



Munich Personal RePEc Archive

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

Mishra, SK

21 October 2006

Online at <https://mpra.ub.uni-muenchen.de/543/>

MPRA Paper No. 543, posted 21 Oct 2006 UTC

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

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

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

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

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

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

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).

Seen as such, the barter method is inferior to the other two methods. Additionally, the convergence rate of the Barter method is slower than the DE as well as the MRPS. However, the DE and the RPS have a history of a full decade behind them and they have been improved many times. In the present exercise, the RPS is a modified version (MRPS) that has an extra ability for local search. The DE version used here uses the latest (available) schemes of crossover, mutation and recombination. In comparison to this, the Barter method is a nascent one. We need a thorough investigation into the nature and performance of the Barter method.

Bibliography

- Bauer, J.M.: "Harnessing the Swarm: Communication Policy in an Era of Ubiquitous Networks and Disruptive Technologies", *Communications and Strategies*, 45, 2002.
- Box, M.J.: "A new method of constrained optimization and a comparison with other methods". *Comp. J.* 8, pp. 42-52, 1965.
- Bukin, A. D.: *New Minimization Strategy For Non-Smooth Functions*, Budker Institute of Nuclear Physics preprint BUDKER-INP-1997-79, Novosibirsk 1997.
- Cerny, V.: "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm", *J. Opt. Theory Appl.*, 45, 1, 41-51, 1985.
- Eberhart R.C. and Kennedy J.: "A New Optimizer using Particle Swarm Theory", *Proceedings Sixth Symposium on Micro Machine and Human Science*, pp. 39-43. IEEE Service Center, Piscataway, NJ, 1995.
- Fleischer, M.: "Foundations of Swarm Intelligence: From Principles to Practice", *Swarming Network Enabled C4ISR*, arXiv:nlin.AO/0502003 v1 2 Feb 2005.
- G.E.P. Box, "Evolutionary operation: A method for increasing industrial productivity", *Applied Statistics*, 6, pp. 81-101, 1957.
- Glover F., "Future paths for Integer Programming and Links to Artificial Intelligence", *Computers and Operations Research*, 5:533-549, 1986.
- Hayek, F.A.: *The Road to Serfdom*, Univ. of Chicago Press, Chicago, 1944.
- Holland, J.: *Adaptation in Natural and Artificial Systems*, Univ. of Michigan Press, Ann Arbor, 1975.
- Karush, W. *Minima of Functions of Several Variables with Inequalities as Side Constraints*. M.Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago, Chicago, Illinois, 1939.
- Kirkpatrick, S., Gelatt, C.D. Jr., and Vecchi, M.P.: "Optimization by Simulated Annealing", *Science*, 220, 4598, 671-680, 1983.
- Kuhn, H.W. and Tucker, A.W.: "Nonlinear Programming", in Neymann, J. (ed) *Proceedings of Second Berkeley Symposium on Mathematical Statistics and Probability*, Univ. of California Press, Berkley, 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 Stengers, 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 Institute of Nuclear Physics, Novosibirsk, Russia, arXiv: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.
- Sumner, D.J.T.: “The Principles of Collective Animal Behaviour”, *Phil. Trans. R. Soc. B.* 361, pp. 5-22, 2006.
- Takayama, A.: *Mathematical Economics*, The Dryden Press, Hinsdale, Illinois, 1974.
- Törn, A.A and Viitanen, S.: “Topographical Global Optimization using Presampled Points”, *J. of Global Optimization*, 5, pp. 267-276, 1994.
- Törn, A.A.: “A search Clustering Approach to Global Optimization” , in Dixon, LCW and Szegö, G.P. (Eds) *Towards Global Optimization – 2*, North Holland, Amsterdam, 1978.
- Tsallis, C. and Stariolo, D.A.: “Generalized Simulated Annealing”, *ArXiv 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.

```

1: C      MAIN PROGRAM : PROVIDES TO USE BARTER METHOD, REPULSIVE PARTICLE
2: C      SWARM METHOD AND DIFFERENTIAL EVOLUTION METHOD.
3: C      -----
4: C      Adjust the parameters suitably in subroutines DE, RPS and BARTER
5: C      When the program asks for parameters, feed them suitably
6: C      -----
7: PROGRAM DERPSBART
8: IMPLICIT DOUBLE PRECISION (A-H, O-Z)
9: COMMON /KFF/KF,NFCALL,FTIT ! FUNCTION CODE, NO. OF CALLS & TITLE
10: CHARACTER *30 METHOD(3)
11: CHARACTER *1 PROCEED
12: CHARACTER *70 FTIT
13: DIMENSION XX(3,50),KKF(3),MM(3),FMINN(3)
14: DIMENSION X(50)! X IS THE DECISION VARIABLE X IN F(X) TO MINIMIZE
15: C      M IS THE DIMENSION OF THE PROBLEM, KF IS TEST FUNCTION CODE AND
16: C      FMIN IS THE MIN VALUE OF F(X) OBTAINED FROM DE OR RPS
17: WRITE(*,*)'===== WARNING ====='
18: WRITE(*,*)'Adjust parameters in subroutines DE, RPS & BARTER'
19: WRITE(*,*)'===== WARNING ====='
20: METHOD(1)=' : DIFFERENTIAL EVALUATION'
21: METHOD(2)=' : BARTER ALGORITHM'
22: METHOD(3)=' : REPULSIVE PARTICLE SWARM'
23: DO I=1,3
24:
25: IF(I.EQ.1) THEN
26: WRITE(*,*)'===== WELCOME TO DE, RPS, BARTER GO PROGRAMM ====='
27: WRITE(*,*)'TO PROCEED TYPE ANY CHARACTER AND STRIKE ENTER'
28: READ(*,*) PROCEED
29: CALL DE(M,X,FMINDE) ! CALLS DE AND RETURNS OPTIMAL X AND FMIN
30: FMIN=FMINDE
31: ENDIF
32: C      -----
33: IF(I.EQ.2) THEN
34: WRITE(*,*)' '
35: WRITE(*,*)' '
36: WRITE(*,*)'=====BARTER ALGORITHM PROGRAM ====='
37: WRITE(*,*)'TO PROCEED TYPE ANY CHARACTER AND STRIKE ENTER'
38: C      READ(*,*) PROCEED
39: CALL BARTER(M,X,FMINEXC) ! CALLS RPS AND RETURNS OPTIMAL X AND FMIN
40: FMIN=FMINEXC
41: ENDIF
42: C      -----
43: IF(I.EQ.3) THEN
44: WRITE(*,*)' '
45: WRITE(*,*)' '
46: WRITE(*,*)'=====REPULSIVE PARTICLE SWARM PROGRAM ====='
47: WRITE(*,*)'TO PROCEED TYPE ANY CHARACTER AND STRIKE ENTER'
48: C      READ(*,*) PROCEED
49: CALL RPS(M,X,FMINRPS) ! CALLS RPS AND RETURNS OPTIMAL X AND FMIN
50: FMIN=FMINRPS
51: ENDIF
52:
53: C      -----
54: DO J=1,M
55: XX(I,J)=X(J)
56: ENDDO
57: KKF(I)=KF
58: MM(I)=M
59: FMINN(I)=FMIN
60: ENDDO
61: WRITE(*,*)' '
62: WRITE(*,*)' '
63: WRITE(*,*)'----- FINAL RESULTS-----'
64: DO I=1,3
65: WRITE(*,*)'FUNCT CODE=',KKF(I),' FMIN=',FMINN(I),' : DIM=',MM(I)
66: WRITE(*,*)'OPTIMAL DECISION VARIABLES : ',METHOD(I)
67: WRITE(*,*) (XX(I,J),J=1,M)

```

```

68:      WRITE(*,*) '////////////////////////////////////'
69:      ENDDO
70:      WRITE(*,*) 'PROGRAM ENDED'
71:      END
72: C -----
73:      SUBROUTINE DE(M,A,FBEST)
74: C      PROGRAM: "DIFFERENTIAL EVOLUTION ALGORITHM" OF GLOBAL OPTIMIZATION
75: C      THIS METHOD WAS PROPOSED BY R. STORN AND K. PRICE IN 1995. REF --
76: C      "DIFFERENTIAL EVOLUTION - A SIMPLE AND EFFICIENT ADAPTIVE SCHEME
77: C      FOR GLOBAL OPTIMIZATION OVER CONTINUOUS SPACES" : TECHNICAL REPORT
78: C      INTERNATIONAL COMPUTER SCIENCE INSTITUTE, BERKLEY, 1995.
79: C      PROGRAM BY SK MISHRA, DEPT. OF ECONOMICS, NEHU, SHILLONG (INDIA)
80: C -----
81: C      PROGRAM DE
82:      IMPLICIT DOUBLE PRECISION (A-H, O-Z) ! TYPE DECLARATION
83:      PARAMETER(NMAX=500,MMAX=50) ! MAXIMUM DIMENSION PARAMETERS
84:      PARAMETER(RX1=0.5, RX2=5.0) ! to be adjusted suitably, if needed
85: C      RX1 AND RX2 CONTROL THE SCHEME OF CROSSOVER. WHEN RX1=RX2=0, ONLY
86: C      SCHEME3 IS USED. WHEN RX1=RX2=0.5, 50% CASES SCHEME 1 AND REST
87: C      SCHEME 2 IS USED AND SO ON.
88: C      PARAMETER(NCROSS=2) ! CROSS-OVER SCHEME (NCROSS <=0 OR =1 OR >=2)
89:      PARAMETER(IPRINT=500,EPS=1.d-08) ! FOR WATCHING INTERMEDIATE RESULTS
90: C      IT PRINTS THE INTERMEDIATE RESULTS AFTER EACH IPRINT ITERATION AND
91: C      EPS DETERMINES ACCURACY FOR TERMINATION. IF EPS= 0, ALL ITERATIONS
92: C      WOULD BE UNDERGONE EVEN IF NO IMPROVEMENT IN RESULTS IS THERE.
93: C      ULTIMATELY "DID NOT CONVERGE" IS REOPORTED.
94:      COMMON /RNDM/IU,IV ! RANDOM NUMBER GENERATION (IU = 4-DIGIT SEED)
95:      INTEGER IU,IV ! FOR RANDOM NUMBER GENERATION
96:      COMMON /KFF/KF,NFCALL,FTIT ! FUNCTION CODE, NO. OF CALLS * TITLE
97:      CHARACTER *70 FTIT ! TITLE OF THE FUNCTION
98: C -----
99: C      THE PROGRAM REQUIRES INPUTS FROM THE USER ON THE FOLLOWING -----
100: C      (1) FUNCTION CODE (KF), (2) NO. OF VARIABLES IN THE FUNCTION (M);
101: C      (3) N=POPULATION SIZE (SUGGESTED 10 TIMES OF NO. OF VARIABLES, M,
102: C      FOR SMALLER PROBLEMS N=100 WORKS VERY WELL);
103: C      (4) PCROS = PROB. OF CROSS-OVER (SUGGESTED : ABOUT 0.85 TO .99);
104: C      (5) FACT = SCALE (SUGGESTED 0.5 TO .95 OR 1, ETC);
105: C      (6) ITER = MAXIMUM NUMBER OF ITERATIONS PERMITTED (5000 OR MORE)
106: C      (7) RANDOM NUMBER SEED (4 DIGITS INTEGER)
107: C -----
108:      DIMENSION X(NMAX,MMAX),Y(NMAX,MMAX),A(MMAX),FV(NMAX)
109:      DIMENSION IR(3)
110: C -----
111: C      ----- SELECT THE FUNCTION TO MINIMIZE AND ITS DIMENSION -----
112:      CALL FSELECT(KF,M,FTIT)
113: C      SPECIFY OTHER PARAMETERS -----
114:      WRITE(*,*) 'POPULATION SIZE [N] AND NO. OF ITERATIONS [ITER] ?'
115:      WRITE(*,*) 'SUGGESTED : N => 100 OR =>10.M; ITER 10000 OR SO'
116:      READ(*,*) N,ITER
117:      WRITE(*,*) 'CROSSOVER PROBABILITY [PCROS] AND SCALE [FACT] ?'
118:      WRITE(*,*) 'SUGGESTED : PCROS ABOUT 0.9; FACT=.5 OR LARGER BUT <=1'
119:      READ(*,*) PCROS,FACT
120:      WRITE(*,*) 'RANDOM NUMBER SEED ?'
121:      WRITE(*,*) 'A FOUR-DIGIT POSITIVE ODD INTEGER, SAY, 1171'
122:      READ(*,*) IU
123:
124:      NFCALL=0 ! INITIALIZE COUNTER FOR FUNCTION CALLS
125:      GBEST=1.D30 ! TO BE USED FOR TERMINATION CRITERION
126: C      INITIALIZATION : GENERATE X(N,M) RANDOMLY
127:      DO I=1,N
128:      DO J=1,M
129:      CALL RANDOM(RAND)
130:      X(I,J)=(RAND-.5D00)*2000
131: C      RANDOM NUMBERS BETWEEN -RRANGE AND +RRANGE (BOTH EXCLUSIVE)
132:      ENDDO
133:      ENDDO
134:      WRITE(*,*) 'COMPUTING --- PLEASE WAIT '

```

```

135:      IPCOUNT=0
136:      DO 100 ITR=1, ITER      ! ITERATION BEGINS
137: C -----
138: C      EVALUATE ALL X FOR THE GIVEN FUNCTION
139:      DO I=1, N
140:      DO J=1, M
141:      A(J)=X(I, J)
142:      ENDDO
143:      CALL FUNC(A, M, F)
144: C      STORE FUNCTION VALUES IN FV VECTOR
145:      FV(I)=F
146:      ENDDO
147: C -----
148: C      FIND THE FITTEST (BEST) INDIVIDUAL AT THIS ITERATION
149:      FBEST=FV(1)
150:      KB=1
151:      DO IB=2, N
152:      IF (FV(IB) .LT. FBEST) THEN
153:      FBEST=FV(IB)
154:      KB=IB
155:      ENDIF
156:      ENDDO
157: C      BEST FITNESS VALUE = FBEST : INDIVIDUAL X(KB)
158: C -----
159: C      GENERATE OFFSPRINGS
160:      DO I=1, N      ! I LOOP BEGINS
161: C      INITIALIZE CHILDREN IDENTICAL TO PARENTS; THEY WILL CHANGE LATER
162:      DO J=1, M
163:      Y(I, J)=X(I, J)
164:      ENDDO
165: C      SELECT RANDOMLY THREE OTHER INDIVIDUALS
166: 20      DO IRI=1, 3      ! IRI LOOP BEGINS
167:      IR(IRI)=0
168:
169:      CALL RANDOM(RAND)
170:      IRJ=INT(RAND*N)+1
171: C      CHECK THAT THESE THREE INDIVIDUALS ARE DISTICT AND OTHER THAN I
172:      IF (IRI .EQ. 1 .AND. IRJ .NE. I) THEN
173:      IR(IRI)=IRJ
174:      ENDIF
175:      IF (IRI .EQ. 2 .AND. IRJ .NE. I .AND. IRJ .NE. IR(1)) THEN
176:      IR(IRI)=IRJ
177:      ENDIF
178:      IF (IRI .EQ. 3 .AND. IRJ .NE. I .AND. IRJ .NE. IR(1) .AND. IRJ .NE. IR(2)) THEN
179:      IR(IRI)=IRJ
180:      ENDIF
181:      ENDDO      ! IRI LOOP ENDS
182: C      CHECK IF ALL THE THREE IR ARE POSITIVE (INTEGERS)
183:      DO IX=1, 3
184:      IF (IR(IX) .LE. 0) THEN
185:      GOTO 20      ! IF NOT THEN REGENERATE
186:      ENDIF
187:      ENDDO
188: C      THREE RANDOMLY CHOSEN INDIVIDUALS DIFFERENT FROM I AND DIFFERENT
189: C      FROM EACH OTHER ARE IR(1), IR(2) AND IR(3)
190: C      ===== randomization of ncross =====
191: C      RANDOMIZES NCROSS
192:      NCROSS=0
193:      CALL RANDOM(RAND)
194:      IF (RAND .GT. RX1) NCROSS=1
195:      IF (RAND .GT. RX2) NCROSS=2
196:
197: C -----
198: C      NO CROSS OVER, ONLY REPLACEMENT THAT IS PROBABILISTIC
199:      IF (NCROSS .LE. 0) THEN
200:      DO J=1, M      ! J LOOP BEGINS
201:      CALL RANDOM(RAND)

```



```

202:      IF (RAND.LE.PCROS) THEN ! REPLACE IF RAND < PCROS
203:      A(J)=X(IR(1),J)+(X(IR(2),J)-X(IR(3),J))*FACT ! CANDIDATE CHILD
204:      ENDIF
205:      ENDDO ! J LOOP ENDS
206:      ENDIF
207:
208: C -----
209: C CROSSOVER SCHEME (EXPONENTIAL) SUGGESTED BY KENNETH PRICE IN HIS
210: C PERSONAL LETTER TO THE AUTHOR (DATED SEPTEMBER 29, 2006)
211: IF (NCROSS.EQ.1) THEN
212:     CALL RANDOM(RAND)
213:     1 JR=INT(RAND*M)+1
214:     J=JR
215:     2 A(J)=X(IR(1),J)+FACT*(X(IR(2),J)-X(IR(3),J))
216:     3 J=J+1
217:     IF (J.GT.M) J=1
218:     4 IF (J.EQ.JR) GOTO 10
219:     5 CALL RANDOM(RAND)
220:     IF (PCROS.LE.RAND) GOTO 2
221:     6 A(J)=X(I,J)
222:     7 J=J+1
223:     IF (J.GT.M) J=1
224:     8 IF (J.EQ.JR) GOTO 10
225:     9 GOTO 6
226:    10 CONTINUE
227: ENDIF
228: C -----
229: C CROSSOVER SCHEME (NEW) SUGGESTED BY KENNETH PRICE IN HIS
230: C PERSONAL LETTER TO THE AUTHOR (DATED OCTOBER 18, 2006)
231: IF (NCROSS.GE.2) THEN
232:     CALL RANDOM(RAND)
233:     IF (RAND.LE.PCROS) THEN
234:         CALL NORMAL(RN)
235:         DO J=1,M
236:             A(J)=X(I,J)+(X(IR(1),J)+X(IR(2),J)-2*X(I,J))*RN
237:         ENDDO
238:     ELSE
239:         DO J=1,M
240:             A(J)=X(I,J)+(X(IR(1),J)-X(IR(2),J))*FACT ASSUMED TO BE 1
241:         ENDDO
242:     ENDIF
243: ENDIF
244: C -----
245:     CALL FUNC(A,M,F) ! EVALUATE THE OFFSPRING
246:     IF (F.LT.FV(I)) THEN ! IF BETTER, REPLACE PARENTS BY THE CHILD
247:         FV(I)=F
248:         DO J=1,M
249:             Y(I,J)=A(J)
250:         ENDDO
251:     ENDIF
252: ENDDO ! I LOOP ENDS
253: DO I=1,N
254:     DO J=1,M
255:         X(I,J)=Y(I,J) ! NEW GENERATION IS A MIX OF BETTER PARENTS AND
256: C BETTER CHILDREN
257:     ENDDO
258: ENDDO
259: IPCOUNT=IPCOUNT+1
260: IF (IPCOUNT.EQ.IPRINT) THEN
261:     DO J=1,M
262:         A(J)=X(KB,J)
263:     ENDDO
264:     WRITE(*,*) (X(KB,J),J=1,M), ' FBEST UPTO NOW = ',FBEST
265:     WRITE(*,*) 'TOTAL NUMBER OF FUNCTION CALLS = ',NFCALL
266:     IF (DABS(FBEST-GBEST).LT.EPS) THEN
267:         WRITE(*,*) FTIT
268:         WRITE(*,*) 'COMPUTATION OVER'

