

广东工业大学

硕士学位论文

基于极坐标方向寻优的进化策略及其应用

姓名：边增远

申请学位级别：硕士

专业：计算机应用技术

指导教师：曾碧

20060501

摘 要

进化策略是借鉴生物进化的思想，在现代遗传学的启发下，发展起来的一种启发式随机搜索优化方法。进化策略作为一个新的交叉学科，目前已发展成一种自组织、自适应的综合技术，广泛用于计算机科学、工程技术、管理科学和社会科学等领域，尤其在信号处理领域受到高度重视。

目前，由于进化策略产生下一子代的方法是通过变异方式实现的，对父代的继承性较差，因此目前进化策略的应用主要是配合遗传算法或其它智能算法使用，单独使用进化策略解决问题的例子较少。针对于此，本文提出了一种新的进化策略—基于极坐标方向寻优的进化策略算法，该算法能够有效地继承父代的优点，能够得到更快、更优的收敛结果。并且本文将这种改进的进化策略思想应用到汽车导航系统中，解决路径寻优问题。本文的主要研究内容包括：

1. 对传统进化策略进行分析，剖析其收敛过程，掌握制约收敛速度和收敛全局最优解的基本要素，通过对传统进化策略的改进，进而得到一种更快、更好的进化策略寻优算法。
2. 提出“基于极坐标方向寻优的进化策略”算法，论述其理论基础、实现方法，并与传统进化策略进行实例仿真对比。
3. 运用数学理论知识证明基于极坐标方向寻优的进化策略算法的收敛性。
4. 通过实例说明该收敛算法比传统进化策略具有更好的收敛速度和更加稳定的收敛特征，能够有效的收敛到全局最优点。
5. 将改进后的进化策略应用到汽车导航系统中，解决汽车行驶路径寻优问题。

本课题是以传统进化策略为基础，所做的探索性研究尝试提供一种新的进化策略方法，改进传统进化策略。本文证明了改进进化策略的收敛性，并且通过多个实例验证了改进后的进化策略，证明其具有更快的收敛速度和更好的稳定性。

本课题为国家自然科学基金（04009464）和广东省自然科学基金项目（05001801）。

关键词：进化策略；变异；选择；依概率收敛；路径寻优

Abstract

The evolution strategy profits from the biological evolution theory, and it is a heuristic stochastic search optimization method in the inspiration of the modern genetics. As a new interdisciplinary study, the evolution strategy has developed as an self-organized, auto-adapted comprehensive technology, which is widely used in the field of computer science, project technology, management science, social sciences and so on, particularly in the signal processing.

At present, the evolution strategy neglects the characteristic of father generation, so the evolution strategy application is not independent; mostly, it is used to coordinate with the genetic algorithms or other intelligent algorithm. Accordingly, we proposed a new kind of evolution strategy algorithm based on the selected direction by the polar coordinates. This algorithm can effectively inherit the meritorious character of father generation, which can obtain a result quickly and precisely. Finally, we applied the improvement evolution strategy in the automobile guidance systems. This article main research content includes:

1. By analyzing the traditional evolution strategy, we grasp the basic essential factor of restricting convergence rate and the overall situation optimal solution. At last we obtain a quicker and the better evolution strategy algorithm.
2. We propose the evolution strategy algorithm based on the selected direction by the polar coordinates. We elaborate its rationale and its implementation method, and contrasts with the traditional evolution strategy by the example simulation.
3. The evolution strategy algorithm based on the selected direction by the polar coordinates is astringent, which is proved by mathematics theory knowledge.
4. The improvement evolution strategy has a better character compared to the traditional evolution strategy.
5. The improvement evolution strategy is applied in the automobile guidance systems.

This paper is supported by National Natural Science Foundation of China (60272089) and Guangdong Provincial Natural Science Foundation of China (980406).

Key words: Evolve strategy; the polar coordinates; colony; direction

第一章 绪论

二十世纪六十年代以来,进化计算(Evolutionary Computation, EC)作为一个新兴的交叉学科,从无到有,由弱到强,现在已经成为人工智能领域中的一个重要分支[1]。在工程技术、社会经济以及科学计算领域中存在着大量的复杂性计算问题,由于这些问题表现出的高度非线性、不可导甚至不连续,复杂度非常高,传统的优化算法在这些问题面前显得无能为力,进化计算的出现为解决这类问题开辟了一条崭新的途径。在过去的四十几年中,以遗传算法(Genetic Algorithms, GA)、进化策略(Evolution Strategy, ES)、进化规划(Evolutionary Programming, EP)为代表的启发式随机搜索算法,在达尔文(Darwin)生物进化思想和后达尔文主义(Neo Darwinian)进化思想的启迪下,在遗传学特别是现代分子生物学的启发下,进化计算在理论和应用两个方面都取得了长足的发展[2]。

进化计算的快速发展是与其广阔的应用前景紧密相关的。随着社会的不断向前发展,现实生活中需要解决的问题越来越多,其复杂程度越来越高,对算法的求解要求也就随之提高了。为了提高算法的性能,必须不断加强对进化计算工作机理的认识,并在此基础上完善算法机制。本论文的基本内容就是针对进化策略的理论问题进行研究,并指导算法设计,更加有效解决实际工程中出现的问题[3,4]。

为了更好的了解进化策略,下面首先对进化算法给予简单介绍。

1.1 进化算法的发展历程

进化计算的起源可以追溯到二十世纪五十年代中期,Friedberg 首先将进化思想与计算机相结合[5,8],用于寻找一个可以计算给定函数的输入输出问题的程序,这一工作可以认为是遗传规划的起点。据目前可查文献记载,Bremermann 是首先将进化思想用于计算机解决数值优化问题的学者[9]。

他们的开拓性工作二十世纪六十年代中期遗传算法、进化策略和进化规划发展的起点,二十世纪八十年代末期的遗传规划也间接地受到了影响。虽然遗传算法、进化策略和进化规划这三种进化计算中的主流算法是在同时期发展起来的,并具有相同的生物学背景,面对数值优化问题也极其相似,但在他们近三十年的早期发展中,彼此间是独立发展的。直到1990年10月份,进化计算领域在多特蒙德召开的一次会议才将不同研究分支的学者聚集在一起,使不同的分支相互认知、融合,从而形成进化计算。

1.2 进化策略的发展历程

Rechenberg、Schwefel 和 Bienert 是进化策略公认的发明者[5]。1964 年,当时他们还在柏林技术大学上学,在设计一种能够评测在风洞中可弯曲的三维物体最小变形的设备的过程中发明了该算法。与其他研究工作一样,开始阶段,他们希望通过传统的数值算法来解决这一问题,结果是失败的,传统的数值算法无法跳出局部最优解。Rechenberg 首先想到了将进化思想应用到算法的设计中,它将进化论中的生物选择、重组、变异应用到参数优化中[10],可以认为进化策略的思想已经形成,虽然这只是进化策略的一个初始形态,但它涵盖了进化策略的基本思想。Rechenberg 的基本思想如下:首先随机产生一个离散的初始个体,然后将一个满足二项分布的变异作用在初始个体上,这样得到一个新的个体,从这两个个体中选择适应值较大的个体作为下一代的父个体,这就完成一次循环。这种算法应该称为 $(1+1)$ 进化策略。

Schwefel 首先将 Rechenberg 的算法付诸于实施[11],Schwefel 的实验非常成功,算法战胜了局部最优值的陷阱,获得了全局最优解,这一成功对与进化规划而言同样具有里程碑式的意义。Bienert 的贡献在于将进化规划思想应用到实际具有工业意义的具体问题上[12],该问题是传统算法所不能解决的问题,Bienert 的硕士论文成功的解决了这一问题,这一工作结合了进化规划的理论与实践,意义重大。

随后 Rechenberg[13]进一步改进了进化规划,在解决 n 维凸函数的优化中,给出了算法的收敛性分析,并在此基础上给出了著名的“ $1/5$ 成功原则”,即通过外在的改变二项分布的方差的经验值。Rechenberg 又给出了多群体下的进化策略算法,即 $(\mu + 1)$ 进化策略,群体内个体间的重组操作也被使用。Rechenberg 的这项工作对于进化策略的进一步发展起到了决定性的作用,群体概念和重组算子的引入从理论上完善了进化策略,为其在实际应用中取得成功奠定了理论基础。但是这一算法由于变异算子缺乏自适应机制而显得先天不足,并未得到广泛的推广和应用。

