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 ($m \geq 2$) real vector of resources (endowments), $x_i = (x_{i1},\ x_{i2},\ ...,\ x_{im})$. 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) the barter, $\beta(x_{ij}, x_{k\ell} : i \neq k;\ j \neq l)$, is feasible across different persons and different resources only, and (ii) the resources will change hands (the barter materializes) 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 52 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			DE	BA	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	yes	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	25	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 much slower (than the DE as well as the MRPS), since, as empirically observed, attempts to barter by the individuals is successful in less than 1 percent cases, on an average. It may be noted, however, that 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. We have found that when the DE or the RPS optimizes, the terminal population is (achievement-wise) homogenous while in case of the Barter method it is not so (i.e. it has diversity). This property of the Barter method has several implications with respect to the Agent-Based Computational Economics [Tesfatsion, 2002].

Bibliography

- Bauer, J.M.: "Harnessing the Swarm: Communication Policy in an Era of Ubiquitous Networks and Disruptive Technologies", *Communications and Strategies*, 45, 2002.
- Box, M.J.: "A New Method of Constrained Optimization and a Comparison with Other Methods". *Comp. J.* 8, pp. 42-52, 1965.
- Bukin, A. D.: *New Minimization Strategy For Non-Smooth Functions*, Budker Institute of Nuclear Physics preprint BUDKER-INP-1997-79, Novosibirsk 1997.
- Cerny, V.: "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm", *J. Opt. Theory Appl.*, 45, 1, 41-51, 1985.
- Eberhart R.C. and Kennedy J.: "A New Optimizer using Particle Swarm Theory", *Proceedings Sixth Symposium on Micro Machine and Human Science*, pp. 39–43. IEEE Service Center, Piscataway, NJ, 1995.
- Fleischer, M.: "Foundations of Swarm Intelligence: From Principles to Practice", Swarming Network Enabled C4ISR, arXiv:nlin.AO/0502003 v1 2 Feb 2005.
- G.E.P. Box, "Evolutionary operation: A Method for Increasing Industrial Productivity", *Applied Statistics*, 6 , pp. 81-101, 1957.
- Glover F.," Future paths for Integer Programming and Links to Artificial Intelligence", *Computers and Operations Research*, 5:533-549, 1986.
- Hayek, F.A.: *The Road to Serfdom*, Univ. of Chicago Press, Chicago, 1944.
- Holland, J.: *Adaptation in Natural and Artificial Systems,* Univ. of Michigan Press, Ann Arbor, 1975.
- Karush, W. *Minima of Functions of Several Variables with Inequalities as Side Constraints*. M.Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago, Chicago, Illinois, 1939.
- Kirkpatrick, S., Gelatt, C.D. Jr., and Vecchi, M.P.: "Optimization by Simulated Annealing", *Science*, 220, 4598, 671-680, 1983.
- Kuhn, H.W. and Tucker, A.W.: "Nonlinear Programming", in Neymann, J. (ed) *Proceedings of Second Berkeley Symposium on Mathematical Statistics and Probability,* Univ. of California Press, Berkrley, Calif. pp. 481-492, 1951.
- Metropolis, N. The Beginning of the Monte Carlo Method. *Los Alamos Science*, No. 15, Special Issue, pp. 125-130, 1987.
- Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E.: "Equation of State Calculations by Fast Computing Machines", *J. Chem. Phys.*,21, 6, 1087-1092, 1953.
- Mishra, S.K.: "Some Experiments on Fitting of Gielis Curves by Simulated Annealing and Particle Swarm Methods of Global Optimization", *Social Science Research Network* (SSRN): http://ssrn.com/abstract=913667, Working Papers Series, 2006 (a).

- Mishra, S.K.: "Least Squares Fitting of Chacón-Gielis Curves by the Particle Swarm Method of Optimization", *Social Science Research Network* (SSRN), Working Papers Series, http://ssrn.com/abstract=917762 , 2006 (b).
- Mishra, S.K.: "Performance of Repulsive Particle Swarm Method in Global Optimization of Some Important Test Functions: A Fortran Program" , *Social Science Research Network* (SSRN), Working Papers Series, http://ssrn.com/abstract=924339 , 2006 (c).
- Mishra, S.K.: "Some New Test Functions for Global Optimization and Performance of Repulsive Particle Swarm Method", *Social Science Research Network* (SSRN) Working Papers Series, http://ssrn.com/abstract=927134, 2006 (d).
- Mishra, S.K.: "Repulsive Particle Swarm Method on Some Difficult Test Problems of Global Optimization" ,SSRN: http://ssrn.com/abstract=928538 , 2006 (e).
- Mishra, SK.: "Global Optimization by Differential Evolution and Particle Swarm Methods: Evaluation on Some Benchmark Functions" SSRN: http://ssrn.com/abstract=933827 ,2006 (f)
- Nagendra, S.: *Catalogue of Test Problems for Optimization Algorithm Verification*, Technical Report 97-CRD-110, General Electric Company, 1997.
- Nelder, J.A. and Mead, R.: "A Simplex Method for Function Minimization" *Computer Journal*, 7: pp. 308-313, 1964.
- Parsopoulos, K.E. and Vrahatis, M.N., "Recent Approaches to Global Optimization Problems Through Particle Swarm Optimization", *Natural Computing*, 1 (2-3), pp. 235- 306, 2002.
- Prigogine, I. and Strengers, I.: *Order Out of Chaos: Man's New Dialogue with Nature*, Bantam Books, Inc. NY, 1984.
- Silagadge, Z.K.: "Finding Two-Dimensional Peaks", Working Paper, Budkar Insttute of Nuclear Physics, Novosibirsk, Russia, arXive:physics/0402085 V3 11 Mar 2004.
- Simon, H.A.: *Models of Bounded Rationality*, Cambridge Univ. Press, Cambridge, MA, 1982.
- Smith, A.: *The Theory of the Moral Sentiments*, The Adam Smith Institute (2001 e-version), 1759.
- Storn, R. and Price, K: "Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces" : Technical Report, International Computer Science Institute, Berkley, 1995.
- Sumper, D.J.T.: "The Principles of Collective Animal Behaviour", *Phil. Trans. R. Soc. B*. 361, pp. 5-22, 2006.
- Takayama, A.: *Mathematical Economics*, The Dryden Press, Hinsdale, Illinois, 1974.
- Tesfatsion, L.: "Agent-Based Computational Economics: Growing Economies from the Bottom Up", SSRN Working Papers Series, http://ssrn.com/abstract=305080, 2002.
- Törn, A.A and Viitanen, S.: "Topographical Global Optimization using Presampled Points", *J. of Global Optimization*, 5, pp. 267-276, 1994.
- Törn, A.A.: "A search Clustering Approach to Global Optimization" , in Dixon, LCW and Szegö, G.P. (Eds) *Towards Global Optimization – 2*, North Holland, Amsterdam, 1978.
- Tsallis, C. and Stariolo, D.A.: "Generalized Simulated Annealing", *ArXive condmat/9501047 v1 12 Jan,* 1995.
- Valentine, R.H.: *Travel Time Curves in Oblique Structures*, Ph.D. Dissertation, MIT, Mass, 1937.
- Veblen, T.B.: "Why is Economics Not an Evolutionary Science" *The Quarterly Journal of Economics*, 12, 1898.
- Veblen, T.B.: *The Theory of the Leisure Class*, The New American library, NY. (Reprint, 1953), 1899.
- Vesterstrøm, J. and Thomsen, R.: "A comparative Study of Differential Evolution, Particle Swarm Optimization, and Evolutionary Algorithms on Numerical Benchmark Problems", *Congress on Evolutionary Computation, 2004. CEC2004***,** 2, pp. 1980-1987, 2004.
- Whitley, D., Mathias, K., Rana, S. and Dzubera, J.: "Evaluating Evolutionary Algorithms", *Artificial Intelligence*, 85, pp. 245-276, 1996.
- Yao, X. and Liu, Y.: "Fast Evolutionary Programming", in Fogel, LJ, Angeline, PJ and Bäck, T (eds) *Proc. 5th Annual Conf. on Evolutionary programming*, pp. 451-460, MIT Press, Mass, 1996.

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

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

```
      FMIN=FMINEXC
      ENDIF

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

```
135:       DIMENSION IR(3),XBAS(500,50)
136: C     ------------------------------------------------------------------
137: C     ------- SELECT THE FUNCTION TO MINIMIZE AND ITS DIMENSION -------
138:       CALL FSELECT(KF,M,FTIT)
139: C     SPECIFY OTHER PARAMETERS ----------------------------------------
140:       WRITE(*,*)'POPULATION SIZE [N] AND NO. OF ITERATIONS [ITER] ?'
141:       WRITE(*,*)'SUGGESTED : N => 100 OR =>10.M; ITER 10000 OR SO'
142:       READ(*,*) N,ITER
143:       WRITE(*,*)'CROSSOVER PROBABILITY [PCROS] AND SCALE [FACT] ?'
144:       WRITE(*,*)'SUGGESTED : PCROS ABOUT 0.9; FACT=.5 OR LARGER BUT <=1'
145:       READ(*,*) PCROS,FACT
146:       WRITE(*,*)'RANDOM NUMBER SEED ?'
147:       WRITE(*,*)'A FOUR-DIGIT POSITIVE ODD INTEGER, SAY, 1171'
148:       READ(*,*) IU
149:
150:       NFCALL=0 ! INITIALIZE COUNTER FOR FUNCTION CALLS
151:       GBEST=1.D30 ! TO BE USED FOR TERMINATION CRITERION
152: C     INITIALIZATION : GENERATE X(N,M) RANDOMLY
153:       DO I=1,N
154:       DO J=1,M
155: C       CALL RANDOM(RAND) ! GENERATES INITION X WITHIN
156: C       X(I,J)=(RAND-.5D00)*2000 ! GENERATES INITION X WITHIN
157: C     RANDOM NUMBERS BETWEEN -RRANGE AND +RRANGE (BOTH EXCLUSIVE)
158:       X(I,J)=XBAS(I,J)! TAKES THESE NUMBERS FROM THE MAIN PROGRAM
159:       ENDDO
160:       ENDDO
161:       WRITE(*,*)'COMPUTING --- PLEASE WAIT '
162:       IPCOUNT=0
163:       DO 100 ITR=1,ITER  ! ITERATION BEGINS
164: C     ------------------------------------------------------------------
165: C     EVALUATE ALL X FOR THE GIVEN FUNCTION
166:       DO I=1,N
167:       DO J=1,M
168:       A(J)=X(I,J)
169:       ENDDO
170:       CALL FUNC(A,M,F)
171: C     STORE FUNCTION VALUES IN FV VECTOR
172:       FV(I)=F
173:       ENDDO
174:       IF(ITR.EQ.1) CALL GINI(FV,N,G0)
175: C     ------------------------------------------------------------------
176: C     FIND THE FITTEST (BEST) INDIVIDUAL AT THIS ITERATION
177:                 FBEST=FV(1)
178:                 KB=1
179:                 DO IB=2,N
180:                      IF(FV(IB).LT.FBEST) THEN
181:                      FBEST=FV(IB)
182:                      KB=IB
183:                      ENDIF
184:                 ENDDO
185: C     BEST FITNESS VALUE = FBEST : INDIVIDUAL X(KB)
186: C     ------------------------------------------------------------------
187: C     GENERATE OFFSPRINGS
188:       DO I=1,N    ! I LOOP BEGINS
189: C     INITIALIZE CHILDREN IDENTICAL TO PARENTS; THEY WILL CHANGE LATER
190:            DO J=1,M
191:            Y(I,J)=X(I,J)
192:            ENDDO
193: C     SELECT RANDOMLY THREE OTHER INDIVIDUALS
194:    20     DO IRI=1,3  ! IRI LOOP BEGINS
195:           IR(IRI)=0
196:
197:           CALL RANDOM(RAND)
198:            IRJ=INT(RAND*N)+1
199: C     CHECK THAT THESE THREE INDIVIDUALS ARE DISTICT AND OTHER THAN I
200:            IF(IRI.EQ.1.AND.IRJ.NE.I) THEN
201:            IR(IRI)=IRJ
```

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

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

