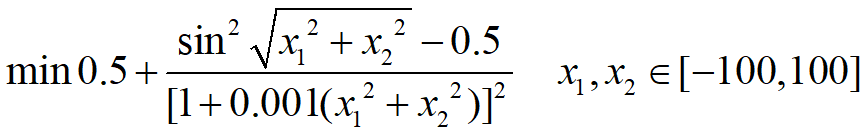
信号处理的智能化方法及应用 第四次作业

1. **分别用经典优化算法和混动搜索算法求解如下优化问题，比较二者的平均搜索效率 （500次平均）。**



1. 实现环境：MATLAB R2019a
2. 环境配置：Windows 10 Enterprise Version 2004, AMD Ryzen R7 2700
3. 实现方法：构造了一个函数，负责通过函数指针调用不同的搜索方法实现，并计算每种方法的最小/平均/最大耗时。
4. 运行结果

程序对比运行了给出的5份代码，分别是：直接使用fminsearch函数进行搜索、使用方程产生的混沌吸引子进行搜索、使用方程产生的混沌吸引子进行搜索和分别使用两种混沌吸引子的同时使用fminsearch搜索的情况，对计算的结果进行整理如下：

计算结果：

Xmin =

1.0e-04 \*

-0.0679 0.2520

Ymin =

0

计算耗时：

直接使用fminsearch函数进行搜索：

"k: min/avg/max

12 / 4.526106e+03 / 29216"

"time: min/avg/max

0.011028 / 4.305820 / 29.620765"

使用方程产生的混沌吸引子进行搜索：

"k: min/avg/max

900983 / 1.366792e+07 / 341587417"

"time: min/avg/max

0.198690 / 3.033614 / 59.220853"

使用方程产生的混沌吸引子进行搜索：

"k: min/avg/max

323071 / 1.771064e+07 / 93287297"

"time: min/avg/max

0.225055 / 12.547952 / 57.714030"

使用方程产生的混沌吸引子配合fminsearch进行搜索：

"k: min/avg/max

267 / 4.528400e+03 / 10324"

"time: min/avg/max

0.269927 / 6.579889 / 16.272447"

使用方程产生的混沌吸引子配合fminsearch进行搜索：

"k: min/avg/max

260 / 7767 / 28417"

"time: min/avg/max

0.253191 / 8.568144 / 31.859297"

1. 小结

由于一些原因，部分混沌搜索函数的次数没有跑满500次，但是次数与500次接近，因此认为没有影响。

在测试中发现，有的时候程序会在很长的一段时间内没有计算出结果，这应该是因为混沌系统对初值的敏感性较大，而随机生成的初值恰巧不是很合适导致的，很长时间还没有得出结果的尝试被直接舍弃了，这也是部分次数没有跑满的原因。

输出中，k是迭代次数，time是所花时间，可以看出，混沌搜索在一定的条件下可以增加搜索的速度，但同时也会导致搜索速度变得不太稳定，即有时搜索时间过长（甚至感觉可能没法收敛）。同时，使用fminsearch辅助搜索可以显著减少迭代的次数，对于部分方法也可以降低所需要的时间，而且使用fminsearch可以一定程度上缓解由于随机初值导致的收敛时间过长的问题，可以看见在应用了fminsearch作为辅助后，最大时间有所下降（同时超时舍弃的次数也变少了），但是还是摆脱不了随机性带来的一些不稳定因素。

另外比较奇怪的一点是，程序对于CPU的利用率很低，之前在学习数字图像处理时，处理大量数据的时候CPU利用率基本上是100%，但是在这次的处理中MATLAB对CPU的占用很低，只有不到10%，而且在执行计算时没有任何一个CPU核心占用高于20%，我怀疑可能是程序的调用中存在一些问题，导致并不是在全力计算优化问题。

完整代码源文件请查看：

<https://git.nju.edu.cn/Minaduki/Intelligent_Signal_Processing>