Schwefel[14]在 $(\mu + 1)$ 进化策略的基础上迈出了重要的一步,将之推广到 $(\mu + \lambda)$ 进化策略和 (μ, λ) 进化策略,这两种形式的进化策略成为后来广泛使用的两种形式。进化规划中使用的自适应机制在进化策略中也被使用。进化策略的研究中心在德国,主要以 Schwefel 在多特蒙德领导的研究团队和 Rechenberg 在柏林领导的研究团队为主。

随着并行思想的深入,进化策略在二十世纪九十年代引入了并行概念并取得了一定的成功。同期正是进化计算各个分支相互融合的时期,进化策略研究团体发挥了重要的作用。

在进化策略的理论研究方面, Hans-Georg Beyer 做出了重要的贡献。Beyer 首先建立了进化策略的数学模型, 并以微分几何为工具研究了非球面适应值空间的性质, 研究了(μ , λ)等多种进化策略的收敛性质和算法的动力学性质, 为进化策略的进一步发展打下了基础。

国内对进化策略的研究已经起步, 并且也取得了很大的成绩, 在国内、国际会议上发表了相当多的论著, 成功的解决了单峰函数最优化问题[15, 16]、智能试题库系统中的抽题算法[17]、用柯西分布来改变进化策略的变异算子提高收敛速度[18]以及对多目标进化算法的研究[19][20]。

1.3 进化策略的主要特点

进化计算是一种基于自然选择机制下的全局性随机搜索算法。它从随机产生的初始群体开始, 经过进化算子的不断作用, 在迭代过程中逐步改进当前群体中个体的适应值, 直到算法得到问题的最优解或满意解为止

进化策略具有以下主要特点:

- (1) 进化策略是首先采用自适应机制的进化算法。
- (2) 进化策略的搜索初始点最初使用的是单一的初始点, 这种方式的进化策略已基本不再使用; 现在普遍使用的进化策略算法也是采用初始群体作为搜索的初始点。
- (3) 进化策略构造简单、易于实现, 可以采用嵌套方式来应用这一算法。
- (4) 进化策略具有典型的动力学特征。特别是在 ONE-MAX(1+1)问题上, 已经有了肯定的结论。

1.4 进化策略的理论研究

进化策略在运行机理方面的研究进行的很缓慢。Rechenberg 在研究用进化策略解决多模态适应值曲面问题时, 给出了如下公式[21]:

$$\forall \alpha, \beta \quad \|\hat{y} - y_\alpha\| \leq \|\hat{y} - y_\beta\| \Leftrightarrow F(y_\alpha) \leq F(y_\beta) \quad (1-1)$$

其中 \hat{y} 被称为单调最小吸引子, $F(\cdot)$ 表示适应值。

Rechenberg 认为满足公式 (1-1) 的适应值曲面, 在进化策略的作用下可以搜索到全

局最优解。虽然公式(1-1)的正确性在算法的实际应用中得到了验证,但在理论上这一问题尚未解决。

进化策略与遗传算法中的模式定理相对应的是进化过程原则(Evolutionary Progress Principle, EPP),即状态空间个体适应值的增加减去个体适应值的损失构成了进化的进展。这一看似简单的结论是进化策略中最基本的假设,它是进化策略理论分析的基础。

对于进化策略中重组算子的作用,Beyer借用生物学中的假设提出了遗传修复假说(Genetic Repair Hypothesis, GR) [21],这一假说与遗传算法中的建筑块假说具有类似的作用。重组算子具有将破坏掉的基因重组的功能,同时可以产生新的基因组合,从而提高进化策略的运行效率。

1998年,Beyer从整个进化计算的角度分析了算法中重组算子和变异算子对群体产生新个体所起的作用。群体中的变异是通过全局性的重组操作将变异的结果在整个群体内扩散,从而生成新的稳定的群体,这一解释被称之为变异通过重组生成物种原则(Mutation-induced speciation by recombination, MISR) [21]。这一原则可表示为:

Dominant Recombination & Physical Mutations \Rightarrow "Species".

Beyer随后给出了MISR原则的度量公式,并研究了非限定搜索空间中MISR的作用,进一步对二进制下遗传算法的MISR效应进行了研究。Beyer还针对进化策略的收敛性进行了研究,主要采用解析方法研究算法在不同类型搜索区域的渐进性质,取得了一些结果。这种收敛性的结果比较适合进化规划,如果将之应用到进化策略和遗传算法则显得比较牵强。

1.5 本文研究背景及主要内容

进化策略作为一个年轻的优化方法,虽然具有较为广泛的应用前景,但是理论研究相对薄弱,加强进化计算的理论研究是广大研究人员的共识。

1.5.1 本文研究背景

本课题由国家自然科学基金(NO. 04009464)和广东省自然科学基金(NO. 05001801)支持。

1.5.2 本文的主要研究内容

本文的主要研究内容包括：

1. 通过对传统进化策略进行分析，获得进化策略的较快和较好的寻优算法，掌握制约收敛速度和收敛全局最优解的基本要素。
2. 根据传统进化策略的收敛特征，将理论研究转移到极坐标系中，利用有利的父代方向改进传统进化策略，设计基于极坐标方向寻优的进化策略算法，并且论述基本实现过程和方法，以及其比传统进化策略优越性的表现。
3. 运用概率方面知识证明基于极坐标方向寻优的进化策略算法的收敛性。
4. 通过实例说明基于极坐标方向寻优的进化策略算法比传统进化策略具有更好的收敛速度和更加稳定的收敛特征，能够有效地收敛到全局最优点。
5. 利用改进的进化策略来解决的汽车导航路径寻优问题。

1.6 本课题的创新之处

- 1) 给出了将进化策略从直角坐标系转移到极坐标系中研究的思想和方法。
- 2) 提出在进化策略中利用父代提供的方向特性作为指导产生新的子代的优化寻优方法。
- 3) 将上述两个方面有机结合，得出较高的收敛速度和较理想的收敛结果。

第二章 进化策略基本原理及算法描述

科学研究总是建立在前人的实验结果、理论和假设之上的,进化策略(Evolution Strategy, ES)也不例外。本章首先从进化计算的生态学背景上进行研究。然后对进化计算理论的研究给出算法描述,

2.1 进化策略的生物学背景

进化策略作为一门交叉性的学科,生物学和遗传学的思想、方法为其提供了解决问题的理论依据,所以,对进化计算的研究,首先应从进化计算的生物学背景开始。

通过吸收布丰(Buffon)和拉马克(J.Lamarck)等人的进化思想,在大量研究家养和自然状态下的动、植物及地质资料的基础上,达尔文(C.Darwin)的进化学说得以形成。该学说主要包括以下思想[6]:

通过对家养状态下动植物的变异和自然状态下的变异的研究,达尔文认为:一切生物都会发生变异,一些变异可以遗传给后代,另一些变异则不能。变异大多是可以遗传的,不能遗传的变异是例外。在自然状态下,显著的偶然变异是少见的,即使出现也会因杂交而消失。自然选择即适者生存,生存斗争和自然选择紧密相关,任何生物产生的生殖细胞或后代的数目要远远多于可能存活的个体数目,自然选择专门保存和积累那些在生命的各个时期,在各种有机和无机生活条件下,都对物种生存有利的变异。在一个种群内,个体间的差异性越大,则在适应不同环境方面越有利,繁殖的个体更多,分布的范围更广,小生境会促进性状分歧和新物种的形成。

生物群体的进化机制表现为三种形式:

- 1、自然选择:自然选择是由于生物体基因型的变化影响了其显型,生物体的适应性随之发生变化。即物种的随机变异的非随机淘汰和保留。生物体的种群内存在发生突变的可能和大量繁殖补充种群内淘汰的个体是自然选择成立的保障。自然选择是建立在随机性基础上的非随机过程。可以分为正常化选择、平衡选择、稳定化选择、歧异化选择、集团选择、定向选择、性选择等等。

- 2、交叉:两个染色单体在二价染色体中相应断裂的部位,断开的部分两两交叉重新连接,产生两条新的染色单体。这样,生物体通过杂交使来自不同父代的染色体上的基因重组,产生携带新的染色体的个体。生物体中的新个体大多是通过这种方式来产生的。