```
      U2=RAND ! U1 IS UNIFORMLY DISTRIBUTED [0, 1]
      R=DSQRT(-2.D0*DLOG(U1))
      R=R*DCOS(U2*6.28318530717958647692528676655900D00)
C     R=R*DCOS(U2*6.28318530718D00)
      RETURN
      END
C     ------------------------------------------------------------------
C     RANDOM NUMBER GENERATOR (UNIFORM BETWEEN 0 AND 1 - BOTH EXCLUSIVE)
      SUBROUTINE RANDOM(RAND1)
       DOUBLE PRECISION  RAND1
       COMMON /RNDM/IU,IV
      INTEGER IU,IV
       RAND=REAL(RAND1)
       IV=IU*65539
       IF(IV.LT.0) THEN
       IV=IV+2147483647+1
       ENDIF
       RAND=IV
       IU=IV
       RAND=RAND*0.4656613E-09
       RAND1= (RAND)
       RETURN
       END
C     ======================================================================
C     ======================================================================
      SUBROUTINE RPS(M,BST,FMINIM,H0,H1)
C     PROGRAM TO FIND GLOBAL MINIMUM BY REPULSIVE PARTICLE SWARM METHOD
C     WRITTEN BY SK MISHRA, DEPT. OF ECONOMICS, NEHU, SHILLONG (INDIA)
C     ------------------------------------------------------------------
      PARAMETER (N=100,NN=50,MX=50,NSTEP=11,ITRN=10000,NSIGMA=1,ITOP=3)
C     PARAMETER(N=50,NN=25,MX=100,NSTEP=9,ITRN=10000,NSIGMA=1,ITOP=3)
C     PARAMETER (N=100,NN=15,MX=100,NSTEP=9,ITRN=10000,NSIGMA=1,ITOP=3)
C     IN CERTAIN CASES THE ONE OR THE OTHER SPECIFICATION WORKS BETTER
C     DIFFERENT SPECIFICATIONS OF PARAMETERS MAY SUIT DIFFERENT TYPES
C     OF FUNCTIONS OR DIMENSIONS - ONE HAS TO DO SOME TRIAL AND ERROR
C     ------------------------------------------------------------------
C     N = POPULATION SIZE. IN MOST OF THE CASES N=30 IS OK. ITS VALUE
C     MAY BE INCREASED TO 50 OR 100 TOO. THE PARAMETER NN IS THE SIZE OF
C     RANDOMLY CHOSEN NEIGHBOURS. 15 TO 25 (BUT SUFFICIENTLY LESS THAN
C     N) IS A GOOD CHOICE. MX IS THE MAXIMAL SIZE OF DECISION VARIABLES.
C     IN F(X1, X2,...,XM) M SHOULD BE LESS THAN OR EQUAL TO MX. ITRN IS
C     THE NO. OF ITERATIONS. IT MAY DEPEND ON THE PROBLEM. 200(AT LEAST)
C     TO 500 ITERATIONS MAY BE GOOD ENOUGH. BUT FOR FUNCTIONS LIKE
C     ROSENBROCKOR GRIEWANK OF LARGE SIZE (SAY M=30) IT IS NEEDED THAT
C     ITRN IS LARGE, SAY 5000 OR EVEN 10000.
C     SIGMA INTRODUCES PERTURBATION & HELPS THE SEARCH JUMP OUT OF LOCAL
C     OPTIMA. FOR EXAMPLE : RASTRIGIN FUNCTION OF DMENSION 30 OR LARGER
C     NSTEP DOES LOCAL SEARCH BY TUNNELLING AND WORKS WELL BETWEEN 5 AND
C     15, WHICH IS MUCH ON THE HIGHER SIDE.
C     ITOP <=1 (RING); ITOP=2 (RING AND RANDOM); ITOP=>3 (RANDOM)
C     NSIGMA=0 (NO CHAOTIC PERTURBATION);NSIGMA=1 (CHAOTIC PERTURBATION)
C     NOTE THAT NSIGMA=1 NEED NOT ALWAYS WORK BETTER (OR WORSE)
C     SUBROUTINE FUNC( ) DEFINES OR CALLS THE FUNCTION TO BE OPTIMIZED.
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      COMMON /KFF/KF,NFCALL,FTIT
      COMMON /XBASE/XBAS
      INTEGER IU,IV
      CHARACTER *70 FTIT
      DIMENSION X(N,MX),V(N,MX),A(MX),VI(MX),XBAS(500,50)
      DIMENSION XX(N,MX),F(N),V1(MX),V2(MX),V3(MX),V4(MX),BST(MX)
C     A1 A2 AND A3 ARE CONSTANTS AND W IS THE INERTIA WEIGHT.
C     OCCASIONALLY, TINKERING WITH THESE VALUES, ESPECIALLY A3, MAY BE
C     NEEDED.
      DATA A1,A2,A3,W,SIGMA,EPSI /.5D0,.5D0,5.D-04,.5D00,1.D-03,1.D-08/
C     ------------------------------------------------------------------
C     CALL SUBROUTINE FOR CHOOSING FUNCTION (KF) AND ITS DIMENSION (M)
```

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

```
C-----------------------------------------------------------------------
      IF(ITOP.EQ.2) THEN
C     RING + RANDOM TOPOLOGY ****************************************
C     REQUIRES THAT THE SUBROUTINE NEIGHBOR IS TURNED ALIVE
       BEST=1.0D30
          CALL NEIGHBOR(I,N,I1,I3)
          DO II=1,NN
                IF(II.EQ.1) NF=I1
                 IF(II.EQ.2) NF=I
                  IF(II.EQ.3) NF=I3
                      IF(II.GT.3) THEN
                     CALL RANDOM(RAND)
                      NF=INT(RAND*N)+1
                     ENDIF
                  IF(BEST.GT.F(NF)) THEN
                  BEST=F(NF)
                  NFBEST=NF
                 ENDIF
             ENDDO
       ENDIF
C-----------------------------------------------------------------------
      IF(ITOP.LE.1) THEN
C     RING TOPOLOGY *************************************************
C     REQUIRES THAT THE SUBROUTINE NEIGHBOR IS TURNED ALIVE
        BEST=1.0D30
          CALL NEIGHBOR(I,N,I1,I3)
             DO II=1,3
             IF (II.NE.I) THEN
            IF(II.EQ.1) NF=I1
             IF(II.EQ.3) NF=I3
                  IF(BEST.GT.F(NF)) THEN
                   BEST=F(NF)
                   NFBEST=NF
                  ENDIF
                  ENDIF
           ENDDO
       ENDIF
C-----------------------------------------------------------------------
C     IN THE LIGHT OF HIS OWN AND HIS BEST COLLEAGUES EXPERIENCE, THE
C     INDIVIDUAL I WILL MODIFY HIS MOVE AS PER THE FOLLOWING CRITERION
C     FIRST, ADJUSTMENT BASED ON ONES OWN EXPERIENCE
C     AND OWN BEST EXPERIENCE IN THE PAST (XX(I))
          DO J=1,M
          CALL RANDOM(RAND)
          V1(J)=A1*RAND*(XX(I,J)-X(I,J))
C     THEN BASED ON THE OTHER COLLEAGUES BEST EXPERIENCE WITH WEIGHT W
C     HERE W IS CALLED AN INERTIA WEIGHT 0.01< W < 0.7
C     A2 IS THE CONSTANT NEAR BUT LESS THAN UNITY
          CALL RANDOM(RAND)
          V2(J)=V(I,J)
          IF(F(NFBEST).LT.F(I)) THEN
          V2(J)=A2*W*RAND*(XX(NFBEST,J)-X(I,J))
          ENDIF
C     THEN SOME RANDOMNESS AND A CONSTANT A3 CLOSE TO BUT LESS THAN UNITY
          CALL RANDOM(RAND)
          RND1=RAND
          CALL RANDOM(RAND)
           V3(J)=A3*RAND*W*RND1
C            V3(J)=A3*RAND*W
C     THEN ON PAST VELOCITY WITH INERTIA WEIGHT W
          V4(J)=W*V(I,J)
C     FINALLY A SUM OF THEM
          V(I,J)= V1(J)+V2(J)+V3(J)+V4(J)
          ENDDO
       ENDDO
C     CHANGE X
       DO I=1,N
```

```
      DO J=1,M
      RANDS=0.D00
C     ----------------------------------------------------------------
      IF(NSIGMA.EQ.1) THEN
       CALL RANDOM(RAND) ! FOR CHAOTIC PERTURBATION
       IF(DABS(RAND-.5D00).LT.SIGMA) RANDS=RAND-0.5D00
C     SIGMA CONDITIONED RANDS INTRODUCES CHAOTIC ELEMENT IN TO LOCATION
C     IN SOME CASES THIS PERTURBATION HAS WORKED VERY EFFECTIVELY WITH
C     PARAMETER (N=100,NN=15,MX=100,NSTEP=9,ITRN=100000,NSIGMA=1,ITOP=2)
      ENDIF
C     ----------------------------------------------------------------
      X(I,J)=X(I,J)+V(I,J)*(1.D00+RANDS)
      ENDDO
      ENDDO
       DO I=1,N
         IF(F(I).LT.FMIN) THEN
         FMIN=F(I)
         II=I
         DO J=1,M
         BST(J)=XX(II,J)
         ENDDO
         ENDIF
         ENDDO
      IF(LCOUNT.EQ.100) THEN
      LCOUNT=0
      WRITE(*,*)'OPTIMAL SOLUTION UPTO THIS (FUNCTION CALLS=',NFCALL,')'
      WRITE(*,*)'X = ',(BST(J),J=1,M),' MIN F = ',FMIN
C      WRITE(*,*)'NO. OF FUNCTION CALLS = ',NFCALL
      IF(DABS(FMIN-GGBEST).LT.EPSI) THEN
           WRITE(*,*)'COMPUTATION OVER'
           FMINIM=FMIN
           GOTO 999
           ELSE
           GGBEST=FMIN
           ENDIF
      ENDIF
      LCOUNT=LCOUNT+1
  100 CONTINUE
      WRITE(*,*)'COMPUTATION OVER:',FTIT
      FMINIM=FMIN
  999 CALL GINI(F,N,H1)
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE LSRCH(A,M,VI,NSTEP,FI)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION A(*),B(100),VI(*)
      AMN=1.0D30
      DO J=1,NSTEP
         DO JJ=1,M
         B(JJ)=A(JJ)+(J-(NSTEP/2)-1)*VI(JJ)
         ENDDO
      CALL FUNC(B,M,FI)
        IF(FI.LT.AMN) THEN
        AMN=FI
        DO JJ=1,M
        A(JJ)=B(JJ)
        ENDDO
        ENDIF
      ENDDO
      FI=AMN
      RETURN
      END
C     ----------------------------------------------------------------
C     THIS SUBROUTINE IS NEEDED IF THE NEIGHBOURHOOD HAS RING TOPOLOGY
```

```
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,G0,G1)
      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)
      COMMON /XBASE/XBAS
      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     ---------------------- VERY IMPORTANT ----------------------------
C     FOR THE BARTER METHOD THE DIMENSION M MUST EXCEED UNITY : OR M =>2
C     ------------------------------------------------------------------
      DIMENSION X(500,50),FV(500),A(50),B(50),BEST(50),XBAS(500,50)
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     ------------------------------------------------------------------
      NBART=0 ! NO. OF TIMES  THE BARTER WAS SUCCESSFUL
      NTRY=0  ! NO. OF TIMES  ATTEMPT WAS MADE TO BARTER
      N=M*10
      IF(N.LT.100) N=100
      ITER=10000
      WRITE(*,*)'RANDOM NUMBER SEED ?'
C      READ(*,*) IU
      IU=7113 ! COULD BE ANY OTHER 4 OR 5 DIGIT ODD INTEGER
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
C      CALL RANDOM(RAND)    ! GENERATE INITIAL X WITHIN
C      X(I,J)=(RAND-.5D00)*2000  ! GENERATE INITIAL X WITHIN
C     RANDOM NUMBERS BETWEEN -RRANGE AND +RRANGE (BOTH EXCLUSIVE)
      X(I,J)=XBAS(I,J) ! : GET X FROM OUTSIDE
      ENDDO
      ENDDO
      WRITE(*,*)'COMPUTING --- PLEASE WAIT '
      IPCOUNT=0
      DO 100 ITR=1,ITER  ! ITERATION BEGINS

```

