Global optimization of some difficult benchmark functions by cuckoo-host co-evolution meta-heuristics

Mishra, SK

11 August 2012

Global Optimization of Some Difficult Benchmark Functions by Cuckoo-Host Co-Evolution Meta-Heuristics

SK Mishra
Dept. of Economics
North-Eastern Hill University
Shillong (India) - 793022
Email: mishrasknehu@yahoo.com

I. Introduction: Global optimization endeavors to find the optima of the functions that are non-linear, non-differentiable, non-convex or multimodal, sometimes having multiple global optimum and numerous local optima, posing insurmountable difficulties before the classical methods of optimization. One encounters such problematic functions in engineering, sciences, operations research, statistics, economics, etc. Since the mid-1950s, efforts have been made to search for a suitable method that addresses the problem of global optimization of such problematic functions.

In the pre-1975 years, the works of Box (1957), Nelder and Mead (1964) and Box (1965) were remarkable. However, the invention of the "Genetic Algorithm" by Holland (1975) ushered an era of research in global optimization. Invented in the subsequent years, the "Clustering Method" of Törn (1978), the "Simulated Annealing Method" of Kirkpatrick et al. (1983) and Cerny (1985), "Tabu Search Method" of Glover (1986), the "Ant Colony Algorithm" of Dorigo (1992), the "Particle Swarm Method" of Kennedy and Eberhart (1995), the "Differential Evolution Method" of Storn and Price (1995), and the "Generalized Simulated Annealing method" of Tsallis and Stariolo (1995) are notable and effective methods of global optimization. Some other recently proposed methods such as the "Harmony Search" of Geem et al. (2001), the "Bee System" of Lŭcíc and Teodorovíc (2001), "Bee Swarm Optimization" of Karboga (2005) and Karboga and Basturk (2007), etc. also are quite promising. Teodorović et al. (2011) is a rich source of information on "Bee Colony Optimization".

Yang and Deb (2009) proposed a new method of global optimization based on the behavior of cuckoos that are parasitic in laying their eggs in the nests of other birds (such as crows) who serve as hosts to hatch their eggs to chicks. It was shown that the so-called "cuckoo search" algorithm may prove to be quite effective for global optimization. Subsequent investigations made by Yang and Deb (2010), Civicioglu and Besdok (2011), Rajabioun (2011) and Valian et al. (2011) further demonstrated that the "cuckoo search" algorithm, in its original or improved version, may be very effective. The method has been tested on a large battery of benchmark (test) functions of varied difficulty levels.

The original "cuckoo search algorithm" of Yang and Deb (2009) or its variants (or its improved versions) is based on the idea how cuckoos lay their eggs in the host nests, how, if not detected (and destroyed), the eggs are hatched by the hosts, how the cuckoo chicks later join the population of cuckoos and how a mathematical representation of all these can be used to search for the global optimum of a function. In brief, the algorithm may be conceptually summarized in the following four idealized rules (Yang and Deb, 2009):

1. Each cuckoo lays a single egg into a randomly chosen host-nest, while there are n nests;
2. The nests with better quality eggs (implying better fitness value of the optimand function), if not detected, would be hatched to be the cuckoo chicks, who would join the next generation;

3. The number of available host nests is fixed. The host can detect the alien egg with a probability [0, 1] and, if detected, it will either destroy the egg or abandon the nest so as to build a new nest elsewhere;
4. When generating new solutions ($x_i^{(t+1)}$) from the old one ($x_i^{(t)}$), Lévy flight is performed with the parameter $1 < \beta < 3$ and, thus, $x_i^{(t+1)} = x_i^{(t)} + \alpha \otimes Levy(\beta)$; for, say cuckoo i; $\alpha = O(1)$; $\otimes$ means entry-wise multiplication (or the Hadamard product). The Lévy flight endows a cuckoo with a capability to take a random walk which has a power law step length distribution with a heavy tail. It has been found (Brown et al., 2007; Pavlyukevich, 2007) that Lévy flights characterize an appropriate type random walk in many real life situations (Viswanathan et al., 1996; 1999; 2002).

II. The Objective of this Paper: In the "Cuckoo Search" algorithm, it may be noted that it has nothing to say as to how the host birds will regenerate their nests in view of the parasitic intruders (cuckoos) and how the two (the cuckoos and the hosts) will co-evolve. Thus, the host birds are immune to the experience of invasion by the parasite cuckoos. However, Davies and Brooke (1989a; 1989b) and Lotem et al. (1995) observe that co-evolution does take place and the arms race theory (Dawkins and Krebs, 1979) would suggest that, in the long run, hosts should evolve good discrimination ability (and the probability of detection as high as 65%), forcing the cuckoos to switch to a new, non-discriminating host (Davies and Brooke, 1989b; Rothstein, 1990). In view of this, in a given area, where the cuckoos and the hosts interact, the rate of rejection would increase over time. On the other hand, cuckoos also would change their strategies.

The objective of this paper is, therefore, to incorporate the co-evolutionary changes into the "Cuckoo Search" algorithm and test the efficiency of the two populations (of the cuckoos and the hosts, say crows) in finding the global optimum of some benchmark functions. This new suggested algorithm may be called the "Cuckoo-Host Co-Evolution" or CHC algorithm.

III. The Proposed CHC Algorithm: For simplicity, let there be n invaders (cuckoos) and equally many hosts (crows, say). Each invader would be represented by a point (in m-dimensional space) and similarly each host would be represented by a point (in m-dimensions). These points will be randomly generated and would lie in the domain of the function to be optimized.

Each cuckoo will take a Lévy flight and if its post-flight fitness is better than its pre-flight fitness (failing which it would not make an attempt to lay any egg in the host nest), then it will randomly choose a host-net that has not as yet been invaded by another cuckoo and the quality of the host eggs are inferior to the cuckoo egg. Its eggs, however, may be detected (with probability p) by the host and be destroyed. If not detected, however, the nestling, after being hatched in the host nest, will join the cuckoo population. Only the best n cuckoos, however, will enter into the next generation. The Lévy flight will follow the rule as given hereunder:

$$x_{ij}^{(t+1)} = x_{ij}^{t} + [\alpha^{*}(r_1 - .5)^{*}\text{Levy}(\beta)] * [(y_{ij}^{t} - x_{ij}^{t})]: \alpha = 0.0001 + (r_x)^2;\ \beta = 3/2$$

Each un-invaded host (crow) will take a Lévy flight and if its post-flight fitness is better than its pre-flight fitness, it will update itself, else it would retain its old status. The Lévy flight will follow the rule as given hereunder:

$$y_{ij}^{(t+1)} = y_{ij}^{t} + [\omega^{*}(r_2 - .5)^{*}\text{Levy}(\gamma)] + [(x_{ij}^{t} - y_{ij}^{t})]: \omega = 0.0001 + (r_y)^2;\ \gamma = 5/3$$

It may be noted that due to the smaller value of β, the cuckoos will have wider flight ranges than the host (crows) will have (Gutowski, 2001).

On completion of this process the cuckoo and the host populations would enter the next generation (iteration). At each next iteration (t), the probability of rejection (of the cuckoo eggs in the host nest) will increase under the rule as $p^{(t)}$=a+b(t/maxt) such that the probability will be almost ($a+b$) at last. In this, a could be a small number such as 0.01 and b could be 0.6 or even 0.7.

This algorithm is co-evolutionary on two accounts: first that both - the invaders (cuckoos) and the hosts (crows, say) – take levy flights in view of their cohort and themselves (y_{ij}^{t} and x_{ij}^{t}) and, secondly, that at every subsequent iteration, the probability of rejection increases. Also, since only the un-invaded hosts take Lévy flights, their strategies depend on the success of the invaders. The details are available in the appended codes.

IV. Performance of the Cuckoo-Host Co-evolution Algorithm on Some Benchmark Functions: To test the effectiveness of the proposed algorithm we have used 30 benchmark functions (Table-1). Many of the selected benchmark functions are relatively difficult to optimize (Mishra, 2010; 2006a; 2006b).

Table-1. Global Optimum of Select Benchmark Function Obtained by Differential Evolution (DE) and Cuckoo-Host Co-Evolution (CHC) Algorithms

Sl No.	Function	Dimension	Best Value Known	Best value(DE)	Best Value(CHC)
1	Giunta	2	0.0644704444	0.064470444	0.064470444
2	Easom	2	-1	-1	-1
3	Trefethen	2	-3.306869	-3.30686865	-3.30686865
4	Shubert	2	-186.730909	-186.730909	-186.730909
5	Levy-8	3	0	1.50E-32	1.50E-32
6	Hartmann	3	-3.86277761	-3.86277761	-3.86277761
7	Perm-1	4	0	4.68E-30	2.93E-09
8	Power Sum	4	0	0.003221538	1.92E-07
9	Wood	4	1.094854	1.09485393	1.09485393
10	Shekel	4	-10.4832696	-10.4832696	-10.4832696
11	Colville	4	0	0	1.44E-29
12	Hougen	5	0.298901	0.298901012	0.30768015
13	Glankwahmdee	5	737	-739.822991	-739.822991
14	Powel	8	0	1.28E-24	6.16E-07
15	ANNS XOR	9	0.959759	0.959758757	0.959758757
16	Quintic	10	0	0	0
17	Perm-2	10	0	0.017898045	5.84E-05
18	AMGM	10	0	2.79E-20	4.81E-10
19	Griewank	10	0	0	0
20	Rastrigin	10	0	0	0
21	Ackley	10	0	-4.44E-16	-4.44E-16
22	Michalewicz	10	-9.66015172	-9.66015172	-9.66015172
23	Schwefel	10	-4189.829	-4189.82887	-4189.82887
24	Paviani	10	-45.77848	-45.7784755	-45.7784755
25	Rosenbrock	10	0	5.42E-27	7.44E-16
26	Trigonometric	10	0	5.05E-28	4.85E-41
27	Eggholder	15	-12860.1768	-12593.1737	-12860.1768
28	Weierstrass	20	0	0	0
29	Trid	20	-1520	-1520	-1520
30	Keane Bump	60	Not known	-0.835835669	-0.838186072

In particular, a mention may be made of the Keane's Bump function, which makes a well known (difficult) nonlinear constrained optimization problem, hard to optimize. Mishra (2007) obtained min(Keane(60)) = -0.835835669 by the DE algorithm and min(Keane(60)) = -0.837746743 by the Repulsive Particle Swarm algorithm. The CHC algorithm obtains min(Keane(60)) = -0.838186072, which is better than both (but not optimal). The coordinates of the min(Keane(60)) are given in Table-2. However, in this regard, the performance of the CHC algorithm is remarkable.

Table-2. Coordinates of the Keane's Bump Function (Dimension = 60) obtained by CHC Algorithm									
6.286422	6.262567	3.158141	3.148193	3.134817	3.125238	3.114645	3.102346	3.092668	3.083107
3.072947	3.060029	3.048501	3.038209	3.028499	3.015386	3.006446	2.993532	2.982767	2.973307
2.959876	2.949808	2.936648	2.923654	2.911653	0.454838	0.450212	0.450305	0.448981	0.446001
0.446424	0.443195	0.442609	0.439527	0.440654	0.438790	0.436510	0.436343	0.433738	0.432823
0.431958	0.430773	0.430592	0.428734	0.426538	0.426770	0.425718	0.425047	0.422194	0.421687
0.420877	0.419390	0.418641	0.416724	0.417424	0.414845	0.412560	0.413390	0.413942	0.411132

V. Concluding Remarks: It appears that the performance of the Cuckoo-Host Co-Evolution (CHC) algorithm is comparable to the Differential Evolution algorithm, which is perhaps the most efficient algorithm for the global optimization (of continuous valued non-convex functions). The CHC algorithm requires further investigation into its behavior as well as a study for making it more efficient.

In the CHC algorithm both cuckoos and the hosts (crows) make attempts to optimize. In case of many functions (whose results have not been presented here, but they are present in the Fortran code appended here), the hosts perform better than the cuckoos, and in case of some other functions both attain the global optimum. In rare cases, however, cuckoos perform better than the crows (hosts). It is not yet understood as to the reason behind such occurrences. It requires further investigation.

References

- Box, G.E.P. (1957) "Evolutionary Operation: A Method for Increasing Industrial Productivity", Applied Statistics, 6 :81-101.
- Box, M.J. (1965) "A New Method of Constrained Optimization and a Comparison with Other Methods", Comp. Journal, 8: 42-52.
- Brown, C., Liebovitch, L. S., Glendon, R. (2007) "Lévy flights in Dobe Ju/'hoansi foraging patterns", Human Ecol., 35(129-138).
- Cerny, V. (1985) "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm", J. Opt. Theory Appl., 45(1):41-51.
- Civicioglu, P. and Besdok, E. (2011) "A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms", Artificial Intelligence Review, DOI 10.1007/s10462-011-9276-0
- Davies, N.B. and Brooke, M. de L. (1989a) "An experimental study of co-evolution between the cuckoo, Cuculus canorus, and its hosts. I. Host egg discrimination". J. Anim. Ecol., 58: 207-224.
- Davies, N.B. and Brooke, M. de L. (1989b) An experimental study of co-evolution between the cuckoo, Cuculus canorus, and its hosts. II. Host egg markings, chick discrimination and general discussion", J. Anim. Ecol., 58:225-236.
- Dawkins, R. and Krebs, J. R. (1979) "Arms races between and within species", Proc. R. Soc. Lond. Ser. B., 205:489–511.
- Dorigo, M. (1992) Optimization, Learning and Natural Algorithms, PhD thesis, Politecnico di Milano, Italie, 1992.

