

学校代码 10701
分 类 号 TP31

学 号 1107310271
密 级 公开

西安电子科技大学

博士学位论文

几类优化问题的人工蜂群算法

作者姓名: 孔翔宇

一级学科: 数学

二级学科: 应用数学

学位类别: 理学博士

指导教师姓名、职称: 刘三阳 教授

学 院: 数学与统计学院

提交日期: 2016 年 4 月



Y3084689

Artificial Bee Colony Algorithm for Some Kinds of Optimization Problems

A dissertation submitted to
XIDIAN UNIVERSITY
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in Applied Mathematics

By
Kong Xiangyu
Supervisor: **Liu Sanyang** Professor
April 2016

西安电子科技大学
学位论文独创性（或创新性）声明

秉承学校严谨的学风和优良的科学道德，本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果；也不包含为获得西安电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同事对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

学位论文若有不实之处，本人承担一切法律责任。

本人签名：日期：

孙翔宇 2016.6.24

西安电子科技大学
关于论文使用授权的说明

本人完全了解西安电子科技大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权属于西安电子科技大学。学校有权保留送交论文的复印件，允许查阅、借阅论文；学校可以公布论文的全部或部分内容，允许采用影印、缩印或其它复制手段保存论文。同时本人保证，结合学位论文研究成果完成的论文、发明专利等成果，署名单位为西安电子科技大学。

保密的学位论文在年解密后适用本授权书。

本人签名：导师签名：刘三阳
孙翔宇
日期：日期：2016.6.24
2016.6.24

摘要

优化问题无处不有，而且各个领域提出的优化问题越来越复杂，许多传统的优化算法在一些复杂的优化问题面前变得无能为力。因此，人们不断寻找新的求解方法。仿生智能优化算法的出现，使得这些复杂优化问题的求解变为现实。由于该方法一般不需要目标函数和约束条件的任何先验信息，能较好适应复杂优化问题的求解，具有并行计算的特点，同时具有很好的鲁棒性，仿生智能优化算法一出现就引起了广大科研人员的关注，被应用于旅行商问题、神经网络训练、图像处理等问题。2005年提出的人工蜂群算法就是一种仿生智能优化算法，已成功应用于众多研究领域。但人工蜂群算法与以往的智能优化算法一样，也存在种群探索能力与开发能力之间不平衡的问题，算法的探索能力强大，但开发能力不足，易于陷入局部最优，收敛速度慢等，且其数学理论基础也十分薄弱。本文针对人工蜂群算法的缺点，提出了几种改进的人工蜂群算法，成功应用于函数优化问题，约束函数优化问题和非负线性最小二乘问题。同时通过引入随机过程中的鞅理论证明了人工蜂群算法的几乎处处必然强收敛性。本文的主要研究工作如下：

1、证明了人工蜂群算法的收敛性。已有的人工蜂群算法收敛性分析是基于算法的遍历性分析，在概率收敛意义下考虑的。这种收敛性分析不确保算法在有限步内收敛到问题的全局最优解。本文尝试运用鞅论研究人工蜂群算法的几乎必然强收敛性，证明了人工蜂群算法确保能以概率1在有限步内达到全局最优解。这一结论为拓宽人工蜂群算法的应用范围奠定理论基础，并为人工蜂群算法的改进及收敛性研究提供了新的理论工具。

2、提出了一种全局优化问题的混合人工蜂群算法。该算法通过引入正交初始化方法，使初始种群分布更加均匀，提高了算法的搜索效率，同时对原始算法雇佣蜂和跟随蜂的搜索方程添加随机扰动项，扩大了算法的搜索范围，利用 `benchmark` 函数测试所提算法的有效性，实验结果、收敛曲线图和箱型图表明算法能快速收敛到全局最优解。

3、提出了一种求解函数优化问题的改进人工蜂群算法。该算法受差分进化算法的启发，对雇佣蜂和跟随蜂提出了两种新的搜索方程，利用 p 概率控制两个方程的使用，同时引入正交初始化方法，利用 `benchmark` 函数测试改进人工蜂群算法的性能，实验结果表明，所提算法比一些改进人工蜂群算法和其他智能优化算法更有效。

4、提出了一种求解非负最小二乘问题的人工蜂群算法。该算法受粒子群算法和差分进化算法的启发，提出两种改进的搜索方程，在算法执行过程中不在区分雇佣蜂和跟随蜂的区别，将其统一为一种蜜蜂，利用 p 概率控制两个搜索方程的使用，通过

测试函数确定参数 p 值，同时引入正交初始化，利用 benchmark 测试函数和非负最小二乘问题的实例测试，表明所提出的算法比其他改进算法和其他智能优化算法收敛速度更快，解的精度更高。

5、提出了一种全局优化问题的混沌人工蜂群算法，并将该算法应用于非负线性最小二乘问题。为了克服人工蜂群算法的不足，算法中使用反学习初始化作为种群初始化方法。为了进一步改进算法的性能，均衡算法的探索能力和开发能力，为新算法设计了新的搜索机制。另外，在算法中加入混沌局部搜索机制，使得算法在最优解周围进行局部搜索。利用 benchmark 测试函数和非负最小二乘问题的实例测试，表明所提出的算法比其他改进算法和其他智能优化算法收敛速度更快，解的精度更高。

6、针对约束函数优化问题，提出了一种算法投资组合的人工蜂群算法。该算法基于自适应约束处理进化策略和 Deb 选择策略下的人工蜂群算法，原有的处理约束函数优化问题的方法大部分是把算法的运行时间全部投入到一种算法或者是约束技术上去，该算法投资组合把算法的运行时间分配到不同的算法和约束处理技术上，避免了算法运行的风险，同时在算法中设计一个迁移阶段，用来反应算法之间的信息交流与共享，benchmark 函数实验结果表明该算法比其他算法在求解约束优化问题时拥有更高的成功率，更有效。

关键词：人工蜂群算法，群智能算法，正交初始化，反学习初始化，非负线性最小二乘，鞅

ABSTRACT

Many problems bring into solving optimization tasks. Along with the developments of natural science, modern optimal problems become more and more complex. Because of the complexity of optimization problems, traditional optimization methods are inadequate for solving these problems. Therefore, the new methods are needed for these problems. Much more universal, about the kinds of solved problems, are the algorithms motivated by simulating biology intelligent. The only assumption needed by those algorithms is the existence of the solution. Those simulating biology intelligent algorithms can solve complex optimization problems adequately and robustly. Those algorithms are parallel algorithms. Therefore, after initial invented of these algorithms many studies have been carried out on these algorithms and applied for solving TSP problem, neural network training, image processing and others. Artificial bee colony (ABC) algorithm has been proposed in 2005, which is a simulating biology intelligent algorithm. Since its invention, ABC algorithm has been applied in many areas. However, there still an insufficiency in ABC algorithm as other intelligent optimization algorithms, such as lower convergence speed for uni-modal problems and easily trapped in local optimal for complex multimodal problems. And the theoretical analysis is lacking. In this dissertation, some improved artificial bee colony algorithms are proposed to overcome these drawbacks, and applied for numerical function optimization, constrained optimization and non-negative linear least square problem. Meanwhile, a martingale analysis method is proposed to study the almost sure convergence of ABC algorithm. The main researches and creative results of this dissertation are shown as follows.

1. The convergence analysis is proposed for artificial bee colony algorithm. The most convergence analysis on artificial bee colony algorithm is based on ergodicity analysis and conducted in the sense of probabilistic convergence. Such analysis cannot infer in general that the ABC algorithm would be convergent to a global optimum in a finite number of evolution steps. In this paper, a martingale analysis method is proposed to study the almost sure convergence of ABC algorithm. It is shown that ABC algorithm can surely converge to a global optimum with probability 1 in a finite number of evolution steps. The obtained results underlies application of the ABC algorithm, and the suggested martingale analysis method provides a new technique for convergence analysis of ABC algorithm.

2. A hybrid artificial bee colony algorithm is proposed for global optimization problems. To further improve the performance of artificial bee colony algorithm (ABC), a hybrid ABC (HABC) for global optimization is proposed via exploring an orthogonal initialization. Furthermore, to maintain population diversity, a novel search strategy is also developed. The algorithm is applied to benchmark functions with various dimensions to verify its performance. Numerical results demonstrate that the proposed algorithm outperforms the ABC in global optimization problems.
3. An improved artificial bee colony algorithm is proposed for numerical function optimization. To further improve the performance of artificial bee colony algorithm (ABC), an improved ABC (IABC) algorithm is proposed for global optimization via employing orthogonal initialization method. Furthermore, to balance the exploration and exploitation abilities, motivated by differential evolution, a new search mechanism is also designed. The performance of this algorithm is verified by using benchmark functions. And the comparison analyses are given between the proposed algorithm and other nature-inspired algorithms. Numerical results demonstrate that the proposed algorithm outperforms the original ABC algorithm and other algorithms for numerical function optimization problems.
4. An effective hybrid artificial bee colony algorithm is proposed in this paper for non-negativelinear least squares problems. To further improve the performance of algorithm, orthogonal initialization method is employed to generate the initial swarm. Furthermore, to balance the exploration and exploitation abilities, motivated by particle swarm optimization and differential evolution algorithm, a new search mechanism is designed. The performance of this algorithm is verified by using benchmark functions and non-negative linear least squares test problems. And the comparison analyses are given between the proposed algorithm and other swarm intelligence algorithms. Numerical results demonstrate that the proposed algorithm obtained better results than other algorithms for global optimization problems and non-negativelinear least squares problems.
5. In this paper, an effective chaotic artificial bee colony approach is proposed to global optimization. And the proposed approach is applied to non-negativelinear least squares

ABSTRACT

problems. To overcome the insufficiency in artificial bee colony algorithm, opposition-based learning initialization method is employed to generate the initial swarm. To further improve the performance of algorithm and to balance the exploration and exploitation abilities, a new search mechanism is designed. Furthermore, a new chaotic local search operator is embedded in algorithm, which can do the local search around the best solution. The performance of this algorithm is verified by using benchmark functions and non-negative linear least squares test problems. Numerical results demonstrate that the proposed algorithm obtained better results than other algorithms for global optimization problems and non-negative linear least squares problems, which shows the feasibility, effectiveness and robustness of proposed algorithm.

6. An algorithm portfolio (AP) based on evolutionary strategy (ES) and artificial bee colony (ABC) algorithm is proposed for constrained optimization problems. Although a wide range of constraints dealing methods have been developed and studied, the performance of an algorithm with different constraints dealing methods may vary significantly from problem to problem. Instead of choosing one algorithm and one constraints dealing method and investing the entire time in it, it would be less risky to distribute computing time in different algorithms and constraint dealing methods. Based on this idea, a EAs algorithm with adaptive trade off model constraint dealing method and a swarm intelligent algorithm with Deb's rules are employed in AP. Moreover, a migration phase is designed to reflect the interactions between two algorithms. The new method is tested on well-known benchmark functions, and the empirical results suggest it outperforms or performs similarly to other state-of-the-art algorithms. In addition, a parameter analysis is conducted and appropriate values for each parameter are obtained.

Keywords: Artificial bee colony algorithm, Swarm intelligent algorithm, Orthogonal initialization, Opposition-based Learning Initialization, Non-negative linear least square, Martingale

插图索引

图 2.1 蜜蜂采蜜行为图.....	13
图 2.2 人工蜂群算法流程图.....	16
图 3.1 收敛曲线比较图.....	29
图 3.2 函数 30 次实验结果箱型图.....	30
图 4.1 四个测试函数在不同选择概率 P 下的结果图.....	40
图 4.2 收敛曲线图.....	42
图 4.3 实验结果箱型图.....	43
图 5.1 四个测试函数在不同选择概率 P 下的结果图.....	54
图 5.2 部分函数收敛曲线图.....	57
图 5.3 $D=30$ 时部分函数收敛曲线图.....	58
图 5.4 $D=60$ 时部分函数收敛曲线图.....	59
图 5.5 NLLS 1 收敛曲线及箱型图.....	61
图 5.6 $n=100$ 时 NLLS 2 收敛曲线及箱型图.....	61
图 5.7 $n=100$ 时 NLLS 3 收敛曲线及箱型图.....	61
图 5.8 $n=100$ 时 NLLS 4 收敛曲线及箱型图.....	62
图 5.9 $n=100$ 时 NLLS 5 收敛曲线及箱型图.....	62
图 6.1 四个测试函数在不同选择概率 P 下的结果图.....	70
图 6.2 部分测试函数收敛曲线图及箱型图.....	72
图 6.3 $D=30$ 时部分测试函数收敛曲线图.....	74
图 6.4 $D=30$ 时部分测试函数箱型图.....	75
图 6.5 $D=60$ 时部分测试函数收敛曲线图.....	76
图 6.6 $D=60$ 时部分测试函数箱型图.....	77
图 6.7 $n=100$ 时 NLLS 1 收敛曲线及箱型图.....	80
图 6.8 $n=100$ 时 NLLS 2 收敛曲线及箱型图.....	80
图 6.9 $n=100$ 时 NLLS 3 收敛曲线及箱型图.....	81
图 6.10 $n=100$ 时 NLLS 4 收敛曲线及箱型图.....	81
图 6.11 $n=100$ 时 NLLS 5 收敛曲线及箱型图.....	81
图 7.1 测试函数 g01、g02 收敛曲线图.....	96
图 7.2 测试函数 g03、g04 收敛曲线图.....	96
图 7.3 测试函数 g05、g06 收敛曲线图.....	96
图 7.4 测试函数 g07、g08 收敛曲线图.....	96

图 7.5 测试函数 g09、g10 收敛曲线图.....	97
图 7.6 测试函数 g11、g12 收敛曲线图.....	97
图 7.7 测试函数 g13 收敛曲线图.....	97
图 7.8 测试函数 g01、g02 可行解比例图.....	97
图 7.9 测试函数 g03、g04 可行解比例图.....	98
图 7.10 测试函数 g05、g06 可行解比例图.....	98
图 7.11 测试函数 g07、g08 可行解比例图.....	98
图 7.12 测试函数 g09、g10 可行解比例图.....	98
图 7.13 测试函数 g11、g12 可行解比例图.....	98
图 7.14 测试函数 g13 可行解比例图.....	99

表格索引

表 3.1 测试函数基本特征表.....	27
表 3.2 混合人工蜂群算法和人工蜂群算法测试结果表.....	28
表 4.1 Benchmark 函数 f_1-f_{16}	38
表 4.2 Benchmark 函数 $f_{17}-f_{27}$	39
表 4.3 改进人工蜂群算法及人工蜂群算法实验结果.....	41
表 4.4 改进人工蜂群算法与其他算法比较结果.....	44
表 4.5 改进人工蜂群算法与其他人工蜂群算法结果比较.....	45
表 5.1 Benchmark 函数 f_1-f_{16}	52
表 5.2 Benchmark 函数 $f_{17}-f_{27}$	53
表 5.3 改进人工蜂群算法及人工蜂群算法实验结果.....	55
表 5.4 改进人工蜂群算法及人工蜂群算法实验结果.....	56
表 5.5 改进人工蜂群算法与其他算法比较结果.....	56
表 5.6 改进人工蜂群算法与其他人工蜂群算法结果比较.....	57
表 5.7 NLLS 问题的实验结果.....	60
表 6.1 Benchmark 函数 f_1-f_{15}	68
表 6.2 Benchmark 函数 $f_{16}-f_{27}$	69
表 6.3 f_1-f_{15} EC-ABC 算法及人工蜂群算法实验结果.....	71
表 6.4 $f_{16}-f_{27}$ EC-ABC 算法及人工蜂群算法实验结果.....	73
表 6.5 EC-ABC 算法与其他算法比较结果.....	78
表 6.6 EC-ABC 算法与其他人工蜂群算法结果比较.....	78
表 6.7 NLLS 问题的实验结果.....	80
表 7.1 Benchmark 测试函数.....	89
表 7.2 migration_interval 和 migration_size 的 20 组取值下得到的均值.....	90
表 7.3 migration_interval 和 migration_size 的 20 组取值下得到的最好值.....	91
表 7.4 migration_interval 和 migration_size 的 20 组取值下得到的最坏值.....	92
表 7.5 migration_interval 和 migration_size 的 20 组取值下得到的标准差.....	93
表 7.6 migration_interval 和 migration_size 的 20 组取值下得到的可行解比率.....	94
表 7.7 实验统计结果.....	95
表 7.8 实验结果的最优值.....	99
表 7.9 最终种群的可行解比率.....	100
表 7.10 测试问题 g01-g06 的实验结果.....	101

表 7.11 测试问题 g07-g13 的实验结果.....101

目录

摘要.....	I
ABSTRACT.....	III
插图索引.....	VII
表格索引.....	IX
第一章 绪论.....	1
1.1 论文选题的研究意义和目的.....	1
1.2 国内外研究现状.....	2
1.2.1 人工蜂群算法的比较与改进.....	3
1.2.2 混合人工蜂群算法的研究.....	5
1.2.3 人工蜂群算法的应用研究.....	5
1.2.4 国内研究现状.....	7
1.3 论文的主要研究内容与结构安排.....	9
第二章 人工蜂群算法及收敛性分析.....	11
2.1 引言.....	11
2.2 蜂群行为.....	12
2.3 人工蜂群算法原理与步骤.....	13
2.4 人工蜂群算法的 Markov 链模型.....	17
2.5 收敛性分析.....	19
2.5.1 依概率收敛.....	20
2.5.2 几乎必然强收敛.....	20
2.6 小结.....	23
第三章 全局优化问题的混合人工蜂群算法.....	25
3.1 引言.....	25
3.2 混合人工蜂群算法.....	25
3.2.1 正交初始化.....	25
3.2.2 新搜索机制.....	26
3.3 数值实验.....	27
3.4 小结.....	31
第四章 函数优化问题的改进人工蜂群算法.....	33
4.1 引言.....	33
4.2 改进人工蜂群算法.....	34

4.2.1 正交初始化.....	34
4.2.2 新搜索机制.....	35
4.2.3 改进人工蜂群算法.....	36
4.3 数值实验.....	37
4.3.1 测试函数.....	37
4.3.2 选择概率 p 的影响分析.....	39
4.3.3 改进人工蜂群算法与人工蜂群算法的比较.....	40
4.3.4 与其他算法比较.....	44
4.4 小结.....	45
第五章 非负线性最小二乘问题的有效人工蜂群算法.....	47
5.1 引言.....	47
5.2 有效人工蜂群算法.....	48
5.2.1 正交初始化.....	48
5.2.2 新搜索方程.....	49
5.2.3 有效人工蜂群算法.....	50
5.3 数值实验及比较.....	51
5.3.1 对 27 个 benchmark 测试函数实验.....	51
5.3.2 对 NLLS 测试问题实验.....	60
5.4 小结.....	62
第六章 非负线性最小二乘问题的混沌人工蜂群算法.....	63
6.1 引言.....	63
6.2 有效混沌人工蜂群算法.....	64
6.2.1 反学习初始化.....	64
6.2.2 新的搜索方程.....	64
6.2.3 混沌局部搜索.....	65
6.2.4 混沌人工蜂群算法.....	66
6.3 数值实验及比较分析.....	66
6.3.1 对 27 个 benchmark 测试函数实验.....	67
6.3.2 对 NLLS 问题实验.....	79
6.4 小结.....	81
第七章 约束优化问题的算法投资组合.....	83
7.1 引言.....	83
7.2 算法投资组合.....	84
7.2.1 基于自适应模型的进化策略.....	84

目录

7.2.2 人工蜂群算法.....	86
7.2.3 算法投资组合.....	88
7.3 数值实验及分析.....	88
7.3.1 参数设置.....	88
7.3.2 结果及分析.....	89
7.4 小结.....	101
第八章结论和展望.....	103
参考文献.....	105
致谢.....	119
作者简介.....	121

第一章 绪论

群智能可简单定义为具有分工和自组织的种群的集体行为。这类种群包含类似于鸟群、鱼群和具有社会性的昆虫群体。尽管蜜蜂群体也明显具备了群智能的基本特征——自组织性和分工，但近期特别是 21 世纪初，学者们才展开对蜜蜂群体智能行为的研究。这些研究当中，人工蜂群算法是目前研究和应用最为广泛的一种算法。本章首先介绍人工蜂群算法的研究意义和目的；然后分析了人工蜂群算法的研究进展；最后，给出了本文的内容提要与结构安排。

1.1 论文选题的研究意义和目的

优化问题是人们在科学工程、国防、交通、管理、经济、金融、计算机等领域经常遇见的问题，通常是在有限种或无限种可行方案中挑选最优方案，构造寻求最优解的计算方法。在人类社会几千年的发展过程中，最优化方法已经深深渗入到人类生活的各个领域，几乎无处不在。最优化方法作为应用数学和运筹学的一个重要分支，从学科诞生之日起就引起了人们的广泛关注，是应用性极强的一门科学。

最优化的思想最早可以追溯到 17 世纪，当时英国的著名学者 Newton 在发明微积分时，就提出了无约束优化问题的极值问题，后来又提出了 Lagrange 乘子法用来求解约束优化问题。直到二十世纪四十年代美国数学家 G. B. Dantzig 在研究美国空军资源配置时，提出了求解线性规划问题的有效方法—单纯形方法，才奠定了最优化方法的基础。随后陆续出现了用于求解线性规划问题的改进单纯形方法，求解非线性规划问题的最速下降法、牛顿法、共轭方向法，求解约束问题的罚函数法等。最优化方法呈现出蓬勃发展的势头。

目前，这些传统的优化方法在求解许多工程优化、经济管理等数学模型时表现出了优异的性能。但随着科学技术的迅猛发展，人们的生产活动也越来越复杂，实际活动所建立的优化模型逐渐向大规模化、多维化发展，有些实际问题可能存在唯一的全局最优解，有些问题存在多个局部最优解，并且有些问题的全局最优解可能还不唯一。传统的优化算法大部分都依赖于初始点的选择、目标函数的连续性、可微性等函数特征来求解优化问题。但对于大规模复杂优化问题，特别是有些实际问题甚至没有显式数学表达式，面对这类问题，传统的优化算法常常无能为力，不能够对优化问题进行求解，这就促使人们探索新的算法。

二十世纪八十年代以来，随着人们对自然界的生物系统和行为特征的仔细观察与研究，通过将生物系统的行为抽象出数学模型，发明了通用性能极强的仿生

智能优化算法，此类算法在求解大规模高维复杂优化问题时表现出强大的生命力。从实际应用的角度进行分析，智能优化算法对目标函数的连续性、可微性、可导性等解析性质都没有要求，甚至无需函数表达式，同时对算法的初始点选择也无要求，对具体的优化问题具有十分强大的适应性，而且算法的计算速度十分迅速，能够快速找到问题的最优解，这些优点使得智能优化算法自诞生之日起就引起了众多学者的广泛关注，展现出蓬勃发展的强劲势头，未来必将得到更深入的研究和更广泛的应用。

智能优化算法主要包括受达尔文进化论思想产生的遗传算法，模拟蚂蚁群体觅食行为的人工蚁群算法、模拟鸟类觅食行为的粒子群算法，受数学中差分思想产生的差分进化算法，此外还有鱼群算法、布谷鸟算法等等。而人工蜂群算法也是此类算法，是2005年由Karaboga通过模拟蜜蜂采蜜行为提出来的。目前，人工蜂群算法是智能优化领域表现特别突出的群体智能优化算法之一。

人工蜂群算法自诞生之日起，由于其算法简单，易于实现，并且收敛速度快引起众多学者的广泛关注。虽然人工蜂群算法与遗传算法类似，都是模拟生物种群行为的智能优化算法，但又有所不同，人工蜂群算法是通过群体的合作机制来产生最优解，这与通过群体竞争机制产生最优解的遗传算法不同。人工蜂群算法现在已广泛应用于函数优化问题、人工神经网络训练、无人机路径选择等问题中，但就是这样一种表现十分出色的智能优化算法，其算法理论基础与应用范围还有待人们进一步研究。通过相关文献分析，人工蜂群算法现在已成为一种新兴的智能优化算法，算法的改进与理论分析以及算法的应用范围成为新的研究方向和热点。因此，对人工蜂群算法的理论分析与应用研究是十分重要的研究内容。

1.2 国内外研究现状

2005年土耳其学者Karaboga在其技术报告中提出人工蜂群算法后^[1]，2006年Basturk和Karaboga首次在国际会议上对人工蜂群算法做了介绍^[2]，2007年二人又将其研究成果首次发表在学术期刊上，详细阐述了人工蜂群算法，并与遗传算法、粒子群算法等对比，评价了人工蜂群算法的优化性能^[3]。2008年，Karaboga和Basturk对人工蜂群算法的优化性能进行了详细的比较研究，进一步说明人工蜂群算法优良的性能^[4]。2009年关于人工蜂群算法研究的网页（<http://mf.erciyes.edu.tr/abc>）建成，内容十分丰富，有使用各种程序语言编写的源代码，也有许多关于人工蜂群算法改进和应用的文章。人工蜂群算法简单且易于实现，能够直接解决优化问题，并且能够以很低的计算成本，有效的求解优化问题。因此，在该算法提出之后，很多学者展开了对人工蜂群算法的研究。这些研究主要集中在以下三个方面：人工蜂群算法的对比与改进研究；混合人工蜂群算法的研究；人工蜂群算法的应用

研究。下面将从这三个方面阐述人工蜂群算法的国内外研究现状。

1.2.1 人工蜂群算法的比较与改进

人工蜂群算法最早是用来求解函数优化问题的，通过benchmark测试函数与遗传算法、粒子群算法、差分进化算法和人工蚁群算法等进行对比研究，说明该算法优良的优化性能。Karaboga和Basturk在多变量函数优化问题^[3]和高维函数优化问题中^[4]，对比了人工蜂群算法、遗传算法和粒子群算法等的实验结果；Karaboga和Akay利用大规模测试函数实验对比了人工蜂群算法、粒子群算法、差分进化算法和进化策略的算法性能^[5]；Mala等通过一系列优化问题对比人工蜂群算法和人工蚁群算法，得出人工蜂群算法比人工蚁群算法更具有优势的结论^[6]；Krishnanand等对比了包括人工蜂群算法在内的5种仿生智能优化算法的性能^[7]；Karaboga和Akay在数值优化问题中对比了采用和声搜索的人工蜂群算法和蜂群算法的性能^[8]；Li等通过8个测试函数对比实验比较了人工蜂群算法、蜂群算法和差分进化算法的性能^[9]；Chu等通过比较分析评论了包括人工蜂群算法在内的群智能算法^[10]；Ruiz-Vanoye和Daz-Parra通过测试函数问题研究包括人工蜂群算法在内的启发式算法与人工生命算法之间的相似性^[11]；Mohammed和El-Abd对包括人工蜂群算法在内的进化算法的觅食行为进行了比较研究^[12]。

对于人工蜂群算法的改进研究，可以从以下几个方面来进行描述：

(1) 设计新的搜索策略。一些学者通过研究人工蜂群算法的控制参数和搜索策略对算法寻优能力的影响提出一些新的搜索策略，进而改进算法的优化性能。Bao和Zeng比较研究了人工蜂群算法在几种不同选择机制下的寻优效果^[13]。Guo等对三个搜索阶段给出新的搜索策略提出全局人工蜂群算法，该方法具有较高的收敛速度，并且能获得高质量的解^[14]。Alam等在人工蜂群算法的变异阶段设计一种新的自适应机制，并将算法与其他群智能优化算法在多模复杂benchmark测试函数上进行了比较研究^[15]。Diwold等通过使用新的人工蜂更新策略提出两种人工蜂群算法的变形^[16]。Zou等根据Von Neumann拓扑结构给出一种新的人工蜂群算法^[17]。

(2) 推广到约束优化问题。原始人工蜂群算法的提出是用于求解无约束优化问题的，此后学者们将其推广到约束优化问题上进行研究。Karaboga和Basturk将人工蜂群算法推广到求解约束优化问题，并将算法应用于求解一系列约束优化问题^[18]。Stanarevic等对约束优化问题提出一种修正人工蜂群算法，该算法通过引入一个“smart bee”来记忆食物源的位置和质量^[19]。Karaboga和Akay给出修正人工蜂群算法，并与其他算法对约束优化问题进行测试比较^[20]。Brajevic等研究了一种新的约束优化问题的改进人工蜂群算法^[21]。

(3) 并行算法设计。当问题规模增加时，增加种群数量可以增大找到最优解

的可能。因此，有学者试图给出并行人工蜂群算法。Subotic等通过区分邻域对种群进行划分，并在种群间以不同形式进行信息交互，设计出一种并行人工蜂群算法^[22]。Parpinelli等研究了并行人工蜂群算法，并比较了三种并行模型：主从模型、带有迁移的多蜂巢模型和混合分层模型^[23]。Luo等描述了并行人工蜂群算法的一种交流策略，交流策略中设计了信息交流环节，让代理个体在子种群间进行信息交流^[24]。Banharnsakun等介绍了一种并行人工蜂群算法，该算法将整个蜂群分解为几个子群，每个子群在每个处理节点进行局部搜索，每个子群的局部最优解将在处理节点进行交换^[25]。

(4) 结合进化优化算法策略的算法改进。对于人工蜂群算法的改进，很多学者考虑将进化优化算法中的相关概念引入人工蜂群算法中，提升算法的优化性能。Liu和Cai对原始人工蜂群算法提出一种变形算法——人工蜂群规划，算法通过采用随机分布、二进制超变异算子和新的交叉算子，提高原始算法的优化性能^[26]。受到粒子群优化算法的启发，Zhu和Kwong提出一种gbest引导的人工蜂群算法，该算法将当前全局最优解信息引入算法的搜索方程，提高算法的收敛速度^[27]。Gao和Liu提出一种受差分进化算法启发的改进人工蜂群算法，该算法在求解函数优化问题时体现出较好的优化性能^[28]。Li等学者通过引入内部权重和加速参数来改进算法的搜索过程，给出一种改进人工蜂群算法^[29]。受差分进化算法的启发，Gao和Liu通过让人工蜂仅在当前最优解周围搜索提高算法的开发能力，提出一种改进人工蜂群算法^[30]。

(5) 引入混沌理论的算法改进。为了提高算法的优化性能，有学者将混沌理论引入算法中，对算法进行改进。Lin等基于混沌序列和情绪心理因子提出情绪混沌人工蜂群算法^[31]。Alatas利用混沌投影对参数进行自适应选择，进而防止人工蜂群算法陷入局部最优，同时提高算法的收敛速度^[32]。为了提高算法的收敛速度并防止算法陷入局部最优，Wu和Fan通过引入混沌搜索方法提出一种混沌人工蜂群算法^[33]。

(6) 推广到多目标优化问题。目标函数多于一个的最优化问题在很多领域中都很常见，这类问题通常称为多目标优化问题。有学者将人工蜂群算法推广到多目标优化问题求解上。Hedaytzadeh等将原始人工蜂群算法用于求解多目标优化问题，算法利用基于梯度的方法来保持Pareto前沿的分布^[34]。Zou等基于人工蜂群算法，结合Pareto支配概念确定人工蜂飞行方向，给出一种新的多目标人工蜂群算法^[35]。

(7) 其他算法改进。Quan和Shi提出一种改进人工蜂群算法，算法基于巴拿赫空间的不动点理论和压缩映射原理设计一种新的搜索迭代算子^[36]。Kang等提出一种Rosenbrock人工蜂群算法，算法结合Rosenbrock旋转方向，用于求解精确函数优化问题^[37]。Liu等针对数值优化问题提出一种新的人工蜂群算法，该算法在搜索

中采用全局和局部最优引导增强算法寻优能力^[38].

1.2.2 混合人工蜂群算法的研究

为了增强人工蜂群算法的优化性能，有学者将该算法与其他传统优化方法和进化优化算法进行混合，这类算法称为混合人工蜂群算法。Kang等将Nelder-Mead单纯形法与人工蜂群算法混合，提出混合单纯形人工蜂群算法，算法提高了计算中的搜索效率^[39]。Duan等将量子进化算法与人工蜂群算法进行了混合，并对混合算法进行了改进应用于连续优化问题^[40]。Kang等提出一种新的混合Hooke Jeeves人工蜂群算法，算法的搜索模式基于Hooke Jeeves搜索模式和人工蜂群算法的搜索模式的混合^[41]。Zhao等描述了一种新的混合群智能方法，该方法基于遗传算法并行计算的思想的优点和人工蜂群算法自我更新的优点进行混合^[42]。El-Abd将人工蜂群算法与粒子群优化算法混合，算法中粒子群优化算法通过人工蜂群算法部分增强粒子的个体竞争力^[43]。Zhong等提出一种混合人工蜂群算法，算法中将细菌觅食优化算法中的趋药性行为植入雇佣蜂和跟随蜂的食物源开发过程中^[44]。Kang等提出一种新的混合Hooke Jeeves人工蜂群算法，算法搜索模式基于人工蜂群算法的搜索模式和Hooke Jeeves搜索模式的混合，并且研究了怎样进行策略混合才能增强人工蜂群算法的优化性能^[45]。Bao和Zeng给出一种新的双种群微分人工蜂群算法，算法结合差分进化算法来提高人工蜂群算法的优化能力^[46]。Li等结合粒子群和人工蜂群特点，提出一种混合搜索算法用于求解高维优化问题^[47]。

1.2.3 人工蜂群算法的应用研究

人工蜂群算法的提出成功应用于处理连续函数空间的单目标优化问题，通过比较研究发现该算法具有参数少、优化速度快等多方面优点，因此使得该算法在其他领域的应用得以拓展。例如：整数规划问题^[48]、支持向量机的参数确定问题^[49]、入侵检测系统的最优参数选择^[49]等等。随着对该算法研究的深入，其应用的范围也越来越广。下面就从几个主要方面对人工蜂群算法的应用进行介绍。

(1) 神经网络训练。人工蜂群算法的应用领域非常多，其中最有意义的一个应用领域是神经网络训练。2007年，Karaboga等利用人工蜂群算法进行反馈神经网络训练，寻找最优权重集合^[50]。此后，Karaboga和Ozturk利用人工蜂群算法进行反馈神经网络训练，来对不同的数据集进行分类^[51]。Garro等基于合成方法提出一种人工蜂群算法用于神经网络训练，该方法极大化准确率和极小化人工神经网络节点数量^[52]。

(2) 电气工程领域的应用。一些学者应用人工蜂群算法求解电气工程中的优化问题。为了解决配电系统最小损失问题，Rao等应用人工蜂群算法给出一种新的方法来确定分段开关操控^[53]。针对辐射式电网最优分布式电源分配问题，

Abu-Mouti和El-Hawary基于人工蜂群算法提出一种优先度排序约束搜索技术^[54]. Ozturk等应用人工蜂群算法解决了无功优化问题，对算法性能在总线系统上进行了测试，并将测试结果与Pareto进化算法进行了比较^[55]. Cobanli等基于人工蜂群算法提出一种电力系统有功网损最小化方法^[56]. Ayan和Kılıç利用带有混沌策略的人工蜂群算法求解最优无功潮流问题，这一问题通常被构造成复杂的非线性优化问题^[57]. Özyon等利用权重方法将环境经济电力调配这一多目标优化问题转化为单目标优化问题，并利用人工蜂群算法进行求解^[58]. Bijami等利用人工蜂群算法研究两个电力系统稳定器间同步联合调谐和阻尼电力系统间的振荡^[59]. Dutta等应用人工蜂群算法求解压电柔性结构最优控制问题，其中压电柔性结构带有压电致动器、传感器、最优传感器位置和反馈增益，这一问题可以通过最大化反馈控制系统能量耗散得到^[60].

(3) 机械和土木工程领域的应用. Rao和Pawar应用人工蜂群算法研究了多路径铣削过程参数优化问题，并将研究结果与粒子群优化算法和模拟退火算法进行比较^[61]. Gomez-Iglesias等设计了一种修正人工蜂群算法用于优化核聚变装置中的等离子体平衡问题，以及网格计算环境适配问题^[62]. Yao等利用人工蜂群算法进行了地铁线路设计问题研究，该问题的目的是使得地铁线路覆盖的人流量最大^[63]. Hadidi等利用人工蜂群算法研究了受压情况下平面和空间桁架的最优结构、位移及弯曲情况^[64]. 基于人工蜂群算法，Hetmaniok提出解决反向热传导问题的一种数值方法^[65]. Mandal等将模糊集理论与人工蜂群算法相结合对支持向量机进行训练，进而应用研究了管线泄漏勘测问题^[66].

(4) 数据挖掘. 另一个人工蜂群算法应用较为广泛的领域是数据挖掘领域，特别是聚类问题、特征提取和模式识别是应用研究的主要方面. Karaboga和Ozturk^[67]将人工蜂群算法的应用推广到模糊聚类中，并通过测试说明该算法对模糊聚类问题是可行有效的方法. 此后，又基于人工蜂群算法，提出一种新的聚类方法，研究通过UCI机器学习库中13个典型测试数据集对该方法性能进行测试^[68]. Zhang等提出一种人工蜂群聚类算法，该算法使用Deb法则确定候选解的搜索方向^[69]. Hsieh和Yeh对机器学习提出一个新概念，对分类问题将网格模式与最小二乘支持向量机结合（GS-LSSVM），其中人工蜂群算法用来对最小二乘支持向量机学习参数进行优化^[70]. Babu和Rao利用人工蜂群算法设计出一种新的“大蒜专家咨询系统”，该系统可以用来区分大蒜生产中的疾病和疾病管理，并对农户给出合理建议，从而得到标准产量^[71]. Shukran等利用人工蜂群算法作为数据挖掘特别是分类的新工具，研究表明人工蜂群算法不仅优于其他进化算法，同时优于其他工业标准算法，如：PART、SOM、kNN等^[72].

(5) 无线传感器网络. Udgata等利用人工蜂群算法解决传感器布点问题^[73]. Mini等则将人工蜂群算法应用于三维地形上的传感器布点问题^[74]. Ozturk等基

于人工蜂群算法给出一种动态布点方法，该方法通过增大网络覆盖区域提高静止和移动的传感器网络效果^[75]。

(6) 图像处理。人工蜂群算法还被应用于图像处理领域中，应用研究主要集中在图像边缘增强^[76]、目标识别^[77]、手写数字识别^[78]、图像分割^[79-82]、飞行器目标识别^[83]、合成孔径雷达图像快速分割^[84]、图像压缩^[85, 86]等方面。

(7) 组合优化问题。尽管原始人工蜂群算法的提出是用于求解函数优化问题的，但是随着算法研究的深入，该算法也被拓展到求解离散问题和组合优化问题中。研究较为集中的问题是车间调度问题^[87-91]、供应链问题^[92]、0-1背包问题^[93]、最小生成树问题^[94]等等。在这些问题中，很多都是NP-hard问题，难于用传统优化方法求解，而人工蜂群算法可以求解这类问题，得到较好的应用效果。

(8) 电子工业、软件及控制工程领域的应用。在电子工业领域，人工蜂群算法用于进行空分多址正交频分复用系统Turbo编码调制的多用户检测^[95]；非均匀滤波器组传输多路复用器设计^[96]；弹性分组环有效负载平衡问题^[97]等等。在软件应用方面，研究主要集中在自动软件重构问题^[98]、软件测试优化技术^[99]、面向对象的软件系统自动维护设计^[100]等方面。在控制工程领域，人工蜂群算法主要用于求解PID控制设计^[101]、离散混沌系统混沌控制^[102]、非线性时滞混沌系统参数确定^[103]等问题中。

(9) 蛋白质结构优化问题。人工蜂群算法还被应用于蛋白质结构优化问题。Bahamish等基于人工蜂群算法给出一种新方法用来预测蛋白质三级结构，利用人工蜂群算法搜索蛋白质构成空间，从而找出最低自由能构造^[104]。Lin和Lee提出一种修正人工蜂群算法来确定序列中的蛋白质结构^[105]。Bahamish和Abdullah利用人工蜂群算法预测C肽和核糖核酸酶A的三级结构，算法通过搜索构造空间确定最低自由能构造^[106]。Li等针对蛋白质结构优化问题提出一种均衡进化人工蜂群算法，用于确定蛋白质结构模型^[107]。

1.2.4 国内研究现状

在国内人工蜂群算法的研究也取得了许多创新性成果。胡玉容等提出了一种基于渐变与突变机制的反向人工蜂群算法，提高了算法的收敛速度^[108]，孙晓雅和林焰提出了改进的人工蜂群算法用来求解任务指派问题，所提算法比原始算法和粒子群算法在求解精度和收敛速度上明显占优^[109]。毕晓君和王艳娇针对人工蜂群算法易于陷入局部最优的缺点，通过引入自由搜索算法和OBL策略，提出了改进的人工蜂群算法，benchmark测试函数实验表明所提算法能精确地搜索到各个峰值点^[110]。汪继文等利用改进的人工蜂群算法求解非线性方程组^[111]。康飞等提出了一种用于材料参数反演分析的混合单纯形人工蜂群算法，成功应用于混凝土重力坝动力材料参数识别问题^[112]。高卫峰等提出一种混合人工蜂群算法，通过调频合成

器参数优化问题测试表明所提算法是快速有效的，并将其应用于线性系统逼近问题^[113]。鲁建夏等将人工蜂群算法应用于混流汽车装配线排序问题^[114]。王文亮等提出了一种基于图形处理器加速的并行人工蜂群算法，提高了算法的运算速度^[115]。刘三阳等提出了一种基于随机动态局部搜索的改进人工蜂群算法，实验结果表明算法的有效性和精确性^[116]。王文全等将人工蜂群算法应用于大型舰船主尺度优化问题^[117]。葛宇等基于极值优化策略高效率的寻优机制提出了改进的人工蜂群算法^[118]。马文强等将人工蜂群算法应用于炼钢连铸生产调度问题^[119]。李鑫滨等将算法应用于认知无线电频谱分配问题^[120]。张新明等提出了一种基于排名映射概率的混沌人工蜂群算法^[121]。李牧东等将算法应用于无线传感器网络无需测距依赖的DV-Hop定位算法中的节点定位问题^[122]。胡中华等将算法应用于TSP问题^[123]。张志成等将算法应用于阵列信号波达方向和多普勒频率的联合估计问题^[124]。刘路和王太勇将人工蜂群算法用于支持向量机参数优化，并将其应用于电机轴承的智能故障诊断^[125]。毕晓君和王艳娇提出了一种加速收敛的人工蜂群算法，该算法对于测试函数问题几乎都能找到全局最优解，明显优于原始算法^[126]。陈久梅和曾波提出了用于求解两级定位-路径问题的路径重连变邻域搜索人工蜂群算法^[127]。柳寅和马良提出了一种基于模糊化处理和蜂群寻优特点的模糊人工蜂群算法求解旅行商问题^[128]。王翔和郑建国提出了一种基于多成员机制的人工蜂群算法求解约束函数优化问题^[129]。高卫峰等针对人工蜂群算法搜索方程探索能力强，开发能力弱的缺点提出一种改进算法用来求解复杂数值优化问题^[130]。林小军和叶东毅受文化算法双层进化空间的启发，提出了利用信度空间中的规范知识引导搜索区域的改进人工蜂群算法。该算法平衡了探索与开发能力^[131]。高明松等将算法应用于辨识水下潜器参数问题^[132]。毕晓君和王艳娇提出了基于小生境的人工蜂群算法用来求解多峰函数优化问题^[133]。张培文等提出混合人工蜂群算法求解以最大完工时间为目标的有限缓冲区流水车间调度问题^[134]。毕晓君和王艳娇利用外部种群和约束多目标问题的特点提出了求解多目标问题的人工蜂群算法，改进算法使获得的Pareto最优解集分布更均匀，覆盖更广^[135]。徐晓飞等针对服务优化问题提出一种面向服务领域的人工蜂群算法范型，为求解服务优化问题提供了新的研究方法^[136]。李田来等采用分治策略进行迭代优化，提出基于分治策略的改进人工蜂群算法^[137]。丁政豪等利用混沌人工蜂群算法对耦合双梁系统的局部损伤进行识别，通过算例表明该方法能有效快速检测出损伤，具有较强的实用性^[138]。周新宇等针对人工蜂群算法中侦查蜂易丢失搜索经验，提出采用正交实验设计的方式生成新食物源，保证侦查蜂能够保留相应的搜索信息，提高算法的搜索效率^[139]。

人工蜂群算法理论和实践的研究成果还有很多，考虑到本文篇幅和内容，其他不再赘述。

人工蜂群算法相关研究取得丰硕成果不过是近几年的事。通过以上对人工蜂

群算法国内外研究现状的总结，关于人工蜂群算法的研究主要集中在算法改进和算法应用方面，对于该算法的研究仍大有发展之势。对于人工蜂群算法还可以进行更深入的研究，特别是算法的自适应控制参数和理论研究；无参数的算法研究；算法的一般理论研究；运行时间及收敛特征研究；算法在动态规划和不确定规划问题上的推广等等。这些都可以作为未来进一步研究的重点内容。

