANIN2(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.55) THEN
C     BOHACHEVSKY FUNCTION#1 2 VARIABLES
      CALL BOHACHEVSKY1(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.56) THEN
C     BOHACHEVSKY FUNCTION#2 2 VARIABLES
      CALL BOHACHEVSKY2(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
```

```
      IF(KF.EQ.57) THEN
C     BOHACHEVSKY FUNCTION#3 2 VARIABLES
      CALL BOHACHEVSKY3(M,X,F)
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.58) THEN
C     EASOM FUNCTION#3 2 VARIABLES
      CALL EASOM(M,X,F)
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.59) THEN
C     ROSENBROCK FUNCTION  M VARIABLES
      CALL ROSENBROCK(M,X,F)
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.60) THEN
C     CROSS-LEGGED TABLE FUNCTION : 2 VARIABLES
      CALL CROSSLEG(M,X,F)
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.61) THEN
C     CROSS FUNCTION : 2 VARIABLES
      CALL CROSS(M,X,F)
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.62) THEN
C     CROSS-IN-TRAY FUNCTION : 2 VARIABLES
      CALL CROSSINTRAY(M,X,F)
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.63) THEN
C     CROWNED-CROSS FUNCTION : 2 VARIABLES
      CALL CROWNEDCROSS(M,X,F)
      RETURN
      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
C     -----------------------------------------------------------------
      IF(KF.EQ.74) THEN
C     HARTMANN Function
      CALL HARTMANN(M,X,F)
      RETURN
      ENDIF
C     -----------------------------------------------------------------
      IF(KF.EQ.75) THEN
C     COVILLE Function
      CALL COLVILLE(M,X,F)
      RETURN
      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
C     FIND X(I), I=1,2,...,M SUCH THAT THEIR PRODUCT IS EQUAL TO FACT.
C     LARGER THE VALUE OF M (=>8) OR SO, HARDER IS THE PROBLEM
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      FACT=1.D00
      P=1.D00
      DO I=1,M
      FACT=FACT*I
      P=P*X(I)
      F=F+DABS(P-FACT)**2
      ENDDO
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE DECANOM(M,F,X)
C     DECANOMIAL FUNCTION; MIN (2, -3) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F1= DABS(X(1)**10-20*X(1)**9+180*X(1)**8-960*X(1)**7+
     & 3360*X(1)**6-8064*X(1)**5+13340*X(1)**4-15360*X(1)**3+
     & 11520*X(1)**2-5120*X(1)+2624.D00)
      F2= DABS(X(2)**4+12*X(2)**3+54*X(2)**2+108*X(2)+81.D00)
      F=0.001D00*(F1+F2)**2
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE JUDGE(M,X,F)
      PARAMETER (N=20)
C     THIS SUBROUTINE IS FROM THE EXAMPLE IN JUDGE ET AL., THE THEORY
C     AND PRACTICE OF ECONOMETRICS, 2ND ED., PP. 956-7. THERE ARE TWO
C     OPTIMA: F(0.86479,1.2357)=16.0817307 (WHICH IS THE GLOBAL MINUMUM)
C     AND F(2.35,-0.319)=20.9805 (WHICH IS LOCAL). ADAPTED FROM BILL
C     GOFFE'S SIMMAN (SIMULATED ANNEALING) PROGRAM
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION Y(N), X2(N), X3(N), X(*)
        DATA (Y(I),I=1,N)/4.284,4.149,3.877,0.533,2.211,2.389,2.145,
     & 3.231,1.998,1.379,2.106,1.428,1.011,2.179,2.858,1.388,1.651,
     & 1.593,1.046,2.152/