3、变异：染色体上基因的突变是变异产生的必要条件，供自然选择作用的变异是经过生物遗传系统本身加工和组装的，是经过挑选、摒弃、编辑过的贮存的变异。生物体的变异过程是一种突发的、不可逆的过程，是一种小概率事件。在生物进化中具有不可替代的作用。

生物进化是一个复杂的适应过程问题，是偶然性和必然性相互作用的结果。生物进化是在一定条件下进行的：首先存在一定规模的种群，该种群内的个体间的适应度存在差异，群体保持一定的多样性；生物体自我繁殖的能力，即有性生物的生殖和无性生物的克隆。种群内的个体承受着不同性质和不同强度的选择压力，他们决定了种群进化的性质。只有在大量遗传变异的基础上才有可能产生大量的遗传重组，通过大量的遗传重组筛选才有可能获得最佳的基因组合，通过自然选择的作用，一方面保留一些变异；另一方面则淘汰一些变异。

任何物种的进化都是一个复杂系统自我完善的过程，是一个优化的过程。因此在计算机仿生方面，具有直接的借鉴意义。构成了进化策略的基本思想和算法形式，进化算子是生物进化的直接模仿，所以说，进化论是进化计算的源泉之一。

2.2 进化策略的算法描述

进化策略最初并不仅使用连续参变量进行优化，离散参变量也被使用，例如在 1970 年，Klockgether 和 Schwefel 便采用离散的实变量作为变异参量应用进化策略进行优化问题的求解。现在普遍采用的在连续的欧式空间进行优化的思想，这主要归结于 1973 年 Rechenberg 在文献[13]中成功地应用了连续参变量的变异。

设 $f: R^n \rightarrow R$ 是一个求最大值的优化问题，即

$$\max_{x \in R^n} f(x) \quad (2-1)$$

其中， $x \in X$ 是一个 n 维的目标参变量， $x = (x_1, \dots, x_n)^T$ 。

在进化策略中的变异方法采取如下方式：

$$X_{t+1} = x_t + \sigma_t Z_t \quad (2-2)$$

其中， $t \in N$ ， x_0 是随机选取的初始值， σ_t 是大于 0 的常数，称为调整系数，用来改变变异的速度。 Z_t 是服从于正态分布 $N(0,1)$ 的相互独立的随机向量。

2.2.1 选择操作的描述

在进化策略中,为了保证能够收敛到全局最优解,通常在进化策略中普遍使用的是 (μ, λ) -ES, $\lambda \geq \mu$ 模型和 $(\mu + \lambda)$ -ES 模型。对这两种不同的模型,所采用的选择方式是不同的。所谓 (μ, λ) 模型就是从 μ 个父代中产生 λ 个后代,然后从 λ 个后代中选择适应度高的 μ 个个体组成下一个父代,这也就是要求 $\lambda \geq \mu$ 的原因。对 $(\mu + \lambda)$ 模型而言,则是从 μ 个父代中产生 λ 个后代,然后从 $\mu + \lambda$ 个后代中选择适应值高的 μ 个个体组成下一个父代。这两种选择方式各有优势,对于实参数优化问题,一般采用前一个模型,而对于组合优化问题一般采用后一个模型。不论哪一种方式,进化策略的选择操作都是确定性的。将这两种模型简记为 $(\mu \ \lambda)$ 。

$(\mu + \lambda)$ -ES 较好的继承了父代的优良特性,收敛性好,但容易陷入局部最优解。相比之下 (μ, λ) -ES 则易于跳出局部最优解,但是由于算法中放弃了对上一代的继承,因此收敛速度很慢。

自然界对生物的选择是通过严酷的自然条件实现的,对进化策略算法的最优值选择则是通过阈值函数实现的,例如求二元函数 $f(x,y)$ 的最小值,利用进化策略,其阈值函数就是 $f(x,y)$, 如果每一个定义域中的点都对应一个函数值 $Z_0=f(x_0,y_0)$, $\dots\dots$, $Z_k=f(x_k,y_k)$, $Z_{k+1}=f(x_{k+1},y_{k+1})\dots\dots$, $Z_n=f(x_n,y_n)$, 当 $Z_k < Z_{k+1}$ 时,说明 Z_k 的适应性强于 Z_{k+1} , 阈值函数有的时候是不明确的,为了应用进化策略解决问题,需要将其转化为明确的阈值函数,例如下面的例子:

用进化策略的方式来构造幻方时,因为幻方问题是一个由数字 1~9 排列的 3×3 方阵。排列的特点是该方阵的每行、每列以及两对角线上的数字之和恒为 15。要首先建立质量函数以比较两幻方的好坏。对此处的 3×3 幻方来说(图 2-1),质量函数 Q 可以表示为

$$Q=(n_1+n_2+n_3-15)^2+(n_4+n_5+n_6-15)^2+(n_7+n_8+n_9-15)^2+(n_1+n_4+n_7-15)^2+(n_2+n_5+n_8-15)^2+(n_3+n_6+n_9-15)^2+(n_1+n_5+n_9-15)^2+(n_3+n_5+n_7-15)^2 \quad (2-3)$$

上式中 Q 函数就是阈值函数, $n_1, n_2, \dots\dots n_9$ 是 3×3 方阵的点值。这样问题就转化为求质量函数 Q 的最小值问题。

n_1	n_2	n_3
n_4	n_5	n_6
n_7	n_8	n_9

图 2-1 幻方问题

Figure 2-1 Issue of magic square

除了为解决幻方问题时候可以采用进化策略，在电力系统无功优化上面已经成功解决了组合优化问题，在多传感器雷达辐射源目标识别，以及在控制混沌中都有着广泛的应用。进化策略的应用范围是相当广泛的，对于更为负责的非线性动力学问题都可以应用进化策略解决。只要能够找到求解问题所适合的阈值函数，就能够应用进化策略。

2.2.2 变异操作的描述

式(2-2)中“ $\sigma_i Z_i$ ”的选取体现了变异过程的特性。变异算子占有很重要的地位。一般情况下，进化算子的设计应遵循以下的基本原则：

- (1) 可行性，即在有限的时间内是可以实现的。
- (2) 可度量，每一步变异的长度是可以度量的。
- (3) 变异的分布满足最大熵原则。
- (4) 对称性要求，即变异的随机变量的数学期望值为零。

根据变异过程的特点，要求“ $\sigma_i Z_i$ ”是均值为 0，方差较小的随机向量。其中 σ_i 是调整参数， Z_i 的选取可以是高斯(Gaussian)分布函数，也可以是柯西(Cauchy)分布函数，其选取以上面的四点原则为依据[24]。

一维柯西密度函数集中在原点附近，其定义为：

$$f_t(x) = \frac{1}{\pi} \frac{t}{t^2 + x^2}, \quad -\infty < x < \infty \quad (2-4)$$

式中 $t > 0$ 为比例参数，相应的分布函数定义为：

$$F_t(x) = \frac{1}{2} + \frac{1}{\pi} \arctan\left(\frac{x}{t}\right) \quad (2-5)$$

一维高斯分布函数的概率密度函数为：

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} \exp[-(z - \mu)^2 / 2\sigma^2] \quad (2-6)$$

其中 μ 和 σ 分别是均值和方差

柯西密度函数类似于高斯密度函数，其差异主要表现在：柯西分布在垂直方向略小于

高斯分布，而柯西分布在水平方向上越接近水平轴，变得越缓慢，图 2 给出了二者的关系。

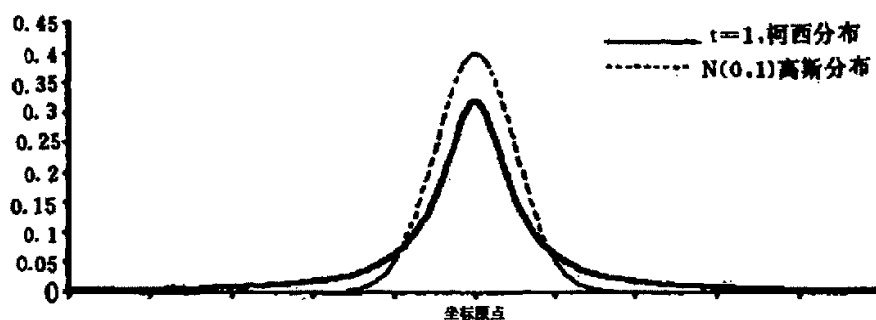


图 2-2 概率分布曲线图

Figure2-2. Graph of probability distributing

在二元进化策略里，了解了选择和变异的方法就可以利用进化策略来解决实际问题了。

2.2.3 二元进化策略算法的基本步骤

为了应用进化策略解决实际问题，对于二元进化策略通常采用如下步骤：

(1) 初始化

生成由两个个体构成的群体。一个作为父辈个体，一个作为后代个体。是由群体中个体的基因不同所决定的，这两个个体构成的群体被称为第一代群体。其中只有作为父辈的个体参与下一个步骤。

