

# 优化算法之粒子群算法（PSO）

lx青萍之末 2018-08-03 10:26:45 124282 收藏 882 版权

分类专栏： # 经典算法及分析 文章标签： 粒子群算法 PSO

## 粒子群算法的概念

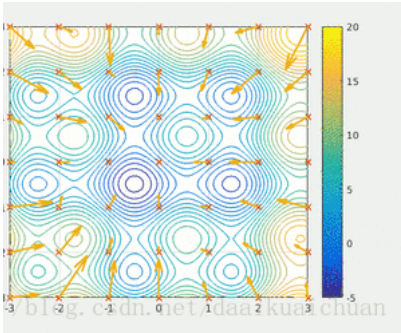
粒子群优化算法(PSO： Particle swarm optimization) 是一种进化计算技术（evolutionary putation）。源于对鸟群捕食的行为研究。粒子群优化算法的基本思想：是通过群体中个体的协作和信息共享来寻找最优解。

PSO的优势：在于简单容易实现并且没有许多参数的调节。目前已被广泛应用于函数优化、神经网络训练、模糊系统控制以及其他遗传算法的应用领域。

## 粒子群算法分析

### 基本思想

粒子群算法通过设计一种无质量的粒子来模拟鸟群中的鸟，粒子仅具有两个属性：**速度和位置**。速度代表移动的快慢，位置代表移动的方向。每个粒子在搜索空间中单独的搜寻最优解，并记为当前个体极值，并将个体极值与整个粒子群里的其他粒子共享，找到最优的那个个体极值为整个粒子群的当前全局最优解，粒子群中的所有粒子根据自己找到的当前个体极值和整个群共享的当前全局最优解来调整自己的速度和位置。下面的动图很形象地展示了PSO算法的



### 更新规则

PSO初始化为一群随机粒子(随机解)。然后通过迭代找到最优解。在每一次的迭代中，粒子跟踪两个“极值”(pbest，gbest)来更新自己。在找到这两个最优值后，粒子通过下面的公式来自己的速度和位置。

式 (1) :

$$v_i = v_i + c_1 \times rand() \times (pbest_i - x_i) + c_2 \times rand() \times (gbest_i - x_i)$$

式 (2) :

$$x_i = x_i + v_i$$

公式 (1) 、 (2) 中,  $i=1, 2, \dots, N$ ,  $N$  是此群中粒子的总数。

$v_i$  是粒子的速度

$rand()$ : 介于 (0, 1) 之间的随机数

$x_i$  粒子的当前位置

$c_1, c_2$ : 是学习因子, 通常  $c_1 = c_2 = 2$

$V_{max}$  最大值为  $V_{max}$  (大于0), 如果  $v_i$  大于  $V_{max}$ , 则  $v_i = V_{max}$

式 (1) 、 (2) 为 **PSO 的标准形式**。 [csdn.net/daaikuaichuan](https://blog.csdn.net/daaikuaichuan)

公式(1)的第一部分称为【记忆项】, 表示上次速度大小和方向的影响; 公式(1)的第二部分称为【自身认知项】, 是从当前点指向粒子自身最好点的一个矢量, 表示粒子的动作来源于自己经验; 公式(1)的第三部分称为【群体认知项】, 是一个从当前点指向种群最好点的矢量, 反映粒子间的协同合作和知识共享。粒子就是通过自己的经验和同伴中最好的经验来决定下一步的位置。以上面两个公式为基础, 形成了**PSO的标准形式**。

式 (3) :

$$v_i = \omega \times v_i + c_1 \times rand() \times (pbest_i - x_i) + c_2 \times rand() \times (gbest_i - x_i)$$

$\omega$  叫做惯性因子, 其值为非负。

$\omega$  其值较大, 全局寻优能力强, 局部寻优能力弱;