```

```

269:         RETURN
270:     ELSE
271:         GBEST=FBEST
272:     ENDIF
273:     IPCOUNT=0
274: ENDIF
275: C -----
276: 100 ENDDO      ! ITERATION ENDS : GO FOR NEXT ITERATION, IF APPLICABLE
277: C -----
278: WRITE(*,*) 'DID NOT CONVERGE. REDUCE EPS OR RAISE ITER OR DO BOTH'
279: WRITE(*,*) 'INCREASE N, PCROS, OR SCALE FACTOR (FACT)'
280: RETURN
281: END
282: C -----
283: SUBROUTINE NORMAL(R)
284: C      Program to generate N(0,1) from Rectangular Random Numbers
285: C      It uses Variate Transformation for this purpose.
286: C -----
287: C      IF U1 AND U2 ARE UNIFORMLY DISTRIBUTED RANDOM NUMBERS (0,1),
288: C      THEN X=[(-2*ln(U1))**.5]*(COS(2*PI*U2) IS N(0,1)
289: C      PI = 4*ARCTAN(1.0)= 3.1415926535897932384626433832795
290: C      2*PI = 6.283185307179586476925286766559
291: C -----
292: IMPLICIT DOUBLE PRECISION (A-H,O-Z)
293: COMMON /RNDM/IU,IV
294: INTEGER IU,IV
295: C -----
296: CALL RANDOM(RAND) ! INVOKES RANDOM TO GENERATE UNIFORM RAND [0, 1]
297: U1=RAND ! U1 IS UNIFORMLY DISTRIBUTED [0, 1]
298: CALL RANDOM(RAND) ! INVOKES RANDOM TO GENERATE UNIFORM RAND [0, 1]
299: U2=RAND ! U2 IS UNIFORMLY DISTRIBUTED [0, 1]
300: R=DSQRT(-2.D0*DLOG(U1))
301: R=R*DCOS(U2*6.283185307179586476925286766559D00)
302: C      R=R*DCOS(U2*6.28318530718D00)
303: RETURN
304: END
305: C -----
306: C      RANDOM NUMBER GENERATOR (UNIFORM BETWEEN 0 AND 1 - BOTH EXCLUSIVE)
307: SUBROUTINE RANDOM(RAND1)
308:     DOUBLE PRECISION RAND1
309:     COMMON /RNDM/IU,IV
310:     INTEGER IU,IV
311:     RAND=REAL(RAND1)
312:     IV=IU*65539
313:     IF (IV.LT.0) THEN
314:         IV=IV+2147483647+1
315:     ENDIF
316:     RAND=IV
317:     IU=IV
318:     RAND=RAND*0.4656613E-09
319:     RAND1= (RAND)
320:     RETURN
321: END
322: C -----
323: C -----
324: SUBROUTINE RPS(M,BST,FMINIM)
325: C      PROGRAM TO FIND GLOBAL MINIMUM BY REPULSIVE PARTICLE SWARM METHOD
326: C      WRITTEN BY SK MISHRA, DEPT. OF ECONOMICS, NEHU, SHILLONG (INDIA)
327: C -----
328: PARAMETER (N=100,NN=50,MX=50,NSTEP=11,ITRN=10000,NSIGMA=1,ITOP=3)
329: C      PARAMETER (N=50,NN=25,MX=100,NSTEP=9,ITRN=10000,NSIGMA=1,ITOP=3)
330: C      PARAMETER (N=100,NN=15,MX=100,NSTEP=9,ITRN=10000,NSIGMA=1,ITOP=3)
331: C      IN CERTAIN CASES THE ONE OR THE OTHER SPECIFICATION WORKS BETTER
332: C      DIFFERENT SPECIFICATIONS OF PARAMETERS MAY SUIT DIFFERENT TYPES
333: C      OF FUNCTIONS OR DIMENSIONS - ONE HAS TO DO SOME TRIAL AND ERROR
334: C -----
335: C      N = POPULATION SIZE. IN MOST OF THE CASES N=30 IS OK. ITS VALUE

```

```

336: C      MAY BE INCREASED TO 50 OR 100 TOO. THE PARAMETER NN IS THE SIZE OF
337: C      RANDOMLY CHOSEN NEIGHBOURS. 15 TO 25 (BUT SUFFICIENTLY LESS THAN
338: C      N) IS A GOOD CHOICE. MX IS THE MAXIMAL SIZE OF DECISION VARIABLES.
339: C      IN F(X1, X2,...,XM) M SHOULD BE LESS THAN OR EQUAL TO MX. ITRN IS
340: C      THE NO. OF ITERATIONS. IT MAY DEPEND ON THE PROBLEM. 200(AT LEAST)
341: C      TO 500 ITERATIONS MAY BE GOOD ENOUGH. BUT FOR FUNCTIONS LIKE
342: C      ROSENBROCK OR GRIEWANK OF LARGE SIZE (SAY M=30) IT IS NEEDED THAT
343: C      ITRN IS LARGE, SAY 5000 OR EVEN 10000.
344: C      SIGMA INTRODUCES PERTURBATION & HELPS THE SEARCH JUMP OUT OF LOCAL
345: C      OPTIMA. FOR EXAMPLE : RASTRIGIN FUNCTION OF DIMENSION 30 OR LARGER
346: C      NSTEP DOES LOCAL SEARCH BY TUNNELLING AND WORKS WELL BETWEEN 5 AND
347: C      15, WHICH IS MUCH ON THE HIGHER SIDE.
348: C      ITOP =<=1 (RING); ITOP=2 (RING AND RANDOM); ITOP=>3 (RANDOM)
349: C      NSIGMA=0 (NO CHAOTIC PERTURBATION); NSIGMA=1 (CHAOTIC PERTURBATION)
350: C      NOTE THAT NSIGMA=1 NEED NOT ALWAYS WORK BETTER (OR WORSE)
351: C      SUBROUTINE FUNC( ) DEFINES OR CALLS THE FUNCTION TO BE OPTIMIZED.
352: C      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
353: C      COMMON /RNDM/IU,IV
354: C      COMMON /KFF/KF,NFCALL,FTIT
355: C      INTEGER IU,IV
356: C      CHARACTER *70 FTIT
357: C      DIMENSION X(N,MX), V(N,MX), A(MX), VI(MX)
358: C      DIMENSION XX(N,MX), F(N), V1(MX), V2(MX), V3(MX), V4(MX), BST(MX)
359: C      A1 A2 AND A3 ARE CONSTANTS AND W IS THE INERTIA WEIGHT.
360: C      OCCASIONALLY, TINKERING WITH THESE VALUES, ESPECIALLY A3, MAY BE
361: C      NEEDED.
362: C      DATA A1,A2,A3,W,SIGMA,EPSI /.5D0,.5D0,5.D-04,.5D00,1.D-03,1.D-08/
363: C      -----
364: C      CALL SUBROUTINE FOR CHOOSING FUNCTION (KF) AND ITS DIMENSION (M)
365: C      CALL FSELECT(KF,M,FTIT)
366: C      -----
367: C      GGBEST=1.D30 ! TO BE USED FOR TERMINATION CRITERION
368: C      LCOUNT=0
369: C      NFCALL=0
370: C      WRITE(*,*) '4-DIGITS SEED FOR RANDOM NUMBER GENERATION'
371: C      WRITE(*,*) 'A FOUR-DIGIT POSITIVE ODD INTEGER, SAY, 1171'
372: C      READ(*,*) IU
373: C      IU=1111
374: C      FMIN=1.0E30
375: C      GENERATE N-SIZE POPULATION OF M-TUPLE PARAMETERS X(I,J) RANDOMLY
376: C      DO I=1,N
377: C      DO J=1,M
378: C      CALL RANDOM(RAND)
379: C      X(I,J)=(RAND-0.5D00)*2000
380: C      WE GENERATE RANDOM(-5,5). HERE MULTIPLIER IS 10. TINKERING IN SOME
381: C      CASES MAY BE NEEDED
382: C      ENDDO
383: C      F(I)=1.0D30
384: C      ENDDO
385: C      INITIALISE VELOCITIES V(I) FOR EACH INDIVIDUAL IN THE POPULATION
386: C      DO I=1,N
387: C      DO J=1,M
388: C      CALL RANDOM(RAND)
389: C      V(I,J)=(RAND-0.5D+00)
390: C      V(I,J)=RAND
391: C      ENDDO
392: C      ENDDO
393: C      DO 100 ITER=1,ITRN
394: C      LET EACH INDIVIDUAL SEARCH FOR THE BEST IN ITS NEIGHBOURHOOD
395: C      DO I=1,N
396: C      DO J=1,M
397: C      A(J)=X(I,J)
398: C      VI(J)=V(I,J)
399: C      ENDDO
400: C      CALL LSRCH(A,M,VI,NSTEP,FI)
401: C      IF (FI.LT.F(I)) THEN
402: C      F(I)=FI

```

```

403:         DO IN=1,M
404:         BST(IN)=A(IN)
405:         ENDDO
406: C      F(I) CONTAINS THE LOCAL BEST VALUE OF FUNCTION FOR ITH INDIVIDUAL
407: C      XX(I,J) IS THE M-TUPLE VALUE OF X ASSOCIATED WITH LOCAL BEST F(I)
408:         DO J=1,M
409:         XX(I,J)=A(J)
410:         ENDDO
411:         ENDIF
412:     ENDDO
413: C      NOW LET EVERY INDIVIDUAL RANDOMLY COSULT NN(<<N) COLLEAGUES AND
414: C      FIND THE BEST AMONG THEM
415: DO I=1,N
416: C -----
417: IF (ITOP.GE.3) THEN
418: C      RANDOM TOPOLOGY *****
419: C      CHOOSE NN COLLEAGUES RANDOMLY AND FIND THE BEST AMONG THEM
420:     BEST=1.0D30
421:     DO II=1,NN
422:     CALL RANDOM(RAND)
423:     NF=INT(RAND*N)+1
424:     IF (BEST.GT.F(NF)) THEN
425:     BEST=F(NF)
426:     NFBEST=NF
427:     ENDIF
428:     ENDDO
429: ENDIF
430: C -----
431: IF (ITOP.EQ.2) THEN
432: C      RING + RANDOM TOPOLOGY *****
433: C      REQUIRES THAT THE SUBROUTINE NEIGHBOR IS TURNED ALIVE
434:     BEST=1.0D30
435:     CALL NEIGHBOR(I,N,I1,I3)
436:     DO II=1,NN
437:     IF (II.EQ.1) NF=I1
438:     IF (II.EQ.2) NF=I
439:     IF (II.EQ.3) NF=I3
440:     IF (II.GT.3) THEN
441:     CALL RANDOM(RAND)
442:     NF=INT(RAND*N)+1
443:     ENDIF
444:     IF (BEST.GT.F(NF)) THEN
445:     BEST=F(NF)
446:     NFBEST=NF
447:     ENDIF
448:     ENDDO
449: ENDIF
450: C -----
451: IF (ITOP.LE.1) THEN
452: C      RING TOPOLOGY *****
453: C      REQUIRES THAT THE SUBROUTINE NEIGHBOR IS TURNED ALIVE
454:     BEST=1.0D30
455:     CALL NEIGHBOR(I,N,I1,I3)
456:     DO II=1,3
457:     IF (II.NE.I) THEN
458:     IF (II.EQ.1) NF=I1
459:     IF (II.EQ.3) NF=I3
460:     IF (BEST.GT.F(NF)) THEN
461:     BEST=F(NF)
462:     NFBEST=NF
463:     ENDIF
464:     ENDIF
465:     ENDDO
466: ENDIF
467: C -----
468: C      IN THE LIGHT OF HIS OWN AND HIS BEST COLLEAGUES EXPERIENCE, THE
469: C      INDIVIDUAL I WILL MODIFY HIS MOVE AS PER THE FOLLOWING CRITERION

```

```

470: C      FIRST, ADJUSTMENT BASED ON ONES OWN EXPERIENCE
471: C      AND OWN BEST EXPERIENCE IN THE PAST (XX(I))
472:         DO J=1,M
473:         CALL RANDOM(RAND)
474:         V1(J)=A1*RAND*(XX(I,J)-X(I,J))
475: C      THEN BASED ON THE OTHER COLLEAGUES BEST EXPERIENCE WITH WEIGHT W
476: C      HERE W IS CALLED AN INERTIA WEIGHT 0.01< W < 0.7
477: C      A2 IS THE CONSTANT NEAR BUT LESS THAN UNITY
478:         CALL RANDOM(RAND)
479:         V2(J)=V(I,J)
480:         IF (F(NFBEST) .LT. F(I)) THEN
481:         V2(J)=A2*W*RAND*(XX(NFBEST,J)-X(I,J))
482:         ENDIF
483: C      THEN SOME RANDOMNESS AND A CONSTANT A3 CLOSE TO BUT LESS THAN UNITY
484:         CALL RANDOM(RAND)
485:         RND1=RAND
486:         CALL RANDOM(RAND)
487:         V3(J)=A3*RAND*W*RND1
488: C         V3(J)=A3*RAND*W
489: C      THEN ON PAST VELOCITY WITH INERTIA WEIGHT W
490:         V4(J)=W*V(I,J)
491: C      FINALLY A SUM OF THEM
492:         V(I,J)= V1(J)+V2(J)+V3(J)+V4(J)
493:         ENDDO
494: ENDDO
495: C      CHANGE X
496: DO I=1,N
497: DO J=1,M
498: RANDS=0.D00
499: C      -----
500: IF (NSIGMA.EQ.1) THEN
501: CALL RANDOM(RAND) ! FOR CHAOTIC PERTURBATION
502: IF (DABS(RAND-.5D00) .LT. SIGMA) RANDS=RAND-0.5D00
503: C      SIGMA CONDITIONED RANDS INTRODUCES CHAOTIC ELEMENT IN TO LOCATION
504: C      IN SOME CASES THIS PERTURBATION HAS WORKED VERY EFFECTIVELY WITH
505: C      PARAMETER (N=100,NN=15,MX=100,NSTEP=9,ITRN=100000,NSIGMA=1,ITOP=2)
506: ENDF
507: C      -----
508: X(I,J)=X(I,J)+V(I,J)*(1.D00+RANDS)
509: ENDDO
510: ENDDO
511: DO I=1,N
512: IF (F(I) .LT. FMIN) THEN
513: FMIN=F(I)
514: II=I
515: DO J=1,M
516: BST(J)=XX(II,J)
517: ENDDO
518: ENDF
519: ENDDO
520: IF (LCOUNT.EQ.100) THEN
521: LCOUNT=0
522: WRITE(*,*) 'OPTIMAL SOLUTION UPTO THIS (FUNCTION CALLS=',NFCALL,')'
523: WRITE(*,*) 'X = ', (BST(J),J=1,M), ' MIN F = ',FMIN
524: C      WRITE(*,*) 'NO. OF FUNCTION CALLS = ',NFCALL
525: IF (DABS(FMIN-GGBEST) .LT. EPSI) THEN
526: WRITE(*,*) 'COMPUTATION OVER'
527: FMINIM=FMIN
528: RETURN
529: ELSE
530: GGBEST=FMIN
531: ENDF
532: ENDF
533: LCOUNT=LCOUNT+1
534: 100 CONTINUE
535: WRITE(*,*) 'COMPUTATION OVER:',FTIT
536: FMINIM=FMIN

```

```

537:      RETURN
538:      END
539: C -----
540:      SUBROUTINE LSRCH(A,M,VI,NSTEP,FI)
541:      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
542:      COMMON /RNDM/IU,IV
543:      INTEGER IU,IV
544:      DIMENSION A(*),B(100),VI(*)
545:      AMN=1.0D30
546:      DO J=1,NSTEP
547:          DO JJ=1,M
548:              B(JJ)=A(JJ)+(J-(NSTEP/2)-1)*VI(JJ)
549:          ENDDO
550:      CALL FUNC(B,M,FI)
551:      IF(FI.LT.AMN) THEN
552:          AMN=FI
553:          DO JJ=1,M
554:              A(JJ)=B(JJ)
555:          ENDDO
556:      ENDIF
557:      ENDDO
558:      FI=AMN
559:      RETURN
560:      END
561: C -----
562: C      THIS SUBROUTINE IS NEEDED IF THE NEIGHBOURHOOD HAS RING TOPOLOGY
563: C      EITHER PURE OR HYBRIDIZED
564:      SUBROUTINE NEIGHBOR(I,N,J,K)
565:      IF(I-1.GE.1 .AND. I.LT.N) THEN
566:          J=I-1
567:          K=I+1
568:      ELSE
569:          IF(I-1.LT.1) THEN
570:              J=N-I+1
571:              K=I+1
572:          ENDIF
573:          IF(I.EQ.N) THEN
574:              J=I-1
575:              K=1
576:          ENDIF
577:      ENDIF
578:      RETURN
579:      END
580: C -----
581: C      BARTER ALGORITHM
582: C -----
583: C      ***** THIS METHOD IS PROPOSED BY SK MISHRA *****
584: C      PROGRAM BY SK MISHRA, DEPT. OF ECONOMICS, NEHU, SHILLONG (INDIA)
585: C -----
586:      SUBROUTINE BARTER(M,BEST,FBEST)
587:      IMPLICIT DOUBLE PRECISION (A-H,O-Z) ! TYPE DECLARATION
588:      PARAMETER(IPRINT=500, EPS=1.D-08)
589:      COMMON /RNDM/IU,IV ! RANDOM NUMBER GENERATION (IU = 4-DIGIT SEED)
590:      INTEGER IU,IV ! FOR RANDOM NUMBER GENERATION
591:      COMMON /KFF/KF,NFCALL,FTIT ! FUNCTION CODE,NO. OF CALLS & TITLE
592:      CHARACTER *70 FTIT ! TITLE OF THE FUNCTION
593: C -----
594: C -----
595: C -----
596:      DIMENSION X(500,50),FV(500),A(50),B(50),BEST(50)
597: C -----
598: C      ----- SELECT THE FUNCTION TO MINIMIZE AND ITS DIMENSION -----
599: C      CALL FSELECT(KF,M,FTIT)
600: C      SPECIFY OTHER PARAMETERS -----
601:      WRITE(*,*)'POPULATION SIZE [N] AND NO. OF ITERATIONS [ITER] ?'
602: C      READ(*,*) N,ITER
603: C -----