1.3 论文的主要研究内容与结构安排

本文围绕人工蜂群算法的改进、具体问题的应用、算法的理论分析三个方面进行了深入研究。在算法改进方面，针对人工蜂群算法探索能力与开发能力之间的不平衡性，提出了一种新的种群初始化方法-正交初始化方法，以维持初始种群分布的均匀性，同时给出采蜜蜂和跟随蜂新的搜索方程，本文给出的两种改进人工蜂群算法在函数优化问题中提高了算法的有效性和可行性；针对约束函数优化问题，原有的智能优化算法是把时间全部分配到一种算法或者约束处理技术上，本文提出了一种基于进化策略和人工蜂群算法投资组合的新的混合算法，把时间分配到不同算法上，通过benchmark测试函数实验表明新算法的有效性；在具体问题应用上，以非负最小二乘问题为实际应用背景，提出了二种新的改进人工蜂群算法，在具体问题上与其他智能优化算法做了对比，实验结果表明改进算法能有效求解这一实际问题，从而拓宽了算法的实际应用领域；在算法的理论分析方面，鉴于已有的算法理论分析不能确保算法在有限步内收敛到问题的全局最优解，本文首次尝试引入随机过程中的鞅理论研究人工蜂群算法的几乎必然强收敛性，证明了人工蜂群算法确保能够在有限步内收敛到全局最优解，为算法的改进和实际应用奠定了坚实的理论基础。

本文章节具体安排如下：

第一章是绪论，介绍了论文选题的意义和目的，着重阐述了人工蜂群算法的国内外研究现状，最后给出了本论文的主要研究工作和结构安排。

第二章是人工蜂群算法及收敛性分析，首先介绍了人工蜂群算法的生物学原理，随后给出了人工蜂群算法的数学模型和具体操作方法，归纳总结了人工蜂群算法的特点，最后利用鞅理论研究了人工蜂群算法的收敛性。已有的人工蜂群算法的收敛性证明都是建立在随机过程的Markov链基础上的，利用Markov链的遍历性，证明了人工蜂群算法在概率意义上是收敛的，但证明结果不能保证人工蜂群算法在有限步内收敛到全局最优解，本文尝试利用随机过程中的鞅理论，证明了人工蜂群算法的几乎必然强收敛性，即能确保在有限步内收敛到全局最优解。这一结论为人工蜂群算法的改进和拓宽实际应用领域奠定了理论基础。

第三章是全局优化问题的混合人工蜂群算法，针对人工蜂群算法优化速度较

慢的缺点，通过对人工蜂群算法的搜索方程添加随机扰动项来提高算法的收敛速度，通过benchmark测试函数和原始的人工蜂群算法对比得出，改进的人工蜂群算法能有效提高算法的收敛速度，是一种有效的算法。

第四章是函数优化问题的改进人工蜂群算法，该算法引入正交设计作为初始化的方法，同时受差分进化算法的启发，引入新的搜索机制，利用P概率来控制搜索方程，平衡了算法的探索和开发能力，通过benchmark测试函数实验，表明改进的人工蜂群算法比原始人工蜂群算法和改进的4种人工蜂群算法以及其他7种知名的智能优化算法更有效。改进的人工蜂群算法在提高收敛速度的同时，有效地维持了种群的多样性，防止算法陷入局部最优解。

第五章是非负线性最小二乘问题的有效人工蜂群算法，该算法利用正交初始化方法来代替随机初始化，提高了算法初始种群分布的均匀性，同时受差分进化算法和粒子群优化算法的启发，引入了两种新的搜索机制，利用p概率进行算法流程的控制，通过benchmark测试函数实验，表明提出的有效人工蜂群算法比其他智能优化算法更有效。把提出的有效人工蜂群算法应用到5个非负线性最小二乘问题上，实验结果表明，新算法比原始人工蜂群算法更有效。

第六章是非负线性最小二乘问题的混沌人工蜂群算法，针对全局优化问题和非负线性最小二乘问题，提出一种有效的混沌人工蜂群算法。为了克服人工蜂群算法的不足，在混沌人工蜂群算法种群初始化阶段利用反学习初始化方法增强种群的分布均匀度。其次，为了提高算法的优化性能，平衡算法的探索与开发能力，提出一种新的搜索机制。最后，为了增强算法的局部搜索能力，在算法中加入混沌局部搜索算子。通过对27个测试函数和5个非负线性最小二乘测试问题的实验，验证了算法的优化性能。比较分析显示，所提算法是可行有效的。

第七章是约束优化问题的算法投资组合，针对约束函数优化问题，提出了一种基于进化策略和人工蜂群算法投资组合的混合人工蜂群算法。原有的处理约束函数优化问题的方法大部分是集中在如何处理约束条件，尽管对于不同的约束条件提出了不同的处理技术，但都是把计算时间全部用在一种算法和处理技术上，本文提出的算法把不同的时间分别分配给不同的算法和约束处理技术，降低了算法运算不成功的风险，同时，算法中设计了一个迁移阶段，用来实现两种算法之间的信息交流，通过benchmark测试函数实验，将提出的混合算法与其他智能优化算法相比，表明提出的混合算法具有更好的优化性能。

第八章是结论与展望，概括了本文的主要工作，分析了现有研究工作的不足之处，同时对今后研究工作的主要方向进行了展望。

第二章 人工蜂群算法及收敛性分析

群智能是模拟种群自组织能力的一个重要领域。蚂蚁群、鸟群或是免疫系统都是典型的群系统。蜂巢周围的蜜蜂群是另一种典型的群智能。人工蜂群算法是基于蜜蜂群体的智能行为提出的一种优化算法。本章将对基本人工蜂群算法进行详细描述，并通过运用随机过程中的鞅理论研究人工蜂群算法的几乎必然强收敛性，证明了人工蜂群算法确保能以概率1在有限步内达到全局最优。所证结论为拓宽算法的应用范围奠定了理论基础，为算法的改进及收敛性研究提供了新的理论分析工具。

2.1 引言

近年来，群智能成为很多科学研究领域关注的研究方向。Bonabeau将群智能定义为“受昆虫群体和其他动物群体启发，试图设计出算法或分布式问题求解方法”^[1]。Bonabeau等主要研究昆虫群体特征，比如白蚁群、蜜蜂群、马蜂群以及其他不同种类的群蚁。然而，种群是一个一般性的名词，指任何限定个体的集合。蜂巢周围的蜜蜂群就是一种常见的种群，不过这种说法可以很容易的推广到任何其他具有相似体系的系统。蚂蚁群可以认为是一个由蚂蚁为个体的种群。同样的，一群鸟是一个鸟群；免疫系统是细胞和分子构成的种群；由人构成的种群是人群。粒子群优化算法模拟的是鸟群或鱼群的社会行为。

两个基本概念自组织和分工是群智能行为必要的两个特点，比如分布式问题求解系统具有自组织性和环境适应性。自组织性可以被定义为一系列动态机制，使得通过低水平个体间的交互行为构造一个全局系统。这些机制建立起了个体与系统间交互行为的基础准则。准则确保交互行为的执行基于单纯的局部信息而与全局结构无关。Bonabeau等对自组织性给出四种基本特征：正反馈、负反馈、波动和多元交互^[1]。正反馈是一种简单的“经验法则”行为，用来促进结构生成、补充和强化，比如蚂蚁的信息素播撒和跟随以及蜜蜂的舞蹈，都是正反馈机制。负反馈用来平衡正反馈，帮助建立集体模式。为了避免饱和现象，负反馈通常发生在提供觅食者、食物源耗尽或食物源竞争时。波动包括随机游走、误差、种群个体间任务的随机转变等，这些对于开发与创新极为重要。一般的，自组织性要求个体间的相似度尽可能的小，这样才能使个体间的差异得到更好的应用。

除了自组织性，种群中的分工也是群智能行为的另一个重要方面。种群中有不同的任务，这些任务被不同的个体同时完成，这种现象称为分工。通过不同个体同时协作完成任务比顺序完成任务更有效。分工还可以使种群对搜索空间中的

条件变化做出反应。人工蜂群算法作为一种群智能算法，也应该具有自组织性和分工的重要特征。下面将对人工蜂群算法的具体内容做详细的介绍。

本章主要安排如下：2.2节对蜜蜂群的行为特点进行介绍；2.3节给出人工蜂群算法的基本原理与步骤；2.4节给出人工蜂群算法的Markov链模型；2.5节是算法收敛性分析；2.6节对本章进行小结。

2.2 蜂群行为

觅食选择最优模型导致了蜜蜂群体智能的出现，其中包含了三个重要因素：食物源、雇佣蜂和非雇佣蜂；以及两种行为模式：对食物源的开采和对食物源的放弃。

(1) 食物源：一个食物源的质量决定于很多因素，比如与巢穴的距离，实物量或能量聚集程度，以及采集难度。为了简洁起见，一个食物源的质量仅用单一的蜜量来确定^[1]。

(2) 雇佣蜂：雇佣蜂是与固定的食物源对应的蜜蜂，经常对该食物源进行采蜜。它们将所对应的食物源信息以一定的概率进行分享，这些信息包括食物源距离蜂巢的距离和方位，还包括食物源的质量情况。

(3) 非雇佣蜂：非雇佣蜂持续对食物源进行开采。非雇佣蜂包括两种类型：侦查蜂和跟随蜂。其中，侦查蜂搜索蜂巢附近的新食物源；跟随蜂在蜂巢中等待，并通过雇佣蜂的共享信息建立一个食物源。蜂群中侦查蜂所占比例大约在5-10%^[1]。

蜜蜂间信息交换是形成集体智慧的重要表现。通过对整个蜂巢进行研究，发现蜂巢可以被划分为几个重要部分。这些划分区域中，最重要的是蜜蜂进行信息交换的舞蹈区。与食物源质量相关的信息交流就发生在舞蹈区。这种舞蹈称为摇摆舞。

由于所有当前蜜量丰富的食物源信息都会在舞蹈区传递给一个跟随蜂，它会看到很多的舞蹈信息，同时决定去对自己最有利的食物源进行食物采集。这时就可能由于对某些食物源信息的循环共享，而出现跟随蜂选择其中某个食物源进行开采。雇佣蜂通过一个关于食物源质量的概率比率共享信息，并通过摇摆舞的持续时间长短反映。因此，对跟随蜂的召集与食物源的质量成比例。

为了了解蜜蜂觅食的基本行为特点，通过图2.1直观给出蜜蜂觅食的具体情形。假设有两处被发现的食物源：A和B。初始时，一个潜在的觅食者将会从非雇佣蜂开始。这只蜜蜂对于蜂巢周围的食物源情况并不了解，那么存在两种可能选择：

(1) 这只蜜蜂可以成为侦查蜂，同时根据一些特定的活动区间或可能的外部线索搜索蜂巢周围自然地发现一个食物源；

(2) 这只蜜蜂可以通过观察摇摆舞被招募为跟随蜂，开始搜索食物源（图2.1中R）。

确定食物源位置后，蜜蜂通过自己的方式对食物源的信息进行记忆，并立刻开始对食物源进行开采。因此，这只蜜蜂将变为一只雇佣蜂。采蜜蜂采集食物源的花蜜，返回蜂巢并将花蜜放入食物储藏间。放下花蜜后，这只蜜蜂有以下三种选择：

(1) 这只蜜蜂在放弃食物源后变为一只自由的跟随者（UF）；

(2) 这只蜜蜂在再次去同一食物源采蜜之前，在舞蹈区跳舞，招募蜂巢中的其他蜜蜂一起去该食物源采蜜（EF1）；

(3) 这只蜜蜂继续采蜜而不召集其他同伴（EF2）。

注意到不是所有蜜蜂同时开始采蜜。有实验证明，新蜜蜂以一定比例开始采蜜，这一比例与所有蜜蜂数量和当前采蜜蜂数量之差成比例。

蜜蜂群体的自组织性体现在以下几个方面：

(1) 正反馈：当食物源的花蜜量增加时，跟随蜂对这些食物源的采蜜次数也会增加；

(2) 负反馈：被放弃的食物源不再被蜜蜂开采；

(3) 波动：侦查蜂采用随机搜索的方式发现新的食物源；

(4) 多元交互：蜜蜂通过在蜂巢中舞蹈区的摇摆舞共享食物源的信息。

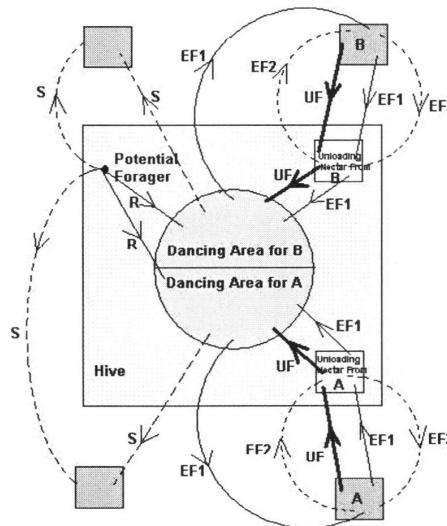


图 2.1 蜜蜂采蜜行为图

2.3 人工蜂群算法原理与步骤

在实际的蜜蜂群体中，一些任务是由特定的蜜蜂来完成的。这些特定的蜜蜂

通过有效的分工和自组织试图使蜂巢储藏的花蜜量最大。近期由Karaboga于2005年对求解实值参数优化问题提出的人工蜂群算法就是模拟蜜蜂采蜜行为的算法^[1]。

在人工蜂群算法中，人工蜂种群由三种不同的蜜蜂构成：雇佣蜂，跟随蜂和侦查蜂。雇佣蜂负责与对应食物源进行采蜜，并将食物源信息带回蜂巢共享；跟随蜂在蜂巢舞蹈区等候雇佣蜂传递信息，并对食物源进行选择；侦查蜂进行随机食物源搜索。人工蜂群算法中，种群一半是雇佣蜂，另一半是跟随蜂。每个食物源与一只雇佣蜂对应，也就是说，雇佣蜂的数量与蜂巢周围食物源的数量相同。当某个食物源被放弃后，该食物源对应的雇佣蜂或跟随蜂就变成侦查蜂。因而，算法的主要步骤如下：

初始化。

REPEAT.

 进行雇佣蜂搜索；

 进行跟随蜂搜索；

 侦查蜂随机搜索新食物源。

UNTIL（终止条件满足）。

人工蜂群算法中，每个循环包含三个步骤：第一步，雇佣蜂开采食物源，并对食物源蜜量进行估计；第二步，跟随蜂根据雇佣蜂的共享信息选择食物源采蜜，并确定食物源蜜量；第三步，确定侦查蜂，并让侦查蜂搜索新食物源。初始化阶段，人工蜂随机选取一批食物源，并对食物源蜜量进行估计。然后，将这批食物源信息带回蜂巢，与蜂巢中舞蹈区等待的人工蜂共享。第二阶段，信息共享过后，由于食物源位置已经存储在对应雇佣蜂记忆中，每个雇佣蜂将按初始路径继续对对应食物源进行食物采集，然后根据当前食物源附近的视觉信息探索一个新食物源。第三阶段，跟随蜂根据雇佣蜂在舞蹈区对食物源蜜量信息的传递选择一个食物源进行采蜜。随着食物源蜜量增加，跟随蜂对该食物源选择的概率也将会增大。当跟随蜂到达选择的食物源后，跟随蜂根据当前食物源附近的视觉信息探索一个新食物源。视觉信息是基于食物源位置给出的。当某个食物源被人工蜂放弃，侦查蜂就随机确定一个新食物源，用来替代被放弃食物源。在人工蜂群算法模型中，每一次循环至多产生一只侦查蜂，进行新食物源的随机搜索，并且模型中雇佣蜂的数量与跟随蜂的数量相同。

人工蜂群算法中，食物源的位置代表最优化问题的可行解，食物源的蜜量表示对应解的质量。种群中雇佣蜂的数量等于跟随蜂的数量，并与可行解的数量相等。第一步中，人工蜂群算法随机生成包含 SN 个解（食物源位置）的初始种群 P ($G=0$)，这里 SN 表示种群数量。每个解（食物源） x_i ($i=1,2,\dots,SN$) 是一个 D 维向量。这里， D 是优化问题的维数。初始化后，种群要重复雇佣蜂、跟随蜂、侦查蜂搜索的循环过程。为了找到新的食物源，人工雇佣蜂或跟随蜂将在其记忆

中对食物源位置进行修正，同时对新食物源的蜜量进行估测。对于真实的蜜蜂来说，发现新食物源是根据原来食物源附近其他食物源位置的视觉信息比较进行的。然而，在人工蜂群算法模型中，人工蜂不进行这样的比较。人工蜂随机选择一个食物源，并通过式(2-2)对记忆中已有的食物源位置进行修正。若新修正的食物源的蜜量高于原来的食源，则将新食物源位置进行记忆，删除原来食物源位置信息；否则，保持对原来食物源位置信息的记忆。当所有雇佣蜂完成了搜索过程，它们将食物源蜜量信息和位置信息在舞蹈区与跟随蜂共享。跟随蜂对所有雇佣蜂带回的食物源蜜量信息进行评估，然后以一个与蜜量相关的概率选择食物源。此后，跟随蜂与雇佣蜂行为相同，对记忆中的食物源位置进行修正，检查候选食物源的蜜量。若新修正的食物源的蜜量高于原来的食源，则将新食物源位置进行记忆，删除原来食物源位置信息；否则，保持对原来食物源位置信息的记忆。

跟随蜂对食物源的选择根据与食物源蜜量相关的概率来进行，概率 p_i 的表达式如下式(2-1)：

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \quad (2-1)$$

其中， fit_i 是第 i 个解的适应值，它与食物源的蜜量成比例； SN 是食物源数量，也等于雇佣蜂数量。通过这样的方式，雇佣蜂与跟随蜂进行了信息交换。

为了对原来的食源产生一个新的食物源位置，人工蜂群算法利用式(2-2)进行食物源位置修正：

$$v_{ij} = x_{ij} + \phi_j(x_{kj} - x_{ij}), \quad (2-2)$$

其中，参数 k 是 $\{1, 2, \dots, SN\}$ 中随机选择的不同于 i 的正整数； $j \in \{1, 2, \dots, D\}$ 是随机抽取的下标； $\phi_j \in [-1, 1]$ 是均匀分布的随机数。式(2-2)控制新食物源的产生在 x_{ij} 的周围，同时其修正表示人工蜂通过视觉信息比较获得新食物源。式(2-2)表明当 x_{ij} 和 x_{kj} 间的差距越来越小时，对位置 x_{ij} 的扰动也越来越小。因此，当搜索过程中，越来越接近于最优解时，搜索步长就会相应的越来越小。当产生修正位置 v_{ij} 后，估计其适应值，并与 x_{ij} 进行比较。若新食物源优于或等于原来的食源，就用新食物源替代原来的食源，并进行记忆；否则，保留原来的食源。也就是说，在新旧食物源间应用贪婪选择机制作为选择算子。

若搜索中变量超出定义域范围，则给变量定义一个可行值。在人工蜂群算法

中, 当变量超出定义域范围时, 将其设定为定义域的边界值.

当食物源被放弃时, 侦查蜂随机搜索一个新食物源替代被放弃食物源. 在人工蜂群算法中, 当一个位置不能被更新超过限制 $limit$ 时, 该位置被人工蜂放弃. 人工蜂群算法的具体流程图如图2.2所示.

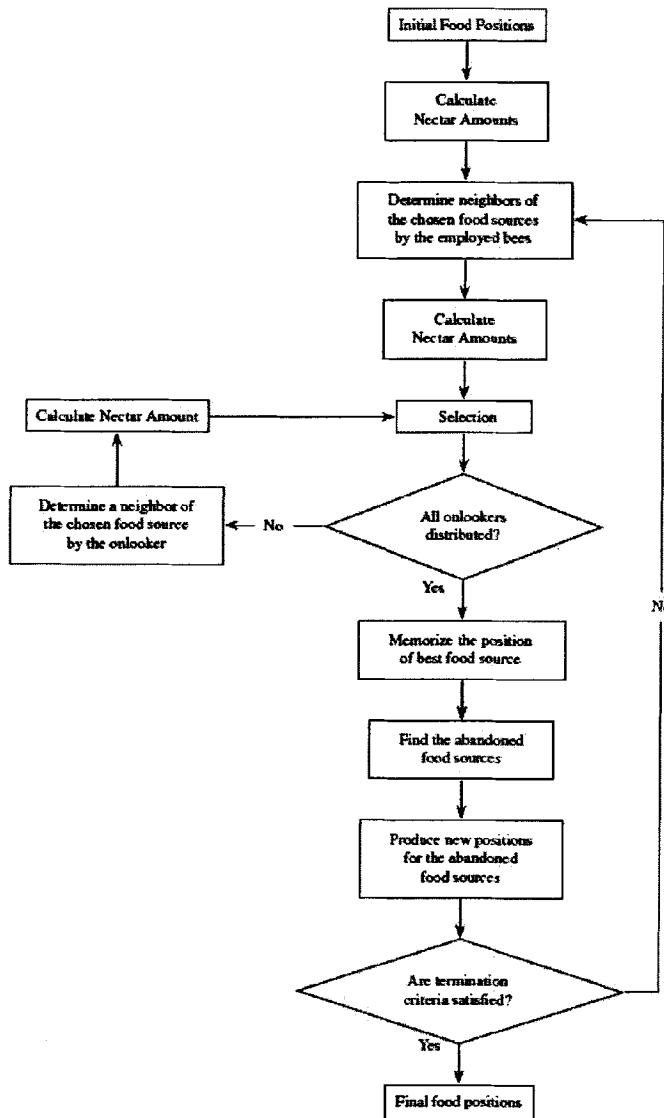


图 2.2 人工蜂群算法流程图

人工蜂群算法实际上进行了四种不同的选择过程:

- (1) 全局选择过程: 人工跟随蜂通过式(2-1)对可行域内的食物源进行选择;
- (2) 局部选择过程: 人工雇佣蜂和人工跟随蜂通过式(2-2)根据局部信息对记忆中的食物源附近进行新食物源搜索;

(3) 贪婪选择过程：对于所有人工蜂来说，若新修正的食物源的蜜量高于原来的食物源，则将新食物源位置进行记忆，删除原来食物源位置信息；否则，保持对原来食物源位置信息的记忆；

(4) 随机选择过程：由侦查蜂进行的随机食物源选取。

通过上面的具体描述，可以明确看到，人工蜂群算法有三个控制参数：食物源数量 SN 、更新上限 $limit$ 和最大循环代数（MNC）。

对于一个鲁棒搜索过程，开发过程和勘探过程都必须存在。人工蜂群算法中，雇佣蜂和跟随蜂进行食物源的开发过程，而侦查蜂则控制算法中对食物源的勘探过程，因此，保证了算法的鲁棒性。综合以上的分析过程，人工蜂群算法的具体搜索步骤如下所示。

Algorithm: 人工蜂群算法

利用随机初始化方法给出初始种群，并评价种群质量，设定参数 $trail_i = 0, (i = 1, 2, \dots, SN)$, $cycle = 1$.

Repeat

Step 1: 对雇佣蜂根据式(2-2)搜索新蜜源，并估计蜜源质量。

Step 2: 利用贪婪选择策略在新蜜源和旧蜜源中选择较好的一个。

Step 3: 如果蜜源没有更新，则 $trail_i = trail_i + 1$ ，否则 $trail_i = 0$ 。

Step 4: 利用轮盘赌选择，为跟随蜂选择一个蜜源。

Step 5: 对跟随蜂根据式(2-2)搜索新蜜源，并估计蜜源质量。

Step 6: 利用贪婪选择策略在新蜜源和旧蜜源中选择较好的一个。

Step 7: 如果蜜源没有更新，则 $trail_i = trail_i + 1$ ，否则 $trail_i = 0$ 。

Step 8: 若 $\max(trail_i) > limit$ ，随机生成一个新蜜源代替旧蜜源。

记忆当前最优解。

$cycle = cycle + 1$ 。

Until ($cycle = Maximum\ Cycle\ Number$)

以后若无特殊说明，都按优化最小值问题进行研究。

2.4 人工蜂群算法的 Markov 链模型

首先给出一些数学描述和定义，来阐述 ABC 算法的 Markov 链模型。

定义 2.1: 设 X 表示一个集合，其序数为 $|X|$ ，依人工蜂群算法的术语，包含有限个元素的集合 S 中的元素称之为个体，用字母 i, j, k, \dots 表示，设 m 为正整数，则所考虑的 Markov 链状态空间取为 $S = s \times s \times \dots \times s$ ，其元素被称为种群，用字母 x, y, z, \dots 表示，即：

$$S = \{(x_1, x_2, \dots, x_m) | x_k \in s, 1 \leq k \leq m\},$$

其中, m 称为种群规模.

定义 2.2: 称 E 上定义的任何非负实值函数 f 是一个适应性函数, 则可定义全局最优适应值 $f^* = \max \{f(i) | i \in s\}$, 全局最优个体集 $B^* = \{i \in s | f(i) = f^*\}$, 每个个体都是全局最优个体的种群 $F^* = \{x \in S | x_k \in B^*, 1 \leq k \leq m\}$.

定义 2.3: 人工蜂群算法迭代过程中, 对于任意两个状态 $X_i \in s, X_j \in s$, 人工蜂由 X_i 一步转移至 X_j , 记为 $T_s(X_i) = X_j$.

定理 2.1: 人工蜂群算法迭代过程中, 人工蜂由状态 X_i 转移至状态 X_j 的一步转移概率 $p(T_s(X_i) = X_j)$ 可由式(2-3)进行计算:

$$p(T_s(X_i) = X_j) = \begin{cases} p_e(T_s(X_i) = X_j), & \text{由雇佣蜂实现} \\ p_o(T_s(X_i) = X_j), & \text{由跟随蜂实现} \\ p_s(T_s(X_i) = X_j), & \text{由侦察蜂实现} \\ p_e(T_s(X_i) = X_j) \times p_o(T_s(X_i) = X_j), & \text{由雇佣蜂和跟随蜂共同实现} \end{cases} \quad (2-3)$$

证明: 在算法迭代过程中, 不妨设种群为超空间的一组点集, 则蜜源的更新过程就等同于超空间中点集间的变换过程. 根据算法的几何性质和定义 2.3, 得出雇佣蜂从状态 X_i 转移至状态 X_j 的一步转移概率为:

$$p_e(T_s(X_i) = X_j) = \begin{cases} \frac{1}{|X_i - X_j|} p_1(x_i \rightarrow x_j), & X_j \in [X_i - (X_i - X_k), X_i + (X_i - X_k)], \\ 0, & \text{otherwise} \end{cases} \quad (2-4)$$

跟随蜂从状态 X_i 转移至状态 X_j 的一步转移概率为:

$$p_o(T_s(X_i) = X_j) = \begin{cases} \frac{1}{|X_i - X_j|} p_1(x_i \rightarrow x_j), & X_j \in [X_i - (X_i - X_k), X_i + (X_i - X_k)], \\ \text{且} \text{rand}[0,1] < p & \\ 0, & \text{otherwise} \end{cases} \quad (2-5)$$

其中：

$$p_l(x_i \rightarrow x_j) = \begin{cases} 1, & f(x_j) < f(x_i) \\ 0, & f(x_j) \geq f(x_i) \end{cases} \quad (2-6)$$

当雇佣蜂对应的蜜源不能被更新时，该蜜源所对应的可行解 X_i 将因无法继续更新而陷入局部最优，此时雇佣蜂转变为侦察蜂，随机生成新的可行解 X_j 替代 X_i 。因此，侦察蜂从状态 X_i 转移至 X_j 的一步转移概率为：

$$p_s(T_s(X_i) = X_j) = \begin{cases} \frac{1}{|X_{\max} - X_{\min}|}, & X_j \in [X_{\min}, X_{\max}] \text{ 且} \\ & \text{状态 } X_i \text{ 的迭代次数 } \text{int} > \text{limit} \\ 0, & \text{otherwise} \end{cases} \quad (2-7)$$

其中， limit 是解未被更新的最大次数。式(2-4)-(2-6)中 X 是高维向量，用绝对值表示超空间立方体的体积，加减号表示向量加减。 X_k 是可行域内随机选取的蜜源。

证毕。

ABC 算法是人工蜂群通过雇佣蜂、跟随蜂、侦察蜂不同角色之间的交流转换来实现的，所以由式(2-4)-(2-6)共同决定人工蜂的一步转移概率 $p(T_s(X_i) = X_j)$ 。

定义 2.4：在算法迭代过程中，对于人工蜂群的任意两个状态 $s_i \in S, s_j \in S$ ，从 s_i 一步转移至 s_j 记为 $T_S(s_i) = s_j$ 。则人工蜂群从 s_i 转移至 s_j 的一步转移概率为：

$$P(T_S(s_i) = s_j) = \prod_{m=1}^{SN} P(T_s(X_{im}) = X_{jm}),$$

即：人工蜂群从状态 s_i 转移至 s_j 的一步转移概率为蜂群 s_i 内所有的人工蜂状态同时变成蜂群 s_j 内所有的人工蜂状态。

2.5 收敛性分析

本节将运用鞅收敛定理来考察人工蜂群算法的几乎必然强收敛性。为了比较，将首先引进如下定义。

2.5.1 依概率收敛

定义 2.5^[14]: 如果 $\lim_{n \rightarrow \infty} p\left(\left[X^n \cap F^*\right] \neq \emptyset\right) = 1$, 称人工蜂群算法 $\{X^n\}$ 为依概率弱收敛到全局最优; 若 $\lim_{n \rightarrow \infty} p\left(\left[X^n \subset F^*\right]\right) = 1$, 则称人工蜂群算法 $\{X^n\}$ 为依概率强收敛到全局最优.

定义 2.6^[14]: 如果 $p\left(\lim_{n \rightarrow \infty} \left[X^n \cap F^* \neq \emptyset\right]\right) = 1$, 称人工蜂群算法 $\{X^n\}$ 为几乎必然弱收敛到全局最优; 若 $p\left(\lim_{n \rightarrow \infty} \left[X^n \subset F^*\right]\right) = 1$, 则称人工蜂群算法 $\{X^n\}$ 为几乎必然强收敛到全局最优.

定理 2.2: 人工蜂群算法种群状态序列 $\{S(t)|t \geq 0\}$ 是有限齐次马尔可夫链.

证明:

(1) 搜索空间对任何优化算法来说都是有限的, 人工蜂群算法中任意人工蜂群状态中的 X_i 也都是有限的. 单个个体的状态空间有限且离散. 一个种群具有 SN 个个体, SN 为有限正整数, 而空间状态 $s = (X_1, X_2, \dots, X_{SN})$ 由 SN 个个体状态组成, 所以种群状态空间 S 是有限的.

(2) 由定义 2.4 得到种群状态序列 $\{S(t)|t \geq 0\}$ 中任意两个种群的状态 $s(t-1) = (X_{1,(t-1)}, X_{2,(t-1)}, \dots, X_{SN,(t-1)})$, $s(t) = (X_{1,t}, X_{2,t}, \dots, X_{SN,t})$, 它们之间的转移概率表示为:

$$p(T_s(s(t-1)) = s(t)) = \prod_{m=1}^{SN} p(T_s(X_{(t-1),m}) = X_{t,m}).$$

由定理 2.1 可知, 蜂群内任一人工蜂的转移状态 $p(T_s(X(t-1)) = X(t))$ 仅与 $t-1$ 时刻的状态 $X(t-1)$ 、随机可行解 X_k 、随机参数 q_y 和优化问题有关, 所以一步状态转移概率 $p(T_s(s(t-1)) = s(t))$ 也仅与 $t-1$ 时刻的状态相关, 即人工蜂群状态序列 $\{S(t)|t \geq 0\}$ 具有马尔可夫性. 又状态空间为可列集, 故人工蜂群状态序列构成一个有限马尔可夫链.

(3) 由定理 2.1 可知, 一步状态转移概率 $p(T_s(X(t-1)) = X(t))$ 仅与 $t-1$ 时刻的状态 $X(t-1)$ 有关, 而与 $t-1$ 无关, 因此人工蜂群算法种群状态序列 $\{S(t)|t \geq 0\}$ 是有限齐次马尔可夫链.

证毕.

定理 2.3: 人工蜂群算法能以概率 1 收敛到全局最优.

证明: 证明过程详见参考文献[140].

2.5.2 几乎必然强收敛

根据上述讨论, 可以发现几乎必然收敛性明显强于依概率收敛性. 人工蜂群算法收敛性已有的证明结果属于依概率收敛, 是弱大数律范畴. 为了得到更强的

收敛性结论，下面将证明人工蜂群算法的几乎必然强收敛性。为了阐述这一结论，先引入如下定义。

定义 2.7^[141]: 若对 $\forall i \in C$, 有 $\sum_{j \in C} p_{ij} = 1$, 即自 C 内任意一点出发, 始终不能达到 C 外的任意状态., 称状态空间 C 为闭集。

定义 2.8^[141]: 若闭集 C 无真闭子集, 则称 C 为不可约的, 否则称 C 为可约的。

定义 2.9^[141]: 设 $\{Y_k, k \geq 0\}$ 与 $\{Z_k, k \geq 0\}$ 是两个随机过程, 称 $\{Y_k, k \geq 0\}$ 关于 $\{Z_k, k \geq 0\}$ 是一个上(或下)鞅, 如果:

- (1) $E|Y_k| < \infty$;
- (2) $E(Y_{k+1}|Z_0, Z_1, \dots, Z_k) \leq Y_k$ (或 $E(Y_{k+1}|Z_0, Z_1, \dots, Z_k) \geq Y_k$);
- (3) $\{Y_k, k \geq 0\}$ 是 Z_0, Z_1, \dots, Z_k 的函数。

Doob 证明了下鞅收敛定理^{[141][142]}, 为了方便分析, 给出其结论。

定理 2.4: 设 $\{Y_k, k \geq 0\}$ 是一个下鞅, $\sup_k E|Y_k| < \infty$, 则存在一个随机变量 $Y^* \in \{Y_k, k \geq 0\}$, 使 $E|Y^*| < \infty$, 且 $Y_k \xrightarrow{a.s.} Y^*$, 即 $P\left\{\lim_{k \rightarrow \infty} Y_k = Y^*\right\} = 1$.

从直观上来说, 人工蜂群算法的目的是找到一个最优状态 $S(t)$, 所以可以把目标函数转换成适应值函数, 写成 $F(S_t)$, 从而可以利用 $S(t)$ 的 Markov 过程相关结论。下面设法把人工蜂群算法的随机过程 $\{F(S_t), t \geq 0\}$ 转变成为一个下鞅来考察 $\{S_t, t \geq 0\}$ 的收敛性。

命题 2.1: 状态空间 S 是一个可约闭集。

证明: 由定理 2.3 和定义 2.7 可知, F^* 为有限闭集, 又 $F^* \subset S$, 由定义 2.8 知 S 为可约的。由定理 2.2 可知 $\{S_t, t \geq 0\}$ 为一离散空间上的有限齐次马尔可夫过程, 则由马尔可夫过程的性质, 有 $\forall i \in S$, $\sum_{j \in S} p_{ij} = 1$, 再由定义 2.7 可知 S 为一个闭集。

由马尔可夫过程的状态空间分解定理, 若 S 是可约闭集, 则可以把闭集 S 分解成为两块, 一块是 $\{S_t, t \geq 0\}$ 的收敛空间 Q , 另一块是 $S - Q$, 其中 $Q = \{S_{t+1} | F(S_{t+1}) \geq F(S_t)\}$.

命题 2.2: Q 为收敛空间的一个闭集。

证明: 由集合 Q 的定义, 所有满足 $F(S_{t+1}) \geq F(S_t)$ 的状态全被限制在集合 Q 中, 而所有满足 $F(S_{t+1}) < F(S_t)$ 的状态全被限制在集合 $S - Q$ 中。因此, 在集合 Q 中不能找到满足条件 $F(S_{t+1}) < F(S_t)$ 的状态, 即集合 Q 中任意一个状态的后继状态只能在集合 Q 中, 而不能在集合 Q 外, 即 $\forall i \in Q, \sum_{j \in S - Q} p_{ij} = 0$, 所以有 $\forall i \in Q, \sum_{j \in Q} p_{ij} = 1$, 即集合 Q 是一个闭集, 且 $Q \in S$ 。

上述做法的目的是将 $\{S_t, t \geq 0\}$ 构成的有限齐次马尔可夫过程限制在其全状态空间 S 的闭子集 Q 中, 即: 集合 Q 外的任意状态构成状态空间的反射壁, 将其他所有满足 $F(S_{t+1}) \geq F(S_t)$ 的状态约束其内, 由随机过程相关理论可知, 这等同于状

态的限制运动边界，与桶的边缘类似。

命题 2.3: 随机过程 $\{F(S_t), t \geq 0\}$ 关于 $\{S_t, t \geq 0\}$ 是一个有界下鞅，即对 $\forall k \geq 0$ ，有 $E(F(S_{t+1})|S_0, S_1, \dots, S_t) \geq F(S_t)$ 。

证明：显然 $\{F(S_t), t \geq 0\}$ 是有界，根据定理 2.2 的马尔可夫性，可得 $E(F(S_{t+1})|S_0, S_1, \dots, S_t) = E(F(S_{t+1})|S_t)$ ，故只需要证明对 $\forall x \in S$ 有：

$$E(F(S_{t+1})|S_t = x) \geq F(x), t \geq 0. \quad (2-8)$$

又根据随机过程相关理论，有 $E(F(S_{t+1})|S_t = x) = \sum_{y \in S} F(y)p_t(S_{t+1} = y|S_t = x)$ 。根据定理 2.2 的证明，我们得出人工蜂群算法所限定的状态就为反射壁，将其他状态约束与条件 $F(x_{t+1}) \geq F(x_t)$ 限定在状态内，故有：

$$\begin{aligned} & \sum_{y \in S} F(y)p_t(S_{t+1} = y|S_t = x) \\ &= \sum_{y \in Q} F(y)p_t(S_{t+1} = y|S_t = x) + \sum_{y \in S-Q} F(y)p_t(S_{t+1} = y|S_t = x) \\ &= \sum_{y \in Q} F(y)p_t(S_{t+1} = y|S_t = x) + \sum_{y \in S-Q} F(y) \cdot 0 \\ &= \sum_{y \in Q} F(y)p_t(S_{t+1} = y|S_t = x) \end{aligned}$$

因为状态 x 的任意后继状态 y ，满足对 $\forall y \in Q$ ，有 $F(S_{t+1} = y) \geq F(S_t = x)$ ，所以：

$$\begin{aligned} \sum_{y \in Q} F(y)p_t(S_{t+1} = y|S_t = x) &\geq \sum_{y \in Q} F(x)p_t(S_{t+1} = y|S_t = x) \\ &= F(x) \sum_{y \in Q} p_t(S_{t+1} = y|S_t = x) \end{aligned}$$

又 Q 是闭集，所以有 $\sum_{y \in Q} p_t(S_{t+1} = y|S_t = x) = 1$ 。从而得到 $\sum_{y \in Q} F(y)p_t(S_{t+1} = y|S_t = x) \geq F(x)$ ，即 $\sum_{y \in S} F(y)p_t(S_{t+1} = y|S_t = x) \geq F(x)$ ，所以有 $E(F(S_{t+1})|S_t = x) \geq F(x)$ ，所以式(2-8)得证，即 $\{F(S_t), t \geq 0\}$ 关于 $\{S_t, t \geq 0\}$ 为一个有界下鞅。

命题 2.4: $\{F(S_t), t \geq 0\}$ 几乎处处强收敛到全局最优解集 F^* 。

证明：由定理 2.3 可知， $\{F(S_t), t \geq 0\}$ 以概率 1 收敛到全局最优解集 F^* ，又由命题 2.3 可知 $\{F(S_t), t \geq 0\}$ 为一个有界下鞅，根据定理 2.4， $\{F(S_t), t \geq 0\}$ 几乎处处强收敛到全局最优解集 F^* 。

由本文的命题 2.4 可知， $S_k \xrightarrow{a.s.} F^*$ ，且 $|S|$ 仅包含有限个不同点，根据

Egoroff^[142]定理可以推导出 $\{F(S_t), t \geq 0\}$ 具有潜在的一致收敛性, 即 $\forall \varepsilon > 0, \exists N(\varepsilon)$, 如果 $k \geq N(\varepsilon)$, 则 $\forall X \in F^*, |X_k - X| < \varepsilon$, 即 $X_k(\varepsilon) \subset F^*$, 说明 ABC 算法确保能够以概率 1 在有限步内 ($N(\varepsilon)$) 收敛到问题的全局最优解, 即 $P\{\bigcup \bigcap (X_k \subset F^*)\} = 1$.

2.6 小结

本章对于原始人工蜂群算法的基本原理进行了详细的阐述. 文中首先通过对蜜蜂种群觅食行为的具体分析, 说明蜜蜂种群具有自组织性和分工, 满足群智能特征. 其次, 根据蜜蜂种群觅食行为具体说明了人工蜂群算法对该行为的模拟, 并给出了算法的基本原理、流程图和具体步骤. 最后, 通过引入鞅理论对人工蜂群算法收敛性进行了分析研究. 通过将人工蜂群算法所形成状态序列转化为鞅过程进行分析, 得出人工蜂群算法几乎必然强收敛到全局最优解集. 与已有结论相比, 得到了算法更强的收敛性证明结果. 对人工蜂群算法改进形式的收敛性研究, 将作为后续的主要研究工作.

第三章 全局优化问题的混合人工蜂群算法

为了提高人工蜂群算法的优化性能，针对全局优化问题，本章提出一种混合人工蜂群算法。该算法通过引入正交初始化方法，提高算法初始解分布的均匀性。同时，设计一种新型的搜索策略，用来保持种群多样性，防止算法陷入局部最优。通过6个测试函数的实验测试，以及与其他算法的比较研究，表明该算法具有较好的优化性能。

3.1 引言

自然科学、工程和经济金融领域中的很多问题都可以构造为全局优化问题来进行求解。目前，很多仿生优化算法都用于求解这类优化问题，比如：遗传算法、蚁群算法、差分进化算法和粒子群优化算法等。

近期，学者Karaboga^[1]提出了一种新的群智能算法——人工蜂群算法，用于求解函数优化问题。该算法通过模拟蜜蜂群体觅食特征，构造算法搜索最优解。通过对函数优化问题的对比实验研究说明人工蜂群算法较其他仿生优化算法具有更好的优化性能^[5]。因此，该算法用于求解很多现实生活问题，如：最小生成树问题^[94]、车间调度问题^[87]、反分析问题^[112]、雷达分布系统的网络识别问题^[53]等。同时，人工蜂群算法也被扩展到求解约束优化问题^[18]、神经网络训练^[51]、无线传感器网络^[72]、TSP问题^[123]等。在这些应用问题中，人工蜂群算法也体现出了优越的优化性能。

然而，与其它仿生优化算法类似，对于多模问题人工蜂群算法也存在收敛速度慢和容易陷入局部最优的缺陷^[5]。为此，本章对于连续变量全局数值优化问题提出一种混合人工蜂群算法。通过引入正交初始化方法^[143]使初始解均匀分布在可行解空间上，从而加强算法的全局收敛性。受差分进化算法的启发，设计出一种新的解的搜索方程，该方程可以最大限度的保持种群多样性。实验对比结果显示出所设计的混合人工蜂群算法是可行有效的。

本章主要安排如下：3.2给出混合人工蜂群算法的具体步骤；3.3节是实验及结论分析；3.4节对本章进行小结。

3.2 混合人工蜂群算法

3.2.1 正交初始化

种群初始化是仿生优化算法中较为重要的一个步骤，它会影响到解的质量和

算法的收敛速度。通常情况下，在求解一个最优化问题之前，不能获得该问题解的位置信息。这就要求仿生优化算法中的初始种群能够均匀分布在问题的可行域内，以便算法在解的搜索过程中能均匀的在整个解空间进行搜索。

正交试验是统计学中的一种试验方法，该方法可以用较少的试验组合均匀反映出所有可能组合结果。因此，正交设计是一种较好的种群初始化方法。本章中使用基于正交数组和量化技术的种群初始化方法生成初始种群。具体方法可参看文献[143]。

3.2.2 新搜索机制

差分进化算法是一种基于种群的函数优化方法，该算法的主要搜索策略是通过计算个体与种群中其它随机抽取个体间的差分来生成新的个体。在很多优化问题和现实应用问题中，该算法都体现出了良好的优化性能。“DE/current-to-rand/1”是差分进化算法中变异策略的一种搜索策略，这种搜索方程可以有效地维持搜索过程中种群的多样性。受差分进化算法的启发，结合人工蜂群算法自身的特点，本节提出一种新的算法搜索策略如下：

$$v_j = x_j + \phi_j(x_{r1,j} - x_{r2,j}) + \psi_j(x_{r2,j} - x_{r3,j}), \quad (3-1)$$

其中，参数 $r1$ ， $r2$ 和 $r3$ 是 $\{1, 2, \dots, SN\}$ 中随机选择的不同于 i 的正整数； $j \in \{1, 2, \dots, D\}$ 是随机抽取的下标； $\phi_j, \psi_j \in [-1, 1]$ 是均匀分布的随机数。

如上(3-1)式所示，候选解中仅有一维被更新。为了提高搜索效率，对式(3-1)进行如下修正：

$$v_{i,m} = x_{i,m} + \phi_{i,m}(x_{i,m} - x_{r1,m}) + \psi_{i,m}(x_{r2,m} - x_{r3,m}), \quad (3-2)$$

其中， $1 \leq m \leq D$ 是随机选择的正整数，用来控制更新维数。

混合人工蜂群算法的搜索流程如下。

Algorithm: 混合人工蜂群算法

利用正交初始化方法给出初始种群，并评价种群质量，设定参数 $trail_i = 0$, ($i = 1, 2, \dots, SN$), $cycle = 1$ 。

Repeat

Step 1：对雇佣蜂根据式(3-2)搜索新蜜源，并估计蜜源质量。

Step 2：利用贪婪选择策略在新蜜源和旧蜜源中选择较好的一个。

Step 3：如果蜜源没有更新，则 $trail_i = trail_i + 1$ ，否则 $trail_i = 0$ 。

- Step 4: 利用轮盘赌选择, 为跟随蜂选择一个蜜源.
 Step 5: 对跟随蜂根据式(3-2)搜索新蜜源, 并估计蜜源质量.
 Step 6: 利用贪婪选择策略在新蜜源和旧蜜源中选择较好的一个.
 Step 7: 如果蜜源没有更新, 则 $trail_i = trail_i + 1$, 否则 $trail_i = 0$.
 Step 8: 若 $\max(trail_i) > \text{limit}$, 随机生成一个新蜜源代替旧蜜源.
 记忆当前最优解.
 $cycle = cycle + 1$.
 Until ($cycle = \text{Maximum Cycle Number}$)

3.3 数值实验

本节选取6个测试函数对混合人工蜂群算法的优化性能进行验证. 测试函数如下:

$$\text{Sphere 函数: } f_1(x) = \sum_{i=1}^D x_i^2;$$

$$\text{Rosenbrock 函数: } f_2(x) = \sum_{i=1}^D \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right];$$

$$\text{Ackley 函数: } f_3(x) = -20 \exp \left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2} \right) - \exp \left(\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i) \right) + 20 - e;$$

$$\text{Griewank 函数: } f_4(x) = \frac{1}{4000} \sum_{i=1}^D x_i^2 - \prod_{i=1}^D \cos(x_i / \sqrt{i}) + 1;$$

$$\text{Rastrigin 函数: } f_5(x) = \sum_{i=1}^D \left[x_i^2 - 10 \cos(2\pi x_i) + 10 \right];$$

$$\text{Schwefel 函数: } f_6(x) = \sum_{i=1}^D \left(-x_i \sin(\sqrt{|x_i|}) \right).$$

这些测试函数的具体特征如表2.1所示.