- Eberhart R.C. and Kennedy J. (1995) “A New Optimizer using Particle Swarm Theory”, in Proceedings Sixth Symposium on Micro Machine and Human Science, pp. 39–43. IEEE Service Center, Piscataway, NJ, 1995.
- Geem, Z.W., Kim, J.H. and Loganathan, G.V. (2001) “A New Heuristic Optimization Algorithm: Harmony Search”, Simulation, 76(2):60-68.
- Glover F. (1986) "Future Paths for Integer Programming and Links to Artificial Intelligence", Computers and Operations Research, 5:533-549.
- Gutowski, M. (2001) “Lévy flights as an underlying mechanism for global optimization algorithms”, arXiv:math-ph/0106003v1[4 Jun 2001]. http://arxiv.org/abs/math-ph/0106003v1.
- Holland, J. (1975) Adaptation in Natural and Artificial Systems, Univ. of Michigan Press, Ann Arbor, 1975.
- Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization, Technical Report-TR06, Computer Engineering Department. Erciyes University, Turkey.
- Karaboga, D. & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of Global Optimization, 39 (3), 459-471.
- Kirkpatrick, S., Gelatt, C.D. Jr., and Vecchi, M.P. (1983) "Optimization by Simulated Annealing", Science, 220 (4598):671-680.
- Lotem, A., Nakamura, H. and Zahavi, A. (1995) “Constraints on egg discrimination and cuckoo–host co-evolution”, Animal Behav., 49(5): 1185–1209
- Lŭcíc, P. and Teodorovíc, D. (2001) “Bee system: modeling combinatorial optimization transportation engineering problems by swarm intelligence”, in Preprints of the TRISTAN IV Triennial Symposium on Transportation Analysis. Sao Miguel, Azores Islands: 441–445.
- Mishra, S.K. (2010) "Performance of Differential Evolution and Particle Swarm Methods on Some Relatively Harder Multi-modal Benchmark Functions", The IUP Journal of Computational Mathematics, III(1): 7-18.
- Mishra, S.K. (2007) “Minimization of Keane's Bump Function by the Repulsive Particle Swarm and the Differential Evolution Methods”, http://mpra.ub.uni-muenchen.de/3098/1/MPRA_paper_3098.pdf
- Mishra, S.K. (2006a) "Global Optimization by Differential Evolution and Particle Swarm Methods: Evaluation on Some Benchmark Functions", Working Paper Series, Munich Personal RePEc Archive. http://mpra.ub.uni-muenchen.de/1005/1/MPRA_paper_1005.pdf.
- Mishra, S.K. (2006b) “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.
- Nelder, J.A. and Mead, R. (1964) “A Simplex Method for Function Minimization” Computer Journal, 7: 308-313.
- Pavlyukevich, I. (2007) “Lévy flights, non-local search and simulated annealing”, J. Computational Physics, 226(1830-1844).
- Rajabioun, R. (2011) “Cuckoo Optimization Algorithm”, Applied Soft Computing, 11: 5508–5518.
- Storn, R. and Price, K. (1995) "Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces": Technical Report, International Computer Science Institute, Berkley.
- Rothstein, S. I. (1990) “A model system for co-evolution: avian brood parasitism”, A. Rev. Ecol. Syst., 21:481–508.
- Teodorović, D., Davidović, T. and Šelmić, M. (2011) “Bee Colony Optimization: The Applications Survey”, http://www.mi.sanu.ac.rs/~tanjad/BCO-ACM-Trans-Ver2.pdf
- Törn, A.A and Viitanen, S. (1994) “Topographical Global Optimization using Presampled Points”, J. of Global Optimization, 5:267-276.
- Tsallis, C. and Stariolo, D.A. (1995) “Generalized Simulated Annealing”, ArXive condmat/9501047 v1 12 Jan, 1995.
- Valian, E., Mohanna, E. and Tavakoli, S. (2011) “Improved Cuckoo Search Algorithm for Global Optimization”, Int. J. Communications and Information Technology, 1(1): 31-44.
- Viswanathan, G. M., Afanasyev, V., Buldyrev, S. V., Murphy, E. J., Prince, P. A., and Stanley, H. E. (1996) “Lévy Flight Search Patterns of Wandering Albatrosses”, Nature, 381: 413–415.

- Viswanathan, G. M., Buldyrev, S. V., Havlin, S., da Luz, M. G. E., Raposo, E. P., and Stanley, H. E. (1999) "Optimizing the Success of Random Searches", Nature, 401: 911–914.
- Viswanathan, G. M., Bartumeus, F., Buldyrev, S. V., Catalan, J.,Fulco, U. L., Havlin, S., da Luz, M. G. E., Lyra, M. L., Raposo,E. P., and Stanley, H. E. (2002) "Lévy Flight Random Searches in Biological Phenomena", Physica - A, 314: 208–213.
- Yang, X. S. and Deb, S. (2009) "Cuckoo search via Lévy flights", Proceeings of World Congress on Nature & Biologically Inspired Computing (NaBIC 2009, India), IEEE Publications, USA: 210-214.
- Yang, X.S. and Deb, S. (2010) "Engineering Optimisation by Cuckoo Search", Int. J. Mathematical Modelling and Numerical Optimisation, 1(4): 330–343.