        DATA (X2(I),I=1,N)/.286,.973,.384,.276,.973,.543,.957,.948,.543,
     & .797,.936,.889,.006,.828,.399,.617,.939,.784,.072,.889/
       DATA (X3(I),I=1,N)/.645,.585,.310,.058,.455,.779,.259,.202,.028,
     & .099,.142,.296,.175,.180,.842,.039,.103,.620,.158,.704/
```

```fortran

      F=0.D00
      DO I=1,N
      F=F+(X(1) + X(2)*X2(I) + (X(2)**2)*X3(I) - Y(I))**2
      ENDDO
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE DODECAL(M,F,X)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     DODECAL POLYNOMIAL MIN F(1,2,3)=0
      DO I=1,M
      IF(DABS(X(I)).GT.5.D0) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*10
      ENDIF
      ENDDO
      F=0.D00
      F1=2*X(1)**3+5*X(1)*X(2)+4*X(3)-2*X(1)**2*X(3)-18.D00
      F2=X(1)+X(2)**3+X(1)*X(2)**2+X(1)*X(3)**2-22.D00
      F3=8*X(1)**2+2*X(2)*X(3)+2*X(2)**2+3*X(2)**3-52.D00
      F=(F1*F3*F2**2+F1*F2*F3**2+F2**2+(X(1)+X(2)-X(3))**2)**2
      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
C     ----------------------------------------------------------------
      SUBROUTINE FUNCT3(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F (0, 0, ... , 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      F1=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*200
      ENDIF
      ENDDO
      DO I=1,M
      F1=0.D00
      DO J=1,I
      F1=F1+X(J)**2
      ENDDO
      F=F+F1
      ENDDO
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE FUNCT4(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F (0, 0, ..., 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      DO I=1,M
      IF(X(I).LT.0.D00 .OR. X(I).GE.M) THEN
      CALL RANDOM(RAND)
      X(I)=RAND*2*M
      ENDIF
      ENDDO
C     FIND MAX(X(I))=MAX(ABS(X(I))) NOTE: HERE X(I) CAN BE ONLY POSITIVE
      XMAX=X(1)
      DO I=1,M
      IF(XMAX.LT.X(I)) XMAX=X(I)
      ENDDO
      F=XMAX
      RETURN
```

```fortran
      END
C     -----------------------------------------------------------------
      SUBROUTINE FUNCT6(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F (-.5, -.5, ..., -.5) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*200
      ENDIF
      ENDDO
      DO I=1,M
       F=F+(X(I)+0.5D00)**2
      ENDDO
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE FUNCT7(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F(0, 0, ..., 0) = 0
      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)
      X(I)=(RAND-.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
      F2=F2+ (X(I)-1.D00)**2 * (1.D00+(DSIN(3*PI*X(I+1))**2))
      ENDDO
      F3=(X(M)-1.D00)*  (1.D00+(DSIN(2*PI*X(M)))**2)
      F4=(DSIN(3*PI*X(1)))**2
      F=0.1*(F4+F2+F3)+F1
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE FUNCT14(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F (-31.98, 31.98) = 0.998
      PARAMETER (N=25,NN=2)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(2), A(NN,N)
      DATA (A(1,J),J=1,N) /-32.D00,-16.D00,0.D00,16.D00,32.D00,-32.D00,
     & -16.D00,0.D00,16.D00,32.D00,-32.D00,-16.D00,0.D00,16.D00,32.D00,
     & -32.D0,-16.D0,0.D0,16.D0,32.D0,-32.D0,-16.D0,0.D0,16.D0,32.D0/
