Performance of Repulsive Particle Swarm Method in Global Optimization of Some Important Test Functions: A Fortran Program

SK Mishra
Dept. of Economics
North-Eastern Hill University
Shillong (India)

I. Global Optimization: Optimization, that is finding minimum or maximum (that is, optimum) of a single-valued function $f(x)$ – often with constraints on x or some other function(s) of x, e.g. $h(x) \leq \kappa$ – is required in many practical applications. If the problem is convex (it has only one minimum or maximum point), the local minimum (maximum) is also a global minimum (maximum). However, this is not always the case. One may have to optimize a function that has many local optima (and sometimes several global optima as well).

Many methods were developed in the 1960s that perform very well in optimization of convex (nonlinear) functions. We find a number of good Fortran programs of such methods in Kuester and Mize (1973). However, these methods (and programs) fail when the optimand function is non-convex (or multi-modal), because the search process is caught in some local optimum of the optimand function.

Starting with the Genetic algorithm, many methods are now available to find the global optimum of non-convex functions. Some of these are: Clustering method, Simulated Annealing, Generalized simulated annealing, Ants Colony meta-heuristic, Tunneling method, Particle Swarm method, differential Evolution method, and so on. Yet, it may be noted, that no method is successful in optimizing an arbitrary function; none can guarantee a sure success. Perhaps, no method can ever do that.

Among these methods, the Particle Swarm method is perhaps the simplest to understand and implement. Due to its simplicity, it can be easily modified to suit the purpose and therefore, it has better prospects as well. It is well founded on philosophical and methodological grounds also.

II. The Particle Swarm Method of Global Optimization: This method is an instance of successful application of the philosophy of *bounded rationality* and decentralized decision-making to solve the global optimization problems (Simon, 1982; Bauer, 2002; Fleischer, 2005). It is observed that a swarm of birds or insects or a school of fish searches for food, protection, etc. in a very typical manner. If one of the members of the swarm sees a desirable path to go, the rest of the swarm will follow quickly. Every member of the swarm searches for the best in its locality - learns from its own experience. Additionally, each member learns from the others, typically from the best performer among them. The Particle Swarm method of optimization mimics this behaviour (see Wikipedia: http://en.**wikipedia**.org/wiki/Particle_**swarm**_optimization). Every individual of the swarm is considered as a particle in a multidimensional space that has a position and a velocity. These particles fly through hyperspace and remember the best position that they have seen. Members of a swarm communicate good positions to

each other and adjust their own position and velocity based on these good positions. There are two main ways this communication is done: (i) "swarm best" that is known to all (ii) "local bests" are known in neighborhoods of particles. Updating the position and velocity is done at each iteration as follows:

$$v_{i+1} = \omega v_i + c_1 r_1 (\hat{x}_i - x_i) + c_2 r_2 (\hat{x}_{gi} - x_i)$$
$$x_{i+1} = x_i + v_{i+1}$$

where,

- x is the position and v is the velocity of the individual particle. The subscripts i and $i+1$ stand for the recent and the next (future) iterations, respectively.
- ω is the inertial constant. Good values are usually slightly less than 1.
- c_1 and c_2 are constants that say how much the particle is directed towards good positions. Good values are usually right around 1.
- r_1 and r_2 are random values in the range [0,1].
- $\hat{x}$ is the best that the particle has seen; $\hat{x}_g$ is the global best seen by the swarm. This can be replaced by $\hat{x}_L$, the local best, if neighborhoods are being used.

The Particle Swarm method (Eberhart and Kennedy, 1995) has many variants. The Repulsive Particle Swarm (RPS) method of optimization (see Wikipedia, http://en.wikipedia.org/wiki/RPSO), one of such variants, is particularly effective in finding out the global optimum in very complex search spaces (although it may be slower on certain types of optimization problems). Other variants use a dynamic scheme (Liang and Suganthan, 2005; Huang et al., 2006).

In RPS the future velocity, v_{i+1} of a particle at position with a recent velocity, v_i, and the position of the particle are calculated by:

$$v_{i+1} = \omega v_i + \alpha r_1 (\hat{x}_i - x_i) + \omega\beta r_2 (\hat{x}_{hi} - x_i) + \omega\gamma r_3 z$$
$$x_{i+1} = x_i + v_{i+1}$$

where,

- x is the position and v is the velocity of the individual particle. The subscripts i and $i+1$ stand for the recent and the next (future) iterations, respectively.
- r_1, r_2, r_3 are random numbers, $\in [0,1]$; α, β, γ are constants
- ω is inertia weight, $\in [0.01,0.7]$; z is a random velocity vector
- $\hat{x}$ is the best position of a particle; x_h is best position of a randomly chosen other particle from within the swarm

Occasionally, when the process is caught in a local optimum, some perturbation of v may be needed

III. Test of Performance of the Repulsive Particle Swarm Method: The objective at present is to test the performance of RPS method (as modified by us through endowing stronger local search abilities to each particle and defining variable randomized

neighbourhood at different iterations). We have chosen a good number of test functions for this purpose. A synoptic idea about the nature of these test functions is given in the sections that follow. Graphical presentations of most of these functions also are given to facilitate conceptualization of the nature of these functions by visual means.

1. A typical multi-modal function: A multi-modal (non-convex) function (in 2 variables, $m = 2$) is:

$$f(x) = -\cos(x_1)\cos(x_2)\exp\left[-(1/4)\sqrt{x_1^2 + x_2^2}\right]$$

It has global minimum $f(x^*) = -1$ at $x^* = (0,\ 0)$.

2. A typical non-linear function: Consider the function in m variables ($m \geq 2$) that has the optimal value $f(x^*) = 0$ in the search domain $|x_i| \leq 10;\ i = 1, 2, ..., m$ given as

$$f(x) = \sum_{i=2}^{m} \cos[|x_i - x_{i-1}| / |x_i + x_{i-1}|] + (m-1)$$

3. Ackley function: An $m-$variable ($m \geq 1$) function with search domain $[-15 \leq x_i \leq 30]$ for $(i = 1, 2, ..., m)$ given as

$$f(x) = 20 + \exp(1) - 20\exp\left[-0.2\left((\sum_{i=1}^{m} x_i^2)/m\right)^{0.5}\right] - \exp\left[\frac{1}{m}\sum_{i=1}^{m}\cos(2\pi x_i)\right]$$

is called the Ackley function. It is a multi-modal function. The global minimum of this function is $f(x^*) = 0$ for $x^* = (0, 0, ..., 0)$.

4. Beale function: A $2-$variable ($m = 2$) function with search domain $[-4.5 \leq x_i \leq 4.5]$; $(i = 1, 2)$ given as

$$f(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$$

is called the Beale function. It is a unimodal function with the global minimum $f(x^*) = 0$ for $x^* = (3,\ 0.5)$. The two values, (3, 0.5), have a definite relationship with the constants in the function. The constant in the first term (i.e. 1.5) is equal to $3 - 3(0.5)$. The constant in the second term (i.e. 2.25) is equal to $3 - 3(0.5)^2$. The constant in the third term (i.e. 2.625) is equal to $3 - 3(0.5)^3$. Let us now call these parameters (a, b). Then the constant in the first term is $a - ab^1$, the constant in the second term is $a - ab^2$ and the constant in the third term is $a - ab^3$. For any positive a and positive fractional b $(0 < b < 1)$ the function may be generalized. The optimal solution would be (a, b) and $f(x^*) = 0$.