(2) 变异

第 g 代的父辈个体 $P^{(g)}$ 产生了其新后代个体 $D^{(g)}$ 。新生成的后代个体在基因上与父辈个体略有不同。其基因的差异具有随机性。

(3) 选择

由于第 g 代中父辈个体与其后代个体的基因不同，因此对环境的适应性也不同。其中只有具有较强适应性的个体有资格再次被选择作为第 $g+1$ 代的父辈个体 $P^{(g+1)}$ 。然后再回到 (2)。

可以看出，如上述所述周而复始的过程是一个简化了的进化过程。在其中做了如下假定：

- (1) 某种生物的进化是通过该生物个体构成的群体的进化完成的，群体中个体数目为常数。
- (2) 原则上，每个个体能无限期地存活，并能无限制地产生后代。

(3) 变异只发生在某个基因点上, 并且个体的基因组成唯一地决定了个体对环境的适应程度。

(4) 个体对环境的适应性函数并不随时间的变化而变化。

上面的简化和假定是出于利于解决工程问题的考虑。其进化的步骤与生物学上的进化有很大差异。变异与选择步骤也不是完全生物学意义上的过程。这样做的目的是为了向优化问题靠近, 使得优化问题能用这样一个框架得到解决。

将上面所述的进化步骤应用于优化问题, 以工程语言描述为:

(1) 初始化

生成一个优化问题的两个解。这两个解代表 n 维欧氏空间的两个点。由于这两个点位置的不同, 决定了其对应的适应性函数值的不同。其中适应性函数值较高的解将参与下面步骤的运算。这是第一次迭代。

(2) 变异

在第 g 次迭代中, 根据被选择的解 x_g , 产生出一个新解 y_g 。 x_g 与 y_g 只是在 n 维欧氏空间位置上略有不同。这两个解所对应的点之差是一个随机向量。

(3) x_g 与 y_g 对应于适应性函数 $F(x)$, 分别具有不同的函数值。其中只有具有较高适应性函数值的解被选择参与第 $g+1$ 次迭代中的运算。

作为一个算法, 应该具有终止条件。迭代的结束通常以算法不能再找到更优解为准则。而在实现时, 需要在每次迭代循环中指明是否要结束。由于在迭代过程中解的变异的步长和该解距极值解的距离越来越小, 因此实际中用到的一个约束规则是: 如果变异的步长已经非常小或每次迭代生成的解与原解相比差异很小, 这时可以认为算法不再可能得到性能更优的解, 算法满足结束条件。当然在迭代结果已经逼近极值时, 步长和解的变化都会较小。但是, 当搜索过程正在通过一个狭窄细长的山脊时, 步长和解的变化都会较小, 因此算法会在极值点很远处停止。所以, 这一准则不能保证我们得到极值点。

梯度方法是求解优化问题的常用方法。该方法要求目标函数可导。在这一条件下, 一个解使目标函数的导数为零是这个解为极值解的必要条件。因此, 在优化过程中, 不断计算 $\Delta F = F(x^{(k+1)}) - F(x^{(k)})$, 并测试该值是否为 0 或者接近 0 的数值。如果 ΔF 小于一个很小的正数, 则使算法终止。但是, 当迭代过程正在通过一个平坦的区域时, ΔF 很小, 而距离极值解可能还很远。这种情况下, 可能通过将步长加大, 而使得 ΔF 变得较大; 另外还可以多次的迭代过程计算 ΔF 的和, 如果 ΔF 的累计量仍然很小, 则令算法终止。因而收敛准则可以写成下面形式:

如果在迭代过程中下面条件满足

$$|F(x^{(g)})-F(x^{(g-k)})| \leq \varepsilon_e \quad (2-7)$$

或

$$[F(x^{(g)})-F(x^{(g-k)})]/\varepsilon_d \leq |F(x^{(g-k)})| \quad (2-8)$$

其中, k 为当前代的前 k 个父代, $k \geq 20n$, n 为正整数, $\varepsilon_e > 0$, $\varepsilon_d > 0$, 则算法结束。

准则中的参数 ε_e 和 ε_d 需要根据优化问题要求的计算精度而定。而 $k \geq 20n$ 是根据 1/5 成功率给出的。这样在极端情况下, 保证标准差的增加和减少至少为 $(0.85)^{\pm 20} \approx (3.9 \times 10^{-2})^{\pm 1}$ 。这样一来, 在迭代过程中就不用每次, 而是每 $20n$ 次测试收敛准则是否满足。

在算法实现中, 很多算法采用一个提前确定的迭代次数作为一个参数。如果算法的迭代次数超过这一参数, 则令算法结束。因此, 这成了算法其他收敛准则之外的附加准则。采用这一附加准则的优点是避免因原收敛准则不满足而执行太多迭代。这一附加准则在限制算法的运行时间时是行之有效的。可以利于计算机中计算运行时间的库函数每隔固定的迭代次数计算算法的时间花费, 一旦算法超过这个运行时间上限, 则强行令算法结束。

第三章 改进的进化策略

采用二元进化策略实现寻找最优解具有方法简单、可操作性强、运算的时间复杂度和空间复杂度低等特点。但是由于在算法中仅采用了变异的方法来产生下一代个体，因此其继承性较差，不能充分的利用父代的有利条件指引下一代的产生，这是与进化算法的思想相背离的。本章以此为入手点提出了一种全新的进化策略—基于极坐标方向寻优的进化策略。

通过对传统进化策略的分析，明确了进化策略的特点以及算法的一般步骤，其最优解的寻找过程可以看成在直角坐标系中向目标点的逼近过程，如图 1 所示。

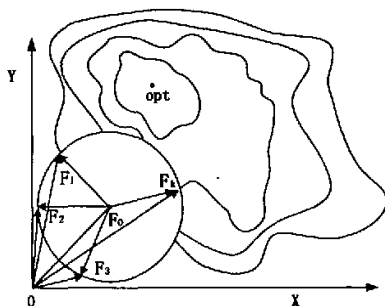


图 3-1 目标逼近过程

Figure 3-1 Producing of approach object

传统的进化策略是从初始点 F_0 出发，产生任意方向，任意步长的点 F_1, F_2, F_3, \dots ，直到产生距离目标点较近的点 F_k 为止，并将子代 F_k 替换父代 F_0 ，继续搜索下一个 F_k 。直到最终到达或者无限逼近 OPT 为止。传统的做法完全忽略了方向对整个寻优过程的积极意义，完全是盲目的搜索，这就浪费了很多的搜索时间。本章提出的极坐标方法正是充分考虑到对有利的方向加以利用，来提高收敛速度，并保证收敛的解为全局最优解。

程序初始化时产生了 μ 个父代，每个父代独立变异，独立进行选择，当所有个体都满足结束条件时寻优过程才结束，种群的数量越大其所得到的进化结果也就越好，这是以耗费更多的进化时间为代价的。为了便于说明，这里我们仅取其中的一个父代作为研究对象。

为了能够利用父代提供的方向，需要把图像所有的点都看成以原点为起点的向量形式，如图 3-1 中， $\overrightarrow{OF_0}$ 是程序初始化时产生的父代， $\overrightarrow{OF_1}$ 是利用 $\overrightarrow{OF_0}$ 进行随机进化的子代，如果 $\overrightarrow{OF_1}$ 的适应性强于 $\overrightarrow{OF_0}$ 那么说明 $\overrightarrow{OF_1}$ 的方向性更好，对子代的产生具有指引作用，为了兼顾 $\overrightarrow{OF_0}$ 的方向性，对于进化子代的选取其方向应该介于 $\overrightarrow{OF_0}$ 和 $\overrightarrow{OF_1}$ 之间的位置；模的大小仍然

采用加上变异算子（正态分布随机向量）的方法。显然解决这样的问题在极坐标系中将有更多的优势。

3.1 基于极坐标方法寻优的算法和步骤

经过反复试验，在极坐标系中如果定义域范围在整个 $[0, 2\pi]$ 范围内其坐标初始点的选择直接影响到进化速度和进化收敛全局最优点，通常在 $[0, \pi/2]$ 范围内效果最好，因此在转换极坐标之前首先将对函数定义域平移至第一象限。