表 3.1 测试函数基本特征表

函数	$f(x)$	可行解空间	特征
$f_1(x)$	$f(\vec{0}) = 0$	$[-100, 100]^D$	单模
$f_2(x)$	$f(\vec{1}) = 0$	$[-2.048, 2.048]^D$	单模
$f_3(x)$	$f(\vec{0}) = 0$	$[-32.768, 32.768]^D$	多模
$f_4(x)$	$f(\vec{0}) = 0$	$[-600, 600]^D$	多模
$f_5(x)$	$f(\vec{0}) = 0$	$[-5.12, 5.12]^D$	多模
$f_6(x)$	$f(\overrightarrow{420.96}) = 418.9829 * D$	$[-500, 500]^D$	多模

表 3.2 混合人工蜂群算法和人工蜂群算法测试结果表

函数	维数	算法	Mean	Std.	Best	Worst
$f_1(x)$	30	ABC	6.87275e-016	1.16244e-016	4.51239e-016	9.3929e-016
		HABC	2.30804e-016	2.74413e-017	1.7433e-016	2.80967e-016
	60	ABC	3.57059e-015	1.35621e-015	1.87316e-015	7.36885e-015
		HABC	6.92762e-016	9.17927e-017	4.5471e-016	8.70451e-016
$f_2(x)$	30	ABC	4.39225	3.58558	0.818608	15.0884
		HABC	3.25105	1.40904	1.4533	8.31785
	60	ABC	43.7263	10.1838	16.588	57.5177
		HABC	6.19937	3.23483	0.823095	10.7344
$f_3(x)$	30	ABC	5.08038e-014	6.40644e-015	3.19744e-014	5.68434e-014
		HABC	1.10134e-014	2.52941e-015	7.10543e-015	1.42109e-014
	60	ABC	1.86723e-009	7.67121e-010	4.90815e-010	4.34077e-009
		HABC	3.43429e-014	3.27655e-015	2.84217e-014	3.90799e-014
$f_4(x)$	30	ABC	3.95728e-013	1.30327e-012	1.11022e-016	6.20848e-012
		HABC	0	0	0	0
	60	ABC	7.03881e-015	6.03749e-015	5.55112e-016	2.05391e-014
		HABC	1.70234e-016	1.19266e-016	0	4.44089e-016
$f_5(x)$	5	ABC	0	0	0	0
		HABC	0	0	0	0
	10	ABC	0	0	0	0
		HABC	0	0	0	0
$f_6(x)$	30	ABC	-12569.5	0.00118634	-12569.5	-12569.5
		HABC	-12569.5	3.36085e-012	-12569.5	-12569.5
	60	ABC	-24852	119.329	-25138.8	-24658.5
		HABC	-25139	9.33637e-012	-25139	-25139

文献[5]对人工蜂群算法进行了比较研究，通过与其他算法在全局优化问题上的比较实验发现，人工蜂群算法具有较好的优化性能，并且算法中的控制参数较少。本章中通过对6个测试函数进行测试，并比较混合人工蜂群算法和人工蜂群算法在这6个测试函数上的实验结果。实验中将对测试函数选取两种不同的维数来进行。Rastrigin函数的测试维数为5维和10维，其余5个测试函数的维数分别为30维和60维。算法中，种群规模设定为100，最大迭代次数为3000，因此，所有的实验中函数评价次数均为300000。同时，算法在每个测试函数上均独立重复运行

30次。

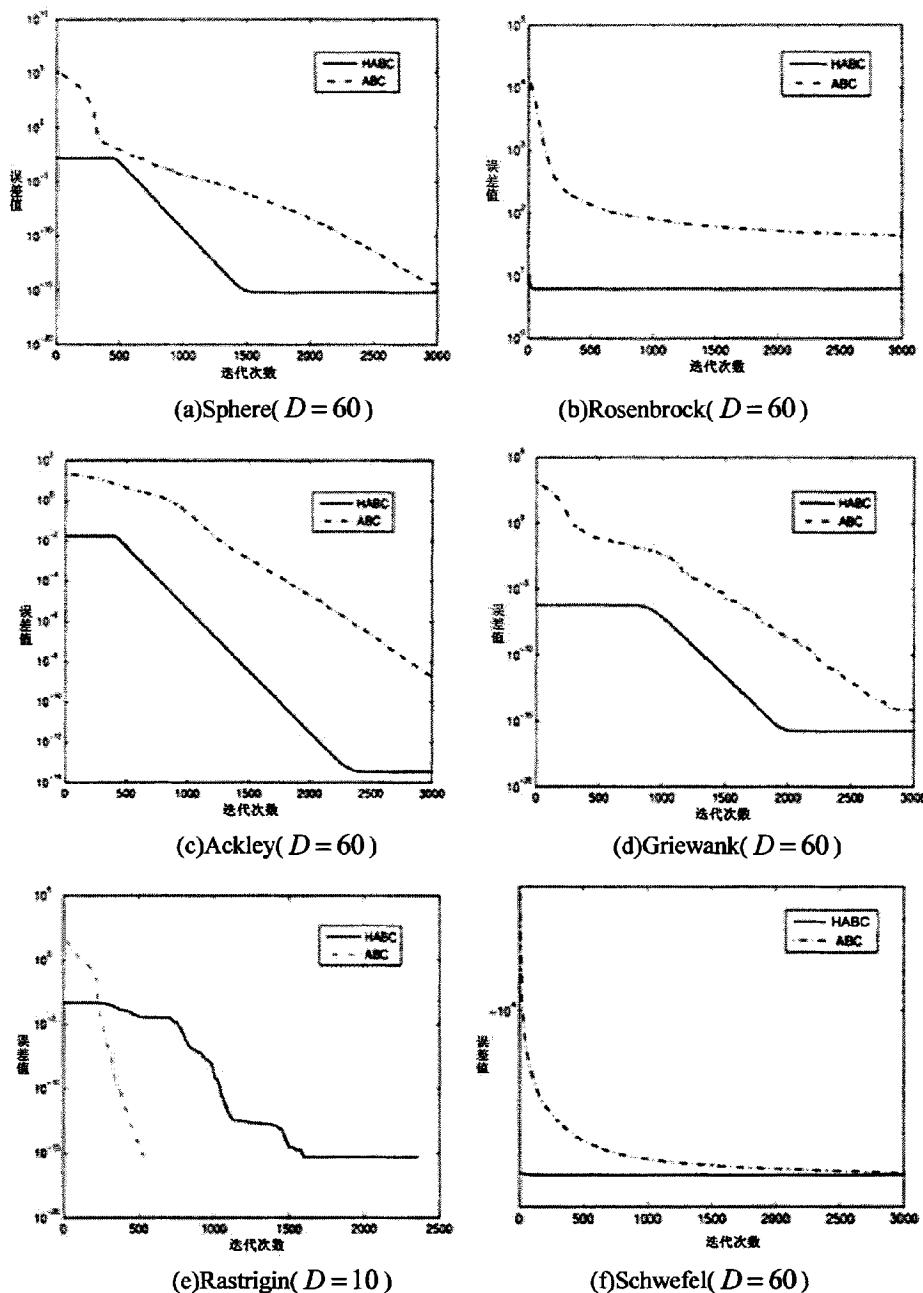


图 3.1 收敛曲线比较图

表3.2给出了实验结果，其中mean、std.、best和worst分别表示运行30次的所有最优函数值的均值、标准差、最好值和最差值。

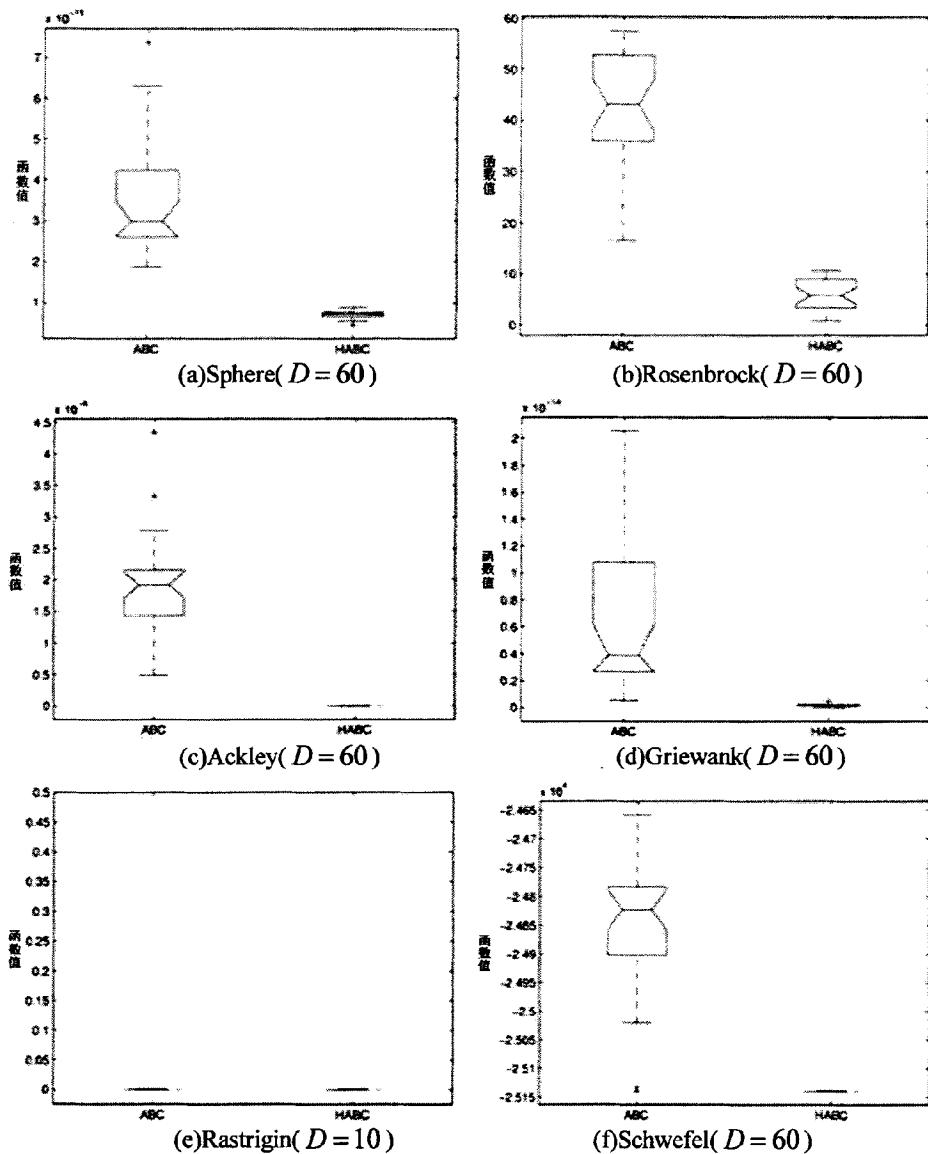


图 3.2 函数 30 次实验结果箱型图

表3.2给出了混合人工蜂群算法与人工蜂群算法的结果比较，通过比较发现，混合人工蜂群算法得到的最优点的均值都等于或接近于测试函数的最优点，并且标准差相对较小。其中，对于单模函数 f_1 ，混合人工蜂群算法得到的结果比人工蜂群算法更接近于最优点；对于多模函数 f_4 ，混合人工蜂群算法可以得到问题的最优点，而人工蜂群算法在同样的实验条件下则不能得到最优点。并且，当测试问题维数增加时，混合人工蜂群算法能得到更稳定的实验结果。这些结果均表明混合人工蜂群算法可以得到最优解或更接近于测试函数的最优解的解，并且算法的稳定性更好。

为了更清楚的展示混合人工蜂群算法的性能，给出图3.1来展示测试函数的收敛曲线图。

图3.1比较了混合人工蜂群算法和人工蜂群算法的收敛曲线，其中黑色直线表示混合人工蜂群算法的收敛曲线，黑色虚线表示人工蜂群算法的收敛曲线。从图中可以观察到，对于Rastrigin函数，尽管混合人工蜂群算法可以收敛到最优解，但其收敛速度较人工蜂群算法更慢些。但对于其他5个函数来说，混合人工蜂群算法收敛速度明显快于人工蜂群算法，并且更接近于最优值，说明混合人工蜂群算法具有更好的收敛特征。

为了说明算法的测试稳定性，图3.2给出了通过箱型图给出测试函数的最优值的统计特征。箱型图^[144]用来描绘30次独立实验结果的分布情况。图中箱子的底和顶分别表示30次独立实验的最优解和最差解；箱子中间的横线表示中值；箱子的形状反映了30次解的分布情况。标记“+”表示其他散点。图3.2说明混合人工蜂群算法可以得到较人工蜂群算法更好的解，并且混合人工蜂群算法更加稳定。图3.2进一步说明了表3.2中的实验结果。

3.4 小结

本章对于全局函数优化问题给出一种混合人工蜂群算法，算法通过引入正交初始化和设计新的搜索模式来提高算法的优化性能。通过对6个测试函数的测试，并与人工蜂群算法进行比较，说明本章中所提出的混合人工蜂群算法在准确性、收敛速度、稳定性及鲁棒性上都比人工蜂群算法更优越。

第四章 函数优化问题的改进人工蜂群算法

本章提出了一种改进人工蜂群算法。为了提高算法的性能，通过引入正交初始化方法和新的搜索机制，平衡算法的探索和开发能力，在提高算法收敛速度的同时，有效保持种群多样性，防止算法陷入局部最优。通过对27个benchmark测试函数的实验来验证算法的优化性能。通过与其他算法的比较研究，表明该算法较人工蜂群算法及其他智能优化算法具有更好的优化性能。

4.1 引言

全局优化问题产生于自然科学、工程和经济金融等众多领域中。实际中这类优化问题通常较为复杂，且不具备可微性，对于传统优化方法来说，在求解时就具有一定的挑战。目前，由于很多仿生优化算法对问题的先验信息及可微性均没有要求，因此成为求解这类优化问题的一类主要方法。这些仿生优化算法通常是模拟自然界的特征而提出的，通常也被称为人工生命计算，其中较为常见的算法包括：遗传算法、蚁群算法、差分进化算法和粒子群优化算法等。近期，学者Karaboga^[1]提出一种新的群智能算法——人工蜂群算法，用于求解全局函数优化问题。该算法通过模拟蜜蜂群体觅食特征构造算法搜索最优解的方式。通过对比实验研究说明人工蜂群算法较其他仿生优化算法具有更好的优化性能，并且算法具有较少的优化参数^[5]。因此，该算法用于求解很多现实生活问题，如：数据挖掘^[67]、图像处理^[79]、神经网络训练^[52]、无线传感器网络^[75]等等。

通过对人工蜂群算法的性能进行测试，以及在实际问题上的应用分析，似乎这一算法具有极其完美的优化性能。然而，与其它仿生优化算法类似，人工蜂群算法也具有一些不足之处，比如：对于单模问题算法存在收敛速度慢的缺点，而对于多模问题人工蜂群算法也存在容易陷入局部最优的缺陷^[5]。众所周知，对于群智能算法，算法的探索能力和开发能力在算法优化过程中起着举足轻重的作用。探索能力赋予算法在未知区域内寻找全局最优解的能力；开发能力则赋予算法对当前较好解进行继续开发和局部搜索以找寻更好解的能力。但对于一个算法来说，探索能力和开发能力在优化过程中是相互矛盾的。为了提高算法的优化性能，就需要对探索能力和开发能力进行平衡。然而，人工蜂群算法的搜索方程具有较好的探索能力，而开发能力较弱。这样就使得算法的局部搜索能力较弱，降低算法的收敛速度。

因此，提高算法的收敛速度，同时保证避免算法早熟就成为对人工蜂群算法改进的主要问题。为了解决这两个问题，提高算法的优化性能，本章对于连续变

量全局数值优化问题提出一种改进人工蜂群算法。受差分进化算法的启发，在改进人工蜂群算法中设计一种新的搜索机制。同时，为了均衡算法的探索能力和开发能力，基于差分进化算法的搜索方程，设计了两种新的搜索方程。给定选择概率 P ，来控制哪个搜索方程将会在后期搜索中用到。进一步，通过引入正交初始化方法使初始解均匀分布在可行解空间上，从而加强算法的全局收敛性^[143]。实验对比结果显示所设计的改进人工蜂群算法是可行的有效算法。

本章主要安排如下：4.2节给出改进人工蜂群算法的具体步骤；4.3节是实验及结论分析；4.4节对本章进行小结。

4.2 改进人工蜂群算法

4.2.1 正交初始化

种群初始化是仿生优化算法中较为重要的一个步骤，它会影响到解的质量和算法的收敛速度。正交试验是统计学中的一种试验方法，本章中使用基于正交数组和量化技术的种群初始化方法生成初始种群。这种初始化方法可以使初始解均匀分布在可行域内，提高算法搜索最优解的速度^{[143][145]}。种群初始化算法流程如下。

Algorithm 4.1： 种群初始化

Step 1：将问题可行域 $[l, u]$ 划分成 S 个子集 $[l_1, u_1], [l_2, u_2], \dots, [l_S, u_S]$ ，具体划分方法如下：

$$\begin{cases} l_i = l + (i-1) \left(\frac{u(s) - l(s)}{S} \right) l_s, \\ u_i = u - (S-i) \left(\frac{u(s) - l(s)}{S} \right) l_s, \end{cases} \quad i = 1, 2, \dots, S. \quad (4-1)$$

其中， $u(s) - l(s) = \max_{1 \leq i \leq D} \{u_i - l_i\}$ 。

Step 2：将子集 $[l_i, u_i]$ 进行 Q 次量化如下：

$$\alpha_{ij} = \begin{cases} l_i, & j=1, \\ l_i + (j-1) \left(\frac{u_i - l_i}{Q_i - 1} \right), & 2 \leq j \leq Q_i - 1, \\ u_i, & j=Q_i, \end{cases} \quad (4-2)$$

其中 Q 是奇数。然后，构造正交数组 $L_{M_i}(Q_i^N) = [a_{ij}]_{M_i \times N}$ ，根据(4-3)式

$$\begin{cases} (\alpha_{1,a_{11}}, \alpha_{2,a_{12}}, \dots, \alpha_{N,a_{1N}}) \\ (\alpha_{1,a_{21}}, \alpha_{2,a_{22}}, \dots, \alpha_{N,a_{2N}}) \\ \dots \\ (\alpha_{1,a_{M_11}}, \alpha_{2,a_{M_12}}, \dots, \alpha_{N,a_{M_1N}}). \end{cases} \quad (4-3)$$

选择 M_1 个个体. 其中 $L_{M_1}(Q_1^N)$ 可按如下方法进行构造: 选取满足 $(Q_1^l - 1)/(Q_1 - 1) \geq N$ 的最小 J_1 . 若 $(Q_1^l - 1)/(Q_1 - 1) = N$, 则 $N' = N$, 否则 $N' = (Q_1^l - 1)/(Q_1 - 1)$. 构造数组的基本列 $j = \frac{Q_1^{k-1} - 1}{Q_1 - 1} + 1$, $a_j = \left\lfloor \frac{i-1}{Q_1^{J_1-k}} \right\rfloor \bmod Q_1$, $i = 1, \dots, M_1$, $k = 1, \dots, J_1$; 非基本列为 $j = \frac{Q_1^{k-1} - 1}{Q_1 - 1} + 1$, $a_{j+(s-1)(Q_1-1)+t} = (a_s \times t + a_j) \bmod Q_1$, $s = 1, \dots, j-1$, $t = 1, \dots, Q_1$. 这样数组 $L_{M_1}(Q_1^N)$ 就构造完成. 删除 $L_{M_1}(Q_1^N)$ 的最后 $N' - N$ 列就得到了 $L_{M_1}(Q_1^N)$, 其中 $M_1 = Q_1^l$.

Step 3: 在 MS 个个体中, 选取 SN 个适应度最高的个体作为初始种群.

4.2.2 新搜索机制

众所周知, 对于群智能算法来说, 如何平衡算法的探索能力和开发能力来寻求最优解是一个非常重要的问题. 算法的探索能力是指算法在解空间内搜索未知区域寻求最优解的一种能力; 而开发能力则指对当前最优解进行局部搜索来发现更好解的能力. 这两种能力通常情况下是相互矛盾的, 不易同时兼顾, 因此需要对这两种搜索能力进行较好的均衡.

在人工蜂群算法中, 雇佣蜂开发蜜源并将蜜源信息传递给跟随蜂. 跟随蜂则根据这些信息选择一个蜜源进行开发. 勘查蜂则对已丢弃的蜜源进行重新探索. 因此, 在人工蜂群算法中, 雇佣蜂和跟随蜂代表了算法的开发能力, 勘查蜂则代表算法的探索能力. 同时, 人工蜂群算法的搜索方程在寻优过程中具有较强的探索能力, 开发能力较差. 为了增强算法的开发能力, 本节给出一种基于差分进化策略的搜索机制.

差分进化算法是一种基于种群的函数优化方法, 该算法的主要搜索策略是通过计算个体与种群中其它随机抽取个体间的差分来生成新的个体. 在很多优化问题和现实应用问题中, 该算法都体现出了良好的优化性能. 其算法步骤与一般的进化算法相同. 在差分进化算法中进化算子包括变异、交叉和选择. 其中, 变异

算子的形式有很多种，这就构成了不同的差分进化算法。在这些变异算子中，“DE/rand/2”可以有效维持种群多样性，防止算法早熟；“DE/best/2”可以提高算法局部搜索能力，加快算法收敛速度。这两种变异算子是差分进化算法中最常用的两种算子，具体方程如下：

$$\text{DE/rand/2: } v_i = x_{r1} + F(x_{r2} + x_{r3} - x_{r4} - x_{r5}), \quad (4-4)$$

$$\text{DE/best/2: } v_i = x_{best} + F(x_{r1} + x_{r2} - x_{r3} - x_{r4}), \quad (4-5)$$

其中 $i \in \{1, 2, \dots, SN\}$ ； $r1, r2, r3, r4$ 和 $r5$ 是 $\{1, 2, \dots, SN\}$ 中随机选择的参数； x_{best} 表示当前全局最优解； $F \in [0.5, 1]$ 是一个正实数。

受差分进化算法的启发，结合人工蜂群算法的特点，本节给出两个新的搜索方程，如下：

$$v_{i,j} = x_{r1,j} + \varphi_{i,j}(x_{i,j} - x_{r2,j}) + \phi_{i,j}(x_{r3,j} - x_{r4,j}), \quad (4-6)$$

$$v_{i,j} = x_{best,j} + \varphi_{i,j}(x_{i,j} - x_{r1,j}) + \phi_{i,j}(x_{r2,j} - x_{r3,j}), \quad (4-7)$$

其中 $i \in \{1, 2, \dots, SN\}$ ； $r1, r2, r3$ 和 $r4$ 是 $\{1, 2, \dots, SN\}$ 中随机选择的整数，且都与 i 不同； $x_{best,j}$ 是当前全局最优解； $j \in \{1, 2, \dots, D\}$ 是随机选择的指标； $\varphi_{ij} \in [-1, 1]$ 和 $\phi_{ij} \in [0.5, 1]$ 是均匀分布的随机数。

与差分进化算法相类似，搜索方程(4-6)可以有效维持种群多样性，搜索方程(4-7)则可以提高收敛速度，也即增强算法的开发能力。所以，为了将两个搜索方程的优点有效结合，本节给出一种新的搜索策略。该策略引入一个选择概率 P 来控制方程(4-6)和(4-7)的使用频率。另外，在人工蜂群算法的搜索方程中，仅对个体中的一维进行搜索，搜索能力较低。为了克服这一问题，在使用新搜索方程进行搜索时，对个体的每一维都进行搜索。新搜索机制的主要步骤如下。

Algorithm 4.2: 搜索机制

Step 1: 给定蜜源 x_i 和选择概率 P 。产生一个新蜜源 v_i 。

Step 2: 若 $rand < p$ ，则使用(4-6)式根据旧蜜源 x_i 产生新蜜源 v_i 。

Step 3: 若 $rand \geq p$ ，则使用(4-7)式根据旧蜜源 x_i 产生新蜜源 v_i 。

4.2.3 改进人工蜂群算法

基于以上分析，改进人工蜂群算法的主要步骤如下所示。

Algorithm 4.3: 改进人工蜂群算法

利用 Algorithm 4.1 给出初始种群，并评价种群质量，设定参数 $trail_i = 0, (i = 1, 2, \dots, SN)$, $cycle = 1$.

Repeat

Step 1: 对雇佣蜂根据Algorithm 4.2搜索新蜜源，并估计蜜源质量.

Step 2: 利用贪婪选择策略在新蜜源和旧蜜源中选择较好的一个.

Step 3: 如果蜜源没有更新，则 $trail_i = trail_i + 1$, 否则 $trail_i = 0$.

Step 4: 利用轮盘赌选择，为跟随蜂选择一个蜜源.

Step 5: 对跟随蜂根据Algorithm 4.2搜索新蜜源，并估计蜜源质量.

Step 6: 利用贪婪选择策略在新蜜源和旧蜜源中选择较好的一个.

Step 7: 如果蜜源没有更新，则 $trail_i = trail_i + 1$, 否则 $trail_i = 0$.

Step 8: 若 $\max(trail_i) > limit$, 随机生成一个新蜜源代替旧蜜源.

记忆当前最优解.

$cycle = cycle + 1$.

Until ($cycle = Maximum\ Cycle\ Number$)

4.3 数值实验

4.3.1 测试函数

本节选取27个benchmark测试函数对改进人工蜂群算法的优化性能进行验证，见表4.1-4.2，表中C表示函数特征，U表示单模函数，M表示多模函数，S表示函数可分，N表示函数不可分。表4.1中第三列给出测试函数的维数。表4.2中的测试函数维数均为 $D=30$ ，且表达式中 $n=D$ 。

测试函数Hartman3中，

$$a = \begin{bmatrix} 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}^T, \quad p = \begin{bmatrix} 0.3689 & 0.1170 & 0.2673 \\ 0.4699 & 0.4387 & 0.7470 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{bmatrix}^T.$$

测试函数Shekel中，

$$a = \begin{bmatrix} 4.0 & 1.0 & 8.0 & 6.0 & 3.0 & 2.0 & 5.0 & 8.0 & 6.0 & 7.0 \\ 4.0 & 1.0 & 8.0 & 6.0 & 7.0 & 9.0 & 5.0 & 1.0 & 2.0 & 3.6 \\ 4.0 & 1.0 & 8.0 & 6.0 & 3.0 & 2.0 & 3.0 & 8.0 & 6.0 & 7.0 \\ 4.0 & 1.0 & 8.0 & 6.0 & 7.0 & 9.0 & 3.0 & 1.0 & 2.0 & 3.6 \end{bmatrix},$$

$$c = \frac{1}{10} [1, 2, 2, 4, 4, 6, 3, 7, 5, 5]^T.$$

表 4.1 Benchmark 函数 f_1-f_{16}

函数	定义域	D	C	表达式
Beale	$[-4.5, 4.5]$	2	UN	$f_1 = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2$
Bohachevsky	$[-100, 100]$	2	MS	$f_2 = x_1^2 + 2x_2^2 - 0.3 \cos(3\pi x_1) - 0.4 \cos(4\pi x_2) + 0.7$
Booth	$[-10, 10]$	2	MS	$f_3 = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$
Branin	$[-5, 10] \times [0, 15]$	2	MS	$f_4 = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) \cos x_1 + 10$
Colville	$[-10, 10]$	4	UN	$f_5 = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 + 10.1((x_2 - 1)^2 + (x_4 - 1)^2)$
Easom	$[-100, 100]$	2	UN	$f_6 = -\cos x_1 \cos x_2 \exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2)$
GoldStein-Price	$[-2, 2]$	2	M N	$f_7 = \begin{bmatrix} 1 + (x_1 + x_2 + 1)^2 \\ (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2) \\ \cdot \begin{bmatrix} 30 + (2x_1 - 3x_2)^2 \\ (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2) \end{bmatrix} \end{bmatrix}$
Hartman3	$[0, 1]$	3	M N	$f_8 = -\sum_{i=1}^4 c_i \exp \left[-\sum_{j=1}^3 a_{ij} (x_j - p_{ij})^2 \right];$ $c = [1.0, 1.2, 3.0, 3.2]$
Six Hump Camel Back	$[-5, 5]$	2	M N	$f_9 = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$
Matyas	$[-10, 10]$	2	UN	$f_{10} = 0.26(x_1^2 + x_2^2) - 0.48x_1 x_2$
Perm	$[-D, D]$	2	M N	$f_{11} = \sum_{k=1}^n \left[\sum_{i=1}^2 (i^k + 0.5)((x_i/i)^k - 1) \right]^2$
Powell	$[-4, 5]$	4	UN	$f_{12} = \sum_{i=1}^{n/k} (x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} + 10x_{4i})^2 + (x_{4i-2} + 10x_{4i})^2 + 10(x_{4i-3} + 10x_{4i})^4$
PowerSum	$[0, D]$	2 4	M N	$f_{13} = \sum_{k=1}^n \left[\sum_{i=1}^4 (x_i^k) - b_k \right]^2; b = [8, 18, 44, 114]$
Shekel	$[0, 10]$	4	M N	$f_{14} = -\sum_{j=1}^m \left[\sum_{i=1}^4 (x_i - a_{ij})^2 + c_i \right]^{-1}$
Shubert	$[-10, 10]$	2	M N	$f_{15} = \left(\sum_{i=1}^5 i \cos((i+1)x_1 + i) \right) \cdot \left(\sum_{i=1}^5 i \cos((i+1)x_2 + i) \right)$
Trid6	$[-36, 36]$	6	UN	$f_{16} = \sum_{i=1}^n (x_i - 1)^2 - \sum_{i=2}^n x_i x_{i-1}$

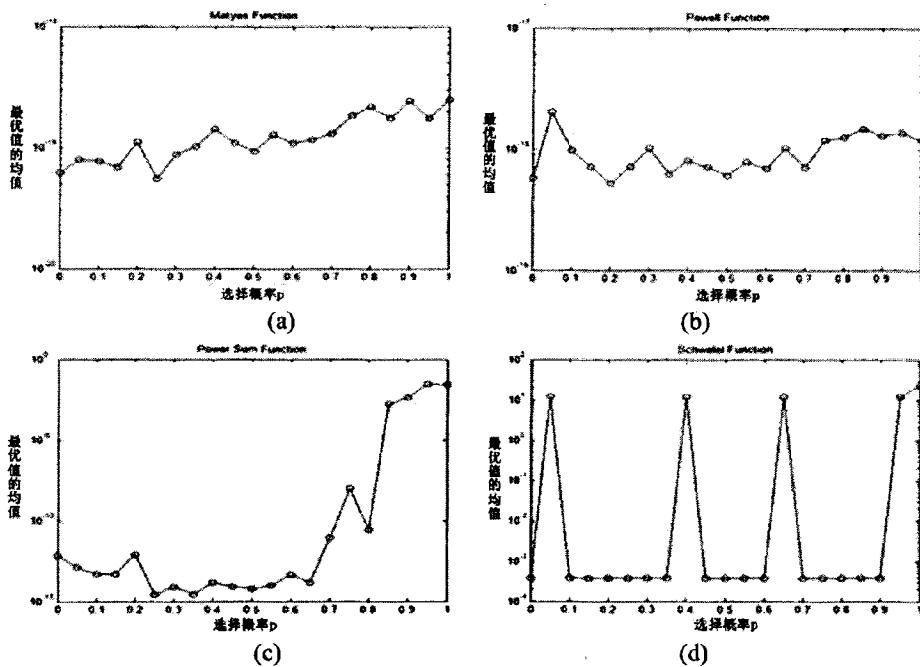
表 4.2 Benchmark 函数 f_{17} - f_{27}

Function	Range	C	Formulation
Ackley	[-32,32]	M	$f_{17} = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i)\right) + 2$
		N	
Dixon-Pric e	[-10,10]	UN	$f_{18} = (x_1 - 1)^2 + \sum_{i=2}^n i(2x_i^2 - x_{i-1})^2$
Griewank	[-600,600]	M N	$f_{19} = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
Levy	[-10,10]	M N	$f_{20} = \sin^2(\pi y_1) + \sum_{i=1}^{n-1} [(y_i - 1)^2 (1 + 10 \sin^2(\pi y_i + 1))] + (y_n - 1)^2 (1 + 10 \sin^2(2\pi y_n))$
			$y_i = 1 + \frac{x_i - 1}{4}, \quad i = 1, \dots, n.$
Michalewi cz	[0, π]	MS	$f_{21} = -\sum_{i=1}^n \sin(x_i) (\sin(ix_i^2 / \pi))^{2m}, m = 10$
Rastrigin	[-5.12,5.1 2]	MS	$f_{22} = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10]$
Rosenbroc k	[-30,30]	UN	$f_{23} = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
Schwefel	[-500,500]	MS	$f_{24} = \sum_{i=1}^n -x_i \sin(\sqrt{ x_i })$
Sphere	[-100,100]	US	$f_{25} = \sum_{i=1}^n x_i^2$
SumSquare s	[-10,10]	US	$f_{26} = \sum_{i=1}^n ix_i^2$
Zakharov	[-5,10]	UN	$f_{27} = \sum_{i=1}^n x_i^2 + \left(\sum_{i=1}^n 0.5ix_i\right)^2 + \left(\sum_{i=1}^n 0.5ix_i\right)^4$

4.3.2 选择概率 P 的影响分析

本小节中将对新算法中的选择概率 P 的影响进行讨论。在27个benchmark测试函数中测试函数Matyas, Powell, PowerSum和Schewefel具有一定的代表性，因此选择概率 P 的测试将在这四个函数上进行。对改进人工蜂群算法独立运行30次，最优值的30次均值图形如图4.1。因为所有的测试函数均是最小化问题，因此均值越小，则结果越好。

从图4.1中可以看到，选择概率 P 确实对实验结果有影响。当 P 约等于0.25时，得到的实验结果是最好的。所以，对其他测试函数的数值实验中将都使用 $P = 0.25$ 。

图 4.1 四个测试函数在不同选择概率 p 下的结果图

4.3.3 改进人工蜂群算法与人工蜂群算法的比较

为了测试改进人工蜂群算法的性能，本小节中将改进人工蜂群算法的实验结果与原始人工蜂群算法进行比较。实验中，改进人工蜂群算法与人工蜂群算法设定相同的参数。其中，种群规模 SN 、 $limit$ 和最大循环代数 (MSN) 分别设定为 50、 $(SN/2)*D$ 和 1000。所有实验均独立重复 30 次，每个测试函数的函数评价次数 (FEs) 为 5.0×10^4 。

表 4.3 给出了实验结果，表中 best、worst、mean 和 std 分别表示 30 次独立实验中的最好解、最差解、均值和标准差。最好的结果均用黑体标注出来。

如表 4.3 所示，改进人工蜂群算法的函数最优值均值均等于或较人工蜂群算法更接近于测试函数的最优值，并且误差较人工蜂群算法更小。特别指出，在单模函数 f_1 、 f_{10} 、 f_{12} 、 f_{25} 、 f_{26} 、 f_{27} 和多模函数 f_3 、 f_{13} 、 f_{17} 、 f_{19} 、 f_{20} 、 f_{22} 、 f_{24} 上，改进人工蜂群算法比人工蜂群算法的优化性能更好；在函数 f_2 、 f_4 、 f_7 和 f_8 上两个算法优化性能相同，都能达到最优值。这些比较结果都说明了改进人工蜂群算法不论在单模问题还是多模问题上优化性能都比原始人工蜂群算法更好。

为了进一步说明改进人工蜂群算法的性能，图 4.2 给出了 10 个测试函数的收敛曲线图。很明显，改进人工蜂群算法的收敛速度比原始人工蜂群算法更快。

图 4.3 是实验结果的箱型图。图形描绘了 30 次独立运行结果的分布图形。图中箱型的上边和下边分别表示最大和最小的值，箱子中间的线表示中值。箱子形状反映了值的分布形状。从图 4.3 中可以看到，改进人工蜂群算法较人工蜂群算法能

得到更好更稳定的解，进一步说明了表4.3和图4.2中所反映的结论。

表 4.3 改进人工蜂群算法及人工蜂群算法实验结果

	ABC				IABC			
	Best	Mean	Worst	Std	Best	Mean	Worst	Std
f_1	8.80e-07	6.37e-06	5.97e-05	1.10e-05	2.68e-22	6.62e-20	2.71e-19	7.75e-20
f_2	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00
f_3	5.82e-20	2.44e-17	7.00e-17	2.27e-17	1.50e-23	3.09e-20	1.61e-19	4.10e-20
f_4	3.98e-01	3.98e-01	3.98e-01	0.00e+00	3.98e-01	3.98e-01	3.98e-01	0.00e+00
f_5	-2.80e+11	-1.10e+10	-1.63e+05	5.02e+10	-1.00e+166	-5.00e+164	-4.30e+07	6.55e+04
f_6	-1.00e+00	-0.99e+00	-0.99e+00	1.38e-03	-1.00e+00	-1.00e+00	-1.00e+00	0.00e+00
f_7	3.00e+00	3.00e+00	3.00e+00	5.43e-04	3.00e+00	3.00e+00	3.00e+00	2.11e-15
f_8	-3.86e+00	-3.86e+00	-3.86e+00	2.24e-15	-3.86e+00	-3.86e+00	-3.86e+00	2.63e-15
f_9	4.65e-08	4.65e-08	4.65e-08	4.05e-17	4.65e-08	4.65e-08	4.65e-08	6.78e-17
f_{10}	8.93e-15	4.58e-10	4.82e-09	1.01e-09	6.54e-22	1.60e-20	5.22e-20	1.53e-20
f_{11}	1.12e+80	2.28e+83	1.93e+84	4.50e+83	1.35e+76	4.59e+85	4.28e+86	1.31e+86
f_{12}	8.58e-06	9.57e-05	1.94e-04	4.21e-05	2.26e-20	1.83e-18	8.75e-18	2.36e-18
f_{13}	1.66e-04	2.47e-02	9.26e-02	2.29e-02	6.51e-16	7.99e-09	1.73e-07	3.16e-08
f_{14}	-10.5364	-10.5347	-10.4967	7.50e-02	-10.5364	-9.10373	-5.12848	2.42e+00
f_{15}	-186.731	-186.731	-186.731	3.58e-14	-186.731	-186.731	-186.731	3.84e-14
f_{16}	-50	-50	-49.9999	1.60e-05	-49.9997	-46.7195	-39.9574	3.27e+00
f_{17}	7.03e-09	1.57e-07	8.94e-07	2.13e-07	8.88e-16	8.88e-16	8.88e-16	0.00e+00
f_{18}	2.47e-03	2.88e-02	0.193e+00	3.49e-02	0.16e+00	4.32e-01	6.67e-01	2.55e-01
f_{19}	2.13e-11	3.36e-04	9.86e-03	1.79e-03	0.00e+00	0.00e+00	0.00e+00	0.00e+00
f_{20}	1.07e-14	1.82e-13	6.51e-13	1.45e-13	5.98e-20	1.94e-18	1.50e-17	2.87e-18
f_{21}	-29.3027	-28.9797	-28.6755	1.68e-01	-17.4652	-13.8766	-11.6965	1.49e+00
f_{22}	3.50e-08	2.53e-01	1.05e+00	4.09e-01	0.00e+00	0.00e+00	0.00e+00	0.00e+00
f_{23}	3.55e-02	9.49e-01	7.16e+00	1.42e+00	3.12e-20	6.75e+00	2.84e+01	1.12e+01
f_{24}	5.20e-04	2.77e+02	5.91e+02	1.48e+02	3.82e-04	3.82e-04	3.82e-04	0.00e+00
f_{25}	3.86e-15	2.33e-14	1.09e-13	2.42e-14	3.99e-20	2.39e-18	9.55e-18	2.62e-18
f_{26}	1.04e-13	2.43e-12	1.62e-11	2.94e-12	1.01e-19	2.00e-18	7.28e-18	2.07e-18
f_{27}	2.16e+02	2.77e+02	3.45e+02	3.17e+01	2.00e-18	4.19e-01	4.11e+00	9.69e-01