5. Bohachevsky functions: Three 2-variable functions ($m = 2$) characterizing $f(x^*) = 0$, $x^* = (0,\ 0)$ in the search domain $[-100 \leq x_i \leq 100];\ i = 1,\ 2$ are called Bohachevsky functions, which are given as

$$f_1(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$$

$$f_2(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1)\cos(4\pi x_2) + 0.3$$

$$f_3(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1 + 4\pi x_2) + 0.3$$

6. Booth Function: A $2-$variable ($m = 2$) function with search domain $[-10 \leq x_i \leq 10]$; $(i = 1, 2)$ given as

$$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$$

This function is multi-modal with the global minimum $f(x^*) = 0$ at $x^* = (1,\ 3)$.

7. Branin function: A 2-variable function ($m = 2$) characterizing $f(x^*) = 0.397887$, with three global minima in the search domain $[-5 \le x_1 \le 10;\ 0 \le x_2 \le 15]$ is called Branin function, which is given as

$$f(x) = (x_2 - 5x_1^2/(4\pi^2) + 5x_1/\pi - 6)^2 + 10(1-(8\pi)^{-1})\cos(x_1) + 10$$

8. Dixon & Price function: This function is in m ($m \ge 2$) variables with search domain $[-10 \le x_i \le 10]$; ($i = 1, 2, ..., m$) and the minimum $f(x^*) = 0$. It is given as

$$f(x) = (x_1 - 1)^2 + \sum_{i=2}^{m} i(2x_i^2 - x_{i-1})^2$$

9. Easom function: This function is in 2 variables ($m = 2$) with search domain $[-100 \le x_i \le 100]$; ($i = 1, 2$) and $f(x^*) = -1$ at $x^* = (\pi,\ \pi)$. It is given as

$$f(x) = -\cos(x_1)\cos(x_2)\exp[-(x_1 - \pi)^2 - (x_2 - \pi)^2].$$

10. Griewank function: It is a typical multi-modal function with a large number of local minima in the search domain $[-600 \le x_i \le 600]$, $i = 1, 2, ..., m$ and global minimum $f(x^*) = 0$ at $x^* = (0, 0, ..., 0)$. It is given as

$$f(x) = \sum_{i=1}^{m} (x_i^2/4000) - \prod_{i=1}^{m} \cos(x_i/\sqrt{i}) + 1$$

11. Himmelblau function: It is a 2-variable ($m = 2$) function with search domain $[-6 \le x_i \le 6]$; ($i = 1, 2$) and 4 global minima $f(x^*) = 0$, one each in the four Cartesian quadrants. The optimal values of x are: (3,2), (-2.805, 3.131), (-3.779, -3.283) and (3.584, -1.848). The function is written as:

$$f(x) = (x_1 + x_2^2 - 7)^2 + (x_1^2 + x_2 - 11)^2$$

The modified Himmelblau function has only one global optimum $f(x^*) = 0$ at $x^* = (3, 2)$. This (modified) function is given as

$$f(x) = (x_1 + x_2^2 - 7)^2 + (x_1^2 + x_2 - 11)^2 + 0.1[(x_1 - 3)^2 + (x_2 - 2)^2]$$

12. Hump function: It is a 2-variable ($m = 2$) function with search domain $[-5 \le x_i \le 5]$; ($i = 1, 2$) and dual (global) minima $f(x^*) \simeq -1.032$ at $x^* = (\pm 1)$ (0.0898, –0.7126). It is given as

$$f(x) = 4x_1^2 - 2.1x_1^4 + x_1^6/3 + x_1x_2 - 4x_2^2 + 4x_2^4$$

13. Levy function No. 3: It is a 2-variable ($m = 2$) multi-modal function with search domain $[-10 \le x_i \le 10]$. It has some 760 local minima and 18 global minima in this search domain. Its global minimum is $f(x^*) = -176.542$.

$$f(x) = \sum_{i=1}^{5} i\cos[(i+1)x_1 + i] \sum_{i=1}^{5} i\cos[(i+1)x_2 + i]$$

14. Levy function No. 5: It is a 2-variable ($m = 2$) multi-modal function with search domain $[-10 \le x_i \le 10]$. It has some 760 local minima in this search domain. Its global minimum is $f(x^*) = -176.1375$ at $x^* = (-1.3068,\ -1.4248)$.

$$f(x) = \sum_{i=1}^{5} i\cos[(i+1)x_1 + i] \sum_{i=1}^{5} i\cos[(i+1)x_2 + i] + (x_1 + 1.42513)^2 + (x_2 + 0.80032)^2$$

15. Levy function No. 8: It is a 3-variable ($m = 3$) multi-modal function with search domain $[-10 \le x_i \le 10]$. It has some 125 local minima in this search domain. Its only global minimum is $f(x^*) = 0$ at $x^* = (1,\ 1,\ 1)$. This function is specified as

$$f(x) = \sin^2(\pi y_1) + \sum_{i=1}^{m-1} (y_i - 1)^2 [1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2$$

$$where\ y_i = 1 + (x_i - 1)/4\ ;\ i = 1, ..., m.$$

16. Matyas function: It is a 2-variable ($m = 2$) function with search domain $[-10 \le x_i \le 10]$; ($i = 1, 2$) and minimum $f(x^*) = 0$ at $x^* = (0,\ 0)$. It is given as

$$f(x) = 0.26(x_1^2 + x_2^2) - 0.48 x_1 x_2$$

17. Michalewicz function: It is an interesting multi-modal function in the search domain $[0 \le x_i \le \pi]$, $i = 1, 2, ..., m$. It has an additional parameter, p, that determines its surface. For p=10, its global minima at different dimensions (m) are : $f(x^*) = -1.8013$ (m=2), $f(x^*) = -4.6877$ (m=5), $f(x^*) = -6.6809$ (m=7), $f(x^*) = -9.6602$ (m=10), $f(x^*) = -19.6370$ (m=20), $f(x^*) = -29.6309$ (m=30), $f(x^*) = -49.6248$ (m=50). This function is given as

$$f(x) = -\sum_{i=1}^{m} \sin(x_i)(\sin(i x_i^2 / \pi))^{2p}$$

18. Paviani function: It is a 10-variable function ($m = 10$) in the search domain $x_i \in (2,\ 10)$, with $f(x^*) \simeq 45.77847$; $x^* \simeq (9.3502,\ 9.3502, ..., 9.3502)$ given as

$$\sum_{i=1}^{m} \left[\ln^2(x_i - 2) + \ln^2(10 - x_i) \right] - \left[\prod_{i=1}^{n} x_i \right]^{0.2}$$

19. Rastrigin function: It is a typical multi-modal function in m ($m \ge 1$) variables with search domain $[-5.12 \le x_i \le 5.12]$; ($i = 1, 2, ..., m$) and the minimum $f(x^*) = 0$ at $x^* = (0, 0, ..., 0)$. It is a difficult function to optimize. It is given as

$$f(x) = 10m + \sum_{i=1}^{m} (x_i^2 - 10\cos(2\pi x_i))$$

20. Rosenbrock function: This function is in m ($m \ge 2$) variables with search domain $[-5 \le x_i \le 10]$; ($i = 1, 2, ..., m$) and the minimum $f(x^*) = 0$ and $x^* = (1,\ 1, ..., 1)$. It is very similar to the Dixon and Price function. It is often referred to for its very slow convergence in the neighbourhood of the minimum. It is given as

$$\sum_{i=1}^{m-1} [100(x_1^2 - x_{i-1})^2 + (x_i - 1)^2]$$

21. Schwefel function: It is another difficult but interesting multi-modal multi-dimensional function in the search domain $[-500 \le x_i \le 500]$, $i = 1, 2, ..., m$ with its global minimum at $f(x^*) = 0$. It is given as

$$f(x) = 418.9829m - \sum_{i=1}^{m} [x_i \sin(\sqrt{|x_i|})]$$

22. Shekel Function: A 4-variable function ($m = 4$) for parameter p ($2 \le p \le 10$; p is an integer) in the search domain $x_i \in (2,\ 10)$ is given as:

$$f_p(x) = -\sum_{i=1}^{p}\left(\sum_{j=1}^{4}(x_j - a_{ij})^2 + c_i\right)^{-1}$$

