**Appendix B**

**PSO** Code in **MATLAB**

clc;

clear;

global NFE;

NFE=0;

ObjFun=@(x) MyFun(x); % Objective Function Nested M-file

nVar=0000; % Number of Decision Variables

VarSize=[1 nVar]; % Size of Decision Variables Matrix

VarMin=[0000 0000]; % Lower Bound of Variables

VarMax=[0000 0000]; % Upper Bound of Variables

% PSO Parameters

MaxIt=0000; % Maximum Number of Iterations

nPop=0000; % Population Size (Swarm Size)

% Constriction Coefficients

phi1=2.05;

phi2=2.05;

phi=phi1+phi2;

chi=2/(phi-2+sqrt(phi^2-4\*phi));

w=chi; % Inertia Weight

wdamp=1; % Inertia Weight Damping Ratio

c1=chi\*phi1; % Personal Learning Coefficient

c2=chi\*phi2; % Global Learning Coefficient

% Velocity Limits

VelMax=0.1\*(VarMax-VarMin);

VelMin=-VelMax;

% Initialization

empty\_particle.Position=[];

empty\_particle.Fun=[];

empty\_particle.Sol=[];

empty\_particle.Velocity=[];

empty\_particle.Best.Position=[];

empty\_particle.Best.Fun=[];

empty\_particle.Best.Sol=[];

particle=repmat(empty\_particle,nPop,1);

GlobalBest.Fun=inf;

for i=1:nPop

% Initialize Position

particle(i).Position=unifrnd(VarMin,VarMax,VarSize);

% Initialize Velocity

particle(i).Velocity=zeros(VarSize);

% Evaluation

[particle(i).Fun particle(i).Sol]=ObjFun(particle(i).Position);

% Update Personal Best

particle(i).Best.Position=particle(i).Position;

particle(i).Best.Fun=particle(i).Fun;

particle(i).Best.Sol=particle(i).Sol;

% Update Global Best

if particle(i).Best.Fun<GlobalBest.Fun

GlobalBest=particle(i).Best;

end

end

BestFun=zeros(MaxIt,1);

nfe=zeros(MaxIt,1);

% PSO Main Loop

for it=1:MaxIt

for i=1:nPop

% Update Velocity

particle(i).Velocity = w\*particle(i).Velocity ...

+c1\*rand(VarSize).\*(particle(i).Best.Position-particle(i).Position)..

+c2\*rand(VarSize).\*(GlobalBest.Position-particle(i).Position);

% Apply Velocity Limits

particle(i).Velocity = max(particle(i).Velocity,VelMin);

particle(i).Velocity = min(particle(i).Velocity,VelMax);

% Update Position

particle(i).Position = particle(i).Position + particle(i).Velocity;

% Velocity Mirror Effect

IsOutside=(particle(i).Position<VarMin | particle(i).Position>VarMax);

particle(i).Velocity(IsOutside)=-particle(i).Velocity(IsOutside);

% Apply Position Limits

particle(i).Position = max(particle(i).Position,VarMin);

particle(i).Position = min(particle(i).Position,VarMax);

% Evaluation

[particle(i).Fun particle(i).Sol] = ObjFun(particle(i).Position);

% Update Personal Best

if particle(i).Fun<particle(i).Best.Fun

particle(i).Best.Position=particle(i).Position;

particle(i).Best.Fun=particle(i).Fun;

particle(i).Best.Sol=particle(i).Sol;

% Update Global Best

if particle(i).Best.Fun<GlobalBest.Fun

GlobalBest=particle(i).Best;

end

end

end

BestFun(it)=GlobalBest.Fun;

nfe(it)=NFE;

w=w\*wdamp;

end

% Results

figure;

plot(nfe,BestFun,'LineWidth',2);

xlabel('NFE');

ylabel('Best Fun');

% End