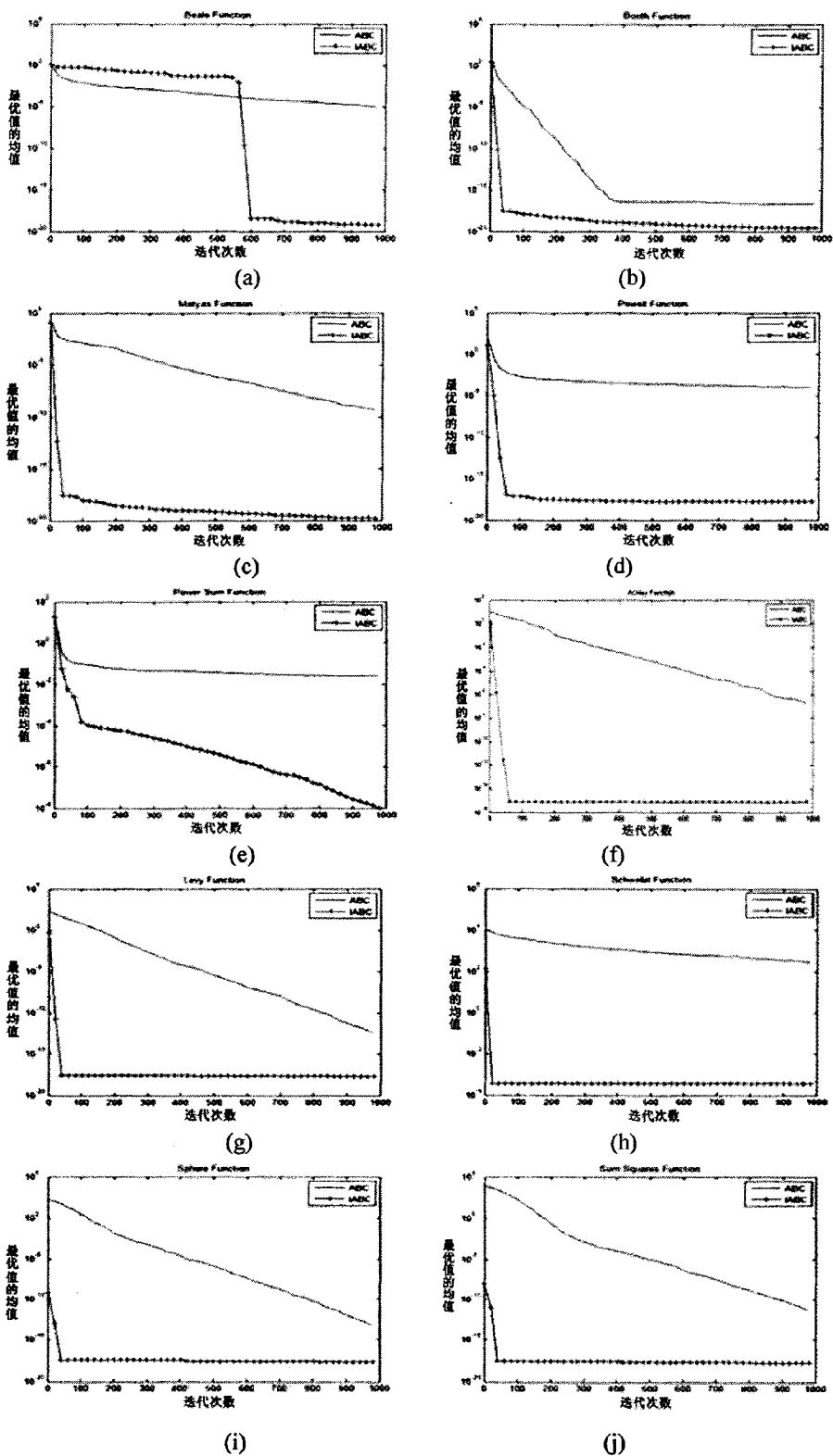


图 4.2 收敛曲线图

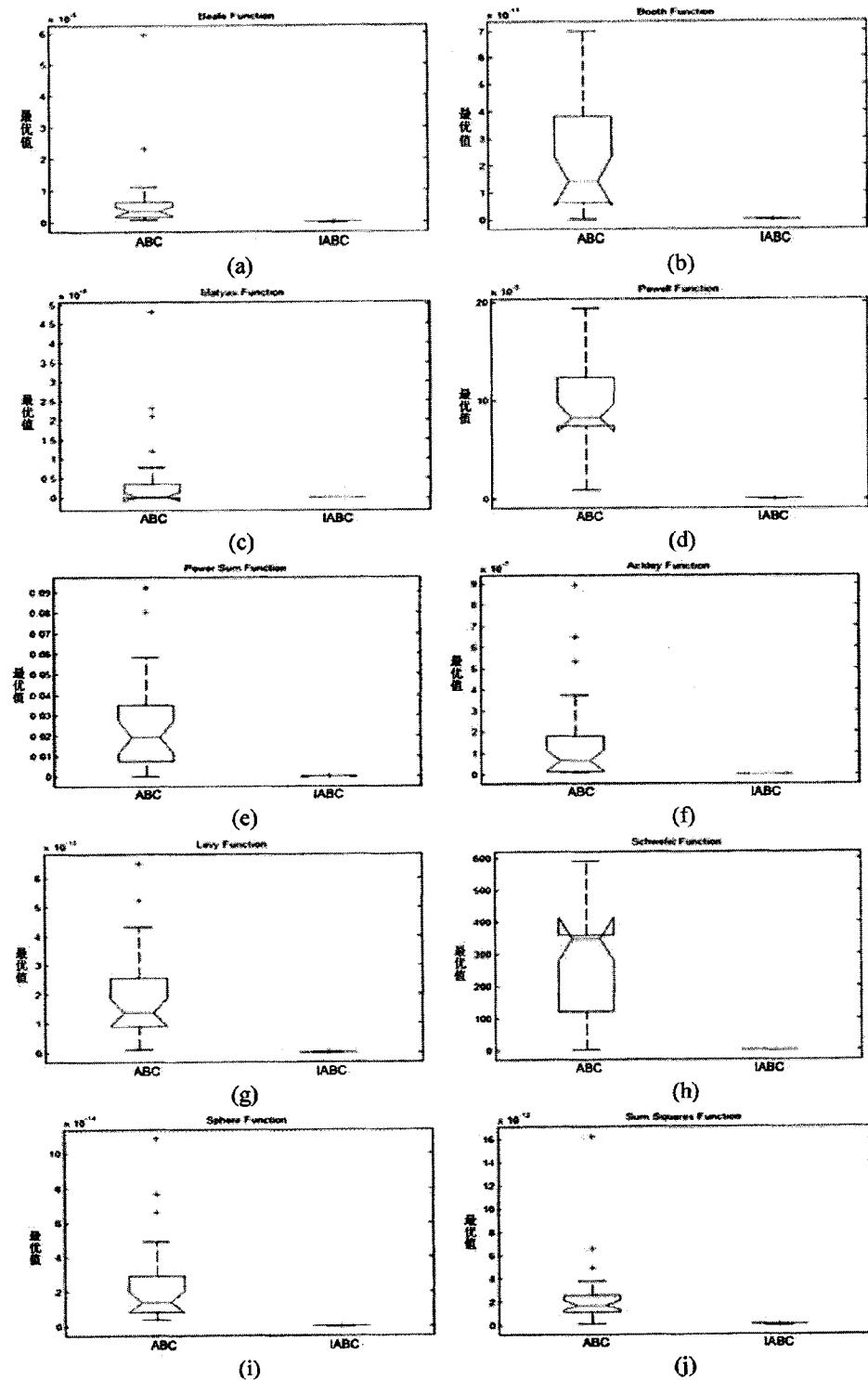


图 4.3 实验结果箱型图

4.3.4 与其他算法比较

为了更进一步测试改进人工蜂群算法的性能，本小节将改进人工蜂群算法与其他群智能算法，包括一些其他的改进人工蜂群算法进行比较研究。比较算法包括DE^[5], PSO^[5], CLPSO^[146], CES^[147], FES^[148], ESLAT^[149], CMA-ES^[149], GABC^[27], I-ABC^[150], PS-ABC^[150]和NABC^[151]。其中DE和PSO的参数设定和文献[5]相同，剩余其他算法的参数设定均按原始论文中的设定给出。

表4.4给出了DE, PSO, CES, FES, ESLAT, CMA-ES和改进人工蜂群算法的比较结果，其中DE, PSO, CES, FES, ESLAT和CMA-ES的实验结果来自于文献[5]。表4.5给出改进人工蜂群算法与其他四种人工蜂群算法的比较结果，其他四种人工蜂群算法结果来自于文献[151]。表4.4和表4.5中的最优结果均用黑体标注。

表 4.4 改进人工蜂群算法与其他算法比较结果

	Ackley	Griewank	Rastrigin	Rosenbrock	Schwefel	Sphere
	Mean	Mean	Mean	Mean	Mean	Mean
DE	3.99e-08	6.15e-04	1.47e+02	4.71e+03	7.27e+03	3.43e-14
PSO	3.23e-01	1.34e-02	3.85e+01	5.74e+03	4.16e+03	2.13e-16
CLPSO	2.01e-12	6.45e-13	2.57e-11	1.10e+01	1.19e+01	1.89e-19
CES	6.00e-13	6.00e-14	1.34e+01	2.77e+01	4.57e+03	1.70e-26
FES	1.20e-02	3.70e-02	1.60e-01	3.33e+01	1.31e+01	2.50e-04
ESLAT	1.80e-08	1.40e-03	4.65e+00	1.93e+00	1.03e+04	2.00e-17
CMA-ES	6.90e-12	7.40e-04	5.18e+01	4.00e-01	4.93e+03	9.70e-23
IABC	8.88e-16	0.00e+00	0.00e+00	6.75e+00	3.82e-04	2.39e-18

从表4.4中可以看出，改进人工蜂群算法在6个测试函数上比DE, PSO和FES结果更好。对于函数Sphere，算法CLPSO, CES和CMA-ES能得到更好解；对于函数Rosenbrock, ESLAT和CMA-ES能得到更好解。除了函数Sphere和Rosenbrock，改进人工蜂群算法在其他函数上的结果比算法CLPSO, CES, ESLAT和CMA-ES更好。实验结果表明，在单模函数上一些算法的结果比改进人工蜂群算法好；而在多模函数问题上，改进人工蜂群算法的优化性能较其他算法更好。

从表4.5中可以看出，改进人工蜂群算法比GABC算法在6个测试函数上优化性能更好。对于函数Schwefel，改进人工蜂群算法较其他四个人工蜂群算法的结果更好。对于函数Ackley，改进人工蜂群算法比NABC算法好，而与I-ABC和PS-ABC结果类似。对于函数Griewank和Rastrigin，改进人工蜂群算法和I-ABC、PS-ABC都能达到最优值。除了函数Rosenbrock，改进人工蜂群算法在其他函数上实验结果

比NABC和I-ABC更好.

表 4.5 改进人工蜂群算法与其他人工蜂群算法结果比较

Fun	GABC	I-ABC	PS-ABC	NABC	IABC
	Mean	Mean	Mean	Mean	Mean
Ackley	7.78e-10	8.88e-16	8.88e-16	1.07e-13	8.88e-16
Griewank	6.96e-04	0.00e+00	0.00e+00	1.11e-16	0.00e+00
Rastrigin	3.31e-02	0.00e+00	0.00e+00	0.00e+00	0.00e+00
Rosenbrock	7.48e+00	2.64e+01	1.59e+00	1.45e-01	6.75e+00
Schwefel	1.62e+02	3.18e+02	5.30e+00	5.73e-01	3.82e-04
Sphere	6.26e-16	0.00e+00	0.00e+00	5.43e-16	2.39e-18

4.4 小结

人工蜂群算法是一种新型的群智能优化技术，体现出较好的优化性能。然而，该算法和其他智能优化技术类似，都具有收敛速度慢和易陷入局部最优的缺点。主要原因是该算法的搜索模式具有较好的探索能力，但开发能力较弱。为了克服这些缺点，本章提出一种改进人工蜂群算法。通过对测试函数的测试和比较研究，说明本章中所提出的改进人工蜂群算法不论在准确性、收敛速度、稳定性及鲁棒性上都比人工蜂群算法更优越。与其他的智能优化算法比较分析说明了本章中提出的改进人工蜂群算法优化性能更优，稳定性更好。

第五章 非负线性最小二乘问题的有效人工蜂群算法

对于非负线性最小二乘问题，本章提出一种有效人工蜂群算法。为了提高算法的优化性能，该算法利用正交初始化方法生成初始种群，提高初始种群分布的均匀性。同时，为了平衡算法的探索能力和开发能力，设计一种新型的搜索模式。通过27个benchmark测试函数的实验测试，以及与其他算法的比较研究，表明该算法具有较好的优化性能。进一步，将该算法应用于非负线性最小二乘问题。通过对5个标准非负线性最小二乘测试问题的测试，并与其他算法比较，说明本章提出的算法对该问题是可行有效的。

5.1 引言

最小二乘方法是一种可以对确定性系统求出近似解的较好方法^[152]。最小二乘问题可根据残差是否是线性的分为两类：线性最小二乘和非线性最小二乘^[153]。在应用数学和统计学当中，线性最小二乘是一种对统计数据进行拟合的方法。这种方法可以通过模型和数据的拟合，找到最适应于数据的模型参数，对数据给出理想的数学模型，进而对未来数据的变化进行预测，还可以用于理解系统的结构^[154]。一般情况下，非负线性最小二乘问题(Nonnegative Linear Least Squares Problem, NLLS)如下式构造：

$$\min_{x \geq 0} f(x) = \frac{1}{2} \|Ax - b\|^2 = \frac{1}{2} (Ax - b)^T (Ax - b)$$

其中， $A \in R^{m \times n}$, $m \geq n$, $\text{rank}(A) = n$, $b \in R^m$ 。

过去几十年中，对于求解线性最小二乘问题通常使用传统优化方法^[155]。在使用传统优化方法时，要求问题中的矩阵满足可替换或可分解。同时，随着问题维数的增加，矩阵规模增大，增加了计算的复杂性。传统优化算法对于初始点的选取具有较高的灵敏性，并且收敛速度慢，易陷入局部最优。因此，近年来，有学者考虑构造适用于非负线性最小二乘问题的智能优化算法来解决这一问题。

2005年，受蜂群觅食特性的启发，学者Karaboga对于无约束问题提出了人工蜂群算法^[1]。自然界中蜂群通常由三类蜜蜂构成：雇佣蜂、跟随蜂和侦查蜂。对于人工蜂群来说，也由这三种人工蜜蜂构成。人工蜂群中一半的蜜蜂是雇佣蜂，另一半是跟随蜂。雇佣蜂负责探索蜜源位置，通过摇摆舞将蜜源信息传递给其他蜜蜂；跟随蜂根据摇摆舞信息随机选择蜜源进行开发；而当雇佣蜂对应的蜜源被丢弃时，

该蜜蜂则转变为侦查蜂，负责开发新蜜源替代丢弃蜜源。算法中每个蜜源代表一个问题可行解，蜜源的蜜量表示解的好坏。

与其他的智能优化算法相比，人工蜂群算法具有较好的优化性能，并且已经被用于很多实际问题的求解当中^[80-95]。因此，本章通过结合非负线性最小二乘问题特点和人工蜂群算法的优点，设计一种有效人工蜂群算法来求解非负线性最小二乘问题。通过对27个benchmark测试函数和一组非负线性最小二乘问题的实验，对比其他算法，本章提出的有效人工蜂群算法是有效可行的算法。

本章主要安排如下：5.2节给出有效人工蜂群算法的具体步骤；5.3节是实验及结论分析；5.4节对本章进行小结。

5.2 有效人工蜂群算法

5.2.1 正交初始化

种群初始化是仿生优化算法中较为重要的一个步骤，它会影响到解的质量和算法的收敛速度。正交试验是统计学中的一种试验方法，本章中使用基于正交数组和量化技术的种群初始化方法生成初始种群。这种初始化方法可以使初始解均匀分布在可行域内，提高算法搜索最优解的速度^{[124][146]}。种群初始化算法流程如下。

Algorithm 5.1： 种群初始化

Step 1： 将问题可行域 $[l, u]$ 划分成 S 个子集 $[l_1, u_1], [l_2, u_2], \dots, [l_S, u_S]$ ，具体划分方法如下：

$$\begin{cases} l_i = l + (i-1) \left(\frac{u(s) - l(s)}{S} \right) l_s, \\ u_i = u - (S-i) \left(\frac{u(s) - l(s)}{S} \right) l_s, \end{cases} \quad i = 1, 2, \dots, S. \quad (5-1)$$

其中， $u(s) - l(s) = \max_{1 \leq i \leq D} \{u_i - l_i\}$ 。

Step 2： 将子集 $[l_i, u_i]$ 进行 Q 次量化如下：

$$\alpha_{ij} = \begin{cases} l_i, & j=1, \\ l_i + (j-1) \left(\frac{u_i - l_i}{Q_i - 1} \right), & 2 \leq j \leq Q_i - 1, \\ u_i, & j=Q_i, \end{cases} \quad (5-2)$$

其中 Q 是奇数。然后，构造正交数组 $L_{M_1}(Q_i^N) = [a_{ij}]_{M_1 \times N}$ ，根据(5-3)式

$$\left\{ \begin{array}{l} (\alpha_{1,a_1}, \alpha_{2,a_2}, \dots, \alpha_{N,a_N}) \\ (\alpha_{1,a_{21}}, \alpha_{2,a_{22}}, \dots, \alpha_{N,a_{2N}}) \\ \dots \\ (\alpha_{1,a_{M_11}}, \alpha_{2,a_{M_12}}, \dots, \alpha_{N,a_{M_1N}}). \end{array} \right. \quad (5-3)$$

选择 M_1 个个体，其中 $L_{M_1}(Q_i^N)$ 可按如下方法进行构造：选取满足 $(Q_i^k - 1)/(Q_i - 1) \geq N$ 的最小 J_1 。若 $(Q_i^k - 1)/(Q_i - 1) = N$ ，则 $N' = N$ ，否则 $N' = (Q_i^k - 1)/(Q_i - 1)$ 。构造数组的基本列 $j = \frac{Q_i^{k-1} - 1}{Q_i - 1} + 1$ ， $a_{ij} = \left\lfloor \frac{i-1}{Q_i^{J_1-k}} \right\rfloor \bmod Q_i$ ， $i = 1, \dots, M_1$ ， $k = 1, \dots, J_1$ ；非基本列为 $j = \frac{Q_i^{k-1} - 1}{Q_i - 1} + 1$ ， $a_{j+(s-1)(Q_i-1)+t} = (a_s \times t + a_j) \bmod Q_i$ ， $s = 1, \dots, j-1$ ， $t = 1, \dots, Q_i$ 。这样数组 $L_{M_1}(Q_i^N)$ 就构造完成。删除 $L_{M_1}(Q_i^N)$ 的最后 $N' - N$ 列就得到了 $L_{M_1}(Q_i^N)$ ，其中 $M_1 = Q_i^N$ 。

Step 3：在 MS 个个体中，选取 SN 个适应度最高的个体作为初始种群。

5.2.2 新搜索方程

如何平衡算法的探索能力和开发能力来寻求最优解，对于群智能算法来说是一个非常重要的问题。算法的探索能力是指算法在解空间内搜索未知区域寻求最优解的一种能力；而开发能力则指对当前最优解进行局部搜索来发现更好解的能力。这两种能力通常情况下不易同时兼顾，因此需要对这两种搜索能力进行较好的均衡。在人工蜂群算法中，搜索方程在寻优过程中具有较强的探索能力，开发能力较差。为了提高算法的优化性能，增强算法的开发能力，本节给出一种新的搜索机制。

差分进化算法是一种基于种群的函数优化方法，该算法的主要搜索策略是通过计算个体与种群中其它随机抽取个体间的差分来生成新的个体。在很多优化问题和现实应用问题中，该算法都体现出了良好的优化性能。差分进化算法中包括变异、交叉和选择三种进化算子。其中，变异算子的形式有很多种，这就构成了不同的差分进化算法。在这些变异算子中，“DE/best/1”可以提高算法局部搜索能力，具体方程如下：

$$\text{DE/best/1: } v_i = x_{best} + F(x_{r1} - x_{r2}), \quad (5-4)$$

其中, $i \in \{1, 2, \dots, SN\}$; $r1$ 和 $r2$ 是 $\{1, 2, \dots, SN\}$ 中两个不同的随机整数; x_{best} 是当前全局最优解; $F \in [0.5, 1]$ 是一个随机正实数.

受差分进化算法启发, 结合人工蜂群算法特点, 这里提出一种新的搜索方程, 如下:

$$v_{i,j} = x_{best,j} + \varphi_{i,j}(x_{i,j} - x_{r1,j}), \quad (5-5)$$

其中, $i \in \{1, 2, \dots, SN\}$; $r1$ 是 $\{1, 2, \dots, SN\}$ 中随机选择且与 i 不同的整数; $x_{best,j}$ 表示当前全局最优解; $j \in \{1, 2, \dots, D\}$ 是随机选的下标; $\varphi_{ij} \in [0, 1]$ 是均匀分布的随机数. 与差分进化算法类似, 搜索方程(5-5)可以提高算法的收敛速度, 也即增强算法的开发能力.

为了提高人工蜂群算法的开发能力, 受粒子群优化算法启发, 结合粒子群优化算法搜索方程的优点, Zhu等^[27]学者提出g-best人工蜂群算法. 该算法中, 改进的搜索方程如下:

$$v_j = x_j + \phi_j(x_j - x_k) + \varphi_j(y_j - x_j), \quad (5-6)$$

其中, $k \in \{1, 2, \dots, SN\}$ 是不同于 i 的随机选择的指标; $j \in \{1, 2, \dots, D\}$ 是随机选取的整数; y_j 是当前全局最优解的第 j 维; $\phi_j \in [-1, 1]$ 和 $\varphi_j \in [0, 1.5]$ 是均匀分布的随机数. 这一搜索方程同样可以提高算法的开发能力.

5.2.3 有效人工蜂群算法

基于以上分析, 为了更好的结合搜索方程(5-5)和搜索方程(5-6)的优势, 有效人工蜂群算法中将对两个搜索方程进行混合. 在有效人工蜂群算法中, 将通过选择概率 P 来控制搜索方程(5-5)和搜索方程(5-6)的混合. 因此, 有效人工蜂群算法的主要步骤如下所示.

Algorithm 5.2: 有效人工蜂群算法

1. 初始化: 设定选择概率 P 和初始种群规模 SN .
2. 应用 Algorithm 5.1 生成一个初始种群, 并计算适应度值.
3. While($FE < \text{Max. } FE$) do
4. for $i = 1$ to SN do
5. 应用搜索方程(5-5)产生一个新解 v_i .
6. If $f(v_i) < f(x_i)$ then $x_i = v_i$

```

7.     else then
8.         if  $rand(0,1) < P$  then
9.             应用搜索方程(5-6)产生新解  $v_i$ .
10.            if  $f(v_i) < f(x_i)$  then  $x_i = v_i$ 
11.            end if
12.        end if
13.    end if
14. end for
15. end while (FE= Max. FE)

```

5.3 数值实验及比较

为了验证有效人工蜂群算法的有效性，本节将对27个benchmark函数进行测试，并对5个典型非负线性最小二乘测试问题进行实验。

5.3.1 对 27 个 benchmark 测试函数实验

本节选取27个benchmark测试函数对有效人工蜂群算法的优化性能进行验证，见表5.1-5.2，表中C表示函数特征，U表示单模函数，M表示多模函数，S表示函数可分，N表示函数不可分。表5.1中第三列给出测试函数的维数。表5.2中的测试函数维数均为 $D = 30$ 和 $D = 60$ ，且表达式中 $n = D$ 。

测试函数Hartman3中，

$$\alpha = \begin{bmatrix} 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}^T,$$

$$P = \begin{bmatrix} 0.3689 & 0.1170 & 0.2673 \\ 0.4699 & 0.4387 & 0.7470 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{bmatrix}^T.$$

测试函数Shekel中，

$$\alpha = \begin{bmatrix} 4.0 & 1.0 & 8.0 & 6.0 & 3.0 & 2.0 & 5.0 & 8.0 & 6.0 & 7.0 \\ 4.0 & 1.0 & 8.0 & 6.0 & 7.0 & 9.0 & 5.0 & 1.0 & 2.0 & 3.6 \\ 4.0 & 1.0 & 8.0 & 6.0 & 3.0 & 2.0 & 3.0 & 8.0 & 6.0 & 7.0 \\ 4.0 & 1.0 & 8.0 & 6.0 & 7.0 & 9.0 & 3.0 & 1.0 & 2.0 & 3.6 \end{bmatrix},$$

$$c = \frac{1}{10} [1, 2, 2, 4, 4, 6, 3, 7, 5, 5]^T.$$

表 5.1 Benchmark 函数 f_1-f_{16}

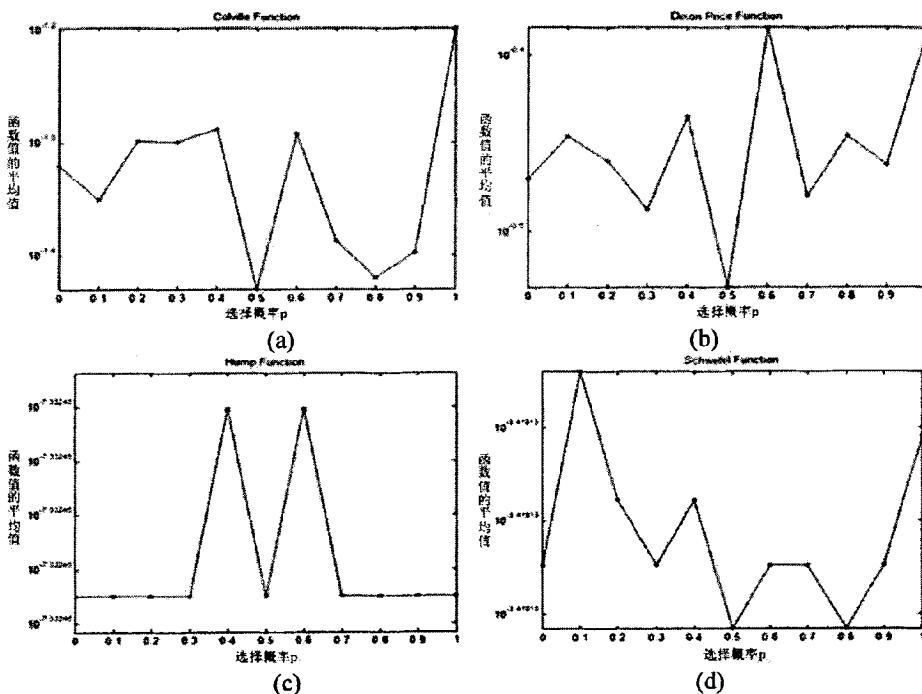
函数	定义域	D	C	表达式
Beale	$[-4.5, 4.5]$	2	UN	$f_1 = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2$
Bohachevsk y	$[-100, 100]$	2	MS	$f_2 = x_1^2 + 2x_2^2 - 0.3 \cos(3\pi x_1) - 0.4 \cos(4\pi x_2) + 0.7$
Booth	$[-10, 10]$	2	MS	$f_3 = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$
Branin	$[-5, 10] \times [0, 15]$	2	MS	$f_4 = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) \cos x_1 + 10$
Colville	$[-10, 10]$	4	UN	$f_5 = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 + 10.1((x_2 - 1)^2 + (x_4 - 1)^2)$
Easom	$[-100, 100]$	2	UN	$f_6 = -\cos x_1 \cos x_2 \exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2)$
GoldStein- Price	$[-2, 2]$	2	M N	$f_7 = \begin{bmatrix} 1 + (x_1 + x_2 + 1)^2 \\ (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2) \\ [30 + (2x_1 - 3x_2)^2] \\ (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2) \end{bmatrix}$
Hartman3	$[0, 1]$	3	M N	$f_8 = -\sum_{i=1}^4 c_i \exp \left[-\sum_{j=1}^3 a_{ij} (x_j - p_{ij})^2 \right];$ $c = [1.0, 1.2, 3.0, 3.2]$
Six Hump Camel Back	$[-5, 5]$	2	M N	$f_9 = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$
Matyas	$[-10, 10]$	2	UN	$f_{10} = 0.26(x_1^2 + x_2^2) - 0.48x_1 x_2$
Perm	$[-D, D]$	2	M N	$f_{11} = \sum_{k=1}^n \left[\sum_{i=1}^2 (i^k + 0.5)((x_i/i)^k - 1) \right]^2$
Powell	$[-4, 5]$	4	UN	$f_{12} = \sum_{i=1}^{n/k} (x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} + 10x_{4i})^2 + (x_{4i-2} + 10x_{4i})^2 + 10(x_{4i-3} + 10x_{4i})^4$
PowerSum	$[0, D]$	2 4	M N	$f_{13} = \sum_{k=1}^n \left[\sum_{i=1}^4 (x_i^k) - b_k \right]^2; b = [8, 18, 44, 114]$
Shekel	$[0, 10]$	4	M N	$f_{14} = -\sum_{j=1}^m \left[\sum_{i=1}^4 (x_i - a_{ij})^2 + c_j \right]^{-1}$
Shubert	$[-10, 10]$	2	M N	$f_{15} = \left(\sum_{i=1}^5 i \cos((i+1)x_1 + i) \right) \cdot \left(\sum_{i=1}^5 i \cos((i+1)x_2 + i) \right)$
Trid6	$[-36, 36]$	6	UN	$f_{16} = \sum_{i=1}^n (x_i - 1)^2 - \sum_{i=2}^n x_i x_{i-1}$

表 5.2 Benchmark 函数 f_{17} - f_{27}

Function	Range	C	Formulation
Ackley	[-32,32]	MN	$f_{17} = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i)\right) + 20 + e$
Dixon-Price	[-10,10]	UN	$f_{18} = (x_1 - 1)^2 + \sum_{i=2}^n i(2x_i^2 - x_{i-1})^2$
Griewank	[-600,600]	MN	$f_{19} = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
			$f_{20} = \sin^2(\pi y_1) + \sum_{i=1}^{n-1} [(y_i - 1)^2 (1 + 10 \sin^2(\pi y_i + 1))]$
Levy	[-10,10]	MN	$+ (y_n - 1)^2 (1 + 10 \sin^2(2\pi y_n))$ $y_i = 1 + \frac{x_i - 1}{4}, \quad i = 1, \dots, n.$
Michalewicz	[0, π]	MS	$f_{21} = -\sum_{i=1}^n \sin(x_i) (\sin(ix_i^2/\pi))^{2m}, m = 10$
Rastrigin	[-5.12,5.12]	MS	$f_{22} = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10]$
Rosenbrock	[-30,30]	UN	$f_{23} = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
Schwefel	[-500,500]	MS	$f_{24} = \sum_{i=1}^n -x_i \sin(\sqrt{ x_i })$
Sphere	[-100,100]	US	$f_{25} = \sum_{i=1}^n x_i^2$
SumSquares	[-10,10]	US	$f_{26} = \sum_{i=1}^n ix_i^2$
Zakharov	[-5,10]	UN	$f_{27} = \sum_{i=1}^n x_i^2 + \left(\sum_{i=1}^n 0.5ix_i\right)^2 + \left(\sum_{i=1}^n 0.5ix_i\right)^4$

本小节中将对新算法中的选择概率 P 的影响进行讨论。在27个benchmark测试函数中测试函数Colville, Dixon-Price, Six Hump Camel Back和Schewefel具有一定的代表性，因此选择概率 P 的测试将在这四个函数上进行。对改进人工蜂群算法独立运行30次，最优值的30次均值图形如图5.1。因为所有的测试函数均是最小化问题，因此均值越小，则结果越好。

从图5.1中可以看到，选择概率 P 确实对实验结果有影响。当 P 约等于0.5时，得到的实验结果是最好的。所以，对其他测试函数的数值实验中将都使用 $P = 0.5$ 。

图 5.1 四个测试函数在不同选择概率 P 下的结果图

为了说明有效人工蜂群算法的优化效果，本节将对原始人工蜂群算法与有效人工蜂群算法进行比较实验。实验中，原始人工蜂群算法与有效人工蜂群算法的参数设定相同，种群规模 SN 、 $limit$ 和最大循环代数（ MSN ）分别设定为 60、 $(SN/2)*D$ 、2000 ($D=30$) 及 4000 ($D=60$)。所有实验均独立重复 30 次。

表 5.3 和表 5.4 给出了实验结果，表中 **best**、**worst**、**mean** 和 **std** 分别表示 30 次独立实验中的最好解、最差解、均值和标准差。最好的结果均用黑体标注出来。

如表 5.3 和表 5.4 所示，有效人工蜂群算法的函数最优值均值均等于或较人工蜂群算法更接近于测试函数的最优值，并且误差较人工蜂群算法更小。特别指出，除了函数 f_1 、 f_{18} 以及 $D=60$ 时函数 f_{23} ，在其他函数上，有效人工蜂群算法比人工蜂群算法的优化性能更好；在函数 f_2 和 f_4 上两个算法优化性能相同，都能达到最优值。这些比较结果都说明了有效人工蜂群算法不论在单模问题还是多模问题上优化性能都比原始人工蜂群算法更好。

为了更进一步测试改进人工蜂群算法的性能，将有效人工蜂群算法与其他群智能算法，包括一些其他的改进人工蜂群算法进行比较研究。比较算法包括 DE [5]，PSO [5]，CLPSO [146]，CES [147]，FES [148]，ESLAT [149]，CMA-ES [149]，GABC [27]，I-ABC [150]，PS-ABC [150] 和 NABC [151]。其中 DE 和 PSO 的参数设定和文献 [5] 相同，剩余其他算法的参数设定均按原始论文中的设定给出。

表 5.3 改进人工蜂群算法及人工蜂群算法实验结果

	ABC				EH-ABC			
	Best	Mean	Worst	Std	Best	Mean	Worst	Std
f_1	7.62e-11	2.49e-09	2.75e-08	5.03e-09	1.58e-10	1.91e-04	4.61e-03	8.38e-04
f_2	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00
f_3	1.43e-19	7.74e-18	2.34e-17	6.69e-18	0.00e+00	0.00e+00	0.00e+00	0.00e+00
f_4	3.98e-01	3.98e-01	3.98e-01	0.00e+00	3.98e-01	3.98e-01	3.98e-01	0.00e+00
f_5	4.29e-02	2.26e-01	4.97e-01	1.389e-01	8.72e-03	4.53e-02	1.26e-01	3.02e-02
f_6	-1	-0.99997	-0.99981	5.48e-05	-1	-1	-1	0.00e+00
f_7	3	3.000014	3.000257	4.97e-05	3	3	3	1.75e-15
f_8	-3.86278	-3.86278	-3.86278	2.32e-15	-3.86278	-3.86278	-3.86278	2.71e-15
f_9	4.65e-08	4.65e-08	4.65e-08	1.03e-16	4.65e-08	4.65e-08	4.65e-08	0.00e+00
f_{10}	6.99e-17	8.26e-14	2.09e-12	3.82e-13	1.74e-16	2.99e-14	2.16e-13	4.63e-14
f_{11}	8.30e+77	1.13e+83	1.13e+84	2.57e+83	2.27e+77	1.31e+81	8.12e+81	2.16e+81
f_{12}	6.17e-07	2.68e-05	5.31e-05	1.59e-05	2.43e-08	1.08e-05	4.65e-05	1.15e-05
f_{13}	7.21e-04	1.67e-02	4.74e-02	1.24e-02	1.93e-05	1.17e-02	3.94e-02	1.15e-02
f_{14}	-10.5364	-10.5364	-10.5364	3.43e-15	-10.5364	-10.5364	-10.5364	2.14e-15
f_{15}	-186.731	-186.731	-186.731	5.28e-15	-186.731	-186.731	-186.731	2.79e-14
f_{16}	-50	-50	-50	9.27e-11	-50	-50	-50	1.50e-06

表5.5给出了DE, PSO, CES, FES, ESLAT, CMA-ES和改进人工蜂群算法的比较结果, 其中DE, PSO, CES, FES, ESLAT和CMA-ES的实验结果来自于文献[5]. 表5.6给出改进人工蜂群算法与其他四种人工蜂群算法的比较结果, 其他四种人工蜂群算法结果来自于文献[151]. 表5.5和表5.6中的最优结果均用黑体标注.

从表5.5中可以看出, 有效人工蜂群算法在7个测试函数上比其他算法结果更好. 实验结果表明, 不论是对单模函数还是多模函数, 有效人工蜂群算法的优化性能较其他算法更好.

表5.6中可以看出, 有效人工蜂群算法比GABC算法在6个测试函数上优化性能更好. 对于函数Ackley、Rosenbrock和Schwefel, 有效人工蜂群算法较其他四种人工蜂群算法的结果更好. 对于函数Griewank和Rastrigin, 有效人工蜂群算法与I-ABC和PS-ABC结果相同, 且都能达到最优点. 而对于函数Sphere, I-ABC和PS-ABC较有效人工蜂群算法结果更好些. 与NABC算法相比较, 有效人工蜂群算法除了函数Rastrigin和NABC结果相同外, 其他结果都比NABC的结果更好.

表 5.4 改进人工蜂群算法及人工蜂群算法实验结果

D	ABC				EH-ABC			
	Best	Mean	Worst	Std	Best	Mean	Worst	Std
f_{17}	30	0.00e+00	1.66e-15	1.78e-14	3.46e-15	0.00e+00	0.00e+00	0.00e+00
	60	0.00e+00	2.61e-15	1.42e-14	3.22e-15	0.00e+00	0.00e+00	0.00e+00
f_{18}	30	1.86e-05	9.93e-05	7.09e-04	1.24e-04	1.22e-01	3.67e-01	6.67e-01
	60	3.89e-05	1.83e-04	8.55e-04	1.49e-04	6.67e-01	6.67e-01	1.78e-10
f_{19}	30	0.00e+00	7.50e-04	2.25e-02	4.11e-03	0.00e+00	0.00e+00	0.00e+00
	60	0.00e+00	6.85e-16	1.61e-14	2.92e-15	0.00e+00	0.00e+00	0.00e+00
f_{20}	30	3.08e-16	5.06e-16	7.09e-16	7.35e-17	2.14e-31	1.66e-30	5.49e-30
	60	7.77e-16	1.21e-15	1.42e-15	1.53e-16	2.75e-30	9.36e-30	1.75e-29
f_{21}	30	-29.5462	-29.4879	-29.4313	2.97e-02	-29.6309	-29.6285	-29.6212
	60	-59.2538	-59.0873	-58.9513	7.21e-02	-59.5723	-59.5117	-59.4571
f_{22}	30	0.00e+00	3.22e-14	5.68e-14	2.86e-14	0.00e+00	0.00e+00	0.00e+00
	60	0.00e+00	1.74e-13	4.55e-13	1.11e-13	0.00e+00	5.31e-14	1.14e-13
f_{23}	30	3.76e-03	5.12e-02	2.10e-01	5.95e-02	1.49e-02	4.46e-02	8.35e-02
	60	2.15e-03	7.10e-02	3.34e-01	7.70e-02	8.04e-02	1.79e-01	2.60e-01
f_{24}	30	3.82e-04	4.13e-04	1.16e-03	1.44e-04	3.82e-04	3.82e-04	3.82e-04
	60	7.64e-04	7.90e+00	1.18e+02	3.00e+01	7.64e-04	7.64e-04	7.64e-04
f_{25}	30	4.02e-16	5.47e-16	7.07e-16	7.8e-17	4.97e-33	1.37e-32	2.48e-32
	60	1.1e-15	1.28e-15	1.44e-15	1.15e-16	5.24e-32	2.84e-31	5.73e-31
f_{26}	30	3.19e-16	5.25e-16	7.49e-16	9.11e-17	1.23e-31	4.11e-31	9.13e-31
	60	9.54e-16	1.21e-15	1.61e-15	1.68e-16	3.25e-30	1.05e-29	2.06e-29
f_{27}	30	1.58e+02	2.28e+02	2.83e+02	2.94e+01	2.25e+00	4.86e+00	9.79e+00
	60	5.08e+02	6.93e+02	7.89e+02	5.52e+01	2.25e+00	6.08e+00	15.01e+00

表 5.5 改进人工蜂群算法与其他算法比较结果

	Ackley	Griewank	Rastrigin	Rosenbrock	Schwefel	Sphere
	Mean	Mean	Mean	Mean	Mean	Mean
DE	3.99e-08	6.15e-04	1.47e+02	4.71e+03	7.27e+03	3.43e-14
PSO	3.23e-01	1.34e-02	3.85e+01	5.74e+03	4.16e+03	2.13e-16
CLPSO	2.01e-12	6.45e-13	2.57e-11	1.10e+01	1.19e+01	1.89e-19
CES	6.00e-13	6.00e-14	1.34e+01	2.77e+01	4.57e+03	1.70e-26
FES	1.20e-02	3.70e-02	1.60e-01	3.33e+01	1.31e+01	2.50e-04
ESLAT	1.80e-08	1.40e-03	4.65e+00	1.93e+00	1.03e+04	2.00e-17
CMA-ES	6.90e-12	7.40e-04	5.18e+01	4.00e-01	4.93e+03	9.70e-23
EH-ABC	0.00e+00	0.00e+00	0.00e+00	4.46e-02	3.82e-04	1.37e-32

表 5.6 改进人工蜂群算法与其他人工蜂群算法结果比较

Fun	GABC	I-ABC	PS-ABC	NABC	EH-ABC
	Mean	Mean	Mean	Mean	Mean
Ackley	7.78e-10	8.88e-16	8.88e-16	1.07e-13	0.00e+00
Griewank	6.96e-04	0.00e+00	0.00e+00	1.11e-16	0.00e+00
Rastrigin	3.31e-02	0.00e+00	0.00e+00	0.00e+00	0.00e+00
Rosenbrock	7.48e+00	2.64e+01	1.59e+00	1.45e-01	4.46e-02
Schwefel	1.62e+02	3.18e+02	5.30e+00	5.73e-01	3.82e-04
Sphere	6.26e-16	0.00e+00	0.00e+00	5.43e-16	1.37e-32

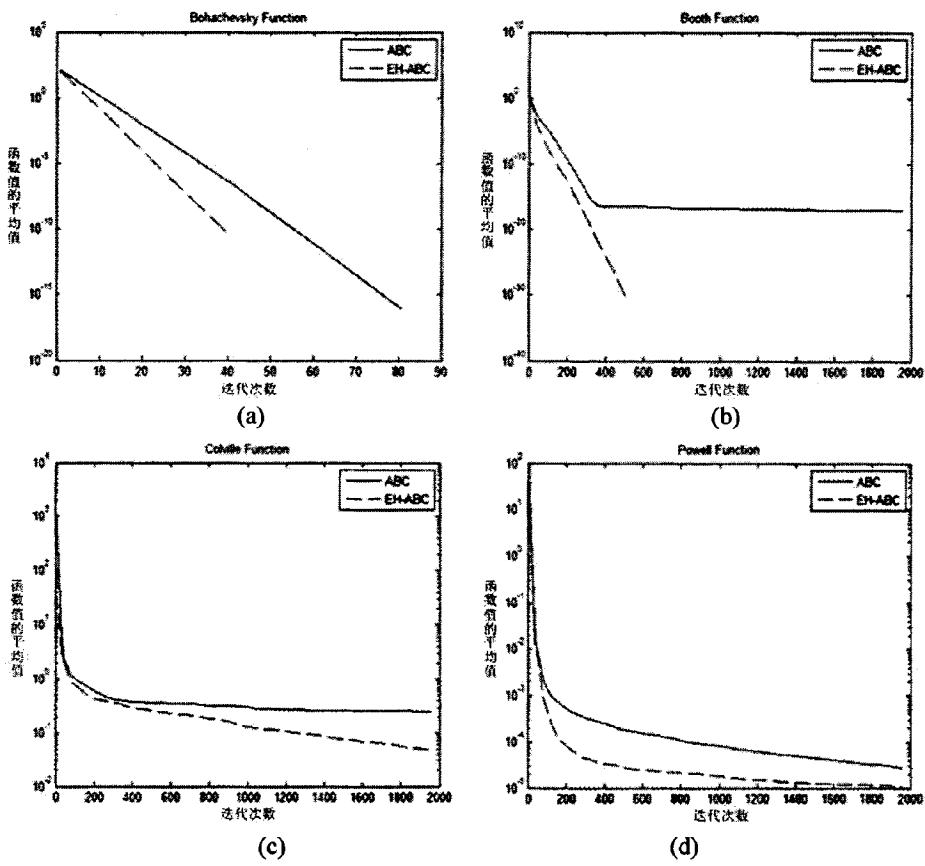
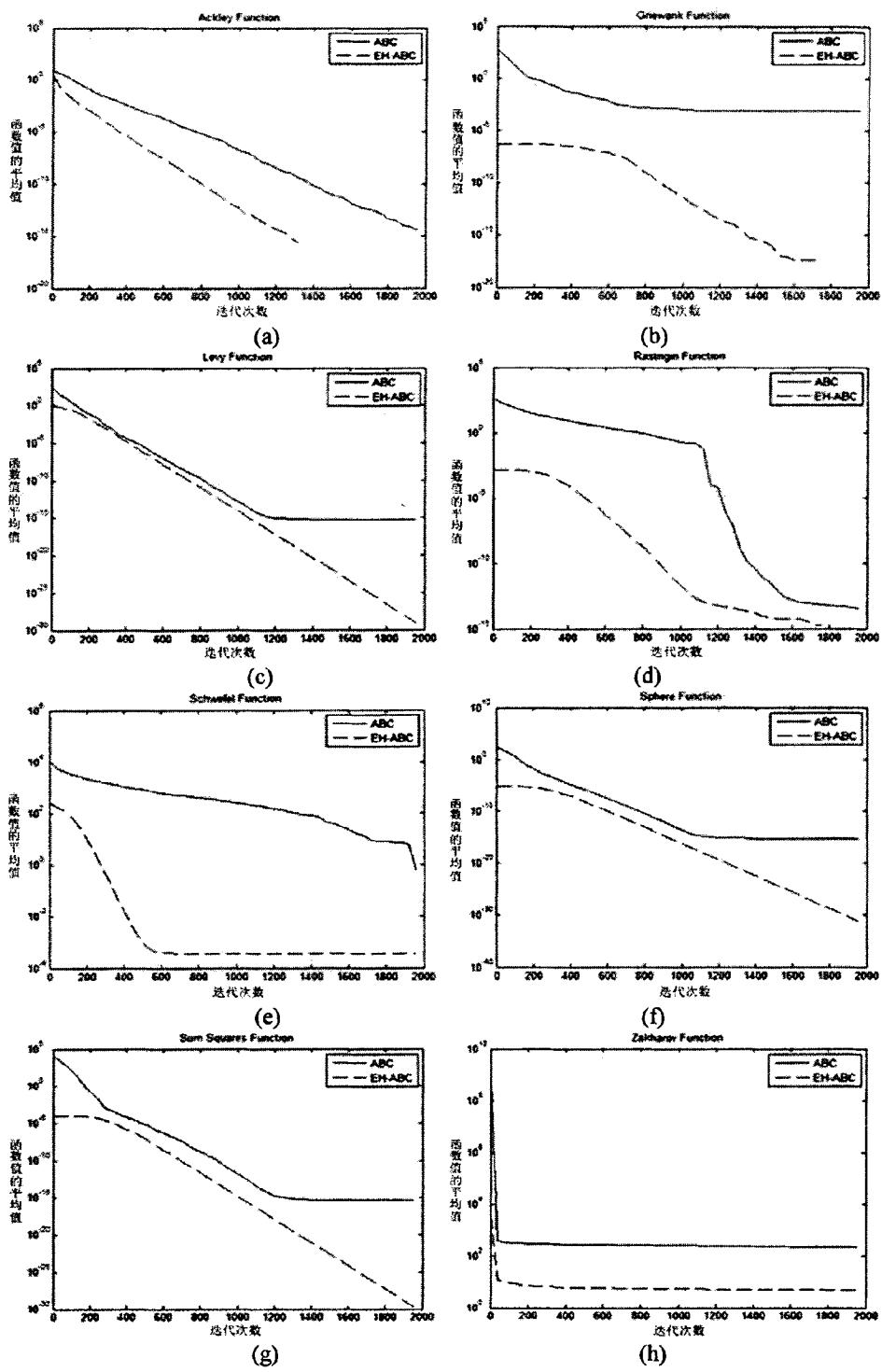