      DATA (A(2,J),J=1,N) /-32.D00,-32.D00,-32.D00,-32.D00,-32.D00,
     & -16.D00,-16.D00,-16.D00,-16.D00,-16.D00,0.D00,0.D00,0.D00,0.D00,
     & 0.D00,16.D00,16.D00,16.D00,16.D00,16.D00,32.D00,32.D00,
     & 32.D00,32.D00,32.D00/

      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*200.D00
      ENDIF
      ENDDO
      F1=0.D00
      DO J=1,N
```

```fortran
      F2=0.D00
      DO I=1,2
      F2=F2+(X(I)-A(I,J))**6
      ENDDO
      F2=1.D00/(J+F2)
      F1=F1+F2
      ENDDO
      F=1.D00/(0.002D00+F1)
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE FUNCT15(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F(.19, .19, .12, .14) = 0.3075
      PARAMETER (N=11)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*), A(N),B(N)
      DATA (A(I),I=1,N) /.1957D00,.1947D00,.1735D00,.16D00,.0844D00,
     & .0627D00,.0456D00,.0342D00,.0323D00,.0235D00,.0246D00/
      DATA (B(I),I=1,N) /0.25D00,0.5D00,1.D00,2.D00,4.D00,6.D00,8.D00,
     & 10.D00,12.D00,14.D00,16.D00/
      DO I=1,N
      B(I)=1.D00/B(I)
      ENDDO
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.5.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*10.D00
      ENDIF
      ENDDO
      DO I=1,N
      F1=X(1)*(B(I)**2+B(I)*X(2))
      F2=B(I)**2+B(I)*X(3)+X(4)
      F=F+(A(I)-F1/F2)**2
      ENDDO
      F=F*1000
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE LINPROG1(M,F,X)
C     LINEAR PROGRAMMING : MINIMIZATION PROBLEM
C     IN THIS PROBLEM : M = NO. OF DECISION VARIABLES = 2
C     MIN F (2.390, 2.033) = -19.7253 APPROX
C     MIN F = OVER J=1, M : DO SUM(A(1,J)*X(J))  SUBJECT TO CONSTRAINTS
C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=2
C     . . .       . . .       . . .       . . .
C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=N
C     ALL X(I) => 0
      PARAMETER (N=3) ! N IS THE NO. OF CONSTRAINTS + 1
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*),A(20,10),C(20),FF(20)
      DATA (A(1,J),J=1,2),C(1)/4.D0,5.D0,0.0D0/!COEFF OF OBJ FUNCTION
      DATA (A(2,J),J=1,2),C(2)/10.D0,3.D0,30D0/!COEFF OF 1ST CONSTRAINT
      DATA (A(3,J),J=1,2),C(3)/6.D0,20.D0,55.D0/!COEFF OF 2ND CONSTRAINT
C     ------------------------------------------------------------------
C     USING ONLY NON-NEGATIVE VALUES OF X(I)
      DO I=1,M
      X(I)=DABS(X(I))
      ENDDO
C     EVALUATION OF OBJ FUNCTION AND CONSTRAINTS
      DO I=1,N
      FF(I)=0.D00
      DO J=1,M
      FF(I)=FF(I)+A(I,J)*X(J)
      ENDDO
      ENDDO
      F=-FF(1) ! CHANGE OF SIGN FOR MINIMIZATION
C     CHECK FOR SATISFYING OR VIOLATING THE CONSTRAINTS
      DO I=2,N
      FF(I)=FF(I)-C(I) ! SLACK
C     PENALTY FOR CROSSING LIMITS
      IF(FF(I).GT.0) F=F+(10+FF(I))**2
      ENDDO
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE LINPROG2(M,F,X)
C     LINEAR PROGRAMMING : MINIMIZATION PROBLEM
C     IN THIS PROBLEM : M = NO. OF DECISION VARIABLES = 3
C     MIN F (250, 625, 0) = -3250
C     MIN F = OVER J=1, M : DO SUM(A(1,J)*X(J))   SUBJECT TO CONSTRAINTS
C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=2
C     . . .         . . .          . . .        . . .
