智能优化方法[[1]](#footnote-1)

计算方法 期末作业

1. **算法任务**

最优化问题，本质上就是在一定的约束条件中，求一个特定目标函数的最值以及取得最值时自变量的值，通常会将优化问题转换为无约束的最小值问题进行求解。最优化问题在许多领域中都有广泛的应用，典型的就是在机器学习领域用来寻找参数，将代价函数的最小值作为最优化问题的优化目标，搜索得到最优的参数。

最优化问题的关键是求解全局最优值的算法，也就是“优化方法”，选择适当的优化方法，可以降低计算时间，提升计算精度。

1. **背景：传统算法**

负梯度下降法是一种使用广泛的经典优化方法，由于局部最优解应为梯度为0的驻点，因此可以将梯度下降最快的方向作为搜索方向，通过一定的方法选定步长，如是循环直到满足终止条件，理论而言，只要步长足够小，梯度总是下降的，因此能够近似得到局部最优解。

牛顿法利用二阶的泰勒展开式寻找与目标函数相近的二次函数的最小值点，从而可以使用更少的迭代次数快速下降，而且对于二次的目标函数可以一次到达最小值点，搜索效率较高；二阶共轭梯度法（Fletcher-Reeves法）利用已有的迭代点的梯度与当前点的梯度共同决定下降方向，相比一阶梯度下降方法也有更快的搜索速度。不过，这两种方法相对一阶梯度法有一定的限制条件，比如需要高阶导数存在保证泰勒展开有意义，数值性相比一阶较差，因此除非优化问题特殊，适合高阶方法，否则一般采用一阶梯度下降法。

上面的三种方法都属于基于梯度的优化算法，在解决凸优化问题时具有很好的精度和鲁棒性，不过根据NFL定理，在解决非凸优化时梯度类算法的表现就较为乏力了，事实也的确如此，线性类的搜索方法只能找到局部最优，解决非凸问题需要不断变换搜索起点以寻找全局最优解，搜索效率较差，结果也不稳定。

混沌搜索方法是一种典型的非线性搜索方法，混沌搜索使用一个特定的公式作为混沌吸引子，通过迭代并映射产生每次搜索的自变量，由于混沌系统具有貌似随机和初值敏感的性质，因此可以用于进行大范围的搜索，从而避免陷入局部最优，遍历尽可能多的值。

但是同时，混沌具有分形特性，这导致混沌吸引子对区间内不同部分的搜索力度是不一样的，因此优化结果精度与质量不稳定，同时混沌算法的搜索方向没有引导，因此在搜索的速度上也不如梯度类算法稳定。

混沌搜索算法对于部分非凸问题的处理强于梯度类算法，但是仍然是一种确定性的搜索，只要起点给定，搜索过程即确定，总体而言在求解非凸问题时仍然比较费力，鲁棒性差。为了高效稳定地处理非凸优化问题，需要一些智能优化方法的帮助。

1. **算法原理简介**

智能优化方法是一系列带有引导的启发式随机性搜索算法的总称，这些算法往往是基于一些物理、生物、社会、思维等领域中涉及的优化过程提出的，一般不依赖于目标函数的梯度，对于特定的非凸问题可以取得比传统算法更好的表现。

模拟进化算法模仿生物进化的规律进行搜索，使用模拟DNA链的方式编码优化问题中的参数作为生物进化中的个体，在每一代进化中中，个体会经过交叉、突变、选择三个操作，从而产生新的一代，以分别模拟生物学中“遗传”、“变异”和“自然选择”的角色。

其中，交叉算子负责以随机的概率在随机的位置切断两个个体的染色体，并将截断部分进行交换，突变算子则以随机重写随机点为随机值，选择算子使用了排序选择的方式，根据不同个体的目标函数值（适应度）得到一张概率分布表，每一个子代个体都依据概率分布表从父代选取。