图 5.2 部分函数收敛曲线图

为了更清楚的反映有效人工蜂群算法的优化性能, 图 5.2-图 5.4 给出部分测试函数的收敛曲线图。很明显, 图中显示有效人工蜂群算法比原始人工蜂群算法收敛速度更快。

图 5.3 $D=30$ 时部分函数收敛曲线图

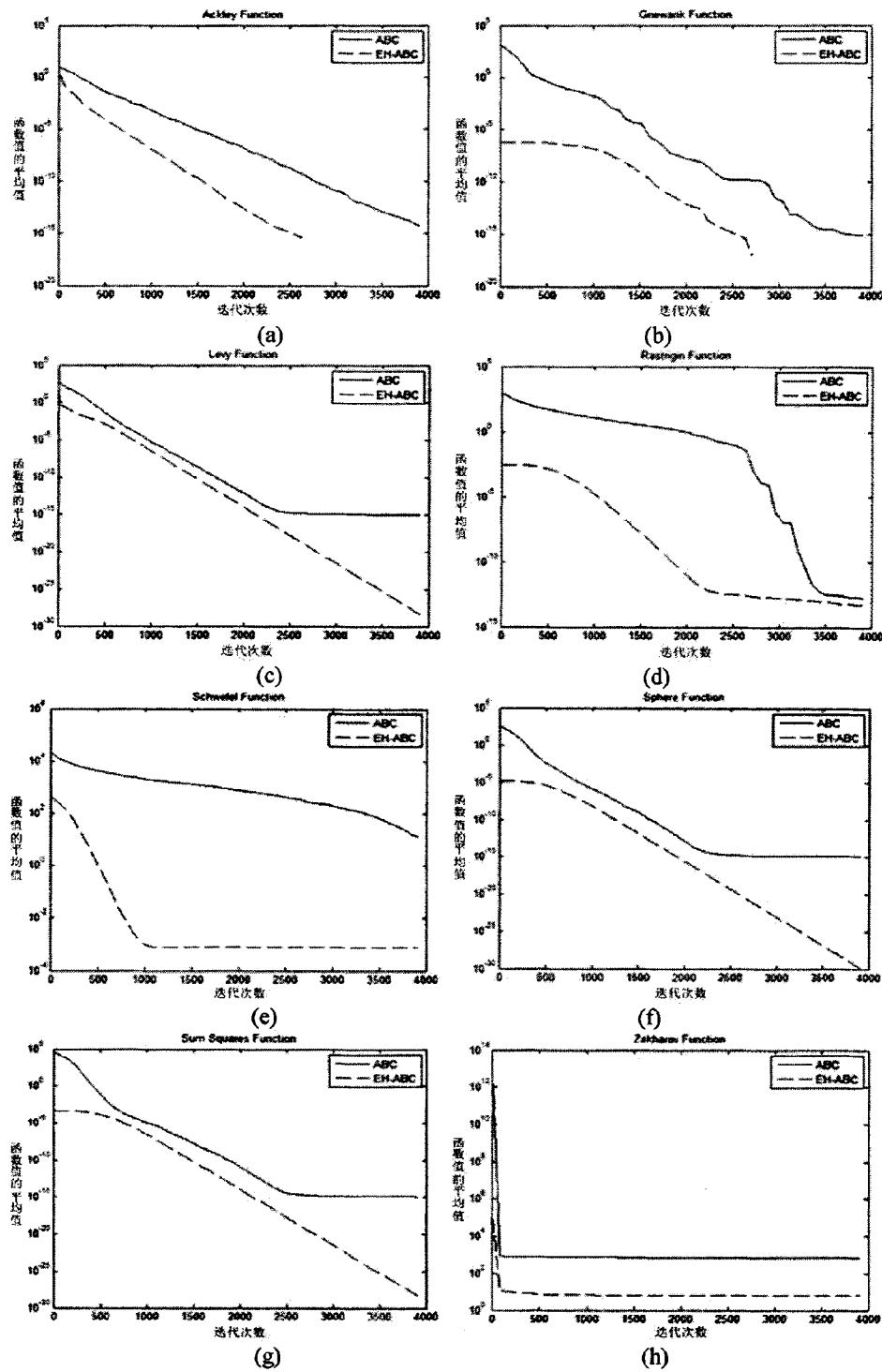


图 5.4 $D = 60$ 时部分函数收敛曲线图

5.3.2 对 NLLS 测试问题实验

本节将对 5 个非负线性最小二乘问题进行实验来验证有效人工蜂群算法在该问题上应用的可行性和有效性。5 个测试问题描述如下：

NLLS 1：考虑如下 NLLS 问题，其中

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 4 & -1 \\ 2 & -2 & 1 \end{bmatrix}, \quad b = \begin{pmatrix} 6 \\ 5 \\ 1 \end{pmatrix},$$

最优解为 $x^* = (1, 1, 3)^T$ 。

NLLS 2：令矩阵 A 的对角元素为 500，非对角元素从区间中随机选取，使得矩阵 A 为对称阵。令 $b = Ae$ ，其中 e 为 $n \times 1$ 的全 1 向量。最优解为 $x^* = (1, 1, \dots, 1)^T \in R^n$ 。

NLLS 3：令矩阵 A 为 $a_{i,i} = 4n$, $a_{i,i+1} = a_{i+1,i} = n$, $a_{i,j} = 0, i = 1, 2, \dots, n$ 。令 $b = Ae$ ，其中 e 为 $n \times 1$ 的全 1 向量。最优解为 $x^* = (1, 1, \dots, 1)^T \in R^n$ 。

NLLS 4：随机生成 NLLS 问题，生成方式如下： $rand('state', 0)$, $m = 200$, $n = 100$, $A = rand(m, n)$, $b = A * ones(n, 1)$ ，其中， $A \in R^{m \times n}$ 。最优解为 $x^* = (1, 1, \dots, 1)^T \in R^n$ 。

NLLS 5：随机生成 NLLS 问题，生成方式如下： $rand('state', 0)$, $n = 100$, $A1 = rand(n, n)$, $A = A1 * A1 + n * eye(n, n)$, $b = A * ones(n, 1)$ ，其中， A 是正定对称矩阵。最优解为 $x^* = (1, 1, \dots, 1)^T$ 。

表 5.7 NLLS 问题的实验结果

Functions	Algorithms	Best	Mean	Worst	Std
NLLS 1	ABC	2.36e-17	2.65e-16	7.49e-16	1.87e-16
	EH-ABC	7.05e-21	1.42e-17	1.38e-16	2.83e-17
NLLS 2	ABC	7.98e+01	2.53e+02	7.47e+02	1.69e+02
	EH-ABC	2.13e+01	3.85e+01	6.21e+01	9.13e+00
NLLS 3	ABC	4.93e+03	1.69e+04	3.77e+04	9.27e+04
	EH-ABC	2.36e+02	2.85e+02	3.61e+02	3.28e+01
NLLS 4	ABC	9.25e+01	1.66e+02	3.06e+02	4.94e+01
	EH-ABC	5.17e-01	9.71e-01	2.08e+00	4.23e-01
NLLS 5	ABC	2.52e+05	4.28e+05	6.52e+05	1.22e+05
	EH-ABC	6.40e+02	1.35e+03	3.43e+03	6.69e+02

为了实验的公平性，有效人工蜂群算法与原始人工蜂群算法设定相同的参数。实验中种群规模 SN 、limit 和最大循环代数(MSN)分别设定为 60 、 $(SN/2)*D$ 、 1000 。所有实验均独立重复 30 次。表 5.7 给出实验比较结果，表中黑体表示较好结果。

为了更明确的说明有效人工蜂群算法的性能, 图 5.5-5.9 给出了收敛曲线图和箱型图。图中可以明确看出, 有效人工蜂群算法不仅对简单非负线性最小二乘问题有效, 对于较复杂的非负线性最小二乘问题也具有较好的优化效果。

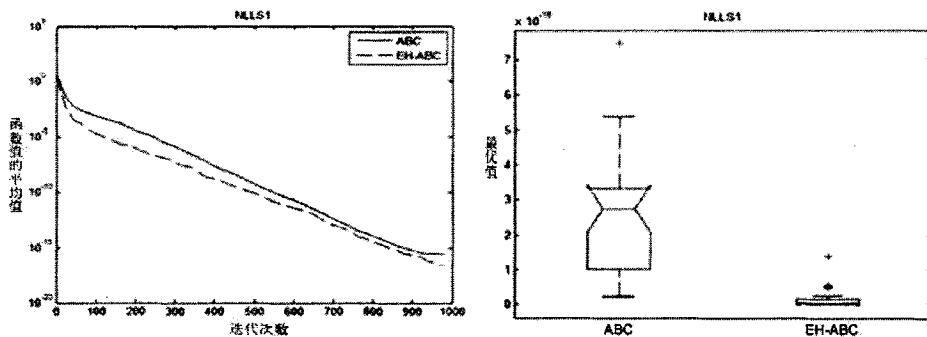
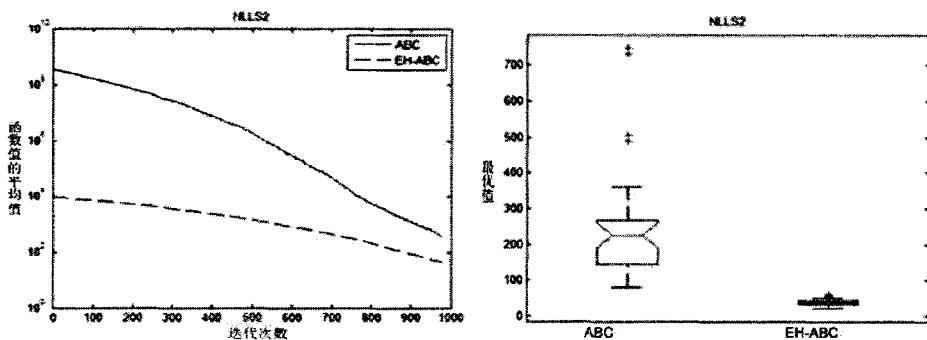
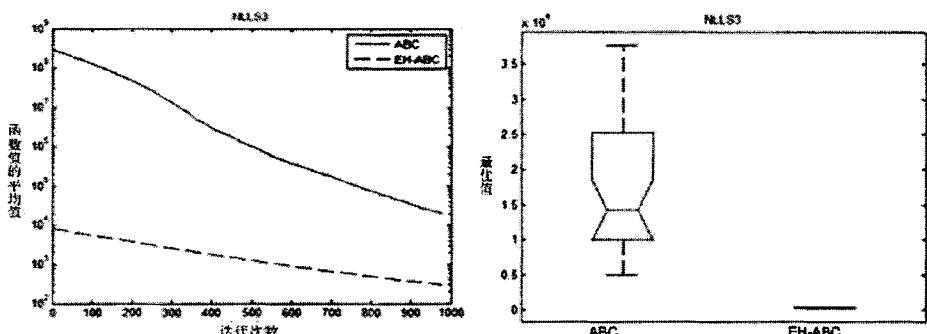
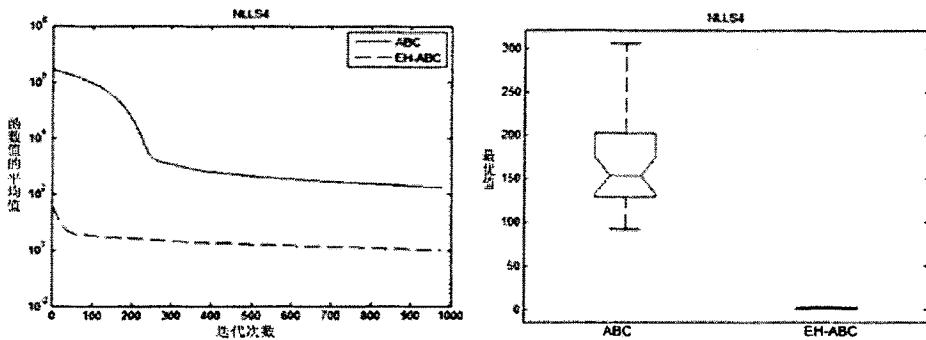
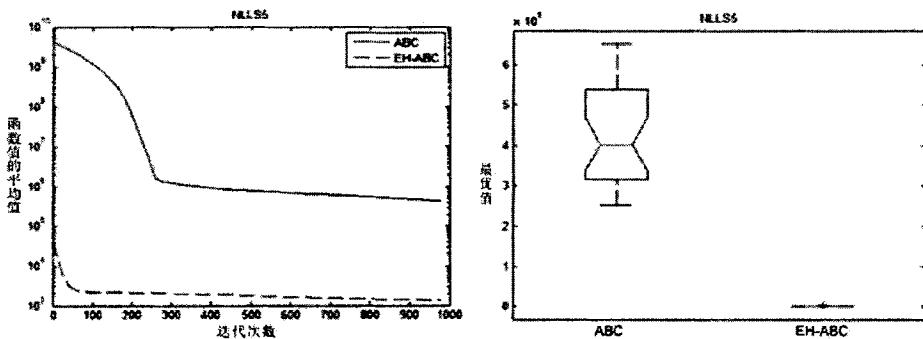


图 5.5 NLLS 1 收敛曲线及箱型图

图 5.6 $n=100$ 时 NLLS 2 收敛曲线及箱型图图 5.7 $n=100$ 时 NLLS 3 收敛曲线及箱型图

图 5.8 $n=100$ 时 NLLS 4 收敛曲线及箱型图图 5.9 $n=100$ 时 NLLS 5 收敛曲线及箱型图

5.4 小结

为了克服人工蜂群算法搜索模式具有较好的探索能力，但开发能力较弱的缺点，本章提出一种有效人工蜂群算法。算法通过引入正交初始化和设计新的搜索模式来提高算法的优化性能。通过对测试函数的测试验证了算法的性能。与其他的智能优化算法比较分析说明了本章中提出的改进人工蜂群算法优化性能更优，稳定性更好。同时，对 5 个非负线性最小二乘测试问题进行了对比实验，实验结果也优于其他算法，说明有效人工蜂群算法对于非负线性最小二乘问题是可行的有效求解算法。

第六章 非负线性最小二乘问题的混沌人工蜂群算法

本章针对全局优化问题和非负线性最小二乘问题，提出一种有效的混沌人工蜂群算法。为了克服人工蜂群算法的不足，在混沌人工蜂群算法种群初始化阶段利用反学习初始化方法增强种群的分布均匀度。其次，为了提高算法的优化性能，平衡算法的探索与开发能力，提出一种新的搜索机制。最后，为了增强算法的局部搜索能力，在算法中加入混沌局部搜索算子。通过对 27 个测试函数和 5 个非负线性最小二乘测试问题的实验验证了算法的优化性能。比较分析显示，本章所提出的算法是可行有效的。

6.1 引言

2005 年，Karaboga^[1]根据蜜蜂的觅食行为提出一种新的优化方法——人工蜂群算法，用来求解函数全局优化问题。很多科学工程领域的问题都可以构造为全局优化问题。目前，用来求解全局优化问题的人工生命计算方法有很多，比如：遗传算法（GAs），蚁群算法（ACO），差分进化算法（DE），粒子群优化（PSO）等等。与这些算法相比较，人工蜂群算法具有优化参数少，简单易行的特点。由于人工蜂群算法具有这些优点，因此被广泛应用于实际问题中，如：最小生成树问题^[94]、车间调度问题^[87]、反分析问题^[112]、聚类问题^[67]、TSP 问题^[123]等等。

尽管原始人工蜂群算法在实际应用中展现出了较强的优化性能，但仍存在需要改进的不足之处^[5]。众所周知，人工蜂群算法的搜索方程具有较强的探索能力，而开发能力较弱，在一定程度上会影响算法的收敛速度。因此，对于人工蜂群算法的改进主要集中在怎样平衡算法的探索能力和开发能力。

为了克服人工蜂群算法存在的问题，提高算法的优化性能，本章提出一种有效的混沌人工蜂群算法（Effective chaotic artificial bee colony approach, EC-ABC）。该算法中设计了新的搜索机制，同时为了平衡算法的探索能力和开发能力，基于 DE 和 PSO 的搜索策略设计两种搜索方程，通过概率 P 控制两个搜索方程的使用。另外，在初始化阶段，采用反学习初始化生成初始种群，增强种群的多样化程度和均衡程度。通过对 27 个测试函数和 5 个非负线性最小二乘测试问题的实验，说明 EC-ABC 算法较其他 ABC 算法的优化性能更强。

本章主要安排如下：6.2 节给出混沌人工蜂群算法的具体步骤；6.3 节是实验及结论分析；6.4 节对本章进行小结。

6.2 有效混沌人工蜂群算法

6.2.1 反学习初始化

群智能算法初始种群分布的均匀性会直接影响算法的收敛速度和解的质量，因此设计合理的初始化方法对提高算法的优化性能非常重要。算法的初始阶段对解在空间上的分布信息一无所知，这就要求初始种群均匀分布于解空间上，以便算法可以在解空间进行均匀搜索。通常情况下，群智能算法的初始种群都是在搜索空间上随机生成，然后对最优解进行全局搜索，这样就不能保证种群分布的均匀性。然而，反学习方法可以同时生成一个解和这个解的反向解，这样就能保证初始种群均匀分布在搜索空间中^[156]。因此，本章将采用根据文献[156]中的反学习方法生成算法的反学习初始化种群。反学习初始化方法如下所示。

Algorithm 6.1: 种群初始化。

Step 1: 设定种群大小 SN 和个体计数器 $i = 1, j = 1$ 。

Step 2: 根据方程(6-1)随机生成一个种群 $X(SN)$ ：

$$x_i^j = x_{\min}^j + \text{rand}(0, 1)(x_{\max}^j - x_{\min}^j), i = 1, 2, \dots, SN, j = 1, 2, \dots, n \quad (6-1)$$

Step 3: 根据方程(6-2)生成种群 $OX(SN)$ ，其中：

$$\alpha x_{ij} = x_{\min,j} + x_{\max,j} - x_{i,j}. \quad (6-2)$$

Step 4: 在种群 $X(SN)$ 和 $OX(SN)$ 选择 SN 个最小造价个体作为初始种群。

6.2.2 新的搜索方程

众所周知，群智能算法中搜索方程的探索能力和开发能力对于算法来说都非常重要，但两者相互矛盾，提高探索能力必然会降低开发能力。因此如何平衡两种能力来提高算法优化性能对于群智能算法来说是亟待解决的问题。对于人工蜂群算法来说，算法的搜索方程具有较强的探索能力，开发能力较弱。为了提高算法的开发能力，本章给出新的搜索方程。

差分进化算法通常遵循进化算法的三个步骤变异、交叉、选择进行搜索。对于变异过程，差分进化算法中根据变异算子的不同可以形成不同的差分进化算法。在所有的变异算子中，“DE/best/1”具有较强的开发能力，方程如下所示：

$$\text{DE/best/1: } v_i = x_{\text{best}} + F(x_{r_1} - x_{r_2}), \quad (6-3)$$

其中, $i \in \{1, 2, \dots, SN\}$; $r1$ 和 $r2$ 是 $\{1, 2, \dots, SN\}$ 中不同的随机整数指标; x_{best} 表示全局最优解; $F \in [0.5, 1]$ 是一个正整数.

受到“DE/best/1”启发, 这里提出一个新搜索方程如下所示:

$$v_{i,j} = x_{best,j} + \varphi_{i,j}(x_{best,j} - x_{i,j}), \quad (6-4)$$

其中, $i \in \{1, 2, \dots, SN\}$, x_{best} 表示全局最优解; $j \in \{1, 2, \dots, D\}$ 表示一个随机下标; $\varphi_{ij} \in [-1, 1]$ 是均匀分布的随机数. 与“DE/best/1”类似, 搜索方程(6-4)可以增强搜索方程的开发能力, 从而提高算法的收敛速度.

受 PSO 启发, Zhu 等^[27]提出 G-best 人工蜂群算法. 该算法是吸取 PSO 算法搜索方程具有较强开发能力的优点提出的新算法. 本章给出改进搜索方程如下所示:

$$v_j = x_j + \phi_j(x_j - x_{ij}) + \varphi_j(x_{best,j} - x_{ij}), \quad (6-5)$$

其中, $k, l \in \{1, 2, \dots, SN\}$ 是不同于 i 的随机下标; $j \in \{1, 2, \dots, D\}$ 是随机下标; x_{best} 表示全局最优解; $\phi_j \in [-1, 1]$ 和 $\varphi_j \in [0, 1.5]$ 为均匀分布的随机数.

6.2.3 混沌局部搜索

为了进一步提高算法的开发能力, 在雇佣蜂阶段和跟随蜂阶段结束后加入混沌局部搜索. 由于混沌映射具有随机性、遍历性和不规律性, 可在最优解附近进行混沌局部搜索. 在混沌局部搜索中, 设计了一个收缩策略, 使得搜索区域在搜索过程中逐步缩小. 混沌局部搜索步骤如下.

Algorithm 6.2: 混沌局部搜索

Step 1: 选择当前种群中的最优解 x_{best} .

Step 2: 生成 Logistic 混沌变量 ch_i :

$$ch_{i+1} = 4 \times ch_i \times (1 - ch_i); \quad i = 1, \dots, K,$$

其中, $ch_i \in (0, 1)$ 是均匀分布的随机数, 且 $ch_i \neq 0.25, 0.5, 0.75$; K 为混沌序列长度.

Step 3: 将混沌变量 ch_i 映射为混沌向量 CH_i :

$$CH_i = LB + ch_i \times (UB - LB); \quad i = 1, \dots, K,$$

其中, LB 为向量定义域下界; UB 为向量定义域上界.

Step 4: 用下式生成新食物源 V_i :

$$V_i = (1-\lambda) \times x_{best} + \lambda \times CH_i; \quad i = 1, \dots, K,$$

其中, λ 为压缩因子, $\lambda = \frac{Maxcycle - cycle + 1}{Maxcycle}$; $Maxcycle$ 表示最大迭代次数, $cycle$ 为当前迭代次数.

Step 5: 若 V_i 的适应值小于 x_{best} , 用 V_i 替换 x_{best} , 或达到混沌序列最大长度 K 终止局部搜索; 否则转 Step 2.

6.2.4 混沌人工蜂群算法

为了结合搜索方程(6-4)和(6-5)的优势, 在 EC-ABC 算法中引入选择概率 P 来控制两个搜索方程的使用频率. 带有反学习初始化的 EC-ABC 算法步骤如下.

Algorithm 6.3: 有效混沌人工蜂群算法

1. 初始化: 设定选择概率 p 和种群数量 SN .
2. 应用 Algorithm 6.1 生成初始种群, 计算种群的适应度.
3. While($FE < Max. FE$) do
4. for $i = 1$ to SN do
5. 应用方程(6-4)生成新解 v_i .
6. If $f(v_i) < f(x_i)$ then $x_i = v_i$
7. else then
8. if $rand(0,1) < P$ then
9. 应用方程(6-5)生成新解 v_i .
10. if $f(v_i) < f(x_i)$ then $x_i = v_i$
11. end if
12. end if
13. end if
14. end for
15. 应用 Algorithm 6.2 更新解.
16. end while ($FE = Max. FE$)

6.3 数值实验及比较分析

为了验证有效人工蜂群算法的有效性, 本节将对 27 个 benchmark 函数进行测试, 并对 5 个典型非负线性最小二乘测试问题进行实验.

6.3.1 对 27 个 benchmark 测试函数实验

本节选取 27 个 benchmark 测试函数对有效人工蜂群算法的优化性能进行验证, 见表 6.1-6.2, 表中 C 表示函数特征, U 表示单模函数, M 表示多模函数, S 表示函数可分, N 表示函数不可分. 表 6.1 中第三列给出测试函数的维数. 表 6.2 中的测试函数维数均为 $D=30$ 和 $D=60$, 且表达式中 $n=D$.

测试函数 Hartman3 中,

$$\alpha = \begin{bmatrix} 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}^T,$$

$$P = \begin{bmatrix} 0.3689 & 0.1170 & 0.2673 \\ 0.4699 & 0.4387 & 0.7470 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{bmatrix}.$$

测试函数 Shekel 中,

$$\alpha = \begin{bmatrix} 4.0 & 1.0 & 8.0 & 6.0 & 3.0 & 2.0 & 5.0 & 8.0 & 6.0 & 7.0 \\ 4.0 & 1.0 & 8.0 & 6.0 & 7.0 & 9.0 & 5.0 & 1.0 & 2.0 & 3.6 \\ 4.0 & 1.0 & 8.0 & 6.0 & 3.0 & 2.0 & 3.0 & 8.0 & 6.0 & 7.0 \\ 4.0 & 1.0 & 8.0 & 6.0 & 7.0 & 9.0 & 3.0 & 1.0 & 2.0 & 3.6 \end{bmatrix},$$

$$c = \frac{1}{10}[1, 2, 2, 4, 4, 6, 3, 7, 5, 5]^T.$$

在 EC-ABC 算法中, 选择概率 P 控制两个搜索方程的使用频率. 为了测试选择概率 P 的影响, 选择六个测试函数进行测试实验, 这六个函数包括 Dixon-Price、Six Hump Camel Back、Michalewicz、Rastrigin、Sphere 和 Trid6. 对 EC-ABC 独立运行 30 次, 最优值的 30 次均值图形如图 6.1. 因为所有的测试函数均是最小化问题, 因此均值越小, 则结果越好.

从图 6.1 中可以看到, 选择概率 P 确实对实验结果有影响. 当 P 约等于 0.3 时, 得到的实验结果是最好的. 所以, 对其他测试函数的数值实验中将都使用 $P=0.3$.

表 6.1 Benchmark 函数 f_1-f_{15}

Function	Range	D	C	Formulation
Beale	$[-4.5, 4.5]$	2	UN	$f_1 = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2$
Bohachevsky y	$[-100, 100]$	2	MS	$f_2 = x_1^2 + 2x_2^2 - 0.3 \cos(3\pi x_1) - 0.4 \cos(4\pi x_2) + 0.7$
Booth	$[-10, 10]$	2	MS	$f_3 = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$
Branin	$[-5, 10] \times [0, 15]$	2	MS	$f_4 = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) \cos x_1 + 10$
Colville	$[-10, 10]$	4	UN	$f_5 = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 + 10.1((x_2 - 1)^2 + (x_4 - 1)^2)$
Easom	$[-100, 100]$	2	UN	$f_6 = -\cos x_1 \cos x_2 \exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2)$
GoldStein- Price	$[-2, 2]$	2	M N	$f_7 = \begin{bmatrix} 1 + (x_1 + x_2 + 1)^2 \\ (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2) \\ . \\ 30 + (2x_1 - 3x_2)^2 \\ (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2) \end{bmatrix}$
Hartman3	$[0, 1]$	3	M N	$f_8 = -\sum_{i=1}^4 c_i \exp \left[-\sum_{j=1}^3 a_{ij} (x_j - p_{ij})^2 \right];$ $c = [1.0, 1.2, 3.0, 3.2]$
Six Hump Camel Back	$[-5, 5]$	2	M N	$f_9 = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$
Matyas	$[-10, 10]$	2	UN	$f_{10} = 0.26(x_1^2 + x_2^2) - 0.48x_1 x_2$
Powell	$[-4, 5]$	4	UN	$f_{11} = \sum_{i=1}^{n/4} (x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} + 10x_{4i})^2 + (x_{4i-2} + 10x_{4i-1})^4 + 10(x_{4i-3} + 10x_{4i})^4$
PowerSum	$[0, D]$	2 4	M N	$f_{12} = \sum_{k=1}^n \left[\sum_{i=1}^4 (x_i^k) - b_k \right]^2; \quad b = [8, 18, 44, 114]$
Shekel	$[0, 10]$	4	M N	$f_{13} = -\sum_{j=1}^m \left[\sum_{i=1}^4 (x_i - a_{ij})^2 + c_i \right]^{-1}$
Shubert	$[-10, 10]$	2	M N	$f_{14} = \left(\sum_{i=1}^5 i \cos((i+1)x_1 + i) \right) \cdot \left(\sum_{i=1}^5 i \cos((i+1)x_2 + i) \right)$
Trid6	$[-36, 36]$	6	UN	$f_{15} = \sum_{i=1}^n (x_i - 1)^2 - \sum_{i=2}^n x_i x_{i-1}$

表 6.2 Benchmark 函数 f_{16} - f_{27}

Function	Range	C	Formulation
Ackley	[-32,32]	M N	$f_{16} = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i)\right) + 20 + e$
Dixon-Price	[-10,10]	UN	$f_{17} = (x_1 - 1)^2 + \sum_{i=2}^n i(2x_i^2 - x_{i-1})^2$
Griewank	[-600,600]	M N	$f_{18} = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
Levy	[-10,10]	M N	$f_{19} = \sin^2(\pi y_1) + \sum_{i=1}^{n-1} \left[(y_i - 1)^2 (1 + 10 \sin^2(\pi y_i + 1)) \right] + (y_n - 1)^2 (1 + 10 \sin^2(2\pi y_n))$ $y_i = 1 + \frac{x_i - 1}{4}, \quad i = 1, \dots, n.$
Michalewicz	[0, π]	MS	$f_{20} = -\sum_{i=1}^n \sin(x_i) \left(\sin(ix_i^2 / \pi) \right)^{2m}, m = 10$
Perm	[-D, D]	M N	$f_{21} = \sum_{k=1}^n \left[\sum_{i=1}^2 (i^k + 0.5)((x_i/i)^k - 1) \right]^2$
Rastrigin	[-5.12,5.12]	MS	$f_{22} = \sum_{i=1}^n \left[x_i^2 - 10 \cos(2\pi x_i) + 10 \right]$
Rosenbrock	[-30,30]	UN	$f_{23} = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
Schwefel	[-500,500]	MS	$f_{24} = \sum_{i=1}^n -x_i \sin(\sqrt{ x_i })$
Sphere	[-100,100]	US	$f_{25} = \sum_{i=1}^n x_i^2$
SumSquares	[-10,10]	US	$f_{26} = \sum_{i=1}^n ix_i^2$
Zakharov	[-5,10]	UN	$f_{27} = \sum_{i=1}^n x_i^2 + \left(\sum_{i=1}^n 0.5ix_i \right)^2 + \left(\sum_{i=1}^n 0.5ix_i \right)^4$

为了说明 EC-ABC 算法的优化效果, 本节将对原始人工蜂群算法与 EC-ABC 算法进行比较实验。实验中, 原始人工蜂群算法与 EC-ABC 算法的参数设定相同。种群规模 SN 、limit 和最大循环代数(MSN)分别设定为 60 、 $(SN/2)*D$ 、 2000 ($D=30$) 及 4000 ($D=60$)。所有实验均独立重复 30 次。

表 6.3 和表 6.4 给出了实验结果, 表中 best、worst、mean 和 std 分别表示 30 次独立实验中的最好解、最差解、均值和标准差。最好的结果均用黑体标注出来。

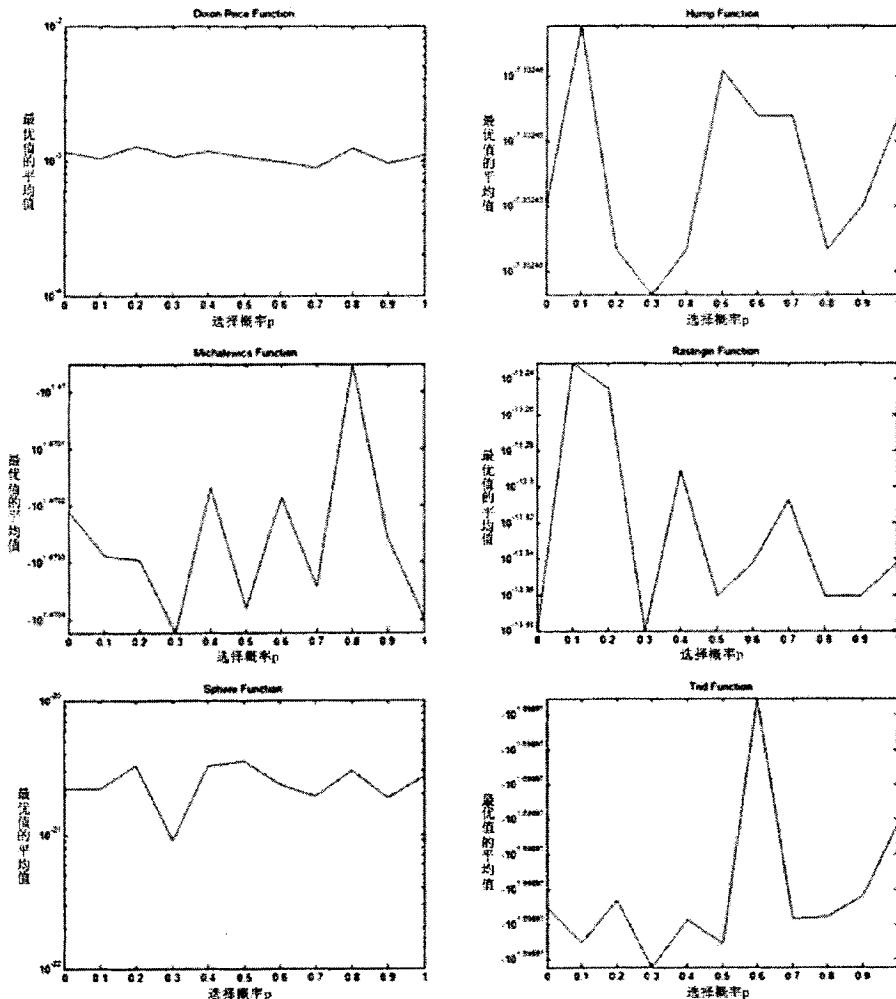
图 6.1 四个测试函数在不同选择概率 P 下的结果图

表 6.3 给出固定维数测试函数比较结果,结果显示 EC-ABC 在测试函数 Booth (f_3)、Easom (f_6)、Powell (f_{11}) 和 PowerSum (f_{12}) 上测试结果优于 ABC 算法; 在测试函数 Bohachevsky (f_2)、Branin (f_4)、GoldStein-Price (f_7)、Six HumpCamel Back (f_9) 和 Shekel (f_{13}) 上与 ABC 算法具有相同的均值,但标准差较 ABC 算法更优; 在测试函数 Hartman3 (f_8)、Shubert (f_{14}) 和 Trid6 (f_{15}) 上与 ABC 算法具有相同的均值,但 ABC 算法的标准差更优; 在测试函数 Beale (f_1)、Colville (f_5) 和 Matyas (f_{10}) 上, ABC 算法的结果优于 EC-ABC 算法. 以上结果表明, 总体来说对于固定维数测试函数 EC-ABC 算法的优化性能优于 ABC 算法.

为了进一步说明表 6.3 中的实验结果, 图 6.2 给出了测试函数 Bohachevsky (f_2)、Booth (f_3)、Powell (f_{11}) 和 PowerSum (f_{12}) 的收敛曲线图以及箱型图. 从图中显然可以看出, EC-ABC 算法较 ABC 算法收敛速度更快, 稳定性更强.

表 6.3 f_1-f_{15} EC-ABC 算法及人工蜂群算法实验结果

	ABC				EC-ABC			
	Best	Mean	Worst	Std	Best	Mean	Worst	Std
f_1	7.08E-08	4.55E-06	2.01E-05	5.08E-06	1.28E-10	3.65E-05	8.21E-04	1.5E-04
f_2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_3	3.78E-19	9.75E-18	5.13E-17	1.09E-17	0.00E+00	2.63E-32	7.89E-31	1.44E-31
f_4	0.397887	0.397887	0.397887	0.00E+00	0.397887	0.397887	0.397887	0.00E+00
f_5	-2.7E+16	-8.9E+14	-9.37E+05	4.85E+15	-2E+157	-7E+155	-4E+123	6.55E+04
f_6	-1	-0.99982	-0.99863	3.35 E-04	-1	-1	-1	6.09E-10
f_7	3	3	3	1.61E-08	3	3	3	2.21E-15
f_8	-3.86278	-3.86278	-3.86278	2.27E-15	-3.86278	-3.86278	-3.86278	2.71E-15
f_9	4.65E-08	4.65E-08	4.65E-08	6.78E-17	4.65E-08	4.65E-08	4.65E-08	4.05E-17
f_{10}	3.92E-15	7.97E-12	8.95E-11	1.69E-11	6.95E-15	8.78E-11	4.2E-10	1.24E-10
f_{11}	3.67E-06	7.81E-05	1.43E-04	3.58E-05	3.52E-11	1.60E-07	2.65E-06	4.88E-07
f_{12}	2.37E-03	3.23E-02	1.54E-01	3.67E-02	4.82E-04	1.90E-02	1.19E-01	2.54E-02
f_{13}	-10.5364	-10.5364	-10.5364	1.05E-07	-10.5364	-10.5364	-10.5364	2.06E-15
f_{14}	-186.731	-186.731	-186.731	1.97E-14	-186.731	-186.731	-186.731	3.88E-14
f_{15}	-50	-50	-49.9999	1.23E-05	-50	-50	-49.9998	5.00E-05

表 6.4 给出测试函数 $f_{16}-f_{27}$ 在 $D=30$ 和 $D=60$ 下的实验结果. 结果显示 EC-ABC 算法较 ABC 算法不论在单模问题还是多模问题上都能得到更好结果.

图 6.3-6.6 给出了测试函数 $f_{16}-f_{27}$ 的收敛曲线图和箱型图. 图 6.3 和图 6.5 分别给出 $D=30$ 和 $D=60$ 下测试函数 $f_{16}-f_{27}$ 的收敛曲线图, 能够反映出 EC-ABC 算法和 ABC 算法的收敛速度. 图 6.4 和图 6.6 给出 $D=30$ 和 $D=60$ 下测试函数 $f_{16}-f_{27}$ 的箱型图, 反映 EC-ABC 算法和 ABC 算法的稳定性. 从图 6.3 和图 6.5 可以看出, EC-ABC 算法的收敛速度更快, 能得到更好的解. 从图 6.4 和图 6.6 可以看出, EC-ABC 算法的稳定性更好.

为了更进一步测试改进人工蜂群算法的性能, 将有效人工蜂群算法与其他群智能算法, 包括一些其他的改进人工蜂群算法进行比较研究. 比较算法包括 DE [5], PSO [5], CLPSO [146], CES [147], FES [148], ESLAT [149], CMA-ES [149], GABC [27], I-ABC [150], PS-ABC [150] 和 NABC [151]. 其中 DE 和 PSO 的参数设定和文献[5]相同, 剩余其他算法的参数设定均按原始论文中的设定给出.

表 6.5 给出了 DE, PSO, CES, FES, ESLAT, CMA-ES 和改进人工蜂群算法

的比较结果，其中 DE, PSO, CES, FES, ESLAT 和 CMA-ES 的实验结果来自于文献[5]. 表 6.6 给出改进人工蜂群算法与其他四种人工蜂群算法的比较结果，其他四种人工蜂群算法结果来自于文献[151]. 表 6.5 和表 6.6 中的最优结果均用黑体标注.