The optimal values of $f_p(x^*)$ lie between 10.02 and 10.54 for $2 \le p \le 10$ and $x^* \simeq (4,\ 4,\ 4,\ 4)$. The matrix A and the vector C are given alongside.

$$A = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{bmatrix};\quad C = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.5 \end{bmatrix}$$

23. Shubert function: It is a 2-variable ($m = 2$) typically difficult multi-modal function with search domain $[-10 \le x_i \le 10]$; $(i = 1,2)$ and minimum $f(x^*) \simeq -186.7309$. It is given as

$$f(x) = \prod_{j=1}^{2}\sum_{i=1}^{5}[i\cos((i+1)x_j + i)]$$

24. Trid function: This function is in m ($m \ge 2$) variables with search domain $[-m^2 \le x_i \le m^2]$; $(i = 1,2,...,m)$.The Trid function is given as

$$f(x) = \sum_{i=1}^{m}(x_i - 1)^2 - \sum_{i=2}^{m} x_i x_{i-1}$$

Optimal values of Trid function of different dimensions (m)																
m	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	$f(x^*)$
15	15	28	39	48	55	60	63	64	63	60	55	48	39	28	15	-665
10	10	18	24	28	30	30	28	24	18	10						-210
6	6	10	12	12	10	6										-50

The values of $f(x^*)$ and those of x^* at different m are given in the table above. The pattern observed in the values taken on by decision variables is interesting.

25. Weierstrass function: The Weierstrass function [in its original form, $f(x) = \sum_{k=0}^{\infty} a^k \cos(b^k x)$ while b is an odd integer, $0 < a < 1;\ ab > (1 + 3\pi/2)$] is one of the most notorious functions (with almost fractal surface) that changed the course of history of mathematics. Weierstrass proved that this function is *throughout continuous but nowhere differentiable* (Hobson, 1926). In its altered form (Liang and Suganthan, 2005) this function in m ($m \ge 1$) variables with search domain $[-0.5 \le x_i \le 0.5]$; $(i = 1,2,...,m)$ and the minimum $f(x^*) = 0$ for $x^* = (0,\ 0,...,0)$; $a = 0.5;\ b = 3;\ \mathbb{k} = 20$, is given as

$$f(x) = \sum_{i=1}^{m}\sum_{k=0}^{\mathbb{k}}[a^k \cos(2\pi b^k (x_i + 0.5))] - m\sum_{k=0}^{\mathbb{k}}[a^k \cos(2\pi b^k 0.5)];\ x_i \in [-0.5,\ 0.5];\ i = 1,2,...,m$$

26. Zakharov function: This function is in m ($m \ge 2$) variables with search domain $[-5 \le x_i \le 10]$; $(i = 1,2,...,m)$ and the minimum $f(x^*) = 0$ and $x^* = (0,\ 0,...,0)$. The function is:

$$f(x) = \sum_{i=1}^{m} x_i^2 + \left[\sum_{i=1}^{m} i x_i / 2\right]^2 + \left[\sum_{i=1}^{m} i x_i / 2\right]^4$$

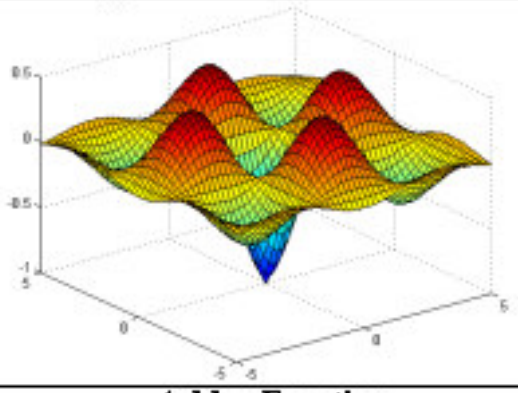
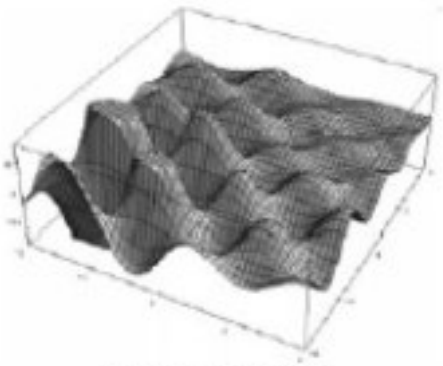
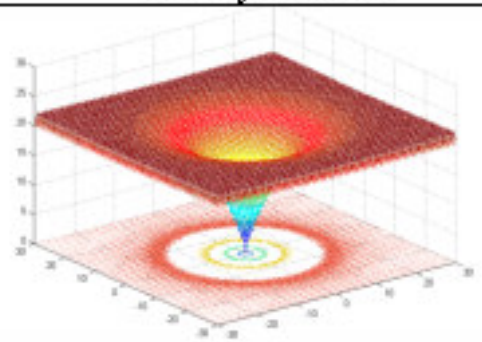
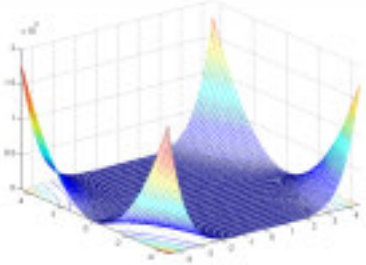
IV. A FORTRAN Computer Program: The functions described above have been used to test the performance of the RPS method. The program is appended.