考虑下列实值函数全局优化问题

$$(P) \quad \min f(x, y): x, y \in \Omega.$$

改进的进化策略算法是在进化策略的基础上,将进化策略的有利方向性添加到算法的变异过程之中,从而使算法的速度成倍提高。将求解问题(P)的改进的进化策略算法实现步骤描述如下:

1) 将优化问题标准化

标准 1: 平移坐标轴

在直角坐标系中将求解问题的定义域平移到坐标轴的正向区间，例如对一维问题则是将 x 的范围域平移至 x 正半轴，对二维问题则是将定义域平移到第一象限。例如存在定义域 $x \in [-a, b]$, $y \in [-c, d]$ ，其中 a, b, c, d 均大于 0，平移时需要作如下变化：

$$X = x + a$$

$$Y = y + c$$

这样定义域就变为 $X \in [0, b+a]$, $Y \in [0, d+c]$ ，变化到第一象限。

标准 2: 将直角坐标系转化为极坐标系

对于二元函数 $f(x, y)$ ，转化为极坐标函数为：

$$f(x, y) \xrightarrow{x=r*\cos(\alpha), y=r*\sin(\alpha)} f(r, \alpha) \quad (3-1)$$

其中 $\alpha \in [0, \pi/2]$, $r \in [a, b]$ 其中 a, b 可根据 x, y 的范围求得，函数 $f(r, \alpha)$ 就是校验函数。对于三元函数 $f(x, y, z)$ 转化为极坐标函数：

$$f(x, y, z) \xrightarrow{x=r*\cos(\beta)*\cos(\alpha), y=r*\cos(\beta)*\sin(\alpha), z=r*\sin(\beta)} f(r, \alpha, \beta) \quad (3-2)$$

其中 $\alpha, \beta \in [0, \pi/2]$, $r \in [a, b]$ 其中 a, b 可根据 x, y, z 的范围求得，函数 $f(r, \alpha, \beta)$ 就是校验函数。对于多元函数 $f(x, y, z, \dots)$ 转化为极坐标函数，只要适当添加角度变量即可实现，当然维数越多函数表达式就越复杂。

2) 变异

$$r(K+1)=r(K)+\sigma \times N(0,1) \quad (3-3)$$

其中 σ 称为调整系数, 可根据实际问题进行取值。 $N(0,1)$ 为随机正态分布函数。

如果连续出现两组较好的点 $(r(K-1), \alpha(K-1))$ 和 $(r(K), \alpha(K))$, 则对其方向角取均值, 即 $\alpha(K+1)=(\alpha(K-1)+\alpha(K))/2$

如果点 $(r(K-1), \alpha(K-1))$ 的适应性劣于 $(r(K), \alpha(K))$, 则角度 α 的选取采用传统进化策略的变异方法, 即 $\alpha(K+1)=\alpha(K)+\sigma \times N(0,1)$

3) 进化算法结束条件

当父代与子代的差异小到一定程度时, 判断进化过程结束。即当 $|f(\gamma_1, \alpha_1, \beta_1) - f(\gamma_0, \alpha_0, \beta_0)| \leq \zeta$ 时, 变异结束。其中 ζ 为非常小的数。传统的进化算法单纯采用这样的结束条件并不能保证所得的解为全局最优解, 只有通过反复的验证才能确定。采用极坐标寻优的方法不仅收敛速度较快的, 而且其搜索范围相当广泛。这是因为点的变化不仅决定于步长的变化, 而且受角度的影响, 点的跳跃性是很大的。因此通过判断父代与子代的差异完全能够判断收敛的解是否为全局最优解。

3.2 基于极坐标方法寻优算法的计算机实现过程

3.2.1 实现步骤

- 1) 初始化程序, 将函数的定义域转移到正半轴区间, 对上面的问题(P)就是将定义域转移到第一象限;
- 2) 然后函数中的直角坐标转化成极坐标, 函数方程变为极坐标方程;
- 3) 在定义域内产生两个随机点作为原始的父代和子代;
- 4) 将父代和子代的值分别保存在临时单元内, 用符合标记为: 父代— μ , 子代— η , 下面论述中出现的符合 μ 、 η 就不再做解释。
- 5) 利用测度函数 (即适应函数) $F(\gamma, \alpha)$ 检测 μ , η 的适应性。
- 6) 如果 $F(\mu) > F(\eta)$, 那么

$$r_{k+1}=r_k+\sigma_1 \times N(0,1) \quad (3-4)$$

$$\alpha_{k+1}=\alpha_k+\sigma_2 \times N(0,1) \quad (3-5)$$

式 3-4 的 r_{k+1} 和式 3-5 中的 α_{k+1} 和代表进化子代的极径和极角, r_k 和 α_k 代表父代的极径和极角。该过程是在进化子代的适应性不如父代的情况下, 采用从父代重新

变异产生子代方式。

7) 如果 $F(\mu) < F(\eta)$, 那么

$$\mathbf{r}_{k+1} = \mathbf{r}_k + \sigma_1 \times \mathbf{N}(0, 1) \quad (3-6)$$

$$\alpha_{k+1} = (\sigma_3 \times \alpha(\mu)) + \sigma_4 \times \alpha(\eta) / (\sigma_3 + \sigma_4) \quad (3-7)$$

式 3-6 的 r_{k+1} 和式 3-7 中的 α_{k+1} 和代表进化子代的极径和极角, r_k 和 α_k 代表父代的极径和极角。该过程是在进化子代的适应性优于父代的情况下, 利用父代和子代的方向指导变异产生子代, σ_3 , σ_4 是调整系数。

8) 如果 $F(r_{k+1}, \alpha_{k+1}) > F(r_k, \alpha_k)$ 那么选择 $F(r_{k+1}, \alpha_{k+1})$ 作为新的子代, 否则选择 $F(r_k, \alpha_k)$ 作为子代。

9) 如果 $|F(r_{k+1}, \alpha_{k+1}) - F(r_k, \alpha_k)| < \zeta$, 则退出循环, 完成进化过程。否则回到步骤 4), 这里的 ζ 是事先取定的非常小的数。

10) 记录结果, 程序结束。

3.2.2 伪代码

进化策略的伪码描述:

```
procedure Evolutionary Strategy( );
```

Initial /*初始化程序代码*/

1

```
int t=0,r0,a0, r1,a1,father=0,son=0;
```

```
Change1 (x, y); /*将定于域平移到第一象限, 返回新的函数*/
```

```
Change2(x, y); /*将直角坐标转换成极坐标*/
```

}

$r_0 = \text{Random}(r^n)$ /*在定义域 r^n 内随机产生极径作为父代*/

$a0 = \text{Random}(a^n)$ /*在定义域 a^n 内随机产生极角作为子代*/

```

r1 = Random(r^n)           /*在定义域 r^n 内随机产生极径作为父代*/

```

```
a1 = Random(a^n) /*在定义域  $a^n$  内随机产生极角作为子代*/
```

1

```

{                               /*发生变异的地方*/

```

father=a0

son=al

```

w0=calculate f (r0,a0);          /*计算函数适应值*/
w1=calculate f (r1,a1);          /*计算函数适应值*/
if w1<w0 then
    r1=r0+  $\sigma_1 \times N(0, 1)$ 
    a1= a0+  $\sigma_4 \times N(0, 1)$  /*参与变异过程*/
else
    r1=r0+  $\sigma_1 \times N(0, 1)$ 
     $\alpha_{k+1} = (\sigma_3 \times \alpha(\text{Father})) + \sigma_4 \times \alpha(\text{Son}) / (\sigma_3 + \sigma_4)$ 
endif
}

w0=calculate f (r0,a0);          /*计算函数适应值*/
w1=calculate f (r1,a1);          /*计算函数适应值*/
if w1>w0 then
    select (r1,a1)                /*选择(r1,a1)作为下次计划的父代*/
else
    select (r0,a0)                /*选择(r0,a0)作为下次计划的父代*/
endif
t = t + 1;
}

While t<MAX or null(stopcriteria) do /*在这里进行退出判断*/
end;