```

```

604:      N=M*10
605:      IF(N.LT.100) N=100
606:      ITER=10000
607:      WRITE(*,*) 'RANDOM NUMBER SEED ?'
608: C      READ(*,*) IU
609:      IU=1111
610: C      -----
611:      NFCALL=0 ! INITIALIZE COUNTER FOR FUNCTION CALLS
612:      GBEST=1.D30 ! TO BE USED FOR TERMINATION CRITERION
613: C      INITIALIZATION : GENERATE X(N,M) RANDOMLY
614:      DO I=1,N
615:      DO J=1,M
616:      CALL RANDOM(RAND)
617:      X(I,J)=(RAND-.5D00)*2000
618: C      RANDOM NUMBERS BETWEEN -RRANGE AND +RRANGE (BOTH EXCLUSIVE)
619:      ENDDO
620:      ENDDO
621:      WRITE(*,*) 'COMPUTING --- PLEASE WAIT '
622:      IPCOUNT=0
623:      DO 100 ITR=1,ITER ! ITERATION BEGINS
624:
625: C      EVALUATE ALL X FOR THE GIVEN FUNCTION
626:      DO I=1,N
627:      DO J=1,M
628:      A(J)=X(I,J)
629:      ENDDO
630:      CALL FUNC(A,M,FA)
631:      FV(I)=FA
632:      CALL SEARCH(A,M,FI)
633: C      STORE FUNCTION VALUES IN FV VECTOR
634:      IF(FI.LT.FV(I)) THEN
635:      DO J=1,M
636:      X(I,J)=A(J)
637:      ENDDO
638:      FV(I)=FI
639:      ENDIF
640:      ENDDO
641: C      -----
642:      DO I=1,N
643: C      CHOOSE IB TH INDIVIDUAL RANDOMLY
644:      CALL RANDOM(RAND)
645:      IB=INT(RAND*N)+1 !THE RANDOM INDIVIDUAL
646:      IB=2*M+1
647: C      STORE ITH IN A AND RANDOMLY SELECTED INDIVIDUAL IN B
648:      DO J=1,M
649:      A(J)=X(I,J)
650:      B(J)=X(IB,J) ! of the individual randomly selected
651:      ENDDO
652: C      CHOSE AN INDEX BETWEEN 1 AND M RANDOMLY
653:      CALL RANDOM(RAND)
654:      JA=INT(RAND*M)+1
655: C      CHOOSE ANOTHER INDEX RANDOMLY : MUST BE DIFFERENT FROM JA
656: 1      CALL RANDOM(RAND)
657:      JB=INT(RAND*M)+1
658:      IF(JA.EQ.JB) GOTO 1
659: C      EXCHANGE A(JA) WITH B(JB)
660:      TEMP1=A(JA)
661:      TEMP2=B(JB)
662:      CALL NORMAL(RN) ! OBTAIN STANDARD NORMAL RANDOM NUMBER
663:      A(JB)=A(JB)+RN*TEMP2
664:      B(JB)=B(JB)-RN*TEMP2
665:      A(JA)=A(JA)-RN*TEMP1
666:      B(JA)=B(JA)+RN*TEMP1
667: C      EVALUATE A AND B VECTORS
668:      CALL FUNC(A,M,FA)
669:      CALL FUNC(B,M,FB)
670: C      CHECK IF FA < FV(I) AND FB < FV(IB)

```

```

671:         IF (FA.LT.FV(I) .and. FB.LT.FV(IB)) THEN
672:             FV(I)=FA
673:             FV(IB)=FB
674:             DO J=1,M
675:                 X(I,J)=A(J)
676:                 X(IB,J)=B(J)
677:             ENDDO
678:         ENDIF
679:     ENDDO
680: C -----
681: C     FIND THE BEST
682:     FBEST=1.D30
683:     DO I=1,N
684:         IF (FV(I) .LT. FBEST) THEN
685:             FBEST=FV(I)
686:             KB=I
687:         ENDIF
688:     ENDDO
689:     DO J=1,M
690:         BEST(J)=X(KB,J)
691:     ENDDO
692: C -----
693:     IPCOUNT=IPCOUNT+1
694:     IF (IPCOUNT.EQ.IPRINT) THEN
695:         WRITE(*,*) (BEST(J),J=1,M), ' FBEST UPTO NOW = ',FBEST
696:         WRITE(*,*) 'TOTAL NUMBER OF FUNCTION CALLS = ',NFCALL
697:         IF (DABS(FBEST-GBEST) .LT. EPS) THEN
698:             WRITE(*,*) FTIT
699:             WRITE(*,*) 'COMPUTATION OVER'
700:             RETURN
701:         ELSE
702:             GBEST=FBEST
703:         ENDIF
704:     IPCOUNT=0
705:     ENDIF
706: C -----
707: 100 ENDDO ! ITERATION ENDS : GO FOR NEXT ITERATION, IF APPLICABLE
708: C -----
709:     WRITE(*,*) 'DID NOT CONVERGE. REDUCE EPS OR RAISE ITER OR DO BOTH'
710:     WRITE(*,*) 'INCREASE N, PCROS, OR SCALE FACTOR (FACT)'
711:     RETURN
712:     END
713: C -----
714:     SUBROUTINE SEARCH(A,M,FI)
715:     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
716:     COMMON /RNDM/IU,IV
717:     INTEGER IU,IV
718:     DIMENSION A(*),B(100)
719:     NSTEP=11
720:     AMN=1.0D30
721:     DO J=1,NSTEP
722:         DO JJ=1,M
723:             CALL RANDOM(RAND)
724:             B(JJ)=A(JJ) + (J-(NSTEP/2)-1)*RAND*0.0001D00
725:         ENDDO
726:     CALL FUNC(B,M,FI)
727:     IF (FI.LT.AMN) THEN
728:         AMN=FI
729:         DO JJ=1,M
730:             A(JJ)=B(JJ)
731:         ENDDO
732:     ENDIF
733:     ENDDO
734:     FI=AMN
735:     RETURN
736:     END
737: C -----

```



```

738:      SUBROUTINE FSELECT(KF,M,FTIT)
739: C      THE PROGRAM REQUIRES INPUTS FROM THE USER ON THE FOLLOWING -----
740: C      (1) FUNCTION CODE (KF), (2) NO. OF VARIABLES IN THE FUNCTION (M);
741: CHARACTER *70 TIT(100),FTIT
742: WRITE(*,*) '-----'
743: DATA TIT(1) / 'KF=1 NEW FUNCTION(N#1) 2-VARIABLES M=2' /
744: DATA TIT(2) / 'KF=2 NEW FUNCTION(N#2) 2-VARIABLES M=2' /
745: DATA TIT(3) / 'KF=3 NEW FUNCTION(N#3) 2-VARIABLES M=2' /
746: DATA TIT(4) / 'KF=4 NEW FUNCTION(N#4) 2-VARIABLES M=2' /
747: DATA TIT(5) / 'KF=5 NEW QUINTIC FUNCTION M-VARIABLES M=?' /
748: DATA TIT(6) / 'KF=6 NEW NEEDLE-EYE FUNCTION (N#6) M-VARIABLES M=?' /
749: DATA TIT(7) / 'KF=7 NEW ZERO-SUM FUNCTION (N#7) M-VARIABLES M=?' /
750: DATA TIT(8) / 'KF=8 CORANA FUNCTION 4-VARIABLES M=4' /
751: DATA TIT(9) / 'KF=9 MODIFIED RCOS FUNCTION 2-VARIABLES M=2' /
752: DATA TIT(10) / 'KF=10 FREUDENSTEIN ROTH FUNCTION 2-VARIABLES M=2' /
753: DATA TIT(11) / 'KF=11 ANNS XOR FUNCTION 9-VARIABLES M=9' /
754: DATA TIT(12) / 'KF=12 PERM FUNCTION #1 (SET BETA) 4-VARIABLES M=4' /
755: DATA TIT(13) / 'KF=13 PERM FUNCTION #2 (SET BETA) M-VARIABLES M=?' /
756: DATA TIT(14) / 'KF=14 POWER-SUM FUNCTION 4-VARIABLES M=4' /
757: DATA TIT(15) / 'KF=15 GOLDSTEIN PRICE FUNCTION 2-VARIABLES M=2' /
758: DATA TIT(16) / 'KF=16 BUKIN 6TH FUNCTION 2-VARIABLES M=2' /
759: DATA TIT(17) / 'KF=17 NEW FUNCTION (N#8) 2-VARIABLES M=2' /
760: DATA TIT(18) / 'KF=18 DEFL CORRUG SPRING FUNCTION M-VARIABLES M=?' /
761: DATA TIT(19) / 'KF=19 NEW FACTORIAL FUNCTION M-VARIABLES M=?' /
762: DATA TIT(20) / 'KF=20 NEW DECANOMIAL FUNCTION 2-VARIABLES M=2' /
763: DATA TIT(21) / 'KF=21 JUDGE FUNCTION 2-VARIABLES M=2' /
764: DATA TIT(22) / 'KF=22 NEW DODECAL FUNCTION 3-VARIABLES M=3' /
765: DATA TIT(23) / 'KF=23 NEW SUM-EQ-PROD FUNCTION 2-VARIABLES M=2' /
766: DATA TIT(24) / 'KF=24 NEW AM-EQ-GM FUNCTION M-VARIABLES M=?' /
767: DATA TIT(25) / 'KF=25 YAO-LIU FUNCTION#2 M-VARIABLES M=?' /
768: DATA TIT(26) / 'KF=26 YAO-LIU FUNCTION#3 M-VARIABLES M=?' /
769: DATA TIT(27) / 'KF=27 YAO-LIU FUNCTION#4 M-VARIABLES M=?' /
770: DATA TIT(28) / 'KF=28 YAO-LIU FUNCTION#6 M-VARIABLES M=?' /
771: DATA TIT(29) / 'KF=29 YAO-LIU FUNCTION#7 M-VARIABLES M=?' /
772: DATA TIT(30) / 'KF=30 YAO-LIU FUNCTION#12 M-VARIABLES M=?' /
773: DATA TIT(31) / 'KF=31 YAO-LIU FUNCTION#13 M-VARIABLES M=?' /
774: DATA TIT(32) / 'KF=32 YAO-LIU FUNCTION#14 2-VARIABLES M=2' /
775: DATA TIT(33) / 'KF=33 YAO-LIU FUNCTION#15 4-VARIABLES M=4' /
776: DATA TIT(34) / 'KF=34 WOOD FUNCTION : 4-VARIABLES M=4' /
777: DATA TIT(35) / 'KF=35 FENTON-EASON FUNCTION : 2-VARIABLES M=2' /
778: DATA TIT(36) / 'KF=36 HOUGEN FUNCTION : 5-VARIABLES M=5' /
779: DATA TIT(37) / 'KF=37 GIUNTA FUNCTION : 2-VARIABLES M=2' /
780: DATA TIT(38) / 'KF=38 EGGHOLDER FUNCTION : M-VARIABLES M=?' /
781: DATA TIT(39) / 'KF=39 TRID FUNCTION : M-VARIABLES M=?' /
782: DATA TIT(40) / 'KF=40 GRIEWANK FUNCTION : M-VARIABLES M=?' /
783: DATA TIT(41) / 'KF=41 WEIERSTRASS FUNCTION : M-VARIABLES M=?' /
784: DATA TIT(42) / 'KF=42 LEVY-3 FUNCTION : 2-VARIABLES M=2' /
785: DATA TIT(43) / 'KF=43 LEVY-5 FUNCTION : 2-VARIABLES M=2' /
786: DATA TIT(44) / 'KF=44 LEVY-8 FUNCTION : 3-VARIABLES M=3' /
787: DATA TIT(45) / 'KF=45 RASTRIGIN FUNCTION : M-VARIABLES M=?' /
788: DATA TIT(46) / 'KF=46 ACKLEY FUNCTION : M-VARIABLES M=?' /
789: DATA TIT(47) / 'KF=47 MICHALEWICZ FUNCTION : M-VARIABLES M=?' /
790: DATA TIT(48) / 'KF=48 SCHWEFEL FUNCTION : M-VARIABLES M=?' /
791: DATA TIT(49) / 'KF=49 SHUBERT FUNCTION : 2-VARIABLES M=2' /
792: DATA TIT(50) / 'KF=50 DIXON-PRICE FUNCTION : M-VARIABLES M=?' /
793: DATA TIT(51) / 'KF=51 SHEKEL FUNCTION : 4-VARIABLES M=4' /
794: DATA TIT(52) / 'KF=52 PAVIANI FUNCTION : 10-VARIABLES M=10' /
795: DATA TIT(53) / 'KF=53 BRANIN FUNCTION#1 : 2-VARIABLES M=2' /
796: DATA TIT(54) / 'KF=54 BRANIN FUNCTION#2 : 2-VARIABLES M=2' /
797: DATA TIT(55) / 'KF=55 BOHACHEVSKY FUNCTION#1 : 2-VARIABLES M=2' /
798: DATA TIT(56) / 'KF=56 BOHACHEVSKY FUNCTION#2 : 2-VARIABLES M=2' /
799: DATA TIT(57) / 'KF=57 BOHACHEVSKY FUNCTION#3 : 2-VARIABLES M=2' /
800: DATA TIT(58) / 'KF=58 EASOM FUNCTION : 2-VARIABLES M=2' /
801: DATA TIT(59) / 'KF=59 ROSENBROCK FUNCTION : M-VARIABLES M=?' /
802: DATA TIT(60) / 'KF=60 CROSS-LEGGED TABLE FUNCTION:2-VARIABLES M=2' /
803: DATA TIT(61) / 'KF=61 CROSS FUNCTION : 2-VARIABLES M=2' /
804: DATA TIT(62) / 'KF=62 CROSS-IN-TRAY FUNCTION : 2-VARIABLES M=2' /

```

```

805: DATA TIT(63) / 'KF=63 CROWNED CROSS FUNCTION : 2-VARIABLES M=2' /
806: DATA TIT(64) / 'KF=64 TT-HOLDER FUNCTION : 2-VARIABLES M=2' /
807: DATA TIT(65) / 'KF=65 HOLDER-TABLE FUNCTION : 2-VARIABLES M=2' /
808: DATA TIT(66) / 'KF=66 CARROM-TABLE FUNCTION : 2-VARIABLES M=2' /
809: DATA TIT(67) / 'KF=67 PENHOLDER FUNCTION : 2-VARIABLES M=2' /
810: DATA TIT(68) / 'KF=68 BIRD FUNCTION : 2-VARIABLES M=2' /
811: DATA TIT(69) / 'KF=69 CHICHINADZE FUNCTION : 2-VARIABLES M=2' /
812: DATA TIT(70) / 'KF=70 MCCORMICK FUNCTION : 2-VARIABLES M=2' /
813: DATA TIT(71) / 'KF=71 GLANKWAHMDEE FUNCTION : 5-VARIABLES M=5' /
814: DATA TIT(72) / 'KF=72 FLETCHER-POWELL FUNCTION : M-VARIABLES M=?' /
815: DATA TIT(73) / 'KF=73 POWELL FUNCTION: M-VARIABLES M (MULT OF 4)=?' /
816: DATA TIT(74) / 'KF=74 HARTMANN FUNCTION: 3-VARIABLES M=3' /
817: DATA TIT(75) / 'KF=75 COLVILLE FUNCTION: 4-VARIABLES M=4' /
818: C -----
819: DO I=1,75
820: WRITE(*,*) TIT(I)
821: ENDDO
822: WRITE(*,*) '-----'
823: WRITE(*,*) 'FUNCTION CODE [KF] AND NO. OF VARIABLES [M] ?'
824: READ(*,*) KF,M
825: FTIT=TIT(KF) ! STORE THE NAME OF THE CHOSEN FUNCTION IN FTIT
826: RETURN
827: END
828: C -----
829: SUBROUTINE FUNC(X,M,F)
830: C TEST FUNCTIONS FOR GLOBAL OPTIMIZATION PROGRAM
831: IMPLICIT DOUBLE PRECISION (A-H,O-Z)
832: COMMON /RNDM/IU,IV
833: COMMON /KFF/KF,NFCALL,FTIT
834: INTEGER IU,IV
835: DIMENSION X(*)
836: CHARACTER *70 FTIT
837: PI=4.D+00*DATAN(1.D+00) ! DEFINING THE VALUE OF PI
838: NFCALL=NFCALL+1 ! INCREMENT TO NUMBER OF FUNCTION CALLS
839: C KF IS THE CODE OF THE TEST FUNCTION
840: C -----
841: IF (KF.EQ.1) THEN
842: C FUNCTION #1 MIN AT -0.18467 APPROX AT (-8.4666, -10) APPROX
843: F=0.D00
844: DO I=1,M
845: IF (DABS(X(I)).GT.10.D00) THEN
846: CALL RANDOM(RAND)
847: X(I)=(RAND-0.5D00)*20
848: ENDIF
849: ENDDO
850: F=DABS(DCOS(DSQRT(DABS(X(1)**2+X(2)))))**0.5 +0.01*X(1)+.01*X(2)
851: RETURN
852: ENDIF
853: C -----
854: IF (KF.EQ.2) THEN
855: C FUNCTION #2 MIN = -0.199409 APPROX AT (-9.94112, -10) APPROX
856: F=0.D00
857: DO I=1,M
858: IF (DABS(X(I)).GT.10.D00) THEN
859: CALL RANDOM(RAND)
860: X(I)=(RAND-0.5D00)*20
861: ENDIF
862: ENDDO
863: F=DABS(DSIN(DSQRT(DABS(X(1)**2+X(2)))))**0.5 +0.01*X(1)+.01*X(2)
864: RETURN
865: ENDIF
866: C -----
867: IF (KF.EQ.3) THEN
868: C FUNCTION #3 MIN = -1.01983 APPROX AT (-1.98682, -10.00000) APPROX
869: F=0.D00
870: DO I=1,M
871: IF (DABS(X(I)).GT.10.D00) THEN

