最优化算法——遍历法

最优化问题通俗来说就是求解函数的最大值和最小值。在理论数学中，常常通过找导数为 0 的点来确定极值点，但是对于一些函数表达式比较复杂甚至没有具体表达式的问题，这种方法就失效了。

因此，在计算机求解最优化问题中，有基于梯度（数值导数）的梯度下降算法，但是更多的是与导数无关的算法，包括遍历法、随机优化算法、二分搜索算法等。

本节首先介绍遍历法，遍历法的思想很简单，就是首先在定义区间内生成一系列自变量点，比如$x=0:0.01:10$，然后带入函数$y=f(x)$，输出因变量向量y，画出图像观察，最后采用 min/max 函数，计算出向量y的最小值/最大值。

遍历法有几个问题。第一、如果需要得到比较精确的解，需要尽可能的缩小步长，但是缩小步长就会导致计算次数增加，需要反复调整；第二、遍历法是采用 for 循环来实现的，对于多元函数，则需要多重的 for 循环来实现，如果一重循环是n次，二次循环就是n^2次，三重循环就是n^3次，所以遍历法至多求解二元函数的优化问题。

求解函数$y=x^4-\dfrac{32}{3}x^3+34x^2-35x+5, x\epsilon[0,6]$上的最小值

这个函数可通过求导计算，但是求解三次方程是比较麻烦的，所以这里采用遍历法画出函数在$[0,6]$上的图像，采用 min 函数，近似计算最小值

```
clear;close all;clc;
d=0.01;
x=0:d:6;
y=x.^4-32/3*x.^3+34*x.^2-35*x+5;
plot(x,y,'k')
xlabel('x')
ylabel('y')
[miny,k1]=min(y)%k是y向量中最小值对应的序号
xmin=x(k1)%x向量第k个元素就是最小值对应x值
hold on
plot(xmin,miny,'ro')%画出最小值点
```

命令行窗口

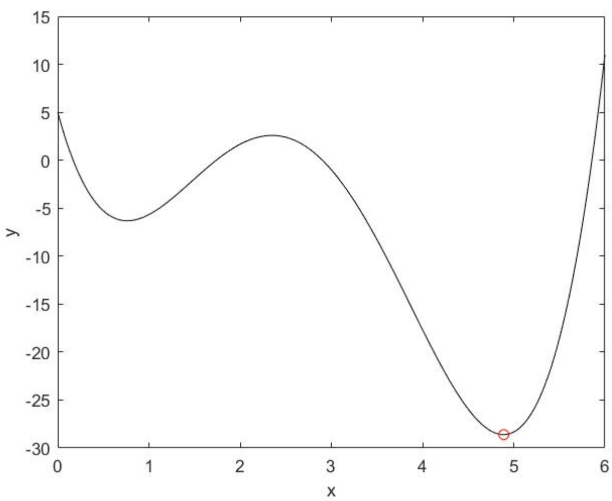
```
   -28.6052

k1 =

   490

xmin =

    4.8900
```



核心就是画出函数图像，然后采用 min/max 函数计算最值。前面也说到过，遍历法计算结果有一定误差，所以**该方法常用于初步确定极值点范围，后续可结合随机优化算法或者二分搜索算法进一步确定极值点精确值**。