Fortran 77 Code

```
PROGRAM CUCKOO_HOST
PARAMETER(NMAX=1000, MMAX=100, MAXITER=1000000)
! NMAX = MAXIMUM NUMBER OF BIRDS TO GENERATE
! MMAX = MAXIMUM DIMENSION OR NO. OF DECISION VARIABLES
! MAXITER = MAXIMUM NO. OF ITERATIONS
PARAMETER(IPRN=1000) ! DISPLAY INTERMEDIATE RESULTS AT EVERY IPRN
IMPLICIT DOUBLE PRECISION (A-H,O-Z)
PARAMETER(FSIGN=1)!(FSIGN = -1 FOR MAXIMIZATION)
DOUBLE PRECISION LEVY ! FUNCTION LEVY(BETA) IS DOUBLE PRECISION
COMMON /RNDM/IU,IV ! TO GENERATE UNIFORM DISTR. RANDOM NUMBERS
COMMON /KFF/KF,NFCALL,FTIT ! KF IS FUNCTION CODE; NFCALL IS THE
!NUMBER OF FUNCTION CALLS; FTIT IS THE TITLE OF THE FUNCTION
CHARACTER *70 TIT(100),FTIT ! TIT IS TITLE OF FUNCTIONS, A BATTERY
! OF 100 TEST FUNCTIONS; FTIT = TITLE OF THE FUNCTION
INTEGER IU,IV! FOR GENERATING UNIFORMLY DISTRIBUTED RANDOM NUMBERS
DIMENSION CUCKOO(NMAX,MMAX),CROW(NMAX,MMAX),ICU(NMAX),ICR(NMAX)
DIMENSION A(MMAX),FCU(NMAX),FCR(NMAX)
!---------------------------------------------------------------
BETA=3/2.0D0 ! NEEDED TO GENERATE LEVY FLIGHTS (CUCKOOS)
!BETA=2.0D0 ! NEEDED TO GENERATE LEVY FLIGHTS  (CUCKOOS)
!BETA=5/3.0D0 ! NEEDED TO GENERATE LEVY FLIGHTS (CUCKOOS)
!GAMMA=3/2.0D0 ! NEEDED TO GENERATE LEVY FLIGHTS (CROWS)
GAMMA=5/3.0D0 ! NEEDED TO GENERATE LEVY FLIGHTS  (CROWS)
SCALE=10 !(SCALING OF INITIAL VALUES OF DECISION VARIABLES)
NFCALL=0  ! NO. OF FUNCTION CALLS : INITIALIZED

!---------------------------------------------------------------
! SELECT/CHOOSE THE FUNCTION TO OPTIMIZE
CALL FSELECT(KF,M,FTIT) ! CHOOSES THE FUNCTION TO OPTIMIZE
WRITE(*,*)'NO. OF CUCKOOS (EQUAL TO NO. OF CROWS) ?'
WRITE(*,*)'THIS COULD BE BETWEEN 30 AND 100, SAY.'
READ(*,*) NCU  ! NO. OF CUCKOOS (& CROWS) TO GENETE.
!NCU SHOULD BE NOT BE MORE THAN A HALF OF NMAX (NMAX > 2*NCU)
NCR=NCU ! THE CROWS ARE AS MANY AS THE CUCKOOS
WRITE(*,*)'FEED THE RANDOM NUMBER SEED'
READ(*,*) IU ! RANDOM NUMBER SEED (5 DIGITS ODD INTEGER NUMBER)
!---------------------------------------------------------------
FACTOR=SCALE ! SCALING FACTOR
```

```
!GENERATE CUCKOOS RANDOMLY AND EVALUATE
CALL RANDOM(RAND)
DO I=1,NCU
DO J=1,M
CALL RANDOM(RAND)
A(J)=(RAND-0.5)*FACTOR
CUCKOO(I,J)=A(J)
ENDDO
CALL FUNC(M,A,F)
FCU(I)=F*FSIGN
ENDDO
!GENERATE CROWS RANDOMLY AND EVALUATE
DO I=1,NCR
DO J=1,M
CALL RANDOM(RAND)
A(J)=(RAND-0.5)*FACTOR
CROW(I,J)=A(J)
ENDDO
CALL FUNC(M,A,F)
FCR(I)=F*FSIGN
ENDDO
!---------------------------------------------------------------
ICOUNT=0
!---------------------------------------------------------------
DO IT=1,MAXITER
PDET=.01+0.25*FLOAT(IT)/MAXITER ! PROBABILITY OF DETECTION
! SET ICU AND ICR TO ZERO
DO I=1,NCU
ICU(I)=0
ENDDO
DO I=1,NCR
ICR(I)=0
ENDDO
! CUCKOOS REGENERATE THEMSELVES (FLY) WITH LEVY FLIGHT
DO I=1,NCU
DO J=1,M

CALL RANDOM(RAND)
ALPHA=0.0001+(RAND)**2  ! AFFECTS THE SPEED OF CONVERGENCE
CALL RANDOM(RAND)
OMEGA=0.0001+(RAND)**2

CALL RANDOM(RC)
A(J)=CUCKOO(I,J)+ALPHA*(RC-.5)*LEVY(BETA)*(CROW(I,J)-CUCKOO(I,J))
ENDDO
CALL FUNC(M,A,F)
! A NEW S0LUTION IS ADMITTED ONLY IF IT IS BETTER
FNEW=F*FSIGN
IF(FCU(I).GT.FNEW) THEN
FCU(I)=FNEW
ICU(I)=1
DO J=1,M
CUCKOO(I,J)=A(J)
```

```
ENDDO
ENDIF
ENDDO
! TRY TO PLACE THE EGGS OF CUCKOOS INTO CROW-NESTS
DO I=1,NCU
CALL RANDOM(RAND)
IX=1+INT(NCR*RAND)
CALL RANDOM(RAND)
MK=0
IF(RAND.GT.PDET.AND.ICR(IX).EQ.0) MK=1
IF(MK.EQ.1.AND.ICU(I).EQ.1.AND.FCR(IX).GT.FCU(I)) THEN
ICR(IX)=1
ICU(I)=1
FCR(IX)=FCU(I)
DO J=1,M
CROW(IX,J)=CUCKOO(I,J)
ENDDO
ENDIF
ENDDO
! SET ICU TO ZERO
DO I=1,NCP
ICU(I)=0
ENDDO
! SET ICR TO ZERO AND CROW(I,J) TO RANDOM. ALSO FIND FITNESS
DO I=1,NCR
IF(ICR(I).NE.0) THEN
DO J=1,M
CALL RANDOM(RK)
A(J)=CROW(I,J)+OMEGA*(RK-.5)*LEVY(GAMMA)+(CUCKOO(I,J)-CROW(I,J))
ENDDO
CALL FUNC(M,A,F)
IF(FCR(I).GT.F*FSIGN) THEN
FCR(I)=F*FSIGN
DO J=1,M
CROW(I,J)=A(J)
ENDDO
ICR(I)=0
ENDIF
ENDIF
ENDDO
! ARRANGE THE CUCKOO AND CROW POPULATIONS ACCORDING TO FITNESS
CALL SORT(CUCKOO,FCU,NQ,M) ! SORT CUCKOO POPULATION
CALL SORT(CROW,FCR,NCR,M)  ! SORT CROW POPULATION

! DISPLAY RESULTS AT EVERY IPRN ITERATIONS
IF(INT(ICOUNT/IPRN).EQ.(FLOAT(ICOUNT)/IPRN))THEN
ICOUNT=0
WRITE(*,*)'CUCKOO COORDINATE VALUES'
WRITE(*,*)(CUCKOO(1,J),J=1,M)
WRITE(*,*)'-----------------------------------------------------'
WRITE(*,*)'CROW COORDINATE VALUES'
WRITE(*,*)(CROW(1,J),J=1,M)
WRITE(*,*)'-----------------------------------------------------'
```

```
      WRITE(*,*)'FITNESS OF CUCKOOS =',FCU(1), ' AND CROWS =', FCR(1)
      WRITE(*,*)'NO. OF FUNCTION CALLS =', NFCALL
      !-----------------------------------------------------------------
      ENDIF
      ICOUNT=ICOUNT+1
      ENDDO ! END OF IT LOOP
      !-----------------------------------------------------------------
      WRITE(*,*)'TOTAL NO. OF FUNCTION CALLS =',NFCALL
      WRITE(*,*)'PROGRAM ENDS. THANK YOU'
      END
C     ----------------------------------------------------------------
      SUBROUTINE NORMAL(R1,R2)
C     PROGRAM TO GENERATE N(0,1) FROM RECTANGULAR RANDOM NUMBERS
C     IT USES BOX-MULLER VARIATE TRANSFORMATION FOR THIS PURPOSE.
C     ----------------------------------------------------------------
C     ----- BOX-MULLER METHOD BY GEP BOX AND ME MULLER (1958) ---------
C     BOX, G. E. P. AND MULLER, M. E. "A NOTE ON THE GENERATION OF
C     RANDOM NORMAL DEVIATES." ANN. MATH. STAT. 29, 610-611, 1958.
C     IF U1 AND U2 ARE UNIFORMLY DISTRIBUTED RANDOM NUMBERS (0,1),
C     THEN X=[(-2*LN(U1))**.5]*(COS(2*PI*U2) IS N(0,1)
C     ALSO,  X=[(-2*LN(U1))**.5]*(SIN(2*PI*U2) IS N(0,1)
C     PI = 4*ARCTAN(1.0)= 3.14159265358979323846264338327 95
C     2*PI = 6.28318530717958647692528676655 9
C     ----------------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
C     ----------------------------------------------------------------
      CALL RANDOM(RAND) ! INVOKES RANDOM TO GENERATE UNIFORM RAND [0, 1]
      U1=RAND ! U1 IS UNIFORMLY DISTRIBUTED [0, 1]
      CALL RANDOM(RAND) ! INVOKES RANDOM TO GENERATE UNIFORM RAND [0, 1]
      U2=RAND ! U1 IS UNIFORMLY DISTRIBUTED [0, 1]
      X=DSQRT(-2.D0*DLOG(U1))
      R1=X*DCOS(U2*6.28318530717958647692528676655 9D00)
      R2=X*DSIN(U2*6.28318530717958647692528676655 9D00)
      RETURN
      END
C     ----------------------------------------------------------------
C     RANDOM NUMBER GENERATOR (UNIFORM BETWEEN 0 AND 1 - BOTH EXCLUSIVE)
      SUBROUTINE RANDOM(RAND)
       DOUBLE PRECISION  RAND
       COMMON /RNDM/IU,IV
      INTEGER IU,IV
       RANDX=REAL(RAND)
       IV=IU*65539
       IF(IV.LT.0) THEN
       IV=IV+2147483647+1
       ENDIF
       RANDX=IV
       IU=IV
       RANDX=RANDX*0.4656613E-09
       RAND= RANDX
       RETURN
```

```
      END
C     ---------------------------------------------------------------
      DOUBLE PRECISION FUNCTION LEVY(BETA)
      !GENERATING LEVY FLIGHT
      ! REFERENCE: GUTOWSKI, M. (2001) "LÉVY FLIGHTS AS AN UNDERLYING
      ! MECHANISM FOR GLOBAL OPTIMIZATION ALGORITHMS", [JUNE 2001].
      ! HTTP://ARXIV.ORG/ABS/MATH-PH/0106003V1.
      DOUBLE PRECISION BETA, RAND
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      CALL RANDOM(RAND)
      LEVY = 1.0D0/RAND**(1.0D0/BETA) - 1.0D0
      RETURN
      END
C     ---------------------------------------------------------------
      DOUBLE PRECISION FUNCTION CAUCHY(BETA)
      ! CAUCHY DISTRIBUTION
      DOUBLE PRECISION R1,R2,BETA
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      CALL NORMAL(R1,R2)
      CAUCHY=R1/R2
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE SORT(X,F,N,M)
      ! ARRANGING F(I) IN ORDER
      PARAMETER(NMAX=1000,MMAX=100)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION F(NMAX),X(NMAX,MMAX)
      DO I=1,N-1
      DO II=I+1,N
      IF(F(I).GT.F(II)) THEN
      T=F(I)
      F(I)=F(II)
      F(II)=T
      DO J=1,M
      T=X(I,J)
      X(I,J)=X(II,J)
      X(II,J)=T
      ENDDO
      ENDIF
      ENDDO
      ENDDO
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE FSELECT(KF,M,FTIT)
C     THE PROGRAM REQUIRES INPUTS FROM THE USER ON THE FOLLOWING ------
C     (1) FUNCTION CODE (KF), (2) NO. OF VARIABLES IN THE FUNCTION (M);
      CHARACTER *70 TIT(100),FTIT
      WRITE(*,*)'-----------------------------------------------------'
      DATA TIT(1)/'KF=1 NEW FUNCTION(N#1) 2-VARIABLES M=2'/
```

```
DATA TIT(2)/'KF=2 NEW FUNCTION(N#2) 2-VARIABLES M=2'/
DATA TIT(3)/'KF=3 NEW FUNCTION(N#3) 2-VARIABLES M=2'/
DATA TIT(4)/'KF=4 NEW FUNCTION(N#4) 2-VARIABLES M=2'/
DATA TIT(5)/'KF=5 NEW QUINTIC FUNCTION M-VARIABLES M=?'/
DATA TIT(6)/'KF=6 NEW NEEDLE-EYE FUNCTION (N#6) M-VARIABLES M=?'/
DATA TIT(7)/'KF=7 NEW ZERO-SUM FUNCTION (N#7) M-VARIABLES M=?'/
DATA TIT(8)/'KF=8 CORANA FUNCTION 4-VARIABLES M=4'/
DATA TIT(9)/'KF=9 MODIFIED RCOS FUNCTION 2-VARIABLES M=2'/
DATA TIT(10)/'KF=10 FREUDENSTEIN ROTH FUNCTION 2-VARIABLES M=2'/
DATA TIT(11)/'KF=11 ANNS XOR FUNCTION 9-VARIABLES M=9'/
DATA TIT(12)/'KF=12 PERM FUNCTION #1 (SET BETA) 4-VARIABLES M=4'/
DATA TIT(13)/'KF=13 PERM FUNCTION #2 (SET BETA) M-VARIABLES M=?'/
DATA TIT(14)/'KF=14 POWER-SUM FUNCTION 4-VARIABLES M=4'/
DATA TIT(15)/'KF=15 GOLDSTEIN PRICE FUNCTION 2-VARIABLES M=2'/
DATA TIT(16)/'KF=16 BUKIN 6TH FUNCTION 2-VARIABLES M=2'/
DATA TIT(17)/'KF=17 NEW FUNCTION (N#8) 2-VARIABLES M=2'/
DATA TIT(18)/'KF=18 DEFL CORRUG SPRING FUNCTION M-VARIABLES M=?'/
DATA TIT(19)/'KF=19 NEW FACTORIAL FUNCTION M-VARIABLES M=?'/
DATA TIT(20)/'KF=20 NEW DECANOMIAL FUNCTION 2-VARIABLES M=2'/
DATA TIT(21)/'KF=21 JUDGE FUNCTION 2-VARIABLES M=2'/
DATA TIT(22)/'KF=22 NEW DODECAL FUNCTION 3-VARIABLES M=3'/
DATA TIT(23)/'KF=23 NEW SUM-EQ-PROD FUNCTION 2-VARIABLES M=2'/
DATA TIT(24)/'KF=24 NEW AM-EQ-GM FUNCTION M-VARIABLES M=?'/
DATA TIT(25)/'KF=25 YAO-LIU FUNCTION#2 M-VARIABLES M=?'/
DATA TIT(26)/'KF=26 YAO-LIU FUNCTION#3 M-VARIABLES M=?'/
DATA TIT(27)/'KF=27 YAO-LIU FUNCTION#4 M-VARIABLES M=?'/
DATA TIT(28)/'KF=28 YAO-LIU FUNCTION#6 M-VARIABLES M=?'/
DATA TIT(29)/'KF=29 YAO-LIU FUNCTION#7 M-VARIABLES M=?'/
DATA TIT(30)/'KF=30 YAO-LIU FUNCTION#12 M-VARIABLES M=?'/
DATA TIT(31)/'KF=31 YAO-LIU FUNCTION#13 M-VARIABLES M=?'/
DATA TIT(32)/'KF=32 YAO-LIU FUNCTION#14 2-VARIABLES M=2'/
DATA TIT(33)/'KF=33 YAO-LIU FUNCTION#15 4-VARIABLES M=4'/
DATA TIT(34)/'KF=34 WOOD FUNCTION : 4-VARIABLES M=4'/