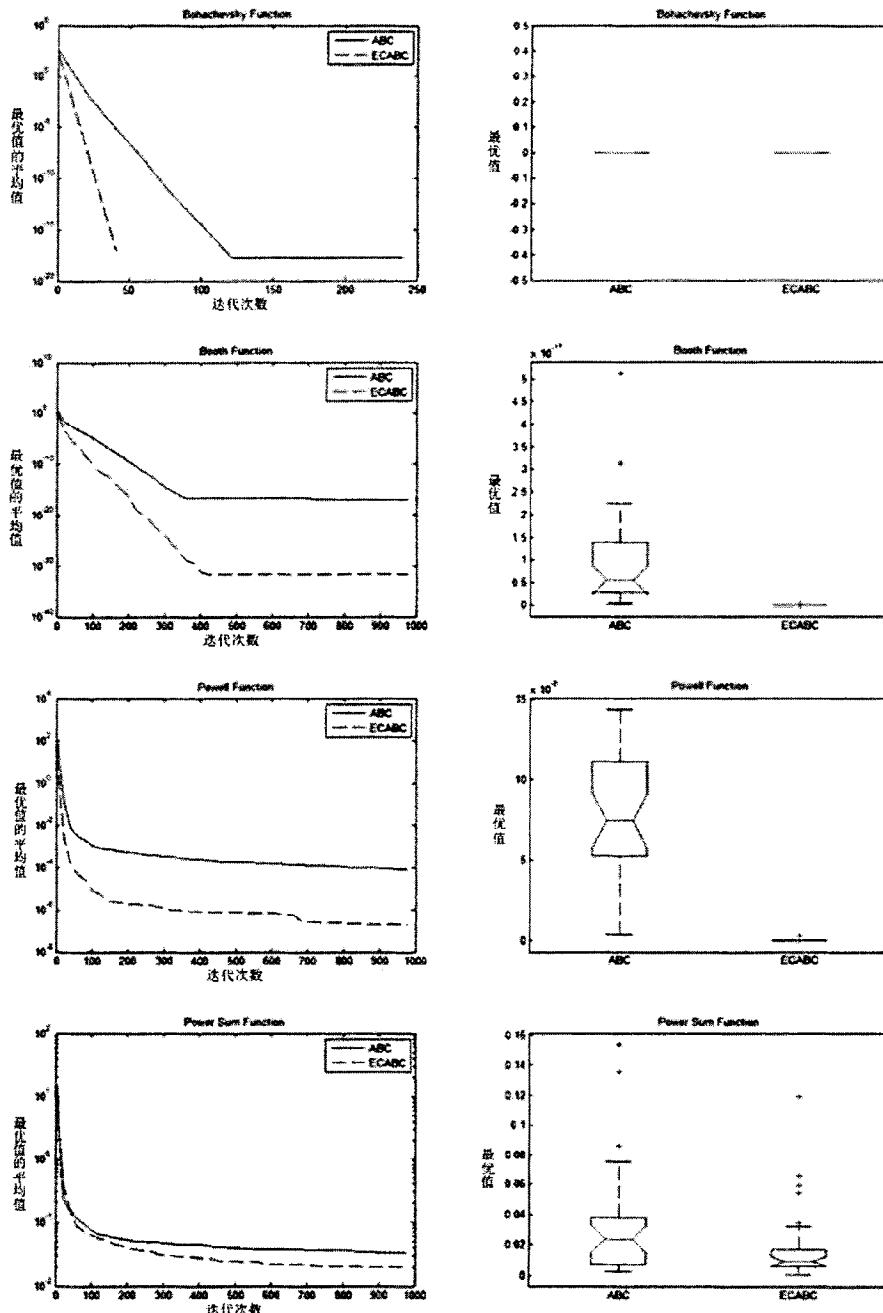
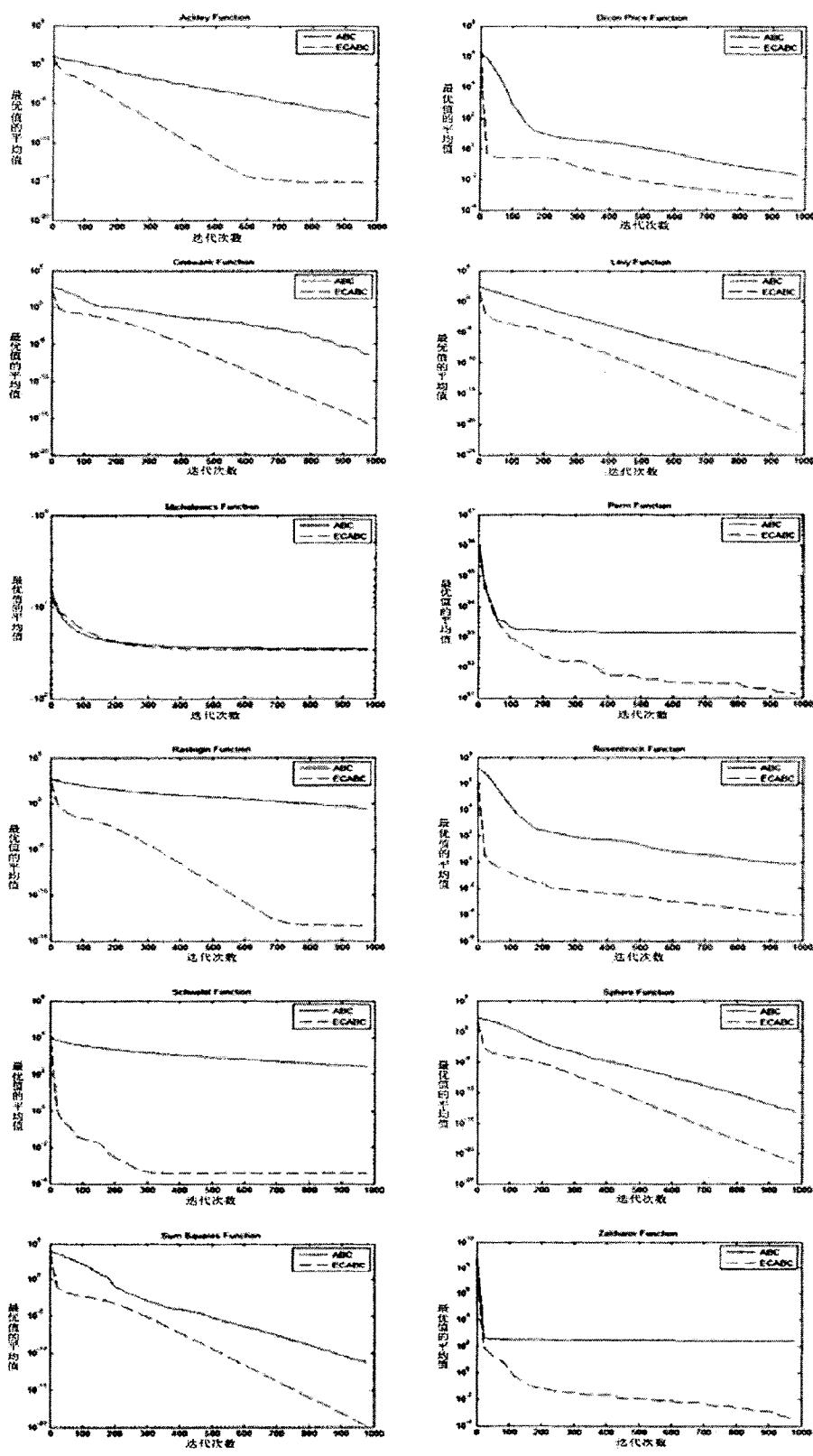


图 6.2 部分测试函数收敛曲线图及箱型图

表 6.4 f_{16} - f_{27} EC-ABC 算法及人工蜂群算法实验结果

D	ABC				EC-ABC			
	Best	Mean	Worst	Std	Best	Mean	Worst	Std
f_{16}	30	3.93E-09	1.12E-07	1.00E-06	1.94E-07	8.88E-16	8.88E-16	8.88E-16
	60	1.52E-04	6.62E-04	2.01E-03	4.53E-04	7.99E-15	3.65E-13	2.27E-12
f_{17}	30	4.47E-03	1.85E-02	1.06E-01	1.90E-02	6.00E-05	4.65E-04	1.13E-03
	60	2.21E-01	6.50E+00	1.33E+01	3.83E+00	3.20E-03	1.20E-02	2.51E-02
f_{18}	30	1.34E-11	1.40E-07	3.01E-06	5.48E-07	0.00E+00	2.59E-17	5.55E-16
	60	1.23E-05	2.66E-02	1.50E-01	3.65E-02	4.47E-10	5.81E-08	6.83E-07
f_{19}	30	1.45E-14	1.91E-13	6.06E-13	1.45E-13	1.11E-24	1.51E-22	1.07E-21
	60	5.07E-07	6.21E-06	2.99E-05	6.35E-06	7.42E-14	5.77E-11	7.21E-10
f_{20}	30	-2.94E+01	-2.90E+01	-2.88E+01	1.55E-01	-2.96E+01	-2.96E+01	-2.95E+01
	60	-5.63E+01	-5.47E+01	-5.36E+01	6.18E-01	-5.81E+01	-5.70E+01	-5.61E+01
f_{21}	30	2.51E+78	1.32E+83	2.73E+84	4.94E+83	7.39E+77	1.21E+81	1.74E+82
	60	6.79E+202	1.01E+208	9.90E+208	6.55E+04	2.56E+201	3.05E+205	3.46E+206
f_{22}	30	5.14E-09	2.15E-01	9.95E-01	4.04E-01	0.00E+00	4.36E-14	1.14E-13
	60	3.36E+00	1.23E+01	1.72E+01	3.53E+00	1.16E-10	3.74E-09	1.45E-08
f_{23}	30	2.71E-02	6.15E-01	2.44E+00	6.05E-01	1.15E-06	5.65E-05	2.37E-04
	60	1.18E+00	6.86E+01	1.58E+02	4.56E+01	1.77E-09	2.99E-05	3.64E-04
f_{24}	30	1.02E-03	2.66E+02	5.93E+02	1.41E+02	3.82E-04	3.82E-04	3.82E-04
	60	1.56E+03	2.04E+03	2.45E+03	2.52E+02	7.64E-04	7.64E-04	7.64E-04
f_{25}	30	2.76E-15	2.28E-14	6.03E-14	1.92E-14	2.01E-24	5.98E-23	3.88E-22
	60	2.81E-07	1.91E-06	5.44E-06	1.53E-06	2.38E-13	2.94E-11	2.44E-10
f_{26}	30	1.56E-13	1.46E-12	5.23E-12	1.15E-12	5.57E-24	4.40E-21	2.19E-20
	60	2.26E-06	8.14E-06	2.17E-05	4.90E-06	8.69E-13	1.16E-09	5.18E-09
f_{27}	30	1.75E+02	2.57E+02	3.44E+02	3.78E+01	6.42E-09	8.45E-05	4.65E-04
	60	6.68E+02	7.51E+02	8.37E+02	4.69E+01	3.22E-06	1.46E-03	1.21E-02

图 6.3 $D=30$ 时部分测试函数收敛曲线图

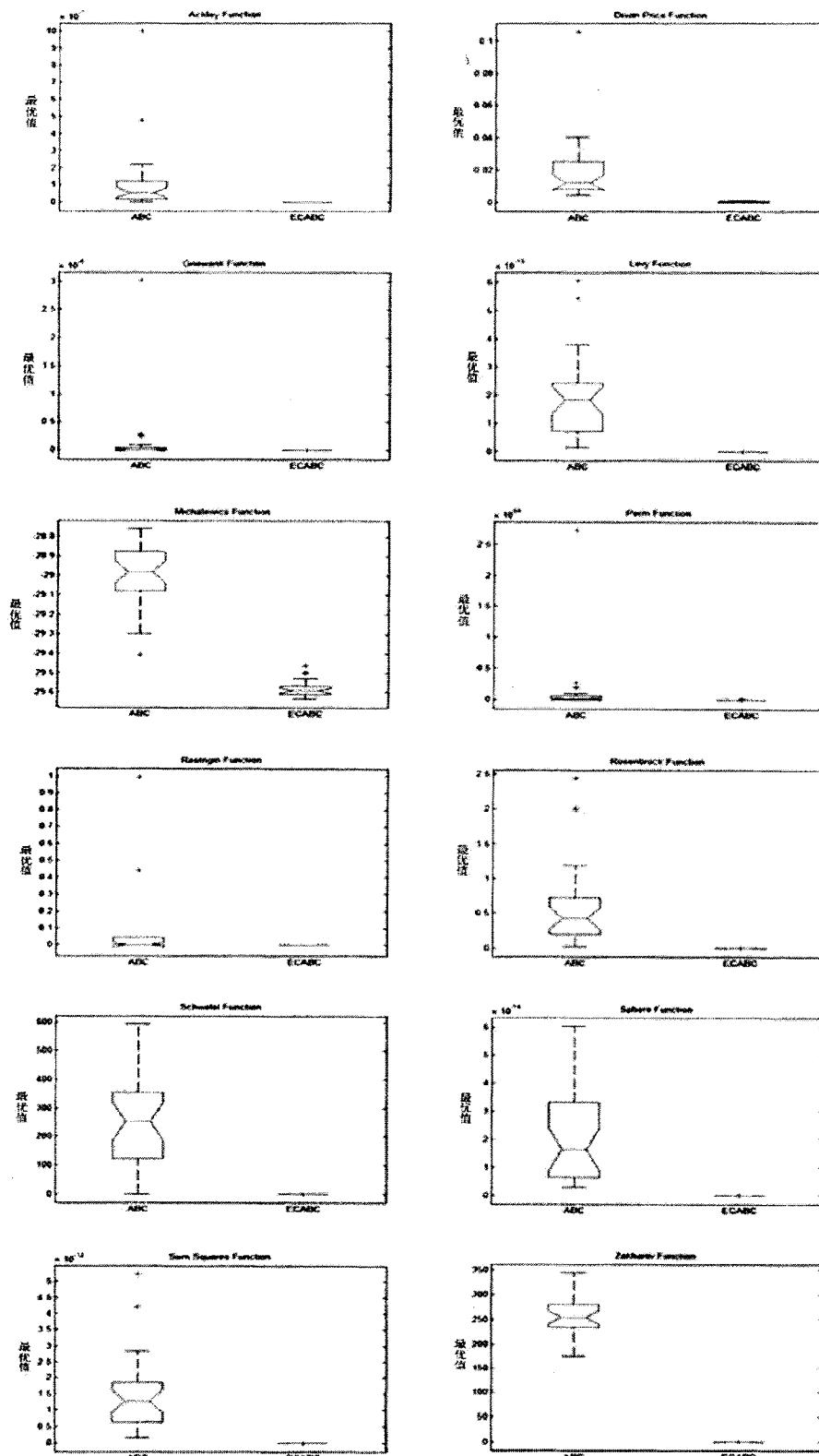
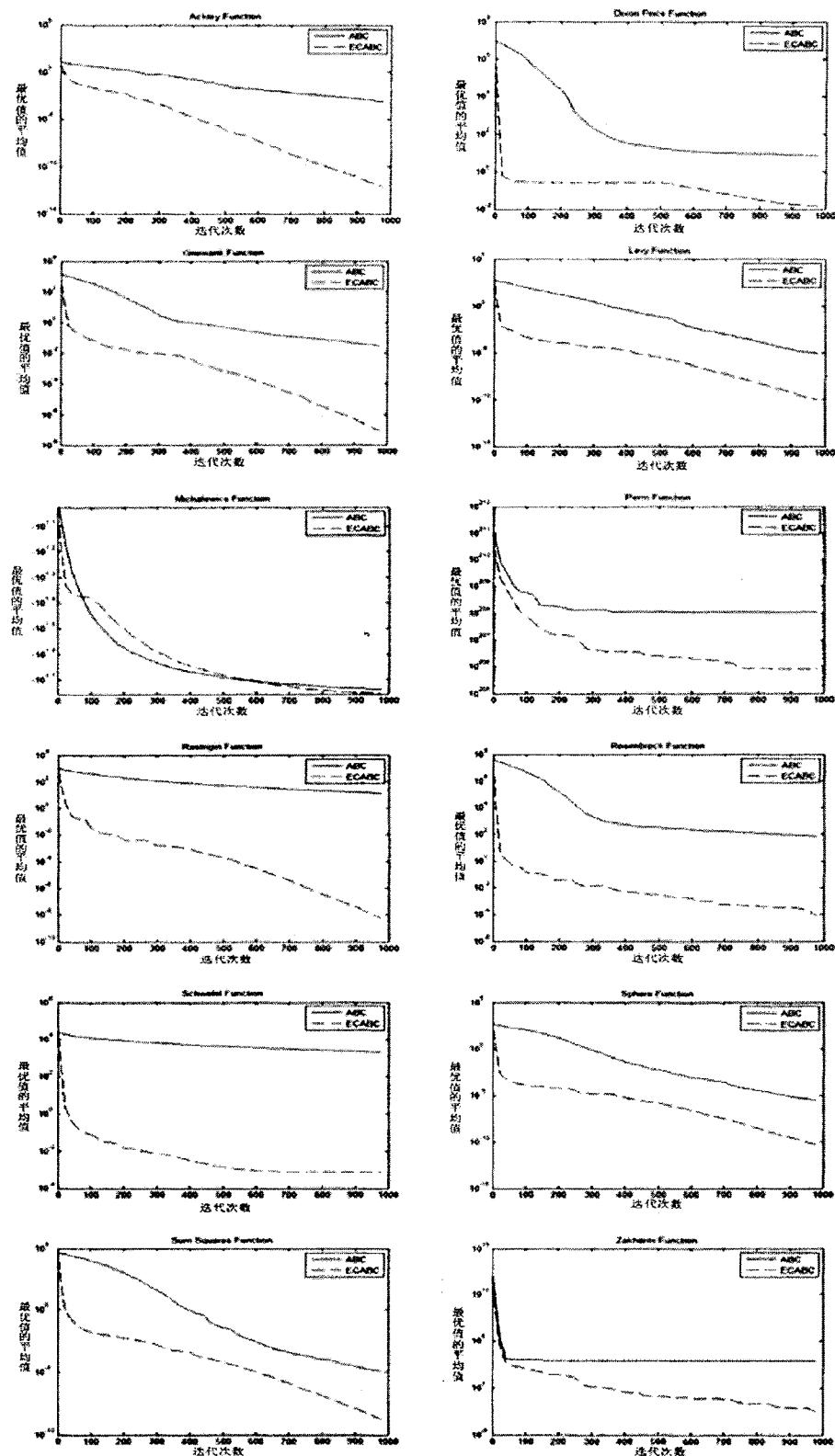


图 6.4 $D=30$ 时部分测试函数箱型图

图 6.5 $D=60$ 时部分测试函数收敛曲线图

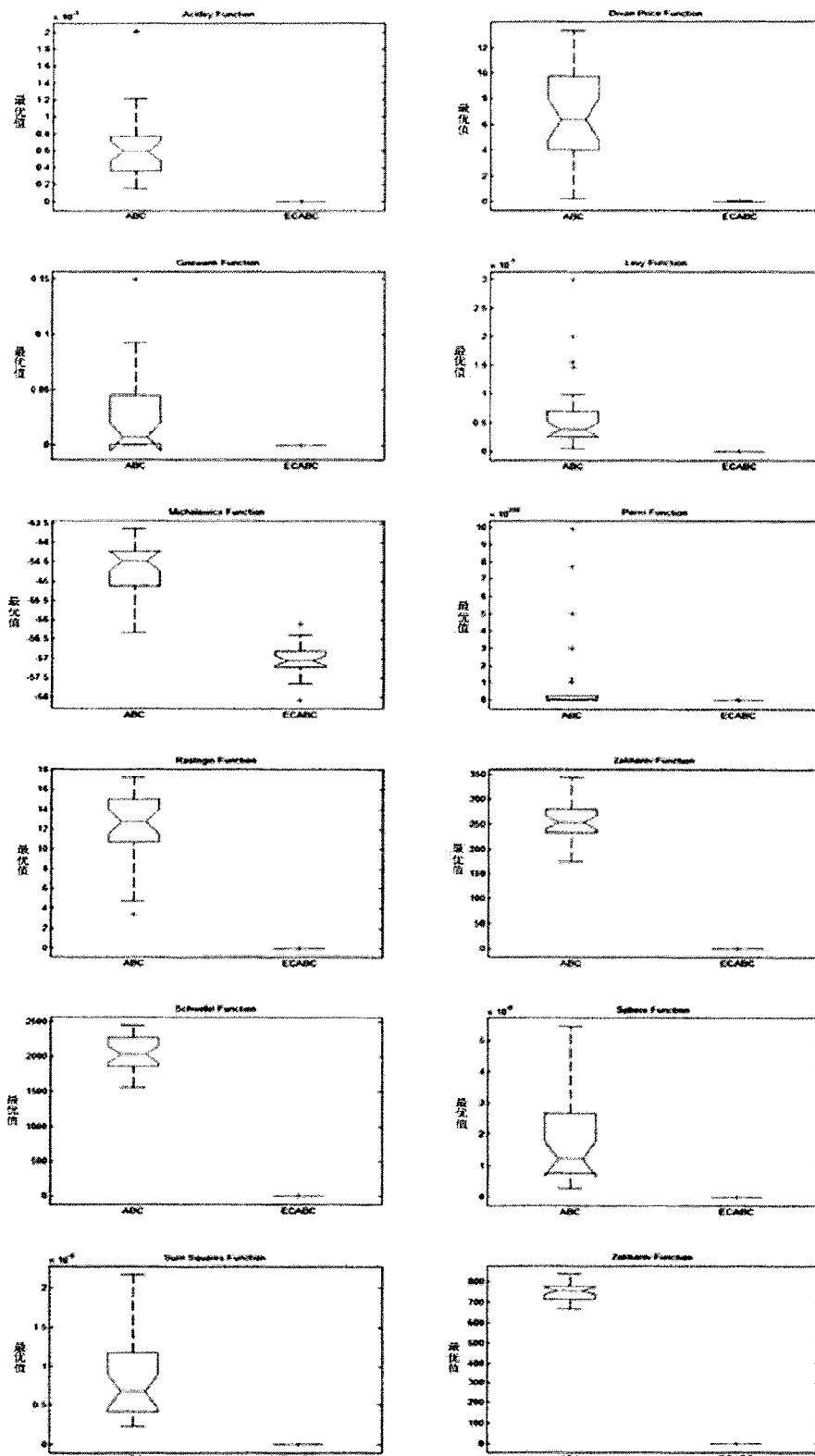


图 6.6 $D=60$ 时部分测试函数箱型图

表 6.5 EC-ABC 算法与其他算法比较结果

	Ackley	Griewank	Rastrigin	Rosenbrock	Schwefel	Sphere
	Mean	Mean	Mean	Mean	Mean	Mean
DE	3.99e-08	6.15e-04	1.47e+02	4.71e+03	7.27e+03	3.43e-14
PSO	3.23e-01	1.34e-02	3.85e+01	5.74e+03	4.16e+03	2.13e-16
CLPSO	2.01e-12	6.45e-13	2.57e-11	1.10e+01	1.19e+01	1.89e-19
CES	6.00e-13	6.00e-14	1.34e+01	2.77e+01	4.57e+03	1.70e-26
FES	1.20e-02	3.70e-02	1.60e-01	3.33e+01	1.31e+01	2.50e-04
ESLAT	1.80e-08	1.40e-03	4.65e+00	1.93e+00	1.03e+04	2.00e-17
CMA-ES	6.90e-12	7.40e-04	5.18e+01	4.00e-01	4.93e+03	9.70e-23
EC-ABC	8.88E-16	2.59E-17	4.36E-14	5.65E-05	3.82E-04	5.98E-23
Sig.	+	+	+	+	+	.

表 6.6 EC-ABC 算法与其他人工蜂群算法结果比较

Fun	GABC	I-ABC	PS-ABC	NABC	EC-ABC	Sig.
	Mean	Mean	Mean	Mean	Mean	
Ackley	7.78e-10	8.88e-16	8.88e-16	1.07e-13	8.88E-16	+
Griewank	6.96e-04	0.00e+00	0.00e+00	1.11e-16	2.59E-17	.
Rastrigin	3.31e-02	0.00e+00	0.00e+00	0.00e+00	4.36E-14	.
Rosenbrock	7.48e+00	2.64e+01	1.59e+00	1.45e-01	5.65E-05	+
Schwefel	1.62e+02	3.18e+02	5.30e+00	5.73e-01	3.82E-04	+
Sphere	6.26e-16	0.00e+00	0.00e+00	5.43e-16	5.98E-23	.

从表 6.5 中可以看出，对所有六个测试函数，除了 Sphere 函数上 EC-ABC 算法较 CES 结果差，较其他算法结果好外，EC-ABC 算法得到的结果优于其他算法，也就是说不论是单模测试函数还是多模测试函数 EC-ABC 算法的优化性能比其他七种算法都要好。

从表 6.6 中可以看出，对所有六个测试函数，EC-ABC 算法得到的结果优于 GABC 算法。与 NABC 比较发现，除了测试函数 Rastrigin，EC-ABC 算法的结果均优于 NABC 算法。另外，对于测试函数 Ackley，EC-ABC 算法的结果与 I-ABC 算和 PS-ABC 算法相同；对于测试函数 Rosenbrock 和 Schwefel，EC-ABC 算法得到的平均函数值优于 I-ABC 算和 PS-ABC 算法；而对于测试函数 Griewank、Rastrigin 和 Sphere，I-ABC 算和 PS-ABC 算法则能得到更好的结果。

综上所述，不论是单模问题还是多模问题，本章所提出的 EC-ABC 算法都能得到较好的优化结果。

6.3.2 对 NLLS 问题实验

本节将对 5 个非负线性最小二乘问题进行实验来验证有效人工蜂群算法在该问题上应用的可行性和有效性。5 个测试问题描述如下：

NLLS 1：考虑如下 NLLS 问题，其中

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 4 & -1 \\ 2 & -2 & 1 \end{bmatrix}, \quad b = \begin{pmatrix} 6 \\ 5 \\ 1 \end{pmatrix}$$

最优解为 $x^* = (1, 1, 3)^T$ 。

NLLS 2：令矩阵 A 的对角元素为 500，非对角元素从区间中随机选取，使得矩阵 A 为对称阵。令 $b = Ae$ ，其中 e 为 $n \times 1$ 的全 1 向量。最优解为 $x^* = (1, 1, \dots, 1)^T \in R^n$ 。

NLLS 3：令矩阵 A 为

$$a_{i,i} = 4n, \quad a_{i,i+1} = a_{i+1,i} = n, \quad a_{i,j} = 0, \quad i = 1, 2, \dots, n.$$

令 $b = Ae$ ，其中 e 为 $n \times 1$ 的全 1 向量。最优解为 $x^* = (1, 1, \dots, 1)^T \in R^n$ 。

NLLS 4：随机生成 NLLS 问题，生成方式如下：

$\text{rand}('state', 0)$, $m = 200$, $n = 100$, $A = \text{rand}(m, n)$, $b = A * \text{ones}(n, 1)$,
其中, $A \in R^{m \times n}$. 最优解为 $x^* = (1, 1, \dots, 1)^T \in R^n$.

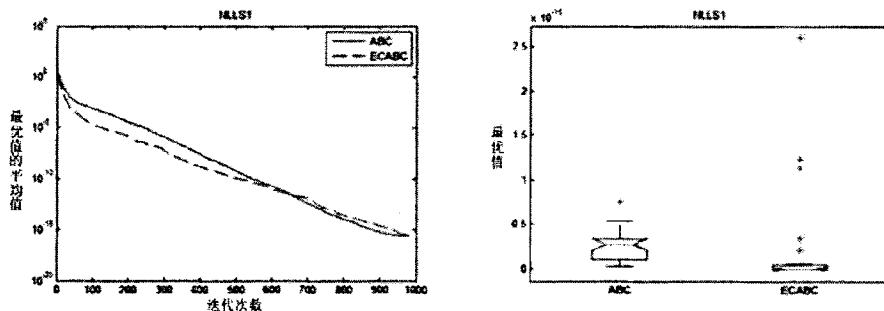
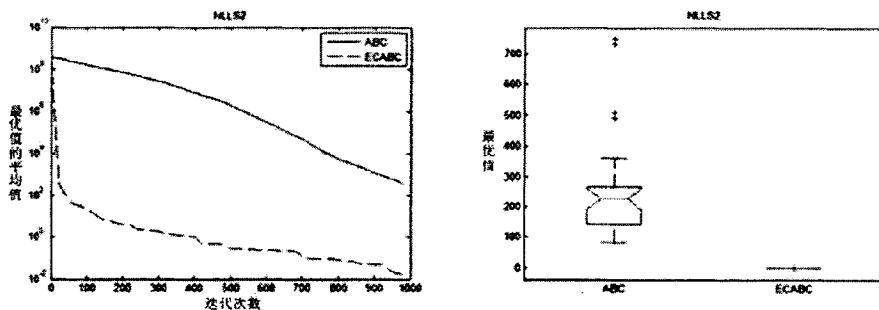
NLLS 5：随机生成 NLLS 问题，生成方式如下： $\text{rand}('state', 0)$, $n = 100$,
 $A1 = \text{rand}(n, n)$, $A = A1 * A1 + n * \text{eye}(n, n)$, $b = A * \text{ones}(n, 1)$, 其中, A 是正定对称矩阵。最优解为 $x^* = (1, 1, \dots, 1)^T$ 。

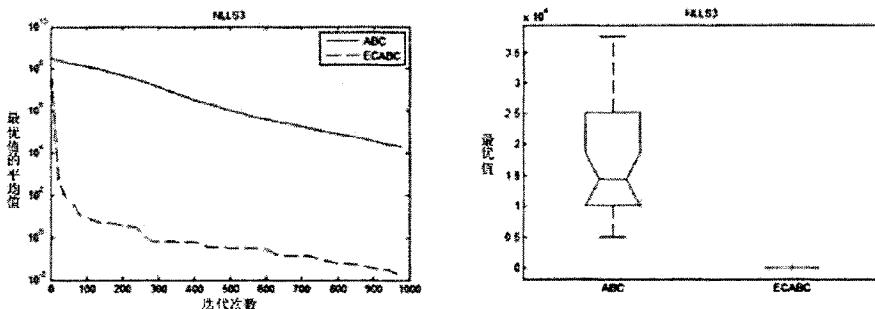
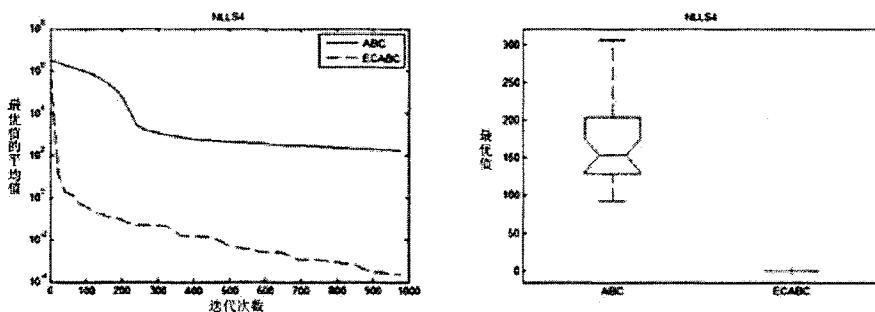
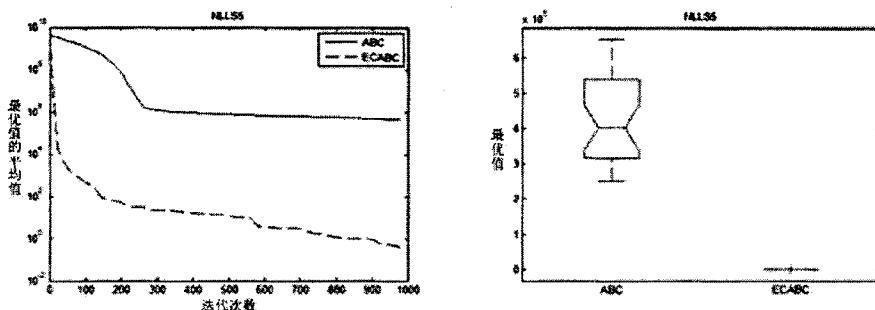
本节实验主要比较 EC-ABC 算法和原始 ABC 算法对 NLLS 问题的优化结果。为了实验的公平性，EC-ABC 算法与原始 ABC 算法设定相同的参数。实验中种群规模 SN 、 limit 和最大循环代数 (MSN) 分别设定为 60、 $(SN/2)*D$ 、1000。为了反映不同算法的准确性，所有实验均独立重复 30 次，记录实验的最好适应值、适应度均值、最差适应值和适应值标准差。表 6.7 给出实验比较结果，表中黑体表示较好结果。

表 6.7 NLLS 问题的实验结果

Functions	Algorithms	Best	Mean	Worst	Std
NLLS 1	ABC	2.36E-17	2.65E-16	7.49E-16	1.87E-16
	EC-ABC	2.68E-23	1.90E-16	2.60E-15	5.45E-16
NLLS 2	ABC	7.98E+01	2.54E+02	7.47E+02	1.69E+02
	EC-ABC	2.52E-09	6.02E-03	6.62E-02	1.27E-02
NLLS 3	ABC	4.93E+03	1.69E+04	3.77E+04	9.27E+03
	EC-ABC	1.23E-07	2.97E-03	2.13E-02	4.86E-03
NLLS 4	ABC	9.25E+01	1.66E+02	3.06E+02	4.94E+01
	EC-ABC	1.49E-07	5.80E-05	4.91E-04	9.71E-05
NLLS 5	ABC	2.52E+05	4.28E+05	6.52E+05	1.22E+05
	EC-ABC	4.61E-06	9.96E-02	6.51E-01	1.43E-01

通过比较表 6.7 中的适应度均值和标准差可以发现, 对于这五个 NLLS 测试问题, EC-ABC 算法得到的结果远远好于 ABC 算法。因此, 对于 NLLS 问题, EC-ABC 算法具有较强的优化性能。

图 6.7 $n=100$ 时 NLLS 1 收敛曲线及箱型图图 6.8 $n=100$ 时 NLLS 2 收敛曲线及箱型图

图 6.9 $n=100$ 时 NLLS 3 收敛曲线及箱型图图 6.10 $n=100$ 时 NLLS 4 收敛曲线及箱型图图 6.11 $n=100$ 时 NLLS 5 收敛曲线及箱型图

为了更明确的说明有 EC-ABC 算法的性能, 图 6.7-6.11 给出了收敛曲线图和箱型图. 图中可以明确看出, 有效人工蜂群算法不仅对简单非负线性最小二乘问题有效, 对于较复杂的非负线性最小二乘问题也具有较好的优化效果.

6.4 小结

尽管人工蜂群算法有较强的优化性能, 但与其他随机算法类似也具有一些不足之处, 比如对单模问题收敛速度较慢, 对多模问题算法易陷入局部最优. 为了

克服人工蜂群算法存在的不足，本章给出一种有效混沌人工蜂群算法（EC-ABC）。该算法利用反学习初始化生成初始种群，引入混沌局部搜索策略，设计新的搜索机制，提高算法优化性能。

为了测试该算法的性能，我们对 27 个测试函数和 5 个非负线性最小二乘测试问题进行了实验。实验结果表明 EC-ABC 算法能有效提高算法的收敛速度，改进解的准确性，对求解非负线性最小二乘问题是可行有效的。

第七章 约束优化问题的算法投资组合

对于约束优化问题，本章基于进化策略和人工蜂群算法，提出一种算法投资组合。尽管以往的研究对于约束优化问题中约束处理方法有了深入的研究，但对于不同的约束优化问题，约束处理方法的优劣也不尽相同。不同于选择将所有计算时间全部投入到一种约束处理方法和算法上，本章提出的算法投资组合将计算时间分别分配给不同的算法和约束处理方法上来减少计算的风险。基于这种思想，算法投资组合中将选取自适应约束处理进化策略和 Deb 选择策略下的群智能算法。另外，算法中设计了一个迁移阶段，用来反映两个算法间的信息交换。通过对 13 个 benchmark 测试函数的实验测试来验证算法的优化性能。通过与其他算法的比较研究，表明该算法较其他智能优化算法具有更好的优化性能。同时，本章对算法参数进行了实验分析，给出了每个参数的合理取值。

7.1 引言

过去二十几年中，群智能算法广泛应用于全局优化问题中。目前，应用群智能算法求解约束优化问题得到学者们的广泛关注。一般情况下，约束优化问题可如下定义：

$$\begin{aligned} \min \quad & f(x), \quad x = (x_1, x_2, \dots, x_n) \in R^n \\ \text{subject to:} \quad & l_i \leq x_i \leq u_i, \quad i = 1, \dots, n, \\ & g_j(x) \leq 0, \quad \text{for } j = 1, \dots, q, \\ & h_j(x) = 0, \quad \text{for } j = q+1, \dots, m. \end{aligned} \tag{7-1}$$

目标函数 f 定义在 R^n 中的一个 n 维多边形上 ($S \subseteq R^n$)。变量的定义域给出了其变化的上界和下界。可行域 $F \subseteq S$ 由 m 个约束确定，变量 x 定义在可行域内。若不等式约束对任意 $x \in F$ 满足 $g_j(x) = 0$ ($j = 1, \dots, q$)，则称其在点 x 是积极约束。所有的等式约束在可行域 F 内都是积极约束。

大多数算法通过结合约束处理方法来引导算法在搜索空间中的可行域内进行搜索。这些约束处理方法大体可以分为四类^[157]：

- (1) 基于保留可行解的方法：通过算子将不可行解转化为可行解来确保问题可行；^[158-159]
- (2) 基于罚函数法：通过对目标函数引入罚项将约束问题转化为无约束问题进行求解；^[160-163]

(3) 明确区分可行解和不可行解的方法; [164-168]

(4) 其他混合方法。[169-172]

所有这些约束处理方法都有各自不同的优点。在这些方法当中，有两种约束处理方法具有较好的效果，一种是 Deb 法则^[173]，另一种是自适应模型^[174]。由于方法的简洁性，Deb 法则通常用来处理约束优化问题。这一方法强调可行解优于不可行解，这样使得算法经常会缺乏种群多样性而陷入局部最优^[168]。而自适应模型在处理约束优化问题时，则会保留一部分不可行解，从不可行解中得到一些利于优化的信息，从而保持种群的多样性。但这种方法在一定程度上会降低算法的收敛速度。

本章中将结合 Deb 法则和自适应模型处理约束的优点，给出一种算法投资组合。在算法投资组合中，不同于将所有的计算时间放在单一的算法中，这里将计算时间“投资”在两个具有互补特性的算法上。并通过迁移阶段实现两种算法间信息的交换，由此来保留种群中较好个体。本章中提出的算法投资组合将会通过对文献[174]中的 13 个约束优化测试问题进行性能测试，测试结果将会与其他算法进行比较^{[157],[171-178]}。

本章主要安排如下：7.2 节给出算法投资组合；7.3 节是实验及结论分析；7.4 节对本章进行小结。

7.2 算法投资组合

近年来，对于函数优化问题有学者提出一种新的算法体系——基于种群的算法投资组合，这种方法将计算时间分配给多个算法来完成优化过程，这样可以取得各个算法的优势，提高算法的优化性能^[179]。受这一思想的启发，本章基于进化策略和人工蜂群算法两种算法提出一种求解约束优化问题的算法投资组合，算法汲取了进化策略和人工蜂群算法的优点。下面将首先描述求解约束优化问题的进化策略和人工蜂群算法，然后给出算法投资组合的具体流程。

7.2.1 基于自适应模型的进化策略

近年来，有学者提出带有自适应模型的进化策略用来求解约束优化问题，该方法在优化过程中体现出较好的性能^[174]。该算法中，自适应模型的设计是用来处理约束的，处理方法涉及三个阶段：

- a. 种群中都是不可行解；
- b. 种群中既包含可行解又包含不可行解；
- c. 种群完全由可行解构成。

记 f_p 为当前种群的可行解比率，这样阶段 a 表示 $f_p = 0$ 。在这一阶段，设计

一种分层非支配个体选择模式。该模式仅选取种群中约束违背较低的非支配个体。首先，根据约束违背大小升序排列非支配个体；其次，选取前 50% 的非支配个体添加到子代种群，同时将这些个体从父代种群中删除；最后，在剩余的父代种群中按排序情况再选取前 50% 的非支配个体，添加到子代种群中，同时从父代种群中删除。重复上述过程直到子代种群数量达到种群数量上限。

阶段 b 表示 $0 < f_p < 1$ 。在这一阶段中，具有较好目标函数值和较低不可行率的不可行个体将保留到子代种群中。为了达到这一目标，给出一个适应值函数，该函数将标准化目标函数和标准化约束违背函数相加，即：

$$f_{final}(x_i) = f_{nor}(x_i) + G_{nor}(x_i), \quad i \in Z, \quad (7-2)$$

其中， $Z = \{1, 2, \dots, \lambda\}$ 表示个体指标集。 Z_1 表示可行解个体指标集， Z_2 表示不可行解个体指标集。转化后目标函数可表示为：

$$f'(x_i) = \begin{cases} f(x_i), & i \in Z_1, \\ \max\{\varphi * f_{min} + (1 - \varphi) * f_{max}, f(x_i)\}, & i \in Z_2, \end{cases} \quad (7-3)$$

其中， φ 表示种群的可行解比率， $f_{min} = \min_{i \in Z_1} f(x_i)$ 且 $f_{max} = \max_{i \in Z_1} f(x_i)$ 。其次，目标函数可进行如下标准化：

$$f_{nor}(x_i) = \frac{f'(x_i) - \min_{j \in Z} f'(x_j)}{\max_{j \in Z} f'(x_j) - \min_{j \in Z} f'(x_j)}, \quad i \in Z, \quad (7-4)$$

且约束违背函数标准化方式如下：

$$G_{nor}(x_i) = \begin{cases} 0, & i \in Z_1, \\ \frac{G(x_i) - \min_{j \in Z_2} G(x_j)}{\max_{j \in Z_2} G(x_j) - \min_{j \in Z_2} G(x_j)} & i \in Z_2. \end{cases} \quad (7-5)$$

阶段 c 表示 $f_p = 1$ 。很明显，这一阶段种群全部由可行解构成。因此，个体通过目标函数值进行比较。

综上所述，自适应模型可由如下算法给出。

Algorithm 7.1: 自适应模型

if $f_p = 0$

```

while |A| < μ do
    找出种群 A 中的非支配个体;
    A'' = {具有较小约束违背的所有非支配个体中的前 50% };
    A = A \ A'', A' = A' ∪ A'';
    end while
    A' = {A' 中前 μ 个个体};
    else if 0 < f_p < 1
        A' = {基于式(7-2)选取 A 中最好的 μ 个个体};
    else
        A' = {基于目标函数值选取 A 中最好的 μ 个个体};
    end if

```

因此，自适应模型的进化策略流程如下。

Algorithm 7.2: 自适应模型的进化策略

Step 1: 种群初始化，设定算法参数，对初始种群进行估计。

Step 2: 基于 Algorithm 7.1 选取种群中的最好的 μ 个个体。

Step 3: 利用进化策略生成子代种群。

Step 4: 检验是否达到停止准则。若满足停止准则，则停止搜索，输出最终结果；否则转向 Step 2。

7.2.2 人工蜂群算法

人工蜂群算法是近期提出的一种函数优化算法，该算法模拟蜜蜂群体采蜜行为实现寻优过程^[1]。由于其优良的优化性能，该算法被广泛应用于实际问题当中。

算法中，种群由雇佣蜂、跟随蜂和侦查蜂组成。雇佣蜂探索食物源，并将食物源信息与蜂巢中的跟随蜂共享。跟随蜂根据共享信息对食物源进一步开采。食物源被舍弃的雇佣蜂转变为侦查蜂探索新食物源。初始种群在定义域内随机生成，即：

$$x_{ij} = x_j^{\min} + rand(0,1)(x_j^{\max} - x_j^{\min}), \quad (7-6)$$

其中， $i \in \{1, 2, \dots, SN\}$ ， $j \in \{1, 2, \dots, D\}$ 。雇佣蜂和跟随蜂的搜索方程为：

$$v_{ij} = x_{ij} + \phi_j(x_{ij} - x_{kj}) \quad (7-7)$$

其中， k, j 是随机选取的下标， $k \in \{1, 2, \dots, SN\}$ ， $k \neq i$ ， $j \in \{1, 2, \dots, N\}$ ， $\phi_j \in [-1, 1]$

是一个随机数.

通过给原始人工蜂群算法加入约束处理机制, 该算法同样可以用来求解约束优化问题. 算法通过在选择过程中使用 Deb 法则代替原来的贪婪选择实现对约束优化问题的求解, 获得问题的可行解. Deb 法则是一个锦标赛选择算子, 其中两个解的比较标准如下:

- a. 任何可行解都优于不可行解;
- b. 两个可行解中, 目标函数值较好的解较好;
- c. 两个不可行解中, 约束违背越小的解越好.

记食物源数量为 SN , 第 i 个食物源记为一个 n 维向量 x_i ($i=1, \dots, SN$). 在人工蜂群算法中, 对于最小化问题第 i 个适应值函数 fit_i 定义为:

$$fit_i = \begin{cases} \frac{1}{1+f_i}, & f_i \geq 0, \\ 1+abs(f_i), & f_i < 0, \end{cases} \quad (7-8)$$

其中, f_i 是第 i 个解的目标函数值. 跟随蜂选择食物源的概率为:

$$p_r = \begin{cases} 0.5 + \left(\frac{fit_i}{\sum_{i=1}^{SN} fit_i} \right) \times 0.5 & \text{if solution is feasible,} \\ \left(1 - \frac{violation_i}{\sum_{i=1}^{SN} violation_i} \right) & \text{if solution is infeasible,} \end{cases} \quad (7-9)$$

其中, $violation_i$ 是解的罚值 x_i .

基于以上分析, 人工蜂群算法的主要步骤如下所示.

Algorithm 7.3: 人工蜂群算法

Step 1: 利用式(7-6)生成初始种群, 设定算法参数.

Step 2: 对雇佣蜂根据式(7-7)搜索新食物源, 并估计食物源的目标函数值、适应值和罚值. 利用 Deb 选择策略在新食物源和旧食物源中选择较好的一个.

Step 3: 根据式(7-9)计算跟随蜂对食物源的选择概率, 利用轮盘赌选择, 为跟随蜂选择一个食物源. 对跟随蜂根据式(7-7)搜索食物源, 并估计食物源的目标函数值、适应值和罚值. 利用 Deb 选择策略在新食物源和旧食物源中选择较好的一个.

Step 4: 若有被放弃食物源, 则用侦查蜂搜索的新食物源对其替代.

Step 5: 检查是否达到停止准则. 若达到, 则停止搜索, 输出最终食物源位置; 否则, 转 Step 2.

7.2.3 算法投资组合

文献[179]中提出的基于种群的算法投资组合将不同的群智能算法组合起来用于求解无约束优化问题, 实验测试说明该方法是可行有效的. 基于这种思想, 本章中提出一种算法投资组合用于求解约束优化问题, 该方法将自适应模型进化策略与人工蜂群算法结合, 将计算时间分配在两种算法上, 并通过交互行为实现信息传递. 算法的计算时间用函数评价次数(EFs)来反映. 假设优化问题 f , 其计算时间为 EFs. 算法中, 计算时间被分配在自适应模型进化策略和人工蜂群算法上, 每种算法独立运行. 两个子种群间的个体定时进行迁移来实现信息共享. 取 migration_interval 和 migration_size 两个参数确定迁移频率和迁移数量. 当算法进行迁移时, 每个算法将丢弃最差的 migration_size 个个体, 并用另一算法的最好的 migration_size 个个体替代. 算法投资组合流程的一般框架如下.

Algorithm 7.4: 基于 EA 和 ABC 的算法投资组合

Step 1: 初始化两个算法的子种群, 设定算法参数、migration_interval 和 migration_size.

Step 2: 评价所有个体, 并用第 i 个算法对第 i 个子种群进行更新($i=1, 2$).

Step 3: 若达到迁移间隔 migration_interval, 则实施算法迁移过程.

Step 4: 检查是否达到停止准则. 若达到, 则停止搜索, 输出最终食物源位置; 否则, 转 Step 2.

7.3 数值实验及分析

为了说明算法投资组合的有效性, 本节选取文献[174]中的 13 个 benchmark 测试函数对算法的优化性能进行验证. 这些测试问题中包含目标函数是线性的、非线性的和二次的. 表 7.1 给出这些测试问题的特征, 表中 LE 表示线性等式约束数量, NE 表示非线性等式约束数量, LI 表示线性不等式数量, NI 表示非线性不等式数量, D 表示变量维数, $\rho = |F|/|S|$ 表示可行域与整个搜索空间的估计比率, 其中 $|F|$ 表示可行解数量, $|S|$ 表示随机生成的所有解的数量^[180].