$\omega$  其值较小, 全局寻优能力弱, 局部寻优能力强。

动态  $\omega$  能获得比固定值更好的寻优结果。动态  $\omega$  可在 PSO 搜索过程中

动态变化, 也可以根据 PSO 性能的某个测度函数动态改变。

目前采用较多的是线性递减权值 (Linearly Decreasing Weight, LDW) 策略。

$$\omega = (\omega_{ini} - \omega_{end})(G_k - g) / G_k + \omega_{end}$$

$G_k$  最大迭代次数

$\omega_{ini}$  初始惯性权值

$\omega_{end}$ : 迭代至最大进化代数时的惯性权值

$\omega_{end}$  权值:

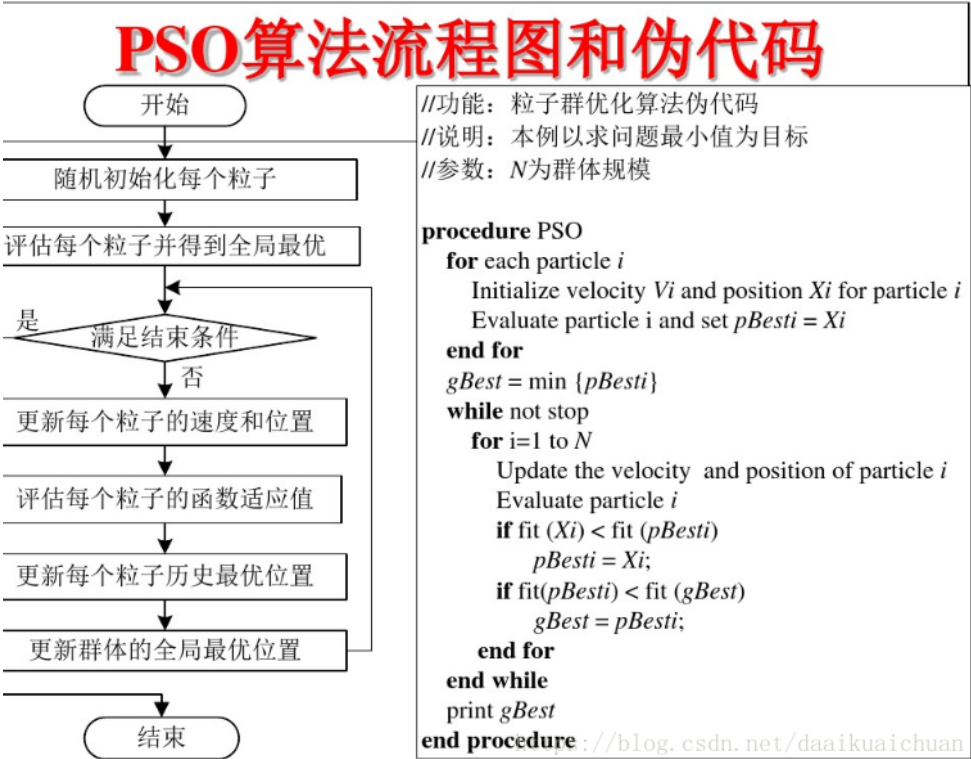
$$\omega_{ini}=0.9, \omega_{end}=0.4$$

引入, 使用 **PSO** 算法性能有了很大的提高, 针对不同的搜索问题,

通过调整全局和局部搜索能力, 也使 **PSO** 算法有成功地应用于很多实际问题。 [daaikuaichuan](https://blog.csdn.net/daaikuaichuan)