```
671: C     EVALUATE ALL X FOR THE GIVEN FUNCTION
672:       DO I=1,N
673:       DO J=1,M
674:       A(J)=X(I,J)
675:       ENDDO
676:       CALL FUNC(A,M,FA)
677:       FV(I)=FA
678:       CALL SEARCH(A,M,FI)
679: C     STORE FUNCTION VALUES IN FV VECTOR
680:       IF(FI.LT.FV(I)) THEN
681:       DO J=1,M
682:       X(I,J)=A(J)
683:       ENDDO
684:       FV(I)=FI
685:       ENDIF
686:       ENDDO
687:       IF(ITR.EQ.1) CALL GINI(FV,N,G0)
688: C     ------------------------------------------------------------------
689:       DO I=1,N
690:       NTRY=NTRY+1
691: C     CHOOSE IB TH INDIVIDUAL RANDOMLY
692:       CALL RANDOM(RAND)
693:       IB=INT(RAND*N)+1 !THE RANDOM INDIVIDUAL
694: C     IB=2*M+1
695: C     STORE ITH IN A AND RANDOMLY SELECTED INDIVIDUAL IN B
696:          DO J=1,M
697:          A(J)=X(I,J)
698:          B(J)=X(IB,J) ! OF THE INDIVIDUAL RANDOMLY SELECTED
699:          ENDDO
700: C     CHOSE AN INDEX BETWEEN 1 AND M RANDOMLY
701:           CALL RANDOM(RAND)
702:          JA=INT(RAND*M)+1
703: C     CHOOSE ANOTHER INDEX RANDOMLY : MUST BE DIFFERENT FROM JA
704:    1      CALL RANDOM(RAND)
705:          JB=INT(RAND*M)+1
706:          IF(JA.EQ.JB) GOTO 1
707: C     EXCHANGE A(JA) WITH B(JB)
708:           TEMP1=A(JA)
709:           TEMP2=B(JB)
710:            CALL NORMAL(RN)! OBTAIN STANDARD NORMAL  RANDOM NUMBER
711:           A(JB)=A(JB)+RN*TEMP2
712:           B(JB)=B(JB)-RN*TEMP2
713:           A(JA)=A(JA)-RN*TEMP1
714:           B(JA)=B(JA)+RN*TEMP1
715: C     EVALUATE A AND B VECTORS
716:           CALL FUNC(A,M,FA)
717:           CALL FUNC(B,M,FB)
718: C     CHECK IF FA < FV(I) AND FB < FV(IB)
719:           IF(FA.LT.FV(I) .AND. FB.LT.FV(IB)) THEN
720:           NBART=NBART+1
721:           FV(I)=FA
722:           FV(IB)=FB
723:           DO J=1,M
724:           X(I,J)=A(J)
725:           X(IB,J)=B(J)
726:           ENDDO
727:           ENDIF
728:       ENDDO
729: C     ------------------------------------------------------------------
730: C     FIND THE  BEST
731:       FBEST=1.D30
732:       DO I=1,N
733:       IF(FV(I).LT.FBEST) THEN
734:       FBEST=FV(I)
735:       KB=I
736:       ENDIF
737:       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'
           GOTO 999
           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  -- MAY MAKE IT ZERO  --'
      WRITE(*,*)'RAISE ITER OR DO BOTH OR INCREASE N, POPULATION SIZE'
  999 CALL GINI(FV,N,G1)
      PSUCCESS=NBART/DFLOAT(NTRY)*100.0
      WRITE(*,*) 'NO. OF TIMES AN ATTEPT TO BARTER WAS MADE = ', NTRY
      WRITE(*,*) 'NO. OF TIMES BARTER WAS SUCCESSFUL =' , NBART
      WRITE(*,*) 'PERCENTAGE SUCCESS OF BARTER ATTEMPTS = ', PSUCCESS
      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 GINI(F,N,G)
      PARAMETER (K=1) !K=1 GINI COEFFICENT; K=2 COEFFICIENT OF VARIATION
C     THIS PROGRAM COMPUTES MEASURE OF INEQUALITY
C     IF K =1 GET THE GINI COEFFICIENT. IF K=2 GET COEFF OF VARIATIONE
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION F(*)
      S=0.D0
      DO I=1,N
      S=S+F(I)
      ENDDO
      S=S/N
      H=0.D00
      DO I=1,N-1
```

```
      DO J=I+1,N
      H=H+(DABS(F(I)-F(J)))**K
      ENDDO
      ENDDO
      H=(H/(N**2))**(1.D0/K)! FOR K=1 H IS MEAN DEVIATION;
C                            FOR K=2 H IS STANDARD DEVIATION
      WRITE(*,*)'MEASURES OF DISPERSION AND CENTRAL TENDENCY = ',G,S
      G=DEXP(-H)! G IS THE MEASURE OF EQUALITY (NOT GINI OR CV)
C     G=H/DABS(S) !IF S NOT ZERO, K=1 THEN G=GINI, K=2 G=COEFF VARIATION
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE FSELECT(KF,M,FTIT)
C     THE PROGRAM REQUIRES INPUTS FROM THE USER ON THE FOLLOWING ------
C     (1) FUNCTION CODE (KF), (2) NO. OF VARIABLES IN THE FUNCTION (M);
      CHARACTER *70 TIT(100),FTIT
      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=?'/
```

```
872:       DATA TIT(51)/'KF=51 SHEKEL FUNCTION : 4-VARIABLES M=4'/
873:       DATA TIT(52)/'KF=52 PAVIANI FUNCTION : 10-VARIABLES M=10'/
874:       DATA TIT(53)/'KF=53 BRANIN FUNCTION#1 : 2-VARIABLES M=2'/
875:       DATA TIT(54)/'KF=54 BRANIN FUNCTION#2 : 2-VARIABLES M=2'/
876:       DATA TIT(55)/'KF=55 BOHACHEVSKY FUNCTION#1 : 2-VARIABLES M=2'/
877:       DATA TIT(56)/'KF=56 BOHACHEVSKY FUNCTION#2 : 2-VARIABLES M=2'/
878:       DATA TIT(57)/'KF=57 BOHACHEVSKY FUNCTION#3 : 2-VARIABLES M=2'/
879:       DATA TIT(58)/'KF=58 EASOM FUNCTION : 2-VARIABLES M=2'/
880:       DATA TIT(59)/'KF=59 ROSENBROCK FUNCTION : M-VARIABLES M=?'/
881:       DATA TIT(60)/'KF=60 CROSS-LEGGED TABLE FUNCTION:2-VARIABLES M=2'/
882:       DATA TIT(61)/'KF=61 CROSS FUNCTION : 2-VARIABLES M=2'/
883:       DATA TIT(62)/'KF=62 CROSS-IN-TRAY FUNCTION : 2-VARIABLES M=2'/
884:       DATA TIT(63)/'KF=63 CROWNED CROSS FUNCTION : 2-VARIABLES M=2'/
885:       DATA TIT(64)/'KF=64 TT-HOLDER FUNCTION : 2-VARIABLES M=2'/
886:       DATA TIT(65)/'KF=65 HOLDER-TABLE FUNCTION : 2-VARIABLES M=2'/
887:       DATA TIT(66)/'KF=66 CARROM-TABLE FUNCTION : 2-VARIABLES M=2'/
888:       DATA TIT(67)/'KF=67 PENHOLDER FUNCTION : 2-VARIABLES M=2'/
889:       DATA TIT(68)/'KF=68 BIRD FUNCTION : 2-VARIABLES M=2'/
890:       DATA TIT(69)/'KF=69 CHICHINADZE FUNCTION : 2-VARIABLES M=2'/
891:       DATA TIT(70)/'KF=70 MCCORMICK FUNCTION : 2-VARIABLES M=2'/
892:       DATA TIT(71)/'KF=71 GLANKWAHMDEE FUNCTION : 5-VARIABLES M=5'/
893:       DATA TIT(72)/'KF=72 FLETCHER-POWELL FUNCTION : M-VARIABLES M=?'/
894:       DATA TIT(73)/'KF=73 POWELL FUNCTION: M-VARIABLES M (MULT OF 4)=?'/
895:       DATA TIT(74)/'KF=74 HARTMANN FUNCTION: 3-VARIABLES M=3'/
896:       DATA TIT(75)/'KF=75 COLVILLE FUNCTION: 4-VARIABLES M=4'/
897: C     ------------------------------------------------------------------
898:       DO I=1,75
899:       WRITE(*,*)TIT(I)
900:       ENDDO
901:       WRITE(*,*)'-------------------------------------------------'
902:       WRITE(*,*)'FUNCTION CODE [KF] AND NO. OF VARIABLES [M] ?'
903:       READ(*,*) KF,M
904:       FTIT=TIT(KF) ! STORE THE NAME OF THE CHOSEN FUNCTION IN FTIT
905:       RETURN
906:       END
907: C     ------------------------------------------------------------------
908:       SUBROUTINE FUNC(X,M,F)
909: C     TEST FUNCTIONS FOR GLOBAL OPTIMIZATION PROGRAM
910:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
911:       COMMON /RNDM/IU,IV
912:       COMMON /KFF/KF,NFCALL,FTIT
913:       INTEGER IU,IV
914:       DIMENSION X(*)
915:       CHARACTER *70 FTIT
916:       PI=4.D+00*DATAN(1.D+00)! DEFINING THE VALUE OF PI
917:       NFCALL=NFCALL+1 ! INCREMENT TO NUMBER OF FUNCTION CALLS
918: C     KF IS THE CODE OF THE TEST FUNCTION
919: C     ------------------------------------------------------------------
920:       IF(KF.EQ.1) THEN
921: C     FUNCTION #1 MIN AT -0.18467 APPROX AT (-8.4666, -10) APPROX
922:       F=0.D00
923:       DO I=1,M
924:       IF(DABS(X(I)).GT.10.D00) THEN
925:       CALL RANDOM(RAND)
926:       X(I)=(RAND-0.5D00)*20
927:       ENDIF
928:       ENDDO
929:       F=DABS(DCOS(DSQRT(DABS(X(1)**2+X(2)))))**0.5 +0.01*X(1)+.01*X(2)
930:       RETURN
931:       ENDIF
932: C     ------------------------------------------------------------------
933:       IF(KF.EQ.2) THEN
934: C     FUNCTION #2 MIN = -0.199409 APPROX AT (-9.94112, -10) APPROX
935:       F=0.D00
936:       DO I=1,M
937:       IF(DABS(X(I)).GT.10.D00) THEN
938:       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
```