```

```

872:      CALL RANDOM(RAND)
873:      X(I)=(RAND-0.5D00)*20
874:      ENDIF
875:      ENDDO
876:      F1=DSIN((DCOS(X(1))+DCOS(X(2)))**2)**2
877:      F2=DCOS((DSIN(X(1))+DSIN(X(2)))**2)**2
878:      F=(F1+F2+X(1))**2 ! IS MULTIMODAL
879:      F=F+ 0.01*X(1)+0.1*X(2) ! MAKES UNIMODAL
880:      RETURN
881:      ENDIF
882: C -----
883:      IF (KF.EQ.4) THEN
884: C      FUNCTION #4 MIN = -2.28395 APPROX AT (2.88631, 1.82326) APPROX
885:      F=0.D00
886:      DO I=1,M
887:      IF (DABS(X(I)).GT.10.D00) THEN
888:      CALL RANDOM(RAND)
889:      X(I)=(RAND-0.5D00)*20
890:      ENDIF
891:      ENDDO
892:      F1=DSIN((DCOS(X(1))+DCOS(X(2)))**2)**2
893:      F2=DCOS((DSIN(X(1))+DSIN(X(2)))**2)**2
894:      F3=-DLOG((F1-F2+X(1))**2)
895:      F=F3+0.1D00*(X(1)-1.D00)**2+0.1D00*(X(2)-1.D00)**2
896:      RETURN
897:      ENDIF
898: C -----
899:      IF (KF.EQ.5) THEN
900: C      QUINTIC FUNCTION:GLOBAL MINIMA,EXTREMELY DIFFICULT TO OPTIMIZE
901: C      MIN VALUE = 0 AT PERMUTATION OF (2, 2,..., 2, -1, -1, ..., -1,
902: C      -0.402627941) GIVES MIN F = 0.
903:      F=0.D00
904:      DO I=1,M
905:      IF (DABS(X(I)).GT.10.D00) THEN
906:      CALL RANDOM(RAND)
907:      X(I)=(RAND-0.5D00)*20
908:      ENDIF
909:      ENDDO
910:      CALL QUINTIC(M,F,X)
911:      RETURN
912:      ENDIF
913: C -----
914:      IF (KF.EQ.6) THEN
915: C      NEEDLE-EYE FUNCTION M=>1;
916: C      MIN = 1 IF ALL ABS(X) ARE SMALLER THAN THE EYE
917: C      SMALLER THE VALUE OF ZZ, MORE DIFFICULT TO ENTER THE EYE
918: C      LARGER THE VALUE OF M, MORE DIFFICULT TO FIND THE OPTIMUM
919:      F=0.D00
920:      EYE=0.000001D00
921:      FP=0.D00
922:      DO I=1,M
923:      IF (DABS(X(I)).GT.EYE) THEN
924:      FP=1.D00
925:      F=F+100.D00+DABS(X(I))
926:      ELSE
927:      F=F+1.D00
928:      ENDIF
929:      ENDDO
930:      IF (FP.EQ.0.D00) F=F/M
931:      RETURN
932:      ENDIF
933: C -----
934:      IF (KF.EQ.7) THEN
935: C      ZERO SUM FUNCTION : MIN = 0 AT SUM(X(I))=0
936:      F=0.D00
937:      DO I=1,M
938:      IF (DABS(X(I)).GT.10.D00) THEN

```

```

939:      CALL RANDOM(RAND)
940:      X(I)=(RAND-0.5D00)*20
941:      ENDIF
942:      ENDDO
943:      SUM=0.D00
944:      DO I=1,M
945:      SUM=SUM+X(I)
946:      ENDDO
947:      IF(SUM.NE.0.D00) F=1.D00+(10000*DABS(SUM))**0.5
948:      RETURN
949:      ENDIF
950:  C -----
951:      IF(KF.EQ.8) THEN
952:  C CORANA FUNCTION : MIN = 0 AT (0, 0, 0, 0) APPROX
953:      F=0.D00
954:      DO I=1,M
955:      IF(DABS(X(I)).GT.1000.D00) THEN
956:      CALL RANDOM(RAND)
957:      X(I)=(RAND-0.5D00)*2000
958:      ENDIF
959:      ENDDO
960:      DO J=1,M
961:      IF(J.EQ.1) DJ=1.D00
962:      IF(J.EQ.2) DJ=1000.D00
963:      IF(J.EQ.3) DJ=10.D00
964:      IF(J.EQ.4) DJ=100.D00
965:      ISGNXJ=1
966:      IF(X(J).LT.0.D00) ISGNXJ=-1
967:      ZJ=(DABS(X(J)/0.2D00)+0.49999)*ISGNXJ*0.2D00
968:      ISGNZJ=1
969:      IF(ZJ.LT.0.D00) ISGNZJ=-1
970:      IF(DABS(X(J)-ZJ).LT.0.05D00) THEN
971:      F=F+0.15D00*(ZJ-0.05D00*ISGNZJ)**2 * DJ
972:      ELSE
973:      F=F+DJ*X(J)**2
974:      ENDIF
975:      ENDDO
976:      RETURN
977:      ENDIF
978:  C -----
979:      IF(KF.EQ.9) THEN
980:  C MODIFIED RCOS FUNCTION MIN=-0.179891 AT (-3.196989, 12.52626)APPRX
981:      F=0.D00
982:      IF(X(1).LT.-5.D00 .OR. X(1).GT.10.D00) THEN
983:      CALL RANDOM(RAND)
984:      X(1)=RAND*15.D00 -5.D00
985:      ENDIF
986:      IF(X(2).LT.0.D00 .OR. X(2).GT.15.D00) THEN
987:      CALL RANDOM(RAND)
988:      X(2)=RAND*15.D00
989:      ENDIF
990:      CA=1.D00
991:      CB=5.1/(4*PI**2)
992:      CC=5.D00/PI
993:      CD=6.D00
994:      CE=10.D00
995:      CF=1.0/(8*PI)
996:      F1=CA*(X(2)-CB*X(1)**2+CC*X(1)-CD)**2
997:      F2=CE*(1.D00-CF)*DCOS(X(1))*DCOS(X(2))
998:      F3=DLOG(X(1)**2+X(2)**2+1.D00)
999:      F=-1.0/(F1+F2+F3+CE)
1000:      RETURN
1001:      ENDIF
1002:  C -----
1003:      IF(KF.EQ.10) THEN
1004:  C FREUDENSTEIN ROTH FUNCTION : MIN = 0 AT (5, 4)
1005:      F=0.D00

```

```

1006:      DO I=1,M
1007:      IF (DABS (X(I)) .GT. 10.D00) THEN
1008:      CALL RANDOM(RAND)
1009:      X(I)=(RAND-0.5D00)*20
1010:      ENDIF
1011:      ENDDO
1012:      F1=(-13.D00+X(1)+((5.D00-X(2))*X(2)-2)*X(2))**2
1013:      F2=(-29.D00+X(1)+((X(2)+1.D00)*X(2)-14.D00)*X(2))**2
1014:      F=F1+F2
1015:      RETURN
1016:      ENDIF
1017: C -----
1018:      IF (KF.EQ.11) THEN
1019: C ANNS XOR FUNCTION (PARSOPOULOS, KE, PLAGIANAKOS, VP, MAGOULAS, GD
1020: C AND VRAHATIS, MN "STRETCHING TECHNIQUE FOR OBTAINING GLOBAL
1021: C MINIMIZERS THROUGH PARTICLE SWARM OPTIMIZATION")
1022: C MIN=0.9597588 FOR X=(1, -1, 1, -1, -1, 1, 1, -1, 0.421134) APPROX
1023: C OBTAINED BY DIFFERENTIAL EVOLUTION PROGRAM
1024:      F=0.D00
1025:      DO I=1,M
1026:      IF (DABS (X(I)) .GT. 1.D00) THEN
1027:      CALL RANDOM(RAND)
1028:      X(I)=(RAND-0.5D00)*2
1029:      ENDIF
1030:      ENDDO
1031:      F11=X(7)/(1.D00+DEXP(-X(1)-X(2)-X(5)))
1032:      F12=X(8)/(1.D00+DEXP(-X(3)-X(4)-X(6)))
1033:      F1=(1.D00+DEXP(-F11-F12-X(9)))**(-2)
1034:      F21=X(7)/(1.D00+DEXP(-X(5)))
1035:      F22=X(8)/(1.D00+DEXP(-X(6)))
1036:      F2=(1.D00+DEXP(-F21-F22-X(9)))**(-2)
1037:      F31=X(7)/(1.D00+DEXP(-X(1)-X(5)))
1038:      F32=X(8)/(1.D00+DEXP(-X(3)-X(6)))
1039:      F3=(1.D00-(1.D00+DEXP(-F31-F32-X(9)))**(-1))**2
1040:      F41=X(7)/(1.D00+DEXP(-X(2)-X(5)))
1041:      F42=X(8)/(1.D00+DEXP(-X(4)-X(6)))
1042:      F4=(1.D00-(1.D00+DEXP(-F41-F42-X(9)))**(-1))**2
1043:      F=F1+F2+F3+F4
1044:      RETURN
1045:      ENDIF
1046: C -----
1047:      IF (KF.EQ.12) THEN
1048: C PERM FUNCTION #1 MIN = 0 AT (1, 2, 3, 4)
1049: C BETA => 0. CHANGE IF NEEDED. SMALLER BETA RAISES DIFFICULTY
1050: C FOR BETA=0, EVERY PERMUTED SOLUTION IS A GLOBAL MINIMUM
1051:      BETA=50.D00
1052:      F=0.D00
1053:      DO I=1,M
1054:      IF (DABS (X(I)) .GT.M) THEN
1055:      CALL RANDOM(RAND)
1056:      X(I)=(RAND-0.5D00)*2*M
1057:      ENDIF
1058:      ENDDO
1059:      DO K=1,M
1060:      SUM=0.D00
1061:      DO I=1,M
1062:      SUM=SUM+(I**K+BETA)*((X(I)/I)**K-1.D00)
1063:      ENDDO
1064:      F=F+SUM**2
1065:      ENDDO
1066:      RETURN
1067:      ENDIF
1068: C -----
1069:      IF (KF.EQ.13) THEN
1070: C PERM FUNCTION #2 MIN = 0 AT (1/1, 1/2, 1/3, 1/4,..., 1/M)
1071: C BETA => 0. CHANGE IF NEEDED. SMALLER BETA RAISES DIFFICULTY
1072: C FOR BETA=0, EVERY PERMUTED SOLUTION IS A GLOBAL MINIMUM

```

```

1073:      BETA=10.D00
1074:      DO I=1,M
1075:      IF (DABS(X(I)).GT.1.D00) THEN
1076:      CALL RANDOM(RAND)
1077:      X(I)=(RAND-.5D00)*2
1078:      ENDIF
1079:      SGN=X(I)/DABS(X(I))
1080:      ENDDO
1081:      F=0.D00
1082:      DO K=1,M
1083:      SUM=0.D00
1084:      DO I=1,M
1085:      SUM=SUM+(I+BETA)*(X(I)**K-(1.D00/I)**K)
1086:      ENDDO
1087:      F=F+SUM**2
1088:      ENDDO
1089:      RETURN
1090:      ENDIF
1091: C -----
1092:      IF (KF.EQ.14) THEN
1093: C      POWER SUM FUNCTION; MIN = 0 AT PERM(1,2,2,3) FOR B=(8,18,44,114)
1094: C      0 =< X <=4
1095:      F=0.D00
1096:      DO I=1,M
1097: C      ANY PERMUTATION OF (1,2,2,3) WILL GIVE MIN = ZERO
1098:      IF (X(I).LT.0.D00 .OR. X(I).GT.4.D00) THEN
1099:      CALL RANDOM(RAND)
1100:      X(I)=RAND*4
1101:      ENDIF
1102:      ENDDO
1103:      DO K=1,M
1104:      SUM=0.D00
1105:      DO I=1,M
1106:      SUM=SUM+X(I)**K
1107:      ENDDO
1108:      IF (K.EQ.1) B=8.D00
1109:      IF (K.EQ.2) B=18.D00
1110:      IF (K.EQ.3) B=44.D00
1111:      IF (K.EQ.4) B=114.D00
1112:      F=F+(SUM-B)**2
1113:      ENDDO
1114:      RETURN
1115:      ENDIF
1116: C -----
1117:      IF (KF.EQ.15) THEN
1118: C      GOLDSTEIN PRICE FUNCTION : MIN VALUE = 3 AT (0, -1)
1119:      F=0.D00
1120:      DO I=1,M
1121:      IF (DABS(X(I)).GT.10.D00) THEN
1122:      CALL RANDOM(RAND)
1123:      X(I)=(RAND-.5D00)*20
1124:      ENDIF
1125:      ENDDO
1126:      F11=(X(1)+X(2)+1.D00)**2
1127:      F12=(19.D00-14*X(1)+ 3*X(1)**2-14*X(2)+ 6*X(1)*X(2)+ 3*X(2)**2)
1128:      F1=1.00+F11*F12
1129:      F21=(2*X(1)-3*X(2))**2
1130:      F22=(18.D00-32*X(1)+12*X(1)**2+48*X(2)-36*X(1)*X(2)+27*X(2)**2)
1131:      F2=30.D00+F21*F22
1132:      F= (F1*F2)
1133:      RETURN
1134:      ENDIF
1135: C -----
1136:      IF (KF.EQ.16) THEN
1137: C      BUKIN'S 6TH FUNCTION MIN = 0 FOR (-10, 1)
1138: C      -15 .LE. X(1) .LE. -5 AND -3 .LE. X(2) .LE. 3
1139:      IF (X(1).LT. -15.D00 .OR. X(1).GT. -5.D00) THEN

```

```

1140:      CALL RANDOM(RAND)
1141:      X(1)=- (RAND*10+5.D00)
1142:      ENDIF
1143:      IF (DABS(X(2)) .GT. 3.D00) THEN
1144:      CALL RANDOM(RAND)
1145:      X(2)= (RAND-.5D00)*6
1146:      ENDIF
1147:      F=100.D0*DSQRT(DABS(X(2)-0.01D0*X(1)**2))+ 0.01D0*DABS(X(1)+10.D0)
1148:      RETURN
1149:      ENDIF
1150: C -----
1151:      IF (KF.EQ.17) THEN
1152: C      NEW N#8 FUNCTION (MULTIPLE GLOBAL MINIMA)
1153: C      MIN VALUE = -1 AT (AROUND .7 AROUND, 0.785 APPROX)
1154:      F=0.D00
1155:      DO I=1,M
1156:      IF (X(I) .LT. 0.5D00 .OR. X(I) .GT. 1.D00) THEN
1157:      CALL RANDOM(RAND)
1158:      X(I)=RAND/2.D00
1159:      ENDIF
1160:      ENDDO
1161:      F=-DEXP(-DABS(DLOG(.001D00+DABS((DSIN(X(1)+X(2))+DSIN(X(1)-X(2)))+
1162: & (DCOS(X(1)+X(2))*DCOS(X(1)-X(2))+.001)**2)+
1163: & .01D00*(X(2)-X(1)**2)))
1164:      RETURN
1165:      ENDIF
1166: C -----
1167:      IF (KF.EQ.18) THEN
1168: C      DEFLECTED CORRUGATED SPRING FUNCTION
1169: C      MIN VALUE = -1 AT (5, 5, ..., 5) FOR ANY K AND ALPHA=5; M VARIABLE
1170:      CALL DCS(M,F,X)
1171:      RETURN
1172:      ENDIF
1173: C -----
1174:      IF (KF.EQ.19) THEN
1175: C      FACTORIAL FUNCTION, MIN =0 AT X=(1,2,3,...,M)
1176:      CALL FACTOR1(M,F,X)
1177:      RETURN
1178:      ENDIF
1179: C -----
1180:      IF (KF.EQ.20) THEN
1181: C      DECANOMIAL FUNCTION, MIN =0 AT X=(2, -3)
1182:      DO I=1,M
1183:      IF (DABS(X(I)) .GT. 4.D00) THEN
1184:      CALL RANDOM(RAND)
1185:      X(I)= (RAND-0.5D00)*8
1186:      ENDIF
1187:      ENDDO
1188:      CALL DECANOM(M,F,X)
1189:      RETURN
1190:      ENDIF
1191: C -----
1192:      IF (KF.EQ.21) THEN
1193: C      JUDGE'S FUNCTION F(0.864, 1.23) = 16.0817; M=2
1194:      CALL JUDGE(M,X,F)
1195:      RETURN
1196:      ENDIF
1197: C -----
1198:      IF (KF.EQ.22) THEN
1199: C      DODECAL FUNCTION
1200:      CALL DODECAL(M,F,X)
1201:      RETURN
1202:      ENDIF
1203: C -----
1204:      IF (KF.EQ.23) THEN
1205: C      WHEN X(1)*X(2)=X(1)*X(2) ? M=2
1206:      CALL SEQP(M,F,X)