DATA TIT(35)/'KF=35 FENTON-EASON FUNCTION : 2-VARIABLES M=2'/
DATA TIT(36)/'KF=36 HOUGEN FUNCTION : 5-VARIABLES M=5'/
DATA TIT(37)/'KF=37 GIUNTA FUNCTION : 2-VARIABLES M=2'/
DATA TIT(38)/'KF=38 EGGHOLDER FUNCTION : M-VARIABLES M=?'/
DATA TIT(39)/'KF=39 TRID FUNCTION : M-VARIABLES M=?'/
DATA TIT(40)/'KF=40 GRIEWANK FUNCTION : M-VARIABLES M=?'/
DATA TIT(41)/'KF=41 WEIERSTRASS FUNCTION : M-VARIABLES M=?'/
DATA TIT(42)/'KF=42 LEVY-3 FUNCTION : 2-VARIABLES M=2'/
DATA TIT(43)/'KF=43 LEVY-5 FUNCTION : 2-VARIABLES M=2'/
DATA TIT(44)/'KF=44 LEVY-8 FUNCTION : 3-VARIABLES M=3'/
DATA TIT(45)/'KF=45 RASTRIGIN FUNCTION : M-VARIABLES M=?'/
DATA TIT(46)/'KF=46 ACKLEY FUNCTION : M-VARIABLES M=?'/
DATA TIT(47)/'KF=47 MICHALEWICZ FUNCTION : M-VARIABLES M=?'/
DATA TIT(48)/'KF=48 SCHWEFEL FUNCTION : M-VARIABLES M=?'/
DATA TIT(49)/'KF=49 SHUBERT FUNCTION : 2-VARIABLES M=2'/
DATA TIT(50)/'KF=50 DIXON-PRICE FUNCTION : M-VARIABLES M=?'/
DATA TIT(51)/'KF=51 SHEKEL FUNCTION : 4-VARIABLES M=4'/
DATA TIT(52)/'KF=52 PAVIANI FUNCTION : 10-VARIABLES M=10'/
DATA TIT(53)/'KF=53 BRANIN FUNCTION#1 : 2-VARIABLES M=2'/
DATA TIT(54)/'KF=54 BRANIN FUNCTION#2 : 2-VARIABLES M=2'/
```

```
      DATA TIT(55)/'KF=55 BOHACHEVSKY FUNCTION#1 : 2-VARIABLES M=2'/
      DATA TIT(56)/'KF=56 BOHACHEVSKY FUNCTION#2 : 2-VARIABLES M=2'/
      DATA TIT(57)/'KF=57 BOHACHEVSKY FUNCTION#3 : 2-VARIABLES M=2'/
      DATA TIT(58)/'KF=58 EASOM FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(59)/'KF=59 ROSENBROCK FUNCTION : M-VARIABLES M=?'/
      DATA TIT(60)/'KF=60 CROSS-LEGGED TABLE FUNCTION:2-VARIABLES M=2'/
      DATA TIT(61)/'KF=61 CROSS FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(62)/'KF=62 CROSS-IN-TRAY FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(63)/'KF=63 CROWNED CROSS FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(64)/'KF=64 TT-HOLDER FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(65)/'KF=65 HOLDER-TABLE FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(66)/'KF=66 CARROM-TABLE FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(67)/'KF=67 PENHOLDER FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(68)/'KF=68 BIRD FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(69)/'KF=69 CHICHINADZE FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(70)/'KF=70 MCCORMICK FUNCTION : 2-VARIABLES M=2'/
      DATA TIT(71)/'KF=71 GLANKWAHMDEE FUNCTION : 5-VARIABLES M=5'/
      DATA TIT(72)/'KF=72 FLETCHER-POWELL FUNCTION : M-VARIABLES M=?'/
      DATA TIT(73)/'KF=73 POWELL FUNCTION: M-VARIABLES M (MULT OF 4)=?'/
      DATA TIT(74)/'KF=74 HARTMANN FUNCTION: 3-VARIABLES M=3'/
      DATA TIT(75)/'KF=75 COLVILLE FUNCTION: 4-VARIABLES M=4'/
      DATA TIT(76)/'KF=76 HIMMELBLAU FUNCTION: 2-VARIABLES M=2'/
      DATA TIT(77)/'KF=77 BEALE FUNCTION: 2-VARIABLES M=2'/
      DATA TIT(78)/'KF=78 BOOTH FUNCTION: 2-VARIABLES M=2'/
      DATA TIT(79)/'KF=79 HUMP FUNCTION: 2-VARIABLES M=2'/
      DATA TIT(80)/'KF=80 MATYAS FUNCTION: 2-VARIABLES M=2'/
      DATA TIT(81)/'KF=81 MISHRA_1 FUNCTION: M-VARIABLES M=?'/
      DATA TIT(82)/'KF=82 MISHRA_2 FUNCTION: M-VARIABLES M=?'/
      DATA TIT(83)/'KF=83 ZAKHAROV FUNCTION: M-VARIABLES M=?'/
      DATA TIT(84)/'KF=84 MULTIMOD FUNCTION: 2-VARIABLES M=2'/
      DATA TIT(85)/'KF=85 NONLINEAR FUNCTION: M-VARIABLES M=?'/
      DATA TIT(86)/'KF=86 QUADRATIC FUNCTION: 2-VARIABLES M=2'/
      DATA TIT(87)/'KF=87 TRIGON FUNCTION: M-VARIABLES M=?'/
      DATA TIT(88)/'KF=88 WAVY-1 FUNCTIONS: M-VARIABLES M=?'/
      DATA TIT(89)/'KF=89 WAVY-2 FUNCTIONS: M-VARIABLES M=?'/
      DATA TIT(90)/'KF=90 TREFETHEN FUNCTION: 2-VARIABLES M=2'/
      DATA TIT(91)/'KF=91 LINPROG1 FUNCTION: 2-VARIABLES M=2'/
      DATA TIT(92)/'KF=92 LINPROG2 FUNCTION: 3-VARIABLES M=3'/
      DATA TIT(93)/'KF=93 NEWF FUNCTION: 1-VARIABLES M=1'/
      DATA TIT(94)/'KF=94 NEWG FUNCTION: 4-VARIABLES M=4'/
      DATA TIT(95)/'KF=95 F8F2 FUNCTION: M-VARIABLES M=?'/
      DATA TIT(96)/'KF=96 KEANE BUMP FUNCTION: M-VARIABLES M=?'/

C     ---------------------------------------------------------------
      DO I=1,96
      WRITE(*,*)TIT(I)
      ENDDO
      WRITE(*,*)'-----------------------------------------------------'
      WRITE(*,*)'FUNCTION CODE [KF] AND NO. OF VARIABLES [M] ?'
      READ(*,*) KF,M
      FTIT=TIT(KF) ! STORE THE NAME OF THE CHOSEN FUNCTION IN FTIT
      RETURN
      END
```

```
C     ---------------------------------------------------------------
      SUBROUTINE FUNC(M,X,F)
C     TEST FUNCTIONS FOR GLOBAL OPTIMIZATION PROGRAM
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      COMMON /KFF/KF,NFCALL,FTIT
      INTEGER IU,IV
      DIMENSION X(*)
      CHARACTER *70 FTIT
      PI=4.D+00*DATAN(1.D+00)! DEFINING THE VALUE OF PI
      NFCALL=NFCALL+1 ! INCREMENT TO NUMBER OF FUNCTION CALLS
C     KF IS THE CODE OF THE TEST FUNCTION
C     ---------------------------------------------------------------
      IF(KF.EQ.1) THEN
C     FUNCTION #1 MIN = -0.18467 APPROX AT (-8.4666, -10) APPROX
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F=DABS(DCOS(DSQRT(DABS(X(1)**2+X(2)))))**0.5 +0.01*X(1)+.01*X(2)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.2) THEN
C     FUNCTION #2 MIN = -0.199409 APPROX AT (-9.94112, -10) APPROX
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F=DABS(DSIN(DSQRT(DABS(X(1)**2+X(2)))))**0.5 +0.01*X(1)+.01*X(2)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.3) THEN
C     FUNCTION #3 MIN = -1.01983 APPROX AT (-1.98682, -10.00000) APPROX
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F1=DSIN(( DCOS(X(1))+DCOS(X(2)) )**2)**2
      F2=DCOS(( DSIN(X(1))+DSIN(X(2)) )**2)**2
      F=(F1+F2+X(1))**2 ! IS MULTIMODAL
      F=F+ 0.01*X(1)+0.1*X(2) ! MAKES UNIMODAL
      RETURN
```

```
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.4) THEN
C     FUNCTION  #4 MIN = -2.28395 APPROX AT (2.88631, 1.82326) APPROX
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F1=DSIN((DCOS(X(1))+DCOS(X(2)))**2)**2
      F2=DCOS((DSIN(X(1))+DSIN(X(2)))**2)**2
      F3=-DLOG((F1-F2+X(1))**2 )
      F=F3+0.1D00*(X(1)-1.D00)**2+0.1D00*(X(2)-1.D00)**2
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.5) THEN
C     QUINTIC FUNCTION:GLOBAL MINIMA,EXTREMELY DIFFICULT TO OPTIMIZE
C     MIN VALUE = 0 AT PERMUTATION OF (2, 2,..., 2, -1, -1, ..., -1,
C     -0.402627941) GIVES MIN F = 0.
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      CALL QUINTIC(M,F,X)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.6) THEN
C     NEEDLE-EYE FUNCTION M=>1;
C     MIN = 1 IF ALL ABS(X) ARE SMALLER THAN THE EYE
C     SMALLER THE VALUE OF ZZ, MORE DIFFICULT TO ENTER THE EYE
C     LARGER THE VALUE OF M, MORE DIFFICULT TO FIND THE OPTIMUM
      F=0.D00
      EYE=0.000001D00
      FP=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.EYE) THEN
      FP=1.D00
       F=F+100.D00+DABS(X(I))
      ELSE
      F=F+1.D00
      ENDIF
      ENDDO
      IF(FP.EQ.0.D00) F=F/M
      RETURN
      ENDIF
C     ---------------------------------------------------------------
```

```
      IF(KF.EQ.7) THEN
C     ZERO SUM FUNCTION : MIN  = 0 AT SUM(X(I))=0
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      SUM=0.D00
      DO I=1,M
      SUM=SUM+X(I)
      ENDDO
      IF(SUM.NE.0.D00) F=1.D00+(10000*DABS(SUM))**0.5
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.8) THEN
C     CORANA FUNCTION : MIN = 0 AT (0, 0, 0, 0) APPROX
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.1000.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*2000
      ENDIF
      ENDDO
      DO J=1,M
       IF(J.EQ.1) DJ=1.D00
       IF(J.EQ.2) DJ=1000.D00
       IF(J.EQ.3) DJ=10.D00
       IF(J.EQ.4) DJ=100.D00
          ISGNXJ=1
          IF(X(J).LT.0.D00) ISGNXJ=-1
          ZJ=(DABS(X(J)/0.2D00)+0.49999)*ISGNXJ*0.2D00
          ISGNZJ=1
          IF(ZJ.LT.0.D00) ISGNZJ=-1
          IF(DABS(X(J)-ZJ).LT.0.05D00) THEN
          F=F+0.15D00*(ZJ-0.05D00*ISGNZJ)**2 * DJ
          ELSE
          F=F+DJ*X(J)**2
          ENDIF
      ENDDO
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.9) THEN
C     MODIFIED RCOS FUNCTION MIN=-0.179891 AT (-3.196989, 12.52626)APPRX
      F=0.D00
      IF(X(1).LT.-5.D00 .OR. X(1).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(1)=RAND*15.D00 -5.D00
      ENDIF
      IF(X(2).LT.0.D00 .OR. X(2).GT.15.D00) THEN
```

```
      CALL RANDOM(RAND)
      X(2)=RAND*15.D00
      ENDIF
      CA=1.D00
      CB=5.1/(4*PI**2)
      CC=5.D00/PI
      CD=6.D00
      CE=10.D00
      CF=1.0/(8*PI)
      F1=CA*(X(2)-CB*X(1)**2+CC*X(1)-CD)**2
      F2=CE*(1.D00-CF)*DCOS(X(1))*DCOS(X(2))
      F3=DLOG(X(1)**2+X(2)**2+1.D00)
      F=-1.0/(F1+F2+F3+CE)
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.10) THEN
C     FREUDENSTEIN ROTH  FUNCTION : MIN  = 0 AT (5, 4)
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F1=(-13.D00+X(1)+((5.D00-X(2))*X(2)-2)*X(2))**2
      F2=(-29.D00+X(1)+((X(2)+1.D00)*X(2)-14.D00)*X(2))**2
      F=F1+F2
      RETURN
      ENDIF
C     ----------------------------------------------------------------
      IF(KF.EQ.11) THEN
C     ANNS XOR FUNCTION (PARSOPOULOS, KE, PLAGIANAKOS, VP, MAGOULAS, GD
C     AND VRAHATIS, MN "STRETCHING TECHNIQUE FOR OBTAINING GLOBAL
C     MINIMIZERS THROUGH PARTICLE SWARM OPTIMIZATION")
C     MIN=0.9597588 FOR X=(1, -1, 1, -1, -1, 1, 1, -1, 0.421134) APPROX
C     OBTAINED BY DIFFERENTIAL EVOLUTION PROGRAM
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.1.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*2
      ENDIF
      ENDDO
      F11=X(7)/(1.D00+DEXP(-X(1)-X(2)-X(5)))
      F12=X(8)/(1.D00+DEXP(-X(3)-X(4)-X(6)))
      F1=(1.D00+DEXP(-F11-F12-X(9)))**(-2)
      F21=X(7)/(1.D00+DEXP(-X(5)))
      F22=X(8)/(1.D00+DEXP(-X(6)))
      F2=(1.D00+DEXP(-F21-F22-X(9)))**(-2)
      F31=X(7)/(1.D00+DEXP(-X(1)-X(5)))
      F32=X(8)/(1.D00+DEXP(-X(3)-X(6)))
      F3=(1.D00-(1.D00+DEXP(-F31-F32-X(9)))**(-1))**2
```

```
      F41=X(7)/(1.D00+DEXP(-X(2)-X(5)))
      F42=X(8)/(1.D00+DEXP(-X(4)-X(6)))
      F4=(1.D00-(1.D00+DEXP(-F41-F42-X(9)))**(-1))**2
      F=F1+F2+F3+F4
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.12) THEN