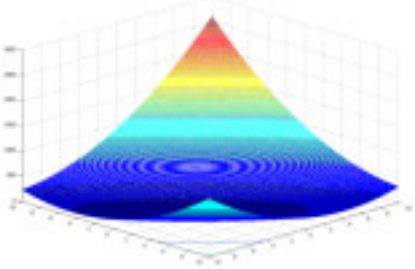
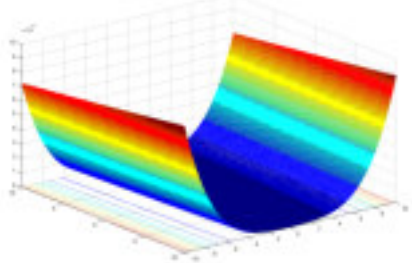
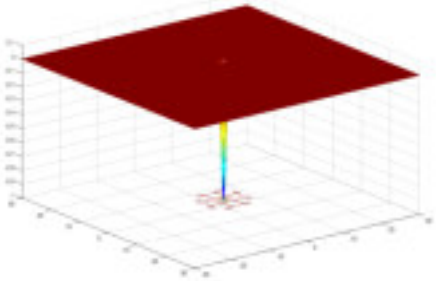
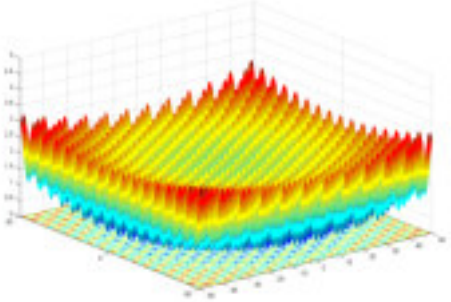
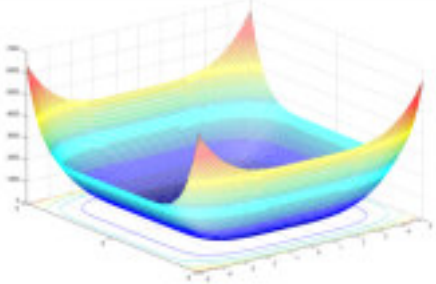
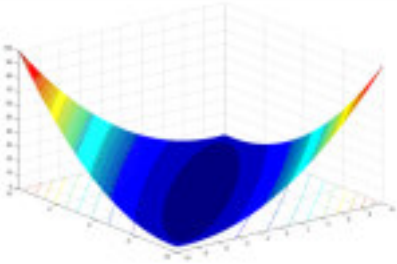
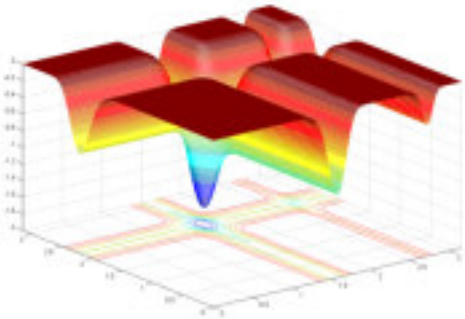
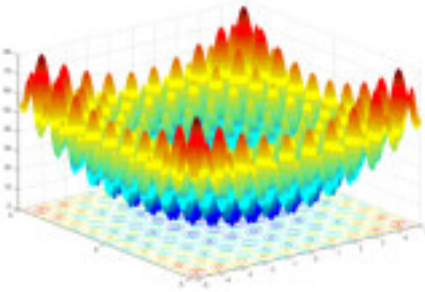
When the program is run, it needs the serial number (KF) to identify the function to optimize and specify its dimensionality (m = no. of variables in the function). The serial no. is indicated by the program. For example, the serial no. (KF) of Ackley function is 10 and it has $m \geq 1$ variables. The user has to feed this information. In some cases, the function has additional parameters (such as the Michalewicz function or Shekel function). These are to be changed in the body of subroutine FUNC(X, M, F) of the program, if needed.

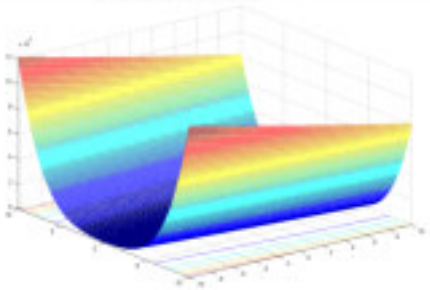
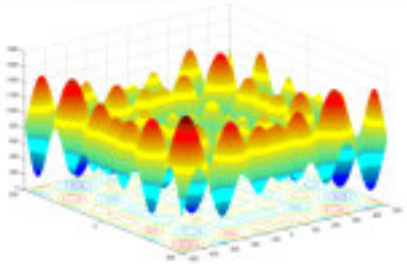
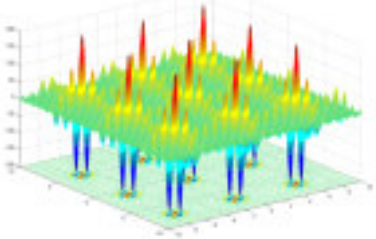
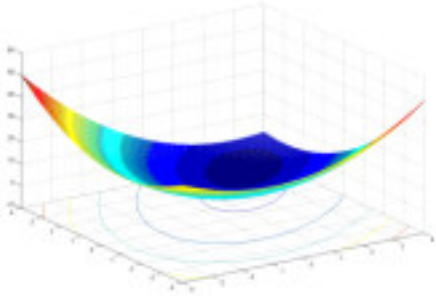
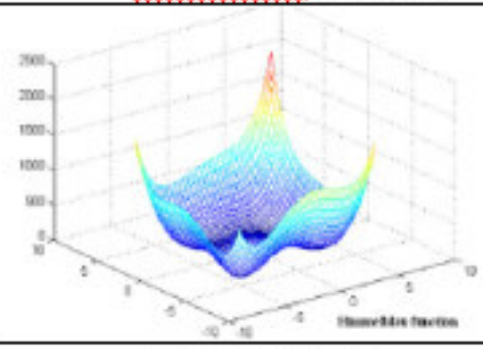
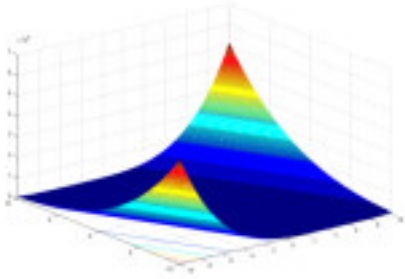
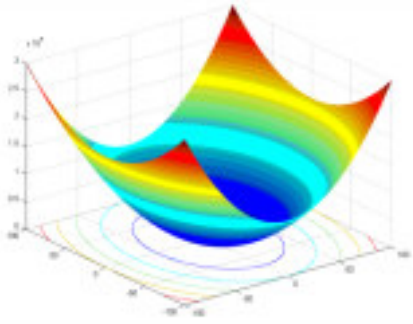
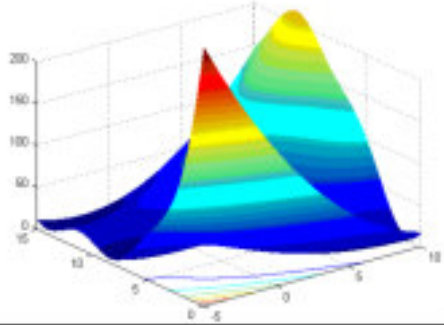
The program specifies a number of parameters (e.g. N=50, NN=25, MX=100, NSTEP=21, ITRN=5000) and A1 (=alpha), A2 (=beta) , A3 (=gamma) and w. The user may suitably change these parameters for sake of experimentation. Necessary comments are given in the program.

V. Conclusion: The program has been run for each function. It has found the optimum value of the function in each case successfully. Some functions with variable dimension (the Weierstrass function in particular) are very difficult to optimize while $m \geq 30$. In some cases, where multiple global optima exist (such as the Himmelblau function), the program finds any one of them, depending on the choice of the random number seed.

Some Important Test Functions for Global Optimization Methods*

A typical multi-modal function	Levy function (#5)
	
Ackley Function	**Beale Function**
	

Booth Function	Dixon & Price Function
	
Easom Function	**Griewank Function**
	
Hump Function	**Matyas Function**
	
Michalewics Function	**Rastrigin Function**
	

Rosenbrock Function	Schwefel Function
	
Shubert Function	**Trid Function**
	
Himmelblau Function	**Zakharov Function**
	
Bohachevsky function	**Branin function**
	

* Graphical presentations (of most of the functions) are creditable to Dr. AR Hedar, Dept. of Computer Science, Faculty of Computer & Information Sciences, Assiut University, Egypt. A few of the functions and their properties mentioned in different pages at the site (below) may, however, be taken with caution.

