function [grandmin,elapsed\_time]=LSA\_Main(F\_index)

tic;%start stopwatch

% F\_index = 1; % Function 1-6 % Refer to other test function 7-24 in [1].

[low,up,dim]=test\_functions\_range(F\_index);

N = 50; % population size/number of agents

D = dim; % number of dimension

T = 500; % maximum number of iteration

% Set upper bound & lower bound

if size(up,2)==1

UB = ones(1,D).\*up;

LB = ones(1,D).\*low;

end

if size(up,2)>1

UB = up;

LB = low;

end

%randomly create 1st population

for d = 1:D

Dpoint(:,d)= rand(N,1)\*(UB(d)-LB(d))+LB(d);

end

ch\_time = 0; % reset

max\_ch\_time = 10;

fit\_old = 10^10\*(ones(1,N));

direct = sign(unifrnd(-1,1,1,dim));

for t = 1:T

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Evaluate performance

fit = evaluateF(Dpoint,F\_index);

Ec = fit;

% update channel

ch\_time = ch\_time+1;

if ch\_time >= max\_ch\_time

[Ms ds]=sort(Ec,'ascend');

Dpoint(ds(N),:) = Dpoint(ds(1),:); % Eliminate the worst channel

Ec(ds(N)) = Ec(ds(1)); % Update

ch\_time = 0; % reset

end

% Rangking the fitness value

[Ms ds]=sort(Ec,'ascend');

best = Ec(ds(1));

worst = Ec(ds(N));

Energy = 2.05 - 2\*exp(-5\*(T-t)/T);% Update energy

% update direction

for d = 1:D

Dpoint\_test = Dpoint(ds(1),:);

Dpoint\_test(d) = Dpoint\_test(d)+direct(d)\*0.005\*(UB(d)-LB(d));

fv\_test = evaluateF(Dpoint\_test,F\_index);

if fv\_test < best % If better, +ve direction

direct(d) = direct(d);

else

direct(d) = -1\*direct(d);

end

end

% update position

for i = 1:N

dist=Dpoint(i,:)-Dpoint(ds(1),:);

for d = 1:D

if Dpoint(i,:)==Dpoint(ds(1),:)

Dpoint\_temp(d) = Dpoint(i,d)+direct(d)\*abs(normrnd(0,Energy));

else

if dist(d)<0

Dpoint\_temp(d) = Dpoint(i,d)+exprnd(abs(dist(d)));

else

Dpoint\_temp(d) = Dpoint(i,d)-exprnd(dist(d));

end

end

if (Dpoint\_temp(d)>UB(d))||(Dpoint\_temp(d)<LB(d))

Dpoint\_temp(d) = rand(1)\*(UB(d)-LB(d))+LB(d); % Re-initialized

end

end

fv = evaluateF(Dpoint\_temp,F\_index);

if fv < Ec(i)

Dpoint(i,:) = Dpoint\_temp;

Ec(i) = fv;

% Focking procedure

if rand < 0.01

for d = 1:D

Dpoint\_fock(d) = UB(d)+LB(d)-Dpoint\_temp(d);% Focking

end

fock\_fit = evaluateF(Dpoint\_fock,F\_index); % Evaluate

if fock\_fit < Ec(i)

Dpoint(i,:) = Dpoint\_fock; % Replace the channel

Ec(i) = fock\_fit;

end

end

end

end

if best == worst

break

end

% record the performance

fitness(t) = min(fit);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

end

elapsed\_time=toc; % time taken for this algorithm

% select the optimal value

grandmin = min(fitness);

y = fitness;

x = 1:length(y);

plot(x,y,'Linewidth',2)

xlabel('No of Iteration');

ylabel('Fitness Value');

axis([0 max(x) min(y) max(y)]);

fprintf(1,'Optimal value = %f\n', grandmin);