公式(2)和 公式(3)被视为**标准PSO算法**。

**PSO** 算法的流程和伪代码



PSO算法举例

## 6.1 已知函数 $y = f(x_1, x_2) = x_1^2 + x_2^2$

中  $-10 \leq x_1, x_2 \leq 10$  , 用粒子群优化算法求解y的最小值。

步骤1: 初始化。

设种群大小是 $N=3$ ; 在搜索空间中随机初始化每个解的速度和位置, 计算适应函数, 并且得到粒子的历史最优位置和群的全局最优位置。

$$\begin{aligned} & \begin{cases} v_1 = (3, 2) \\ x_1 = (8, -5) \end{cases} \begin{cases} f_1 = 8^2 + (-5)^2 = 64 + 25 = 89 \\ pBest_1 = x_1 = (8, -5) \end{cases} \\ & \begin{cases} v_2 = (-3, -2) \\ x_2 = (-5, 9) \end{cases} \begin{cases} f_2 = (-5)^2 + 9^2 = 25 + 81 = 106 \\ pBest_2 = x_2 = (-5, 9) \end{cases} \\ & \begin{cases} v_3 = (5, 3) \\ x_3 = (-7, -8) \end{cases} \begin{cases} f_3 = (-7)^2 + (-8)^2 = 49 + 64 = 113 \\ pBest_3 = x_3 = (-7, -8) \end{cases} \\ & gBest = pBest_1 = (8, -5) \end{aligned}$$

步骤3: 评估粒子的适应度函数值。

新粒子的历史最优位置和全局的最优位置。

$$= 9.5^2 + (-4)^2 = 90.25 + 16 = 106.25 > f_1 = 89$$

$$= 89$$

$$Best_1 = (8, -5)$$

$$= 1.1^2 + 10^2 = 1.21 + 100 = 101.21 < 106 = f_2$$

$$= f_2 = 101.21$$

$$Best_2 = x_2 = (1.1, 10)$$

$$= (-3.5)^2 + (-1.7)^2 = 12.25 + 2.89 = 15.14 < 113 = f_3$$

$$= f_3 = 15.14$$

$$Best_3 = x_3 = (-3.5, -1.7)$$

$$gBest = pBest_3 = (-3.5, -1.7)$$

步骤2: 粒子的速度和位置更新。

根据自身的历史最优位置和全局的最优位置, 更新每个粒子的速度和位置。

$$\begin{aligned} p_1 = \begin{cases} v_1 = \omega \times v_1 + c_1 \times r_1 \times (pBest_1 - x_1) + c_2 \times r_2 \times (gBest - x_1) \\ \Rightarrow v_1 = \begin{cases} 0.5 \times 3 + 0 + 0 = 1.5 \\ 0.5 \times 2 + 0 + 0 = 1 \end{cases} = (1.5, 1) \\ x_1 = x_1 + v_1 = (8, -5) + (1.5, 1) = (9.5, -4) \end{cases} \end{aligned}$$

$$\begin{aligned} p_2 = \begin{cases} v_2 = \omega \times v_2 + c_1 \times r_1 \times (pBest_2 - x_2) + c_2 \times r_2 \times (gBest - x_2) \\ \Rightarrow v_2 = \begin{cases} 0.5 \times (-3) + 0 + 2 \times 0.3 \times (8 - (-5)) = 6.1 \\ 0.5 \times (-2) + 0 + 2 \times 0.1 \times ((-5) - 9) = 1.8 \end{cases} = (6.1, 1.8) \\ x_2 = x_2 + v_2 = (-5, 9) + (6.1, 1.8) = (1.1, 10.8) = (1.1, 10) \end{cases} \end{aligned}$$



注意! 对于越界的位置, 需要进行合法性调整

$$\begin{aligned} p_3 = \begin{cases} v_3 = \omega \times v_3 + c_1 \times r_1 \times (pBest_3 - x_3) + c_2 \times r_2 \times (gBest - x_3) \\ \Rightarrow v_3 = \begin{cases} 0.5 \times 5 + 0 + 2 \times 0.05 \times (8 - (-7)) = 3.5 \\ 0.5 \times 3 + 0 + 2 \times 0.8 \times ((-5) - (-8)) = 6.3 \end{cases} = (3.5, 6.3) \\ x_3 = x_3 + v_3 = (-7, -8) + (3.5, 6.3) = (-3.5, -1.7) \end{cases} \end{aligned}$$