C     PERM FUNCTION #1 MIN = 0 AT (1, 2, 3, 4)
C     BETA => 0. CHANGE IF NEEDED. SMALLER BETA RAISES DIFFICULY
C     FOR BETA=0, EVERY PERMUTED SOLUTION IS A GLOBAL MINIMUM
      BETA=50.D00
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.M) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*2*M
      ENDIF
      ENDDO
      DO K=1,M
      SUM=0.D00
      DO I=1,M
      SUM=SUM+(I**K+BETA)*((X(I)/I)**K-1.D00)
      ENDDO
      F=F+SUM**2
      ENDDO
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.13) THEN
C     PERM FUNCTION #2 MIN = 0 AT (1/1, 1/2, 1/3, 1/4,..., 1/M)
C     BETA => 0. CHANGE IF NEEDED. SMALLER BETA RAISES DIFFICULY
C     FOR BETA=0, EVERY PERMUTED SOLUTION IS A GLOBAL MINIMUM
      BETA=10.D00
      DO I=1,M
      IF(DABS(X(I)).GT.1.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*2
      ENDIF
      SGN=X(I)/DABS(X(I))
      ENDDO
      F=0.D00
      DO K=1,M
      SUM=0.D00
      DO I=1,M
      SUM=SUM+(I+BETA)*(X(I)**K-(1.D00/I)**K)
      ENDDO
      F=F+SUM**2
      ENDDO
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.14) THEN
```

```
C     POWER SUM FUNCTION; MIN = 0 AT PERM(1,2,2,3) FOR B=(8,18,44,114)
C     0 =< X <=4
      F=0.D00
      DO I=1,M
C     ANY PERMUTATION OF (1,2,2,3) WILL GIVE MIN = ZERO
      IF(X(I).LT.0.D00 .OR. X(I).GT.4.D00) THEN
      CALL RANDOM(RAND)
      X(I)=RAND*4
      ENDIF
      ENDDO
      DO K=1,M
      SUM=0.D00
      DO I=1,M
      SUM=SUM+X(I)**K
      ENDDO
      IF(K.EQ.1) B=8.D00
      IF(K.EQ.2) B=18.D00
      IF(K.EQ.3) B=44.D00
      IF(K.EQ.4) B=114.D00
      F=F+(SUM-B)**2
      ENDDO
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.15) THEN
C     GOLDSTEIN PRICE  FUNCTION  : MIN VALUE = 3 AT (0, -1)
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*20
      ENDIF
      ENDDO
      F11=(X(1)+X(2)+1.D00)**2
      F12=(19.D00-14*X(1)+ 3*X(1)**2-14*X(2)+ 6*X(1)*X(2)+ 3*X(2)**2)
      F1=1.00+F11*F12
      F21=(2*X(1)-3*X(2))**2
      F22=(18.D00-32*X(1)+12*X(1)**2+48*X(2)-36*X(1)*X(2)+27*X(2)**2)
      F2=30.D00+F21*F22
      F= (F1*F2)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.16) THEN
C     BUKIN'S 6TH FUNCTION  MIN = 0 FOR (-10, 1)
C     -15. LE. X(1) .LE. -5 AND -3 .LE. X(2) .LE. 3
       IF(X(1).LT. -15.D00 .OR. X(1) .GT. -5.D00) THEN
       CALL RANDOM(RAND)
       X(1)=-(RAND*10+5.D00)
       ENDIF
       IF(DABS(X(2)).GT.3.D00) THEN
       CALL RANDOM(RAND)
       X(2)=(RAND-.5D00)*6
```

```
      ENDIF
      F=100.D0*DSQRT(DABS(X(2)-0.01D0*X(1)**2))+ 0.01D0*DABS(X(1)+10.D0)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.17) THEN
C     NEW N#8 FUNCTION (MULTIPLE GLOBAL MINIMA)
C     MIN VALUE = -1 AT (AROUND .7 AROUND, 0.785 APPROX)
      F=0.D00
      DO I=1,M
      IF(X(I).LT.0.5D00 .OR. X(I).GT.1.D00) THEN
      CALL RANDOM(RAND)
      X(I)=RAND/2.D00
      ENDIF
      ENDDO
      F=-DEXP(-DABS(DLOG(.001D00+DABS((DSIN(X(1)+X(2))+DSIN(X(1)-X(2))+
     & (DCOS(X(1)+X(2))*DCOS(X(1)-X(2))+.001))**2)+
     & .01D00*(X(2)-X(1))**2)))
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.18) THEN
C     DEFLECTED CORRUGATED SPRING FUNCTION
C     MIN VALUE = -1 AT (5, 5, ..., 5) FOR ANY K AND ALPHA=5; M VARIABLE
      CALL DCS(M,F,X)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.19) THEN
C     FACTORIAL FUNCTION, MIN =0 AT X=(1,2,3,...,M)
      CALL FACTOR1(M,F,X)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.20) THEN
C     DECANOMIAL FUNCTION, MIN =0 AT X=(2, -3)
      DO I=1,M
      IF(DABS(X(I)).GT.4.D00) THEN
      CALL RANDOM(RAND)
      X(I)= (RAND-0.5D00)*8
      ENDIF
      ENDDO
      CALL DECANOM(M,F,X)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.21) THEN
C     JUDGE'S FUNCTION  F(0.864, 1.23) = 16.0817; M=2
      CALL JUDGE(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.22) THEN
```

```
C    DODECAL FUNCTION
     CALL DODECAL(M,F,X)
     RETURN
     ENDIF
C    ---------------------------------------------------------------
     IF(KF.EQ.23) THEN
C    WHEN X(1)*X(2)=X(1)*X(2) ? M=2
     CALL SEQP(M,F,X)
     RETURN
     ENDIF
C    ---------------------------------------------------------------
     IF(KF.EQ.24) THEN
C    WHEN ARITHMETIC MEAN = GEOMETRIC MEAN ? : M =>1
     CALL AMGM(M,F,X)
     RETURN
     ENDIF
C    ---------------------------------------------------------------
     IF(KF.EQ.25) THEN
C    M =>2
     CALL FUNCT2(M,F,X)
     RETURN
     ENDIF
C    ---------------------------------------------------------------
     IF(KF.EQ.26) THEN
C    M =>2
     CALL FUNCT3(M,F,X)
     RETURN
     ENDIF
C    ---------------------------------------------------------------
     IF(KF.EQ.27) THEN
C    M =>2
     CALL FUNCT4(M,F,X)
     RETURN
     ENDIF
C    ---------------------------------------------------------------
     IF(KF.EQ.28) THEN
C    M =>2
     CALL FUNCT6(M,F,X)
     RETURN
     ENDIF
C    ---------------------------------------------------------------
     IF(KF.EQ.29) THEN
C    M =>2
     CALL FUNCT7(M,F,X)
     RETURN
     ENDIF
C    ---------------------------------------------------------------
     IF(KF.EQ.30) THEN
C    M =>2
     CALL FUNCT12(M,F,X)
     RETURN
     ENDIF
C    ---------------------------------------------------------------
```

```
      IF(KF.EQ.31) THEN
C     M =>2
      CALL FUNCT13(M,F,X)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.32) THEN
C     M =2
      CALL FUNCT14(M,F,X)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.33) THEN
C     M =4
      CALL FUNCT15(M,F,X)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.34) THEN
C     WOOD FUNCTION : F MIN  : M=4
      CALL WOOD(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.35) THEN
C     FENTON & EASON FUNCTION :   : M=2
      CALL FENTONEASON(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.36) THEN
C     HOUGEN FUNCTION 5 VARIABLES : M =3
      CALL HOUGEN(X,M,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.37) THEN
C     GIUNTA FUNCTION 2 VARIABLES  :M =2
      CALL GIUNTA(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.38) THEN
C     EGGHOLDER FUNCTION M VARIABLES
      CALL EGGHOLD(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.39) THEN
C     TRID FUNCTION M VARIABLES
      CALL TRID(M,X,F)
      RETURN
      ENDIF
```

```
C   ---------------------------------------------------------------
    IF(KF.EQ.40) THEN
C   GRIEWANK FUNCTION M VARIABLES
    CALL GRIEWANK(M,X,F)
    RETURN
    ENDIF
C   ---------------------------------------------------------------
    IF(KF.EQ.41) THEN
C   WEIERSTRASS FUNCTION M VARIABLES
    CALL WEIERSTRASS(M,X,F)
    RETURN
    ENDIF
C   ---------------------------------------------------------------
    IF(KF.EQ.42) THEN
C   LEVY-3 FUNCTION 2 VARIABLES
    CALL LEVY3(M,X,F)
    RETURN
    ENDIF
C   ---------------------------------------------------------------
    IF(KF.EQ.43) THEN
C   LEVY-5 FUNCTION 2 VARIABLES
    CALL LEVY5(M,X,F)
    RETURN
    ENDIF
C   ---------------------------------------------------------------
    IF(KF.EQ.44) THEN
C   LEVY-8 FUNCTION 3 VARIABLES
    CALL LEVY8(M,X,F)
    RETURN
    ENDIF
C   ---------------------------------------------------------------
    IF(KF.EQ.45) THEN
C   RASTRIGIN FUNCTION M VARIABLES
    CALL RASTRIGIN(M,X,F)
    RETURN
    ENDIF
C   ---------------------------------------------------------------
    IF(KF.EQ.46) THEN
C   ACKLEY FUNCTION M VARIABLES
    CALL ACKLEY(M,X,F)
    RETURN
    ENDIF
C   ---------------------------------------------------------------
    IF(KF.EQ.47) THEN
C   MICHALEWICZ FUNCTION M VARIABLES
    CALL MICHALEWICZ(M,X,F)
    RETURN
    ENDIF
C   ---------------------------------------------------------------
    IF(KF.EQ.48) THEN
C   SCHWEFEL FUNCTION M VARIABLES
    CALL SCHWEFEL(M,X,F)
    RETURN
```

```
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.49) THEN
C     SHUBERT FUNCTION 2 VARIABLES
      CALL SHUBERT(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.50) THEN
C     DIXON AND PRICE FUNCTION M VARIABLES
      CALL DIXPRICE(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.51) THEN
C     SHEKEL FUNCTION 4 VARIABLES
      CALL SHEKEL(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.52) THEN
C     PAVIANI FUNCTION 10 VARIABLES
      CALL PAVIANI(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.53) THEN
C     BRANIN FUNCTION#1 2 VARIABLES
      CALL BRANIN1(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.54) THEN
C     BRANIN FUNCTION#2 2 VARIABLES
      CALL BRANIN2(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.55) THEN
C     BOHACHEVSKY FUNCTION#1 2 VARIABLES
      CALL BOHACHEVSKY1(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.56) THEN
C     BOHACHEVSKY FUNCTION#2 2 VARIABLES
      CALL BOHACHEVSKY2(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.57) THEN
C     BOHACHEVSKY FUNCTION#3 2 VARIABLES
      CALL BOHACHEVSKY3(M,X,F)
```

```
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.58) THEN
C     EASOM FUNCTION#3 2 VARIABLES
      CALL EASOM(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.59) THEN
C     ROSENBROCK FUNCTION  M VARIABLES
      CALL ROSENBROCK(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.60) THEN
C     CROSS-LEGGED TABLE FUNCTION : 2 VARIABLES
      CALL CROSSLEG(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.61) THEN
C     CROSS FUNCTION : 2 VARIABLES
      CALL CROSS(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.62) THEN
C     CROSS-IN-TRAY FUNCTION : 2 VARIABLES
      CALL CROSSINTRAY(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.63) THEN
C     CROWNED-CROSS FUNCTION : 2 VARIABLES
      CALL CROWNEDCROSS(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.64) THEN
C     TT-HOLDER FUNCTION : 2 VARIABLES, MIN F([+/-]1.5706, 0)= -10.8723
      CALL TTHOLDER(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.65) THEN
C     HOLDER-TABLE FUNCTION : 2 VARIABLES
C     MIN F ([+/-] 9.64617, [+/-] 9.64617) APPROX = -26.92034 APPROX
      CALL HOLDERTABLE(M,X,F)
      RETURN
      ENDIF
C     ------------------------------------------------------------------
      IF(KF.EQ.66) THEN
```