```

3.3 改进进化策略父代的群体规模

变异的过程要通过调整系数对进化的速度和进化的收敛最终结果有着密切的关系，如果调整系数较小，则收敛速度比较快，但是很容易错过全局最优解，形成局部收敛的特征；如果调整系数较大，则收敛速度比较慢，但是却能扩大收敛空间，保证收敛到全局最优解，因此，调整系数对于进化策略有着积极、至关重要的作用。

在式 3-4、式 3-5、式 3-6、式 3-7 中出现的 σ 是调整系数，它是一个事先确定的实数，对于不同的问题对应的值也不同，这就需要经过人为的试验，找到较好的值。传统的进化策略并没有考虑到这个调整系数，将调整系数的值默认取定为 1，实际上这是不科学的，如果变异过程完全依靠变异概率函数不一定能够很好的适应具体问题的需要，有的问题的

定义域范围比较小,有的问题的定义域范围比较大,如果采用同样的变异方式则必然导致收敛速度或者收敛结果不理想,对多峰值函数表现尤其明显。

对于一个确定的求极值函数,采用改进进化策略方法,调整系数都有一个较好范围,能够确保值的收敛速度和收敛结果都达到全局最优。

由于调整系数的特点,为了避免由于调整系数而影响进化策略的收敛特性,通常采用多个父代群体的方式,以独立的方式进行变异,父代之间不存在交叉和选择的操作。选择操作只发生在父代和进化子代之间,这样点的分布就较为均匀,覆盖面交广。群体的规模视问题的复杂程度而定,对于单峰值问题,一个父代个体足够;对于一般程度的多峰值问题,父代规模取3个也基本上能够满足要求;对于较为复杂的多峰值问题,父代规模取5~7个可以达到较好的收敛特性。

本文采用的父代规模为1,仅仅是为了叙述和论证方便而已,在实际解题中应当采用多个父代规模为宜。采用1个父代种群达到较好的收敛特性的条件是选择恰当的调整系数。这是根据具体的问题反复实验得到的。实际上,在问题未确定的情况下,可以适当扩大种群规模来弥补调整系数的不足。

调整系数是可以根据极值函数的特点做一个事先的估算,根据较为系统的实验得出下面估算方法:

- (1) 如果极值函数的函数图像较为简单(峰值较少,全局最优点较为突出),则调整系数取1即可。
- (2) 如果函数的定义域较为狭长,例如 $x \in [-1000, 1000]$, $y \in [0.1, 0.2]$, 对于这样的问题,调整系数取0.1~0.5之间。
- (3) 如果函数的峰值图像较为复杂,则调整系数范围在0.05~0.1之间。

在选取上面的估算方式后,如果得到的结果收敛速度较慢,则适当缩小调整系数;反之,如果得到的结果全局收敛性不理想,则适当扩大调整系数。

在选择多个初始父代群体之后,调整系数对整个进化过程的影响降低很多,所以,解决实际问题的时候都是采用多个初始个体的方式完成的。

第四章 改进进化策略的收敛性分析

根据前面对改进算法的描述,改进算法其本质上是采纳了传统进化算法的核心内容,即进化步骤仍然采取变异、选择这样的基本思想,其改变的地方是更加注重父代的方向指引作用,如果进化后的子代优于父代,则利用子代和父代的方向特点来寻找下一个子代的位置;反之如果经过一次进化后子代的适应性不如父代,则采用传统进化策略的方法进行变异。在直角坐标系中利用角度远不如在极坐标系,所以采用极坐标来作为寻优过程的背景平台。算法的稳定性集中体现在具有较好的收敛性,能够收敛到极值点,并且是全局最优解。

下面对本文提出的改进进化策略算法的收敛性进行证明[22][23][24][25],该算法不仅能很好地收敛,并且能够收敛到全局最优解。

考虑全局优化问题

$$(p) \quad \min\{f(x,y): x,y \in R^n\}, f: R^n \rightarrow R^1 \quad (4-1)$$

转化为极坐标形式为 $\min\{f(r, \alpha): r \in R^n, \alpha \in [0, \pi/2]\}$, 对函数 $f(r, \alpha)$, 始终有 $S=\{(r, \alpha)|\arg$

$$\min_{r \in R^n, \alpha \in [0, \frac{\pi}{2}]} f(r, \alpha) = \min_{r \in R^n, \alpha \in [0, \frac{\pi}{2}]} f(r, \alpha) = f^*, \text{ 任取 } \varepsilon > 0, \text{ 记}$$

$$D_0 = \{(r, \alpha) | f(r, \alpha) - f^* < \varepsilon\} \quad (4-2)$$

$$D_0 = R^n \setminus D_0$$

把上述算法中产生的点分成两种状态:

A. 至少有一个点属于 D_0 的, 记为状态 S_0 ;

B. 所有的点都属于 D_1 的, 记为状态 S_1 。

设 $\{(r_1(0), \alpha_1(0)), (r_2(0), \alpha_2(0)), \dots, (r_\mu(0), \alpha_\mu(0))\}$ 为上述算法产生的初始种群, 记 $\alpha = \max_{1 \leq i \leq \mu} f(r_i(0), \alpha_i(0))$ 。首先给出如下定理。

定理 1 设 $(R(m), \alpha(m))$ 是有算法产生的种群序列, 其中 $(R^*(m), \alpha^*(m)) \in (R(m), \alpha(m))$ 为 m 代种群中的最优点, 即 $(R^*(m), \alpha^*(m)) = \arg \min_{1 \leq j \leq \mu} f(r_j(m), \alpha_j(m))$, 如果算法初始种群对应的水平集 $L_\alpha = \{x | f(x) < \beta\}$ 有界, 则任取 $\varepsilon > 0$, 都有:

$$\lim_{m \rightarrow \infty} p(|f(r^*(m), \alpha^*(m)) - f^*| < \varepsilon) = 1 \quad (4-3)$$

即该算法依概率收敛于全局最优点。

为了证明上述定理, 先给出如下两个引理:

引理 1 设 $r=(r^1, r^2, \dots, r^n) \in R^n$, $\Delta r \in (\Delta r^1, \Delta r^2, \dots, \Delta r^n)$, 且 Δr^i 服从 $N(0, \sigma_i^2)$ 的正态分布, $\alpha=(\alpha^1, \alpha^2, \dots, \alpha^n) \in [0, \pi/2]$, $\Delta \alpha \in (\Delta \alpha^1, \Delta \alpha^2, \dots, \Delta \alpha^n)$, 且 $\Delta \alpha^i$ 服从 $N(0, \sigma_i^2)$ 的正态分布, 记 $Q_{y,\delta}=\{(r, \alpha) | \|(r, \alpha)-y\|_\infty \leq \delta\}$, 则对任意的 $y \in R^n$ 与 $\delta > 0$, 有:

$$\begin{aligned} P((r+\Delta r) \in Q_{y,\delta}) &> 0; \\ P((\alpha+\Delta \alpha) \in Q_{y,\delta}) &> 0 \end{aligned} \quad (4-4)$$

证明 因为

$$\begin{aligned} P((r+\Delta r) \in Q_{y,\delta}) &= \prod_{i=1}^n P(|r^i + \Delta r^i - y^i| \leq \delta) \\ &= \prod_{i=1}^n P(y^i - r^i - \delta \leq \Delta r^i \leq y^i - r^i + \delta) \end{aligned}$$

又

$$\Delta r^i \sim N(0, \sigma_i^2),$$

故有

$$\begin{aligned} P((r+\Delta r) \in Q_{y,\delta}) &> 0 \\ P((\alpha+\Delta \alpha) \in Q_{y,\delta}) &> 0 \end{aligned}$$

同理可证:

可以简写为: $P(((r+\Delta r), (\alpha+\Delta \alpha)) \in Q_{y,\delta}) > 0$

这里, 记 $(R(k), \alpha(k))$ 为算法中第 K 代产生的 μ 个点所组成的集合。

引理 2 设算法的初始点满足定理 1 中的条件, 记 $q_{ij}(i, j=0, 1)$ 为当 $(R(k), \alpha(k))$ 为状态 S_i 时, $(R(k+1), \alpha(k+1))$ 为状态 S_j 的概率, 则有:

- (1) 对于任意状态为 S_0 的点集 $(R(k), \alpha(k))$, $q_{00}=1$;
- (2) 对于任意状态为 S_1 的点集 $(R(k), \alpha(k))$, 存在 $0 < C_0 < 1$, 使得 $q_{11} \leq C_0$