$w$ 是惯性权重, 一般取 $[0, 1]$ 区间的数, 这里假设为 $0.5$

$c_1$ 和 $c_2$ 为加速系数, 通常取固定值 $2.0$

$r_1$ 和 $r_2$ 是 $[0, 1]$ 区间的随机数

步骤4: 如果满足结束条件, 则输出全局最优结果并结束程序, 否则, 转向步骤2继续执行。

### PSO算法的demo

```
1 #include <iostream>
2 #include <vector>
3 #include <cmath>
4 #include <map>
5 #include <algorithm>
6 #include <random>
7 #include <ctime>
8 #include <Eigen/Dense>
9 using namespace Eigen;
10 using namespace std;
11
12 const int dim = 1; // 维数
13 const int p_num = 10; // 粒子数量
14 const int iters = 100; // 迭代次数
15 const int inf = 999999; // 极大值
16 const double pi = 3.1415;
17 // 定义粒子的位置和速度的范围
18 const double v_max = 4;
19 const double v_min = -2;
20 const double pos_max = 2;
21 const double pos_min = -1;
22 // 定义位置向量和速度向量
23 vector<double> pos;
24 vector<double> spd;
```

```
5 //定义粒子的历史最优位置和全局最优位置
6 vector<double> p_best;
7 double g_best;
8 //使用eigen库定义函数值矩阵和位置矩阵
9 Matrix<double, iters, p_num> f_test;
10 Matrix<double, iters, p_num> pos_mat;
11
12 //定义适应度函数
13 double fun_test(double x)
14 {
15     double res = x * x + 1;
16     return res;
17 }
18
19 //初始化粒子群的位置和速度
20 void init()
21 {
22     //矩阵中所有元素初始化为极大值
23     f_test.fill(1e10);
24     pos_mat.fill(1e10);
25     //生成范围随机数
26     static std::mt19937 rng;
27     static std::uniform_real_distribution<double> distribution1(-1, 2);
28     static std::uniform_real_distribution<double> distribution2(-2, 4);
29     for (int i = 0; i < p_num; ++i)
30     {
31         pos.push_back(distribution1(rng));
32         spd.push_back(distribution2(rng));
33     }
34     vector<double> vec;
35     for (int i = 0; i < p_num; ++i)
36     {
37         auto temp = fun_test(pos[i]); //计算函数值
38         //初始化函数值矩阵和位置矩阵
39         f_test(0, i) = temp;
40         pos_mat(0, i) = pos[i];
41         p_best.push_back(pos[i]); //初始化粒子的历史最优位置
42     }
43     std::ptrdiff_t minRow, minCol;
44     f_test.row(0).minCoeff(&minRow, &minCol); //返回函数值矩阵第一行中极小值对应的位置
45     g_best = pos_mat(minRow, minCol); //初始化全局最优位置
46 }
47
48 void PSO()
49 {
50     static std::mt19937 rng;
51     static std::uniform_real_distribution<double> distribution(0, 1);
52     for (int step = 1; step < iters; ++step)
53     {
54         for (int i = 0; i < p_num; ++i)
55         {
56             //更新速度向量和位置向量
57             spd[i] = 0.5 * spd[i] + 2 * distribution(rng) * (p_best[i] - pos[i]) +
58                 2 * distribution(rng) * (g_best - pos[i]);
59             pos[i] = pos[i] + spd[i];
60             //如果越界则取边界值
61             if (spd[i] < -2 || spd[i] > 4)
62                 spd[i] = 4;
63             if (pos[i] < -1 || pos[i] > 2)
64                 pos[i] = -1;
65             //更新位置矩阵
66             pos_mat(step, i) = pos[i];
67         }
68         //更新函数值矩阵
69         for (int i = 0; i < p_num; ++i)
70         {
```

```
1         auto temp = fun_test(pos[i]);
2         f_test(step, i) = temp;
3     }
4     for (int i = 0; i < p_num; ++i)
5     {
6         MatrixXd temp_test;
7         temp_test = f_test.col(i);// 取函数值矩阵的每一列
8         std::ptrdiff_t minRow, minCol;
9         temp_test.minCoeff(&minRow, &minCol);// 获取每一列的极小值对应的位置
0         p_best[i] = pos_mat(minRow, i);// 获取每一列的极小值，即每个粒子的历史最优位置
1     }
2     g_best = *min_element(p_best.begin(), p_best.end());// 获取全局最优位置
3 }
4     cout << fun_test(g_best);
5 }
6
7 int main()
8 {
9     init();
0     PSO();
1     system("pause");
2     return 0;
3 }
```