http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/go.htm

Bibliography

- Bauer, J.M.: "Harnessing the Swarm: Communication Policy in an Era of Ubiquitous Networks and Disruptive Technologies", *Communications and Strategies*, 45, 2002.
- Eberhart R.C. and Kennedy J.: "A New Optimizer using Particle Swarm Theory", *Proceedings Sixth Symposium on Micro Machine and Human Science*, pp. 39–43. IEEE Service Center, Piscataway, NJ, 1995.
- Fleischer, M.: "Foundations of Swarm Intelligence: From Principles to Practice", Swarming Network Enabled C4ISR, arXiv:nlin.AO/0502003 v1 2 Feb 2005.
- Hobson, E.W.: *The Theory of Functions of a Real Variable*, Vol-2 (2nd ed) Cambridge Univ. Press, London, 1926.
- Huang, V.L., Suganthan, P.N. and Liang, J.J. "Comprehensive Learning Particle Swarm Optimizer for Solving Multiobjective Optimization Problems", *International Journal of Intelligent Systems*, 21, pp.209–226 (Wiley Periodicals, Inc. Published online in Wiley InterScience www.interscience.wiley.com) , 2006
- Jung, B.S. and Karney, B.W.: "Benchmark Tests of Evolutionary Computational Algorithms", Environmental Informatics Archives (International Society for Environmental Information Sciences), 2, pp. 731-742, 2004.
- Kuester, J.L. and Mize, J.H.: *Optimization Techniques with Fortran*, McGraw-Hill Book Co. New York, 1973.
- Liang, J.J. and Suganthan, P.N. "Dynamic Multi-Swarm Particle Swarm Optimizer", *International Swarm Intelligence Symposium*, IEEE # 0-7803-8916-6/05/$20.00. pp. 124-129, 2005.
- Mishra, S.K.: "Some Experiments on Fitting of Gielis Curves by Simulated Annealing and Particle Swarm Methods of Global Optimization", *Social Science Research Network* (SSRN): http://ssrn.com/abstract=913667, Working Papers Series, 2006 (a).
- Mishra, S.K.: "Least Squares Fitting of Chacón-Gielis Curves by the Particle Swarm Method of Optimization", *Social Science Research Network* (SSRN), Working Papers Series, http://ssrn.com/abstract=917762 , 2006 (b).
- Parsopoulos, K.E. and Vrahatis, M.N., "Recent Approaches to Global Optimization Problems Through Particle Swarm Optimization", *Natural Computing*, 1 (2-3), pp. 235-306, 2002.
- Simon, H.A.: *Models of Bounded Rationality*, Cambridge Univ. Press, Cambridge, MA, 1982.