模拟进化算法具有并行、随机、自适应的特点，既保证了搜索力度足够，同时也在搜索中保留了足够的已知信息引导下一步搜索，提升搜索效率。

普通的模拟进化算法相对比经典的优化算法，可以在一定程度上提升搜索的效率，并拥有更强的鲁棒性，但是也存在一定的不足，比如解的精度往往不是很高，而且容易出现“早熟”现象，即陷入局部最优。

人工免疫算法工作机理与模拟进化算法有一些相似，但是在产生新的个体时模仿的是人体免疫细胞分裂、分化、受体编辑等的机理，同时引入了局部搜索的概念以模拟成熟克隆种群的生成。

小生境进化算法是对普通的模拟进化算法的一个改进，旨在“提升物种多样性”，强化全局勘探能力，改善搜索时可能遇到的早熟问题。基于预选择的小生境方法是在模拟进化算法中，不应用选择算子，而是在交叉、变异后，比较子代个体与其对应的父代个体的适应度，并选取父子中适应度更高的个体保留，以保证整个种群的潜力，提升多样性。

猴王爬山算法是一种混合进化算法，将基本的模拟进化算法与爬山法相结合，算法中，新一代的群体主要由猴王（上代最优个体）与其他随机个体交叉产生（也包括猴王个体），同时对新群体产生的猴王再进行局部爬山的操作，通过这样的设计弥补模拟进化算法局部寻优能力不足，解精度低的缺陷。

1. **算法实现**

**算法流程：**

模拟进化算法：

1. 在目标函数的定义空间随机生成一定数量的个体作为初始父代群体
2. 评估初始父代群体，如果满足停止条件则跳出，否则继续
3. 根据父代群体的适应度评估结果应用选择算子，初步产生子代群体
4. 在子代群体中应用交叉算子与突变算子
5. 评估子代群体，若满足停止条件或超时则跳出，否则作为新的父代群体转步骤3

人工免疫算法：

1. 随机产生一定规模的初始抗体种群
2. 对于每一个抗体，计算亲合度（目标函数），选取其中亲合度最高的n个抗体
3. 对于选取出的每个抗体，产生与其亲合度成比例的数目的克隆，作为克隆种群C
4. 对种群C的个体实施超变异（局部搜索），产生成熟克隆种群C\*
5. 重新计算种群C\*个体的亲合度，并选取亲合度最高的一个抗体，总共有n组克隆种群，因此最后会得到n个抗体
6. 将得到的抗体中亲合度最低的d个抗体剔除，随机生成d个抗体取代之，以模拟受体编辑机制
7. 如果满足终止条件或超时则跳出，否则转步骤2

小生境改进模拟进化算法：

1. 在目标函数的定义空间随机生成一定数量的个体作为初始父代群体
2. 在群体中应用交叉算子与突变算子，产生子代群体
3. 评估子代个体并与其父代个体进行比较，保留两者中适应度更佳的，产生新群体
4. 评估新群体，若满足停止条件或超时则跳出，否则作为新的父代群体转步骤2

猴王爬山算法：

1. 随机生成数个个体并按照适应度排序，取最高的个体为猴王
2. 将猴王与随机个体进行交叉，得到新一代种群
3. 评估新一代种群，得到一个临时的猴王
4. 在猴王的邻域内产生若干个随机个体，与猴王进行排序，最优点成为新猴王，此步骤迭代若干次
5. 如果满足终止条件或超时则跳出，否则转步骤2

**代码实现：**

实现环境：MATLAB R2019a

环境配置：Windows 10 Enterprise Version 2004, AMD Ryzen R7 2700 @ 3.65 GHz

原始代码请在附录中查看。

1. **算法运行结果**

使用不同的算法计算如下的优化问题：

图片包含 游戏机, 钟表

描述已自动生成

该问题在0.0097处存在一个局部最优解。

程序运行对比了使用的几种优化算法，分别是：

1. 使用方程作为吸引子的混沌优化方法
2. 模拟进化算法
3. 人工免疫算法
4. 小生境进化算法
5. 猴王爬山算法

每一个算法运行了100次以避免偶然情况，运行结果如下，其中k为迭代次数，time为花费的时间，迭代次数上限为1e9（混沌算法）或10000（智能优化算法）：

1. 混沌优化方法

"k: min/avg/max

900983 / 1.366792e+07 / 341587417"