证明 根据算法的步骤, (1) 式显然成立。下面来证明 (2) 式成立。

设 $M = \overline{C_0(L_\alpha)}$ 为水平集 L_α 的闭凸包。由于 f 连续, 设 $(r(0), \alpha(0))$ 为其一个最小值点, 故存在 $\varepsilon' > 0$, 使得当

$$\|(r, \alpha), (r(0), \alpha(0))\| \leq \varepsilon'$$

时, 有

$$|f(r, \alpha) - f(r(0), \alpha(0))| < \varepsilon/2$$

因此

$$Q_{(r(0), \alpha(0)), \varepsilon} \subset D_0$$

记

$$P_1(r)=P((r+\Delta r)\in Q_{r(0),\varepsilon})$$

$$P_1(r,\alpha)=P(((r+\Delta r),(\alpha+\Delta\alpha))\in Q_{(r(0),\alpha(0)),\varepsilon}), \text{ 其中 } (r,\alpha)\in M$$

由引理 1 可知 $P_1(r,\alpha)>0$, 且由于 $\Delta r, \Delta\alpha$ 服从正态分布, 故 $P_1(r,\alpha)$ 连续, 又由于 M 为有界闭集, 故存在 $(r^0, \alpha^0)\in M$, 使得 $P_1(r^0, \alpha^0)=\min_{(r,\alpha)\in M} P_1(r,\alpha)$ 且 $P_1(r^0, \alpha^0)>0$

记 $d=P_1(r^0, \alpha^0)$, 则 $0<d<1$, 由于 q_{10} 表示当 $(R(k), \alpha(k))$ 为状态 S_1 时, $(R(k+1), \alpha(k+1))$ 为 S_0 的概率, 故有 $q_{00}\geq d$, 令 $C_0=1-d$, 又因为 $q_{11}+q_{10}=1$, 所以 $q_{11}\leq C_0$, $(0<C_0<1)$ 。

定理 1 的证明如下:

证明 首先记 $q_{ij}^{(m)} (i,j=0,1)$ 作为初始状态 S_i , 经过 m 代后状态为 S_j 的概率, 显然:

$$q_{i0}^{(m)} + q_{i1}^{(m)} = 1$$

$$\text{且 } \forall m, q_{00}^{(m)} = 1$$

由于状态为 S_0 的点集不会再到状态 S_1 , 则由引理 2

$$q_{11}^{(m)} \leq (C_0)^m$$

故

$$\lim_{m \rightarrow \infty} q_{11}^{(m)} = 0$$

$$\lim_{m \rightarrow \infty} q_{10}^{(m)} = 1$$

即无论所选取的点初始状态如何, 只要水平集 L_α 有界; 则当 $m \rightarrow \infty$ 时, 算法产生的点集状态为 S_0 的概率为 1。即 $\forall \varepsilon > 0$, 都有

$$\lim_{m \rightarrow \infty} p(|f(r^*(m), \alpha^*(m)) - f^*| < \varepsilon) = 1 \quad (4-5)$$

定理 1 证毕

定理 2 设 $(r^*(m), \alpha^*(m))$ 与定理 1 定义相同, 初始点选取满足定理 1 中的条件。记 T 为函数 $f(r, \alpha)$ 的最小点组成的集合, 记 $d((r, \alpha), T)$ 为 x 与集合 T 的距离。则 $\forall \varepsilon > 0$, 都有

$$\lim_{m \rightarrow \infty} p(d((r^*(m), \alpha^*(m)), T) < \varepsilon) = 1 \quad (4-6)$$

证明 采用反证法。假设结论不成立, 则 $\exists \varepsilon > 0, \exists \delta_0 > 0, \forall m_0 \in \mathbb{Z}^+, \exists m > m_0$, 使得

$$p(d((r^*(m), \alpha^*(m)), T) < \varepsilon) < 1 - \delta_0$$

即

$$p(d((r^*(m), \alpha^*(m)), T) \geq \varepsilon) \geq \delta_0 \quad (4-7)$$

设 $G = \{(r, \alpha) | d((r, \alpha), T) \geq \varepsilon\}$, 则 G 为闭集, 令 $g(r, \alpha) = f(r, \alpha) - f^* > 0, (x \in G)$, 且设

$\min_{x \in G} g(r, \alpha) = C_1$ (C_1 为某一正常数), 则由定理 1 有, 对于 C_1, δ_0 , 存在 $m_1 \in \mathbb{Z}^+$, 使得 $\forall m > m_1$,

有

$$p(|f(r^*(m), \alpha^*(m)) - f^*| < C_1) > 1 - \delta_0$$

而假设中说, 对于 m_1 , 存在 $m > m_1$, 使得

$$p(d((r^*(m), \alpha^*(m)), T) \geq \varepsilon) \geq \delta_0$$

因为

$$\{(r, \alpha) \mid f(r, \alpha) - f^* \geq C_1\} \supset G$$

故

$$p(|f(r^*(m), \alpha^*(m)) - f^*| \geq C_1) > P(d((r^*(m), \alpha^*(m)), T) \geq \varepsilon) \geq \delta_0 \quad (4-8)$$

与 4-7 式矛盾, 因此假设错误, 原命题成立。

定理 3 在算法中, 初始点选取满足定理 1 中的条件, 若随机产生的每一个 $\Delta r^i, \Delta \alpha^i$ 服从的概率密度分别函数 $\psi(r, \alpha)$ 满足下列条件:

1) $\psi(r, \alpha)$ 连续

$$2) \forall a, b \in \mathbb{R}, a < b, \text{ 有 } \int_0^{\frac{\pi}{2}} \int_a^b \psi(r, \alpha) d\alpha dr > 0$$

则此算法产生的序列 $(r^*(m), \alpha^*(m))$ 满足 $\forall \varepsilon > 0$, 有

$$\lim_{m \rightarrow \infty} p(d((r^*(m), \alpha^*(m)), T) < \varepsilon) = 1 \quad (4-9)$$

由上述讨论可见, 对于改进的二元进化策略算法依概率收敛于函数的全局最优点, 对上述收敛的证明也可以利用马尔可夫链来完成。

第五章 改进算法的实例仿真

上一章已经证明改进算法具有较好的收敛性，下面，要通过对五个典型的多极值函数实例的寻优过程[26]，分别采用传统的进化策略和改进的进化策略的对比来说明改进算法的优越性，五个函数表达式如下：

$$f(x, y) = x * \sin(3\pi x) + y * \sin(10\pi y) + 30 \quad x \in [-3, 10], y \in [5, 5.5]。求最小值$$

$$f_2(x_1, x_2) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2 \quad -2.048 \leq x_i \leq 2.048 \quad i = 1, 2 \quad (\text{橡胶函数}) \text{求最小值}$$

$$f_3(x, y) = (4 - 2.1x^2 + \frac{x^4}{3})x^2 + xy + (-4 + 4y^2)y^2 \quad -3 \leq x \leq 3, -2 \leq y \leq 2, \text{求最大值}$$

$$f_4(x_1, x_2) = 0.5 + \frac{\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5}{[1 + 0.1(x_1^2 + x_2^2)]^2} \quad -100 \leq x_i \leq 100, i = 1, 2。求最小值$$

$$f(x, y) = \sqrt{x^2 + y^2} + x * y - y + 3 - \sin(x^2 + y^3 + 3) + 4 \quad x, y \in [-8, 8], \text{求其最小值。}$$

为了论述方便，作出如下规定：

1. 子代的规定

进化过程中，如果进化子代的适应性强于父代，对整个进化过程起到积极、指导作用，我们就称其为“有效进化子代”。反之，如果进化过程出现的子代适应性差于父代，我们称其为“无效子代”。

2. 种群规模

本文所采用的例题，无论采用传统算法还是改进算法，其种群规模采用的都是 1，即单个体，在实际应用里面应该尽量增加个体的数量，提高最终寻优点的可靠性。

5.1 实例 1

对二元多极值函数表达式如下：

$$f(x, y) = x * \sin(3\pi x) + y * \sin(10\pi y) + 30 \quad (5-1)$$

函数的实际图像如图 5-1 所示，求其最小值，其中 $x \in [-3, 10], y \in [5, 5.5]$ 。

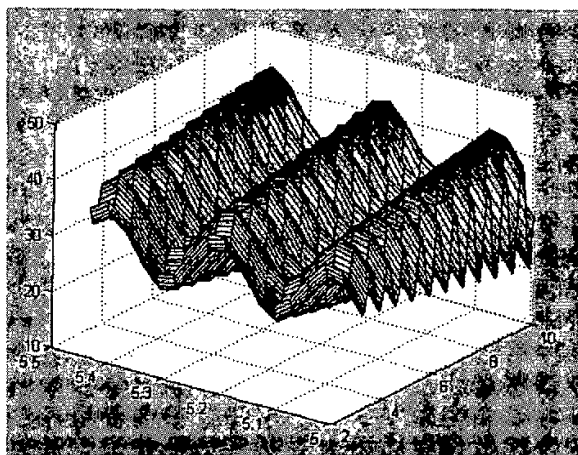


图 5-1 函数 (5-1)

