The main code of BPSO algorithm used to solve the UC problem :

1. BPSOmain.m is the main script file, in the beginning, import the data of 10 unit benchmark, defined the ITERmax times, population size and some matrix.

```
%%%%%%%%%%%%%%%%%%10 unit benchmark %%%%%%%%%%%%%%%%%%%%%%
        UnitNum=10;
        G=10;
        InitialTON=[8,8,0,0,0,0,0,0,0,0];
        InitialTOFF=[0,0,5,5,6,3,3,1,1,1];
        MDT=[8,8,5,5,6,3,3,1,1,1];
        MUT=[8,8,5,5,6,3,3,1,1,1];
        x0=[1,1,0,0,0,0,0,0,0,0];
        PMAX=[455,455,130,130,162,80,85,55,55,55];
        PMIN=[150,150,20,20,25,20,25,10,10,10];
        PD=[700,750,850,950,1000,1100,1150,1200,1300,1400,1450,1500,1400,1300,1200,105
        0,1000,1100,1200,1400,1300,1100,900,800];
        SR=0.1*PD;
        a=[1000,970,700,680,450,370,480,660,665,670];
        b=[16.19,17.26,16.6,16.5,19.7,22.26,27.74,25.92,27.27,27.79];
        c=[0.00048,0.00031,0.002,0.00211,0.00398,0.00712,0.00079,0.00413,0.00222,0.00173]
        ;
        SUH=[4500,5000,550,560,900,170,260,30,30,30];
        SUC=[9000,10000,1100,1120,1800,340,520,60,60,60];
        Tcold=[5,5,4,4,4,2,2,0,0,0];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

2. Then initialize the swarm use the UnitStatusInitial.m script, applied the CheckUCConstraints.m to check the UC constraints.

```
%%%%%%%%%%%%%%%%%%initialize%%%%%%%%%%%%%%%%%%%%%%
  swarml=UnitStatusInitial(G,T);
  [swarm(n,:),swarmTOFF(:,:,n)]=CheckUCConstraints(swarml,G,T,x0,MUT,MDT,InitialTO
  N,InitialTOFF,PMAX,PD,SR,h);%% check the constraints
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

3. Choose the binary particle swarm optimizer(PSO) algorithm to optimize the population. In the last, calculate the bestfitness value by the lambda iteration which shown in ‘LanEcoDisp.m’ and ‘ObjectFitness.m’ file.

```
%%%%%%%%%%%%%%%%%%BPSO algorithm%%%%%%%%%%%%%%%%%%%%%%
disp('                              BPSO                              ')
```

```
                tic;
                %**************%**************%**************%**************%**********
*********%**************
                c1=2;
                c2=2;%Social parameter
                w=0.6;%weight
                Vmax=4.*ones(NP,Dimension);
                Vmin=-Vmax;
%                v=4.*(2.*rand(NP,Dimension)-ones(NP,Dimension));%Á£×ÓµÄ³õÊ¼ËÙ¶È
                v=Vmin+(Vmax-Vmin).*rand(NP,Dimension);
                Pr=zeros(NP,Dimension);
               fitness2=fitness1(:,iter2);
          for iter1=1:ITERmax
                    R1=rand(NP,Dimension);
                    R2=rand(NP,Dimension);
          %             w=Wmax
                    v=v*w+c1*R1.*(Pmbest-swarm)+c2*R2.*(repmat(Gmbest,NP,1)-swarm);%
                    v=Checkbound(v,Vmin,Vmax,NP,Dimension,1);
                    for i=1:NP
                        for j=1:Dimension
%                             Pr(i,j)=(1+exp(-v(i,j)))^(-1);
                            Pr(i,j)=2*abs((1+exp(-v(i,j)))^(-1)-0.5);

                            if rand<Pr(i,j)
                                Swarm1(i,j)=1;
                            else
                                Swarm1(i,j)=0;
                            end
                        end

[Swarm1(i,:),swarmTOFF(:,:,i)]=CheckUCConstraints(Swarm1(i,:),G,T,x0,MUT,MDT,InitialTO
N,InitialTOFF,PMAX,PD,SR,h);

[fitness_new(i),Pt_all]=ObjectFitness(Swarm1(i,:),G,T,x0,MDT,PMAX,PMIN,PD,a,b,c,SUH,SU
C,Tcold,swarmTOFF(:,:,i));
                        if fitness_new(i)<fitness2(i)
                         else
                            fitness_new(i)=fitness2(i);
                            Pmbest(i,:)=Swarm1(i,:);%
                        end

                            if fitness_new(i)<Gfitness1(iter2)
                            Gmbest=Swarm1(i,:);
                            Gfitness1(iter2)=fitness_new(i);
```

```
                    end
            end
            ge_all(1,iter1)=Gfitness1(iter2);
       end
        tim1=toc;
%                   fprintf('The cost time of BPSO %15.5f.\n',tim1);
    ge_all2(k,1:ITERmax)=ge_all(1,1:ITERmax);
    ge_all2(k,K)=tim1;
    GmPEVbest=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];
[Gmbest,swarmTOFFgbest]=CheckUCConstraints(Gmbest,G,T,x0,MUT,MDT,InitialTON,InitialTOFF,PMAX,PD,SR,h);
[Pt,Fobject]=SQPeld(Gmbest,G,T,x0,MDT,PMAX,PMIN,PD,a,b,c,SUH,SUC,Tcold,swarmTOFFgbest,GmPEVbest);
   PtBest_all(:,:,k)=Pt;
   bestfitness(1,k)=Fobject;
   bestfitness(2,k)=tim1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

4. Get the total economic cost and generation output by the ‘SQPeld.m’, and the experimental result saved in the CECBPSO.mat.

```
%%%%%%%%%%%%%%    save the result    %%%%%%%%%%%%%%%%%%%%%%%
  Filename='CECBPSO.mat';
  save(Filename)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

The value of bestfitness and PtBest_all in the 'CECBPSO.mat' represents the total economic cost and power output of unit during the 24 hours respectively.