: <https://blog.csdn.net/myarrow/article/details/51507671>  
: <https://blog.csdn.net/google19890102/article/details/30044945>  
: [wenku.baidu.com/view/65c600b9294ac850ad02de80d4d8d15abe230048.html](https://wenku.baidu.com/view/65c600b9294ac850ad02de80d4d8d15abe230048.html)  
: <https://blog.csdn.net/darin1997/article/details/80675933>

群算法及应用

09-30

群算法及应用 主要讲解蚁群粒子群算法的原理和若干应用场景

群优化算法（PSO）简介及MATLAB实现

Yancy的博客 4万+

粒子群优化算法概述 PSO算法步骤 PSO（粒子群优化算法）与GA（遗传算法）对比 PSO的MATLAB实现 粒子群优...

优质评论可以帮助作者获得更高权重

评论

大威天龙世尊地藏： 讲解的真清楚，不过在实例步骤2中的第三个粒子P3写错了，P3的位置应该是x3=x3+v3 感谢博主！ 2年前 回复 ...

weixin\_47477263： 所有的教材都不如这个例子讲得透彻。赞！ 8月前 回复 ...

guofei9987： 这里有整理好的粒子群算法Python实现 <https://github.com/guofei9987/scikit-opt> 2年前 回复 ...

HigCol： 博主您好，看完您的博客后，写的真是太好了。速度这个向量其实就是跟梯度下降算法中的微分是一样的道理，就是控制鸟往接近全局最优解的地方去飞；然后PSO比梯度下降好的地方是，大家一起来确定怎么飞，梯度下降只是一只鸟往哪儿飞，所以梯度下降很容易飞到局部最优解，PSO可以飞到大家都认为的全局最优解；初始化的速度向量，初始化的量应该不会影响PSO算法的最终结果，但是会影响计算的过程对吧？ 2年前 回复 ...

 lx青萍之末 [博主] 回复： 初始值可以随机取嘛，只是迭代的过程不同，最终的趋势是一样的 2年前 回复 ...

码哥 萧逸凡： 感谢博主，阅读后对PSO的理解更深刻了，算例中是不是有错误？ 5天前 回复 ...

爱码士 小周同学的博客： 博主不光能写的一手好代码，还能写的一手好文章。 18天前 回复 ...

weixin\_43558200： 请问粒子群算法求多个最优解怎么得呀，比如说选址问题，我要得到多个最佳位置 1月前 回复 ...

 qq\_36545870 回复： 按照上面的做法之后，做个排序？记录一下位置 1月前 回复 ...

My Alter： 感谢 2月前 回复 ...

点赞196

评论30

分享

收藏882

打赏

举报

关注

一键三连

[https://blog.csdn.net/daaikuaiquan/article/details/81382794?ops\\_request\\_misc=%25257B%252522request%2525Fid%252522%25253A%25252216098126011678...](https://blog.csdn.net/daaikuaiquan/article/details/81382794?ops_request_misc=%25257B%252522request%2525Fid%252522%25253A%25252216098126011678...) 6/9

- fuzimango: 动图很帅 3月前 回复

2
- 码哥\*hkcooll: 这个算法容易陷入局部最优，死循环 6月前 回复

2
-  qq\_45385524 回复： 粒子数取大点试试 6月前 回复

2
- <

1

2

3

>

**能算法】粒子群寻优算法** weixin\_30752699的博客 2476

2基础 **粒子群算法**（particle swarm optimization，**PSO**）是计算智能领域中的一种生物启发式方法，属于**群体智能**优...

**群算法** yy2050645的博客 1万+

翁博客，为大家简单介绍**粒子群算法（PSO）**。**粒子群算法**同遗传**算法**相似，也是根据生物界中的

**化算法之粒子群算法(PSO)** - weixin\_44244371的博客 - CSDN博客 10-7

LABJ 经典智能**算法1:粒子群优化算法PSO** 阅读量 245 经典智能**算法**文章集:**粒子群优化算法PSO**:一、**粒子群优化算...**

**化算法---粒子群算法(PSO)\_BIRD\_CHARLES** 12-11

属于进化**算法**的一种,和模拟退火相似,也是从随机解出发,通过迭代寻找最优解,也是通过适应度来评价解的品质,比遗...

