Apéndice I

IMPLEMENTACIÓN EN MATLAB DEL ALGORITMO PSO PROPUESTO

AI.1 IMPLEMENTACIÓN PROGRAMA PRINCIPAL

En este apéndice se implementa el algoritmo PSO propuesto en el capítulo 7 para la resolución del problema. El programa utilizado para su implementación es MATLAB. El programa principal queda de la siguiente manera:

```
%=============================================================
% ALGORITMO PSO BINARIO
% AUTOR: Manuel Gómez González
% FECHA: Enero de 2008
%=============================================================
clc;
clear all; close all;
%=============================================================
disp('EMPLAZAMIENTO DE CENTRAL DE PROCESADO DE BIOMASA MEDIANTE PSO
BINARIO');
%=============================================================
%Parametros de ajuste del PSO
inicodif;  %Se definen los parámetros bitsx, bitsy, bitstam
D=bitsi+bitsj+bitstam;  %Dimension del problema
Tam_Swarm=30; %Tamaño de la nube de particulas
Nitmax=60; %Número de iteraciones del algoritmo
pc_max=0.5; %Probabilidad inercial máxima o inicial
pc_min=0.001; %Probabilidad inercial mínima o final

fprintf('Numero máximo de iteraciones Nitmax: %d\n',Nitmax);
fprintf('Dimension del problema D: %d\n',D);
fprintf('Tamaño de la nube de particula Tam_Swarm: %d\n',Tam_Swarm);
fprintf('Maxima probabilidad de cambio pc_max: %f\n',pc_max);
fprintf('Minima probabilidad de cambio pc_min: %f\n',pc_min);
```

```
%=====================================================================
T=LeerTabla('REGION'); %se carga la base de datos de las parcelas
Numpruebas=10; %se definen el número de ejecuciones del algoritmo
PRUEBAS=[];
EVOLUCION=zeros(1,Nitmax+1);
%=====================================================================
%Se dibujan las gráficas características de la región
DibujarGeografia(T);
DibujarDensidad(T);
DibujarPotencial(T);
DibujarUtilizable(T);
DibujarCosteExtraccion(T);
DibujarTipoResiduos(T);
%=====================================================================
for hh=1:Numpruebas

X=(rand(Tam_Swarm,D)>0.5); %Generación aleatoria de las partículas
V=(rand(Tam_Swarm,D)>0.5); %Generación aleatoria de las velocidades

fprintf('Evaluacion inicial\n');
part=X;

Fpbest=Evaluacion(X,T); %se realiza la evaluación inicial
%La función Evaluacion() se encarga de obtener el valor de la aptitud
de todas las partículas de la nube

%se obtiene la partícula con mejor aptitud de la nube inicial
indice_gbest=find(max(Fpbest)==Fpbest);
indice_gbest=indice_gbest(1);
gbest=X(indice_gbest,:); %mejor particula global inicial

%Resultados evaluación inicial
[f_gbest,c_gbest,tam_gbest]=Codificacion1bin2dec(gbest);
fprintf('La solución tiene la central situada
en(%d,%d)\n',f_gbest,c_gbest);
fprintf('La solución es una region cuadrada de %f por %f
parcelas\n',2*tam_gbest+1,2*tam_gbest+1);
```

```
Fgbest=Fpbest(indice_gbest);
fprintf('El indice de rentabilidad que se tiene es %f\n',Fgbest);

Evolucion_Fgbest=Fgbest;

%CUERPO PRINCIPAL DEL ALGORITMO PSO

for I=1:Nitmax

    fprintf('Prueba: %d Iteración: %d\n',hh,I);

    d1=xor(pbest,X);
    d2=xor(repmat(gbest,Tam_Swarm,1),X);

    c1=(rand(Tam_Swarm,D)>0.5);
    c2=(rand(Tam_Swarm,D)>0.5);

    lambda_pc=log(pc_max/pc_min)/(Nitmax-1);  %decaimiento exponencial
    pc=pc_max*exp(-lambda_pc*(I-1));%cálculo de probabilidad inercial

    W_n=(rand(Tam_Swarm,D)<pc); %factor de inercia negado
    W=not(W_n);
    Vant=V;
    V=W_n|and(W,(and(c1,d1)|and(c2,d2)));  %cálculo vector velocidad

    X=xor(X,V); cálculo nuevas partículas

    %se obtiene la partícula con mejor aptitud de la nube inicial
    F=Evaluacion(X,T);
    indice_nuevos_pbest=find(F>Fpbest);

    for K=indice_nuevos_pbest
        pbest(K,:)=X(K,:); %mejores partículas
        Fpbest(K)=F(K); %mejores aptitudes de cada partícula
    end;
```

```
    %se obtiene la partícula con mejor aptitud de la nube
    if (max(Fpbest)>Fgbest)
        Fgbest=max(Fpbest);
        indice_gbest=find(max(Fpbest)==Fpbest);
        indice_gbest=indice_gbest(1);
        gbest=X(indice_gbest,:);
    end;

%se convierten los valores de binario a real
  [f_gbest,c_gbest,tam_gbest]=Codificacion1bin2dec(gbest);

%se calculan y presentan los valores obtenidos en esta iteración
fprintf('La probabilidad de cambio (factor de inercia) es %f\n', pc);
fprintf('La solución tiene la central situada en
(%d,%d)\n',f_gbest,c_gbest);
fprintf('La solución es una region cuadrada de %f por %f
parcelas\n',2*tam_gbest+1,2*tam_gbest+1);
fprintf('El indice de rentabilidad que se tiene es %f\n',Fgbest);
[RegionSolucion]=ObtenerVecinos(T,f_gbest,c_gbest,tam_gbest);
Pe_solucion=PotenciaProd(RegionSolucion);
fprintf('La solución produce %f MW\n',Pe_solucion);
  Area_solucion=ObtenerArea(RegionSolucion);
  fprintf('La solución toma Biomasa de %f Km2\n',Area_solucion);
  fprintf('El valor actual neto que se tiene es
%f\n',ValorActualNeto(RegionSolucion));
end;

Evolucion_Fgbest=[Evolucion_Fgbest, Fgbest];

end;

PRUEBAS=[PRUEBAS,Fgbest];
EVOLUCION=EVOLUCION+Evolucion_Fgbest;
```

```
%============================================================
%Ahora sacamos gráficas acerca de los resultados obtenidos
%Solucion obtenida
DibujarSolucion(T,f_gbest,c_gbest,tam_gbest);
%DibujarSolucion representa las parcelas seleccionadas en el mapa
end;

fprintf('FIN DEL ALGORITMO %f\n');
EVOLUCION=EVOLUCION/Numpruebas;
figure;
plot(0:Nitmax,EVOLUCION); %Evolución de la función objetivo
figure;
bar(PRUEBAS); %Estadísticas de las ejecuciones realizadas
```