```

```

1207:      RETURN
1208:      ENDIF
1209: C -----
1210:      IF (KF.EQ.24) THEN
1211: C      WHEN ARITHMETIC MEAN = GEOMETRIC MEAN ? : M =>1
1212:      CALL AMGM(M,F,X)
1213:      RETURN
1214:      ENDIF
1215: C -----
1216:      IF (KF.EQ.25) THEN
1217: C      M =>2
1218:      CALL FUNCT2(M,F,X)
1219:      RETURN
1220:      ENDIF
1221: C -----
1222:      IF (KF.EQ.26) THEN
1223: C      M =>2
1224:      CALL FUNCT3(M,F,X)
1225:      RETURN
1226:      ENDIF
1227: C -----
1228:      IF (KF.EQ.27) THEN
1229: C      M =>2
1230:      CALL FUNCT4(M,F,X)
1231:      RETURN
1232:      ENDIF
1233: C -----
1234:      IF (KF.EQ.28) THEN
1235: C      M =>2
1236:      CALL FUNCT6(M,F,X)
1237:      RETURN
1238:      ENDIF
1239: C -----
1240:      IF (KF.EQ.29) THEN
1241: C      M =>2
1242:      CALL FUNCT7(M,F,X)
1243:      RETURN
1244:      ENDIF
1245: C -----
1246:      IF (KF.EQ.30) THEN
1247: C      M =>2
1248:      CALL FUNCT12(M,F,X)
1249:      RETURN
1250:      ENDIF
1251: C -----
1252:      IF (KF.EQ.31) THEN
1253: C      M =>2
1254:      CALL FUNCT13(M,F,X)
1255:      RETURN
1256:      ENDIF
1257: C -----
1258:      IF (KF.EQ.32) THEN
1259: C      M =2
1260:      CALL FUNCT14(M,F,X)
1261:      RETURN
1262:      ENDIF
1263: C -----
1264:      IF (KF.EQ.33) THEN
1265: C      M =4
1266:      CALL FUNCT15(M,F,X)
1267:      RETURN
1268:      ENDIF
1269: C -----
1270:      IF (KF.EQ.34) THEN
1271: C      WOOD FUNCTION : F MIN : M=4
1272:      CALL WOOD(M,X,F)
1273:      RETURN

```



```

1274:      ENDIF
1275: C -----
1276:      IF (KF.EQ.35) THEN
1277: C      FENTON & EASON FUNCTION :      : M=2
1278:      CALL FENTONEASON (M, X, F)
1279:      RETURN
1280:      ENDIF
1281: C -----
1282:      IF (KF.EQ.36) THEN
1283: C      HOUGEN FUNCTION 5 VARIABLES : M =3
1284:      CALL HOUGEN (X, M, F)
1285:      RETURN
1286:      ENDIF
1287: C -----
1288:      IF (KF.EQ.37) THEN
1289: C      GIUNTA FUNCTION 2 VARIABLES :M =2
1290:      CALL GIUNTA (M, X, F)
1291:      RETURN
1292:      ENDIF
1293: C -----
1294:      IF (KF.EQ.38) THEN
1295: C      EGGHOLDER FUNCTION M VARIABLES
1296:      CALL EGGHOLD (M, X, F)
1297:      RETURN
1298:      ENDIF
1299: C -----
1300:      IF (KF.EQ.39) THEN
1301: C      TRID FUNCTION M VARIABLES
1302:      CALL TRID (M, X, F)
1303:      RETURN
1304:      ENDIF
1305: C -----
1306:      IF (KF.EQ.40) THEN
1307: C      GRIEWANK FUNCTION M VARIABLES
1308:      CALL GRIEWANK (M, X, F)
1309:      RETURN
1310:      ENDIF
1311: C -----
1312:      IF (KF.EQ.41) THEN
1313: C      WEIERSTRASS FUNCTION M VARIABLES
1314:      CALL WEIERSTRASS (M, X, F)
1315:      RETURN
1316:      ENDIF
1317: C -----
1318:      IF (KF.EQ.42) THEN
1319: C      LEVY-3 FUNCTION 2 VARIABLES
1320:      CALL LEVY3 (M, X, F)
1321:      RETURN
1322:      ENDIF
1323: C -----
1324:      IF (KF.EQ.43) THEN
1325: C      LEVY-5 FUNCTION 2 VARIABLES
1326:      CALL LEVY5 (M, X, F)
1327:      RETURN
1328:      ENDIF
1329: C -----
1330:      IF (KF.EQ.44) THEN
1331: C      LEVY-8 FUNCTION 3 VARIABLES
1332:      CALL LEVY8 (M, X, F)
1333:      RETURN
1334:      ENDIF
1335: C -----
1336:      IF (KF.EQ.45) THEN
1337: C      RASTRIGIN FUNCTION M VARIABLES
1338:      CALL RASTRIGIN (M, X, F)
1339:      RETURN
1340:      ENDIF

```

```

1341: C -----
1342: IF (KF.EQ.46) THEN
1343: C ACKLEY FUNCTION M VARIABLES
1344: CALL ACKLEY (M, X, F)
1345: RETURN
1346: ENDF
1347: C -----
1348: IF (KF.EQ.47) THEN
1349: C MICHALEWICZ FUNCTION M VARIABLES
1350: CALL MICHALEWICZ (M, X, F)
1351: RETURN
1352: ENDF
1353: C -----
1354: IF (KF.EQ.48) THEN
1355: C SCHWEFEL FUNCTION M VARIABLES
1356: CALL SCHWEFEL (M, X, F)
1357: RETURN
1358: ENDF
1359: C -----
1360: IF (KF.EQ.49) THEN
1361: C SHUBERT FUNCTION 2 VARIABLES
1362: CALL SHUBERT (M, X, F)
1363: RETURN
1364: ENDF
1365: C -----
1366: IF (KF.EQ.50) THEN
1367: C DIXON AND PRICE FUNCTION M VARIABLES
1368: CALL DIXPRICE (M, X, F)
1369: RETURN
1370: ENDF
1371: C -----
1372: IF (KF.EQ.51) THEN
1373: C SHEKEL FUNCTION 4 VARIABLES
1374: CALL SHEKEL (M, X, F)
1375: RETURN
1376: ENDF
1377: C -----
1378: IF (KF.EQ.52) THEN
1379: C PAVIANI FUNCTION 10 VARIABLES
1380: CALL PAVIANI (M, X, F)
1381: RETURN
1382: ENDF
1383: C -----
1384: IF (KF.EQ.53) THEN
1385: C BRANIN FUNCTION#1 2 VARIABLES
1386: CALL BRANIN1 (M, X, F)
1387: RETURN
1388: ENDF
1389: C -----
1390: IF (KF.EQ.54) THEN
1391: C BRANIN FUNCTION#2 2 VARIABLES
1392: CALL BRANIN2 (M, X, F)
1393: RETURN
1394: ENDF
1395: C -----
1396: IF (KF.EQ.55) THEN
1397: C BOHACHEVSKY FUNCTION#1 2 VARIABLES
1398: CALL BOHACHEVSKY1 (M, X, F)
1399: RETURN
1400: ENDF
1401: C -----
1402: IF (KF.EQ.56) THEN
1403: C BOHACHEVSKY FUNCTION#2 2 VARIABLES
1404: CALL BOHACHEVSKY2 (M, X, F)
1405: RETURN
1406: ENDF
1407: C -----

```

```

1408:      IF (KF.EQ.57) THEN
1409: C      BOHACHEVSKY FUNCTION#3 2 VARIABLES
1410:      CALL BOHACHEVSKY3 (M,X,F)
1411:      RETURN
1412:      ENDIF
1413: C      -----
1414:      IF (KF.EQ.58) THEN
1415: C      EASOM FUNCTION#3 2 VARIABLES
1416:      CALL EASOM (M,X,F)
1417:      RETURN
1418:      ENDIF
1419: C      -----
1420:      IF (KF.EQ.59) THEN
1421: C      ROSENBROCK FUNCTION M VARIABLES
1422:      CALL ROSENBROCK (M,X,F)
1423:      RETURN
1424:      ENDIF
1425: C      -----
1426:      IF (KF.EQ.60) THEN
1427: C      CROSS-LEGGED TABLE FUNCTION : 2 VARIABLES
1428:      CALL CROSSLEG (M,X,F)
1429:      RETURN
1430:      ENDIF
1431: C      -----
1432:      IF (KF.EQ.61) THEN
1433: C      CROSS FUNCTION : 2 VARIABLES
1434:      CALL CROSS (M,X,F)
1435:      RETURN
1436:      ENDIF
1437: C      -----
1438:      IF (KF.EQ.62) THEN
1439: C      CROSS-IN-TRAY FUNCTION : 2 VARIABLES
1440:      CALL CROSSINTRAY (M,X,F)
1441:      RETURN
1442:      ENDIF
1443: C      -----
1444:      IF (KF.EQ.63) THEN
1445: C      CROWNED-CROSS FUNCTION : 2 VARIABLES
1446:      CALL CROWNEDCROSS (M,X,F)
1447:      RETURN
1448:      ENDIF
1449: C      -----
1450:      IF (KF.EQ.64) THEN
1451: C      TT-HOLDER FUNCTION : 2 VARIABLES, MIN F ([+/-]1.5706, 0) = -10.8723
1452:      CALL TTHOLDER (M,X,F)
1453:      RETURN
1454:      ENDIF
1455: C      -----
1456:      IF (KF.EQ.65) THEN
1457: C      HOLDER-TABLE FUNCTION : 2 VARIABLES
1458: C      MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -26.92034 APPROX
1459:      CALL HOLDERTABLE (M,X,F)
1460:      RETURN
1461:      ENDIF
1462: C      -----
1463:      IF (KF.EQ.66) THEN
1464: C      CARROM-TABLE FUNCTION : 2 VARIABLES
1465: C      MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -24.15682 APPROX
1466:      CALL CARROMTABLE (M,X,F)
1467:      RETURN
1468:      ENDIF
1469: C      -----
1470:      IF (KF.EQ.67) THEN
1471: C      PEN-HOLDER FUNCTION : 2 VARIABLES
1472: C      MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -0.963535 APPROX
1473:      CALL PENHOLDER (M,X,F)
1474:      RETURN

```

[illegible]

```

1542:      C(I)=ALPHA
1543:      R2=R2+(X(I)-C(I))**2
1544:      ENDDO
1545:      R=DSQRT(R2)
1546:      F=-DCOS(K*R)+0.1D00*R2
1547:      RETURN
1548:      END
1549: C -----
1550:      SUBROUTINE QUINTIC(M,F,X)
1551: C      QUINTIC FUNCTION: GLOBAL MINIMA,EXTREMELY DIFFICULT TO OPTIMIZE
1552: C      MIN VALUE = 0 AT PERM( -1, -0.402627941, 2)
1553:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1554:      DIMENSION X(*)
1555:      F=0.D00
1556:      DO I=1,M
1557:      F=F+DABS(X(I)**5-3*X(I)**4+4*X(I)**3+2*X(I)**2-10*X(I)-4.D00)
1558:      ENDDO
1559:      F=1000*F
1560:      RETURN
1561:      END
1562: C -----
1563:      SUBROUTINE FACTOR1(M,F,X)
1564: C      FACTORIAL FUNCTION; MIN (1, 2, 3, ....., M) = 0
1565: C      FACT = FACTORIAL(M) = 1 X 2 X 3 X 4 X .... X M
1566: C      FIND X(I), I=1,2,...,M SUCH THAT THEIR PRODUCT IS EQUAL TO FACT.
1567: C      LARGER THE VALUE OF M (=>8) OR SO, HARDER IS THE PROBLEM
1568:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1569:      DIMENSION X(*)
1570:      F=0.D00
1571:      FACT=1.D00
1572:      P=1.D00
1573:      DO I=1,M
1574:      FACT=FACT*I
1575:      P=P*X(I)
1576:      F=F+DABS(P-FACT)**2
1577:      ENDDO
1578:      RETURN
1579:      END
1580: C -----
1581:      SUBROUTINE DECANOM(M,F,X)
1582: C      DECANOMIAL FUNCTION; MIN (2, -3) = 0
1583:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1584:      DIMENSION X(*)
1585:      F1= DABS(X(1)**10-20*X(1)**9+180*X(1)**8-960*X(1)**7+
1586: & 3360*X(1)**6-8064*X(1)**5+13340*X(1)**4-15360*X(1)**3+
1587: & 11520*X(1)**2-5120*X(1)+2624.D00)
1588:      F2= DABS(X(2)**4+12*X(2)**3+54*X(2)**2+108*X(2)+81.D00)
1589:      F=0.001D00*(F1+F2)**2
1590:      RETURN
1591:      END
1592: C -----
1593:      SUBROUTINE JUDGE(M,X,F)
1594:      PARAMETER (N=20)
1595: C      THIS SUBROUTINE IS FROM THE EXAMPLE IN JUDGE ET AL., THE THEORY
1596: C      AND PRACTICE OF ECONOMETRICS, 2ND ED., PP. 956-7. THERE ARE TWO
1597: C      OPTIMA: F(0.86479,1.2357)=16.0817307 (WHICH IS THE GLOBAL MINIMUM)
1598: C      AND F(2.35,-0.319)=20.9805 (WHICH IS LOCAL). ADAPTED FROM BILL
1599: C      GOFFE'S SIMMAN (SIMULATED ANNEALING) PROGRAM
1600:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1601:      DIMENSION Y(N), X2(N), X3(N), X(*)
1602:      DATA (Y(I),I=1,N)/4.284,4.149,3.877,0.533,2.211,2.389,2.145,
1603: & 3.231,1.998,1.379,2.106,1.428,1.011,2.179,2.858,1.388,1.651,
1604: & 1.593,1.046,2.152/
1605:      DATA (X2(I),I=1,N)/.286,.973,.384,.276,.973,.543,.957,.948,.543,
1606: & .797,.936,.889,.006,.828,.399,.617,.939,.784,.072,.889/
1607:      DATA (X3(I),I=1,N)/.645,.585,.310,.058,.455,.779,.259,.202,.028,
1608: & .099,.142,.296,.175,.180,.842,.039,.103,.620,.158,.704/

```

```

1609:
1610:      F=0.D00
1611:      DO I=1,N
1612:      F=F+(X(1) + X(2)*X2(I) + (X(2)**2)*X3(I) - Y(I))**2
1613:      ENDDO
1614:      RETURN
1615:      END
1616: C -----
1617:      SUBROUTINE DODECAL(M,F,X)
1618:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1619:      DIMENSION X(*)
1620: C      DODECAL POLYNOMIAL MIN F(1,2,3)=0
1621:      DO I=1,M
1622:      IF (DABS(X(I)) .GT. 5.D0) THEN
1623:      CALL RANDOM(RAND)
1624:      X(I)=(RAND-0.5D00)*10
1625:      ENDIF
1626:      ENDDO
1627:      F=0.D00
1628:      F1=2*X(1)**3+5*X(1)*X(2)+4*X(3)-2*X(1)**2*X(3)-18.D00
1629:      F2=X(1)+X(2)**3+X(1)*X(2)**2+X(1)*X(3)**2-22.D00
1630:      F3=8*X(1)**2+2*X(2)*X(3)+2*X(2)**2+3*X(2)**3-52.D00
1631:      F=(F1*F3*F2**2+F1*F2*F3**2+F2**2+(X(1)+X(2)-X(3))**2)**2
1632:      RETURN
1633:      END
1634: C -----
1635:      SUBROUTINE SEQP(M,F,X)
1636:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1637:      DIMENSION X(*)
1638: C      FOR WHAT VALUES X(1)+X(2)=X(1)*X(2) ? ANSWER: FOR (0,0) AND (2,2)
1639: C      WHILE X(1), X(2) ARE INTEGERS.
1640:      X(1)=INT(X(1)) ! X(1) CONVERTED TO INTEGER
1641:      X(2)=INT(X(2)) ! X(2) CONVERTED TO INTEGER
1642:
1643:      F1=X(1)+X(2)
1644:      F2=X(1)*X(2)
1645:      F=(F1-F2)**2 ! TURN ALIVE THIS XOR
1646: C      F=DABS(F1-F2) ! TURN ALIVE THIS - BUT NOT BOTH -----
1647:      RETURN
1648:      END
1649: C -----
1650:      SUBROUTINE AMGM(M,F,X)
1651:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1652:      DIMENSION X(*)
1653: C      FOR WHAT VALUES ARITHMETIC MEAN = GEOMETRIC MEAN ? THE ANSWER IS:
1654: C      IF X(1)=X(2)=...=X(M) AND ALL X ARE NON-NEGATIVE
1655: C      TAKE ONLY THE ABSOLUTE VALUES OF X
1656:      SUM=0.D00
1657:      DO I=1,M
1658:      X(I)=DABS(X(I))
1659:      ENDDO
1660: C      SET SUM = SOME POSITIVE NUMBER. THIS MAKES THE FUNCTION UNIMODAL
1661:      SUM= 100.D00 ! TURNED ALIVE FOR UNIQUE MINIMUM AND SET SUM TO
1662: C      SOME POSITIVE NUMBER. HERE IT IS 100; IT COULD BE ANYTHING ELSE.
1663:      F1=0.D00
1664:      F2=1.D00
1665:      DO I=1,M
1666:      F1=F1+X(I)
1667:      F2=F2*X(I)
1668:      ENDDO
1669:      XSUM=F1
1670:      F1=F1/M ! SUM DIVIDED BY M = ARITHMETIC MEAN
1671:      F2=F2**(1.D00/M) ! MTH ROOT OF THE PRODUCT = GEOMETRIC MEAN
1672:      F=(F1-F2)**2
1673:      IF (SUM.GT.0.D00) F=F+(SUM-XSUM)**2
1674:      RETURN
1675:      END

```

```

1676: C -----
1677: SUBROUTINE FUNCT2 (M,F,X)
1678: C REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1679: C IN FOGEL, L.J., ANGELIN, P.J. AND BACK, T. (ED) PROCEEDINGS OF THE
1680: C FIFTH ANNUAL CONFERENCE ON EVOLUTIONARY PROGRAMMING, PP. 451-460,
1681: C MIT PRESS, CAMBRIDGE, MASS.
1682: C MIN F (0, 0, ..., 0) = 0
1683: IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1684: DIMENSION X(*)
1685: F=0.D00
1686: F1=1.D00
1687: DO I=1,M
1688: IF (DABS (X(I)) .GT.10.D00) THEN
1689: CALL RANDOM(RAND)
1690: X(I)=(RAND-.5D00)*20
1691: ENDIF
1692: ENDDO
1693: DO I=1,M
1694: F=F+DABS (X(I))
1695: F1=F1*DABS (X(I))
1696: ENDDO
1697: F=F+F1
1698: RETURN
1699: END
1700: C -----
1701: SUBROUTINE FUNCT3 (M,F,X)
1702: C REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1703: C MIN F (0, 0, ..., 0) = 0
1704: IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1705: DIMENSION X(*)
1706: F=0.D00
1707: F1=0.D00
1708: DO I=1,M
1709: IF (DABS (X(I)) .GT.100.D00) THEN
1710: CALL RANDOM(RAND)
1711: X(I)=(RAND-.5D00)*200
1712: ENDIF
1713: ENDDO
1714: DO I=1,M
1715: F1=0.D00
1716: DO J=1,I
1717: F1=F1+X(J)**2
1718: ENDDO
1719: F=F+F1
1720: ENDDO
1721: RETURN
1722: END
1723: C -----
1724: SUBROUTINE FUNCT4 (M,F,X)
1725: C REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1726: C MIN F (0, 0, ..., 0) = 0
1727: IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1728: DIMENSION X(*)
1729: F=0.D00
1730: DO I=1,M
1731: IF (X(I) .LT.0.D00 .OR. X(I) .GE.M) THEN
1732: CALL RANDOM(RAND)
1733: X(I)=RAND*2*M
1734: ENDIF
1735: ENDDO
1736: C FIND MAX(X(I))=MAX(ABS(X(I))) NOTE: HERE X(I) CAN BE ONLY POSITIVE
1737: XMAX=X(1)
1738: DO I=1,M
1739: IF (XMAX .LT. X(I)) XMAX=X(I)
1740: ENDDO
1741: F=XMAX
1742: RETURN