```
1073:       CE=10.D00
1074:       CF=1.0/(8*PI)
1075:       F1=CA*(X(2)-CB*X(1)**2+CC*X(1)-CD)**2
1076:       F2=CE*(1.D00-CF)*DCOS(X(1))*DCOS(X(2))
1077:       F3=DLOG(X(1)**2+X(2)**2+1.D00)
1078:       F=-1.0/(F1+F2+F3+CE)
1079:       RETURN
1080:       ENDIF
1081: C     ------------------------------------------------------------------
1082:       IF(KF.EQ.10) THEN
1083: C     FREUDENSTEIN ROTH  FUNCTION : MIN  = 0 AT (5, 4)
1084:       F=0.D00
1085:       DO I=1,M
1086:       IF(DABS(X(I)).GT.10.D00) THEN
1087:       CALL RANDOM(RAND)
1088:       X(I)=(RAND-0.5D00)*20
1089:       ENDIF
1090:       ENDDO
1091:       F1=(-13.D00+X(1)+((5.D00-X(2))*X(2)-2)*X(2))**2
1092:       F2=(-29.D00+X(1)+((X(2)+1.D00)*X(2)-14.D00)*X(2))**2
1093:       F=F1+F2
1094:       RETURN
1095:       ENDIF
1096: C     ------------------------------------------------------------------
1097:       IF(KF.EQ.11) THEN
1098: C     ANNS XOR FUNCTION (PARSOPOULOS, KE, PLAGIANAKOS, VP, MAGOULAS, GD
1099: C     AND VRAHATIS, MN "STRETCHING TECHNIQUE FOR OBTAINING GLOBAL
1100: C     MINIMIZERS THROUGH PARTICLE SWARM OPTIMIZATION")
1101: C     MIN=0.9597588 FOR X=(1, -1, 1, -1, -1, 1, 1, -1, 0.421134) APPROX
1102: C     OBTAINED BY DIFFERENTIAL EVOLUTION PROGRAM
1103:       F=0.D00
1104:       DO I=1,M
1105:       IF(DABS(X(I)).GT.1.D00) THEN
1106:       CALL RANDOM(RAND)
1107:       X(I)=(RAND-0.5D00)*2
1108:       ENDIF
1109:       ENDDO
1110:       F11=X(7)/(1.D00+DEXP(-X(1)-X(2)-X(5)))
1111:       F12=X(8)/(1.D00+DEXP(-X(3)-X(4)-X(6)))
1112:       F1=(1.D00+DEXP(-F11-F12-X(9)))**(-2)
1113:       F21=X(7)/(1.D00+DEXP(-X(5)))
1114:       F22=X(8)/(1.D00+DEXP(-X(6)))
1115:       F2=(1.D00+DEXP(-F21-F22-X(9)))**(-2)
1116:       F31=X(7)/(1.D00+DEXP(-X(1)-X(5)))
1117:       F32=X(8)/(1.D00+DEXP(-X(3)-X(6)))
1118:       F3=(1.D00-(1.D00+DEXP(-F31-F32-X(9)))**(-1))**2
1119:       F41=X(7)/(1.D00+DEXP(-X(2)-X(5)))
1120:       F42=X(8)/(1.D00+DEXP(-X(4)-X(6)))
1121:       F4=(1.D00-(1.D00+DEXP(-F41-F42-X(9)))**(-1))**2
1122:       F=F1+F2+F3+F4
1123:       RETURN
1124:       ENDIF
1125: C     ------------------------------------------------------------------
1126:       IF(KF.EQ.12) THEN
1127: C     PERM FUNCTION #1 MIN = 0 AT (1, 2, 3, 4)
1128: C     BETA => 0. CHANGE IF NEEDED. SMALLER BETA RAISES DIFFICULY
1129: C     FOR BETA=0, EVERY PERMUTED SOLUTION IS A GLOBAL MINIMUM
1130:       BETA=50.D00
1131:       F=0.D00
1132:       DO I=1,M
1133:       IF(DABS(X(I)).GT.M) THEN
1134:       CALL RANDOM(RAND)
1135:       X(I)=(RAND-0.5D00)*2*M
1136:       ENDIF
1137:       ENDDO
1138:       DO K=1,M
1139:       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)
```

```
1207:       F1=1.00+F11*F12
1208:       F21=(2*X(1)-3*X(2))**2
1209:       F22=(18.D00-32*X(1)+12*X(1)**2+48*X(2)-36*X(1)*X(2)+27*X(2)**2)
1210:       F2=30.D00+F21*F22
1211:       F= (F1*F2)
1212:       RETURN
1213:       ENDIF
1214: C     ------------------------------------------------------------------
1215:       IF(KF.EQ.16) THEN
1216: C     BUKIN'S 6TH FUNCTION  MIN = 0 FOR (-10, 1)
1217: C     -15. LE. X(1) .LE. -5 AND -3 .LE. X(2) .LE. 3
1218:         IF(X(1).LT. -15.D00 .OR. X(1) .GT. -5.D00) THEN
1219:         CALL RANDOM(RAND)
1220:         X(1)=-(RAND*10+5.D00)
1221:         ENDIF
1222:         IF(DABS(X(2)).GT.3.D00) THEN
1223:         CALL RANDOM(RAND)
1224:         X(2)=(RAND-.5D00)*6
1225:         ENDIF
1226:       F=100.D0*DSQRT(DABS(X(2)-0.01D0*X(1)**2))+ 0.01D0*DABS(X(1)+10.D0)
1227:        RETURN
1228:        ENDIF
1229: C     ------------------------------------------------------------------
1230:       IF(KF.EQ.17) THEN
1231: C     NEW N#8 FUNCTION (MULTIPLE GLOBAL MINIMA)
1232: C     MIN VALUE = -1 AT (AROUND .7 AROUND, 0.785 APPROX)
1233:       F=0.D00
1234:       DO I=1,M
1235:       IF(X(I).LT.0.5D00 .OR. X(I).GT.1.D00) THEN
1236:       CALL RANDOM(RAND)
1237:       X(I)=RAND/2.D00
1238:       ENDIF
1239:       ENDDO
1240:       F=-DEXP(-DABS(DLOG(.001D00+DABS((DSIN(X(1)+X(2))+DSIN(X(1)-X(2))+
1241:      & (DCOS(X(1)+X(2))*DCOS(X(1)-X(2))+.001))**2)+
1242:      & .01D00*(X(2)-X(1))**2)))
1243:        RETURN
1244:        ENDIF
1245: C     ------------------------------------------------------------------
1246:       IF(KF.EQ.18) THEN
1247: C     DEFLECTED CORRUGATED SPRING FUNCTION
1248: C     MIN VALUE = -1 AT (5, 5, ..., 5) FOR ANY K AND ALPHA=5; M VARIABLE
1249:       CALL DCS(M,F,X)
1250:       RETURN
1251:       ENDIF
1252: C     ------------------------------------------------------------------
1253:       IF(KF.EQ.19) THEN
1254: C     FACTORIAL FUNCTION, MIN =0 AT X=(1,2,3,...,M)
1255:       CALL FACTOR1(M,F,X)
1256:       RETURN
1257:       ENDIF
1258: C     ------------------------------------------------------------------
1259:       IF(KF.EQ.20) THEN
1260: C     DECANOMIAL FUNCTION, MIN =0 AT X=(2, -3)
1261:         DO I=1,M
1262:         IF(DABS(X(I)).GT.4.D00) THEN
1263:         CALL RANDOM(RAND)
1264:         X(I)= (RAND-0.5D00)*8
1265:         ENDIF
1266:         ENDDO
1267:       CALL DECANOM(M,F,X)
1268:       RETURN
1269:       ENDIF
1270: C     ------------------------------------------------------------------
1271:       IF(KF.EQ.21) THEN
1272: C     JUDGE'S FUNCTION  F(0.864, 1.23) = 16.0817; M=2
1273:       CALL JUDGE(M,X,F)
```

```
1274:       RETURN
1275:       ENDIF
1276: C     ------------------------------------------------------------------
1277:       IF(KF.EQ.22) THEN
1278: C     DODECAL FUNCTION
1279:       CALL DODECAL(M,F,X)
1280:       RETURN
1281:       ENDIF
1282: C     ------------------------------------------------------------------
1283:       IF(KF.EQ.23) THEN
1284: C     WHEN X(1)*X(2)=X(1)*X(2)  ? M=2
1285:       CALL SEQP(M,F,X)
1286:       RETURN
1287:       ENDIF
1288: C     ------------------------------------------------------------------
1289:       IF(KF.EQ.24) THEN
1290: C     WHEN ARITHMETIC MEAN = GEOMETRIC MEAN ? : M =>1
1291:       CALL AMGM(M,F,X)
1292:       RETURN
1293:       ENDIF
1294: C     ------------------------------------------------------------------
1295:       IF(KF.EQ.25) THEN
1296: C     M =>2
1297:       CALL FUNCT2(M,F,X)
1298:       RETURN
1299:       ENDIF
1300: C     ------------------------------------------------------------------
1301:       IF(KF.EQ.26) THEN
1302: C     M =>2
1303:       CALL FUNCT3(M,F,X)
1304:       RETURN
1305:       ENDIF
1306: C     ------------------------------------------------------------------
1307:       IF(KF.EQ.27) THEN
1308: C     M =>2
1309:       CALL FUNCT4(M,F,X)
1310:       RETURN
1311:       ENDIF
1312: C     ------------------------------------------------------------------
1313:       IF(KF.EQ.28) THEN
1314: C     M =>2
1315:       CALL FUNCT6(M,F,X)
1316:       RETURN
1317:       ENDIF
1318: C     ------------------------------------------------------------------
1319:       IF(KF.EQ.29) THEN
1320: C     M =>2
1321:       CALL FUNCT7(M,F,X)
1322:       RETURN
1323:       ENDIF
1324: C     ------------------------------------------------------------------
1325:       IF(KF.EQ.30) THEN
1326: C     M =>2
1327:       CALL FUNCT12(M,F,X)
1328:       RETURN
1329:       ENDIF
1330: C     ------------------------------------------------------------------
1331:       IF(KF.EQ.31) THEN
1332: C     M =>2
1333:       CALL FUNCT13(M,F,X)
1334:       RETURN
1335:       ENDIF
1336: C     ------------------------------------------------------------------
1337:       IF(KF.EQ.32) THEN
1338: C     M =2
1339:       CALL FUNCT14(M,F,X)
1340:       RETURN
```

```
1341:       ENDIF
1342: C     ------------------------------------------------------------------
1343:       IF(KF.EQ.33) THEN
1344: C     M =4
1345:       CALL FUNCT15(M,F,X)
1346:       RETURN
1347:       ENDIF
1348: C     ------------------------------------------------------------------
1349:       IF(KF.EQ.34) THEN
1350: C     WOOD FUNCTION : F MIN  : M=4
1351:       CALL WOOD(M,X,F)
1352:       RETURN
1353:       ENDIF
1354: C     ------------------------------------------------------------------
1355:       IF(KF.EQ.35) THEN
1356: C     FENTON & EASON FUNCTION :   : M=2
1357:       CALL FENTONEASON(M,X,F)
1358:       RETURN
1359:       ENDIF
1360: C     ------------------------------------------------------------------
1361:       IF(KF.EQ.36) THEN
1362: C     HOUGEN FUNCTION 5 VARIABLES : M =3
1363:       CALL HOUGEN(X,M,F)
1364:       RETURN
1365:       ENDIF
1366: C     ------------------------------------------------------------------
1367:       IF(KF.EQ.37) THEN
1368: C     GIUNTA FUNCTION 2 VARIABLES  :M =2
1369:       CALL GIUNTA(M,X,F)
1370:       RETURN
1371:       ENDIF
1372: C     ------------------------------------------------------------------
1373:       IF(KF.EQ.38) THEN
1374: C     EGGHOLDER FUNCTION M VARIABLES
1375:       CALL EGGHOLD(M,X,F)
1376:       RETURN
1377:       ENDIF
1378: C     ------------------------------------------------------------------
1379:       IF(KF.EQ.39) THEN
1380: C     TRID FUNCTION M VARIABLES
1381:       CALL TRID(M,X,F)
1382:       RETURN
1383:       ENDIF
1384: C     ------------------------------------------------------------------
1385:       IF(KF.EQ.40) THEN
1386: C     GRIEWANK FUNCTION M VARIABLES
1387:       CALL GRIEWANK(M,X,F)
1388:       RETURN
1389:       ENDIF
1390: C     ------------------------------------------------------------------
1391:       IF(KF.EQ.41) THEN
1392: C     WEIERSTRASS FUNCTION M VARIABLES
1393:       CALL WEIERSTRASS(M,X,F)
1394:       RETURN
1395:       ENDIF
1396: C     ------------------------------------------------------------------
1397:       IF(KF.EQ.42) THEN
1398: C     LEVY-3 FUNCTION 2 VARIABLES
1399:       CALL LEVY3(M,X,F)
1400:       RETURN
1401:       ENDIF
1402: C     ------------------------------------------------------------------
1403:       IF(KF.EQ.43) THEN
1404: C     LEVY-5 FUNCTION 2 VARIABLES
1405:       CALL LEVY5(M,X,F)
1406:       RETURN
1407:       ENDIF
```