C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=N
C     ALL X(I) => 0
      PARAMETER (N=4) ! N IS THE NO. OF CONSTRAINTS + 1
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*),A(20,10),C(20),FF(20)
      DATA (A(1,J),J=1,3),C(1)/30.D0,40.D0,20.D0,0.0D0/! COEFF OF OBJ FUNCTION
      DATA (A(2,J),J=1,3),C(2)/10.D0,12.D0,7.D0,10000.0D0/!COEFF OF 1ST CONSTRAINT
      DATA (A(3,J),J=1,3),C(3)/7.D0,10.D0,8.D0,8000.D0/! COEFF OF 2ND CONSTRAINT
      DATA (A(4,J),J=1,3),C(4)/1.D0,1.D0,1.D0,1000.D0/! COEFF OF 3RD CONSTRAINT
C     -----------------------------------------------------------------
C     USING ONLY NON-NEGATIVE VALUES OF X(I)
      DO I=1,M
      X(I)=DABS(X(I))
      ENDDO
C     EVALUATION OF OBJ FUNCTION AND CONSTRAINTS
      DO I=1,N
      FF(I)=0.D00
      DO J=1,M
      FF(I)=FF(I)+A(I,J)*X(J)
      ENDDO
      ENDDO
      F=-FF(1) ! CHANGE OF SIGN FOR MINIMIZATION
C     CHECK FOR SATISFYING OR VIOLATING THE CONSTRAINTS
      DO I=2,N
      FF(I)=FF(I)-C(I) ! SLACK
C     PENALTY FOR CROSSING LIMITS
      IF(FF(I).GT.0.D00) F=F+(100.D00+FF(I))**2
      ENDDO
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE HOUGEN(A,M,F)
      PARAMETER(N=13,K=3)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(N,K),RATE(N),A(*)
C     ---------------------------------------------------------------
C     HOUGEN FUNCTION (HOUGEN-WATSON MODEL FOR REACTION KINATICS)
C     NO. OF PARAMETERS (A) TO ESTIMATE = 5 = M

C     BEST RESULTS ARE:
C     A(1)=1.253031; A(2)=1.190943; A(3)=0.062798; A(4)=0.040063
C     A(5)=0.112453  ARE BEST ESTIMATES OBTAINED BY ROSENBROCK &
C     QUASI-NEWTON METHOD WITH SUM OF SQUARES OF DEVIATION =0.298900994
C     AND R=0.99945.

C     THE NEXT BEST RESULTS GIVEN BY HOOKE-JEEVES & QUASI-NEWTON
C     A(1)=2.475221;A(2)=0.599177; A(3)=0.124172; A(4)=0.083517
C     A(5)=0.217886; SUM OF SQUARES OF DEVIATION = 0.318593458
C     R=0.99941
C     MOST OF THE OTHER METHODS DO NOT PERFORM WELL
```

```fortran
C     -----------------------------------------------------------------
      DATA X(1,1),X(1,2),X(1,3),RATE(1) /470,300,10,8.55/
      DATA X(2,1),X(2,2),X(2,3),RATE(2) /285,80,10,3.79/
      DATA X(3,1),X(3,2),X(3,3),RATE(3) /470,300,120,4.82/
      DATA X(4,1),X(4,2),X(4,3),RATE(4) /470,80,120,0.02/
      DATA X(5,1),X(5,2),X(5,3),RATE(5) /470,80,10,2.75/
      DATA X(6,1),X(6,2),X(6,3),RATE(6) /100,190,10,14.39/
      DATA X(7,1),X(7,2),X(7,3),RATE(7) /100,80,65,2.54/
      DATA X(8,1),X(8,2),X(8,3),RATE(8) /470,190,65,4.35/
      DATA X(9,1),X(9,2),X(9,3),RATE(9) /100,300,54,13/
      DATA X(10,1),X(10,2),X(10,3),RATE(10) /100,300,120,8.5/
      DATA X(11,1),X(11,2),X(11,3),RATE(11) /100,80,120,0.05/
      DATA X(12,1),X(12,2),X(12,3),RATE(12) /285,300,10,11.32/
      DATA X(13,1),X(13,2),X(13,3),RATE(13) /285,190,120,3.13/
C     WRITE(*,1)((X(I,J),J=1,K),RATE(I),I=1,N)
C   1 FORMAT(4F8.2)
      DO J=1,M
      IF(DABS(A(J)).GT.5.D00) THEN
      CALL RANDOM(RAND)
      A(J)=(RAND-0.5D00)*10.D00