Author's Contact: mishrasknehu@hotmail.com

```
C     PROGRAM TO FIND GLOBAL MINIMUM BY REPULSIVE PARTICLE SWARM METHOD
C     WRITTEN BY SK MISHRA, DEPT. OF ECONOMICS, NEHU, SHILLONG (INDIA)
      PARAMETER (N=50,NN=25,MX=100,NSTEP=9,ITRN=10000)
C     N = POPULATION SIZE. IN MOST OF THE CASES N=30 IS OK. ITS VALUE
C     MAY BE INCREASED TO 50 OR 100 TOO. THE PARAMETER NN IS THE SIZE OF
C     RANDOMLY CHOSEN NEIGHBOURS. 15 TO 25 (BUT SUFFICIENTLY LESS THAN
C     N) IS A GOOD CHOICE. MX IS THE MAXIMAL SIZE OF DECISION VARIABLES.
C     IN F(X1, X2,...,XM) M SHOULD BE LESS THAN OR EQUAL TO MX. ITRN IS
C     THE NO. OF ITERATIONS. IT MAY DEPEND ON THE PROBLEM. 200(AT LEAST)
C     TO 500 ITERATIONS MAY BE GOOD ENOUGH. BUT FOR FUNCTIONS LIKE
C     ROSENBROCKOR GRIEWANK OF LARGE SIZE (SAY M=30) IT IS NEEDED THAT
C     ITRN IS LARGE, SAY 5000 OR EVEN 10000. SIGMA INTRODUCES DIVERSITY
C     AND HELPS THE SEARCH JUMP OUT OF LOCAL OPTIMA.
C     NSTEP DOES LOCAL SEARCH BY TUNNELLING AND WORKS WELL BETWEEN 5 AND
C     15, WHICH IS MUCH ON THE HIGHER SIDE.
C     THE SUBROUTINE FUNC( ) DEFINES THE FUNCTION TO BE OPTIMIZED.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /RNDM/IU,IV
      COMMON /KFF/KF,nfcall
      INTEGER IU,IV
      DIMENSION X(N,MX),V(N,MX),A(MX),VI(MX)
      DIMENSION XX(N,MX),F(N),R(3),V1(MX),V2(MX),V3(MX),V4(MX),BST(MX)
C     A1 A2 AND A3 ARE CONSTANTS AND W IS THE INERTIA WEIGHT.
C     OCCASIONALLY, TINKERING WITH THESE VALUES, ESPECIALLY A3, MAY BE
C     NEEDED.
      DATA A1,A2,A3,W,SIGMA /.5D00,.5D00,.0005D00,.5D00,1.D-03/

      WRITE(*,*)'-----------------------------------------------------'
      WRITE(*,*)'KF=1 SIMPLE QUADRATIC FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=2 ROSENBROCK FUNCTION M-VARIABLES M=DEFINE (?)'
      WRITE(*,*)'KF=3 LEVY NO 5 FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=4 LEVYS NO 8 FUCTION 3-VARIABLES M=3'
      WRITE(*,*)'KF=5 RASTRIGIN FUNCTION M-VARIABLES M=DEFINE (?)'
      WRITE(*,*)'KF=6 HIMMELBLAU FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=7 MODIFIED HIMMELBLAU FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=8 GRIEWANK FUNCTION M-VARIABLES M=DEFINE (?)'
      WRITE(*,*)'KF=9 EASOM FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=10 ACKLEY FUNCTION M-VARIABLES M=DEFINE (?)'

      WRITE(*,*)'KF=11 BEALE FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=12 GENERALIZED BEALE FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=13 BOOTH FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=14 MICHALEWICZ FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=15 SCHWEFEL FUNCTION M-VARIABLES M=DEFINE (?)'
      WRITE(*,*)'KF=16 SHUBERT FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=17 HUMP FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=18 MATYAS FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=19 DIXON & PRICE FUNCTION M-VARIABLES M=DEFINE (?)'
      WRITE(*,*)'KF=20 TRID FUNCTION M-VARIABLES M=DEFINE (?)'

      WRITE(*,*)'KF=21 ZAKHAROV FUNCTION M-VARIABLES M=DEFINE (?)'
      WRITE(*,*)'KF=22 A TYPICAL MULTI-MODAL FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=23 LEVY NO 3 FUNCTION 2-VARIABLES M=2'
      WRITE(*,*)'KF=24 WEIERSTRASS FUNCTION M-VARIABLES M=DEFINE (?)'
      WRITE(*,*)'KF=25 SHEKEL FUNCTION 4-VARIABLES M=4'
      WRITE(*,*)'KF=26 PAVIANI FUNCTION 10-VARIABLES M=10'
      WRITE(*,*)'KF=27 BRANIN FUNCTION (#1) 2-VARIABLES M=2'
      WRITE(*,*)'KF=28 BRANIN FUNCTION (#2) 2-VARIABLES M=2'
      WRITE(*,*)'KF=29 A TYPICAL N-LINEAR FUNCTION M-VARIABLES M=(?)'
      WRITE(*,*)'KF=30,31,32 : 3 BOHACHEVSKY FUNCTIONS 2-VARIABLES M=2'

      WRITE(*,*)'-----------------------------------------------------'
      WRITE(*,*)'CHOOSE KF AND SPECIFY M'
      READ(*,*) KF,M
      nfcall=0
      LCOUNT=0
```

```
      WRITE(*,*)'4-DIGITS SEED FOR RANDOM NUMBER GENERATION'
      READ(*,*) IU
      DATA ZERO,ONE,FMIN /0.0D00,1.0D00,1.0E30/
C     GENERATE N-SIZE POPULATION OF M-TUPLE PARAMETERS X(I,J) RANDOMLY
      DO I=1,N
        DO J=1,M
        CALL RANDOM(RAND)
         X(I,J)=(RAND-0.5D00)*10

C     WE GENERATE RANDOM(-5,5). HERE MULTIPLIER IS 10. TINKERING IN SOME
C     CASES MAY BE NEEDED
         ENDDO
        F(I)=1.0E30
      ENDDO
C     INITIALISE VELOCITIES V(I) FOR EACH INDIVIDUAL IN THE POPULATION
      DO I=1,N
      DO J=1,M
      CALL RANDOM(RAND)
       V(I,J)=(RAND-.5D+00)
C        V(I,J)=RAND
      ENDDO
      ENDDO
      ZZZ=1.0E+30
      ICOUNT=0
      DO 100 ITER=1,ITRN

C     LET EACH INDIVIDUAL SEARCH FOR THE BEST IN ITS NEIGHBOURHOOD
        DO I=1,N
           DO J=1,M
           A(J)=X(I,J)
           VI(J)=V(I,J)
           ENDDO
           CALL LSRCH(A,M,VI,NSTEP,FI)
           IF(FI.LT.F(I)) THEN
            F(I)=FI
            DO IN=1,M
            BST(IN)=A(IN)
            ENDDO
C     F(I) CONTAINS THE LOCAL BEST VALUE OF FUNCTION FOR ITH INDIVIDUAL
C     AND XX(I,J) IS THE M-TUPLE VALUE OF X ASSOCIATED WITH THE LOCAL BEST F(I)
             DO J=1,M
             XX(I,J)=A(J)
             ENDDO
             ENDIF
        ENDDO

C      NOW LET EVERY INDIVIDUAL RANDOMLY COSULT NN(<<N) COLLEAGUES AND
C      FIND THE BEST AMONG THEM

      DO I=1,N
C       CHOOSE NN COLLEAGUES RANDOMLY AND FIND THE BEST AMONG THEM
          BEST=1.0E30
           DO II=1,NN
                CALL RANDOM(RAND)
                NF=INT(RAND*N)+1
                IF(BEST.GT.F(NF)) THEN
                BEST=F(NF)
                NFBEST=NF
                ENDIF
           ENDDO
C     IN THE LIGHT OF HIS OWN AND HIS BEST COLLEAGUES EXPERIENCE, THE
C     INDIVIDUAL I WILL MODIFY HIS MOVE AS PER THE FOLLOWING CRITERION
C     FIRST, ADJUSTMENT BASED ON ONES OWN EXPERIENCE
C     AND OWN BEST EXPERIENCE IN THE PAST (XX(I))
           DO J=1,M
           CALL RANDOM(RAND)
           V1(J)=A1*RAND*(XX(I,J)-X(I,J))
```

```
C     THEN BASED ON THE OTHER COLLEAGUES BEST EXPERIENCE WITH WEIGHT W
C     HERE W IS CALLED AN INERTIA WEIGHT 0.01< W < 0.7
C     A2 IS THE CONSTANT NEAR BUT LESS THAN UNITY
           CALL RANDOM(RAND)
           V2(J)=V(I,J)
           IF(F(NFBEST).LT.F(I)) THEN
           V2(J)=A2*W*RAND*(XX(NFBEST,J)-X(I,J))
           ENDIF
C     THEN SOME RANDOMNESS AND A CONSTANT A3 CLOSE TO BUT LESS THAN UNITY
           CALL RANDOM(RAND)
           RND1=RAND
           CALL RANDOM(RAND)