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

```
      IF(KF.EQ.74) THEN
C     HARTMANN FUNCTION
      CALL HARTMANN(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.75) THEN
C     COVILLE FUNCTION
      CALL COLVILLE(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.76) THEN
C     HIMMELBLAU FUNCTION
      CALL HIMMELBLAU(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.77) THEN
C     BEALE FUNCTION
      CALL BEALE(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.78) THEN
C     BOOTH FUNCTION
      CALL BOOTH(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.79) THEN
C     HUMP FUNCTION
      CALL HUMP(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.80) THEN
C     MATYAS FUNCTION
      CALL MATYAS(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.81) THEN
C     MISHRA_1 FUNCTION
      CALL MISHRA_1(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.82) THEN
C     MISHRA_2 FUNCTION
      CALL MISHRA_2(M,X,F)
      RETURN
      ENDIF
```

```
C     ---------------------------------------------------------------
      IF(KF.EQ.83) THEN
C     ZAKHAROV FUNCTION
      CALL ZAKHAROV(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.84) THEN
C     MULTOMOD FUNCTION
      CALL MULTIMOD(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.85) THEN
C     NONLINEAR  FUNCTION
      CALL NONLIN(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.86) THEN
C     QUADRATIC  FUNCTION
      CALL QUADRATIC(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.87) THEN
C     TRIGON FUNCTION
      CALL TRIGON(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.88) THEN
C     WAVY-1 FUNCTION
      CALL  WAVY1(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.89) THEN
C     WAVY-2 FUNCTION
      CALL  WAVY2(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.90) THEN
C     TREFETHEN FUNCTION
      CALL  TREFETHEN(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.91) THEN
C     LINPROG1 FUNCTION
      CALL  LINPROG1(M,F,X)
      RETURN
```

```
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.92) THEN
C     LINPROG2 FUNCTION
      CALL  LINPROG2(M,F,X)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.93) THEN
C     NEWF FUNCTION
      CALL  NEWF(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.94) THEN
C     NEWG FUNCTION
      CALL  NEWG(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.95) THEN
C     F8F2 FUNCTION
      CALL  F8F2(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
      IF(KF.EQ.96) THEN
C     KEANE BUMP FUNCTION
      CALL  KEANEBUMP(M,X,F)
      RETURN
      ENDIF
C     ---------------------------------------------------------------
C     IF(KF.EQ.99) THEN
C     *****  FUNCTION
C     CALL SUBROUTINE(M,X,F)
C     RETURN
C     ENDIF
C     ===============================================================
      WRITE(*,*)'FUNCTION NOT DEFINED. PROGRAM ABORTED'
      STOP
      END
C     >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
      SUBROUTINE DCS(M,F,X)
C     FOR DEFLECTED CORRUGATED SPRING FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*),C(100)
C     OPTIMAL VALUES OF (ALL) X ARE ALPHA , AND K IS ONLY FOR SCALING
C     OPTIMAL VALUE OF F IS -1. DIFFICULT TO OPTIMIZE FOR LARGER M.
      DATA K,ALPHA/5,5.D00/ ! K AND ALPHA COULD TAKE ON ANY OTHER VALUES
      R2=0.D00
      DO I=1,M
      C(I)=ALPHA
      R2=R2+(X(I)-C(I))**2
```

```
      ENDDO
      R=DSQRT(R2)
      F=-DCOS(K*R)+0.1D00*R2
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE QUINTIC(M,F,X)
C     QUINTIC FUNCTION: GLOBAL MINIMA,EXTREMELY DIFFICULT TO OPTIMIZE
C     MIN VALUE = 0 AT PERM( -1, -0.402627941, 2)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      DO I=1,M
      F=F+DABS(X(I)**5-3*X(I)**4+4*X(I)**3+2*X(I)**2-10*X(I)-4.D00)
      ENDDO
      F=1000*F
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE FACTOR1(M,F,X)
C     FACTORIAL FUNCTION; MIN (1, 2, 3, ...., M) = 0
C     FACT = FACTORIAL(M) = 1 X 2 X 3 X 4 X .... X M
C     FIND X(I), I=1,2,...,M SUCH THAT THEIR PRODUCT IS EQUAL TO FACT.
C     LARGER THE VALUE OF M (=>8) OR SO, HARDER IS THE PROBLEM
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      FACT=1.D00
      P=1.D00
      DO I=1,M
      FACT=FACT*I
      P=P*X(I)
      F=F+DABS(P-FACT)**2
      ENDDO
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE DECANOM(M,F,X)
C     DECANOMIAL FUNCTION; MIN (2, -3) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F1= DABS(X(1)**10-20*X(1)**9+180*X(1)**8-960*X(1)**7+
     & 3360*X(1)**6-8064*X(1)**5+13340*X(1)**4-15360*X(1)**3+
     & 11520*X(1)**2-5120*X(1)+2624.D00)
      F2= DABS(X(2)**4+12*X(2)**3+54*X(2)**2+108*X(2)+81.D00)
      F=0.001D00*(F1+F2)**2
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE JUDGE(M,X,F)
      PARAMETER (N=20)
C     THIS SUBROUTINE IS FROM THE EXAMPLE IN JUDGE ET AL., THE THEORY
C     AND PRACTICE OF ECONOMETRICS, 2ND ED., PP. 956-7. THERE ARE TWO
```

```
C     OPTIMA: F(0.86479,1.2357)=16.0817307 (WHICH IS THE GLOBAL MINUMUM)
C     AND F(2.35,-0.319)=20.9805 (WHICH IS LOCAL). ADAPTED FROM BILL
C     GOFFE'S SIMMAN (SIMULATED ANNEALING) PROGRAM
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION Y(N), X2(N), X3(N), X(*)
       DATA (Y(I),I=1,N)/4.284,4.149,3.877,0.533,2.211,2.389,2.145,
     & 3.231,1.998,1.379,2.106,1.428,1.011,2.179,2.858,1.388,1.651,
     & 1.593,1.046,2.152/
       DATA (X2(I),I=1,N)/.286,.973,.384,.276,.973,.543,.957,.948,.543,
     & .797,.936,.889,.006,.828,.399,.617,.939,.784,.072,.889/
       DATA (X3(I),I=1,N)/.645,.585,.310,.058,.455,.779,.259,.202,.028,
     & .099,.142,.296,.175,.180,.842,.039,.103,.620,.158,.704/

      F=0.D00
      DO I=1,N
      F=F+(X(1) + X(2)*X2(I) + (X(2)**2)*X3(I) - Y(I))**2
      ENDDO
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE DODECAL(M,F,X)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     DODECAL POLYNOMIAL MIN F(1,2,3)=0
      DO I=1,M
      IF(DABS(X(I)).GT.5.D0) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*10
      ENDIF
      ENDDO
      F=0.D00
      F1=2*X(1)**3+5*X(1)*X(2)+4*X(3)-2*X(1)**2*X(3)-18.D00
      F2=X(1)+X(2)**3+X(1)*X(2)**2+X(1)*X(3)**2-22.D00
      F3=8*X(1)**2+2*X(2)*X(3)+2*X(2)**2+3*X(2)**3-52.D00
      F=(F1*F3*F2**2+F1*F2*F3**2+F2**2+(X(1)+X(2)-X(3))**2)**2
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE SEQP(M,F,X)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     FOR WHAT VALUES X(1)+X(2)=X(1)*X(2) ? ANSWER: FOR (0,0) AND (2,2)
C     WHILE X(1), X(2) ARE INTEGERS.
      X(1)=INT(X(1)) ! X(1) CONVERTED TO INTEGER
      X(2)=INT(X(2)) ! X(2) CONVERTED TO INTEGER

      F1=X(1)+X(2)
      F2=X(1)*X(2)
      F=(F1-F2)**2    ! TURN ALIVE THIS  XOR
C      F=DABS(F1-F2)   ! TURN ALIVE THIS  - BUT NOT BOTH ----------
      RETURN
      END
C     ----------------------------------------------------------------
```

```
      SUBROUTINE AMGM(M,F,X)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     FOR WHAT VALUES ARITHMETIC MEAN = GEOMETRIC MEAN ? THE ANSWER IS:
C     IF X(1)=X(2)=....=X(M) AND ALL X ARE NON-NEGATIVE
C     TAKE ONLY THE ABSOLUTE VALUES OF X
      SUM=0.D00
      DO I=1,M
      X(I)=DABS(X(I))
      ENDDO
C     SET SUM = SOME POSITIVE NUMBER. THIS MAKES THE FUNCTION UNIMODAL
      SUM= 100.D00 ! TURNED  ALIVE FOR UNIQUE MINIMUM AND SET SUM TO
C     SOME POSITIVE NUMBER. HERE IT IS 100; IT COULD BE ANYTHING ELSE.
      F1=0.D00
      F2=1.D00
      DO I=1,M
      F1=F1+X(I)
      F2=F2*X(I)
      ENDDO
      XSUM=F1
      F1=F1/M ! SUM DIVIDED BY M = ARITHMETIC MEAN
      F2=F2**(1.D00/M) ! MTH ROOT OF THE PRODUCT = GEOMETRIC MEAN
      F=(F1-F2)**2
      IF(SUM.GT.0.D00) F=F+(SUM-XSUM)**2
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE FUNCT2(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     IN FOGEL, L.J., ANGELIN, P.J. AND BACK, T. (ED) PROCEEDINGS OF THE
C     FIFTH ANNUAL CONFERENCE ON EVOLUTIONARY PROGRAMMING, PP. 451-460,
C     MIT PRESS, CAMBRIDGE, MASS.
C     MIN F (0, 0, ..., 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      F1=1.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*20
      ENDIF
      ENDDO
      DO I=1,M
      F=F+DABS(X(I))
      F1=F1*DABS(X(I))
      ENDDO
      F=F+F1
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE FUNCT3(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
```

```
C     MIN F (0, 0, ... , 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      F1=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*200
      ENDIF
      ENDDO
      DO I=1,M
      F1=0.D00
      DO J=1,I
      F1=F1+X(J)**2
      ENDDO
      F=F+F1
      ENDDO
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE FUNCT4(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F (0, 0, ..., 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      DO I=1,M
      IF(X(I).LT.0.D00 .OR. X(I).GE.M) THEN
      CALL RANDOM(RAND)
      X(I)=RAND*2*M
      ENDIF
      ENDDO
C     FIND MAX(X(I))=MAX(ABS(X(I))) NOTE: HERE X(I) CAN BE ONLY POSITIVE
      XMAX=X(1)
      DO I=1,M
      IF(XMAX.LT.X(I)) XMAX=X(I)
      ENDDO
      F=XMAX
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE FUNCT6(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F (-.5, -.5, ..., -.5) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*200
      ENDIF
```

```
      ENDDO
      DO I=1,M
       F=F+(X(I)+0.5D00)**2
      ENDDO
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE FUNCT7(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F(0, 0, ..., 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
      F=0.D00
       DO I=1,M
       IF(DABS(X(I)).GT.1.28D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*2.56D00
       ENDIF
       ENDDO
      DO I=1,M
      CALL RANDOM(RAND)
      F=F+(I*X(I)**4)
      ENDDO
       CALL RANDOM(RAND)
       F=F+RAND
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE FUNCT12(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(100),Y(100)
      DATA A,B,C /10.D00,100.D00,4.D00/
      PI=4.D00*DATAN(1.D00)
      F=0.D00
C     MIN F (-1, -1, -1, ..., -1) = 0
C       X(I)=-1.D00  ! TO CHECK, TURN IT ALIVE
       DO I=1,M
       IF(DABS(X(I)).GT.50.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*100.D00
       ENDIF
       ENDDO
      F1=0.D00
      DO I=1,M
      XX=DABS(X(I))
      U=0.D00
      IF(XX.GT.A) U=B*(XX-A)**C
      F1=F1+U
      ENDDO
      F2=0.D00
```

```
      DO I=1,M-1
      Y(I)=1.D00+.25D00*(X(I)+1.D00)
      F2=F2+ (Y(I)-1.D00)**2 * (1.D00+10.D00*(DSIN(PI*X(I+1))**2))
      ENDDO
      Y(M)=1.D00+.25D00*(X(M)+1.D00)
      F3=(Y(M)-1.D00)**2
      Y(1)=1.D00+.25D00*(X(1)+1.D00)
      F4=10.D00*(DSIN(PI*Y(1)))**2
      F=(PI/M)*(F4+F2+F3)+F1
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE FUNCT13(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(100)
      DATA A,B,C /5.D00,100.D00,4.D00/
      PI=4*DATAN(1.D00)
      F=0.D00
C     MIN F (1, 1, 1, ..., 4.7544 APPROX) = -1.15044 APPROX
C       X(I)=1.D00  ! TO CHECK, TURN IT ALIVE
C       X(M)=-4.7544  ! TO CHECK, TURN IT ALIVE
      DO I=1,M
      IF(DABS(X(I)).GT.50.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*100.D00
      ENDIF
      ENDDO
      F1=0.D00
      DO I=1,M
      XX=DABS(X(I))
      U=0.D00
      IF(XX.GT.A) U=B*(XX-A)**C
      F1=F1+U
      ENDDO
      F2=0.D00
      DO I=1,M-1
      F2=F2+ (X(I)-1.D00)**2 * (1.D00+(DSIN(3*PI*X(I+1))**2))
      ENDDO
      F3=(X(M)-1.D00)*  (1.D00+(DSIN(2*PI*X(M)))**2)
      F4=(DSIN(3*PI*X(1)))**2
      F=0.1*(F4+F2+F3)+F1
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE FUNCT14(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F (-31.98, 31.98) = 0.998
      PARAMETER (N=25,NN=2)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(2), A(NN,N)
      DATA (A(1,J),J=1,N) /-32.D00,-16.D00,0.D00,16.D00,32.D00,-32.D00,
     & -16.D00,0.D00,16.D00,32.D00,-32.D00,-16.D00,0.D00,16.D00,32.D00,
```

```
     & -32.D0,-16.D0,0.D0,16.D0,32.D0,-32.D0,-16.D0,0.D0,16.D0,32.D0/
      DATA (A(2,J),J=1,N) /-32.D00,-32.D00,-32.D00,-32.D00,-32.D00,
     & -16.D00,-16.D00,-16.D00,-16.D00,-16.D00,0.D00,0.D00,0.D00,0.D00,
     & 0.D00,16.D00,16.D00,16.D00,16.D00,16.D00,32.D00,32.D00,
     & 32.D00,32.D00,32.D00/

      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*200.D00
      ENDIF
      ENDDO
      F1=0.D00
      DO J=1,N
      F2=0.D00
      DO I=1,2
      F2=F2+(X(I)-A(I,J))**6
      ENDDO
      F2=1.D00/(J+F2)
      F1=F1+F2
      ENDDO
      F=1.D00/(0.002D00+F1)
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE FUNCT15(M,F,X)
C     REF: YAO, X. AND LIU, Y. (1996): FAST EVOLUTIONARY PROGRAMMING
C     MIN F(.19, .19, .12, .14) = 0.3075
      PARAMETER (N=11)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*), A(N),B(N)
      DATA (A(I),I=1,N) /.1957D00,.1947D00,.1735D00,.16D00,.0844D00,
     & .0627D00,.0456D00,.0342D00,.0323D00,.0235D00,.0246D00/
      DATA (B(I),I=1,N) /0.25D00,0.5D00,1.D00,2.D00,4.D00,6.D00,8.D00,
     & 10.D00,12.D00,14.D00,16.D00/
      DO I=1,N
      B(I)=1.D00/B(I)
      ENDDO
      F=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.5.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D00)*10.D00
      ENDIF