```
1408: C     ------------------------------------------------------------------
1409:       IF(KF.EQ.44) THEN
1410: C     LEVY-8 FUNCTION 3 VARIABLES
1411:       CALL LEVY8(M,X,F)
1412:       RETURN
1413:       ENDIF
1414: C     ------------------------------------------------------------------
1415:       IF(KF.EQ.45) THEN
1416: C     RASTRIGIN FUNCTION M VARIABLES
1417:       CALL RASTRIGIN(M,X,F)
1418:       RETURN
1419:       ENDIF
1420: C     ------------------------------------------------------------------
1421:       IF(KF.EQ.46) THEN
1422: C     ACKLEY FUNCTION M VARIABLES
1423:       CALL ACKLEY(M,X,F)
1424:       RETURN
1425:       ENDIF
1426: C     ------------------------------------------------------------------
1427:       IF(KF.EQ.47) THEN
1428: C     MICHALEWICZ FUNCTION M VARIABLES
1429:       CALL MICHALEWICZ(M,X,F)
1430:       RETURN
1431:       ENDIF
1432: C     ------------------------------------------------------------------
1433:       IF(KF.EQ.48) THEN
1434: C     SCHWEFEL FUNCTION M VARIABLES
1435:       CALL SCHWEFEL(M,X,F)
1436:       RETURN
1437:       ENDIF
1438: C     ------------------------------------------------------------------
1439:       IF(KF.EQ.49) THEN
1440: C     SHUBERT FUNCTION 2 VARIABLES
1441:       CALL SHUBERT(M,X,F)
1442:       RETURN
1443:       ENDIF
1444: C     ------------------------------------------------------------------
1445:       IF(KF.EQ.50) THEN
1446: C     DIXON AND PRICE FUNCTION M VARIABLES
1447:       CALL DIXPRICE(M,X,F)
1448:       RETURN
1449:       ENDIF
1450: C     ------------------------------------------------------------------
1451:       IF(KF.EQ.51) THEN
1452: C     SHEKEL FUNCTION 4 VARIABLES
1453:       CALL SHEKEL(M,X,F)
1454:       RETURN
1455:       ENDIF
1456: C     ------------------------------------------------------------------
1457:       IF(KF.EQ.52) THEN
1458: C     PAVIANI FUNCTION 10 VARIABLES
1459:       CALL PAVIANI(M,X,F)
1460:       RETURN
1461:       ENDIF
1462: C     ------------------------------------------------------------------
1463:       IF(KF.EQ.53) THEN
1464: C     BRANIN FUNCTION#1 2 VARIABLES
1465:       CALL BRANIN1(M,X,F)
1466:       RETURN
1467:       ENDIF
1468: C     ------------------------------------------------------------------
1469:       IF(KF.EQ.54) THEN
1470: C     BRANIN FUNCTION#2 2 VARIABLES
1471:       CALL BRANIN2(M,X,F)
1472:       RETURN
1473:       ENDIF
1474: 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     ------------------------------------------------------------------
```

```
1542:       IF(KF.EQ.66) THEN
1543: C     CARROM-TABLE FUNCTION : 2 VARIABLES
1544: C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -24.15682 APPROX
1545:       CALL CARROMTABLE(M,X,F)
1546:       RETURN
1547:       ENDIF
1548: C     ------------------------------------------------------------------
1549:       IF(KF.EQ.67) THEN
1550: C     PEN-HOLDER FUNCTION : 2 VARIABLES
1551: C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -0.963535 APPROX
1552:       CALL PENHOLDER(M,X,F)
1553:       RETURN
1554:       ENDIF
1555: C     ------------------------------------------------------------------
1556:       IF(KF.EQ.68) THEN
1557: C     BIRD FUNCTION : 2 VARIABLES
1558: C     MIN F (4.70104, 3.15294) APPROX = -106.764537 APPROX  OR
1559: C     MIN F (-1.58214, -3.13024) APPROX = -106.764537 APPROX
1560:       CALL BIRD(M,X,F)
1561:       RETURN
1562:       ENDIF
1563: C     ------------------------------------------------------------------
1564:       IF(KF.EQ.69) THEN
1565: C     CHICHINADZE FUNCTION :  -30 <=X(I)<= 30;  M=2
1566: C     MIN F (5.901329, 0.5) = -43.3158621
1567:       CALL CHICHINADZE(M,X,F)
1568:       RETURN
1569:       ENDIF
1570: C     ------------------------------------------------------------------
1571:       IF(KF.EQ.70) THEN
1572: C     MCCORMICK FUNCTION : -1.5<= X(1)<=4; -3<=X(2)<=4 ; M=2
1573: C     MIN F (-0.54719755, -1.54719755) = -1.913223 APPROX
1574:       CALL MCCORMICK(M,X,F)
1575:       RETURN
1576:       ENDIF
1577: C     ------------------------------------------------------------------
1578:       IF(KF.EQ.71) THEN
1579: C     GLANKWAHMDEE FUNCTION:
1580:       CALL GLANKWAHMDEE(M,X,F)
1581:       RETURN
1582:       ENDIF
1583: C     ------------------------------------------------------------------
1584:       IF(KF.EQ.72) THEN
1585: C     FLETCHER-POWELL FUNCTION
1586:       CALL FLETCHER(M,X,F)
1587:       RETURN
1588:       ENDIF
1589: C     ------------------------------------------------------------------
1590:       IF(KF.EQ.73) THEN
1591: C     POWELL FUNCTION
1592:       CALL POWELL(M,X,F)
1593:       RETURN
1594:       ENDIF
1595: C     ------------------------------------------------------------------
1596:       IF(KF.EQ.74) THEN
1597: C     HARTMANN FUNCTION
1598:       CALL HARTMANN(M,X,F)
1599:       RETURN
1600:       ENDIF
1601: C     ------------------------------------------------------------------
1602:       IF(KF.EQ.75) THEN
1603: C     COVILLE FUNCTION
1604:       CALL COLVILLE(M,X,F)
1605:       RETURN
1606:       ENDIF
1607: C     ------------------------------------------------------------------
1608: C      IF(KF.EQ.76) THEN
```

```
1609: C      *****  FUNCTION
1610: C       CALL SUBROUTINE(M,X,F)
1611: C       RETURN
1612: C       ENDIF
1613: C      ================================================================
1614:        WRITE(*,*)'FUNCTION NOT DEFINED. PROGRAM ABORTED'
1615:        STOP
1616:        END
1617: C      >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
1618:        SUBROUTINE DCS(M,F,X)
1619: C      FOR DEFLECTED CORRUGATED SPRING FUNCTION
1620:        IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1621:        DIMENSION X(*),C(100)
1622: C      OPTIMAL VALUES OF (ALL) X ARE ALPHA , AND K IS ONLY FOR SCALING
1623: C      OPTIMAL VALUE OF F IS -1. DIFFICULT TO OPTIMIZE FOR LARGER M.
1624:        DATA K,ALPHA/5,5.D00/ ! K AND ALPHA COULD TAKE ON ANY OTHER VALUES
1625:        R2=0.D00
1626:        DO I=1,M
1627:        C(I)=ALPHA
1628:        R2=R2+(X(I)-C(I))**2
1629:        ENDDO
1630:        R=DSQRT(R2)
1631:        F=-DCOS(K*R)+0.1D00*R2
1632:        RETURN
1633:        END
1634: C      ----------------------------------------------------------------------
1635:        SUBROUTINE QUINTIC(M,F,X)
1636: C      QUINTIC FUNCTION: GLOBAL MINIMA,EXTREMELY DIFFICULT TO OPTIMIZE
1637: C      MIN VALUE = 0 AT PERM( -1, -0.402627941, 2)
1638:        IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1639:        DIMENSION X(*)
1640:        F=0.D00
1641:        DO I=1,M
1642:        F=F+DABS(X(I)**5-3*X(I)**4+4*X(I)**3+2*X(I)**2-10*X(I)-4.D00)
1643:        ENDDO
1644:        F=1000*F
1645:        RETURN
1646:        END
1647: C      ----------------------------------------------------------------------
1648:        SUBROUTINE FACTOR1(M,F,X)
1649: C      FACTORIAL FUNCTION; MIN (1, 2, 3, ...., M) = 0
1650: C      FACT = FACTORIAL(M) = 1 X 2 X 3 X 4 X .... X M
1651: C      FIND X(I), I=1,2,...,M SUCH THAT THEIR PRODUCT IS EQUAL TO FACT.
1652: C      LARGER THE VALUE OF M (=>8) OR SO, HARDER IS THE PROBLEM
1653:        IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1654:        DIMENSION X(*)
1655:        F=0.D00
1656:        FACT=1.D00
1657:        P=1.D00
1658:        DO I=1,M
1659:        FACT=FACT*I
1660:        P=P*X(I)
1661:        F=F+DABS(P-FACT)**2
1662:        ENDDO
1663:        RETURN
1664:        END
1665: C      ----------------------------------------------------------------------
1666:        SUBROUTINE DECANOM(M,F,X)
1667: C      DECANOMIAL FUNCTION; MIN (2, -3) = 0
1668:        IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1669:        DIMENSION X(*)
1670:        F1= DABS(X(1)**10-20*X(1)**9+180*X(1)**8-960*X(1)**7+
1671:       & 3360*X(1)**6-8064*X(1)**5+13340*X(1)**4-15360*X(1)**3+
1672:       & 11520*X(1)**2-5120*X(1)+2624.D00)
1673:        F2= DABS(X(2)**4+12*X(2)**3+54*X(2)**2+108*X(2)+81.D00)
1674:        F=0.001D00*(F1+F2)**2
1675:        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/

      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
```

```
1743:       X(I)=DABS(X(I))
1744:       ENDDO
1745: C     SET SUM = SOME POSITIVE NUMBER. THIS MAKES THE FUNCTION UNIMODAL
1746:       SUM= 100.D00 ! TURNED  ALIVE FOR UNIQUE MINIMUM AND SET SUM TO
1747: C     SOME POSITIVE NUMBER. HERE IT IS 100; IT COULD BE ANYTHING ELSE.
1748:       F1=0.D00
1749:       F2=1.D00
1750:       DO I=1,M
1751:       F1=F1+X(I)
1752:       F2=F2*X(I)
1753:       ENDDO
1754:       XSUM=F1
1755:       F1=F1/M ! SUM DIVIDED BY M = ARITHMETIC MEAN
1756:       F2=F2**(1.D00/M) ! MTH ROOT OF THE PRODUCT = GEOMETRIC MEAN
1757:       F=(F1-F2)**2
1758:       IF(SUM.GT.0.D00) F=F+(SUM-XSUM)**2
1759:       RETURN
1760:       END
1761: C     ------------------------------------------------------------------
1762:       SUBROUTINE FUNCT2(M,F,X)
1763: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1764: C     IN FOGEL, L.J., ANGELIN, P.J. AND BACK, T. (ED) PROCEEDINGS OF THE
1765: C     FIFTH ANNUAL CONFERENCE ON EVOLUTIONARY PROGRAMMING, PP. 451-460,
1766: C     MIT PRESS, CAMBRIDGE, MASS.
1767: C     MIN F (0, 0, ..., 0) = 0
1768:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1769:       DIMENSION X(*)
1770:       F=0.D00
1771:       F1=1.D00
1772:       DO I=1,M
1773:       IF(DABS(X(I)).GT.10.D00) THEN
1774:       CALL RANDOM(RAND)
1775:       X(I)=(RAND-.5D00)*20
1776:       ENDIF
1777:       ENDDO
1778:       DO I=1,M
1779:       F=F+DABS(X(I))
1780:       F1=F1*DABS(X(I))
1781:       ENDDO
1782:       F=F+F1
1783:       RETURN
1784:       END
1785: C     ------------------------------------------------------------------
1786:       SUBROUTINE FUNCT3(M,F,X)
1787: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1788: C     MIN F (0, 0, ... , 0) = 0
1789:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1790:       DIMENSION X(*)
1791:       F=0.D00
1792:       F1=0.D00
1793:       DO I=1,M
1794:       IF(DABS(X(I)).GT.100.D00) THEN
1795:       CALL RANDOM(RAND)
1796:       X(I)=(RAND-.5D00)*200
1797:       ENDIF
1798:       ENDDO
1799:       DO I=1,M
1800:       F1=0.D00
1801:       DO J=1,I
1802:       F1=F1+X(J)**2
1803:       ENDDO
1804:       F=F+F1
1805:       ENDDO
1806:       RETURN
1807:       END
1808: C     ------------------------------------------------------------------
1809:       SUBROUTINE FUNCT4(M,F,X)
```