      ENDIF
      ENDDO
      F=0.D00
      DO I=1,N
      D=1.D00
         DO J=1,K
         D=D+A(J+1)*X(I,J)
         ENDDO
      FX=(A(1)*X(I,2)-X(I,3)/A(M))/D
C     FX=(A(1)*X(I,2)-X(I,3)/A(5))/(1.D00+A(2)*X(I,1)+A(3)*X(I,2)+
C     A(4)*X(I,3))
      F=F+(RATE(I)-FX)**2
      ENDDO
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE GIUNTA(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     GIUNTA FUNCTION
C     X(I) = -1 TO 1;  M=2
        DO I=1,M
        IF(DABS(X(I)).GT.1.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*2.D00
        ENDIF
        ENDDO
      C=16.D00/15.D00
      F=DSIN(C*X(1)-1.D0)+DSIN(C*X(1)-1.D0)**2+DSIN(4*(C*X(1)-1.D0))/50+
     &DSIN(C*X(2)-1.D0)+DSIN(C*X(2)-1.D0)**2+DSIN(4*(C*X(2)-1.D0))/50+.6
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE EGGHOLD(M,X,F)
C     EGG HOLDER FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      DO I=1,M
       IF(DABS(X(I)).GT.512.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*1024.D00
       ENDIF
       ENDDO
       F=0.D00
      DO I=1,M-1
      F1=-(X(I+1)+47.D00)
      F2=DSIN(  DSQRT(  DABS(  X(I+1)+X(I)/2+47.D00  )  )  )
      F3=DSIN(  DSQRT(  DABS(  X(I)-(X(I+1)+47.D00) ) ))
      F4=-X(I)
      F=F+ F1*F2+F3*F4
      ENDDO
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE TRID(M,X,F)
C     TRID FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F1=0.D00
      F2=0.D00
      DO I=1, M
      F1=F1+(X(I)-1.D00)**2
      ENDDO
      DO I=2, M
      F2=F2+X(I)*X(I-1)
      ENDDO
      F=F1-F2
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE GRIEWANK(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     GRIEWANK FUNCTION
      F1=0.D00
      F2=1.0D00
      DO I=1,M
      F1=F1+X(I)**2
      FI=DFLOAT(I)
      F2=F2*DCOS(X(I)/DSQRT(FI))
      ENDDO
      F=F1/4000.D00-F2+1.0D00
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE WEIERSTRASS(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
C     WEIERSTRASS FUNCTION ------------------------------------
      DATA AX,BX,KMAX/0.5D00,3.D00,20/
      f1=0.D00
      f2=0.D00

        DO I=1,M
        IF(DABS(X(I)).GT.0.5D00) THEN
        CALL RANDOM(RAND)
        X(I)=RAND-0.5D00
        ENDIF
        ENDDO

      DO I=1,M
      S=0.D00
          DO KK=1,KMAX+1
          K=KK-1
          S=S+(AX**K*DCOS(2*PI*BX**K*(X(I)+0.5D00)))
          ENDDO
      F1=F1+S
      ENDDO

      DO KK=1,KMAX+1
      K=KK-1
      F2=F2+(AX**K*DCOS(2*PI*BX**K*0.5D00))
      ENDDO
      F=F1-M*F2
      RETURN
      END
C     ------------------------------------------------------------
      SUBROUTINE LEVY3(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     LEVY # 3 (LEVY ET AL. 1981) ------------------------------------
         DO I=1,M
         IF(DABS(X(I)).GT.10.D00) THEN
         CALL RANDOM(RAND)
         X(I)=(RAND-0.5D00)*20
         ENDIF
         ENDDO
          F1=0.0D+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
         F=F1*F2
         RETURN
         END