            V3(J)=A3*RAND*W*RND1
C             V3(J)=A3*RAND*W
C     THEN ON PAST VELOCITY WITH INERTIA WEIGHT W
           V4(J)=W*V(I,J)
C     FINALLY A SUM OF THEM
           V(I,J)= V1(J)+V2(J)+V3(J)+V4(J)
           ENDDO
      ENDDO
C     CHANGE X
      DO I=1,N
      DO J=1,M
      RANDS=0.D00
      CALL RANDOM(RAND)
      IF(DABS(RAND-.5D00).LT.SIGMA) RANDS=RAND-0.5D00
C     SIGMA CONDITIONED RANDS INTRODUCES CHAOTIC ELEMENT IN TO LOCATION
      X(I,J)=X(I,J)+V(I,J)*(1.D00+RANDS)
      ENDDO
      ENDDO

       DO I=1,N
         IF(F(I).LT.FMIN) THEN
         FMIN=F(I)
         II=I
         DO J=1,M
         BST(J)=XX(II,J)
         ENDDO
         ENDIF
         ENDDO
      Z=FMIN
      IF(LCOUNT.EQ.100) THEN
      LCOUNT=0
      WRITE(*,*)'OPTIMAL SOLUTION UPTO THIS'
      WRITE(*,*)'X = ',(BST(J),J=1,M),' MIN F = ',FMIN
      write(*,*)'No. of function calls = ',nfcall
      ENDIF
      LCOUNT=LCOUNT+1
  999 FORMAT(5F15.6)


  100 CONTINUE
      WRITE(*,*)'OVER'
      END

      SUBROUTINE LSRCH(A,M,VI,NSTEP,FI)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /KFF/KF,nfcall
      COMMON /RNDM/IU,IV
      INTEGER IU,IV
      DIMENSION A(*),B(100),VI(*)
      AMN=1.0E30
      DO J=1,NSTEP
         DO JJ=1,M
         B(JJ)=A(JJ)+(J-NSTEP/2-1)*VI(JJ)
         ENDDO
```

```
202:       CALL FUNC(B,M,FI)
203:         IF(FI.LT.AMN) THEN
204:         AMN=FI
205:         DO JJ=1,M
206:         A(JJ)=B(JJ)
207:         ENDDO
208:         ENDIF
209:       ENDDO
210:       FI=AMN
211:       RETURN
212:       END
213:
214:       SUBROUTINE RANDOM(RAND1)
215:        DOUBLE PRECISION  RAND1
216:        COMMON /RNDM/IU,IV
217:       INTEGER IU,IV
218:        RAND=REAL(RAND1)
219:        IV=IU*65539
220:        IF(IV.LT.0) THEN
221:        IV=IV+2147483647+1
222:        ENDIF
223:        RAND=IV
224:        IU=IV
225:        RAND=RAND*0.4656613E-09
226:        RAND1=DBLE(RAND)
227:        RETURN
228:        END
229:
230:       SUBROUTINE FUNC(X,M,F)
231:       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
232:        COMMON /RNDM/IU,IV
233:       COMMON /KFF/KF,nfcall
234:        INTEGER IU,IV
235:       DIMENSION X(*),Y(100)
236:       PI=4.D+00*DATAN(1.D+00)
237:       nfcall=nfcall+1
238:       IF(KF.EQ.1) THEN
239:       F=-3803.84-138.08*X(1)-232.92*X(2)+128.08*X(1)**2+203.64*X(2)**2+
240:      & 182.25*X(1)*X(2)
241:       RETURN
242:       ENDIF
243:       IF(KF.EQ.2) THEN
244: C     ROSENBROCK FUNCTION
245:       F=0.D00
246:       DO I=1,M-1
247:       F=F+ (100.D00*(X(I+1)-X(I)**2)**2 + (X(I)-1.D00)**2)
248:       ENDDO
249:       RETURN
250:       ENDIF
251:       IF(KF.EQ.3) THEN
252: C     LEVY # 5 (LEVY ET AL. 1981) ------------------------------------
253:             F1=0.0D+00
254:           F2=0.0D+00
255:         DO I=1,5
256:        F1=F1+(I*DCOS((I-1)*X(1)+I))
257:           F2=F2+(I*DCOS((I+1)*X(2)+I))
258:           ENDDO
259:           F3=(X(1)+1.42513D+00)**2
260:          F4=(X(2)+0.80032D+00)**2
261:          F=(F1*F2) + (F3+F4)
262:          RETURN
263:          ENDIF
264:        IF(KF.EQ.4) THEN
265: C     LEVY # 8 FUNCTION -----------------------------------------------
266:          DO I=1,3
267:          Y(I)=1.D+00+(X(I)-1.D+00)/4.D+00
268:          END DO
```

```
269:           F1=DSIN(PI*Y(1))**2
270:          F3=(Y(3)-1.D+00)**2
271:         F2=0.D+00
272:          DO I=1,2
273:           F2=F2+((Y(I)-1.D+00)**2)*(1.D+00+10.D+00*(DSIN(PI*Y(I+1)))**2)
274:           ENDDO
275:         F=F1+F2+F3
276:         RETURN
277:         ENDIF
278:        IF(KF.EQ.5) THEN
279: C      RASTRIGIN'S FUNCTION
280:        F=0
281:        DO I=1,M
282:        F=F+(X(I)**2-10*DCOS(2*PI*X(I))+10)
283:        ENDDO
284:        RETURN
285:        ENDIF
286:        IF(KF.EQ.6) THEN
287: C      HIMMELBLAU FUNCTION. IT HAS MULTIPLE (4) GLOBAL OPTIMA
288:        F= (X(1)**2+X(2)-11)**2+  (X(1)+X(2)**2-7)**2
289:        RETURN
290:        ENDIF
291:        IF(KF.EQ.7) THEN
292: C      MODIFIED HIMMELBLAU FUNCTION. IT HAS ONLY ONE GLOBAL OPTIMUM
293:        F= (X(1)**2+X(2)-11)**2+(X(1)+X(2)**2-7)**2+0.1D00*((X(1)-3)**2 +
294:       1  (X(2)-2)**2)
295:        RETURN
296:        ENDIF
297:        IF(KF.EQ.8) THEN
298: C      GRIEWANK FUNCTION
299:        F1=0.D00
300:        F2=1.0D00
301:        DO I=1,M
302:        F1=F1+X(I)**2
303:        FI=DFLOAT(I)
304:        F2=F2*DCOS(X(I)/DSQRT(FI))
305:        ENDDO
306:        F=F1/4000.D00-F2+1.0D00
307:        RETURN
308:        ENDIF
309:
310:        IF(KF.EQ.9) THEN
311: C      EASOM FUNCTION
312:        F=-DCOS(X(1))*DCOS(X(2))*DEXP(-(X(1)-PI)**2 -(X(2)-PI)**2)
313:        RETURN
314:        ENDIF
315:
316:        IF(KF.EQ.10) THEN
317: C      ACKLEY FUNCTION
318:
319:        F=20.D00+DEXP(1.D00)
320:        F1=0.D00
321:        F2=0.D00
322:          DO I=1,M
323:           F1=F1+X(I)**2
324:           F2=F2+DCOS(2*PI*X(I))
325:          ENDDO
326:        F1=-20*DEXP(-0.2D00*DSQRT(F1/M))
327:        F2=-DEXP(F2/M)
328:        F=F+F1+F2
329:        RETURN
330:        ENDIF
331:
332:        IF(KF.EQ.11) THEN
333: C      BEALE FUNCTION
334:
335:        F1=(1.5D00-X(1)+X(1)*X(2))**2
```

```
336:       F2=(2.25D00-X(1)+X(1)*X(2)**2)**2
337:       F3=(2.625D00-X(1)+X(1)*X(2)**3)**2
338:       F=F1+F2+F3
339:
340:       RETURN
341:       ENDIF
342:       IF(KF.EQ.12) THEN
343: C     GENERALIZED BEALE FUNCTION
344: C     PARAMETERS OF THE FUNCTION MM => 1; C1 => 1; 0 < C2 < 1
345:           MM=7   ! DEFINE
346:           C1=9.D00 ! DEFINE
347:           C2=0.1D00   ! DEFINE
348:
349:       F=0.D00
350:       DO I=1,MM
351:       D= C1*(1.D00-C2**I)
352:       F=F+(D-X(1)+X(1)*X(2)**I)**2
353:       ENDDO
354:       RETURN
355:       ENDIF
356:
357:       IF(KF.EQ.13) THEN
358: C     BOOTH FUNCTION
359:       F=(X(1)+2*X(2)-7.0D00)**2+(2*X(1)+X(2)-5.0D00)**2
360:       RETURN
361:       ENDIF
362:
363:       IF(KF.EQ.14) THEN
364: C     MICHALEWICZ FUNCTION   [ 0 <= X(I) <= PI ] MP IS A PARAMETER
365: C     min f= -1.8013(M=2); -4.6877(M=5); -6.6809(M=7); -9.6602(M=10)
366: C     -19.6370(M=20); -29.6309(M=30); -49.6248(M=50), -99.620194(M=100)
367: C     for MP=10.
368:       MP=10   ! SET IT TO THE DESIRED VALUE
369:
370:       F=0.D00
371:       F1=0.D00
372:       DO I=1,M
373:       F=F-DSIN(X(I))*(DSIN(I*X(I)**2/PI))**(2*MP)