```
1810: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1811: C     MIN F (0, 0, ..., 0) = 0
1812:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1813:       DIMENSION X(*)
1814:       F=0.D00
1815:       DO I=1,M
1816:       IF(X(I).LT.0.D00 .OR. X(I).GE.M) THEN
1817:       CALL RANDOM(RAND)
1818:       X(I)=RAND*2*M
1819:       ENDIF
1820:       ENDDO
1821: C     FIND MAX(X(I))=MAX(ABS(X(I))) NOTE: HERE X(I) CAN BE ONLY POSITIVE
1822:       XMAX=X(1)
1823:       DO I=1,M
1824:       IF(XMAX.LT.X(I)) XMAX=X(I)
1825:       ENDDO
1826:       F=XMAX
1827:       RETURN
1828:       END
1829: C     ------------------------------------------------------------------
1830:       SUBROUTINE FUNCT6(M,F,X)
1831: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1832: C     MIN F (-.5, -.5, ..., -.5) = 0
1833:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1834:       DIMENSION X(*)
1835:       F=0.D00
1836:       DO I=1,M
1837:       IF(DABS(X(I)).GT.100.D00) THEN
1838:       CALL RANDOM(RAND)
1839:       X(I)=(RAND-.5D00)*200
1840:       ENDIF
1841:       ENDDO
1842:       DO I=1,M
1843:        F=F+(X(I)+0.5D00)**2
1844:       ENDDO
1845:       RETURN
1846:       END
1847: C     ------------------------------------------------------------------
1848:       SUBROUTINE FUNCT7(M,F,X)
1849: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1850: C     MIN F(0, 0, ..., 0) = 0
1851:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1852:       COMMON /RNDM/IU,IV
1853:       INTEGER IU,IV
1854:       DIMENSION X(*)
1855:       F=0.D00
1856:         DO I=1,M
1857:         IF(DABS(X(I)).GT.1.28D00) THEN
1858:         CALL RANDOM(RAND)
1859:         X(I)=(RAND-0.5D00)*2.56D00
1860:         ENDIF
1861:         ENDDO
1862:       DO I=1,M
1863:       CALL RANDOM(RAND)
1864:       F=F+(I*X(I)**4)
1865:       ENDDO
1866:        CALL RANDOM(RAND)
1867:        F=F+RAND
1868:       RETURN
1869:       END
1870: C     ------------------------------------------------------------------
1871:       SUBROUTINE FUNCT12(M,F,X)
1872: C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1873:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1874:       DIMENSION X(100),Y(100)
1875:       DATA A,B,C /10.D00,100.D00,4.D00/
1876:       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
      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
```

```
2011:       PARAMETER (N=3) ! N IS THE NO. OF CONSTRAINTS + 1
2012:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2013:       DIMENSION X(*),A(20,10),C(20),FF(20)
2014:       DATA (A(1,J),J=1,2),C(1)/4.D0,5.D0,0.0D0/!COEFF OF OBJ FUNCTION
2015:       DATA (A(2,J),J=1,2),C(2)/10.D0,3.D0,30D0/!COEFF OF 1ST CONSTRAINT
2016:       DATA (A(3,J),J=1,2),C(3)/6.D0,20.D0,55.D0/!COEFF OF 2ND CONSTRAINT
2017: C     ------------------------------------------------------------------
2018: C     USING ONLY NON-NEGATIVE VALUES OF X(I)
2019:       DO I=1,M
2020:       X(I)=DABS(X(I))
2021:       ENDDO
2022: C     EVALUATION OF OBJ FUNCTION AND CONSTRAINTS
2023:       DO I=1,N
2024:       FF(I)=0.D00
2025:       DO J=1,M
2026:       FF(I)=FF(I)+A(I,J)*X(J)
2027:       ENDDO
2028:       ENDDO
2029:       F=-FF(1) ! CHANGE OF SIGN FOR MINIMIZATION
2030: C     CHECK FOR SATISFYING OR VIOLATING THE CONSTRAINTS
2031:       DO I=2,N
2032:       FF(I)=FF(I)-C(I) ! SLACK
2033: C     PENALTY FOR CROSSING LIMITS
2034:       IF(FF(I).GT.0) F=F+(10+FF(I))**2
2035:       ENDDO
2036:       RETURN
2037:       END
2038: C     ------------------------------------------------------------------
2039:       SUBROUTINE LINPROG2(M,F,X)
2040: C     LINEAR PROGRAMMING : MINIMIZATION PROBLEM
2041: C     IN THIS PROBLEM : M = NO. OF DECISION VARIABLES = 3
2042: C     MIN F (250, 625, 0) = -3250
2043: C     MIN F = OVER J=1, M : DO SUM(A(1,J)*X(J))    SUBJECT TO CONSTRAINTS
2044: C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=2
2045: C     . . .          . . .          . . .          . . .
2046: C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=N
2047: C     ALL X(I) => 0
2048:       PARAMETER (N=4) ! N IS THE NO. OF CONSTRAINTS + 1
2049:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2050:       DIMENSION X(*),A(20,10),C(20),FF(20)
2051:       DATA (A(1,J),J=1,3),C(1)/30.D0,40.D0,20.D0,0.0D0/! COEFF OF OBJ FUNCTION
2052:       DATA (A(2,J),J=1,3),C(2)/10.D0,12.D0,7.D0,10000.0D0/!COEFF OF 1ST CONSTRAINT
2053:       DATA (A(3,J),J=1,3),C(3)/7.D0,10.D0,8.D0,8000.D0/! COEFF OF 2ND CONSTRAINT
2054:       DATA (A(4,J),J=1,3),C(4)/1.D0,1.D0,1.D0,1000.D0/! COEFF OF 3RD CONSTRAINT
2055: C     ------------------------------------------------------------------
2056: C     USING ONLY NON-NEGATIVE VALUES OF X(I)
2057:       DO I=1,M
2058:       X(I)=DABS(X(I))
2059:       ENDDO
2060: C     EVALUATION OF OBJ FUNCTION AND CONSTRAINTS
2061:       DO I=1,N
2062:       FF(I)=0.D00
2063:       DO J=1,M
2064:       FF(I)=FF(I)+A(I,J)*X(J)
2065:       ENDDO
2066:       ENDDO
2067:       F=-FF(1) ! CHANGE OF SIGN FOR MINIMIZATION
2068: C     CHECK FOR SATISFYING OR VIOLATING THE CONSTRAINTS
2069:       DO I=2,N
2070:       FF(I)=FF(I)-C(I) ! SLACK
2071: C     PENALTY FOR CROSSING LIMITS
2072:       IF(FF(I).GT.0.D00) F=F+(100.D00+FF(I))**2
2073:       ENDDO
2074:       RETURN
2075:       END
2076: C     ------------------------------------------------------------------
2077:       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
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+
```

```
2145:      &DSIN(C*X(2)-1.D0)+DSIN(C*X(2)-1.D0)**2+DSIN(4*(C*X(2)-1.D0))/50+.6
2146:       RETURN
2147:       END
2148: C     ------------------------------------------------------------------
2149:       SUBROUTINE EGGHOLD(M,X,F)
2150: C     EGG HOLDER FUNCTION
2151:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2152:       DIMENSION X(*)
2153:       DO I=1,M
2154:        IF(DABS(X(I)).GT.512.D00) THEN
2155:         CALL RANDOM(RAND)
2156:         X(I)=(RAND-0.5D00)*1024.D00
2157:        ENDIF
2158:        ENDDO
2159:        F=0.D00
2160:       DO I=1,M-1
2161:       F1=-(X(I+1)+47.D00)
2162:       F2=DSIN(  DSQRT(  DABS(  X(I+1)+X(I)/2+47.D00  )  )  )
2163:       F3=DSIN(  DSQRT(  DABS(  X(I)-(X(I+1)+47.D00) ) ))
2164:       F4=-X(I)
2165:       F=F+ F1*F2+F3*F4
2166:       ENDDO
2167:       RETURN
2168:       END
2169: C     ------------------------------------------------------------------
2170:       SUBROUTINE TRID(M,X,F)
2171: C     TRID FUNCTION
2172:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2173:       DIMENSION X(*)
2174:       F1=0.D00
2175:       F2=0.D00
2176:       DO I=1, M
2177:       F1=F1+(X(I)-1.D00)**2
2178:       ENDDO
2179:       DO I=2, M
2180:       F2=F2+X(I)*X(I-1)
2181:       ENDDO
2182:       F=F1-F2
2183:       RETURN
2184:       END
2185: C     ------------------------------------------------------------------
2186:       SUBROUTINE GRIEWANK(M,X,F)
2187:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2188:       DIMENSION X(*)
2189: C     GRIEWANK FUNCTION
2190:       F1=0.D00
2191:       F2=1.0D00
2192:       DO I=1,M
2193:       F1=F1+X(I)**2
2194:       FI=DFLOAT(I)
2195:       F2=F2*DCOS(X(I)/DSQRT(FI))
2196:       ENDDO
2197:       F=F1/4000.D00-F2+1.0D00
2198:       RETURN
2199:       END
2200: C     ------------------------------------------------------------------
2201:       SUBROUTINE WEIERSTRASS(M,X,F)
2202:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2203:       DIMENSION X(*)
2204:       PI=4*DATAN(1.D00)
2205: C     WEIERSTRASS FUNCTION ------------------------------------
2206:       DATA AX,BX,KMAX/0.5D00,3.D00,20/
2207:       F1=0.D00
2208:       F2=0.D00
2209:
2210:         DO I=1,M
2211:         IF(DABS(X(I)).GT.0.5D00) THEN
```

```
2212:         CALL RANDOM(RAND)
2213:         X(I)=RAND-0.5D00
2214:         ENDIF
2215:         ENDDO
2216:
2217:       DO I=1,M
2218:       S=0.D00
2219:           DO KK=1,KMAX+1
2220:           K=KK-1
2221:           S=S+(AX**K*DCOS(2*PI*BX**K*(X(I)+0.5D00)))
2222:           ENDDO
2223:       F1=F1+S
2224:       ENDDO
2225:
2226:       DO KK=1,KMAX+1
2227:       K=KK-1
2228:       F2=F2+(AX**K*DCOS(2*PI*BX**K*0.5D00))
2229:       ENDDO
2230:       F=F1-M*F2
2231:       RETURN
2232:       END
2233: C     ------------------------------------------------------------------
2234:       SUBROUTINE LEVY3(M,X,F)
2235:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2236:       DIMENSION X(*)
2237: C     LEVY # 3 (LEVY ET AL. 1981) ----------------------------------
2238:          DO I=1,M
2239:          IF(DABS(X(I)).GT.10.D00) THEN
2240:          CALL RANDOM(RAND)
2241:          X(I)=(RAND-0.5D00)*20
2242:          ENDIF
2243:          ENDDO
2244:           F1=0.0D+00
2245:           F2=0.0D+00
2246:           DO I=1,5
2247:           F1=F1+(I*DCOS((I-1)*X(1)+I))
2248:           F2=F2+(I*DCOS((I+1)*X(2)+I))
2249:           ENDDO
2250:          F=F1*F2
2251:          RETURN
2252:          END
2253: C     ------------------------------------------------------------------
2254:       SUBROUTINE LEVY5(M,X,F)
2255:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2256:       DIMENSION X(*)
2257:             F1=0.0D+00
2258:           F2=0.0D+00
2259:         DO I=1,5
2260:        F1=F1+(I*DCOS((I-1)*X(1)+I))
2261:           F2=F2+(I*DCOS((I+1)*X(2)+I))
2262:           ENDDO
2263:           F3=(X(1)+1.42513D+00)**2
2264:          F4=(X(2)+0.80032D+00)**2
2265:          F=(F1*F2) + (F3+F4)
2266:          RETURN
2267:          END
2268: C     ------------------------------------------------------------------
2269:       SUBROUTINE LEVY8(M,X,F)
2270:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2271:       DIMENSION X(*),Y(3)
2272:       PI=4*DATAN(1.D00)
2273: C     LEVY # 8 FUNCTION --------------------------------------------
2274:          DO I=1,3
2275:          Y(I)=1.D+00+(X(I)-1.D+00)/4.D+00
2276:          ENDDO
2277:          F1=DSIN(PI*Y(1))**2
2278:         F3=(Y(3)-1.D+00)**2
```