```

```

1743:      END
1744: C      -----
1745:      SUBROUTINE FUNCT6 (M,F,X)
1746: C      REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1747: C      MIN F (-.5, -.5, ..., -.5) = 0
1748:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1749:      DIMENSION X(*)
1750:      F=0.D00
1751:      DO I=1,M
1752:      IF (DABS (X (I)) .GT.100.D00) THEN
1753:      CALL RANDOM(RAND)
1754:      X (I) = (RAND-.5D00) *200
1755:      ENDIF
1756:      ENDDO
1757:      DO I=1,M
1758:      F=F+ (X (I) +0.5D00) **2
1759:      ENDDO
1760:      RETURN
1761:      END
1762: C      -----
1763:      SUBROUTINE FUNCT7 (M,F,X)
1764: C      REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1765: C      MIN F (0, 0, ..., 0) = 0
1766:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1767:      COMMON /RNDM/IU,IV
1768:      INTEGER IU,IV
1769:      DIMENSION X(*)
1770:      F=0.D00
1771:      DO I=1,M
1772:      IF (DABS (X (I)) .GT.1.28D00) THEN
1773:      CALL RANDOM(RAND)
1774:      X (I) = (RAND-0.5D00) *2.56D00
1775:      ENDIF
1776:      ENDDO
1777:      DO I=1,M
1778:      CALL RANDOM(RAND)
1779:      F=F+ (I*X (I) **4)
1780:      ENDDO
1781:      CALL RANDOM(RAND)
1782:      F=F+RAND
1783:      RETURN
1784:      END
1785: C      -----
1786:      SUBROUTINE FUNCT12 (M,F,X)
1787: C      REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1788:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1789:      DIMENSION X(100),Y(100)
1790:      DATA A,B,C /10.D00,100.D00,4.D00/
1791:      PI=4.D00*DATAN (1.D00)
1792:      F=0.D00
1793: C      MIN F (-1, -1, -1, ..., -1) = 0
1794: C      X(I)=-1.D00 ! TO CHECK, TURN IT ALIVE
1795:      DO I=1,M
1796:      IF (DABS (X (I)) .GT.50.D00) THEN
1797:      CALL RANDOM(RAND)
1798:      X (I) = (RAND-0.5D00) *100.D00
1799:      ENDIF
1800:      ENDDO
1801:      F1=0.D00
1802:      DO I=1,M
1803:      XX=DABS (X (I))
1804:      U=0.D00
1805:      IF (XX.GT.A) U=B* (XX-A) **C
1806:      F1=F1+U
1807:      ENDDO
1808:      F2=0.D00
1809:      DO I=1,M-1

```



```

1810:      Y(I)=1.D00+.25D00*(X(I)+1.D00)
1811:      F2=F2+ (Y(I)-1.D00)**2 * (1.D00+10.D00*(DSIN(PI*X(I+1))**2))
1812:      ENDDO
1813:      Y(M)=1.D00+.25D00*(X(M)+1.D00)
1814:      F3=(Y(M)-1.D00)**2
1815:      Y(1)=1.D00+.25D00*(X(1)+1.D00)
1816:      F4=10.D00*(DSIN(PI*Y(1))**2)
1817:      F=(PI/M)*(F4+F2+F3)+F1
1818:      RETURN
1819:      END
1820: C -----
1821:      SUBROUTINE FUNCT13(M,F,X)
1822: C      REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1823:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1824:      DIMENSION X(100)
1825:      DATA A,B,C /5.D00,100.D00,4.D00/
1826:      PI=4*DATAN(1.D00)
1827:      F=0.D00
1828: C      MIN F (1, 1, 1, ..., 4.7544 APPROX) = -1.15044 APPROX
1829: C      X(I)=1.D00 ! TO CHECK, TURN IT ALIVE
1830: C      X(M)=-4.7544 ! TO CHECK, TURN IT ALIVE
1831:      DO I=1,M
1832:      IF (DABS(X(I)).GT.50.D00) THEN
1833:      CALL RANDOM(RAND)
1834:      X(I)=(RAND-.5D00)*100.D00
1835:      ENDIF
1836:      ENDDO
1837:      F1=0.D00
1838:      DO I=1,M
1839:      XX=DABS(X(I))
1840:      U=0.D00
1841:      IF (XX.GT.A) U=B*(XX-A)**C
1842:      F1=F1+U
1843:      ENDDO
1844:      F2=0.D00
1845:      DO I=1,M-1
1846:      F2=F2+ (X(I)-1.D00)**2 * (1.D00+(DSIN(3*PI*X(I+1))**2))
1847:      ENDDO
1848:      F3=(X(M)-1.D00)* (1.D00+(DSIN(2*PI*X(M))**2))
1849:      F4=(DSIN(3*PI*X(1))**2)
1850:      F=0.1*(F4+F2+F3)+F1
1851:      RETURN
1852:      END
1853: C -----
1854:      SUBROUTINE FUNCT14(M,F,X)
1855: C      REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1856: C      MIN F (-31.98, 31.98) = 0.998
1857:      PARAMETER (N=25,NN=2)
1858:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1859:      DIMENSION X(2), A(NN,N)
1860:      DATA (A(1,J),J=1,N) /-32.D00,-16.D00,0.D00,16.D00,32.D00,-32.D00,
1861:      & -16.D00,0.D00,16.D00,32.D00,-32.D00,-16.D00,0.D00,16.D00,32.D00,
1862:      & -32.D0,-16.D0,0.D0,16.D0,32.D0,-32.D0,-16.D0,0.D0,16.D0,32.D0/
1863:      DATA (A(2,J),J=1,N) /-32.D00,-32.D00,-32.D00,-32.D00,-32.D00,
1864:      & -16.D00,-16.D00,-16.D00,-16.D00,-16.D00,0.D00,0.D00,0.D00,0.D00,
1865:      & 0.D00,16.D00,16.D00,16.D00,16.D00,16.D00,16.D00,32.D00,32.D00,
1866:      & 32.D00,32.D00,32.D00/
1867:
1868:      F=0.D00
1869:      DO I=1,M
1870:      IF (DABS(X(I)).GT.100.D00) THEN
1871:      CALL RANDOM(RAND)
1872:      X(I)=(RAND-.5D00)*200.D00
1873:      ENDIF
1874:      ENDDO
1875:      F1=0.D00
1876:      DO J=1,N

```

```

1877:      F2=0.D00
1878:      DO I=1,2
1879:      F2=F2+(X(I)-A(I,J))**6
1880:      ENDDO
1881:      F2=1.D00/(J+F2)
1882:      F1=F1+F2
1883:      ENDDO
1884:      F=1.D00/(0.002D00+F1)
1885:      RETURN
1886:      END
1887: C -----
1888:      SUBROUTINE FUNCT15(M,F,X)
1889: C      REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
1890: C      MIN F(.19, .19, .12, .14) = 0.3075
1891:      PARAMETER (N=11)
1892:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1893:      DIMENSION X(*), A(N), B(N)
1894:      DATA (A(I), I=1, N) /.1957D00, .1947D00, .1735D00, .16D00, .0844D00,
1895: & .0627D00, .0456D00, .0342D00, .0323D00, .0235D00, .0246D00/
1896:      DATA (B(I), I=1, N) /0.25D00, 0.5D00, 1.D00, 2.D00, 4.D00, 6.D00, 8.D00,
1897: & 10.D00, 12.D00, 14.D00, 16.D00/
1898:      DO I=1, N
1899:      B(I)=1.D00/B(I)
1900:      ENDDO
1901:      F=0.D00
1902:      DO I=1, M
1903:      IF (DABS(X(I)) .GT. 5.D00) THEN
1904:      CALL RANDOM(RAND)
1905:      X(I)=(RAND-.5D00)*10.D00
1906:      ENDIF
1907:      ENDDO
1908:      DO I=1, N
1909:      F1=X(1)*(B(I)**2+B(I)*X(2))
1910:      F2=B(I)**2+B(I)*X(3)+X(4)
1911:      F=F+(A(I)-F1/F2)**2
1912:      ENDDO
1913:      F=F*1000
1914:      RETURN
1915:      END
1916: C -----
1917:      SUBROUTINE LINPROG1(M,F,X)
1918: C      LINEAR PROGRAMMING : MINIMIZATION PROBLEM
1919: C      IN THIS PROBLEM : M = NO. OF DECISION VARIABLES = 2
1920: C      MIN F (2.390, 2.033) = -19.7253 APPROX
1921: C      MIN F = OVER J=1, M : DO SUM(A(1,J)*X(J)) SUBJECT TO CONSTRAINTS
1922: C      OVER J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=2
1923: C      . . .
1924: C      OVER J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=N
1925: C      ALL X(I) => 0
1926:      PARAMETER (N=3) ! N IS THE NO. OF CONSTRAINTS + 1
1927:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1928:      DIMENSION X(*), A(20,10), C(20), FF(20)
1929:      DATA (A(1,J), J=1, 2), C(1)/4.D0, 5.D0, 0.0D0/!COEFF OF OBJ FUNCTION
1930:      DATA (A(2,J), J=1, 2), C(2)/10.D0, 3.D0, 30D0/!COEFF OF 1ST CONSTRAINT
1931:      DATA (A(3,J), J=1, 2), C(3)/6.D0, 20.D0, 55.D0/!COEFF OF 2ND CONSTRAINT
1932: C -----
1933: C      USING ONLY NON-NEGATIVE VALUES OF X(I)
1934:      DO I=1, M
1935:      X(I)=DABS(X(I))
1936:      ENDDO
1937: C      EVALUATION OF OBJ FUNCTION AND CONSTRAINTS
1938:      DO I=1, N
1939:      FF(I)=0.D00
1940:      DO J=1, M
1941:      FF(I)=FF(I)+A(I,J)*X(J)
1942:      ENDDO
1943:      ENDDO

```

```

1944:      F=-FF(1) ! CHANGE OF SIGN FOR MINIMIZATION
1945: C      CHECK FOR SATISFYING OR VIOLATING THE CONSTRAINTS
1946:      DO I=2,N
1947:      FF(I)=FF(I)-C(I) ! SLACK
1948: C      PENALTY FOR CROSSING LIMITS
1949:      IF (FF(I) .GT. 0) F=F+(10+FF(I))**2
1950:      ENDDO
1951:      RETURN
1952:      END
1953: C      -----
1954:      SUBROUTINE LINPROG2(M,F,X)
1955: C      LINEAR PROGRAMMING : MINIMIZATION PROBLEM
1956: C      IN THIS PROBLEM : M = NO. OF DECISION VARIABLES = 3
1957: C      MIN F (250, 625, 0) = -3250
1958: C      MIN F = OVER J=1, M : DO SUM(A(1,J)*X(J)) SUBJECT TO CONSTRAINTS
1959: C      OVER J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=2
1960: C      . . .
1961: C      OVER J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=N
1962: C      ALL X(I) => 0
1963:      PARAMETER (N=4) ! N IS THE NO. OF CONSTRAINTS + 1
1964:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1965:      DIMENSION X(*),A(20,10),C(20),FF(20)
1966:      DATA (A(1,J),J=1,3),C(1)/30.D0,40.D0,20.D0,0.0D0/! COEFF OF OBJ FUNCTION
1967:      DATA (A(2,J),J=1,3),C(2)/10.D0,12.D0,7.D0,10000.0D0/! COEFF OF 1ST CONSTRAINT
1968:      DATA (A(3,J),J=1,3),C(3)/7.D0,10.D0,8.D0,8000.0D0/! COEFF OF 2ND CONSTRAINT
1969:      DATA (A(4,J),J=1,3),C(4)/1.D0,1.D0,1.D0,1000.0D0/! COEFF OF 3RD CONSTRAINT
1970: C      -----
1971: C      USING ONLY NON-NEGATIVE VALUES OF X(I)
1972:      DO I=1,M
1973:      X(I)=DABS(X(I))
1974:      ENDDO
1975: C      EVALUATION OF OBJ FUNCTION AND CONSTRAINTS
1976:      DO I=1,N
1977:      FF(I)=0.D00
1978:      DO J=1,M
1979:      FF(I)=FF(I)+A(I,J)*X(J)
1980:      ENDDO
1981:      ENDDO
1982:      F=-FF(1) ! CHANGE OF SIGN FOR MINIMIZATION
1983: C      CHECK FOR SATISFYING OR VIOLATING THE CONSTRAINTS
1984:      DO I=2,N
1985:      FF(I)=FF(I)-C(I) ! SLACK
1986: C      PENALTY FOR CROSSING LIMITS
1987:      IF (FF(I) .GT. 0.D00) F=F+(100.D00+FF(I))**2
1988:      ENDDO
1989:      RETURN
1990:      END
1991: C      -----
1992:      SUBROUTINE HOUGEN(A,M,F)
1993:      PARAMETER (N=13,K=3)
1994:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
1995:      DIMENSION X(N,K),RATE(N),A(*)
1996: C      -----
1997: C      HOUGEN FUNCTION (HOUGEN-WATSON MODEL FOR REACTION KINATICS)
1998: C      NO. OF PARAMETERS (A) TO ESTIMATE = 5 = M
1999:
2000: C      BEST RESULTS ARE:
2001: C      A(1)=1.253031; A(2)=1.190943; A(3)=0.062798; A(4)=0.040063
2002: C      A(5)=0.112453 ARE BEST ESTIMATES OBTAINED BY ROSENBROCK &
2003: C      QUASI-NEWTON METHOD WITH SUM OF SQUARES OF DEVIATION =0.298900994
2004: C      AND R=0.99945.
2005:
2006: C      THE NEXT BEST RESULTS GIVEN BY HOOKE-JEEVES & QUASI-NEWTON
2007: C      A(1)=2.475221;A(2)=0.599177; A(3)=0.124172; A(4)=0.083517
2008: C      A(5)=0.217886; SUM OF SQUARES OF DEVIATION = 0.318593458
2009: C      R=0.99941
2010: C      MOST OF THE OTHER METHODS DO NOT PERFORM WELL

```

```

2011: C -----
2012: DATA X(1,1),X(1,2),X(1,3),RATE(1) /470,300,10,8.55/
2013: DATA X(2,1),X(2,2),X(2,3),RATE(2) /285,80,10,3.79/
2014: DATA X(3,1),X(3,2),X(3,3),RATE(3) /470,300,120,4.82/
2015: DATA X(4,1),X(4,2),X(4,3),RATE(4) /470,80,120,0.02/
2016: DATA X(5,1),X(5,2),X(5,3),RATE(5) /470,80,10,2.75/
2017: DATA X(6,1),X(6,2),X(6,3),RATE(6) /100,190,10,14.39/
2018: DATA X(7,1),X(7,2),X(7,3),RATE(7) /100,80,65,2.54/
2019: DATA X(8,1),X(8,2),X(8,3),RATE(8) /470,190,65,4.35/
2020: DATA X(9,1),X(9,2),X(9,3),RATE(9) /100,300,54,13/
2021: DATA X(10,1),X(10,2),X(10,3),RATE(10) /100,300,120,8.5/
2022: DATA X(11,1),X(11,2),X(11,3),RATE(11) /100,80,120,0.05/
2023: DATA X(12,1),X(12,2),X(12,3),RATE(12) /285,300,10,11.32/
2024: DATA X(13,1),X(13,2),X(13,3),RATE(13) /285,190,120,3.13/
2025: C WRITE(*,1)((X(I,J),J=1,K),RATE(I),I=1,N)
2026: C 1 FORMAT(4F8.2)
2027: DO J=1,M
2028: IF(DABS(A(J)).GT.5.D00) THEN
2029: CALL RANDOM(RAND)
2030: A(J)=(RAND-0.5D00)*10.D00
2031: ENDIF
2032: ENDDO
2033: F=0.D00
2034: DO I=1,N
2035: D=1.D00
2036: DO J=1,K
2037: D=D+A(J+1)*X(I,J)
2038: ENDDO
2039: FX=(A(1)*X(I,2)-X(I,3)/A(M))/D
2040: C FX=(A(1)*X(I,2)-X(I,3)/A(5))/(1.D00+A(2)*X(I,1)+A(3)*X(I,2)+
2041: C A(4)*X(I,3))
2042: F=F+(RATE(I)-FX)**2
2043: ENDDO
2044: RETURN
2045: END
2046: C -----
2047: SUBROUTINE GIUNTA(M,X,F)
2048: IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2049: DIMENSION X(*)
2050: C GIUNTA FUNCTION
2051: C X(I) = -1 TO 1; M=2
2052: DO I=1,M
2053: IF(DABS(X(I)).GT.1.D00) THEN
2054: CALL RANDOM(RAND)
2055: X(I)=(RAND-0.5D00)*2.D00
2056: ENDIF
2057: ENDDO
2058: C=16.D00/15.D00
2059: F=DSIN(C*X(1)-1.D0)+DSIN(C*X(1)-1.D0)**2+DSIN(4*(C*X(1)-1.D0))/50+
2060: &DSIN(C*X(2)-1.D0)+DSIN(C*X(2)-1.D0)**2+DSIN(4*(C*X(2)-1.D0))/50+.6
2061: RETURN
2062: END
2063: C -----
2064: SUBROUTINE EGGHOLD(M,X,F)
2065: C EGG HOLDER FUNCTION
2066: IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2067: DIMENSION X(*)
2068: DO I=1,M
2069: IF(DABS(X(I)).GT.512.D00) THEN
2070: CALL RANDOM(RAND)
2071: X(I)=(RAND-0.5D00)*1024.D00
2072: ENDIF
2073: ENDDO
2074: F=0.D00
2075: DO I=1,M-1
2076: F1=-(X(I+1)+47.D00)
2077: F2=DSIN( DSQRT( DABS( X(I+1)+X(I)/2+47.D00 ) ) ) )