7.3.1 参数设置

在算法投资组合中, 自适应模型进化策略的参数设置如下: 种群数量(μ)为 40, 最大迭代次数为 3000; 人工蜂群算法参数设定为: 种群数量(SN)为 40, 最大

循环次数(MCN)为 3000, limit 为 $0.5 * SN * D$. 从而, 函数评价次数 FEs 为 240000. 参数 migration_interval 考虑四种取值, migration_size 考虑五种取值, 因此需要对 20 对参数进行测试. migration_interval 分别取最大迭代次数除以 20, 30, 40, 50; migration_size 分别取 1, 5, 10, 15, 20.

表 7.1 Benchmark 测试函数

Fun	D	Type of f	ρ	LI	NI	LE	NE
g01	13	Quadratic	0.0003%	9	0	0	0
g02	20	Nonlinear	99.9973%	1	1	0	0
g03	10	Nonlinear	0.0026%	0	0	0	1
g04	5	Quadratic	27.0079%	0	6	0	0
g05	4	Nonlinear	0.0000%	2	0	0	3
g06	2	Nonlinear	0.0057%	0	2	0	0
g07	10	Quadratic	0.0000%	3	5	0	0
g08	2	Nonlinear	0.8581%	0	2	0	0
g09	7	Nonlinear	0.5199%	0	4	0	0
g10	8	Linear	0.0020%	3	3	0	0
g11	2	Quadratic	0.0973%	0	0	0	1
g12	3	Quadratic	4.7697%	0	9 ³	0	0
g13	5	Nonlinear	0.0000%	0	0	1	2

7.3.2 结果及分析

参数 migration_interval 和 migration_size 可以反映算法的信息交换情况, 是算法投资组合中较为重要的两个参数. 因此, 这里测试 20 对设定的参数值, 实验结果见表 7.2-7.6, 包括均值, 最好值, 最坏值, 标准差和可行解比率. 在 20 组测试环境下, 表 7.2 和表 7.3 中, 13 个测试函数的最好值和均值都没有差异; 表 7.4 中, 测试函数 g01, g03, g04, g08, g11 和 g12 的最坏值也没有差异, 而其他测试函数最坏值上差异较明显. 根据这些差异和表 7.6 中可行解的比率可以看出, 当 migration_interval=MaxGen/40 且 migration_size=15 时, 可行解比率较高且优化结果较好. 因此, 表 7.7 给出在这组参数选择下的实验结果, 同时此后的比较实验均是基于这一参数设置进行的.

表 7.2 migration_interval 和 migration_size 的 20 组取值下得到的均值

Fun	Size	1	5	10	15	20
	Interval	Mean	Mean	Mean	Mean	Mean
g01	MaxGen/20	-15	-15	-15	-15	-15
	MaxGen/30	-15	-15	-15	-15	-15
	MaxGen/40	-15	-15	-15	-15	-15
	MaxGen/50	-15	-15	-15	-15	-15
g02	MaxGen/20	-0.794953	-0.786254	-0.782498	-0.790190	-0.790704
	MaxGen/30	-0.786231	-0.785623	-0.781732	-0.782408	-0.790898
	MaxGen/40	-0.786517	-0.785087	-0.785417	-0.785874	-0.787728
	MaxGen/50	-0.785728	-0.782530	-0.788152	-0.784236	-0.780989
g03	MaxGen/20	-1	-1	-1	-1	-1
	MaxGen/30	-1	-1	-1	-1	-1
	MaxGen/40	-1	-1	-1	-1	-1
	MaxGen/50	-1	-1	-1	-1	-1
g04	MaxGen/20	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	MaxGen/30	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	MaxGen/40	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	MaxGen/50	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
g05	MaxGen/20	5128.540	5127.702	5128.117	5128.792	5129.511
	MaxGen/30	5127.764	5128.322	5127.855	5128.008	5129.171
	MaxGen/40	5131.089	5128.412	5129.717	5127.973	5128.647
	MaxGen/50	5128.319	5127.959	5127.911	5127.510	5127.584
g06	MaxGen/20	-6896.394	-6921.283	-6834.415	-6839.697	-6919.100
	MaxGen/30	-6917.618	-6931.529	-6915.231	-6912.976	-6896.716
	MaxGen/40	-6872.422	-6920.707	-6906.445	-6961.814	-6910.795
	MaxGen/50	-6909.967	-6892.552	-6961.814	-6961.814	-6908.634
g07	MaxGen/20	24.377	24.355	24.357	24.392	24.397
	MaxGen/30	24.383	24.362	24.384	24.396	24.368
	MaxGen/40	24.390	24.371	24.399	24.360	24.362
	MaxGen/50	24.371	24.352	24.386	24.380	24.372
g08	MaxGen/20	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	MaxGen/30	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	MaxGen/40	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	MaxGen/50	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
g09	MaxGen/20	680.651	680.649	680.650	680.646	680.643
	MaxGen/30	680.649	680.657	680.653	680.665	680.645
	MaxGen/40	680.646	680.647	680.650	680.647	680.656
	MaxGen/50	680.668	680.645	680.649	680.647	680.649
g10	MaxGen/20	7771.139	7841.535	7499.980	7723.480	8816.98
	MaxGen/30	7495.561	7465.675	7660.040	7572.411	8559.809
	MaxGen/40	7662.441	8085.996	8150.769	7863.292	8404.890
	MaxGen/50	7398.856	8353.002	7502.655	8429.347	8304.723
g11	MaxGen/20	0.75	0.75	0.75	0.75	0.75
	MaxGen/30	0.75	0.75	0.75	0.75	0.75
	MaxGen/40	0.75	0.75	0.75	0.75	0.75
	MaxGen/50	0.75	0.75	0.75	0.75	0.75
g12	MaxGen/20	-1	-1	-1	-1	-1
	MaxGen/30	-1	-1	-1	-1	-1
	MaxGen/40	-1	-1	-1	-1	-1
	MaxGen/50	-1	-1	-1	-1	-1
g13	MaxGen/20	0.053962	0.053973	0.079672	0.079673	0.053978
	MaxGen/30	0.066842	0.053963	0.079694	0.053974	0.053967
	MaxGen/40	0.053973	0.053965	0.053972	0.053970	0.066823
	MaxGen/50	0.053994	0.053966	0.053980	0.053978	0.056340

表 7.3 migration_interval 和 migration_size 的 20 组取值下得到的最好值

Fun	Size	1	5	10	15	20
	Interval	Best	Best	Best	Best	Best
g01	MaxGen/20	-15	-15	-15	-15	-15
	MaxGen/30	-15	-15	-15	-15	-15
	MaxGen/40	-15	-15	-15	-15	-15
	MaxGen/50	-15	-15	-15	-15	-15
g02	MaxGen/20	-0.803566	-0.803595	-0.803602	-0.803593	-0.803590
	MaxGen/30	-0.803591	-0.803588	-0.803565	-0.803571	-0.803589
	MaxGen/40	-0.803577	-0.803571	-0.803590	-0.803589	-0.803590
	MaxGen/50	-0.803583	-0.803586	-0.803584	-0.803583	-0.803593
g03	MaxGen/20	-1	-1	-1	-1	-1
	MaxGen/30	-1	-1	-1	-1	-1
	MaxGen/40	-1	-1	-1	-1	-1
	MaxGen/50	-1	-1	-1	-1	-1
g04	MaxGen/20	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	MaxGen/30	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	MaxGen/40	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	MaxGen/50	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
g05	MaxGen/20	5126.498	5126.499	5126.499	5126.498	5126.498
	MaxGen/30	5126.499	5126.498	5126.498	5126.500	5126.499
	MaxGen/40	5126.504	5126.498	5126.498	5126.498	5126.499
	MaxGen/50	5126.498	5126.498	5126.498	5126.498	5126.498
g06	MaxGen/20	-6961.814	-6961.814	-6961.814	-6961.814	-6961.814
	MaxGen/30	-6961.814	-6961.814	-6961.814	-6961.814	-6961.814
	MaxGen/40	-6961.814	-6961.814	-6961.814	-6961.814	-6961.814
	MaxGen/50	-6961.814	-6961.814	-6961.814	-6961.814	-6961.814
g07	MaxGen/20	24.309	24.309	24.313	24.311	24.310
	MaxGen/30	24.310	24.311	24.308	24.308	24.313
	MaxGen/40	24.310	24.309	24.310	24.309	24.307
	MaxGen/50	24.310	24.310	24.307	24.313	24.308
g08	MaxGen/20	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	MaxGen/30	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	MaxGen/40	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	MaxGen/50	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
g09	MaxGen/20	680.630	680.630	680.631	680.631	680.631
	MaxGen/30	680.631	680.631	680.631	680.631	680.631
	MaxGen/40	680.630	680.630	680.631	680.631	680.630
	MaxGen/50	680.631	680.631	680.631	680.631	680.631
g10	MaxGen/20	7098.173	7066.553	7175.518	7103.405	7104.271
	MaxGen/30	7061.186	7098.489	7073.880	7099.146	7092.593
	MaxGen/40	7066.323	7091.180	7075.431	7083.960	7069.939
	MaxGen/50	7116.424	7098.683	7106.457	7162.459	7097.007
g11	MaxGen/20	0.75	0.75	0.75	0.75	0.75
	MaxGen/30	0.75	0.75	0.75	0.75	0.75
	MaxGen/40	0.75	0.75	0.75	0.75	0.75
	MaxGen/50	0.75	0.75	0.75	0.75	0.75
g12	MaxGen/20	-1	-1	-1	-1	-1
	MaxGen/30	-1	-1	-1	-1	-1
	MaxGen/40	-1	-1	-1	-1	-1
	MaxGen/50	-1	-1	-1	-1	-1
g13	MaxGen/20	0.053950	0.053950	0.053950	0.053950	0.053950
	MaxGen/30	0.053950	0.053950	0.053950	0.053950	0.053950
	MaxGen/40	0.053950	0.053950	0.053950	0.053950	0.053950
	MaxGen/50	0.053950	0.053950	0.053950	0.053950	0.053950

表 7.4 migration_interval 和 migration_size 的 20 组取值下得到的最坏值

Fun	Size	1	5	10	15	20
	Interval	Worst	Worst	Worst	Worst	Worst
g01	MaxGen/20	-15	-15	-15	-15	-15
	MaxGen/30	-15	-15	-15	-15	-15
	MaxGen/40	-15	-15	-15	-15	-15
	MaxGen/50	-15	-15	-15	-15	-15
g02	MaxGen/20	-0.778593	-0.696590	-0.738088	-0.747106	-0.752927
	MaxGen/30	-0.736499	-0.725900	-0.713170	-0.733640	-0.760143
	MaxGen/40	-0.714859	-0.723579	-0.749189	-0.740359	-0.737514
	MaxGen/50	-0.745561	-0.740909	-0.747582	-0.746313	-0.732557
g03	MaxGen/20	-1	-1	-1	-1	-1
	MaxGen/30	-1	-1	-1	-1	-1
	MaxGen/40	-1	-1	-1	-1	-1
	MaxGen/50	-1	-1	-1	-1	-1
g04	MaxGen/20	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	MaxGen/30	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	MaxGen/40	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	MaxGen/50	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
g05	MaxGen/20	5137.909	5134.538	5144.485	5161.126	5145.701
	MaxGen/30	5142.567	5142.026	5137.214	5138.390	5143.950
	MaxGen/40	5182.405	5142.735	5151.381	5138.728	5154.452
	MaxGen/50	5154.481	5134.937	5140.406	5135.445	5132.772
g06	MaxGen/20	-6277.733	-5745.900	-5154.238	-5573.369	-5680.405
	MaxGen/30	-6459.563	-6053.272	-5564.339	-5496.691	-5008.873
	MaxGen/40	-5936.136	-5728.611	-5300.747	-6961.814	-5431.247
	MaxGen/50	-5894.130	-5521.902	-6961.814	-6961.814	-6062.710
g07	MaxGen/20	24.658	24.708	24.477	24.872	24.760
	MaxGen/30	24.577	24.496	24.610	24.766	24.583
	MaxGen/40	24.690	24.559	24.775	24.596	24.521
	MaxGen/50	24.834	24.427	24.647	24.753	24.546
g08	MaxGen/20	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	MaxGen/30	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	MaxGen/40	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	MaxGen/50	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
g09	MaxGen/20	680.724	680.734	680.756	680.724	680.681
	MaxGen/30	680.739	680.787	680.793	680.970	680.701
	MaxGen/40	680.729	680.737	680.775	680.697	680.821
	MaxGen/50	680.902	680.723	680.748	680.727	680.712
g10	MaxGen/20	11127.857	11521.765	8599.069	12867.418	20374.472
	MaxGen/30	9180.307	9305.2742	13971.321	8696.503	20435.165
	MaxGen/40	13024.844	16499.153	17001.814	15172.649	19005.174
	MaxGen/50	8106.476	20650.701	10389.831	20490.951	20858.492
g11	MaxGen/20	0.76	0.76	0.76	0.76	0.77
	MaxGen/30	0.76	0.76	0.76	0.76	0.76
	MaxGen/40	0.76	0.76	0.76	0.77	0.76
	MaxGen/50	0.76	0.75	0.76	0.76	0.75
g12	MaxGen/20	-1	-1	-1	-1	-1
	MaxGen/30	-1	-1	-1	-1	-1
	MaxGen/40	-1	-1	-1	-1	-1
	MaxGen/50	-1	-1	-1	-1	-1
g13	MaxGen/20	0.054026	0.054172	0.439605	0.439755	0.054273
	MaxGen/30	0.440254	0.054098	0.440287	0.054048	0.054160
	MaxGen/40	0.054061	0.054019	0.054088	0.054186	0.439718
	MaxGen/50	0.054848	0.054050	0.054153	0.054375	0.124942

表 7.5 migration_interval 和 migration_size 的 20 组取值下得到的标准差

Fun	Size	1	5	10	15	20
	Interval	Std.	Std.	Std.	Std.	Std.
g01	MaxGen/20	6.6E-16	0	0	0	0
	MaxGen/30	0	0	0	0	0
	MaxGen/40	0	0	0	0	0
	MaxGen/50	0	0	0	0	0
g02	MaxGen/20	6.1E-03	2.1E-02	1.6E-02	1.4E-02	1.1E-02
	MaxGen/30	1.5E-02	1.7E-02	1.9E-02	1.6E-02	1.2E-02
	MaxGen/40	1.9E-02	1.6E-02	1.7E-02	1.5E-02	1.6E-02
	MaxGen/50	1.2E-02	1.9E-02	1.3E-02	1.9E-02	1.9E-02
g03	MaxGen/20	1.1E-05	8.1E-06	1.1E-05	9.1E-06	1.1E-05
	MaxGen/30	8.0E-06	9.8E-06	6.1E-06	9.0E-06	9.9E-06
	MaxGen/40	1.0E-05	1.7E-05	8.1E-06	1.3E-05	6.5E-06
	MaxGen/50	1.1E-05	1.1E-05	8.2E-06	7.7E-06	7.7E-06
g04	MaxGen/20	1.0E-11	9.9E-12	1.0E-11	1.0E-11	1.4E-11
	MaxGen/30	1.0E-11	9.4E-12	1.1E-11	1.1E-11	1.0E-11
	MaxGen/40	1.0E-11	1.0E-11	1.0E-11	1.1E-11	1.0E-11
	MaxGen/50	1.1E-11	9.9E-12	1.1E-11	9.9E-12	1.0E-11
g05	MaxGen/20	2.7E+00	2.2E+00	3.6E+00	6.4E+00	5.5E+00
	MaxGen/30	3.1E+00	3.1E+00	2.3E+00	2.9E+00	4.4E+00
	MaxGen/40	1.1E+01	3.5E+00	5.9E+00	3.1E+00	5.1E+00
	MaxGen/50	5.3E+00	2.3E+00	2.8E+00	2.0E+00	1.7E+00
g06	MaxGen/20	2.0E+02	2.2E+02	4.1E+02	3.8E+02	2.3E+02
	MaxGen/30	1.4E+02	1.7E+02	2.6E+02	2.7E+02	3.6E+02
	MaxGen/40	2.8E+02	2.3E+02	3.0E+02	1.9E-12	2.8E+02
	MaxGen/50	2.1E+02	2.8E+02	1.9E-12	1.9E-12	2.0E+02
g07	MaxGen/20	6.7E-02	7.2E-02	4.9E-02	1.1E-01	1.1E-01
	MaxGen/30	6.5E-02	5.1E-02	7.9E-02	1.1E-01	5.5E-02
	MaxGen/40	8.4E-02	6.8E-02	1.2E-01	5.3E-02	4.8E-02
	MaxGen/50	9.9E-02	3.6E-02	8.9E-02	8.3E-02	5.7E-02
g08	MaxGen/20	8.55E-18	7.29E-18	8.15E-18	8.55E-18	8.93E-18
	MaxGen/30	6.31E-18	7.73E-18	7.73E-18	8.15E-18	6.82E-18
	MaxGen/40	9.29E-18	6.31E-18	4.46E-18	5.76E-18	7.29E-18
	MaxGen/50	8.93E-18	8.55E-18	8.15E-18	9.29E-18	3.18E-17
g09	MaxGen/20	2.6E-02	2.7E-02	3.1E-02	2.3E-02	1.3E-02
	MaxGen/30	2.4E-02	3.9E-02	3.4E-02	6.7E-02	1.5E-02
	MaxGen/40	1.9E-02	2.6E-02	3.0E-02	1.7E-02	4.4E-02
	MaxGen/50	5.7E-02	1.9E-02	2.7E-02	2.2E-02	2.2E-02
g10	MaxGen/20	9.8E+02	1.2E+03	3.4E+02	1.1E+03	3.3E+03
	MaxGen/30	4.98E+02	4.1E+02	1.3E+03	3.8E+02	3.2E+03
	MaxGen/40	1.1E+03	2.0E+03	2.1E+03	1.6E+03	2.7E+03
	MaxGen/50	2.1E+02	3.2E+03	5.8E+02	2.9E+023	2.9E+03
g11	MaxGen/20	1.8E-03	1.9E-03	1.9E-03	1.8E-03	3.3E-03
	MaxGen/30	3.1E-03	2.6E-03	1.6E-03	2.6E-03	1.6E-03
	MaxGen/40	2.5E-03	2.5E-03	3.1E-03	3.1E-03	2.5E-03
	MaxGen/50	2.7E-03	1.1E-03	1.5E-03	2.6E-03	1.2E-03
g12	MaxGen/20	0	0	0	0	0
	MaxGen/30	0	0	0	0	0
	MaxGen/40	0	0	0	0	0
	MaxGen/50	0	0	0	0	0
g13	MaxGen/20	1.6E-05	4.1E-05	9.8E-02	9.8E-02	6.2E-05
	MaxGen/30	7.1E-02	2.7E-05	9.8E-02	3.0E-05	3.8E-05
	MaxGen/40	3.1E-05	2.0E-05	3.3E-05	4.6E-05	7.0E-02
	MaxGen/50	1.6E-04	2.3E-05	4.5E-05	7.8E-05	1.3E-02

表 7.6 migration_interval 和 migration_size 的 20 组取值下得到的可行解比率

Fun	Size	1	5	10	15	20
	Interval	Percent	Percent	Percent	Percent	Percent
g01	MaxGen/20	1	1	1	1	1
	MaxGen/30	1	1	1	1	1
	MaxGen/40	1	1	1	1	1
	MaxGen/50	1	1	1	1	1
g02	MaxGen/20	0.375	0.275	0.475	0.2	0.35
	MaxGen/30	0.55	0.425	0.4	0.35	0.4
	MaxGen/40	0.275	0.45	0.325	0.575	0.45
	MaxGen/50	0.425	0.3	0.425	0.35	0.375
g03	MaxGen/20	1	1	1	1	1
	MaxGen/30	1	1	1	1	1
	MaxGen/40	1	1	1	1	1
	MaxGen/50	1	1	1	1	1
g04	MaxGen/20	0.125	0.35	0.15	0.1	0.325
	MaxGen/30	0.05	0.375	0.075	0.225	0
	MaxGen/40	0.125	0.125	1	0.15	0.625
	MaxGen/50	0.575	0.7	0.125	0.475	0.1
g05	MaxGen/20	0	0	0	0	0
	MaxGen/30	0.775	0	0	0	0
	MaxGen/40	1	0	0	1	0
	MaxGen/50	0.55	0	0	0	1
g06	MaxGen/20	0.45	0.55	0	0.55	0.425
	MaxGen/30	0.525	0.575	0.475	0.75	0.65
	MaxGen/40	0.675	0.9	0.575	0.575	0.425
	MaxGen/50	0.375	0	0.55	0.5	0.625
g07	MaxGen/20	1	1	1	1	1
	MaxGen/30	1	1	1	1	1
	MaxGen/40	1	1	1	1	1
	MaxGen/50	1	1	1	1	1
g08	MaxGen/20	1	1	1	1	1
	MaxGen/30	1	1	1	1	1
	MaxGen/40	1	1	1	1	1
	MaxGen/50	1	1	1	1	1
g09	MaxGen/20	1	1	1	0.975	0.95
	MaxGen/30	1	1	0.975	1	1
	MaxGen/40	1	1	1	1	1
	MaxGen/50	1	1	0.975	1	0.975
g10	MaxGen/20	0.775	1	1	1	1
	MaxGen/30	1	1	0	0	0
	MaxGen/40	1	1	1	1	1
	MaxGen/50	1	0.45	1	1	0
g11	MaxGen/20	0.6	0.85	0.95	1	0.65
	MaxGen/30	0.875	0.85	0.925	1	1
	MaxGen/40	0.65	0.825	0.475	0.875	0.5
	MaxGen/50	0.425	0.475	0.55	0.6	0.825
g12	MaxGen/20	1	1	1	1	1
	MaxGen/30	1	1	1	1	1
	MaxGen/40	1	1	1	1	1
	MaxGen/50	1	1	1	1	1
g13	MaxGen/20	0	0	0	1	0
	MaxGen/30	1	1	1	0	0
	MaxGen/40	0	0	0	1	1
	MaxGen/50	0.95	0	0	0	0

表 7.7 30 次独立实验统计结果

Fun	Optimal	Best	Mean	Worst	Std.	Percentage
g01	-15	-15	-15	-15	0	1
g02	-0.803619	-0.803589	-0.785874	-0.740359	1.5E-02	0.575
g03	-1	-1	-1	-1	1.3E-05	1
g04	-30665.539	-30665.539	-30665.539	-30665.539	1.1E-11	0.15
g05	5126.498	5126.498	5127.973	5138.728	3.1E+00	1
g06	-6961.814	-6961.814	-6961.814	-6961.814	1.9E-12	0.575
g07	24.306	24.309	24.360	24.596	5.3E-02	1
g08	-0.095825	-0.095825	-0.095825	-0.095825	5.76E-18	1
g09	680.630	680.631	680.647	680.697	1.7E-02	1
g10	7049.248	7083.960	7863.292	15172.649	1.6E+03	1
g11	0.75	0.75	0.75	0.77	3.1E-03	0.875
g12	-1	-1	-1	-1	0	1
g13	0.053950	0.053950	0.053970	0.054186	4.6E-05	1

为了进一步说明算法投资组合的优化性能，图 7.1-图 7.7 给出了测试函数 g01-g13 的收敛曲线图。从图中可以看出，当迭代到 500 代时，所有测试函数的收敛曲线都非常接近甚至达到最优值。这也说明，对于 13 个测试函数来说，AP 具有较快的收敛速度。

图 7.8-图 7.14 给出 13 个测试函数迭代每代可行解比例图，此图可反映出算法对目标函数和约束违反度间的均衡能力。

除了测试函数 g02、g03 和 g13 外，其他测试函数在初始阶段可行解比例都为 0，这是由于初始时可行域较小造成的，此后可行解比例快速增长。另外，对于测试函数 g08 和 g12，在 500 代以内可行解比例收敛到 1，说明在有限代内种群可以达到相对稳定的可行解比例；对于测试函数 g01、g07 和 g09，当迭代代数增加时，种群可以达到相对稳定的可行解比例，并且最终可行解比例在 1 周围小幅波动；对于测试函数 g11 和 g13，可行解比例快速增长且在最大和最小值间波动，说明 AP 方法中约束处理方法起到作用，根据适应值函数值适时调整可行解比例；对于剩下的测试函数（g02-g06, g10），可行解比例在 0 到 1 间波动。

通过图 7.8-图 7.14 可以看到，可行解比例随着迭代次数增加可以达到 1 或 1 的邻域内。进而说明 AP 方法具有较强的均衡目标函数和约束违反能力，从而在迭代过程中表现出较强的优化性能。

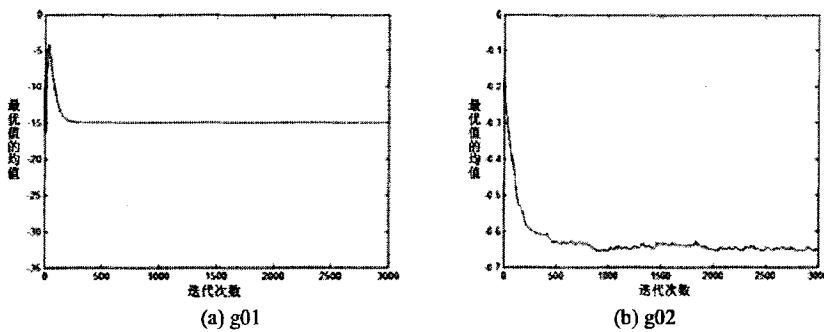


图 7.1 测试函数 g01、g02 收敛曲线图

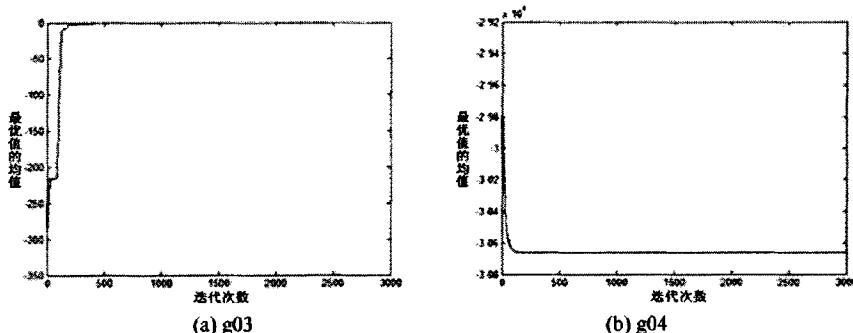


图 7.2 测试函数 g03、g04 收敛曲线图

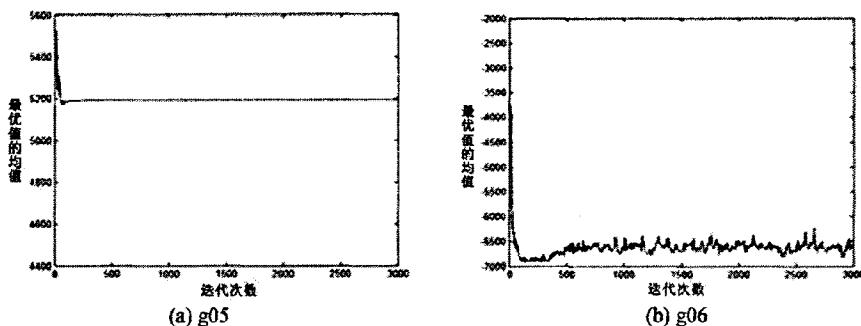


图 7.3 测试函数 g05、g06 收敛曲线图

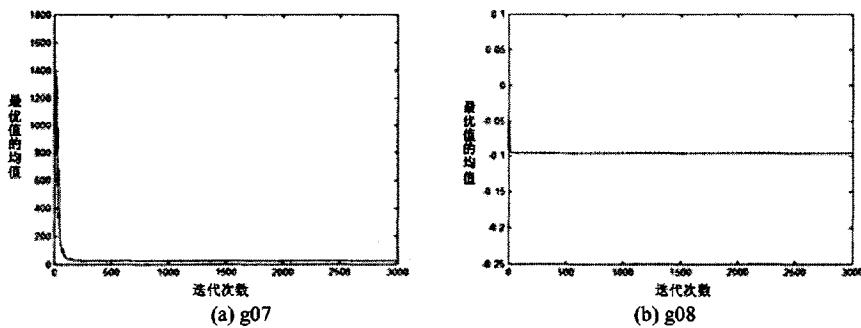


图 7.4 测试函数 g07、g08 收敛曲线图

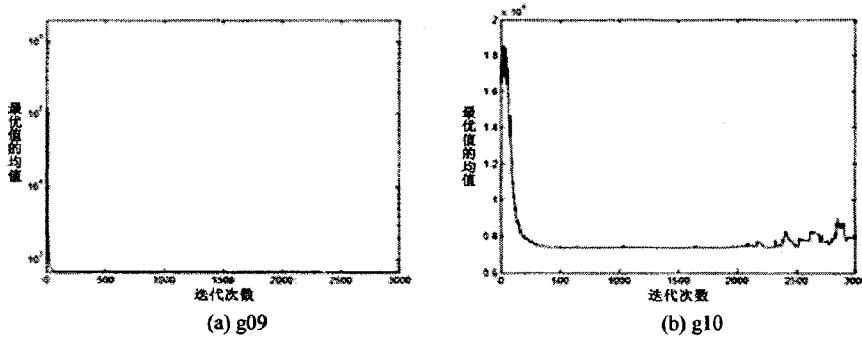


图 7.5 测试函数 g09、g10 收敛曲线图

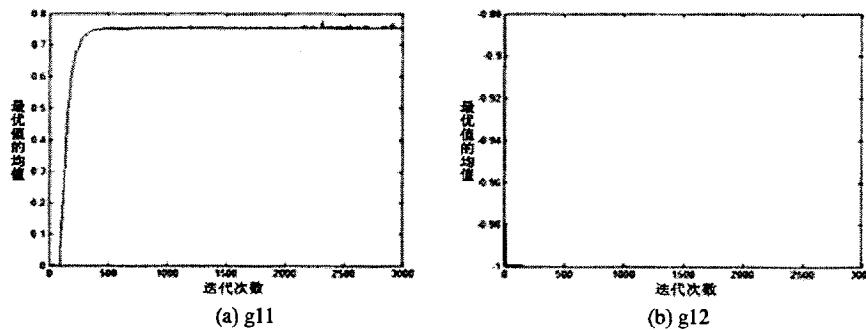


图 7.6 测试函数 g11、g12 收敛曲线图

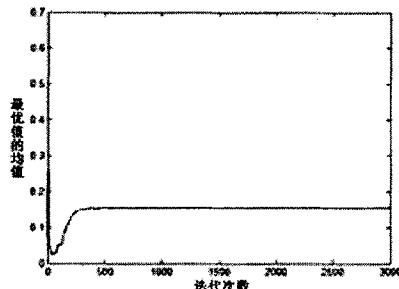


图 7.7 测试函数 g13 收敛曲线图

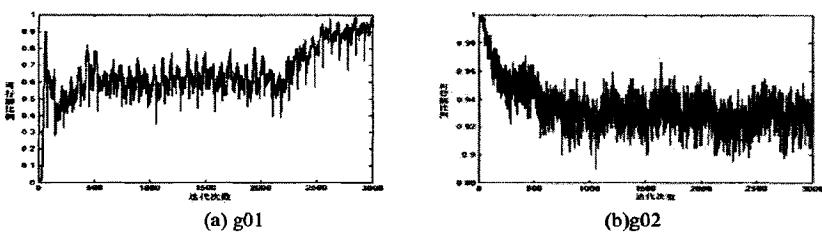


图 7.8 测试函数 g01、g02 可行解比例图

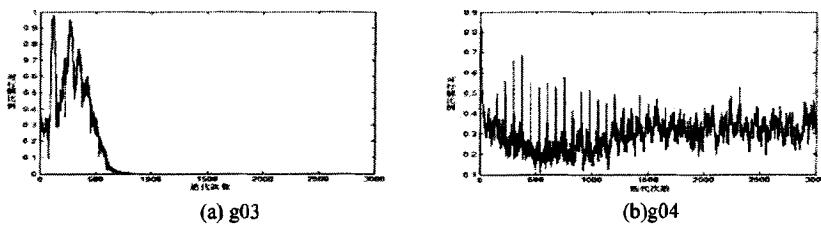


图 7.9 测试函数 g03、g04 可行解比例图

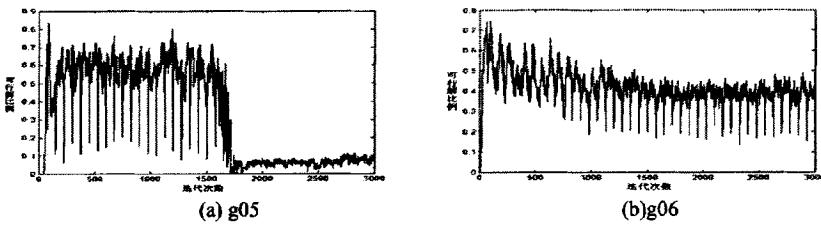


图 7.10 测试函数 g05、g06 可行解比例图

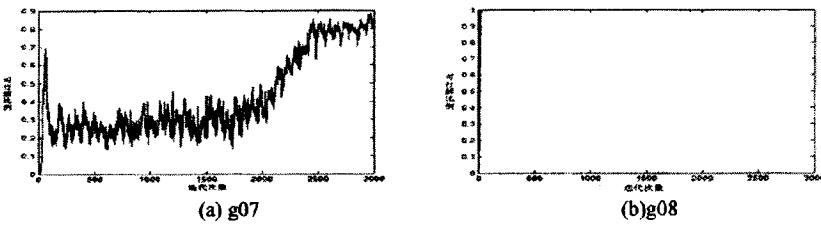


图 7.11 测试函数 g07、g08 可行解比例图

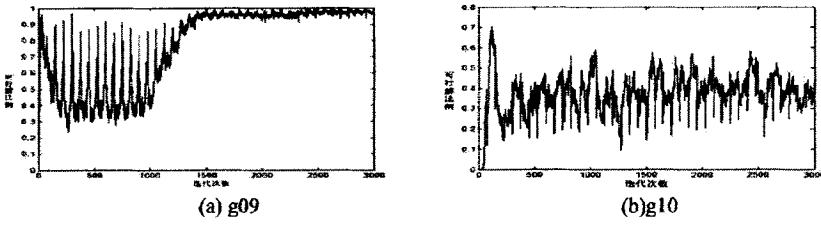


图 7.12 测试函数 g09、g10 可行解比例图

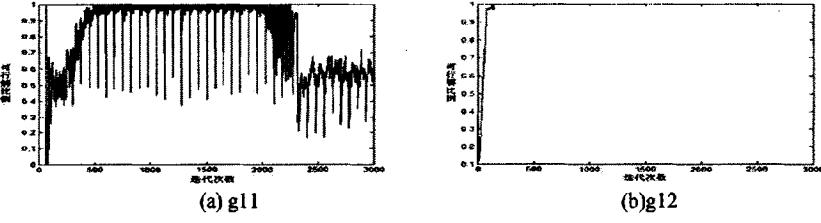


图 7.13 测试函数 g11、g12 可行解比例图

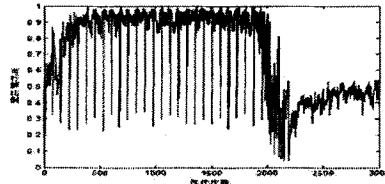


图 7.14 测试函数 g13 可行解比例图

为了说明算法投资组合的性能，将其得到的最好解与自适应模型进化策略和人工蜂群算法得到的最好解进行比较，比较结果见表 6.8。这里自适应模型进化策略和人工蜂群算法的参数设定分别与文献[173]和文献[174]相同，并且函数评价次数均为 240000 次。从表 6.8 中可以看到，三种算法在以下 7 个测试问题上均能找到全局最优解：g01, g03, g04, g06, g08, g11, g12。算法投资组合和自适应模型进化策略可以找到问题 g05 的最优解。对于问题 g07 和 g09，自适应模型进化策略可以达到最优值，而算法投资组合的结果接近于最优值并且优于人工蜂群算法。对于问题 g02 和 g10，尽管算法投资组合得到的结果接近最优值，但不如自适应模型进化策略和人工蜂群算法接近。

表 7.8 实验结果的最优值

Fun	Optimal	AP	ATMES	ABC
g01	-15	-15	-15	-15
g02	-0.803619	-0.803589	-0.803388	-0.803598
g03	-1	-1	-1	-1
g04	-30665.539	-30665.539	-30665.539	-30665.539
g05	5126.498	5126.498	5126.498	5126.484
g06	-6961.814	-6961.814	-6961.814	-6961.814
g07	24.306	24.309	24.306	24.330
g08	-0.095825	-0.095825	-0.095825	-0.095825
g09	680.630	680.631	680.630	680.634
g10	7049.248	7083.960	7052.253	7053.904
g11	0.75	0.75	0.75	0.75
g12	-1	-1	-1	-1
g13	0.053950	0.053950	0.53950	0.760

表 7.8 中的结果不能反映出本章提出的算法明显优于其他两种算法，为了能深入反映这一问题，在表 7.9 中给出了三种算法最终种群的可行解比率。很明显，在

大多数测试问题上，算法投资组合的最终种群可行解比率为 100%，而自适应进化策略则仅能在两个测试问题上达到这一比率。这表明算法投资组合比其他两个算法得到的结果更好。

表 7.9 最终种群的可行解比率

Fun	AP	ATMES
g01	1	0.75
g02	0.575	0.74
g03	1	0.54
g04	0.15	0.8
g05	1	0.49
g06	0.575	0.62
g07	1	0.57
g08	1	1
g09	1	0.99
g10	1	0.78
g11	0.875	0.73
g12	1	1
g13	1	0.7

为了进一步验证算法投资组合的性能，这里将其最好值与 HM, SR, ISR, OPA, ASCHEA, SMES, GA, DE 和 PSO 进行比较，这些算法的参数设置与文献[170]相同。需要说明的是 SMES, GA 和 DE 的函数评价次数与算法投资组合相同为 240000；SR, ISR, OPA 和 PSO 为 350000；HM 为 1400000；ASCHEA 为 1500000。比较实验结果见表 7.10 和 7.11。通过比较结果可以看出，算法投资组合在最好值上优于 HM, SR, ASCHEA, SMES, GA 和 PSO；与 ISR, OPA 和 DE 结果相同。考虑到函数评价次数，比较结果说明算法投资组合在约束优化问题上较其他算法能得到更好的结果。

众所周知，任何一个智能优化算法对于约束优化问题的优化性能与约束处理方法高度相关。事实上，实验 Deb 法则的算法由于选择过程中可行解始终优于不可行解而使得算法缺乏种群多样性。尽管人工蜂群算法中侦查蜂环节可以增强种群多样性，但不可行解中所包含的信息都会因为选择方式而损失。然而，在自适应模型进化策略中，就设计了对可行解和不可行解的均衡处理。不可行解的信息得到应用，但这样会影响算法的收敛速度。在算法投资组合中，将这两种方法的

优点进行了结合，避免了缺点，从而使得所给方法在求解约束优化问题中是可行的有效算法。以上的所有实验结果也进一步说明了算法投资组合具有更优良的优化性能。

表 7.10 测试问题 g01-g06 的实验结果

Fun	g01	g02	g03	g04	g05	g06
Optimal	-15	-0.803619	-1	-30665.539	5126.498	-6961.814
HM	-14.7864	-0.79953	-0.9997	-30664.5	-	-6952.1
SR	-15	-0.803515	-1	-30665.539	5126.497	-6961.814
ISR	-15	-0.803619	-1.001	-30665.539	5126.497	-6961.814
OPA	-15	-0.803619	-0.747	-30665.539	5126.497	-6961.814
ASCHEA	-15	-0.785	-1	-30665.5	5126.5	-6961.81
SMES	-14.44	-0.796231	-0.99	-30626.053	-	-6952.472
GA	-15	-0.803601	-1	-30665.539	5126.599	-6961.814
PSO	-15	-0.669158	-0.993930	-30665.539	5126.484	-6961.814
DE	-15	-0.472	-1	-30665.539	5126.484	-6954.434
AP	-15	-0.803589	-1	-30665.539	5126.498	-6961.814

表 7.11 测试问题 g07-g13 的实验结果

Fun	g07	g08	g09	g10	g11	g12	g13
Optimal	24.306	-0.095825	680.630	7049.248	0.75	-1	0.05395
HM	24.620	-0.0958250	680.91	7147.9	0.75	NA	NA
SR	24.307	-0.095825	680.630	7054.316	0.750	-1	0.053957
ISR	24.306	-0.095825	680.630	7049.248	0.750	-1	0.053942
OPA	24.306	-0.095825	680.630	7049.248	0.750	-1	0.447118
ASCHEA	24.3323	-0.095825	680.630	7061.13	0.75	NA	NA
SMES	31.097	-0.095825	685.994	9097.770	0.75	-1	0.134057
GA	24.327	-0.095825	680.632	7051.903	0.75	-1	0.053986
PSO	24.370153	-0.095825	680.630	7049.381	0.749	-1	0.085655
DE	24.306	-0.095825	680.630	7049.248	0.752	-1	0.385
AP	24.309	-0.095825	680.631	7083.96	0.75	-1	0.05395

7.4 小结

本章给出一种算法投资组合，该方法将进化策略和人工蜂群算法混合来求解

约束优化问题。算法投资组合是一种新的算法混合方法，可以综合这些算法的优点进行优化。对参数的统计分析给出算法参数的合理取值。比较研究发现，算法投资组合对约束优化问题是一种有效的求解方法。

第八章 结论和展望

人工蜂群算法是 2005 年由土耳其学者 Karaboga Dervis 通过模拟蜜蜂群体采蜜行为而提出的一种新型群体智能优化算法。该算法把蜜蜂群体分为雇佣蜂、跟随蜂、侦察蜂三种类型。算法的生物学背景简单，自诞生之日起，就由于算法简单、易于实现，并且收敛速度快而引起了众多学者的广泛关注。现已成功应用于神经网络训练、支持向量机优化、TSP 等实际领域，成为继遗传算法、人工蚁群算法、粒子群算法、差分进化算法之后的又一研究热点，对群体智能优化算法的发展与进步起到了推动作用。

但人工蜂群算法作为一种新兴的群体智能优化算法，仍然存在算法探索与开发之间的矛盾，算法的收敛速度不快，容易陷入局部最优解，同时算法的数学理论基础也十分薄弱。虽然原始人工蜂群算法与其他群智能优化算法相比具有很大优势，但仍有很大的改进空间，需要我们不断的探索。本文通过引入随机过程中的鞅理论分析了人工蜂群算法的几乎必然强收敛性，同时针对人工蜂群算法的不足和缺点，面向函数优化问题、约束函数优化问题、非负最小二乘问题提出了几种改进的人工蜂群算法，实验结果表明所提出算法的有效性。本文所做的主要工作现将归纳如下：

(1) 针对全局优化问题，提出了一种混合人工蜂群算法，算法通过对雇佣蜂和跟随蜂的搜索方程添加随机扰动项，扩大解的搜索范围，同时引入正交初始化的方法，取代随机初始化的方法，使解的分布更加均匀，利用 `benchmark` 函数测试，改进的算法比原始算法更有效。

(2) 针对函数优化问题，提出了一种改进人工蜂群算法，算法受差分进化算法的启发，对雇佣蜂和跟随蜂的搜索方程引入两种新的搜索机制，采用 P 概率进行控制，同时引入正交初始化的方法，利用 `benchmark` 函数测试，改进的人工蜂群算法比其他一些著名的人工蜂群算法改进算法和其他群体智能优化算法在收敛速度、解的精度和解的稳定性上更具有优势。

(3) 针对非负最小二乘问题，提出了一种有效的人工蜂群算法，该算法受粒子群算法和差分进化算法的启发，给出了两种新的搜索机制，采用 p 概率进行控制，改进的算法不在区分雇佣蜂和跟随蜂，统一为一种采蜜机制，和其他智能优化算法利用 `benchmark` 函数和非负最小二乘问题测试，表明所提出的有效人工蜂群算法在非负最小二乘问题上具有更好的优化性能。

(4) 提出了一种全局优化问题的有效混沌人工蜂群算法，并将该算法应用于非负线性最小二乘问题。为了克服人工蜂群算法的不足，算法中使用反学习初始

化作为种群初始化方法。为了进一步改进算法的性能，均衡算法的探索能力和开发能力，为新算法设计了新的搜索机制。另外，在算法中加入混沌局部搜索机制，使得算法在最优解周围进行局部搜索。利用 `benchmark` 测试函数和非负最小二乘问题的实例测试，表明所提出的算法比其他改进算法和其他智能优化算法收敛速度更快，解的精度更高。