```
2279:         F2=0.D+00
2280:          DO I=1,2
2281:           F2=F2+((Y(I)-1.D+00)**2)*(1.D+00+10.D+00*(DSIN(PI*Y(I+1)))**2)
2282:          ENDDO
2283:         F=F1+F2+F3
2284:         RETURN
2285:         END
2286: C     ------------------------------------------------------------------
2287:       SUBROUTINE RASTRIGIN(M,X,F)
2288:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2289:       DIMENSION X(*)
2290:       PI=4*DATAN(1.D00)
2291: C     RASTRIGIN'S FUNCTION
2292:       F=0.D00
2293:       DO I=1,M
2294:       F=F+ X(I)**2 -10*DCOS(2*PI*X(I)) + 10.D00
2295:       ENDDO
2296:       RETURN
2297:       END
2298: C     ------------------------------------------------------------------
2299:       SUBROUTINE ACKLEY(M,X,F)
2300: C     ACKLEY FUNCTION
2301:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2302:       DIMENSION X(*)
2303:       PI=4*DATAN(1.D00)
2304:       F=20.D00+DEXP(1.D00)
2305:       DO I=1,M
2306:       IF(X(I).LT. -15.D00 .OR. X(I).GT.30.D00) THEN
2307:       CALL RANDOM(RAND)
2308:       X(I)=(RAND-0.5D00)*90 -15.D00
2309:       ENDIF
2310:       ENDDO
2311:       F1=0.D00
2312:       F2=0.D00
2313:         DO I=1,M
2314:          F1=F1+X(I)**2
2315:          F2=F2+DCOS(2*PI*X(I))
2316:         ENDDO
2317:       F1=-20*DEXP(-0.2D00*DSQRT(F1/M))
2318:       F2=-DEXP(F2/M)
2319:       F=F+F1+F2
2320:       RETURN
2321:       END
2322: C     ------------------------------------------------------------------
2323:       SUBROUTINE MICHALEWICZ(M,X,F)
2324:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2325:       DIMENSION X(*)
2326: C     MICHALEWICZ FUNCTION   [ 0 <= X(I) <= PI ] MP IS A PARAMETER
2327:       MP=10  ! SET IT TO THE DESIRED VALUE
2328:       PI=4*DATAN(1.D00)
2329:       DO I=1,M
2330:       IF(X(I).LT.0.D00 .OR. X(I).GT.PI)THEN
2331:       CALL RANDOM(RAND)
2332:       X(I)=RAND*PI
2333:       ENDIF
2334:       ENDDO
2335:       F=0.D00
2336:       DO I=1,M
2337:       F=F-DSIN(X(I))*(DSIN(I*X(I)**2/PI))**(2*MP)
2338:       ENDDO
2339:       RETURN
2340:       END
2341: C     ------------------------------------------------------------------
2342:       SUBROUTINE SCHWEFEL(M,X,F)
2343: C     SCHWEFEL FUNCTION
2344:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2345:       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
```

```
2413:         ENDDO
2414:       F=F-1.D00/(S+C(I))
2415:       ENDDO
2416:       RETURN
2417:       END
2418: C     ------------------------------------------------------------------
2419:       SUBROUTINE PAVIANI(M,X,F)
2420: C     PAVIANI FUNCTION : MIN F(9.3502,...,9.3502)=45.77847 APPROX
2421: C     IN THE DOMAIN 2<= X(I) <= 10 FOR I=1,2,...,10.
2422:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2423:       DIMENSION X(*)
2424:       DO I=1,M
2425:       IF(X(I).LE.2.D00.OR.X(I).GE.10.D00) THEN
2426:       CALL RANDOM(RAND)
2427:       X(I)=RAND*8+2.D00
2428:       ENDIF
2429:       ENDDO
2430:       F1=0.D00
2431:       F2=1.D00
2432:       DO I=1,M
2433:       F1=F1+ DLOG(X(I)-2.D00)**2+DLOG(10.D00-X(I))**2
2434:       F2=F2*X(I)
2435:       ENDDO
2436:       F=F1-F2**0.2
2437:       RETURN
2438:       END
2439: C     ------------------------------------------------------------------
2440:       SUBROUTINE BRANIN1(M,X,F)
2441: C     BRANIN FUNCTION #1  MIN F (1, 0) = 0
2442:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2443:       DIMENSION X(*)
2444:       PI=4*DATAN(1.D00)
2445:       DO I=1,M
2446:       IF(DABS(X(I)).GT.10.D00) THEN
2447:       CALL RANDOM(RAND)
2448:       X(I)=(RAND-0.5D00)*20
2449:       ENDIF
2450:       ENDDO
2451:       F=(1.D00-2*X(2)+DSIN(4*PI*X(2))/2.D00-X(1))**2+(X(2)-
2452:      & DSIN(2*PI*X(1))/2.D00)**2
2453:       RETURN
2454:       END
2455: C     ------------------------------------------------------------------
2456:       SUBROUTINE BRANIN2(M,X,F)
2457: C     BRANIN FUNCTION #2 MIN F (3.1416, 2.25)= 0.397887 APPROX
2458:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2459:       DIMENSION X(*)
2460:       PI=4*DATAN(1.D00)
2461:       IF(X(1).LT.-5.D00 .OR. X(1).GT.10.D00) THEN
2462:       CALL RANDOM(RAND)
2463:       X(1)=RAND*15-5.D00
2464:       ENDIF
2465:       IF(X(2).LT.0.D00 .OR. X(2).GT.15.D00) THEN
2466:       CALL RANDOM(RAND)
2467:       X(2)=RAND*15
2468:       ENDIF
2469:       F=(X(2)-5.D00*X(1)**2/(4*PI**2)+5*X(1)/PI-6.D00)**2 +
2470:      &    10*(1.D00-1.D00/(8*PI))*DCOS(X(1))+10.D00
2471:       RETURN
2472:       END
2473: C     ------------------------------------------------------------------
2474:       SUBROUTINE BOHACHEVSKY1(M,X,F)
2475: C     BOHACHEVSKY FUNCTION #1 : MIN F (0, 0) = 0
2476:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2477:       DIMENSION X(*)
2478:       PI=4*DATAN(1.D00)
2479:       DO I=1,M
```

```
2480:       IF(DABS(X(I)).GT.100.D00) THEN
2481:       CALL RANDOM(RAND)
2482:       X(I)=(RAND-0.5D00)*200
2483:       ENDIF
2484:       ENDDO
2485:       F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1))-0.4D00*DCOS(4*PI*X(2))
2486:      & +0.7D00
2487:       RETURN
2488:       END
2489: C     ------------------------------------------------------------------
2490:       SUBROUTINE BOHACHEVSKY2(M,X,F)
2491: C     BOHACHEVSKY FUNCTION #2   : MIN F (0, 0) = 0
2492:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2493:       DIMENSION X(*)
2494:       PI=4*DATAN(1.D00)
2495:       DO I=1,M
2496:       IF(DABS(X(I)).GT.100.D00) THEN
2497:       CALL RANDOM(RAND)
2498:       X(I)=(RAND-0.5D00)*200
2499:       ENDIF
2500:       ENDDO
2501:       F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1))*DCOS(4*PI*X(2))+0.3D00
2502:       RETURN
2503:       END
2504: C     ------------------------------------------------------------------
2505:       SUBROUTINE BOHACHEVSKY3(M,X,F)
2506: C     BOHACHEVSKY FUNCTION #3   : MIN F (0, 0) = 0
2507:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2508:       DIMENSION X(*)
2509:       PI=4*DATAN(1.D00)
2510:       DO I=1,M
2511:       IF(DABS(X(I)).GT.100.D00) THEN
2512:       CALL RANDOM(RAND)
2513:       X(I)=(RAND-0.5D00)*200
2514:       ENDIF
2515:       ENDDO
2516:       F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1)+4*PI*X(2))+0.3D00
2517:       RETURN
2518:       END
2519: C     ------------------------------------------------------------------
2520:       SUBROUTINE EASOM(M,X,F)
2521: C     EASOM FUNCTION : 2-VARIABLES, MIN F (PI, PI) = -1.
2522:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2523:       DIMENSION X(*)
2524:       PI=4*DATAN(1.D00)
2525:       DO I=1,M
2526:       IF(DABS(X(I)).GT.100.D00) THEN
2527:       CALL RANDOM(RAND)
2528:       X(I)=(RAND-0.5D00)*200
2529:       ENDIF
2530:       ENDDO
2531:       F=-DCOS(X(1))*DCOS(X(2))*DEXP(-(X(1)-PI)**2 -(X(2)-PI)**2)
2532:       RETURN
2533:       END
2534: C     ------------------------------------------------------------------
2535:       SUBROUTINE ROSENBROCK(M,X,F)
2536: C     ROSENBROCK FUNCTION : M VARIABLE; MIN F (1, 1,...,1)=0
2537:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2538:       DIMENSION X(*)
2539:       DO I=1,M
2540:       IF(X(I).LT.-5.D00 .OR. X(I).GT.10.D00) THEN
2541:       CALL RANDOM(RAND)
2542:       X(I)=RAND*15-5.D00
2543:       ENDIF
2544:       ENDDO
2545:       F=0.D00
2546:       DO I=1,M-1
```

```
2547:       F=F+ (100.D00*(X(I+1)-X(I)**2)**2 + (X(I)-1.D00)**2)
2548:       ENDDO
2549:       RETURN
2550:       END
2551: C     ------------------------------------------------------------------
2552:       SUBROUTINE CROSSLEG(M,X,F)
2553:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2554:       DIMENSION X(*)
2555: C     CROSS-LEGGED TABLE FUNCTION ; -10<= X(I) <=10; M=2
2556: C     MIN F(0 , X ) OR F(X, 0) = -1.
2557:       PI=4*DATAN(1.D00)
2558:         DO I=1,M
2559:         IF( DABS(X(I)).GT.10.D00) THEN
2560:         CALL RANDOM(RAND)
2561:         X(I)=(RAND-0.5D00)*20
2562:         ENDIF
2563:         ENDDO
2564:       F=-(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
2565:      & (X(1)**2+X(2)**2)/PI))))+1.D00)**(-.1)
2566:       RETURN
2567:       END
2568: C     ------------------------------------------------------------------
2569:       SUBROUTINE CROSS(M,X,F)
2570:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2571:       DIMENSION X(*)
2572: C     CROSS FUNCTION ; -10<= X(I) <=10; M=2;
2573: C     MIN F(A, B)=0 APPROX; A, B=1.3494 APPROX OF EITHER SIGN (+ OR -)
2574:       PI=4*DATAN(1.D00)
2575:         DO I=1,M
2576:         IF( DABS(X(I)).GT.10.D00) THEN
2577:         CALL RANDOM(RAND)
2578:         X(I)=(RAND-0.5D00)*20
2579:         ENDIF
2580:         ENDDO
2581:       F=(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
2582:      & (X(1)**2+X(2)**2)/PI))))+1.D00)**(-.1)
2583:       RETURN
2584:       END
2585: C     ------------------------------------------------------------------
2586:       SUBROUTINE CROSSINTRAY(M,X,F)
2587:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2588:       DIMENSION X(*)
2589: C     CROSS IN TRAY FUNCTION ; -10<= X(I) <=10; M=2;
2590: C     MIN F(A, B)=-20626.1218 APPROX; A, B=1.3494 APPROX OF EITHER SIGN
2591:       PI=4*DATAN(1.D00)
2592:         DO I=1,M
2593:         IF( DABS(X(I)).GT.10.D00) THEN
2594:         CALL RANDOM(RAND)
2595:         X(I)=(RAND-0.5D00)*20
2596:         ENDIF
2597:         ENDDO
2598:       F=-(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
2599:      & (X(1)**2+X(2)**2)/PI))))+1.D00)**(.1)
2600:       RETURN
2601:       END
2602: C     ------------------------------------------------------------------
2603:       SUBROUTINE CROWNEDCROSS(M,X,F)
2604:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2605:       DIMENSION X(*)
2606: C     CROWNED CROSS FUNCTION ; -10<= X(I) <=10; M=2; MIN F = 1
2607:       PI=4*DATAN(1.D00)
2608:         DO I=1,M
2609:         IF( DABS(X(I)).GT.10.D00) THEN
2610:         CALL RANDOM(RAND)
2611:         X(I)=(RAND-0.5D00)*20
2612:         ENDIF
2613:         ENDDO
```