374:       IF(X(I).LT.0.D00 .OR. X(I).GT.PI) F1=F1 + F**2 + X(I)**2
375:       ENDDO
376:       IF(F1.GT.0) F=F1
377:       RETURN
378:       ENDIF
379:
380:       IF(KF.EQ.15) THEN
381: C     SCHWEFEL FUNCTION
382:       F=0.D00
383:       F1=0.D00
384:       DO I=1,M
385:       F=F+ X(I)*DSIN(DSQRT(DABS(X(I))))
386:       IF(DABS(X(I)).GT.500) F1=(500.D00+DABS(X(I)))**2
387:       ENDDO
388:       F=418.9829D00*M - F
389:       IF(F1.GT.0.D00) F=F1
390:       RETURN
391:       ENDIF
392:
393:       IF(KF.EQ.16) THEN
394: C     SHUBERT FUNCTION
395:       F1=0.D00
396:       F2=0.D00
397:       FP=0.D00
398:       DO I=1,5
399:       F1=F1+I*DCOS((I+1.D00)*X(1)+I)
400:       F2=F2+I*DCOS((I+1.D00)*X(2)+I)
401:       ENDDO
402:       F=F1*F2
```

```
403:       DO I=1,M
404:       IF(DABS(X(I)).GT.10.D00) FP=FP+X(I)**2
405:       ENDDO
406:       IF(FP.GT.0) F=FP
407:       RETURN
408:       ENDIF
409:
410:       IF(KF.EQ.17) THEN
411: C     HUMP FUNCTION
412:       F=4*X(1)**2 - 2.1D00*X(1)**4 + (X(1)**6)/3.D00 + X(1)*X(2) -
413:      & 4*X(2)**2 + 4*X(2)**4
414:       RETURN
415:       ENDIF
416:
417:       IF(KF.EQ.18) THEN
418: C     MATYAS FUNCTION
419:       F=0.26D00*(X(1)**2 + X(2)**2) - 0.48D00*X(1)*X(2)
420:       RETURN
421:       ENDIF
422:
423:       IF(KF.EQ.19) THEN
424: C     DIXON & PRICE FUNCTION
425:       F=0.D00
426:       DO I=2, M
427:       F=F + I*(2*X(I)**2-X(I-1))**2
428:       ENDDO
429:       F=F+(X(1)-1.D00)**2
430:       RETURN
431:       ENDIF
432:
433:       IF(KF.EQ.20) THEN
434: C     TRID FUNCTION
435:       F1=0.D00
436:       F2=0.D00
437:       DO I=1, M
438:       F1=F1+(X(I)-1.D00)**2
439:       ENDDO
440:       DO I=2, M
441:       F2=F2+X(I)*X(I-1)
442:       ENDDO
443:       F=F1-F2
444:       RETURN
445:       ENDIF
446:
447:       IF(KF.EQ.21) THEN
448: C     ZAKHAROV FUNCTION
449:       F1=0.D00
450:       F2=0.D00
451:       DO I=1, M
452:       F1=F1+ X(I)**2
453:       F2=F2 + I*X(I)/2.D00
454:       ENDDO
455:       F=F1+F2**2+F2**4
456:       RETURN
457:       ENDIF
458:
459:       IF(KF.EQ.22) THEN
460: C     A TYPICAL MULTIMODAL FUNCTION
461:       F=-DCOS(X(1))*DCOS(X(2))*DEXP(-(X(1)**2 + X(2)**2)**2/4.D00)
462:       RETURN
463:       ENDIF
464:
465:       IF(KF.EQ.23) THEN
466: C     LEVY # 3 (LEVY ET AL. 1981) ----------------------------------
467:           F1=0.0D+00
468:           F2=0.0D+00
469:           FP=0.D00
```

```
470:            DO I=1,5
471:            F1=F1+(I*DCOS((I-1)*X(1)+I))
472:            F2=F2+(I*DCOS((I+1)*X(2)+I))
473:            ENDDO
474:           F=F1*F2
475:           DO I=1,M
476:           IF(DABS(X(I)).GT.10.D00) FP=FP+X(I)**2
477:           ENDDO
478:           IF(FP.GT.0.D00) F=FP
479:           RETURN
480:           ENDIF
481:
482:       IF(KF.EQ.24) THEN
483: C     WEIERSTRASS FUNCTION ----------------------------------
484:       DATA AX,BX,KMAX/0.5D00,3.D00,20/
485:       f1=0.D00
486:       f2=0.D00
487:       fp=0.D00
488:       NVIO=0
489:       DO I=1,M
490:       IF(DABS(X(I)).GT.0.5D00) NVIO=1
491:       ENDDO
492:                 IF(NVIO.NE.1) THEN
493:       DO I=1,M
494:       S=0.D00
495:           DO KK=1,KMAX+1
496:           K=KK-1
497:           S=S+(AX**K*DCOS(2*PI*BX**K*(X(I)+0.5D00)))
498:           ENDDO
499:       F1=F1+S
500:       ENDDO
501:
502:       DO KK=1,KMAX+1
503:       K=KK-1
504:       F2=F2+(AX**K*DCOS(2*PI*BX**K*0.5D00))
505:       ENDDO
506:       F=F1-M*F2
507:       RETURN
508:                  ELSE
509: C     EXPONENTIAL PENALTY ON VIOLATION OF LIMITS ON X(I)
510:       DO I=1,M
511:        FP=FP+DEXP(DABS(X(I)))
512:       ENDDO
513:       F=FP
514:       RETURN
515:                 ENDIF
516:       ENDIF
517:
518:       IF(KF.EQ.25) THEN
519: C     SHEKEL FUNCTION
520:       DATA NR /8/ ! NR IS INTEGER. IT WILL LIE BETWEEN 2 AND 10
521: C     THE VALUE OF NR MAY BE CHANGED BY THE USER
522:       CALL SHEKEL(M,NR,X,F)
523:       RETURN
524:       ENDIF
525:
526:       IF(KF.EQ.26) THEN
527: C     PAVIANI FUNCTION
528:       F1=0.D00
529:       F2=1.D00
530:       FP=0.D00
531:       DO I=1,M
532:       F1=F1+ DLOG(X(I)-2.D00)**2+DLOG(10.D00-X(I))**2
533:       F2=F2*X(I)
534:       IF(X(I).GE.10.D00 .OR. X(I).LE. 2.D00) FP=FP+ X(I)**2
535:       ENDDO
536:       F=F1-F2**0.2
```

```
      IF(FP.GT.0.D00)F=FP
      RETURN
      ENDIF

      IF(KF.EQ.27) THEN
C     BRANIN FUNCTION #1
      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
      FP=0.D00
      DO I=1,M
      IF(DABS(X(I)).GT.10.D00) FP=FP+X(I)**2
      ENDDO
      IF(FP.GT.0.D00) F=FP
      RETURN
      ENDIF
      IF(KF.EQ.28) THEN
C     BRANIN FUNCTION #2
      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
      ENDIF

      IF(KF.EQ.29) THEN
C     A TYPICAL NONLINEAR MULTI-MODAL FUNCTION
      F=0.D00
      FP=0.D00
      IF(DABS(X(1)).GT.10.D00) FP=FP+DEXP(DABS(X(1)))
      DO I=2,M
      F=F+DCOS(  DABS(X(I)-X(I-1)) / DABS(X(I-1)+X(I)) )
      IF(DABS(X(I)).GT.10.D00) FP=FP+DEXP(DABS(X(I)))
      ENDDO
      F=F+(M-1.D00)
C     IF 0.001*X(1) IS ADDED TO F, IT BECOMES UNIMODAL
      F=F+0.001*X(1)
      IF(FP.GT.0.D00) F=FP
      RETURN
      ENDIF

      IF(KF.EQ.30) THEN
C     BOHACHEVSKY FUNCTION #1
      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
      ENDIF

      IF(KF.EQ.31) THEN
C     BOHACHEVSKY FUNCTION #2
      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1))*DCOS(4*PI*X(2))+0.3D00
      RETURN
      ENDIF

      IF(KF.EQ.32) THEN
C     BOHACHEVSKY FUNCTION #3
      F=X(1)**2+2*X(2)**2-0.3D00*DCOS(3*PI*X(1)+4*PI*X(2))+0.3D00
      RETURN
      ENDIF

      WRITE(*,*)'FUNCTION NOT DEFINED. PROGRAM ABORTED'
      STOP
      END

      SUBROUTINE SHEKEL(M,NR,X,F)
C     SHEKEL FUNCTION FOR TEST OF GLOBAL OPTIMIZATION METHODS
      PARAMETER(NROW=10,NCOL=4)
      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.,
```

```
604:      & 8.,8.,6.,6.,6.,6.,3.,7.,3.,7.,2.,9.,2.,9.,5.,5.,3.,3.,8.,1.,8.,
605:      & 1.,6.,2.,6.,2.,7.,3.6D00,7.,3.6D00/
606:       DATA (C(I),I=1,NROW)/0.1D00,0.2D00,0.2D00,0.4D00,0.4D00,0.6D00,
607:      & 0.3D00,0.7D00,0.5D00,0.5D00/
608:       F=0.D00
609:       DO I=1,NR
610:         S=0.D00
611:         DO J=1,M
612:         S=S+(X(J)-A(I,J))**2
613:         ENDDO
614:       F=F-1.D00/(S+C(I))
615:       ENDDO
616:       RETURN
617:       END
```