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
            ENDIF
          ENDDO
         F=0.D00
         DO I=2, M
         F=F + I*(2*X(I)**2-X(I-1))**2
         ENDDO
         F=F+(X(1)-1.D00)**2
         RETURN
         END
C        -----------------------------------------------------------------
         SUBROUTINE SHEKEL(M,X,F)
C        SHEKEL FUNCTION FOR TEST OF GLOBAL OPTIMIZATION METHODS
         PARAMETER(NROW=10,NCOL=4, NR=9)! NR MAY BE 2 TO 10
         IMPLICIT DOUBLE PRECISION (A-H, O-Z)
         DIMENSION A(NROW,NCOL),C(NROW),X(*)
         DATA ((A(I,J),J=1,NCOL),I=1,NROW)/4.,4.,4.,4.,1.,1.,1.,1.,8.,8.,
        & 8.,8.,6.,6.,6.,6.,3.,7.,3.,7.,2.,9.,2.,9.,5.,5.,3.,3.,8.,1.,8.,
        & 1.,6.,2.,6.,2.,7.,3.6D00,7.,3.6D00/
         DATA (C(I),I=1,NROW)/0.1D00,0.2D00,0.2D00,0.4D00,0.4D00,0.6D00,
        & 0.3D00,0.7D00,0.5D00,0.5D00/
         F=0.D00
         DO I=1,NR
           S=0.D00
           DO J=1,M
           S=S+(X(J)-A(I,J))**2
           ENDDO
         F=F-1.D00/(S+C(I))
         ENDDO
         RETURN
         END
C        ----------------------------------------------------------------
         SUBROUTINE PAVIANI(M,X,F)
C        PAVIANI FUNCTION : MIN F(9.3502,...,9.3502)=45.77847 APPROX
C        IN THE DOMAIN 2<= X(I)  <= 10 FOR I=1,2,...,10.
         IMPLICIT DOUBLE PRECISION (A-H, O-Z)
         DIMENSION X(*)
         DO I=1,M
         IF(X(I).LE.2.D00.OR.X(I).GE.10.D00) THEN
         CALL RANDOM(RAND)
         X(I)=RAND*8+2.D00
         ENDIF
         ENDDO
         F1=0.D00
```

```
      F2=1.D00
      DO I=1,M
      F1=F1+ DLOG(X(I)-2.D00)**2+DLOG(10.D00-X(I))**2
      F2=F2*X(I)
      ENDDO
      F=F1-F2**0.2
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE BRANIN1(M,X,F)
C     BRANIN FUNCTION #1  MIN F (1, 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.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F=(1.D00-2*X(2)+DSIN(4*PI*X(2))/2.D00-X(1))**2+(X(2)-
     & DSIN(2*PI*X(1))/2.D00)**2
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE BRANIN2(M,X,F)
C     BRANIN FUNCTION #2 MIN F (3.1416, 2.25)= 0.397887 APPROX
      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
      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
      END
C     -----------------------------------------------------------------
      SUBROUTINE MCCORMICK(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     MCCORMICK FUNCTION : -1.5<= X(1)<=4; -3<=X(2)<=4 ; M=2
C     MIN F (-0.54719755, -1.54719755) = -1.913223 APPROX
        IF(X(1).LT. -1.5D00 .OR. X(1) .GT. 4.D00) THEN
        CALL RANDOM(RAND)
        X(1)=RAND*5.5D00-1.5D00
        ENDIF
        IF(X(2).LT. -3.D00 .OR. X(2) .GT. 4.D00) THEN
        CALL RANDOM(RAND)
        X(2)=RAND*7.D00-3.D00
        ENDIF
      F=DSIN(X(1)+X(2))+(X(1)-X(2))**2-1.5*X(1)+2.5*X(2)+1.D00
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE FENTONEASON(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     FENTON & EASON FUNCTION  FMIN(1.74345, -2.029695) = 1.744152