(5) 针对约束函数优化问题，提出了一种混合人工蜂群算法，该算法基于进化策略和人工蜂群算法的投资组合，大部分智能优化算法在处理约束函数优化问题时都是把计算时间投入到一种算法或者是约束处理技术上，而本文提出的算法把运行时间分配给不同的算法，同时在算法中设计了一种迁移手段，有利于两种算法之间的信息交流，利用 `benchmark` 函数测试，表明所提出的混合人工蜂群算法在求解约束函数优化问题时成功率以及解的有效性更优秀。

(6) 针对原始人工蜂群算法的数学理论基础较为薄弱，本文尝试引入随机过程中的鞅理论分析了算法的几乎必然强收敛性，原有的人工蜂群算法收敛性证明是在以概率 1 意义下收敛的，不能保证算法在有限步内收敛到全局最优解，通过引入鞅过程，可以证明算法确保能在有限步内收敛到全局最优解，所证结论为人工蜂群算法的改进、收敛性证明以及应用提供了新的理论分析工具。

本文虽然在人工蜂群算法的收敛性证明、算法的改进及其应用方面做了一些研究，但限于个人能力，仍有不足和有待进一步探讨的问题：

(1) 本文对于人工蜂群算法的研究同以往研究群智能优化算法的方法类似，主要是引入在其他算法中应用较为先进的技术，通过大量的函数仿真实验来验证改进算法的有效性。但改进人工蜂群算法的收敛性分析研究还不够深入，需要我们进一步考虑。

(2) 混合算法主要是融合不同算法的优点，一般能够有效地求解复杂的优化问题，但如何更好地吸收不同算法的优点，克服单个算法的缺点，到底是串行式还是并行式，或其他混合方法更优秀，还需进一步研究。

(3) 本文借助随机过程中的鞅理论证明了原始人工蜂群算法的收敛性，是否能够利用其他数学理论来证明算法的收敛性，以及能否证明改进人工蜂群算法的全局收敛性，还有待探讨。

(4) 现实生活中经济、管理、工程上的优化问题，大部分是多目标优化问题和动态优化问题，但现阶段利用人工蜂群算法求解多目标优化问题和动态优化问题的文献还很少，如何利用人工蜂群算法来求解多目标优化问题和动态优化问题是我们的下一步研究的重点。

参考文献

- [1] Karaboga D. An idea based on honey bee swarm for numerical optimization[R]. Technical report. Computer Engineering Department, Engineering Faculty, Erciyes University, 2005.
- [2] Basturk B, Karaboga D. An artificial bee colony (abc) algorithm for numeric function optimization[C]. IEEE swarm intelligence symposium 2006, Indianapolis, IN, USA, 2006.
- [3] Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm[J]. Journal of Global Optimization, 2007, 39(3): 459-471.
- [4] Karaboga D, Basturk B. On the performance of artificial bee colony (abc) algorithm[J]. Applied Soft Computing, 2008, 8(1): 687-697.
- [5] Karaboga D, Akay B. A comparative study of artificial bee colony algorithm[J]. Applied Mathematic and Computation, 2009, 214(1): 108-132.
- [6] Mala DJ, Kamalapriya M, Shobana R, Mohan V. A non-pheromone based intelligent swarm optimization technique in software test suite optimization[C]. IAMA: 2009 international conference on intelligent agent and multi-agent systems, IEEE Madras Section; IEEE Computer Society, Madras Chapter; Computer Society of India Div II; Council of Science & Industrial Research; Govt India, Department of Information Technology, 2009: 188-192.
- [7] Krishnanand KR, Nayak SK, Panigrahi BK, Rout PK. Comparative study of five bio-inspired evolutionary optimization techniques[C]. Abraham A, Herrera F, Carvalho A, Pai V. 2009 world congress on nature and biologically inspired computing (NABIC 2009), 2009: 1230-1235.
- [8] Karaboga D, Akay B. Artificial bee colony (abc), harmony search and bees algorithms on numerical optimization[C]. 2009 innovative production machines and systems virtual conference (IPROMS 2009), 2009.
- [9] Li H, Liu K, Li X. A comparative study of artificial bee colony, bees algorithms and differential evolution on numerical benchmark problems[M]. Cai Z, Tong HJ, Kang Z, Liu Y. Computational intelligence and intelligent systems, China University of Geosciences; China University of Geosciences, School of Computer Science, Communications in computer and information science, vol 107, 2010: 198-207.
- [10] Chu SC, Huang HC, Roddick J, Pan JS. Overviewof algorithms for swarm intelligence[C]. Jedrzejowicz P, Nguyen N, Hoang K. Computational collective intelligence. Technologies and applications. Lecture notes in computer science, vol 6922. Springer, Berlin, 2011: 28-41.
- [11] Ruiz-Vanoye J, Daz-Parra O. Similarities between meta-heuristics algorithms and the science

- of life[J]. Central European Journal of Operations Research, 2011, 19(4): 445-466.
- [12] Mohammed, El-Abd. Performance assessment of foraging algorithms vs. evolutionary algorithms[J]. Information Science, 2012, 182(1): 243-263.
- [13] Bao L, Zeng JC. Comparison and analysis of the selection mechanism in the artificial bee colony algorithm[C]. International Conference on Hybrid Intelligent Systems, Shenyang 2009: 411-416.
- [14] Guo P, Cheng W, Liang J. Global artificial bee colony search algorithm for numerical function optimization[C]. 2011 seventh international conference on natural computation (ICNC), vol 3, Shanghai, 2011: 1280-1283.
- [15] Alam MS, Ul Kabir MW, Islam MM. Self-adaptation of mutation step size in artificial bee colony algorithm for continuous function optimization[C]. 2010 13th international conference on computer and information technology (ICCIT), Dhaka, 2010: 69-74.
- [16] Diwold K, Aderhold A, Scheidler A, Middendorf M. Performance evaluation of artificial bee colony optimization and new selection schemes[J]. Memetic Computing, 2011, 3(3): 149-162.
- [17] Zou W, Zhu Y, Chen H, Ku T. Clustering approach based on von neumann topology artificial bee colony algorithm[C]. 2011 international conference on data mining (DMIN'11), Las Vegas, 2011.
- [18] Karaboga D, Basturk B. Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems[C]. Proceedings of the 12th international fuzzy systems association world congress on foundations of fuzzy logic and soft computing. Springer, Berlin, IFSA '07, 2007: 789-798.
- [19] Stanarevic N, Tuba M, Bacanin N. Enhanced artificial bee colony algorithm performance[C]. Proceedings of the 14thWSEAS international conference on computers: part of the 14thWSEAS CSCC multiconference-volume II, World Scientific and Engineering Academy and Society (WSEAS). Stevens Point, Wisconsin, USA, ICCOMP'10, 2010: 440-445.
- [20] Karaboga D, Akay B. A modified artificial bee colony (abc) algorithm for constrained optimization problems[J]. Applied Soft Computing, 2011, 11(3): 3021-3031.
- [21] Brajevic I, Tuba M, SuboticM. Performance of the improved artificial bee colony algorithm on standard engineering constrained problems[J]. International Journal of Mathematics and Computers in Simulation, 2011, 5(2): 135-143.
- [22] Subotic M, Tuba M, Stanarevic N. Parallelization of the artificial bee colony (abc) algorithm[C]. Proceedings of the 11th WSEAS international conference on nural networks and 11th WSEAS international conference on evolutionary computing and 11th WSEAS international conference on Fuzzy systems, World Scientific and Engineering Academy and Society (WSEAS). Stevens Point, Wisconsin, USA, NN'10/EC'10/FS'10, 2010: 191-196.

- [23] Parpinelli RS, Benitez CMV, Lopes HS. Parallel approaches for the artificial bee colony algorithm[C]. Panigrahi BK, Shi Y, Lim MH, Hiot LM, Ong YS. Handbook of swarm intelligence, adaptation, learning, and optimization, vol 8. Springer, Berlin , 2010: 329-345.
- [24] Luo R, Pan TS, Tsai PW, Pan JS. Parallelized artificial bee colony with ripple-communication strategy[C]. 2010 fourth international conference on genetic and evolutionary computing (ICGEC), Shenzhen, 2010: 350-353.
- [25] Banharnsakun A, Achalakul T, Sirinaovakul B. Artificial bee colony algorithm on distributed environments[C]. 2010 second world congress on nature and biologically inspired computing (NaBIC), Fukuoka, 2010: 13-18.
- [26] Liu X, Cai Z. Artificial bee colony programming made faster[C]. 2009. ICNC '09. Fifth international conference on natural computation, vol 4, Tianjin, 2009: 154-158.
- [27] Zhu G, Kwong S. Gbest-Guided Artificial Bee ColonyAlgorithm for Numerical FunctionOptimization[J]. Applied Mathematics and Computation, 2010, 217(7): 3166-3173.
- [28] Gao W, Liu S. Improved artificial bee colony algorithm for global optimization[J]. Information Processing Letters, 2011, 111(17): 871-882.
- [29] Li G, Niu P, Xiao X. Development and investigation of efficient artificial bee colony algorithm for numerical function optimization[J]. Applied Soft Computing, 2012, 12(1): 320-332.
- [30] Gao WF, Liu SY. A modified artificial bee colony algorithm[J]. Computers and Operations Research, 2012, 39(3): 687-697
- [31] Lin JH, Lin MR, Huang LR. A novel bee swarm optimization algorithm with chaotic sequence and psychology model of emotion[C]. Proceedings of the 9th WSEAS international conference on systems theory and scientific computation, World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA, 2009: 87-92.
- [32] Alatas B. Chaotic bee colony algorithms for global numerical optimization[J]. Expert Systems with Applications, 2010, 37(8): 5682-5687.
- [33] Wu B, Fan SH. Improved artificial bee colony algorithm with chaos[C]. Yu Y, Yu Z, Zhao J. Computer science for environmental engineering and ecoinformatics, Communications in computer and information science, vol 158. Springer, Berlin , 201: 51-56.
- [34] Hedayatzadeh R, Hasanizadeh B, Akbari R, Ziarati K. A multi-objective artificial bee colony for optimizing multi-objective problems[C]. 2010 3rd international conference on advanced computer theory and engineering (ICACTE), vol 5, Chengdu, 2010: V5-277-V5-281.
- [35] Zou W, Zhu Y, Chen H, Zhang B. Solving multiobjective optimization problems using artificial bee colony algorithm[J]. Discret Dyn Nat Soc, 2012, Vol. 2012, Article ID 569784, 37pages.
- [36] Quan H, Shi X. On the analysis of performance of the improved artificial-bee-colony algorithm[C]. Proceedings of the 2008 fourth international conference on natural computation,

- vol 07, ICNC '08, Jinan, 2008: 654-658.
- [37] Kang F, Li J, Ma Z. Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions[J]. Information Science, 2011, 181(16): 3508-3531.
- [38] Jianjun Liu, Hongqiu Zhu, Qiang Ma, Lanlan Zhang, Honglei Xu. An artificial bee colony algorithm with guide of global & local optima and asynchronous scaling factors for numerical optimization [J]. Applied Soft Computing, 2015, 37: 608-618.
- [39] Kang F, Li J, Xu Q. Hybrid simplex artificial bee colony algorithm and its application in material dynamic parameter back analysis of concrete dams[J]. Journal of Hydraulic Engineering, 2009, 40(6): 736-742.
- [40] Duan HB, Xu CF, Xing ZH. A hybrid artificial bee colony optimization and quantum evolutionary algorithm for continuous optimization problems[J]. International Journal of Neural Systems, 2010, 20(1): 39-50.
- [41] Kang F, Li J, Li H, Ma Z, Xu Q. An improved artificial bee colony algorithm[C]. 2010 2nd international workshop on intelligent systems and applications (ISA), Wuhan, 2010: 1-4.
- [42] Zhao H, Pei Z, Jiang J, Guan R, Wang C, Shi X. A hybrid swarm intelligent method based on genetic algorithm and artificial bee colony[C]. Tan Y, Shi YH, Tan KC. Advances in swarm intelligence, proceedings, Lecture notes in computer science, vol 6145, 2010: 558-565.
- [43] El-AbdM. Ahybrid abc-spspso algorithm for continuous function optimization[C]. 2011 IEEE symposium on swarm intelligence (SIS), Paris, 2011: 1-6.
- [44] Zhong Y, Lin J, Ning J, Lin X. Hybrid artificial bee colony algorithm with chemotaxis behavior of bacterial foraging optimization algorithm[C]. 2011 seventh international conference on natural computation (ICNC), vol 2, Shanghai, 2011: 1171-1174.
- [45] Kang F, Li J, Ma Z, Li H. Artificial bee colony algorithm with local search for numerical optimization[J]. Journal of Software, 2011, 6(3): 490-497.
- [46] 暴励, 曾建潮.一种双种群差分蜂群算法[J]. 控制理论与应用, 2011, 28(2): 266-272.
- [47] Zhiyong Li, Weiyou Wang, Yanyan Yan, Zheng Li. PS-ABC: A hybrid algorithm based on particle swarm and artificial bee colony for high-dimensional optimization problems [J]. Expert System with Applications, 2015, 42(22): 8881-8895.
- [48] Akay B, Karaboga D. Solving integer programming problems by using artificial bee colony algorithm[C]. Serra R, Cucchiara R. AI (ASTERISK) IA 2009: emergent perspectives in artificial intelligence. Italian Association of Artificial Intelligence; University Modena Reggio Emilia, Lecture notes in artificial intelligence, vol 5883, 2009: 355-364.
- [49] Wang J, Li T, Ren R. A real time idss based on artificial bee colony-support vector machine algorithm[C]. 2010 third international workshop on advanced computational intelligence (IWACI), SuZhou, 2010: 91-96.

- [50] Karaboga D, Akay B. Artificial bee colony (abc) algorithm on training artificial neural networks[C]. 2007 IEEE 15th signal processing and communications applications, vols 1-3, Eskisehir, 2007: 818-821.
- [51] Karaboga D, Ozturk C. Neural networks training by artificial bee colony algorithm on pattern classification[J]. Neural Network World, 2009, 19(3): 279-292.
- [52] Garro BA, Sossa H, Vazquez RA. Artificial neural network synthesis by means of artificial bee colony (abc) algorithm[C]. 2011 IEEE congress on evolutionary computation (CEC), New Orleans, LA, 2011: 331-338.
- [53] Rao RS, Narasimham SVL, Ramalingaraju M. Optimization of distribution network configuration for loss reduction using artificial bee colony algorithm[J]. International Journal of Electrical Power and Energy Systems Engineering, 2008, 1(2): 116-122.
- [54] Abu-Mouti FS, El-Hawary ME. Modified artificial bee colony algorithm for optimal distributed generation sizing and allocation in distribution systems[C]. 2009 IEEE Electrical power energy conference (EPEC), Montreal, 2009: 1-9.
- [55] Ozturk A, Cobanli S, Erdosmus P, Tosun S. Reactive power optimization with artificial bee colony algorithm[J]. Scientific Research and Essays, 2010, 5(19): 2848-2857.
- [56] Cobanli S, Ozturk A, Guvenc U, Tosun S. Active power loss minimization in electric power systems through artificial bee colony algorithm[J]. International Review of Electrical Engineering, 2010, 5(5, Part b): 2217-2223.
- [57] Ayan K, Kılıç U. Optimal reactive power flow solution with chaotic artificial bee colony[C]. 6th international advanced technologies symposium (IATS11), Elazīg, Turkey, 2011: 20-24.
- [58] Özyön S, Ya,sar C, Özcan G, Temurta,s H. An artificial bee colony algorithm (abc) aproach to environmental economic power dispatch problems (in Turkish)[C]. 2011 national electrical-electronics and computer symposium, Elazīg, Turkey, 2011: 222-228.
- [59] Bijami E, Shahriari-kahkeshi M, Zamzam H. Simultaneous coordinated tuning of power system stabilizers using artificial bee colony algorithm[C]. 26th international power system conference (PSC), 2011: 1-8.
- [60] Dutta R, Ganguli R,Mani V. Swarm intelligence algorithms for integrated optimization of piezoelectric actuator and sensor placement and feedback gains[J]. Smart Materials and Structures, 2011, 20(10): 105018.
- [61] Rao RV, Pawar PJ. Parameter optimization of a multi-pass milling process using non-traditional optimization algorithms[J]. Applied Soft Computing, 2010, 10(2):445-456.
- [62] Gomez-Iglesias A, Vega-Rodriguez MA, Castejon F, Cardenas-Montes M, Morales-Ramos E. Artificial bee colony inspired algorithm applied to fusion research in a grid computing environment[C]. 2010 18th Euromicro international conference on parallel, distributed and

- network-based processing (PDP), Pasi, 2010: 508-512.
- [63] Yao B, Yang C, Hu J, Yu B. The optimization of urban subway routes based on artificial bee colony algorithm[M]. Chen F, Gao L, Bai Y. Key technologies of railway engineering—high speed railway, heavy haul railway and urban rail transit. Beijing Jiaotong University, Beijing, 2010: 747-751.
- [64] Hadidi A, Azad SK, Azad SK. Structural optimization using artificial bee colony algorithm[C]. 2nd international conference on engineering optimization, 2010.
- [65] Hetmaniok E, Slota D, Zielonka A. Solution of the inverse heat conduction problem by using the abc algorithm[C]. Szczyka M, Kryszkiewicz M, Ramanna S, Jensen R, Hu QH. Rough sets and current trends in computing, proceedings, Lecture notes in artificial intelligence, vol 6086, 2010: 659-668.
- [66] Mandal SK, Chan FTS, Tiwari MK. Leak detection of pipeline: An integrated approach of rough set theory and artificial bee colony trained svm[J]. Expert Systems with Applications, 2011, 39(3): 3071-3080.
- [67] Karaboga D, Ozturk C. Fuzzy clustering with artificial bee colony algorithm[J]. Scientific Research and Essays, 2010, 5(14):1899-1902.
- [68] Karaboga D, Ozturk C. A novel clustering approach: Artificial bee colony (abc) algorithm[J]. Applied Soft Computing, 2011, 11(1):652-657.
- [69] Zhang C, Ouyang D, Ning J. An artificial bee colony approach for clustering[J]. Expert Systems with Applications, 2010, 37(7):4761-4767.
- [70] Hsieh TJ, Yeh WC. Knowledge discovery employing grid scheme least squares support vectormachines based on orthogonal design bee colony algorithm[J]. IEEE Transaction on System Man and Cybernetics, Part B: Cybern, 2011, 41(5):1198-1212.
- [71] Babu MSP, Rao NT. Implementation of artificial bee colony (abc) algorithm on garlic expert advisory system[J]. International Journal of Computer Science and Research, 2010, 1(1):69-74.
- [72] Shukran MAM, Chung YY, Yeh WC, Wahid N, Zaidi AMA. Artificial bee colony based data mining algorithms for classification tasks[J]. Modern Applied Science, 2011, 5(4):217-231.
- [73] Udgata SK, Sabat SL, Mini S. Sensor deployment in irregular terrain using artificial bee colony algorithm[J]. Abraham A, Herrera F, Carvalho A, Pai V. 2009 world congress on nature and biologically inspired computing (NABIC 2009), 2009: 1308-1313.
- [74] Mini S, Udgata S, Sabat S. Artificial bee colony based sensor deployment algorithm for target coverage problem in 3-d terrain[C]. Natarajan R, Ojo A. Distributed computing and internet technology. Lecture notes in computer science, vol 6536. Springer, Berlin, 2011: 313-324.
- [75] Öztürk C, Karaboga D, Görkemli B. Artificial bee colony algorithm for dynamic deployment of wireless sensor networks[J]. Turkish Journal of Electrical Engineering and Computer

- Sciences, 2012, 20(2):1-8.
- [76] Ye Z, Zeng M, Hu Z, Chen H. Image enhancement based on artificial bee colony algorithm and fuzzy set[C]. International Symposium on Information Engineering and Electronic Commerce, 3rd (IEEC 2011), 2011.
- [77] Chidambaram C, Lopes HS. An improved artificial bee colony algorithm for the object recognition problem in complex digital images using template matching[J]. International Journal of Natural Computing Research, 2010, 1(2):54-70.
- [78] Nebti S, Boukerram A. Handwritten digits recognition based on swarm optimization methods[M]. Zavoral F, Yaghob J, Pichappan P, El-Qawasmeh E. Networked digital technologies, Communications in computer and information science, vol 87. Springer, Berlin, 2010: 45-54.
- [79] Zhang Y, Wu L. Optimal multi-level thresholding based on maximum tsallis entropy via an artificial bee colony approach[J]. Entropy, 2011, 13(4):841-859.
- [80] Kazim Hanbay, M. Ftih Talu. Segmentation of SAR images using improved artificial bee colony algorithm and neutrosophic set [J]. Applied Soft Computing, 2014, 21: 433-443.
- [81] A. K. Bhandari, A. Kumar, G. K. Singh. Modified artificial bee colony based computationally efficient multilevel thresholding for satellite image segmentation using Kapur's, Otsu and Tsallis functions [J]. Expert System with Applications, 2015, 42(3): 1573-1601.
- [82] Tahir Sag, Mehmet Çunkas. Color image segmentation based on multiobjective artificial bee colony optimization [J]. Applied Soft Computing, 2015, 34: 389-401.
- [83] Xu C, Duan H. Artificial bee colony (abc) optimized edge potential function (epf) approach to target recognition for low-altitude aircraft[J]. Pattern Recognition Letters, 2010, 31(13):1759-1772.
- [84] Ma M, Liang J, Guo M, Fan Y, Yin Y. Sar image segmentation based on artificial bee colony algorithm[J]. Applied Soft Computing, 2011, 11(8):5205-5214.
- [85] Akay B, Karaboga D. Wavelet packets optimization using artificial bee colony algorithm[C]. 2011 IEEE congress on evolutionary computation (CEC), New Orleans, LA, 2011: 89-94.
- [86] Amer Draa, Amira Bouaziz. An artificial bee colony algorithm for image contract enhancement [J]. Swarm and Evolutionary Computation, 2014, 16: 69-84.
- [87] Pan QK, Tasgetiren MF, Suganthan PN, Chua TJ. A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem[J]. Information Science, 2011, 181(12):2455-2468.
- [88] Jun-Qing Li, Quan-Ke Pan, M. Fatih Tasgetiren. A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities [J]. Applied Mathematical Modelling, 2014, 38(3): 1111-1132.

- [89] Quan-Ke Pan, Ling Wang, Jun-Qing Li, Jun-Hua Duan. A novel discrete artificial bee colony algorithm for the hybrid flowshop scheduling problem with makespan minimization [J]. Omega, 2014, 45: 42-56.
- [90] A. Muthiah, R. Rajkumar. A comparison of artificial bee colony algorithm and genetic algorithm to minimize the makespan for job shop scheduling [J]. Procedia Engineering, 2014, 97: 1745-1754.
- [91] Imma Ribas, Ramon Companys, Xavier Tort-Martorell. An efficient discrete artificial bee colony algorithm for the blocking flow shop problem with total flowtime minimization [J]. Expert System with Applications, 2015, 42(15-16): 6155-6167.
- [92] Pal A, Chan FTS, Mahanty B, Tiwari MK. Aggregate procurement, production, and shipment planning decision problem for a three-echelon supply chain using swarm-based heuristics[J]. International Journal of Production Research, 2011, 49(10):2873-2905.
- [93] Sundar S, Singh A, Rossi A. An artificial bee colony algorithm for the 0-1 multidimensional knapsack problem[C]. Ranka S, Banerjee A, Biswas KK, Dua S, Mishra P, Moona R, Poon SH, Wang CL. Contemporary computing, pt 1. Jaypee Institute of Information Technology; University of Florida, Communications in computer and information science, vol 94, 2010: 141-151
- [94] Singh A. An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem. Applied Soft Computing, 2009, 9(2):625-631.
- [95] Haris P, Gopinathan E, Ali C. Artificial bee colony and tabu search enhanced ttcn assisted mmse multi-user detectors for rank deficient sdma-ofdm system[J]. Wireless Personal Communications, 2012, 65(2): 425-442.
- [96] Manoj VJ, Elias E. Artificial bee colony algorithm for the design of multiplier-less nonuniform filter bank transmultiplexer[J]. Information Science, 2011, 192(1): 193-203.
- [97] Bernardino A, Bernardino E, Snchez-Prez J, Gmez-Pulido J, Vega-Rodrguez M. Efficient load balancing for a resilient packet ring using artificial bee colony[C]. Di Chio C, Brabazon A, Di Caro G, Ebner M, Farooq M, Fink A, Grahl J, Greenfield G, Machado P, ONeill M, Tarantino E, Urquhart N. Applications of evolutionary computation. Lecture notes in computer science, vol 6025. Berlin: Springer, 2010: 61-70.
- [98] Suri B, Kalkal S. Review of artificial bee colony algorithm to software testing[J]. International Journal of Research & Reviews in Computer Science, 2011, 2(3):706-711.
- [99] AdiSrikanth , Kulkarni NJ, Naveen KV, Singh P, Srivastava PR. Test case optimization using artificial bee colony algorithm[C]. Abraham A, Mauri JL, Buford JF, Suzuki J, Thamphi SM. Advances in computing and communications, communications in computer and information science. Berlin: Springer, 2011: 570-579.

- [100]Koc E, Ersoy N, Andac A, Camlidere ZS, Cereci I, Kilic H. An empirical study about search-based refactoring using alternative multiple and population-based search techniques[C]. Gelenbe E, Lent R, Sakellari G. Computer and information sciences II. London: Springer, 2012: 59-66.
- [101]Yan G, Li C. An effective refinement artificial bee colony optimization algorithm based on chaotic search and application for pid control tuning[J]. Journal of Computational Information System, 2011, 7(9):3309-3316.
- [102]Gao F, Qi Y, Yin Q, Xiao J. A novel non-lyapunov approach in discrete chaos system with rational fraction control by artificial bee colony algorithm. In: 2010 IEEE international conference on progress in informatics and computing (PIC), Shanghai, 2010: 317-320.
- [103]Gao F, Qi Y, Yin Q, Xiao J. An artificial bee colony algorithm for unknown parameters and timedelays identification of chaotic systems[C]. In: 2010 5th international conference on computer sciences and convergence information technology (ICCIT), Seoul, 2010: 659-664.
- [104]Bahamish HAA, Abdullah R, Salam RA. Protein tertiary structure prediction using artificial bee colony algorithm[C]. In: 2009 third Asia international conference on modelling and simulation, Bali, 2009: 258-263.
- [105]Lin CJ, Lee CY. An efficient artificial bee colony algorithm for 3d protein folding simulation[C]. In: 17th national conference on fuzzy theory and its applications, 2009: 705-710.
- [106]Bahamish HAA, Abdullah R. Prediction of c-peptide structure using artificial bee colony algorithm[C]. In: 2010 international symposium in information technology (ITSim), Kuala Lumpur, 2010: 754-759.
- [107]Bai Li, Raymond Chiong, Mu Lin. A balance-evolution artificial bee colony algorithm for protein structure optimization based on a three-dimensional AB off-lattice model [J]. Computational Biology and Chemistry, 2015, 54: 1-12.
- [108]胡玉荣,丁立新,谢大同等.采用渐变与突变机制的反向人工蜂群算法[J].武汉大学学报(理学版),2013,59(2): 123-128.
- [109]孙晓雅,林焰.改进的人工蜂群算法求解任务指派问题[J].微电子学与计算机, 2012,29(1): 23-26.
- [110]毕晓君,王艳娇.改进人工蜂群算法[J].哈尔滨工程大学学报, 2012,33(1): 117-123.
- [111]汪继文,杨丹,邱剑锋等.改进人工蜂群算法求解非线性方程组[J].安徽大学学报(自然科学版), 2014,38(3): 16-23.
- [112]康飞,李俊杰,许青.混合蜂群算法及其在混凝土坝动力材料参数反演中的应用[J].水利学报, 2009,40(6): 736-742.
- [113]高卫峰,刘三阳,姜飞等.混合人工蜂群算法[J].系统工程与电子技术, 2011,33(5): 1167-1170.

- [114] 鲁建厦, 翁耀炜, 李修琳等.混合人工蜂群算法在混流装配线排序中的应用[J].计算机集成制造系统, 2014,20(1): 121-127.
- [115] 王文亮, 王智广, 刘伟峰.基于GPU加速的细粒度并行人工蜂群算法[J].微电子学与计算机, 2013,30(3): 110-114.
- [116] 刘三阳, 张平, 朱明敏.基于局部搜索的人工蜂群算法[J].控制与决策, 2014,29(1): 123-128.
- [117] 王文全, 黄胜, 侯远航等.基于改进人工蜂群算法的大型舰船主尺度优化[J].武汉理工大学学报, 2012,34(6): 58-62.
- [118] 葛宇, 梁静, 王学平.基于极值优化策略的改进的人工蜂群算法[J].计算机科学, 2013,40(6): 247-251.
- [119] 马文强, 唐秋华, 张超勇等.基于离散人工蜂群算法的炼钢连铸调度优化方法[J].计算机集成制造系统, 2014,20(3): 586-594.
- [120] 李鑫滨, 刘磊, 马锴.基于离散人工蜂群算法的认知无线电频谱分配[J].系统工程与电子技术, 2012,34(10): 2136-2141.
- [121] 张新明, 李晓安, 何文涛等.基于排名映射概率的混沌人工蜂群算法[J].计算机科学, 2013,40(12): 98-103.
- [122] 李牧东, 熊伟, 郭龙.基于人工蜂群算法的DV-Hop定位改进[J].计算机科学, 2013,40(1): 33-36.
- [123] 胡中华, 赵敏.基于人工蜂群算法的TSP仿真[J].北京理工大学学报, 2009,29(11): 978-982.
- [124] 张志成, 林君, 石要武等.基于人工蜂群算法的波达方向和多普勒频率联合估计[J].吉林大学学报(工学版), 2013,43(4): 1104-1109.
- [125] 刘路, 王太勇.基于人工蜂群算法的支持向量机优化[J].天津大学学报, 2011,44(9): 803-809.
- [126] 毕晓君, 王艳娇.加速收敛的人工蜂群算法[J].系统工程与电子技术, 2011,33(12): 2755-2761.
- [127] 陈久梅, 曾波.两级定位-路径问题的路径重连变邻域搜索人工蜂群算法[J].计算机集成制造系统, 2014,20(5): 1228-1236.
- [128] 柳寅, 马良.模糊人工蜂群算法的旅行商问题求解[J].计算机应用研究, 2013,30(9): 2694-2696.
- [129] 王翔, 郑建国.求解约束优化问题的多成员人工蜂群算法[J].西安交通大学学报, 2012,46(2): 38-44.
- [130] 高卫峰, 刘三阳, 黄玲玲.受启发的人工蜂群算法在全局优化问题中的应用[J].电子学报, 2012,40(12): 2396-2403.
- [131] 林小军, 叶东毅.一种带规范知识引导的改进人工蜂群算法[J].模式识别与人工智能, 2013,26(3): 307-314.
- [132] 高明松, 李素明, 周卫东.应用人工蜂群算法辨识潜器参数[J].哈尔滨工程大学学报,

- 2013,34(8): 1023-1027.
- [133]毕晓君, 王艳娇. 用于多峰函数优化的小生境人工蜂群算法[J]. 系统工程与电子技术, 2011,33(11): 2564-2568.
- [134]张培文, 潘全科, 李俊青等. 有限缓流区流水车间调度的混合人工蜂群算法[J]. 计算机集成制造系统, 2013,19(10): 2510-2520.
- [135]毕晓君, 王艳娇. 约束多目标人工蜂群算法[J]. 吉林大学学报(工学版), 2013,43(2): 397-403.
- [136]徐晓飞, 刘志中, 王忠杰, 闵寻优, 刘睿霖, 王海芳. S-ABC——面向服务领域的人工蜂群算法范型[J]. 计算机学报, 2015,38(63): 1-18.
- [137]李田来, 刘方爱, 王新华. 基于分治策略的改进人工蜂群算法[J]. 控制与决策, 2015,30(2): 316-320.
- [138]丁政豪, 徐浩杰, 刘济科, 吕中荣. 基于混沌人工蜂群算法的结构损伤识别[J]. 中山大学学报(自然科学版), 2015,54(5): 39-42.
- [139]周新宇, 吴志健, 王明文. 基于正交实验设计的人工蜂群算法[J]. 软件学报, 2015,26(9): 2167-2190.
- [140]宁爱平, 张雪英. 人工蜂群算法的收敛性分析[J]. 控制与决策, 2013, 28(10): 1554-1558.
- [141]张波, 张景肖. 应用随机过程[M]. 北京: 清华大学出版社, 2004.
- [142]Edward P C K. An Introduction to Stochastic Processes[M]. Belmont, Calif.London: Duxbury Press, 1997.
- [143]Leung Y W, Wang Y P. An orthogonal genetic algorithm with quantization for global numerical optimization[J]. IEEE Transactions of Evolutionary Computation, 2001, 5(1):41-53.
- [144]McGill R, Tukey J, Larsen W. Variations of boxplots[J]. American Statistician, 1978, 32(1):12-16.
- [145]Kong X, et al. . Hybrid Artificial Bee Colony Algorithm for Global Numerical Optimization[J]. Journal of Computational Information System, 2012, 8(6):2367-2374.
- [146]Liang J J, Qin A K, Suganthan P N, Baskar S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions[J]. IEEE Transactions of Evolutionary Computation, 2006, 10(3):281-295.
- [147]Yao X, Liu Y. Fast evolution strategies[J]. Control and Cybernetics, 1997, 26(3):467-496.
- [148]Hansen N, Ostermeier A. Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation[C]. In Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC' 96), Nagoya, 1996: 312-317.
- [149]Hedar A, Fukushima M. Evolution strategies learned with automatic termination criteria[C]. In Proceedings of the Conference on Soft Computing and Intelligent Systems and the International Symposium on Advanced Intelligent Systems, Tokyo, Japan, 2006: 1-9.

- [150] Li G, Niu P, Xiao X. Development and investigation of efficient artificial bee colony algorithm for numerical function optimization[J]. *Applied Soft Computing*, 2012, 12(1):320-332.
- [151] Xu Yunfeng, Fan Ping, Yuan Ling. A Simple and Efficient Artificial Bee Colony Algorithm[J]. *Mathematical Problems in Engineering*, 2013, vol. 2013, Article ID 526315, 9 pages.
- [152] Nocedal J, Wright SJ. *Numerical Optimization*[M]. Springer, 2006.
- [153] Lawson C L, Hanson R J. *Solving least squares problems*[M]. Philadelphia: SIAM Society for Industrial and Applied Mathematics, 1987: 9-256.
- [154] Bjorck Ake. *Numerical methods for least squares problems*[M]. Philadelphia: SIAM, 1996.
- [155] Yong Longquan. A feasible interior point algorithm for a class of nonnegative least squares problems[C]. In *Proceedings of the IEEE International Conference on Future Computer and Communication*, Wnhan, 2009.IEEE, 2009:157-159.
- [156] S.Rahnamayan, H.R.Tizhoosh, M.M.A.Salama. Opposition-based differential evolution. *IEEE Transaction on Evolutionary Computation*, 2008, 12(1):64-79.
- [157] Koziel S, Michalewicz Z. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization[J]. *Evolutionary Computation*, 1999, 7(1): 19-44.
- [158] Michalewicz Z, Janikow CZ. Handling constraints in genetic algorithms[C]. Belew RK, Booker LB. *Proceedings of the Fourth International Conference on Genetic Algorithms (ICGA-91)*. San Mateo, California: University of California, San Diego, Morgan Kaufmann Publishers, 1991: 151-157.
- [159] Michalewicz Z, Schoenauer M. Evolutionary algorithms for constrained parameter optimization problems[J]. *Evolutionary Computation*, 1996, 4(1): 1-32.
- [160] Bean JC, Hadj-Alouane AB. A dual genetic algorithm for bounded integer programs[R]. Technical Report TR 92-53, Department of Industrial and Operations Engineering, The University of Michigan, 1992.
- [161] Homaifar A, Lai SHY, Qi X. Constrained optimization via genetic algorithms[J]. *Simulation*, 1994, 62(4): 242-254.
- [162] Joines J, Houck C. On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with Gas[C]. Fogel D. *Proceedings of the first IEEE Conference on Evolutionary Computation*. Florida: IEEE Press, Orlando, 1994: 579-584.
- [163] Michalewicz Z, Attia NF. Evolutionary optimization of constrained problems[C]. *Proceedings of the 3rd Annual Conference on Evolutionary Programming*. World Scientific, 1994: 98-108.
- [164] Richardson JT, Palmer MR, Liepins G, Hilliard M. Some guidelines for genetic algorithms with penalty functions[C]. Schaffer JD. *Proceedings of the Third International Conference on Genetic Algorithms (ICGA-89)*. California: George Mason University, Morgan Kaufmann Publishers, San Mateo, 1989: 191-197.

- [165] Schoenauer M, Xanthakis S. Constrained GA optimization[C]. Forrest S. Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA- 93). California: University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers, San Mateo, 1993: 573-580.
- [166] Powell D, Skolnick MM. Using genetic algorithms in engineering design optimization with non-linear constraints[C]. Forrest S. Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA-93). California: University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers, San Mateo, 1993: 424-431.
- [167] Michalewicz Z, Nazhiyath G. Genocop III:a co-evolutionary algorithm for numerical optimization with nonlinear constraints[C]. Fogel DB. Proceedings of the Second IEEE International Conference on Evolutionary Computation. NJ: IEEE Press, Piscataway, 1995: 647-651.
- [168] Deb K. An efficient constraint handling method for genetic algorithms[J]. Computer Methods in Applied Mechanics and Engineering, 2000, 186(2-4): 311-338.
- [169] Paredis J. Co-evolutionary constraint satisfaction[C], Proceedings of the 3rd Conference on Parallel Problem Solving from Nature, Springer-Verlag, New York, 1994: 46-55.
- [170] Parmee IC, Purchase G. The development of a directed genetic search technique for heavily constrained design spaces[C].In: Parmee IC. Adaptive Computing in Engineering Design and Control-'94. UK: University of Plymouth, Plymouth, 1994: 97-102.
- [171] Myung H, Kim JH, Fogel DB. Preliminary investigation into a two-stage method of evolutionary optimization on constrained problems[C].In: McDonnell JR, Reynolds RG, Fogel DB. Proceedings of the Fourth Annual Conference on Evolutionary Programming. MA: MIT Press, Cambridge, 1995: 449-463.
- [172] Reynolds RG, Michalewicz Z, Cavaretta M. Using cultural algorithms for constraint handling inGENOCOP [C]. In: McDonnell JR, Reynolds RG, Fogel DB. Proceedings of the Fourth Annual Conference on Evolutionary Programming. MA: MIT Press, Cambridge, 1995: 298-305.
- [173] Karaboga D, Akay B. A modified artificial bee colony (ABC) algorithm for constrained optimization problems[J]. Applied Soft Computing, 2011, 11(3): 3021-2031.
- [174] Wang Y, Cai Z, Zhou Y, Zeng W. An adaptive tradeoff model for constrained evolutionary optimization[J]. IEEE Transactions On Evolutionary Computation, 2008, 12(1): 80-92.
- [175] Hamida SB, Schoenauer M. ASCHEA: new results using adaptive segregational constraint handling[C].In: Proceedings of the Congress on Evolutionary Computation 2002 (CEC'2002), vol. 1, IEEE Service Center, Piscataway, NJ, May, 2002: 884-889.
- [176] Runarsson TP, Yao X. Search biases in constrained evolutionary optimization[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 2005, 35

(2): 233-243.

- [177] Runarsson TP, Yao X. Stochastic ranking for constrained evolutionary optimization[J]. IEEE Transactions on Evolutionary Computation, 2000, 4 (3): 284-294.
- [178] Mezura-Montes E, Coello CA. A simple multimembered evolution strategy to solve constrained optimization problems[R]. México: Technical Report EVOCINV-04-2003, Evolutionary Computation Group at CINVESTAV, Sección de Computación, Departamento de Ingeniería Eléctrica, CINVESTAV-IPN, México D.F., México, 2003.
- [179] Peng F, Tang K, Chen G, Yao X. Population-based algorithm portfolio for numerical optimization[J]. IEEE Transactions On Evolutionary Computation, 2010, 14(5): 782-800.
- [180] Michalewicz Z, Schoenauer M. Evolutionary algorithms for constrained parameter optimization problems[J]. Evolutionary Computation, 1995, 4(1): 1-32.

致谢

时光荏苒，如白驹过隙。蓦然回首，有太多感激。

首先我要特别致谢我最敬爱的导师刘三阳教授。从恩师身上，我可学和学到的东西将受益终生。在导师刘三阳教授的悉心指导下，顺利完成了博士阶段的学习。本论文是在刘老师的精心指导和亲切关怀下完成的，论文的每一步工作都倾注着导师的心血。从论文的选题、论证到论文的撰写各阶段，刘老师都提出了许多宝贵的意见和建议，使得论文的研究工作得以顺利开展。刘老师以其渊博的学识，敏锐的洞察力，严谨求实的治学态度和勤奋创新的钻研精神使我终生受益，并将对我今后的学习、生活和工作产生深远的影响。刘老师不仅为我提供了宽松的学术氛围，良好的科研环境，更培养了我独立从事科研的能力。在博士学习期间，刘老师从生活、学习上都给予了我特别的关怀和帮助，我每天都亲身感受着刘老师的人格魅力和治学态度，在他身上我时刻可以学习做人和做学问的道理。值此论文完成之际，谨向辛勤培育和关心爱护我的导师刘三阳教授致以衷心的感谢和深深的敬意！

另外，在成文过程中，还得到很多领导、老师、朋友的帮助和支持。

感谢数学与统计学院所有领导和老师，是他们的悉心栽培和辛勤劳动，我的研究生活才能顺利完成。

感谢所有同学在生活上的照顾和学业上的勉励，特别感谢雍龙泉、黄元元、王峰等同学给予了我无私的帮助和鼓励，以及众同门师兄师姐师弟师妹们。和他们在一起的生活、学习使我获得了很多在其他方面得不到的知识和经验，借此机会向他们致谢。

感谢我的家人，是他们在我漫长的学习生活中给予了我极大的关心、支持和理解，并尽其所能为我创造最好的条件、使我能够顺利完成学业。

最后，向百忙之中抽出时间审稿和参加论文答辩的老师们致以深深的谢意！

师恩亲情重如山，学业努力无止境。我将用百倍的努力和更大的进步来回报所有给予我帮助的老师、朋友和家人。真诚致谢！

孔翔宇
2016年4月

作者简介

1. 基本情况

孔翔宇，男，吉林榆树人，1982年10月出生，西安电子科技大学数学与统计学院应用数学专业2011级博士研究生。

2. 教育背景

2000.09~2004.06 湖南大学，本科，专业：数学与应用数学

2007.08~2010.03 西安电子科技大学，硕士研究生，专业：应用数学

2011.08~西安电子科技大学，博士研究生，专业：应用数学

3. 攻读博士学位期间的研究成果

3.1 发表的学术论文

- [1] Xiangyu Kong, Sanyang Liu, Zhen Wang et al. Hybrid Artificial Bee Colony Algorithm for Global Numerical Optimization[J]. Journal of Computational Information Systems,2012,8(6):2367-2374. (EI: 20122215069904)
- [2] Xiangyu Kong, Sanyang Liu, Zhen Wang. An improved artificial bee colony algorithm and its application [J]. International Journal of Signal Processing,Image Processing and Pattern Recognition,2013,6 (6): 259-274.(EI: 20140517244376)
- [3] Xiangyu Kong, Sanyang Liu, Zhen Wang. An Effective Hybrid Artificial Bee Colony Algorithm for Nonnegative Linear Least Squares Problems[J].Journal of Engineering Science and Technology Review, 2014, 7(3), 96-107. (EI: 20143418090308)
- [4] 孔翔宇, 刘三阳, 王贞. 人工蜂群算法的几乎必然强收敛性: 鞍方法[J]. 计算机科学, 2015, 42(9): 246-248, 277.
- [5] Xiangyu Kong, Sanyang Liu, Zhen Wang et al. An Effective Chaotic Artificial Bee Colony Approach to Global Optimization and Its Application[J]. Journal of Computational and Theoretical Nanoscience,Accept.(SCI, EI)
- [6] Xiangyu Kong, Sanyang Liu, Zhen Wang et al. Algorithm Portfolio Based on Evolutionary Strategy and Artificial Bee Colony Algorithm for Constrained

Optimization[J]. Applied and Computational Mathematics,Accept.(SCI)

3.2 参与科研项目及获奖

- [1] 国家自然科学基金项目, 基于 Contoutlets 的路面病害自适应检查与识别,
2015.01-2015.12, 结题, 参与人.