"time: min/avg/max

0.198690 / 3.033614 / 59.220853"

搜素速度很快，但是效果不够稳定，在100次运行中，混沌算法有9次触发了超时（实际上是迭代次数上限而非时间）且没有完成计算（误差限9e-3），这也是所有的算法中唯一一个没有跳出局部最优而触发了超时的算法，证明对于非凸问题，混沌优化算法总体表现尚可，但是鲁棒性不佳。

1. 模拟进化算法

**（对照组）种群大小为1000，交叉率0.8，变异率0.9，误差限9e-3：**

"k: min/avg/max

53 / 692.1 / 1977"

"time: min/avg/max

1.893075 / 24.912441 / 70.977121"

**增加种群大小至2000，其余参数与对照组一致：**

"k: min/avg/max

49 / 121.4 / 207"

"time: min/avg/max

3.772765 / 9.285405 / 15.759829"

种群大小增加，意味着一次迭代能够搜索、变化的个体增加，搜索力度增大，因此迭代次数和搜索时间均有显著的下降，同时程序占用的内存有一定的提升，这也是模拟进化算法的一个特征：可以并行运算，以空间换取时间。

**减小交叉率至0.5，其余参数与对照组一致：**

"k: min/avg/max

160 / 544.5 / 1851"

"time: min/avg/max

4.930906 / 16.777650 / 57.063620"

**减小交叉率至0.1，其余参数与对照组一致：**

"k: min/avg/max

0 / 1340.4 / 2704"

"time: min/avg/max

0.013636 / 31.863193 / 64.200378"

当交叉率过低时，从理论上讲，这会导致产生新的特征的概率降低，增加早熟的风险，因此搜索力度会降低。从测得数据上，搜索时间增加了一些，但是增加幅度不算很大，迭代次数显著增加，这应该是因为降低了交叉率会导致一个iter的工作量降低，从而单位iter的耗时减少，因此，iter总数的增加对总时间的变化影响不大。不过这个结论是针对当前选用的简单测试问题而言，应该要注意如果求解规模比较大，更多的迭代次数可能需要更大的I/O吞吐量，I/O瓶颈有可能造成更加显著的性能下降。

**减小变异率至0.5，其余参数与对照组一致：**

"k: min/avg/max

175 / 502.8 / 1272"

"time: min/avg/max

6.342956 / 18.212994 / 46.167151"

**减小变异率至0.1，其余参数与对照组一致：**

"k: min/avg/max

34 / 144.9 / 526"

"time: min/avg/max

1.197066 / 5.166721 / 18.744069"

比较反直觉的一点是，变异有助于新的特性产生，因此高变异率应该有助于提升搜索的遍历性，提高搜索速度，但是实际结果却表示，变异率减小反而有助于提升搜索力度。

这是因为在特定的进化问题中，变异率有一个最佳区间，如果变异率过高，虽然会有更大的概率产生新的特性，但是也造成种群中原有的最佳个体话语权下降，而最佳个体更有搜索潜力，是进化计算算法的核心，变异率过高则最佳个体的特征会被稀释，实质上相当于进化算法部分退化为了传统的随机搜索，因此变异率过高并不总能提升搜索力度。

1. 人工免疫算法

**（对照组）免疫个体数量为1000，变异率0.9，误差限9e-3：**

"k: min/avg/max

3 / 85 / 379"

"time: min/avg/max

1.041724 / 15.772979 / 68.839765"

