function [best\_buffalo,fmin]=Abo\_algorithmv6(n)

clear

clc

if nargin<1,

% Number of buffalos (or different solutions)

n=10;

end

%% Change this if you want to get better results

% list of paramters

m\_k=0;

lp1=0.7;

lp2=0.5;

bg=0;

w\_k=0;

N\_IterTotal=2000;

nd=2; %% Simple bounds of the search domain

Lb=0\*ones(1,nd); % Lower bounds

Ub=15\*ones(1,nd); % Upper bounds

% Lb= [-5 0];

% Ub= [10 15];

%

% Lb= [-5.12, -5.12];

% Ub= [5.12, 5.12];

%beala function

% [ x , y ]

Lb= [-4.5, -4.5];

Ub= [4.5, 4.5];

% tuning = 20;

sum =0;

% overall\_best = fobj(Lb+(Ub-Lb).\*rand(size(Lb)));

for t=1:1;

% Random initial solutions

disp('buffalo(i,:) bp\_k(i)');

for i=1:n,

buffalo(i,:)=Lb+(Ub-Lb).\*rand(size(Lb));

w\_array(i,:)= buffalo(i,:);

bp\_k(i,:)= buffalo(i,:);

disp(strcat(num2str(buffalo(i,:))));

end

% % for test

% buffalo(1,:)=[7, 9];

% buffalo(2,:)=[11, 15];

% buffalo(3,:)=[4, 15];

% for i=1:3,

% % buffalo(i,:)=Lb+(Ub-Lb).\*rand(size(Lb));

% w\_array(i,:)= buffalo(i,:);

% bp\_k(i,:)= buffalo(i,:);

% disp(strcat(num2str(buffalo(i,:)),' =[',num2str(bp\_k(i)),']'));

% end

% best\_buffalo=[11, 15];

% Get the current best

fitness=10^10\*ones(n,1);

[fmin,best\_buffalo,buffalo,fitness]=get\_best\_buffalo(buffalo,buffalo,fitness);

% best\_buffalo=[11, 15];

N\_iter=0;

%% Starting iterations

for iter=1:N\_IterTotal,

for jj=1:n

s=buffalo(jj,:);

w=w\_array(jj,:);

%disp(strcat(num2str(jj),' - buffalo before:[',num2str(s),']'));

% Step2. Update the buffalos exploitation using Equation (3.1)

s=s + lp1\*(best\_buffalo-w)\*rand + lp2\*(bp\_k(jj,:)\*rand - w);

disp(strcat(num2str(jj),' - buffalo after :[',num2str(s),']=',num2str(fobj(s))));

%Step2 Update the location of buffalos using (3.2):

w =((w+ s ))/rand;

% Apply simple bounds/limits

s=simplebounds(s,Lb,Ub);

w\_array(jj,:)=w;

%disp(strcat('buffalo :[',num2str(s),']'));

% disp(strcat('w :[',num2str(w),']'));

% Evaluating all new solutions

if fobj(s)< fobj(buffalo(jj,:)) %fnew<fitness(jj),

buffalo(jj,:)=s;

end

if fobj(s)<fobj(bp\_k(jj,:)),

%buffalo(jj,:)=s;

bp\_k(jj,:)=s;%fobj(bp\_k(jj,:));

end

% find global best

%fnew=fobj(s);

if fobj(s)<fobj(best\_buffalo),

%fitness(jj)=fnew;

best\_buffalo=s;

end

end

% Find the current buffalo

%[fmin,K]=min(fitness) ;

%best\_buffalo=buffalo(K,:);

best\_buffalo

fmin= fobj(best\_buffalo)

% disp(strcat('bestbuffalo :[',num2str(best\_buffalo),'] = ',num2str(fmin),''));

%buffalo

end %% End of iterations

end

%Avg= sum/tuning;

%Avg

%overall\_best

%overall\_best\_buffalo

%% Post-optimization processing

%% Display all the buffalos

% disp(strcat('Total number of iterations=',num2str(N\_iter)));

% fmin

% best\_buffalo

% fmin

%buffalo

%% --------------- All subfunctions are list below ------------------

%% Find the current best buffalo

function [fmin,best,buffalo,fitness]=get\_best\_buffalo(buffalo,newbuffalo,fitness)

% Evaluating all new solutions

for j=1:size(buffalo,1),

fnew=fobj(newbuffalo(j,:));

if fnew<=fitness(j),

fitness(j)=fnew;

buffalo(j,:)=newbuffalo(j,:);

end

end

% Find the current best

[fmin,K]=min(fitness) ;

best=buffalo(K,:);

% Application of simple constraints

function s=simplebounds(s,Lb,Ub)

% Apply the lower bound

ns\_tmp=s;

I=ns\_tmp<Lb;

ns\_tmp(I)=Lb(I);

% Apply the upper bounds

J=ns\_tmp>Ub;

ns\_tmp(J)=Ub(J);

% Update this new move

s=ns\_tmp;

%% You can replace the following by your own functions

% A d-dimensional objective function

function f=fobj(x)

%% d-dimensional sphere function sum\_j=1^d (u\_j-1)^2.

% with a minimum at (1,1, ...., 1);

%% x^3 + 60x^2 + 900x + 100

%z=u^2;

%z=u(1)^2+u(2);

%z=u(1)^3+ (60\*u(1)^2) + (900\*u(1)) + 100;

%z=(-u(1)+9)^2;

%f = x(1)^3 - 60\*x(1)^2 + 900\*x(1) + 100;

%f = x(1)^2 -3\*x(1) - 10;

%z=(1-u(1))^2+100\*(u(2)-u(1)^2)^2;

%z=sum((u-1).^2);

%z=sin(u);

%f = (1-x)^2 + 100\*(x(1)-x^2)^2; %ROSENBROCK FUNCTION

%f=x(1)^+2\*x(2)+x(2);

%f=x(1)^2+x(2)^2+x(3)^2+x(4)^2+x(5)^2+x(6)^2;

%f = x(1)^3 - 9\*x(1);

f=sum((x-1).^2); %SPHERE FUNCTION

% a = 1;

% b = 5.1/(4\*3.14^2);

% c = 5/pi;

% r = 6;

% s = 10 ;

% t = 1/(8\*pi);

%

% term1 = a \* (x(2) - b\*x(1)^2 + c\*x(1) - r)^2;

% term2 = s\*(1-t)\*cos(x(1));

% f = term1 +term2 +s;

%

%

% f= (x(2)-(5.1/(4\*pi^2))\*x(1)^2+5\*x(1)/pi-6)^2+10\*(1-1/(8\*pi))\*cos(x(1))+10;

%

%

% frac1 = 1 + cos(12\*sqrt(x(1)^2+x(2)^2));

% frac2 = 0.5\*(x(1)^2+x(2)^2) + 2;

%

% f = -frac1/frac2;

%

%

%

% f= (1.5-x(1)\*(1-x(2)))^2+(2.25-x(1)\*(1-x(2)^2))^2+(2.625-x(1)\*(1-x(2)^3))^2

%f= (100\*(x(2)-x(1)^2)^2 + (x(1)-1)^2) + (100\*(x(3)-x(2)^2)^2 + (x(2)-1)^2);

% n = 25;

% s1 = 0;

% for j = 2:n;

% s1 = s1+j\*(2\*x(j)^2-x(j-1))^2;

% end

% y = s1+(x(1)-1)^2;