Codes in MATLAB for Particle Swarm Optimization

Code · March 2016

DOI: 10.13140/RG.2.1.1078.7608

Mahamad Nabab Alam
Indian Institute of Technology Roorkee

Codes in MATLAB for Particle Swarm Optimization

Mahamad Nabab Alam, Research Scholar

Particle swarm optimization (PSO) codes in MATLAB suitable for solving constrained optimization problem

Save the following codes in MATLAB script file (*.m) and save as ofun.m.

---start

```
function f=ofun(x)

% objective function (minimization)
of=10*(x(1)-1)^2+20*(x(2)-2)^2+30*(x(3)-3)^2;

% constraints (all constraints must be converted into <=0 type)
% if there is no constraints then comments all c0 lines below

c0=[];
c0(1)=x(1)+x(2)+x(3)-5;      % <=0 type constraints
c0(2)=x(1)^2+2*x(2)-x(3);    % <=0 type constraints

% defining penalty for each constraint
for i=1:length(c0)
    if c0(i)>0
        c(i)=1;
    else
        c(i)=0;
    end
end
penalty=10000;            % penalty on each constraint violation
f=of+penalty*sum(c);      % fitness function
```

---end

Save the following main program codes in MATLAB script file (*.m) as run_pso.m (any name can be used) and run.

---start

```
tic
clc
clear all
close all
rng default

LB=[0 0 0];          %lower bounds of variables
UB=[10 10 10];       %upper bounds of variables

% pso parameters values
m=3;              % number of variables
n=100;            % population size
wmax=0.9;         % inertia weight
wmin=0.4;         % inertia weight
c1=2;             % acceleration factor
c2=2;             % acceleration factor

% pso main program----------------------------------------------------start
maxite=1000;     % set maximum number of iteration
```

```
maxrun=10;       % set maximum number of runs need to be
for run=1:maxrun
    run
    % pso initialization----------------------------------------------start
    for i=1:n
        for j=1:m
            x0(i,j)=round(LB(j)+rand()*(UB(j)-LB(j)));
        end
    end
    x=x0;        % initial population
    v=0.1*x0;    % initial velocity
    for i=1:n
        f0(i,1)=ofun(x0(i,:));
    end
    [fmin0,index0]=min(f0);
    pbest=x0;                % initial pbest
    gbest=x0(index0,:);      % initial gbest
    % pso initialization------------------------------------------------end

    % pso algorithm---------------------------------------------------start
    ite=1;
    tolerance=1;
    while ite<=maxite && tolerance>10^-12

        w=wmax-(wmax-wmin)*ite/maxite; % update inertial weight

        % pso velocity updates
        for i=1:n
            for j=1:m
                v(i,j)=w*v(i,j)+c1*rand()*(pbest(i,j)-x(i,j))...
                        +c2*rand()*(gbest(1,j)-x(i,j));
            end
        end

        % pso position update
        for i=1:n
            for j=1:m
                x(i,j)=x(i,j)+v(i,j);
            end
        end

        % handling boundary violations
        for i=1:n
            for j=1:m
                if x(i,j)<LB(j)
                    x(i,j)=LB(j);
                elseif x(i,j)>UB(j)
                    x(i,j)=UB(j);
                end
            end
        end

        % evaluating fitness
        for i=1:n
            f(i,1)=ofun(x(i,:));
        end

        % updating pbest and fitness
        for i=1:n
            if f(i,1)<f0(i,1)
```

```
                pbest(i,:)=x(i,:);
                f0(i,1)=f(i,1);
            end
        end

        [fmin,index]=min(f0);    % finding out the best particle
        ffmin(ite,run)=fmin;     % storing best fitness
        ffite(run)=ite;          % storing iteration count

        % updating gbest and best fitness
        if fmin<fmin0
            gbest=pbest(index,:);
            fmin0=fmin;
        end

        % calculating tolerance
        if ite>100;
            tolerance=abs(ffmin(ite-100,run)-fmin0);
        end

        % displaying iterative results
        if ite==1
            disp(sprintf('Iteration    Best particle    Objective fun'));
        end
        disp(sprintf('%8g  %8g          %8.4f',ite,index,fmin0));
        ite=ite+1;
    end
    % pso algorithm-----------------------------------------------------end
    gbest;
    fvalue=10*(gbest(1)-1)^2+20*(gbest(2)-2)^2+30*(gbest(3)-3)^2;
    fff(run)=fvalue;
    rgbest(run,:)=gbest;
    disp(sprintf('---------------------------------------'));
end
% pso main program------------------------------------------------------end
disp(sprintf('\n'));
disp(sprintf('*********************************************************'));
disp(sprintf('Final Results-----------------------------'));
[bestfun,bestrun]=min(fff)
best_variables=rgbest(bestrun,:)
disp(sprintf('*********************************************************'));
toc

% PSO convergence characteristic
plot(ffmin(1:ffite(bestrun),bestrun),'-k');
xlabel('Iteration');
ylabel('Fitness function value');
title('PSO convergence characteristic')
%#############################################################################
```

--end

Enjoy with PSO;

****