      ENDDO
      DO I=1,N
      F1=X(1)*(B(I)**2+B(I)*X(2))
      F2=B(I)**2+B(I)*X(3)+X(4)
      F=F+(A(I)-F1/F2)**2
      ENDDO
      F=F*1000
      RETURN
```

```
      END

C     ---------------------------------------------------------------
      SUBROUTINE HOUGEN(A,M,F)
      PARAMETER(N=13,K=3)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(N,K),RATE(N),A(*)
C     ---------------------------------------------------------------
C     HOUGEN FUNCTION (HOUGEN-WATSON MODEL FOR REACTION KINATICS)
C     NO. OF PARAMETERS (A) TO ESTIMATE = 5 = M

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

C     THE NEXT BEST RESULTS GIVEN BY HOOKE-JEEVES & QUASI-NEWTON
C     A(1)=2.475221;A(2)=0.599177; A(3)=0.124172; A(4)=0.083517
C     A(5)=0.217886; SUM OF SQUARES OF DEVIATION = 0.318593458
C     R=0.99941
C     MOST OF THE OTHER METHODS DO NOT PERFORM WELL
C     ---------------------------------------------------------------
      DATA X(1,1),X(1,2),X(1,3),RATE(1) /470,300,10,8.55/
      DATA X(2,1),X(2,2),X(2,3),RATE(2) /285,80,10,3.79/
      DATA X(3,1),X(3,2),X(3,3),RATE(3) /470,300,120,4.82/
      DATA X(4,1),X(4,2),X(4,3),RATE(4) /470,80,120,0.02/
      DATA X(5,1),X(5,2),X(5,3),RATE(5) /470,80,10,2.75/
      DATA X(6,1),X(6,2),X(6,3),RATE(6) /100,190,10,14.39/
      DATA X(7,1),X(7,2),X(7,3),RATE(7) /100,80,65,2.54/
      DATA X(8,1),X(8,2),X(8,3),RATE(8) /470,190,65,4.35/
      DATA X(9,1),X(9,2),X(9,3),RATE(9) /100,300,54,13/
      DATA X(10,1),X(10,2),X(10,3),RATE(10) /100,300,120,8.5/
      DATA X(11,1),X(11,2),X(11,3),RATE(11) /100,80,120,0.05/
      DATA X(12,1),X(12,2),X(12,3),RATE(12) /285,300,10,11.32/
      DATA X(13,1),X(13,2),X(13,3),RATE(13) /285,190,120,3.13/
C     WRITE(*,1)((X(I,J),J=1,K),RATE(I),I=1,N)
C  1 FORMAT(4F8.2)
      DO J=1,M
      IF(DABS(A(J)).GT.5.D00) THEN
      CALL RANDOM(RAND)
      A(J)=RAND
      ENDIF
      ENDDO
      F=0.D00
      DO I=1,N
      D=1.D00
        DO J=1,K
        D=D+A(J+1)*X(I,J)
        ENDDO
      FX=(A(1)*X(I,2)-X(I,3)/A(M))/D
C     FX=(A(1)*X(I,2)-X(I,3)/A(5))/(1.D00+A(2)*X(I,1)+A(3)*X(I,2)+
C     A(4)*X(I,3))
```

```
      F=F+(RATE(I)-FX)**2
      ENDDO
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE GIUNTA(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     GIUNTA FUNCTION
C     X(I) = -1 TO 1;  M=2
       DO I=1,M
       IF(DABS(X(I)).GT.1.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*2.D00
       ENDIF
       ENDDO
      C=16.D00/15.D00
      F=DSIN(C*X(1)-1.D0)+DSIN(C*X(1)-1.D0)**2+DSIN(4*(C*X(1)-1.D0))/50+
     &DSIN(C*X(2)-1.D0)+DSIN(C*X(2)-1.D0)**2+DSIN(4*(C*X(2)-1.D0))/50+.6
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE EGGHOLD(M,X,F)
C     EGG HOLDER FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      DO I=1,M
       IF(DABS(X(I)).GT.512.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*1024.D00
       ENDIF
       ENDDO
       F=0.D00
      DO I=1,M-1
      F1=-(X(I+1)+47.D00)
      F2=DSIN(  DSQRT(  DABS(  X(I+1)+X(I)/2+47.D00  )  )  )
      F3=DSIN(  DSQRT(  DABS(  X(I)-(X(I+1)+47.D00) ) ))
      F4=-X(I)
      F=F+ F1*F2+F3*F4
      ENDDO
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE TRID(M,X,F)
C     TRID FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      F1=0.D00
      F2=0.D00
      DO I=1, M
      F1=F1+(X(I)-1.D00)**2
      ENDDO
      DO I=2, M
```

```
      F2=F2+X(I)*X(I-1)
      ENDDO
      F=F1-F2
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE GRIEWANK(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     GRIEWANK FUNCTION
      F1=0.D00
      F2=1.0D00
      DO I=1,M
      F1=F1+X(I)**2
      FI=DFLOAT(I)
      F2=F2*DCOS(X(I)/DSQRT(FI))
      ENDDO
      F=F1/4000.D00-F2+1.0D00
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE WEIERSTRASS(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
C     WEIERSTRASS FUNCTION ------------------------------------
      DATA AX,BX,KMAX/0.5D00,3.D00,20/
      F1=0.D00
      F2=0.D00

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

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

      DO KK=1,KMAX+1
      K=KK-1
      F2=F2+(AX**K*DCOS(2*PI*BX**K*0.5D00))
      ENDDO
      F=F1-M*F2
      RETURN
      END
```

```
C     ---------------------------------------------------------------
      SUBROUTINE LEVY3(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     LEVY # 3 (LEVY ET AL. 1981) ------------------------------------
        DO I=1,M
        IF(DABS(X(I)).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
        ENDDO
         F1=0.0D+00
         F2=0.0D+00
         DO I=1,5
         F1=F1+(I*DCOS((I-1)*X(1)+I))
         F2=F2+(I*DCOS((I+1)*X(2)+I))
         ENDDO
        F=F1*F2
        RETURN
        END
C     ---------------------------------------------------------------
      SUBROUTINE LEVY5(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
           F1=0.0D+00
        F2=0.0D+00
       DO I=1,5
      F1=F1+(I*DCOS((I-1)*X(1)+I))
         F2=F2+(I*DCOS((I+1)*X(2)+I))
         ENDDO
         F3=(X(1)+1.42513D+00)**2
        F4=(X(2)+0.80032D+00)**2
        F=(F1*F2) + (F3+F4)
        RETURN
        END
C     ---------------------------------------------------------------
      SUBROUTINE LEVY8(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*),Y(3)
      PI=4*DATAN(1.D00)
C     LEVY # 8 FUNCTION ----------------------------------------------
        DO I=1,3
        Y(I)=1.D+00+(X(I)-1.D+00)/4.D+00
        ENDDO
        F1=DSIN(PI*Y(1))**2
       F3=(Y(3)-1.D+00)**2
      F2=0.D+00
       DO I=1,2
        F2=F2+((Y(I)-1.D+00)**2)*(1.D+00+10.D+00*(DSIN(PI*Y(I+1)))**2)
        ENDDO
      F=F1+F2+F3
      RETURN
      END
```

```
C     ---------------------------------------------------------------
      SUBROUTINE RASTRIGIN(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
C     RASTRIGIN'S FUNCTION
      F=0.D00
      DO I=1,M
      F=F+ X(I)**2 -10*DCOS(2*PI*X(I)) + 10.D00
      ENDDO
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE ACKLEY(M,X,F)
C     ACKLEY FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      F=20.D00+DEXP(1.D00)
      DO I=1,M
      IF(X(I).LT. -15.D00 .OR. X(I).GT.30.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*90 -15.D00
      ENDIF
      ENDDO
      F1=0.D00
      F2=0.D00
       DO I=1,M
        F1=F1+X(I)**2
        F2=F2+DCOS(2*PI*X(I))
       ENDDO
      F1=-20*DEXP(-0.2D00*DSQRT(F1/M))
      F2=-DEXP(F2/M)
      F=F+F1+F2
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE MICHALEWICZ(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     MICHALEWICZ FUNCTION   [ 0 <= X(I) <= PI ] MP IS A PARAMETER
      MP=10  ! SET IT TO THE DESIRED VALUE
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF(X(I).LT.0.D00 .OR. X(I).GT.PI)THEN
      CALL RANDOM(RAND)
      X(I)=RAND*PI
      ENDIF
      ENDDO
      F=0.D00
      DO I=1,M
      F=F-DSIN(X(I))*(DSIN(I*X(I)**2/PI))**(2*MP)
      ENDDO
```

```
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE SCHWEFEL(M,X,F)
C     SCHWEFEL FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      DO I=1,M
      IF(DABS(X(I)).GT.500) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*1000
      ENDIF
      ENDDO
      F=0.D00
      DO I=1,M
      F=F+ X(I)*DSIN(DSQRT(DABS(X(I))))
      ENDDO
      !F=418.9829D00*M - F
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE SHUBERT(M,X,F)
C     SHUBERT FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
        DO I=1,M
        IF(DABS(X(I)).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
        ENDDO
      F1=0.D00
      F2=0.D00
      DO I=1,5
      F1=F1+I*DCOS((I+1.D00)*X(1)+I)
      F2=F2+I*DCOS((I+1.D00)*X(2)+I)
      ENDDO
      F=F1*F2
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE DIXPRICE(M,X,F)
C     DIXON & PRICE FUNCTION
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
       DO I=1,M
        IF(DABS(X(I)).GT.10.D00) THEN
        CALL RANDOM(RAND)
        X(I)=(RAND-0.5D00)*20
        ENDIF
       ENDDO
      F=0.D00
      DO I=2, M
```

```
      F=F + I*(2*X(I)**2-X(I-1))**2
      ENDDO
      F=F+(X(1)-1.D00)**2
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE SHEKEL(M,X,F)
C     SHEKEL FUNCTION FOR TEST OF GLOBAL OPTIMIZATION METHODS
      PARAMETER(NROW=10,NCOL=4, NR=9)! NR MAY BE 2 TO 10
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION A(NROW,NCOL),C(NROW),X(*)
      DATA ((A(I,J),J=1,NCOL),I=1,NROW)/4.,4.,4.,4.,1.,1.,1.,1.,8.,8.,
     & 8.,8.,6.,6.,6.,6.,3.,7.,3.,7.,2.,9.,2.,9.,5.,5.,3.,3.,8.,1.,8.,
     & 1.,6.,2.,6.,2.,7.,3.6D00,7.,3.6D00/
      DATA (C(I),I=1,NROW)/0.1D00,0.2D00,0.2D00,0.4D00,0.4D00,0.6D00,
     & 0.3D00,0.7D00,0.5D00,0.5D00/
      F=0.D00
      DO I=1,NR
        S=0.D00
        DO J=1,M
        S=S+(X(J)-A(I,J))**2
        ENDDO
      F=F-1.D00/(S+C(I))
      ENDDO
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE PAVIANI(M,X,F)
C     PAVIANI FUNCTION : MIN F(9.3502,...,9.3502)=45.77847 APPROX
C     IN THE DOMAIN 2<= X(I) <= 10 FOR I=1,2,...,10.
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      DO I=1,M
      IF(X(I).LE.2.D00.OR.X(I).GE.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=RAND*8+2.D00
      ENDIF
      ENDDO
      F1=0.D00
      F2=1.D00
      DO I=1,M
      F1=F1+ DLOG(X(I)-2.D00)**2+DLOG(10.D00-X(I))**2
      F2=F2*X(I)
      ENDDO
      F=F1-F2**0.2
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE BRANIN1(M,X,F)
C     BRANIN FUNCTION #1  MIN F (1, 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
```

```
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*20
      ENDIF
      ENDDO
      F=(1.D00-2*X(2)+DSIN(4*PI*X(2))/2.D00-X(1))**2+(X(2)-
     & DSIN(2*PI*X(1))/2.D00)**2
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE BRANIN2(M,X,F)
C     BRANIN FUNCTION #2 MIN F (3.1416, 2.25)= 0.397887 APPROX
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      IF(X(1).LT.-5.D00 .OR. X(1).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(1)=RAND*15-5.D00
      ENDIF
      IF(X(2).LT.0.D00 .OR. X(2).GT.15.D00) THEN
      CALL RANDOM(RAND)
      X(2)=RAND*15
      ENDIF
      F=(X(2)-5.D00*X(1)**2/(4*PI**2)+5*X(1)/PI-6.D00)**2 +
     &   10*(1.D00-1.D00/(8*PI))*DCOS(X(1))+10.D00
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE BOHACHEVSKY1(M,X,F)
C     BOHACHEVSKY FUNCTION #1 : MIN F (0, 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1))-0.4D00*DCOS(4*PI*X(2))
     & +0.7D00
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE BOHACHEVSKY2(M,X,F)
C     BOHACHEVSKY FUNCTION #2  : MIN F (0, 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
```

```
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1))*DCOS(4*PI*X(2))+0.3D00
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE BOHACHEVSKY3(M,X,F)
C     BOHACHEVSKY FUNCTION #3   : MIN F (0, 0) = 0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1)+4*PI*X(2))+0.3D00
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE EASOM(M,X,F)
C     EASOM FUNCTION : 2-VARIABLES, MIN F (PI, PI) = -1.
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=-DCOS(X(1))*DCOS(X(2))*DEXP(-(X(1)-PI)**2 -(X(2)-PI)**2)
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE ROSENBROCK(M,X,F)
C     ROSENBROCK FUNCTION : M VARIABLE; MIN F (1, 1,...,1)=0
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      DO I=1,M
      IF(X(I).LT.-5.D00 .OR. X(I).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=RAND*15-5.D00
      ENDIF
      ENDDO
      F=0.D00
      DO I=1,M-1
      F=F+ (100.D00*(X(I+1)-X(I)**2)**2 + (X(I)-1.D00)**2)
      ENDDO
      RETURN
      END
```

```
C     ---------------------------------------------------------------
      SUBROUTINE CROSSLEG(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     CROSS-LEGGED TABLE FUNCTION ; -10<= X(I) <=10; M=2
C     MIN F(0 , X ) OR F(X, 0) = -1.
      PI=4*DATAN(1.D00)
       DO I=1,M
       IF( DABS(X(I)).GT.10.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*20
       ENDIF
       ENDDO
      F=-(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
     & (X(1)**2+X(2)**2)/PI))))+1.D00)**(-.1)
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE CROSS(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     CROSS FUNCTION ; -10<= X(I) <=10; M=2;
C     MIN F(A, B)=0 APPROX; A, B=1.3494 APPROX OF EITHER SIGN (+ OR -)
      PI=4*DATAN(1.D00)
       DO I=1,M
       IF( DABS(X(I)).GT.10.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*20
       ENDIF
       ENDDO
      F=(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
     & (X(1)**2+X(2)**2)/PI))))+1.D00)**(-.1)
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE CROSSINTRAY(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     CROSS IN TRAY FUNCTION ; -10<= X(I) <=10; M=2;
C     MIN F(A, B)=-20626.1218 APPROX; A, B=1.3494 APPROX OF EITHER SIGN
      PI=4*DATAN(1.D00)
       DO I=1,M
       IF( DABS(X(I)).GT.10.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*20
       ENDIF
       ENDDO
      F=-(DABS(DSIN(X(1))*DSIN(X(2))*DEXP(DABS(100.D00-(DSQRT
     & (X(1)**2+X(2)**2)/PI))))+1.D00)**(.1)
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE CROWNEDCROSS(M,X,F)
```