```
2614:       F=(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-
2615:      & (DSQRT(X(1)**2+X(2)**2)/PI))))+1.D00)**(.1)
2616:       RETURN
2617:       END
2618: C     ------------------------------------------------------------------
2619:       SUBROUTINE TTHOLDER(M,X,F)
2620:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2621:       DIMENSION X(*)
2622: C     TEST-TUBE HOLDER FUNCTION ; -10<= X(I) <=10; M=2;
2623: C     MIN F([+/-]1.5706, 0)= -10.8723
2624:       PI=4*DATAN(1.D00)
2625:         DO I=1,M
2626:         IF( DABS(X(I)).GT.10.D00) THEN
2627:         CALL RANDOM(RAND)
2628:         X(I)=(RAND-0.5D00)*20
2629:         ENDIF
2630:         ENDDO
2631:       F=-4*DABS(DSIN(X(1))*DCOS(X(2))*DEXP(DABS(DCOS((X(1)**2+X(2)**2)/
2632:      & 200))))
2633:       RETURN
2634:       END
2635: C     ------------------------------------------------------------------
2636:       SUBROUTINE HOLDERTABLE(M,X,F)
2637:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2638:       DIMENSION X(*)
2639: C     HOLDER-TABLE FUNCTION  ; -10<= X(I) <=10; M=2;
2640: C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -26.92034 APPROX
2641:       PI=4*DATAN(1.D00)
2642:         DO I=1,M
2643:         IF( DABS(X(I)).GT.10.D00) THEN
2644:         CALL RANDOM(RAND)
2645:         X(I)=(RAND-0.5D00)*20
2646:         ENDIF
2647:         ENDDO
2648:       F=-DABS(DCOS(X(1))*DCOS(X(2))*DEXP(DABS(1.D00-(DSQRT(X(1)**2+
2649:      & X(2)**2)/PI))))
2650:       RETURN
2651:       END
2652: C     ------------------------------------------------------------------
2653:       SUBROUTINE CARROMTABLE(M,X,F)
2654:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2655:       DIMENSION X(*)
2656: C     CARROM-TABLE FUNCTION  ; -10<= X(I) <=10; M=2;
2657: C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -24.15682 APPROX
2658:       PI=4*DATAN(1.D00)
2659:         DO I=1,M
2660:         IF( DABS(X(I)).GT.10.D00) THEN
2661:         CALL RANDOM(RAND)
2662:         X(I)=(RAND-0.5D00)*20
2663:         ENDIF
2664:         ENDDO
2665:       F=-1.D00/30*(DCOS(X(1))*DCOS(X(2))*DEXP(DABS(1.D00-
2666:      & (DSQRT(X(1)**2 + X(2)**2)/PI))))**2
2667:       RETURN
2668:       END
2669: C     ------------------------------------------------------------------
2670:       SUBROUTINE PENHOLDER(M,X,F)
2671:       IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2672:       DIMENSION X(*)
2673: C     PENHOLDER FUNCTION  ; -11<= X(I) <=11; M=2;
2674: C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -0.963535 APPROX
2675:       PI=4*DATAN(1.D00)
2676:         DO I=1,M
2677:         IF( DABS(X(I)).GT.11.D00) THEN
2678:         CALL RANDOM(RAND)
2679:         X(I)=(RAND-0.5D00)*22
2680:         ENDIF
```

```
2681:         ENDDO
2682:        F=-DEXP(-(DABS(DCOS(X(1))*DCOS(X(2))*DEXP(DABS(1.D0-(DSQRT
2683:       & (X(1)**2+X(2)**2)/PI))))**(-1)))
2684:        RETURN
2685:        END
2686: C     ------------------------------------------------------------------
2687:        SUBROUTINE BIRD(M,X,F)
2688:        IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2689:        DIMENSION X(*)
2690: C      BIRD FUNCTION ; -2PI<= X(I) <=2PI; M=2;
2691: C      MIN F (4.70104, 3.15294) APPROX = -106.764537 APPROX  OR
2692: C      MIN F (-1.58214, -3.13024) APPROX = -106.764537 APPROX
2693:        PI=4*DATAN(1.D00)
2694:         DO I=1,M
2695:         IF( DABS(X(I)).GT.2*PI) THEN
2696:         CALL RANDOM(RAND)
2697:         X(I)=(RAND-0.5D00)*4*PI
2698:         ENDIF
2699:         ENDDO
2700:        F=(DSIN(X(1))*DEXP((1.D00-DCOS(X(2)))**2) +
2701:       & DCOS(X(2))*DEXP((1.D00-DSIN(X(1)))**2))+(X(1)-X(2))**2
2702:        RETURN
2703:        END
2704: C     ------------------------------------------------------------------
2705:        SUBROUTINE CHICHINADZE(M,X,F)
2706:        IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2707:        DIMENSION X(*)
2708: C      CHICHINADZE FUNCTION :  -30 <=X(I)<= 30;  M=2
2709: C      MIN F (5.901329, 0.5) = -43.3158621
2710:         PI=4*DATAN(1.D00)
2711:         DO I=1,M
2712:         IF( DABS(X(I)).GT.30) THEN
2713:         CALL RANDOM(RAND)
2714:         X(I)=(RAND-0.5D00)*60
2715:         ENDIF
2716:         ENDDO
2717:        F=X(1)**2-12*X(1)+11.D00+10*DCOS(PI*X(1)/2)+8*DSIN(5*PI*X(1))-
2718:       & (1.D00/DSQRT(5.D00))*DEXP(-(X(2)-0.5D00)**2/2)
2719:        RETURN
2720:        END
2721: C     ------------------------------------------------------------------
2722:        SUBROUTINE MCCORMICK(M,X,F)
2723:        IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2724:        DIMENSION X(*)
2725: C      MCCORMICK FUNCTION : -1.5<= X(1)<=4; -3<=X(2)<=4 ; M=2
2726: C      MIN F (-0.54719755, -1.54719755) = -1.913223 APPROX
2727:         IF(X(1).LT. -1.5D00 .OR. X(1) .GT. 4.D00) THEN
2728:         CALL RANDOM(RAND)
2729:         X(1)=RAND*5.5D00-1.5D00
2730:         ENDIF
2731:         IF(X(2).LT. -3.D00 .OR. X(2) .GT. 4.D00) THEN
2732:         CALL RANDOM(RAND)
2733:         X(2)=RAND*7.D00-3.D00
2734:         ENDIF
2735:        F=DSIN(X(1)+X(2))+(X(1)-X(2))**2-1.5*X(1)+2.5*X(2)+1.D00
2736:        RETURN
2737:        END
2738: C     ------------------------------------------------------------------
2739:        SUBROUTINE FENTONEASON(M,X,F)
2740:        IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2741:        DIMENSION X(*)
2742: C      FENTON & EASON FUNCTION  FMIN(1.74345, -2.029695) = 1.744152
2743:        DO I=1,M
2744:        IF(DABS(X(I)).GT.100.D00) THEN
2745:        CALL RANDOM(RAND)
2746:        X(I)=(RAND-0.5D00)*200
2747:        ENDIF
```

```
2748:       ENDDO
2749:       F=1.2D00+0.1*X(1)**2 +(0.1D00+0.1*X(2)**2)/X(1)**2+
2750:      & (.1*X(1)**2*X(2)**2+10.D00)/((X(1)*X(2))**4)
2751:       RETURN
2752:       END
2753: C     ------------------------------------------------------------------
2754:       SUBROUTINE WOOD(M,X,F)
2755:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
2756:       COMMON /RNDM/IU,IV
2757:       INTEGER IU,IV
2758:       DIMENSION X(*)
2759: C     WOOD FUNCTION:FMIN(0.443546,-0.194607,1.466077,2.15115)=1.09485393
2760:       DO I=1,M
2761:       IF(DABS(X(I)).GT.5.D00) THEN
2762:       CALL RANDOM(RAND)
2763:       X(I)=(RAND-0.5D00)*10
2764:       ENDIF
2765:       ENDDO
2766:       F1=100*(X(2)+X(1)**2)**2 + (1.D0-X(1))**2 +90*(X(4)-X(3)**2)**2
2767:       F2=(1.D0-X(3))**2 +10.1*((X(2)-1.D0)**2+(X(4)-1.D0)**2)
2768:       F3=19.8*(X(2)-1.D0)*(X(4)-1.D0)
2769:       F=F1+F2+F3
2770:       RETURN
2771:       END
2772: C     ------------------------------------------------------------------
2773:       SUBROUTINE GLANKWAHMDEE(M,X,F)
2774:       PARAMETER (N=5)
2775:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
2776:       COMMON /RNDM/IU,IV
2777:       INTEGER IU,IV
2778:       DIMENSION X(*),A(N,N), B(N)
2779: C     ----------------- GLANKWAHMDEE FUNCTION -------------------------
2780: C     GLANKWAHMDEE A, LIEBMAN JS AND HOGG GL (1979) "UNCONSTRAINED
2781: C     DISCRETE NONLINEAR PROGRAMMING. ENGINEERING OPTIMIZATION 4: 95-107
2782: C     PARSOPOULOS, KE AND VRAHATIS, MN (2002) RECENT APPROACHES TO
2783: C     GLOBAL OPTIMIZATION PROBLEMS THROUGH PARTICLE SWARM OPTIMIZATION,
2784: C     NATURAL COMPUTING 1: 235-306, 2002.  REPORT THE BEST OBTAINED
2785: C     FMIN (0,12,23,17,6)= -737 OR  MIN F(0, 11,22,16,6)= -737
2786: C     WE GET FMIN(-.232, 11.489, 22.273, 16.540, 6.115) = -739.822991
2787: C     ------------------------------------------------------------------
2788:       DATA ((A(I,J),J=1,N),I=1,N) /35,-20,-10,32,-10,-20, 40,-6,-31,32,
2789:      & -10,-6,11,-6,-10,32,-31,-6,38,-20,-10,32,-10,-20,31/
2790:       DATA (B(J),J=1,N) /-15,-27,-36,-18,-12/
2791: C     ------------------------------------------------------------------
2792:       DO I=1,M
2793:       IF(DABS(X(I)).GT.100.D00) THEN
2794:       CALL RANDOM(RAND)
2795:       X(I)=(RAND-0.5D00)*200
2796:       ENDIF
2797:       ENDDO
2798:       F=0.D0
2799:       DO J=1,M
2800:       F=F+B(J)*X(J)
2801:       ENDDO
2802:       DO I=1,M
2803:       C=0.D0
2804:       DO J=1,M
2805:       C=C+X(J)*A(J,I)
2806:       ENDDO
2807:       F=F+C*X(I)
2808:       ENDDO
2809:       RETURN
2810:       END
2811: C     ------------------------------------------------------------------
2812:       SUBROUTINE FLETCHER(M,X,F)
2813: C     FLETCHER-POWELL FUNCTION, M <= 10, ELSE IT IS VERY MUCH SLOW
2814: 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
C     -----------------------------------------------------------------
```

```
2949: C     SUBROUTINE NAME(M,X,F)
2950: C      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2951: C     PI=4*DATN(1.D0) ! IF NEEDED
2952: C      DIMENSION X(*)
2953: C     **** FUNCTION  ;  A<= X(I) <=B; M= MA;
2954: C     MINF( , , , )= SUCH & SUCH
2955: C        DO I=1,M
2956: C        IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
2957: C        CALL RANDOM(RAND)
2958: C        X(I)=(RAND-0.5D00)*APPROPRIATE + APPROPRATE ETC
2959: C        ENDIF
2960: C        ENDDO
2961: C      F= ETC
2962: C     RETURN
2963: C     END
```