总体性能与模拟进化算法相近。

**减小变异率至0.7，其余参数与对照组一致：**

"k: min/avg/max

7 / 33.9 / 72"

"time: min/avg/max

1.757468 / 6.595245 / 13.358293"

**增加免疫个体数量至2000，其余参数与对照组一致：**

"k: min/avg/max

4 / 15.8 / 54"

"time: min/avg/max

4.452056 / 12.941700 / 40.513180"

与模拟进化算法一样，降低变异概率至合适区间或者提升个体数量都可以有效提升算法的遍历性，从而提升搜索速度，不过在免疫算法中，提升个体数量后，每个iter增加的计算量要多于模拟进化算法，因此虽然迭代次数大幅减少，但是计算时间降低的幅度没有进化算法显著。

1. 小生境进化算法

小生境算法的主要目标是摆脱早熟，下面是误差限设定为9e-3时的计算性能：

"k: min/avg/max

1 / 20.7 / 40"

"time: min/avg/max

0.258069 / 4.393240 / 8.480705"

性能与相比模拟进化算法有不小的提升，算法在很少几个iter后就摆脱了0.0097的早熟点。

不过对应的，小生境算法没有对精度作出优化，如果将误差限设定为1e-6，算法无法在给定的10000代以内收敛，如果增加收敛代数限制至100000，可以计算出结果，但是耗时非常长，因此我没有花时间进行足够次数的验证，这也证明了局部搜索机制的必要性。

1. 猴王爬山算法

猴王爬山算法通过局部爬山搜索来增加搜索的精度，同时整体的进化算法通过引入随机个体避免早熟。

"k: min/avg/max

10 / 218.9 / 543"

"time: min/avg/max

0.035923 / 0.806553 / 1.985902"

算法的表现非常好，在很短时间内就完成了搜索，同时结果有着很高的精度，通过反复测试，误差限大于等于1e-8时，算法可以在3s内完成搜索。

1. **小结**

智能优化算法之所以“智能”，是因为算法不是“贪婪”地只看见邻域，而是利用了已完成的优化进程中的信息来指导后续优化的进程，具体到作业中介绍的几种基于进化计算的算法，这个信息就是最优个体。因此算法的核心应该是围绕着最优个体进行一些取舍，一方面要保证最优个体在进化过程中的话语权，比如通过适当选择变异概率提升搜索的性能，另一方面也要保证多样性，确保最优个体能够充分接触各个新产生的“可能性”，这也是生物进化中遗传和变异分别起到的作用，只有两个方面的参数都达到最佳，才能得到最好的搜索性能。

当然，智能优化方法不止于进化计算，也有模拟物理规律的退火算法，模拟生物活动的蚁群算法与粒子群算法以及策略驱动的禁忌搜索，不过，智能优化方法也存在一些局限，智能优化方法严格上说并不是一类非常“数学”的算法，人类对自然认知的局限可能会影响由此产生的智能优化方法的性能。

从实验结果看来，单一的算法无论是在精确度还是搜索能力方面都过于中庸，像猴王爬山算法那样将多种算法结合起来以加强在特定问题上的性能是智能优化方法的一个发展方向，比如，进化退火算法等基于进化算法的混合拓展方法常用于提升结果的精度。同时，如何根据优化问题选择适合的算法与参数也是一个很重要的问题，根据NFL定理，没有适用于所有领域的算法，按照策略选择、结合不同低层次启发算法用于解决特定优化问题的超启发式算法就是针对这个目标提出的。但是不论怎么说，更稳定更精确的算法永远会是算法发展的未来，也是学习与努力的方向。

# 附录[[2]](#footnote-2)

完整代码源文件请查看附件或：

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

1. 本报告中介绍的算法来自刘红星老师在《信号处理的智能化方法及应用》课程中讲解的部分优化算法 [↑](#footnote-ref-1)
2. Git repo: https://git.nju.edu.cn/Minaduki/Computational\_Methods [↑](#footnote-ref-2)