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=1.2D00+0.1*X(1)**2 +(0.1D00+0.1*X(2)**2)/X(1)**2+
     & (.1*X(1)**2*X(2)**2+10.D00)/((X(1)*X(2))**4)
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE WOOD(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
C     WOOD FUNCTION:FMIN(0.443546,-0.194607,1.466077,2.15115)=1.09485393
      DO I=1,M
      IF(DABS(X(I)).GT.5.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*10
      ENDIF
      ENDDO
      F1=100*(X(2)+X(1)**2)**2 + (1.D0-X(1))**2 +90*(X(4)-X(3)**2)**2
      F2=(1.D0-X(3))**2 +10.1*((X(2)-1.D0)**2+(X(4)-1.D0)**2)
      F3=19.8*(X(2)-1.D0)*(X(4)-1.D0)
      F=F1+F2+F3
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE GLANKWAHMDEE(M,X,F)
      PARAMETER (N=5)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*),A(N,N), B(N)
C     ------------------ GLANKWAHMDEE FUNCTION ------------------------
C     Glankwahmdee A, Liebman JS and Hogg GL (1979) "Unconstrained
C     Discrete Nonlinear Programming. Engineering Optimization 4: 95-107
C     PARSOPOULOS, KE and VRAHATIS, MN (2002) Recent Approaches to
C     Global Optimization Problems through Particle Swarm Optimization,
C     Natural Computing 1: 235-306, 2002.  REPORT THE BEST OBTAINED
C     FMIN (0,12,23,17,6)= -737 OR  MIN F(0, 11,22,16,6)= -737
C     WE GET FMIN(-.232, 11.489, 22.273, 16.540, 6.115) = -739.822991
C     ------------------------------------------------------------------
      Data ((a(I,j),j=1,n),I=1,n) /35,-20,-10,32,-10,-20, 40,-6,-31,32,
     & -10,-6,11,-6,-10,32,-31,-6,38,-20,-10,32,-10,-20,31/
      DATA (b(j),j=1,n) /-15,-27,-36,-18,-12/
C     -----------------------------------------------------------------
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=0.d0
      Do J=1,m
      F=f+b(j)*x(j)
      Enddo
      Do I=1,m
      C=0.d0
      Do j=1,m
      C=c+x(j)*a(j,I)
      Enddo
      F=f+c*x(i)
      Enddo
      RETURN
      END
C     -----------------------------------------------------------------
      SUBROUTINE FLETCHER(M,X,F)
C     FLETCHER-POWELL FUNCTION, M <= 10, ELSE IT IS VERY MUCH SLOW
C     SOLUTION: MIN F = 0 FOR X=(C1, C2, C3,...,CM)
      PARAMETER(N=10) ! FOR DIMENSION OF DIFFERENT MATRICES AND VECTORS
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*),A(N,N),B(N,N),AA(N),BB(N),AL(N),C(N),C1(N)
      PI=4*DATAN(1.D00)
C     GENERATE A(I,J) AND B(I,J) BETWEEN (-100, 100) RANDOMLY.
C     C(I) = BETWEEN (-PI, PI) IS EITHER GIVEN OR RANDOMLY GENERATED.
C     DATA (C(I),I=1,10)/1,2,3,-3,-2,-1,0,1,2,3/ ! BETWEEN -PI AND PI
      DATA (C1(I),I=1,N)/-3,-3.02,-3.01,1,1.03,1.02,1.03,-.08,.001,3/
C     DATA (C1(I),I=1,N)/0,0,0,0,0,0,0,0,0,0/ ! another example c1 = 0
      NC=0  ! DEFINE NC HERE 0 OR 1 OR 2
C     IF NC=0, C1 FROM DATA IS USED (THAT IS FIXED C);
C     IF NC=1, C IS MADE FROM C1 BY ADDING RANDOM PART - THAT IS C=C1+r
C     IF NC=2 THEN C IS PURELY RANDOM THAT IS C= 0 + r
C     IN ANY CASE C LIES BETWEEN -PI AND PI.
