学号 专业 姓名

理论课任务5

【任务名称】 进化算法

【任务目的】

分别利用遗传算法(种群大小100、迭代100次)和梯度下降法(学习率0.1、迭代100次)求以下函数的最小值，记录两个算法每代得到的最小函数值

【源码】

主函数：

clc;clear;close all;

f = @(x) sum((x - 0.5).^2 + sin((8 \* x - 4.5) \* pi) + 1);

pf = @(x) 2 \* (x - 0.5) - 8 \* pi \* cos((8 \* x - 4.5) \* pi);

fitFun = @(x) sum((x - 0.5).^2 + sin((8 \* x - 4.5) \* pi) + 1, 2);

Plot\_GA\_AG(f , pf, fitFun,10^-1,10^3)

figure;

f = @(x) sum((x - 0.5).^2);

pf = @(x) 2 \* (x - 0.5);

fitFun = @(x) sum((x-0.5).^2, 2);

Plot\_GA\_AG(f , pf, fitFun,10^-20,10^0)

绘图函数:

function Plot\_GA\_AG(f , pf, fitFun,y0,y1)

% GA

N = 100;

G = 100;

Lower = zeros(1, 10);

Upper = ones(1, 10);

Pdec = unifrnd(repmat(Lower, N, 1), repmat(Upper, N ,1));

Pobj = fitFun(Pdec);

bestValues = zeros(G, 1);

for gen = 1 : G

Pdecl = MatingSelection(Pdec, Pobj);

Odec = Crossover(Pdecl);

Odec = Mutation(Odec, Lower, Upper);

[Pdec, Pobj] = EnvironmentalSelection([Pdec; Odec], [Pobj; fitFun(Odec)]);

bestValues(gen) = min(Pobj);

end

semilogy(1:G, bestValues,'r');

hold on

% GD

x0 = rand(10, 1);

learning\_rate = 0.1;

iterations = 100;

min\_values = zeros(iterations, 1);

x = x0;

for i = 1:iterations

gradient = pf(x);

x = x - learning\_rate \* gradient;

min\_values(i) = f(x);

end

semilogy(1:iterations,min\_values,'b');

xlabel('Generation');

ylabel('Minimum Function Value');

ylim([y0,y1])

legend('Genetic algorithm', 'Gradient algorithm');

end

二进制交叉

function Odec = Crossover(Pdecl)

N = size(Pdecl, 1);

D = size(Pdecl, 2);

Odec = zeros(size(Pdecl));

for i = 1:2:N

x1 = Pdecl(i, :);

x2 = Pdecl(i+1, :);

beta = zeros(1, D);

u = rand(1, D);

eta = 20;

for j = 1:D

if u(j) <= 0.5

beta(j) = (2\*u(j))^(1/(eta+1));

else

beta(j) = (2\*(1-u(j)))^(-1/(eta+1));

end

end

c1 = 0.5 \* ((1 + beta) .\* x1 + (1 - beta) .\* x2);

c2 = 0.5 \* ((1 - beta) .\* x1 + (1 + beta) .\* x2);

Odec(i, :) = c1;

Odec(i+1, :) = c2;

end

end

环境选择

function [Pdec, Pobj] = EnvironmentalSelection(P, Pobj)

N = size(Pobj, 1) / 2;

[~, sortedIdx] = sort(Pobj);

selectedIdx = sortedIdx(1:N);

Pdec = P(selectedIdx, :);

Pobj = Pobj(selectedIdx, :);

end

交配池选择

function Pdecl = MatingSelection(Pdec, Pobj)

N = size(Pdec, 1);

Pdecl = zeros(size(Pdec));

for i = 1:N

idx1 = randi(N);

idx2 = randi(N);

if Pobj(idx1) < Pobj(idx2)

Pdecl(i, :) = Pdec(idx1, :);

else

Pdecl(i, :) = Pdec(idx2, :);

end

end

end

多项式变异

function Odec = Mutation(Odec, Lower, Upper)

N = size(Odec, 1);

D = size(Odec, 2);

eta = 20;

for i = 1:N

for j = 1:D

if rand() < 1/D

u\_i = rand();

c\_i = Odec(i, j);

low\_i = Lower(j);

up\_i = Upper(j);

delta1 = (c\_i - low\_i) / (up\_i - low\_i);

delta2 = (up\_i - c\_i) / (up\_i - low\_i);

if u\_i <= 0.5

beta\_i = (2\*u\_i + (1 - 2 \* u\_i) \* (1 - delta1)^(eta+1))^(1/(eta+1)) - 1;

else

beta\_i = 1 - (2\*(1-u\_i) + 2 \* (u\_i -0.5) \* (1 - delta2)^(eta+1))^(1/(eta+1));

end

Odec(i, j) = c\_i + beta\_i \* (up\_i - low\_i);

Odec(i, j) = min(max(Odec(i, j), low\_i), up\_i);

end

end

end

end

【运行结果截图】

|  |  |
| --- | --- |
|  |  |

【任务总结】

思考：根据两组结果，总结遗传算法和梯度下降法各自的优势，并根据算法的特点分析优势产生的原因。

梯度下降法，通过计算函数的导数信息来计算最优解，其优势为：代码和原理简单，易于调试和实现，编写程序的过程可以体现出来，梯度下降法的代码很少且简洁易懂。且通过图一可以看出来，对于一些函数，在相同迭代次数下，梯度下降法接近最小值的速度远远快于遗传算法，这是由于对于梯度下降法根据函数的导数来进行最小值的逼近，由于函数1的导师简单且单调，即函数1中不存在多个极小值，梯度下降法就可以很快的迭代出最小值；但是在函数2中可以看出，梯度下降法并不能得到最小值，这是因为函数2的导数形式较为复杂，梯度下降法在迭代的过程中容易专注在某一个极小值而不会向其他方向迭代，这就有可能错过真正的最小值。

遗传算法：遗传算法通过模拟自然界的遗传和进化过程，实现全局搜索和并行优化。遗传算法的缺点在于：原理复杂，不利于编程实现，且逼近速度较慢。函数1中遗传算法虽然也在逐步逼近最小值，但是其速度明显慢于梯度下降法，这是因为遗传算法从问题的串集开始搜索，而非从单个解开始，这增加了搜索的覆盖面，有利于全局择优，遗传算法在计算时同时对搜索空间中的多个解进行适应度分析，这使得算法能够更全面地搜索解空间，提高找到全局最优解的可能性。但这势必就会导致其速度较慢，求解效率低。不过其准确率高，而且在处理复杂问题时，遗传算法不需要过多的数学要求和专业知识，降低了算法的使用门槛。

总之，遗传算法和梯度下降法各有优势，在实际使用过程中可以合理选择适合的算法。