```

```

2078:      F3=DSIN( DSQRT( DABS( X(I)-(X(I+1)+47.D00) ) ) )
2079:      F4=-X(I)
2080:      F=F+ F1*F2+F3*F4
2081:      ENDDO
2082:      RETURN
2083:      END

```

```

2084: C -----
2085:      SUBROUTINE TRID(M,X,F)
2086: C      TRID FUNCTION
2087:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2088:      DIMENSION X(*)
2089:      F1=0.D00
2090:      F2=0.D00
2091:      DO I=1, M
2092:      F1=F1+(X(I)-1.D00)**2
2093:      ENDDO
2094:      DO I=2, M
2095:      F2=F2+X(I)*X(I-1)
2096:      ENDDO
2097:      F=F1-F2
2098:      RETURN
2099:      END

```

```

2100: C -----
2101:      SUBROUTINE GRIEWANK(M,X,F)
2102:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2103:      DIMENSION X(*)
2104: C      GRIEWANK FUNCTION
2105:      F1=0.D00
2106:      F2=1.0D00
2107:      DO I=1, M
2108:      F1=F1+X(I)**2
2109:      FI=DFLOAT(I)
2110:      F2=F2*DCOS(X(I)/DSQRT(FI))
2111:      ENDDO
2112:      F=F1/4000.D00-F2+1.0D00
2113:      RETURN
2114:      END

```

```

2115: C -----
2116:      SUBROUTINE WEIERSTRASS(M,X,F)
2117:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2118:      DIMENSION X(*)
2119:      PI=4*DATAN(1.D00)
2120: C      WEIERSTRASS FUNCTION -----
2121:      DATA AX,BX,KMAX/0.5D00,3.D00,20/
2122:      f1=0.D00
2123:      f2=0.D00
2124:
2125:      DO I=1,M
2126:      IF (DABS(X(I)).GT.0.5D00) THEN
2127:      CALL RANDOM(RAND)
2128:      X(I)=RAND-0.5D00
2129:      ENDIF
2130:      ENDDO
2131:
2132:      DO I=1, M
2133:      S=0.D00
2134:      DO KK=1, KMAX+1
2135:      K=KK-1
2136:      S=S+(AX**K*DCOS(2*PI*BX**K*(X(I)+0.5D00)))
2137:      ENDDO
2138:      F1=F1+S
2139:      ENDDO
2140:
2141:      DO KK=1, KMAX+1
2142:      K=KK-1
2143:      F2=F2+(AX**K*DCOS(2*PI*BX**K*0.5D00))
2144:      ENDDO

```

```

2145:      F=F1-M*F2
2146:      RETURN
2147:      END
2148: C -----
2149:      SUBROUTINE LEVY3 (M,X,F)
2150:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2151:      DIMENSION X(*)
2152: C      LEVY # 3 (LEVY ET AL. 1981) -----
2153:      DO I=1,M
2154:      IF (DABS(X(I)) .GT. 10.D00) THEN
2155:      CALL RANDOM(RAND)
2156:      X(I)=(RAND-0.5D00)*20
2157:      ENDIF
2158:      ENDDO
2159:      F1=0.0D+00
2160:      F2=0.0D+00
2161:      DO I=1,5
2162:      F1=F1+(I*DCOS((I-1)*X(1)+I))
2163:      F2=F2+(I*DCOS((I+1)*X(2)+I))
2164:      ENDDO
2165:      F=F1*F2
2166:      RETURN
2167:      END
2168: C -----
2169:      SUBROUTINE LEVY5 (M,X,F)
2170:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2171:      DIMENSION X(*)
2172:      F1=0.0D+00
2173:      F2=0.0D+00
2174:      DO I=1,5
2175:      F1=F1+(I*DCOS((I-1)*X(1)+I))
2176:      F2=F2+(I*DCOS((I+1)*X(2)+I))
2177:      ENDDO
2178:      F3=(X(1)+1.42513D+00)**2
2179:      F4=(X(2)+0.80032D+00)**2
2180:      F=(F1*F2) + (F3+F4)
2181:      RETURN
2182:      END
2183: C -----
2184:      SUBROUTINE LEVY8 (M,X,F)
2185:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2186:      DIMENSION X(*),Y(3)
2187:      PI=4*DATAN(1.D00)
2188: C      LEVY # 8 FUNCTION -----
2189:      DO I=1,3
2190:      Y(I)=1.D+00+(X(I)-1.D+00)/4.D+00
2191:      ENDDO
2192:      F1=DSIN(PI*Y(1))**2
2193:      F3=(Y(3)-1.D+00)**2
2194:      F2=0.D+00
2195:      DO I=1,2
2196:      F2=F2+((Y(I)-1.D+00)**2)*(1.D+00+10.D+00*(DSIN(PI*Y(I+1))))**2)
2197:      ENDDO
2198:      F=F1+F2+F3
2199:      RETURN
2200:      END
2201: C -----
2202:      SUBROUTINE RASTRIGIN(M,X,F)
2203:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2204:      DIMENSION X(*)
2205:      PI=4*DATAN(1.D00)
2206: C      RASTRIGIN'S FUNCTION
2207:      F=0.D00
2208:      DO I=1,M
2209:      F=F+ X(I)**2 -10*DCOS(2*PI*X(I)) + 10.D00
2210:      ENDDO
2211:      RETURN

```

```

2212:      END
2213: C -----
2214:      SUBROUTINE ACKLEY (M,X,F)
2215: C      ACKLEY FUNCTION
2216:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2217:      DIMENSION X(*)
2218:      PI=4*DATAN(1.D00)
2219:      F=20.D00+DEXP(1.D00)
2220:      DO I=1,M
2221:      IF (X(I).LT. -15.D00 .OR. X(I).GT.30.D00) THEN
2222:      CALL RANDOM(RAND)
2223:      X(I)=(RAND-0.5D00)*90 -15.D00
2224:      ENDIF
2225:      ENDDO
2226:      F1=0.D00
2227:      F2=0.D00
2228:      DO I=1,M
2229:      F1=F1+X(I)**2
2230:      F2=F2+DCOS(2*PI*X(I))
2231:      ENDDO
2232:      F1=-20*DEXP(-0.2D00*DSQRT(F1/M))
2233:      F2=-DEXP(F2/M)
2234:      F=F+F1+F2
2235:      RETURN
2236:      END
2237: C -----
2238:      SUBROUTINE MICHALEWICZ (M,X,F)
2239:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2240:      DIMENSION X(*)
2241: C      MICHALEWICZ FUNCTION [ 0 <= X(I) <= PI ] MP IS A PARAMETER
2242:      MP=10 ! SET IT TO THE DESIRED VALUE
2243:      PI=4*DATAN(1.D00)
2244:      DO I=1,M
2245:      IF (X(I).LT.0.D00 .OR. X(I).GT.PI) THEN
2246:      CALL RANDOM(RAND)
2247:      X(I)=RAND*PI
2248:      ENDIF
2249:      ENDDO
2250:      F=0.D00
2251:      DO I=1,M
2252:      F=F-DSIN(X(I))*(DSIN(I*X(I)**2/PI))**(2*MP)
2253:      ENDDO
2254:      RETURN
2255:      END
2256: C -----
2257:      SUBROUTINE SCHWEFEL (M,X,F)
2258: C      SCHWEFEL FUNCTION
2259:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2260:      DIMENSION X(*)
2261:      DO I=1,M
2262:      IF (DABS(X(I)).GT.500) THEN
2263:      CALL RANDOM(RAND)
2264:      X(I)=(RAND-0.5D00)*1000
2265:      ENDIF
2266:      ENDDO
2267:      F=0.D00
2268:      DO I=1,M
2269:      F=F+ X(I)*DSIN(DSQRT(DABS(X(I))))
2270:      ENDDO
2271:      F=418.9829D00*M - F
2272:      RETURN
2273:      END
2274: C -----
2275:      SUBROUTINE SHUBERT (M,X,F)
2276: C      SHUBERT FUNCTION
2277:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2278:      DIMENSION X(*)

```

```

2279:      DO I=1,M
2280:      IF (DABS(X(I)) .GT.10.D00) THEN
2281:      CALL RANDOM(RAND)
2282:      X(I)=(RAND-0.5D00)*20
2283:      ENDIF
2284:      ENDDO
2285:      F1=0.D00
2286:      F2=0.D00
2287:      DO I=1,5
2288:      F1=F1+I*DCOS((I+1.D00)*X(1)+I)
2289:      F2=F2+I*DCOS((I+1.D00)*X(2)+I)
2290:      ENDDO
2291:      F=F1*F2
2292:      RETURN
2293:      END
2294: C -----
2295:      SUBROUTINE DIXPRICE(M,X,F)
2296: C      DIXON & PRICE FUNCTION
2297:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2298:      DIMENSION X(*)
2299:      DO I=1,M
2300:      IF (DABS(X(I)) .GT.10.D00) THEN
2301:      CALL RANDOM(RAND)
2302:      X(I)=(RAND-0.5D00)*20
2303:      ENDIF
2304:      ENDDO
2305:      F=0.D00
2306:      DO I=2, M
2307:      F=F + I*(2*X(I)**2-X(I-1))**2
2308:      ENDDO
2309:      F=F+(X(1)-1.D00)**2
2310:      RETURN
2311:      END
2312: C -----
2313:      SUBROUTINE SHEKEL(M,X,F)
2314: C      SHEKEL FUNCTION FOR TEST OF GLOBAL OPTIMIZATION METHODS
2315:      PARAMETER(NROW=10,NCOL=4, NR=9)! NR MAY BE 2 TO 10
2316:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2317:      DIMENSION A(NROW,NCOL),C(NROW),X(*)
2318:      DATA ((A(I,J),J=1,NCOL),I=1,NROW)/4.,4.,4.,4.,1.,1.,1.,1.,8.,8.,
2319: & 8.,8.,6.,6.,6.,6.,3.,7.,3.,7.,2.,9.,2.,9.,5.,5.,3.,3.,8.,1.,8.,
2320: & 1.,6.,2.,6.,2.,7.,3.6D00,7.,3.6D00/
2321:      DATA (C(I),I=1,NROW)/0.1D00,0.2D00,0.2D00,0.4D00,0.4D00,0.6D00,
2322: & 0.3D00,0.7D00,0.5D00,0.5D00/
2323:      F=0.D00
2324:      DO I=1,NR
2325:      S=0.D00
2326:      DO J=1,M
2327:      S=S+(X(J)-A(I,J))**2
2328:      ENDDO
2329:      F=F-1.D00/(S+C(I))
2330:      ENDDO
2331:      RETURN
2332:      END
2333: C -----
2334:      SUBROUTINE PAVIANI(M,X,F)
2335: C      PAVIANI FUNCTION : MIN F(9.3502,...,9.3502)=45.77847 APPROX
2336: C      IN THE DOMAIN 2<= X(I) <= 10 FOR I=1,2,...,10.
2337:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2338:      DIMENSION X(*)
2339:      DO I=1,M
2340:      IF(X(I) .LE.2.D00.OR.X(I) .GE.10.D00) THEN
2341:      CALL RANDOM(RAND)
2342:      X(I)=RAND*8+2.D00
2343:      ENDIF
2344:      ENDDO
2345:      F1=0.D00

```



```

2346:      F2=1.D00
2347:      DO I=1,M
2348:      F1=F1+ DLOG(X(I)-2.D00)**2+DLOG(10.D00-X(I))**2
2349:      F2=F2*X(I)
2350:      ENDDO
2351:      F=F1-F2**0.2
2352:      RETURN
2353:      END
2354: C -----
2355:      SUBROUTINE BRANIN1(M,X,F)
2356: C      BRANIN FUNCTION #1 MIN F (1, 0) = 0
2357:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2358:      DIMENSION X(*)
2359:      PI=4*DATAN(1.D00)
2360:      DO I=1,M
2361:      IF(DABS(X(I)).GT.10.D00) THEN
2362:      CALL RANDOM(RAND)
2363:      X(I)=(RAND-0.5D00)*20
2364:      ENDIF
2365:      ENDDO
2366:      F=(1.D00-2*X(2)+DSIN(4*PI*X(2))/2.D00-X(1))**2+(X(2)-
2367: & DSIN(2*PI*X(1))/2.D00)**2
2368:      RETURN
2369:      END
2370: C -----
2371:      SUBROUTINE BRANIN2(M,X,F)
2372: C      BRANIN FUNCTION #2 MIN F (3.1416, 2.25)= 0.397887 APPROX
2373:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2374:      DIMENSION X(*)
2375:      PI=4*DATAN(1.D00)
2376:      IF(X(1).LT.-5.D00 .OR. X(1).GT.10.D00) THEN
2377:      CALL RANDOM(RAND)
2378:      X(1)=RAND*15-5.D00
2379:      ENDIF
2380:      IF(X(2).LT.0.D00 .OR. X(2).GT.15.D00) THEN
2381:      CALL RANDOM(RAND)
2382:      X(2)=RAND*15
2383:      ENDIF
2384:      F=(X(2)-5.D00*X(1))**2/(4*PI**2)+5*X(1)/PI-6.D00)**2 +
2385: & 10*(1.D00-1.D00/(8*PI))*DCOS(X(1))+10.D00
2386:      RETURN
2387:      END
2388: C -----
2389:      SUBROUTINE BOHACHEVSKY1(M,X,F)
2390: C      BOHACHEVSKY FUNCTION #1 : MIN F (0, 0) = 0
2391:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2392:      DIMENSION X(*)
2393:      PI=4*DATAN(1.D00)
2394:      DO I=1,M
2395:      IF(DABS(X(I)).GT.100.D00) THEN
2396:      CALL RANDOM(RAND)
2397:      X(I)=(RAND-0.5D00)*200
2398:      ENDIF
2399:      ENDDO
2400:      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1))-0.4D00*DCOS(4*PI*X(2))
2401: & +0.7D00
2402:      RETURN
2403:      END
2404: C -----
2405:      SUBROUTINE BOHACHEVSKY2(M,X,F)
2406: C      BOHACHEVSKY FUNCTION #2 : MIN F (0, 0) = 0
2407:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2408:      DIMENSION X(*)
2409:      PI=4*DATAN(1.D00)
2410:      DO I=1,M
2411:      IF(DABS(X(I)).GT.100.D00) THEN
2412:      CALL RANDOM(RAND)

```

```

2413:      X(I)=(RAND-0.5D00)*200
2414:      ENDIF
2415:      ENDDO
2416:      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1))*DCOS(4*PI*X(2))+0.3D00
2417:      RETURN
2418:      END
2419: C -----
2420:      SUBROUTINE BOHACHEVSKY3(M,X,F)
2421: C      BOHACHEVSKY FUNCTION #3 : MIN F(0, 0) = 0
2422:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2423:      DIMENSION X(*)
2424:      PI=4*DATAN(1.D00)
2425:      DO I=1,M
2426:      IF (DABS(X(I)).GT.100.D00) THEN
2427:      CALL RANDOM(RAND)
2428:      X(I)=(RAND-0.5D00)*200
2429:      ENDIF
2430:      ENDDO
2431:      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1))+4*PI*X(2))+0.3D00
2432:      RETURN
2433:      END
2434: C -----
2435:      SUBROUTINE EASOM(M,X,F)
2436: C      EASOM FUNCTION : 2-VARIABLES, MIN F(PI, PI) = -1.
2437:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2438:      DIMENSION X(*)
2439:      PI=4*DATAN(1.D00)
2440:      DO I=1,M
2441:      IF (DABS(X(I)).GT.100.D00) THEN
2442:      CALL RANDOM(RAND)
2443:      X(I)=(RAND-0.5D00)*200
2444:      ENDIF
2445:      ENDDO
2446:      F=-DCOS(X(1))*DCOS(X(2))*DEXP(-(X(1)-PI)**2-(X(2)-PI)**2)
2447:      RETURN
2448:      END
2449: C -----
2450:      SUBROUTINE ROSENBROCK(M,X,F)
2451: C      ROSENBROCK FUNCTION : M VARIABLE; MIN F(1, 1,...,1)=0
2452:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2453:      DIMENSION X(*)
2454:      DO I=1,M
2455:      IF (X(I).LT.-5.D00 .OR. X(I).GT.10.D00) THEN
2456:      CALL RANDOM(RAND)
2457:      X(I)=RAND*15-5.D00
2458:      ENDIF
2459:      ENDDO
2460:      F=0.D00
2461:      DO I=1,M-1
2462:      F=F+(100.D00*(X(I+1)-X(I)**2)**2+(X(I)-1.D00)**2)
2463:      ENDDO
2464:      RETURN
2465:      END
2466: C -----
2467:      SUBROUTINE CROSSLEG(M,X,F)
2468:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2469:      DIMENSION X(*)
2470: C      CROSS-LEGGED TABLE FUNCTION ; -10<= X(I) <=10; M=2
2471: C      MIN F(0, X) OR F(X, 0) = -1.
2472:      PI=4*DATAN(1.D00)
2473:      DO I=1,M
2474:      IF (DABS(X(I)).GT.10.D00) THEN
2475:      CALL RANDOM(RAND)
2476:      X(I)=(RAND-0.5D00)*20
2477:      ENDIF
2478:      ENDDO
2479:      F=-(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT

```

```

2480:      & (X(1)**2+X(2)**2)/PI))) +1.D00)**(-.1)
2481:      RETURN
2482:      END
2483: C -----
2484:      SUBROUTINE CROSS(M,X,F)
2485:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2486:      DIMENSION X(*)
2487:      CROSS FUNCTION ; -10<= X(I) <=10; M=2;
2488: C      MIN F(A, B)=0 APPROX; A, B=1.3494 APPROX OF EITHER SIGN (+ OR -)
2489:      PI=4*DATAN(1.D00)
2490:      DO I=1,M
2491:      IF( DABS(X(I)) .GT.10.D00) THEN
2492:      CALL RANDOM(RAND)
2493:      X(I)=(RAND-0.5D00)*20
2494:      ENDIF
2495:      ENDDO
2496:      F=(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
2497: & (X(1)**2+X(2)**2)/PI)))) +1.D00)**(-.1)
2498:      RETURN
2499:      END
2500: C -----
2501:      SUBROUTINE CROSSINTRAY(M,X,F)
2502:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2503:      DIMENSION X(*)
2504: C      CROSS IN TRAY FUNCTION ; -10<= X(I) <=10; M=2;
2505: C      MIN F(A, B)=-20626.1218 APPROX; A, B=1.3494 APPROX OF EITHER SIGN
2506:      PI=4*DATAN(1.D00)
2507:      DO I=1,M
2508:      IF( DABS(X(I)) .GT.10.D00) THEN
2509:      CALL RANDOM(RAND)
2510:      X(I)=(RAND-0.5D00)*20
2511:      ENDIF
2512:      ENDDO
2513:      F=- (DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
2514: & (X(1)**2+X(2)**2)/PI)))) +1.D00)**(.1)
2515:      RETURN
2516:      END
2517: C -----
2518:      SUBROUTINE CROWNEDCROSS(M,X,F)
2519:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2520:      DIMENSION X(*)
2521: C      CROWNED CROSS FUNCTION ; -10<= X(I) <=10; M=2; MIN F = 1
2522:      PI=4*DATAN(1.D00)
2523:      DO I=1,M
2524:      IF( DABS(X(I)) .GT.10.D00) THEN
2525:      CALL RANDOM(RAND)
2526:      X(I)=(RAND-0.5D00)*20
2527:      ENDIF
2528:      ENDDO
2529:      F=(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-
2530: & (DSQRT(X(1)**2+X(2)**2)/PI)))) +1.D00)**(.1)
2531:      RETURN
2532:      END
2533: C -----
2534:      SUBROUTINE TTHOLDER(M,X,F)
2535:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2536:      DIMENSION X(*)
2537: C      TEST-TUBE HOLDER FUNCTION ; -10<= X(I) <=10; M=2;
2538: C      MIN F([+/-]1.5706, 0)= -10.8723
2539:      PI=4*DATAN(1.D00)
2540:      DO I=1,M
2541:      IF( DABS(X(I)) .GT.10.D00) THEN
2542:      CALL RANDOM(RAND)
2543:      X(I)=(RAND-0.5D00)*20
2544:      ENDIF
2545:      ENDDO
2546:      F=-4*DABS(DSIN(X(1))*DCOS(X(2))*DEXP(DABS(DCOS((X(1)**2+X(2)**2)/

```

```

2547:      & 200))) )
2548:      RETURN
2549:      END
2550: C -----
2551:      SUBROUTINE HOLDERTABLE(M,X,F)
2552:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2553:      DIMENSION X(*)
2554:      HOLDER-TABLE FUNCTION ; -10<= X(I) <=10; M=2;
2555: C MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -26.92034 APPROX
2556:      PI=4*DATAN(1.D00)
2557:      DO I=1,M
2558:      IF ( DABS(X(I)) .GT.10.D00) THEN
2559:      CALL RANDOM(RAND)
2560:      X(I)=(RAND-0.5D00)*20
2561:      ENDIF
2562:      ENDDO
2563:      F=-DABS(DCOS(X(1))*DCOS(X(2))*DEXP(DABS(1.D00-(DSQRT(X(1)**2+
2564: & X(2)**2)/PI))))
2565:      RETURN
2566:      END
2567: C -----
2568:      SUBROUTINE CARROMTABLE(M,X,F)
2569:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2570:      DIMENSION X(*)
2571: C CARROM-TABLE FUNCTION ; -10<= X(I) <=10; M=2;
2572: C MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -24.15682 APPROX
2573:      PI=4*DATAN(1.D00)
2574:      DO I=1,M
2575:      IF ( DABS(X(I)) .GT.10.D00) THEN
2576:      CALL RANDOM(RAND)
2577:      X(I)=(RAND-0.5D00)*20
2578:      ENDIF
2579:      ENDDO
2580:      F=-1.D00/30*(DCOS(X(1))*DCOS(X(2))*DEXP(DABS(1.D00-
2581: & (DSQRT(X(1)**2 + X(2)**2)/PI))))**2
2582:      RETURN
2583:      END
2584: C -----
2585:      SUBROUTINE PENHOLDER(M,X,F)
2586:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2587:      DIMENSION X(*)
2588: C PENHOLDER FUNCTION ; -11<= X(I) <=11; M=2;
2589: C MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -0.963535 APPROX
2590:      PI=4*DATAN(1.D00)
2591:      DO I=1,M
2592:      IF ( DABS(X(I)) .GT.11.D00) THEN
2593:      CALL RANDOM(RAND)
2594:      X(I)=(RAND-0.5D00)*22
2595:      ENDIF
2596:      ENDDO
2597:      F=-DEXP(-(DABS(DCOS(X(1))*DCOS(X(2))*DEXP(DABS(1.D0-(DSQRT
2598: & (X(1)**2+X(2)**2)/PI))))**(-1)))
2599:      RETURN
2600:      END
2601: C -----
2602:      SUBROUTINE BIRD(M,X,F)
2603:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2604:      DIMENSION X(*)
2605: C BIRD FUNCTION ; -2PI<= X(I) <=2PI; M=2;
2606: C MIN F (4.70104, 3.15294) APPROX = -106.764537 APPROX OR
2607: C MIN F (-1.58214, -3.13024) APPROX = -106.764537 APPROX
2608:      PI=4*DATAN(1.D00)
2609:      DO I=1,M
2610:      IF ( DABS(X(I)) .GT.2*PI) THEN
2611:      CALL RANDOM(RAND)
2612:      X(I)=(RAND-0.5D00)*4*PI
2613:      ENDIF

```

```

2614:      ENDDO
2615:      F=(DSIN(X(1))*DEXP((1.D00-DCOS(X(2)))**2) +
2616: & DCOS(X(2))*DEXP((1.D00-DSIN(X(1)))**2)+(X(1)-X(2))**2
2617:      RETURN
2618:      END
2619: C -----
2620:      SUBROUTINE CHICHINADZE(M,X,F)
2621:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2622:      DIMENSION X(*)
2623: C      CHICHINADZE FUNCTION : -30 <=X(I)<= 30; M=2
2624: C      MIN F (5.901329, 0.5) = -43.3158621
2625:      PI=4*DATAN(1.D00)
2626:      DO I=1,M
2627:      IF( DABS(X(I)).GT.30) THEN
2628:      CALL RANDOM(RAND)
2629:      X(I)=(RAND-0.5D00)*60
2630:      ENDIF
2631:      ENDDO
2632:      F=X(1)**2-12*X(1)+11.D00+10*DCOS(PI*X(1)/2)+8*DSIN(5*PI*X(1))-
2633: & (1.D00/DSQRT(5.D00))*DEXP(-(X(2)-0.5D00)**2/2)
2634:      RETURN
2635:      END
2636: C -----
2637:      SUBROUTINE MCCORMICK(M,X,F)
2638:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2639:      DIMENSION X(*)
2640: C      MCCORMICK FUNCTION : -1.5<= X(1)<=4; -3<=X(2)<=4 ; M=2
2641: C      MIN F (-0.54719755, -1.54719755) = -1.913223 APPROX
2642:      IF(X(1).LT. -1.5D00 .OR. X(1) .GT. 4.D00) THEN
2643:      CALL RANDOM(RAND)
2644:      X(1)=RAND*5.5D00-1.5D00
2645:      ENDIF
2646:      IF(X(2).LT. -3.D00 .OR. X(2) .GT. 4.D00) THEN
2647:      CALL RANDOM(RAND)
2648:      X(2)=RAND*7.D00-3.D00
2649:      ENDIF
2650:      F=DSIN(X(1)+X(2))+(X(1)-X(2))**2-1.5*X(1)+2.5*X(2)+1.D00
2651:      RETURN
2652:      END
2653: C -----
2654:      SUBROUTINE FENTONEASON(M,X,F)
2655:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2656:      DIMENSION X(*)
2657: C      FENTON & EASON FUNCTION FMIN(1.74345, -2.029695) = 1.744152
2658:      DO I=1,M
2659:      IF(DABS(X(I)).GT.100.D00) THEN
2660:      CALL RANDOM(RAND)
2661:      X(I)=(RAND-0.5D00)*200
2662:      ENDIF
2663:      ENDDO
2664:      F=1.2D00+0.1*X(1)**2+(0.1D00+0.1*X(2)**2)/X(1)**2+
2665: & (.1*X(1)**2*X(2)**2+10.D00)/((X(1)*X(2))**4)
2666:      RETURN
2667:      END
2668: C -----
2669:      SUBROUTINE WOOD(M,X,F)
2670:      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
2671:      COMMON /RNDM/IU,IV
2672:      INTEGER IU,IV
2673:      DIMENSION X(*)
2674: C      WOOD FUNCTION:FMIN(0.443546,-0.194607,1.466077,2.15115)=1.09485393
2675:      DO I=1,M
2676:      IF(DABS(X(I)).GT.5.D00) THEN
2677:      CALL RANDOM(RAND)
2678:      X(I)=(RAND-0.5D00)*10
2679:      ENDIF
2680:      ENDDO

```

```

2681:      F1=100*(X(2)+X(1)**2)**2 + (1.D0-X(1))**2 +90*(X(4)-X(3)**2)**2
2682:      F2=(1.D0-X(3))**2 +10.1*((X(2)-1.D0)**2+(X(4)-1.D0)**2)
2683:      F3=19.8*(X(2)-1.D0)*(X(4)-1.D0)
2684:      F=F1+F2+F3
2685:      RETURN
2686:      END
2687: C -----
2688:      SUBROUTINE GLANKWAHMDEE(M,X,F)
2689:      PARAMETER (N=5)
2690:      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
2691:      COMMON /RNDM/IU,IV
2692:      INTEGER IU,IV
2693:      DIMENSION X(*),A(N,N), B(N)
2694: C ----- GLANKWAHMDEE FUNCTION -----
2695: C Glankwahnmddee A, Liebman JS and Hogg GL (1979) "Unconstrained
2696: C Discrete Nonlinear Programming. Engineering Optimization 4: 95-107
2697: C PARSOPOULOS, KE and VRAHATIS, MN (2002) Recent Approaches to
2698: C Global Optimization Problems through Particle Swarm Optimization,
2699: C Natural Computing 1: 235-306, 2002. REPORT THE BEST OBTAINED
2700: C FMIN (0,12,23,17,6)= -737 OR MIN F(0, 11,22,16,6)= -737
2701: C WE GET FMIN(-.232, 11.489, 22.273, 16.540, 6.115) = -739.822991
2702: C -----
2703:      Data ((a(I,j),j=1,n),I=1,n) /35,-20,-10,32,-10,-20, 40,-6,-31,32,
2704: & -10,-6,11,-6,-10,32,-31,-6,38,-20,-10,32,-10,-20,31/
2705:      DATA (b(j),j=1,n) /-15,-27,-36,-18,-12/
2706: C -----
2707:      DO I=1,M
2708:      IF(DABS(X(I)).GT.100.D00) THEN
2709:      CALL RANDOM(RAND)
2710:      X(I)=(RAND-0.5D00)*200
2711:      ENDIF
2712:      ENDDO
2713:      F=0.d0
2714:      DO J=1,m
2715:      F=f+b(j)*x(j)
2716:      Enddo
2717:      DO I=1,m
2718:      C=0.d0
2719:      DO j=1,m
2720:      C=c+x(j)*a(j,I)
2721:      Enddo
2722:      F=f+c*x(i)
2723:      Enddo
2724:      RETURN
2725:      END
2726: C -----
2727:      SUBROUTINE FLETCHER(M,X,F)
2728: C FLETCHER-POWELL FUNCTION, M <= 10, ELSE IT IS VERY MUCH SLOW
2729: C SOLUTION: MIN F = 0 FOR X=(C1, C2, C3,...,CM)
2730: C PARAMETER(N=10) ! FOR DIMENSION OF DIFFERENT MATRICES AND VECTORS
2731: C IMPLICIT DOUBLE PRECISION (A-H,O-Z)
2732: C COMMON /RNDM/IU,IV
2733: C INTEGER IU,IV
2734: C DIMENSION X(*),A(N,N),B(N,N),AA(N),BB(N),AL(N),C(N),C1(N)
2735: C PI=4*DATAN(1.D00)
2736: C GENERATE A(I,J) AND B(I,J) BETWEEN (-100, 100) RANDOMLY.
2737: C C(I) = BETWEEN (-PI, PI) IS EITHER GIVEN OR RANDOMLY GENERATED.
2738: C DATA (C(I),I=1,10)/1,2,3,-3,-2,-1,0,1,2,3/ ! BETWEEN -PI AND PI
2739: C DATA (C1(I),I=1,N)/-3,-3.02,-3.01,1,1.03,1.02,1.03,-.08,.001,3/
2740: C DATA (C1(I),I=1,N)/0,0,0,0,0,0,0,0,0,0/ ! another example c1 = 0
2741: C NC=0 ! DEFINE NC HERE 0 OR 1 OR 2
2742: C IF NC=0, C1 FROM DATA IS USED (THAT IS FIXED C);
2743: C IF NC=1, C IS MADE FROM C1 BY ADDING RANDOM PART - THAT IS C=C1+r
2744: C IF NC=2 THEN C IS PURELY RANDOM THAT IS C= 0 + r
2745: C IN ANY CASE C LIES BETWEEN -PI AND PI.
2746: C -----
2747: C FIND THE MAX MAGNITUDE ELEMENT IN C1 VECTOR (UPTO M ELEMENTS)

```

```

2748:      CMAX=DABS(C1(1))
2749:      DO J=2,M
2750:      IF(DABS(C1(J)).GT.CMAX) CMAX=DABS(C1(J))
2751:      ENDDO
2752:      RANGE=PI-CMAX
2753: C-----
2754:      DO J=1,M
2755:      DO I=1,M
2756:      CALL RANDOM(RAND)
2757:      A(I,J)=(RAND-0.5D00)*200.D00
2758:      CALL RANDOM(RAND)
2759:      B(I,J)=(RAND-0.5D00)*200.D00
2760:      ENDDO
2761:      IF(NC.EQ.0) AL(J)=C1(J) ! FIXED OR NON-STOCHASTIC C
2762:      IF(NC.EQ.1) THEN
2763:      CALL RANDOM(RAND)
2764:      AL(J)=C1(J)+(RAND-0.5D00)*2*RANGE ! A PART FIXED, OTHER STOCHASTIC
2765:      ENDIF
2766:      IF(NC.EQ.2) THEN
2767:      CALL RANDOM(RAND)
2768:      AL(J)=(RAND-0.5D00)*2*PI ! PURELY STOCHASTIC
2769:      ENDIF
2770:      ENDDO
2771:      DO I=1,M
2772:      AA(I)=0.D00
2773:      DO J=1,M
2774:      AA(I)=AA(I)+A(I,J)*DSIN(AL(J))+B(I,J)*DCOS(AL(J))
2775:      ENDDO
2776:      ENDDO
2777: C-----
2778:      DO I=1,M
2779:      IF(DABS(X(I)).GT.PI) THEN
2780:      CALL RANDOM(RAND)
2781:      X(I)=(RAND-0.5D00)*2*PI
2782:      ENDIF
2783:      ENDDO
2784:      DO I=1,M
2785:      BB(I)=0.D00
2786:      DO J=1,M
2787:      BB(I)=BB(I)+A(I,J)*DSIN(X(J))+B(I,J)*DCOS(X(J))
2788:      ENDDO
2789:      ENDDO
2790:      F=0.D00
2791:      DO I=1,M
2792:      F=F+(AA(I)-BB(I))**2
2793:      ENDDO
2794:      RETURN
2795:      END
2796: C-----
2797:      SUBROUTINE POWELL(M,X,F)
2798:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2799:      DIMENSION X(*)
2800: C POWELL FUNCTION ; -4<= X(I) <=5; M=A MULTIPLE OF 4;
2801: C MIN F = 0.0
2802:      DO I=1,M
2803:      IF(X(I).LT.-4.D00 .OR. X(I).GT.5.D00) THEN
2804:      CALL RANDOM(RAND)
2805:      X(I)=(RAND-0.5D00)*9+.5D00
2806:      ENDIF
2807:      ENDDO
2808:      M4=M/4
2809:      F=0.D00
2810:      do I=1,m4
2811:      j=4*i
2812:      f=f+(x(j-3)+10*x(j-2))**2+5*(x(j-1)-x(j))**2+(x(j-2)-x(j-1))**4 +
2813:      & 10*(x(j-3)-x(j))**4
2814:      enddo

```

```

2815:      RETURN
2816:      END
2817: C-----
2818:      SUBROUTINE HARTMANN(M,X,F)
2819:      PARAMETER (N=4)
2820:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2821:      DIMENSION X(*),P(N,3),A(N,3),C(N)
2822: C      HARTMANN FUNCTION
2823: C      MIN F = -3.86278 APPROX : 0 < X < 1.
2824:      DATA ((P(I,J),J=1,3),I=1,4) /0.6890,0.1170,0.2673,0.4699,0.4387,
2825: & 0.7470,0.1091,0.8732,0.5547,0.0381,0.5743,0.8828/
2826:      DATA ((A(I,J),J=1,3),I=1,4) /3.0,10.0,30.0,0.1,10.0,35.0,3.0,
2827: & 10.0,30.0,0.1,10.0,35.0/
2828:      DATA (C(J),J=1,4) /1.0,1.2,3.0,3.2/
2829:      DO I=1,M
2830:      IF (X(I).LE.0.D00 .OR. X(I).GE.1.D00) THEN
2831:      CALL RANDOM(RAND)
2832:      X(I)=RAND
2833:      ENDIF
2834:      ENDDO
2835:      F=0.D00
2836:      DO I=1,N
2837:      S=0.D00
2838:      DO J=1,M
2839:      S=S+A(I,J)*(X(J)-P(I,J))**2
2840:      ENDDO
2841:      F=F+C(I)*DEXP(-S)
2842:      ENDDO
2843:      F=-F
2844:      RETURN
2845:      END
2846: C-----
2847:      SUBROUTINE COLVILLE(M,X,F)
2848:      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
2849:      DIMENSION X(*)
2850: C      COLVILLE FUNCTION ; -10<= X(I) <=10; M= 4;
2851: C      MINF(1,1,1,1)= 0.0
2852:      DO I=1,M
2853:      IF (X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
2854:      CALL RANDOM(RAND)
2855:      X(I)=(RAND-0.5D00)*20
2856:      ENDIF
2857:      ENDDO
2858:      F=100*(X(1)**2-X(2))**2 + (X(1)-1.D00)**2 + (X(3)-1.D00)**2+
2859: & 90*(X(3)**2-X(4))**2+10.1*(X(2)-1.D0)**2+ (X(4)-1.D00)**2)+
2860: & 19.8*(X(2)-1.D00)*(X(4)-1.D00)
2861:      RETURN
2862:      END

```