C     ----------------------------------------------------------------
C     FIND THE MAX MAGNITUDE ELEMENT IN C1 VECTOR (UPTO M ELEMENTS)
```

```
       CMAX=DABS(C1(1))
       DO J=2,M
       IF(DABS(C1(J)).GT.CMAX) CMAX=DABS(C1(J))
       ENDDO
       RANGE=PI-CMAX
C     -----------------------------------------------------------------
      DO J=1,M
      DO I=1,M
      CALL RANDOM(RAND)
      A(I,J)=(RAND-0.5D00)*200.D00
      CALL RANDOM(RAND)
      B(I,J)=(RAND-0.5D00)*200.D00
      ENDDO
      IF(NC.EQ.0) AL(J)=C1(J) ! FIXED OR NON-STOCHASTIC C
      IF(NC.EQ.1) THEN
      CALL RANDOM(RAND)
      AL(J)=C1(J)+(RAND-0.5D0)*2*RANGE ! A PART FIXED, OTHER STOCHASTIC
      ENDIF
      IF(NC.EQ.2) THEN
      CALL RANDOM(RAND)
      AL(J)=(RAND-0.5D00)*2*PI ! PURELY STOCHASTIC
      ENDIF
      ENDDO
      DO I=1,M
      AA(I)=0.D00
      DO J=1,M
      AA(I)=AA(I)+A(I,J)*DSIN(AL(J))+B(I,J)*DCOS(AL(J))
      ENDDO
      ENDDO
C     -----------------------------------------------------------------
      DO I=1,M
      IF(DABS(X(I)).GT.PI) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*2*PI
      ENDIF
      ENDDO
      DO I=1,M
      BB(I)=0.D00
      DO J=1,M
      BB(I)=BB(I)+A(I,J)*DSIN(X(J))+B(I,J)*DCOS(X(J))
      ENDDO
      ENDDO
      F=0.D00
      DO I=1,M
      F=F+(AA(I)-BB(I))**2
      ENDDO
      RETURN
      END
C-----------------------------------------------------------------------
      SUBROUTINE POWELL(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     POWELL FUNCTION  ; -4<= X(I) <=5; M=A MULTIPLE OF 4;
C     MIN F   =   0.0
        DO I=1,M
        IF(X(I).LT.-4.D00 .OR. X(I).GT.5.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*9+.5D00
        ENDIF
        ENDDO
      M4=M/4
      F=0.D00
      do I=1,m4
      j=4*i
      f=f+(x(j-3)+10*x(j-2))**2+5*(x(j-1)-x(j))**2+(x(j-2)-x(j-1))**4 +
     & 10*(x(j-3)-x(j))**4
      enddo
      RETURN
      END
C-----------------------------------------------------------------------
      SUBROUTINE HARTMANN(M,X,F)
      PARAMETER (N=4)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*),P(N,3),A(N,3),C(N)
C     HARTMANN FUNCTION
C     MIN F  =  -3.86278 APPROX  : 0 < X < 1.
      DATA ((P(I,J),J=1,3),I=1,4) /0.6890,0.1170,0.2673,0.4699,0.4387,
     &  0.7470,0.1091,0.8732,0.5547,0.0381,0.5743,0.8828/
      DATA ((A(I,J),J=1,3),I=1,4) /3.0,10.0,30.0,0.1,10.0,35.0,3.0,
     &  10.0,30.0,0.1,10.0,35.0/
      DATA (C(J),J=1,4) /1.0,1.2,3.0,3.2/
        DO I=1,M
        IF(X(I).LE.0.D00 .OR. X(I).GE.1.D00) THEN
        CALL RANDOM(RAND)
        X(I)=RAND
        ENDIF
        ENDDO
      F=0.D00
      DO I=1,N
      S=0.D00
      DO J=1,M
      S=S+A(I,J)*(X(J)-P(I,J))**2
      ENDDO
      F=F+C(I)*DEXP(-S)
      ENDDO
      F=-F
      RETURN
      END
C-----------------------------------------------------------------------
      SUBROUTINE COLVILLE(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     COLVILLE FUNCTION  ; -10<= X(I) <=10; M= 4;
C     MINF(1,1,1,1)= 0.0
        DO I=1,M
        IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
        ENDDO
      F=100*(X(1)**2-X(2))**2 + (X(1)-1.D00)**2 +(X(3)-1.D00)**2+
     & 90*(X(3)**2-X(4))**2+10.1*((X(2)-1.D0)**2+(X(4)-1.D00)**2)+
     & 19.8*(X(2)-1.D00)*(X(4)-1.D00)
      RETURN
      END
```