**pso（粒子群优化）算法代码合集** 07-21

代码，共13类

**群算法详解** ZCC的专栏 9万+

生背景 \***粒子群算法**(particleswarm optimization，**PSO**)由Kennedy和Eberhart在1995年提出，该**算法**对于Hepper...

**群优化算法(PSO)\_森的博客\_粒子群算法** 12-18

**群算法**的发展过程。**粒子群优化算法**(Partical Swarm Optimization **PSO**),**粒子群**中的每一个**粒子**都代表一个问题的可...

**群优化算法(PSO)\_DragonBallSuper的博客** 12-5

**群优化算法(PSO**:Particle swarm optimization)是一种进化计算技术(evolutionary computation)。源于对鸟**群**捕食的行...

**解决TSP问题（粒子群算法解决旅行商问题）--python实现** DomicZhong 1万+

感谢这位github上的Marcos Castro de Souza大神 本文利用代码链接如下：https://github.com/marcoscastro/tsp\_ **ps**o...

**群优化算法(PSO)** MyArrow的专栏 9万+

念 **粒子群优化算法(PSO**：Particle swarm optimization) 是一种进化计算技术（evolutionary computation）。源...

**化算法 之 PSO算法\_YoYoDelphine的博客** 12-22

**SO算法**简要介绍 **PSO算法**是一种**最优优化算法**。它预定义一组**粒子**,和计算**粒子**适应值(fitness value)的函数。然后进...

**群算法(PSO)示例的C++实现\_larry233的博客** 12-25

**群算法**的介绍和原理见**最优优化算法之粒子群算法(PSO)** 优化目标函数如下: 该目标函数可改为其他函数,设定的**粒子**数...

**thon3实现粒子群优化算法（PSO）** FREEMAN 4万+

**群优化算法**（Particle Swarm Optimization，**PSO**）属于进化**算法**的一种，是通过模拟鸟**群**捕食行为设计的。从随机...

**群算法原理及Matlab实现（PSO — Particle Swarm Optimization）** Tan\_HandSome的博客 2万+

在计算机科学中，**粒子群**优化（**PSO**）是一种计算方法，可以通过迭代来改进候选方案的优化问题。它通过**粒子**的...

**群算法(PSO)详解以及使用\_cxy\_hust的博客** 12-30

原理(particle swarm optimazition) 1.1 **算法**使用范围 根据名字: **粒子群**优化,其实就界定了该**算法**是作为一个优化算...

 lx青萍之末  
码龄6年 暂无认证

196 30 882 117



1万+

积分

1251

粉丝

1297

获赞

238

评论

5152

收藏



私信

关注

搜博文文章

Q

热门文章

矩阵求导、几种重要的矩阵及常用的矩阵求导公式  194962

最优化算法之粒子群算法（PSO）  123685

epoll原理详解及epoll反应堆模型  55483

进程、线程和协程之间的区别和联系  52870

MATLAB读取mat数据并绘图  48285

分类专栏

 编程语言

 C/C++基础知识46篇

 C++进阶23篇

 C++STL14篇

 C++内存管理5篇

 C++并发编程10篇

▼

最新评论

UML各种类图总结

codertalk: 学习了

最优化算法之粒子群算法（PSO）

萧逸凡: 感谢博主，阅读后对PSO的理解更深刻了，算例中是不是有错误？

matlab自带各种分类器的使用示例

Gahd: 老哥，这代码配色不太行啊

余弦相似度

qq591840685: 结果是根号9和根号8吧？最终结果0.7071

进程间通信之管道（pipe）和命名管道（...

\_llc: 单双工

最新文章

Thread参数传递问题

cmake知识点总结

vscode远程开发环境搭建

2020年8篇

2019年159篇



2016年 18篇