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

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

```
      PI=4*DATAN(1.D00)
      DO I=1,M
      IF( DABS(X(I)).GT.30) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*60
      ENDIF
      ENDDO
     F=X(1)**2-12*X(1)+11.D00+10*DCOS(PI*X(1)/2)+8*DSIN(5*PI*X(1))-
     & (1.D00/DSQRT(5.D00))*DEXP(-(X(2)-0.5D00)**2/2)
     RETURN
     END
C    ---------------------------------------------------------------
     SUBROUTINE MCCORMICK(M,X,F)
     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
     DIMENSION X(*)
C    MCCORMICK FUNCTION : -1.5<= X(1)<=4; -3<=X(2)<=4 ; M=2
C    MIN F (-0.54719755, -1.54719755) = -1.913223 APPROX
      IF(X(1).LT. -1.5D00 .OR. X(1) .GT. 4.D00) THEN
      CALL RANDOM(RAND)
      X(1)=RAND*5.5D00-1.5D00
      ENDIF
      IF(X(2).LT. -3.D00 .OR. X(2) .GT. 4.D00) THEN
      CALL RANDOM(RAND)
      X(2)=RAND*7.D00-3.D00
      ENDIF
     F=DSIN(X(1)+X(2))+(X(1)-X(2))**2-1.5*X(1)+2.5*X(2)+1.D00
     RETURN
     END
C    ---------------------------------------------------------------
     SUBROUTINE FENTONEASON(M,X,F)
     IMPLICIT DOUBLE PRECISION (A-H, O-Z)
     DIMENSION X(*)
C    FENTON & EASON FUNCTION  FMIN(1.74345, -2.029695) = 1.744152
     DO I=1,M
     IF(DABS(X(I)).GT.100.D00) THEN
     CALL RANDOM(RAND)
     X(I)=(RAND-0.5D00)*200
     ENDIF
     ENDDO
     F=1.2D00+0.1*X(1)**2 +(0.1D00+0.1*X(2)**2)/X(1)**2+
     & (.1*X(1)**2*X(2)**2+10.D00)/((X(1)*X(2))**4)
     RETURN
     END
C    ---------------------------------------------------------------
     SUBROUTINE WOOD(M,X,F)
     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
     COMMON /RNDM/IU,IV
     INTEGER IU,IV
     DIMENSION X(*)
C    WOOD FUNCTION:FMIN(0.443546,-0.194607,1.466077,2.15115)=1.09485393
     DO I=1,M
     IF(DABS(X(I)).GT.5.D00) THEN
     CALL RANDOM(RAND)
```

```
      X(I)=(RAND-0.5D00)*10
      ENDIF
      ENDDO
      F1=100*(X(2)+X(1)**2)**2 + (1.D0-X(1))**2 +90*(X(4)-X(3)**2)**2
      F2=(1.D0-X(3))**2 +10.1*((X(2)-1.D0)**2+(X(4)-1.D0)**2)
      F3=19.8*(X(2)-1.D0)*(X(4)-1.D0)
      F=F1+F2+F3
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE GLANKWAHMDEE(M,X,F)
      PARAMETER (N=5)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*),A(N,N), B(N)
C     ------------------ GLANKWAHMDEE FUNCTION ------------------------
C     GLANKWAHMDEE A, LIEBMAN JS AND HOGG GL (1979) "UNCONSTRAINED
C     DISCRETE NONLINEAR PROGRAMMING. ENGINEERING OPTIMIZATION 4: 95-107
C     PARSOPOULOS, KE AND VRAHATIS, MN (2002) RECENT APPROACHES TO
C     GLOBAL OPTIMIZATION PROBLEMS THROUGH PARTICLE SWARM OPTIMIZATION,
C     NATURAL COMPUTING 1: 235-306, 2002.  REPORT THE BEST OBTAINED
C     FMIN (0,12,23,17,6)= -737 OR  MIN F(0, 11,22,16,6)= -737
C     WE GET FMIN(-.232, 11.489, 22.273, 16.540, 6.115) = -739.822991
C     ----------------------------------------------------------------
      DATA ((A(I,J),J=1,N),I=1,N) /35,-20,-10,32,-10,-20, 40,-6,-31,32,
     & -10,-6,11,-6,-10,32,-31,-6,38,-20,-10,32,-10,-20,31/
      DATA (B(J),J=1,N) /-15,-27,-36,-18,-12/
C     ----------------------------------------------------------------
      DO I=1,M
      IF(DABS(X(I)).GT.100.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*200
      ENDIF
      ENDDO
      F=0.D0
      DO J=1,M
      F=F+B(J)*X(J)
      ENDDO
      DO I=1,M
      C=0.D0
      DO J=1,M
      C=C+X(J)*A(J,I)
      ENDDO
      F=F+C*X(I)
      ENDDO
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE FLETCHER(M,X,F)
C     FLETCHER-POWELL FUNCTION, M <= 10, ELSE IT IS VERY MUCH SLOW
C     SOLUTION: MIN F = 0 FOR X=(C1, C2, C3,...,CM)
      PARAMETER(N=10) ! FOR DIMENSION OF DIFFERENT MATRICES AND VECTORS
```

```
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*),A(N,N),B(N,N),AA(N),BB(N),AL(N),C(N),C1(N)
      PI=4*DATAN(1.D00)
C     GENERATE A(I,J) AND B(I,J) BETWEEN (-100, 100) RANDOMLY.
C     C(I) = BETWEEN (-PI, PI) IS EITHER GIVEN OR RANDOMLY GENERATED.
C     DATA (C(I),I=1,10)/1,2,3,-3,-2,-1,0,1,2,3/ ! BETWEEN -PI AND PI
      DATA (C1(I),I=1,N)/-3,-3.02,-3.01,1,1.03,1.02,1.03,-.08,.001,3/
C     DATA (C1(I),I=1,N)/0,0,0,0,0,0,0,0,0,0/ ! ANOTHER EXAMPLE C1 = 0
      NC=0  ! DEFINE NC HERE 0 OR 1 OR 2
C     IF NC=0, C1 FROM DATA IS USED (THAT IS FIXED C);
C     IF NC=1, C IS MADE FROM C1 BY ADDING RANDOM PART - THAT IS C=C1+R
C     IF NC=2 THEN C IS PURELY RANDOM THAT IS C= 0 + R
C     IN ANY CASE C LIES BETWEEN -PI AND PI.
C     ---------------------------------------------------------------
C     FIND THE MAX MAGNITUDE ELEMENT IN C1 VECTOR (UPTO M ELEMENTS)
      CMAX=DABS(C1(1))
      DO J=2,M
      IF(DABS(C1(J)).GT.CMAX) CMAX=DABS(C1(J))
      ENDDO
      RANGE=PI-CMAX
C     ---------------------------------------------------------------
      DO J=1,M
      DO I=1,M
      CALL RANDOM(RAND)
      A(I,J)=(RAND-0.5D00)*200.D00
      CALL RANDOM(RAND)
      B(I,J)=(RAND-0.5D00)*200.D00
      ENDDO
      IF(NC.EQ.0) AL(J)=C1(J) ! FIXED OR NON-STOCHASTIC C
      IF(NC.EQ.1) THEN
      CALL RANDOM(RAND)
      AL(J)=C1(J)+(RAND-0.5D0)*2*RANGE ! A PART FIXED, OTHER STOCHASTIC
      ENDIF
      IF(NC.EQ.2) THEN
      CALL RANDOM(RAND)
      AL(J)=(RAND-0.5D00)*2*PI ! PURELY STOCHASTIC
      ENDIF
      ENDDO
      DO I=1,M
      AA(I)=0.D00
      DO J=1,M
      AA(I)=AA(I)+A(I,J)*DSIN(AL(J))+B(I,J)*DCOS(AL(J))
      ENDDO
      ENDDO
C     ---------------------------------------------------------------
      DO I=1,M
      IF(DABS(X(I)).GT.PI) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-0.5D00)*2*PI
      ENDIF
      ENDDO
```

```
      DO I=1,M
      BB(I)=0.D00
      DO J=1,M
      BB(I)=BB(I)+A(I,J)*DSIN(X(J))+B(I,J)*DCOS(X(J))
      ENDDO
      ENDDO
      F=0.D00
      DO I=1,M
      F=F+(AA(I)-BB(I))**2
      ENDDO
      RETURN
      END
C-----------------------------------------------------------------------
      SUBROUTINE POWELL(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     POWELL FUNCTION  ; -4<= X(I) <=5; M=A MULTIPLE OF 4;
C     MIN F  =  0.0
       DO I=1,M
       IF(X(I).LT.-4.D00 .OR. X(I).GT.5.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*9+.5D00
       ENDIF
       ENDDO
      M4=M/4
      F=0.D00
      DO I=1,M4
      J=4*I
      F=F+(X(J-3)+10*X(J-2))**2+5*(X(J-1)-X(J))**2+(X(J-2)-X(J-1))**4 +
     & 10*(X(J-3)-X(J))**4
      ENDDO
      RETURN
      END
C-----------------------------------------------------------------------
      SUBROUTINE HARTMANN(M,X,F)
      PARAMETER (N=4)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*),P(N,3),A(N,3),C(N)
C     HARTMANN FUNCTION
C     MIN F  =  -3.86278 APPROX  : 0 < X < 1.
      DATA ((P(I,J),J=1,3),I=1,4) /0.6890,0.1170,0.2673,0.4699,0.4387,
     &  0.7470,0.1091,0.8732,0.5547,0.0381,0.5743,0.8828/
      DATA ((A(I,J),J=1,3),I=1,4) /3.0,10.0,30.0,0.1,10.0,35.0,3.0,
     &  10.0,30.0,0.1,10.0,35.0/
      DATA (C(J),J=1,4) /1.0,1.2,3.0,3.2/
       DO I=1,M
       IF(X(I).LE.0.D00 .OR. X(I).GE.1.D00) THEN
       CALL RANDOM(RAND)
       X(I)=RAND
       ENDIF
       ENDDO
      F=0.D00
      DO I=1,N
```

```
      S=0.D00
      DO J=1,M
      S=S+A(I,J)*(X(J)-P(I,J))**2
      ENDDO
      F=F+C(I)*DEXP(-S)
      ENDDO
      F=-F
      RETURN
      END
C-----------------------------------------------------------------------
      SUBROUTINE COLVILLE(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
C     COLVILLE FUNCTION  ; -10<= X(I) <=10; M= 4;
C     MINF(1,1,1,1)= 0.0
       DO I=1,M
       IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*20
       ENDIF
       ENDDO
      F=100*(X(1)**2-X(2))**2 + (X(1)-1.D00)**2 +(X(3)-1.D00)**2+
     & 90*(X(3)**2-X(4))**2+10.1*((X(2)-1.D0)**2+(X(4)-1.D00)**2)+
     & 19.8*(X(2)-1.D00)*(X(4)-1.D00)
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE HIMMELBLAU(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
      NF=0   ! SET NF = 0 MULTIPLE OPTIMA NF=1 SINGLE OPTIMUM
       DO I=1,M
       IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*20
       ENDIF
       ENDDO
C     HIMMELBLAU FUNCTION. IT HAS MULTIPLE (4) GLOBAL OPTIMA : MINF=0
C     (3, 2); (-2.8051, 3.1313); (3.5744, -1.8481); (-3.779, -3.283)
      IF(NF.EQ.0) THEN
      F= (X(1)**2+X(2)-11)**2+  (X(1)+X(2)**2-7)**2
      RETURN
      ENDIF
      IF(NF.EQ.1) THEN
C     MODIFIED HIMMELBLAU FUNCTION. IT HAS ONLY ONE GLOBAL OPTIMUM
C     MINF=0 AT (3,2)
      F= (X(1)**2+X(2)-11)**2+(X(1)+X(2)**2-7)**2+0.1D00*((X(1)-3)**2 +
     1  (X(2)-2)**2)
      ENDIF
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE BEALE(M,X,F)
```

```
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
       DO I=1,M
       IF(X(I).LT.-4.500 .OR. X(I).GT.4.500) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*9
       ENDIF
       ENDDO
C     BEALE FUNCTION  : MINF =0 AT (3, 0.5)
      F1=(1.5D00-X(1)+X(1)*X(2))**2
      F2=(2.25D00-X(1)+X(1)*X(2)**2)**2
      F3=(2.625D00-X(1)+X(1)*X(2)**3)**2
      F=F1+F2+F3
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE BOOTH(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
       DO I=1,M
       IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*20
       ENDIF
       ENDDO
C     BOOTH FUNCTION  MINF (1,3)=0
      F=(X(1)+2*X(2)-7.0D00)**2+(2*X(1)+X(2)-5.0D00)**2
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE HUMP(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
       DO I=1,M
       IF(X(I).LT.-5.D00 .OR. X(I).GT.5.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*10
       ENDIF
       ENDDO
C     HUMP FUNCTION  MINF (0.0898, -0.7127) = -1.0316
      F=4*X(1)**2 - 2.1D00*X(1)**4 + (X(1)**6)/3.D00 + X(1)*X(2) -
     & 4*X(2)**2 + 4*X(2)**4
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE MATYAS(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*)
       DO I=1,M
       IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5D00)*20
       ENDIF
```

```
      ENDDO
C     MATYAS FUNCTION  MIN F (0, 0) = 0
      F=0.26D00*(X(1)**2 + X(2)**2) - 0.48D00*X(1)*X(2)
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE MISHRA_1(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
C     MIN F (1, 1, ..., 1) =2
      DO I=1,M
      IF(X(I).LT.0.D00 .OR. X(I).GT.1.D00) THEN
      CALL RANDOM(RAND)
      X(I)=RAND
      ENDIF
      ENDDO
      S=0.D00
      DO I=1,M-1
      S=S+X(I)
      ENDDO
      X(M)=(M-S)
      F=(1.D00+X(M))**X(M)
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE MISHRA_2(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
C     MIN F (1, 1, ..., 1) =2
      DO I=1,M
      IF(X(I).LT.0.D00 .OR. X(I).GT.1.D00) THEN
      CALL RANDOM(RAND)
      X(I)=RAND
      ENDIF
      ENDDO
      S=0.D00
      DO I=1,M-1
      S=S+(X(I)+X(I+1))/2.D00
      ENDDO
      X(M)=(M-S)
      F=(1.D00+X(M))**X(M)
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE ZAKHAROV(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
```

```
C     ZAKHAROV FUNCTIONMIN MINF = (0, 0, ..., 0) =0
      DO I=1,M
      IF(X(I).LT.-5.D00 .OR. X(I).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D0)*15 + 2.5D0
      ENDIF
      ENDDO
      F1=0.D00
      F2=0.D00
      DO I=1, M
      F1=F1+ X(I)**2
      F2=F2 + I*X(I)/2.D00
      ENDDO
      F=F1+F2**2+F2**4
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE MULTIMOD(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
C     MULTIMODAL  FUNCTION  MINF = (0,0)= -1  : M=2
      DO I=1,M
      IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D0)*20
      ENDIF
      ENDDO
C     A TYPICAL MULTIMODAL FUNCTION
      F=-DCOS(X(1))*DCOS(X(2))*DEXP(-(X(1)**2 + X(2)**2)**2/4.D00)
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE NONLIN(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
C     NONLINEAR  MULTIMODAL FUNCTION  MINF = 0
      DO I=1,M
      IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D0)*20
      ENDIF
      ENDDO
      F=0.D0
      DO I=2,M
      F=F+DCOS(DABS(X(I)-X(I-1)) / DABS(X(I-1)+X(I)))
      ENDDO
      F=F+(M-1.D00)
C     IF 0.001*X(1) IS ADDED TO F, IT BECOMES UNIMODAL
C     F=F+0.001*X(1)
```

```
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE QUADRATIC(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
C     QUADRATIC FUNCTION  MINF (0.19388, 0.48513) = -3873.7243 (M=2)
      DO I=1,M
      IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D00) THEN
      CALL RANDOM(RAND)
      X(I)=(RAND-.5D0)*200
      ENDIF
      ENDDO
      F=-3803.84-138.08*X(1)-232.92*X(2)+128.08*X(1)**2+203.64*X(2)**2+
     & 182.25*X(1)*X(2)
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE TRIGON(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
C     TRIGON FUNCTION F MIN (0, 0, 0,..., 0) OR (PI, 0, 0, ...,0) =0
      PI=4*DATAN(1.D00)
       DO I=1,M
       IF(X(I).LT.0.D00 .OR. X(I).GT.PI) THEN
       CALL RANDOM(RAND)
       X(I)=RAND*PI
       ENDIF
       ENDDO
      F=0.D00
      DO I=2,M
      F=F+(DCOS(I+0.D00)*DSIN(X(I)-X(I-1))**2 +
     & (I-1.D00)*(1.D0-DCOS(X(I))))**2
      ENDDO
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE WAVY1(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
C     WAVY-1 FUNCTION F MIN (24, 24,...,24)=0
       DO I=1,M
       IF(X(I).LT.-100.D00 .OR. X(I).GT.100.D0) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-.5D0)*200
       ENDIF
       ENDDO
```

```
      F=0.D00
      DO I=1,M
      F=F+DABS(2*(X(I)-24.D0)+(X(I)-24.D0)*DSIN(X(I)-24.D0))
      ENDDO
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE WAVY2(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
C     WAVY-2 FUNCTION F MIN (0,0,...,0)=0
       DO I=1,M
       IF(X(I).LT.-100.D00 .OR. X(I).GT.100.D0) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-.5D0)*200
       ENDIF
       ENDDO
      F=0.D00
      DO I=1,M
      F=F+X(I)**6*(2.D0+DSIN(1.D0/X(I)))
      ENDDO
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE TREFETHEN(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
C     TREFETHEN FUNCTION:FMIN(-0.02440307923,0.2106124261)=-3.3068686475
       DO I=1,M
       IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D0) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-.5D0)*20
       ENDIF
       ENDDO
      F=0.D00
C     THE JAN-FEB ISSUE OF SIAM NEWS CONTAINED THE FOLLOWING CHALLENGE
C     FROM L. N. TREFETHEN OF OXFORD UNIVERSITY.
C     FMIN = F(-0.02440307923, 0.2106124261)= -3.3068686475
      F=DEXP(DSIN(50*X(1))) + DSIN(60*DEXP(X(2))) + DSIN(70*DSIN(X(1)))
     &  + DSIN(DSIN(80*X(2))) - DSIN(10*(X(1)+X(2))) +
     & 1.D0/4*(X(1)**2 + X(2)**2)
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE ALPINE(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
```

```
C     ALPINE FUNCTION F MIN (7.917, 7.917, ..., 7.917)= -2.808131 APPROX
C     THE DOMAIN : X IN [-10, 10].
C     NOTE: IN THE ORIGINAL FUNCTION MIN F = -(2.808131)^M
       DO I=1,M
       IF(X(I).LT.-10.D00 .OR. X(I).GT.10.D0) THEN
       CALL RANDOM(RAND)
       X(I)=(RAND-0.5)*20
       ENDIF
       ENDDO
      F1=1.D0
      F2=1.D0
      DO I=1,M
      F1=F1*DSIN(X(I))
      F2=F2*X(I)
      ENDDO
      F=-DEXP(DLOG(F1*DSQRT(F2))/M)
      RETURN
      END
C     ----------------------------------------------------------------
      SUBROUTINE LINPROG1(M,F,X)
C     LINEAR PROGRAMMING : MINIMIZATION PROBLEM
C     IN THIS PROBLEM : M = NO. OF DECISION VARIABLES = 2
C     MIN F (2.390, 2.033) = -19.7253 APPROX
C     MIN F = OVER J=1, M : DO SUM(A(1,J)*X(J))  SUBJECT TO CONSTRAINTS
C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=2
C     . . .       . . .       . . .      . . .
C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=N
C     ALL X(I) => 0
      PARAMETER (N=3) ! N IS THE NO. OF CONSTRAINTS + 1
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*),A(20,10),C(20),FF(20)
      DATA (A(1,J),J=1,2),C(1)/4.D0,5.D0,0.0D0/!COEFF OF OBJ FUNCTION
      DATA (A(2,J),J=1,2),C(2)/10.D0,3.D0,30D0/!COEFF OF 1ST CONSTRAINT
      DATA (A(3,J),J=1,2),C(3)/6.D0,20.D0,55.D0/!COEFF OF 2ND CONSTRAINT
C     ----------------------------------------------------------------
C     USING ONLY NON-NEGATIVE VALUES OF X(I)
      DO I=1,M
      X(I)=DABS(X(I))
      ENDDO
C     EVALUATION OF OBJ FUNCTION AND CONSTRAINTS
      DO I=1,N
      FF(I)=0.D00
      DO J=1,M
      FF(I)=FF(I)+A(I,J)*X(J)
      ENDDO
      ENDDO
      F=-FF(1) ! CHANGE OF SIGN FOR MINIMIZATION
C     CHECK FOR SATISFYING OR VIOLATING THE CONSTRAINTS
      DO I=2,N
      FF(I)=FF(I)-C(I) ! SLACK
C     PENALTY FOR CROSSING LIMITS
      IF(FF(I).GT.0) F=F+(10+FF(I))**2
      ENDDO
```

```
      RETURN
      END
C     ---------------------------------------------------------------
      SUBROUTINE LINPROG2(M,F,X)
C     LINEAR PROGRAMMING : MINIMIZATION PROBLEM
C     IN THIS PROBLEM : M = NO. OF DECISION VARIABLES = 3
C     MIN F (250, 625, 0) = -32500
C     MIN F = OVER J=1, M : DO SUM(A(1,J)*X(J))   SUBJECT TO CONSTRAINTS
C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=2
C     . . .       . . .       . . .      . . .
C     OVER  J=1, M : DO SUM(A(I,J)*X(J)) <= C(I) ; I=N
C     ALL X(I) => 0
      PARAMETER (N=4) ! N IS THE NO. OF CONSTRAINTS + 1
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(*),A(20,10),C(20),FF(20)
      DATA (A(1,J),J=1,3),C(1)/30.D0,40.D0,20.D0,0.0D0/! COEFF OF OBJ FUNCTION
      DATA (A(2,J),J=1,3),C(2)/10.D0,12.D0,7.D0,10000.0D0/!COEFF OF 1ST CONSTRAINT
      DATA (A(3,J),J=1,3),C(3)/7.D0,10.D0,8.D0,8000.D0/! COEFF OF 2ND CONSTRAINT
      DATA (A(4,J),J=1,3),C(4)/1.D0,1.D0,1.D0,1000.D0/! COEFF OF 3RD CONSTRAINT
C     ---------------------------------------------------------------
C     USING ONLY NON-NEGATIVE VALUES OF X(I)
      DO I=1,M
      X(I)=DABS(X(I))
      ENDDO
C     EVALUATION OF OBJ FUNCTION AND CONSTRAINTS
      DO I=1,N
      FF(I)=0.D00
      DO J=1,M
      FF(I)=FF(I)+A(I,J)*X(J)
      ENDDO
      ENDDO
      F=-FF(1) ! CHANGE OF SIGN FOR MINIMIZATION
C     CHECK FOR SATISFYING OR VIOLATING THE CONSTRAINTS
      DO I=2,N
      FF(I)=FF(I)-C(I) ! SLACK
C     PENALTY FOR CROSSING LIMITS
      IF(FF(I).GT.0.D00) F=F+(100.D00+FF(I))**2
      ENDDO
      RETURN
      END

C     ---------------------------------------------------------------
      SUBROUTINE NEWF(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
      PI=4*DATAN(1.D00)
      A1=X(1)
      F=-COS(A1)*COS(SIN(A1+1))*EXP(1-(A1**2+(A1+1)**2)/PI)
      RETURN
      END
C     ---------------------------------------------------------------
```

```
      SUBROUTINE NEWG(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
      PI=4*DATAN(1.D00)

      DO I=1,M
      IF(DABS(X(I)).GT.1.D0) THEN
      CALL RANDOM(RAND)
      X(I)=RAND
      ENDIF
      ENDDO
      F=(X(1)-1.5)**2 + SIN(0.8 * X(2) ** 2 + 15)**4 + COS(0.8*X(3)**2 +
     & 15)**4 + (X(4)-7.5)**4
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE F8F2(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
      DO I=1,M
      IF(DABS(X(I)).GT.100) THEN
      CALL RANDOM(RAND)
      SG=1
      IF(X(I).LT.0) SG=-1
      X(I)=RAND*100*SG
      ENDIF
      ENDDO
      S=0.D0
      DO J=1,M
      XJ=(X(J)-1.D0)**2
      DO I=1,M
      YJI=100*(X(I)-X(J)**2)**2 + XJ
      S=S+(YJI**2/4000.D0 -DCOS(YJI)+1.D0)
      ENDDO
      ENDDO
      F=S
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE KEANEBUMP(M,X,F)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION X(*)
      DO I=1,M
      IF(X(I).LE.0.D0.OR.X(I).GE.10.0D0) THEN
      CALL RANDOM(RAND)
      X(I)=RAND*10
      ENDIF
```

```
 ENDDO
1 GX=1.D0
 FX=0.D0
 DO I=1,M
 GX=GX*X(I)
 FX=FX+X(I)
 ENDDO
 GX=0.75D0-GX
 FX=FX-7.5D0*M
 IF(GX.GE.0.D0.OR.FX.GE.0.D0) THEN
 DO I=1,M
 CALL RANDOM(RAND)
 X(I)=RAND*10
 ENDDO
 GOTO 1
 ENDIF
 F1=0.D0
 F2=1.D0
 F3=0.D0
 DO I=1,M
 F1=F1+DCOS(X(I))**4
 F2=F2*DCOS(X(I))**2
 F3=F3+I*X(I)**2
 ENDDO
 F=-DABS((F1-2*F2)/(DSQRT(F3)))
 RETURN
 END
```