Fig. 5-1 function (5-1)

5.1.1 传统进化策略

采用传统的进化策略所得的函数收敛过程及对应的代数如表 5-1 所示:

迭代次数	对应函数值	迭代次数	对应函数值
1	27.224	9	15.032
2	20.812	10	15.016
3	19.706	11	14.912
4	17.833	12	14.843
5	17.324	13	14.836
6	15.985	14	14.818
7	15.508	15	14.817
8	15.053	16	14.816

表 5-1 传统进化策略寻优结果

Table 5-1: Optimum seeking result by traditional evolution strategy

在表 5.1 中所列举的就是有效进化子代,“对应函数值”是在进化过程中所得到的最优解。“迭代次数”指的是有效进化次数,例如 1 表示第一次取得有效子代;2 表示的第二次取得有效子代……。而实际上该过程一共进行了 9768 次迭代运算,有效进化共 16 代,收敛的最终最优解 14.816。

寻优数据及其分布状况如图 5-2 中所示:

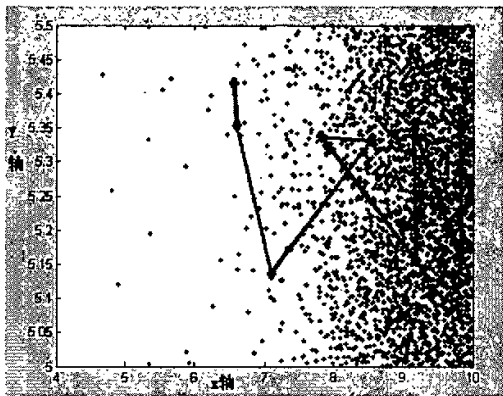


图 5-2 函数寻优过程的变化曲线



图 5-3 函数值的变化曲线

Fig. 5-2 The path of (X, Y) for optimum seeking Fig. 5-3 The curve of function value

图 5-2 中的线为有效进化代的连线,孤立点为无效子代点。对应二元函数图形(图 5-1),点的分布多汇集在 $x=10$ 附近,这是因为这里有条“山底”,其他地方有少量的点分布。其搜索范围广,但是没有搜索侧重点,受初值的影响较大。图 5-3 为有效子代对应的测度函数值,即二元极值函数值折线图,可以看出收敛的速度并不理想。

传统进化策略是在直角坐标系下研究的,有效进化点的所对应的函数变量 x, y 的变化曲线如图 5-4, 图 5-5 所示:

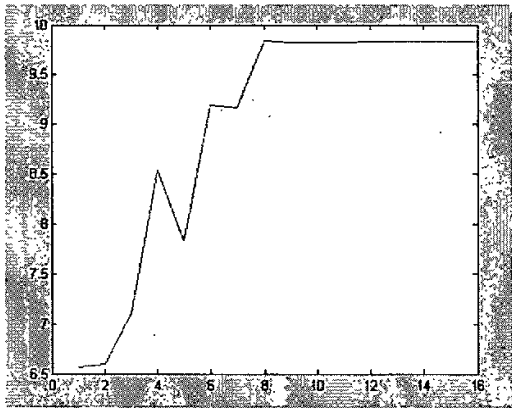


图 5-4 变量 x 变化折线

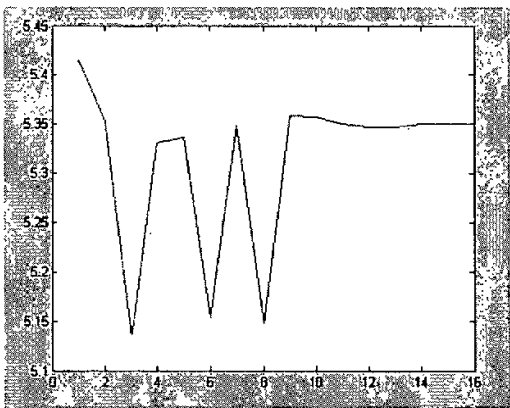


图 5-5 变量 y 变化折线

Fig. 5-4 The path of x Fig. 5-5 The path of y

由于传统进化策略是在直角坐标系下实现的,因此,函数变量 x, y 的变化可以作为改进算法的一个有利参考方面,由于变异因子是正态分布函数,因此, x, y 的变化还是比较舒缓的。一方面有利于局部收敛,另一方面不利于全局最优解的获取。

进化结束时的相邻函数值差为: 0.0012943。详细数据见附表 1。

5.1.2 改进进化策略

采用改进的进化策略所得的结果及代数如表 5-2 所示

迭代次数	对应函数值	迭代次数	对应函数值
1	24.048	32	17.264
2	23.853	33	17.224
3	22.112	34	17.143
4	21.758	35	17.117
5	21.695	36	16.964
6	21.53	37	16.821
7	21.31	38	16.819
8	20.995	39	16.375
9	20.941	40	16.368
10	20.869	41	16.354
11	20.753	42	16.205
12	20.596	43	16.02
13	20.587	44	15.97
14	20.547	45	15.781
15	20.383	46	15.724
16	20.26	47	15.58
17	20.177	48	15.519
18	19.899	49	15.501
19	19.79	50	15.496
20	19.641	51	15.483
21	19.576	52	15.483
22	19.507	53	15.078
23	19.503	54	15.017
24	19.501	55	15.016
25	19.5	56	15.016
26	19.486	57	14.87
27	19.242	58	14.84
28	18.922	59	14.821
29	18.846	60	14.82
30	18.824	61	14.817
31	18.819	62	14.816

表 5-2 改进的进化策略的结果

Table. 5-2 :Optimum seeking result by evolution strategy after improving

采用改进进化策略一共进化了 3470 代，有效进化 16 代。在表 5.2 中所列举的就是有效进化子代，“对应函数值”是在进化过程中所得到的最优解。“迭代次数”指的是有效进化次数。与传统进化策略对比：改进进化策略的进化速度占有明显的优势，在这里收敛速度提高了接近 3 倍。表 5-2 所列举的有效进化代数的增加说明改进算法其收敛稳定性较好，能够在局部达到较好的收敛特性。

寻优数据及其分布状况如图 5-6 中所示：

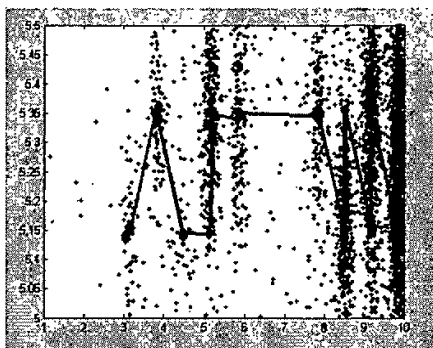


图 5-6 函数寻优过程的变化曲线

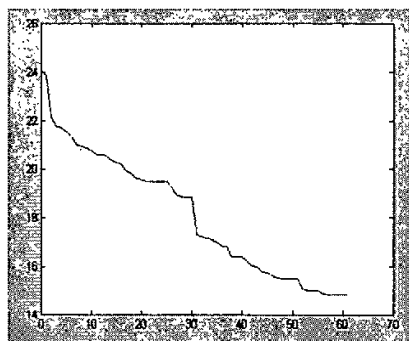


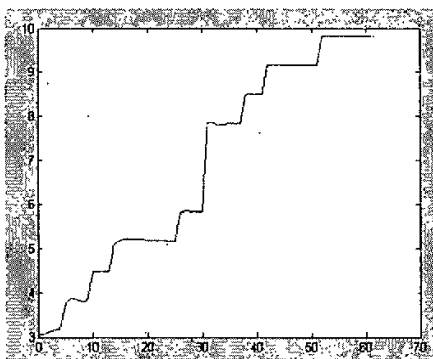
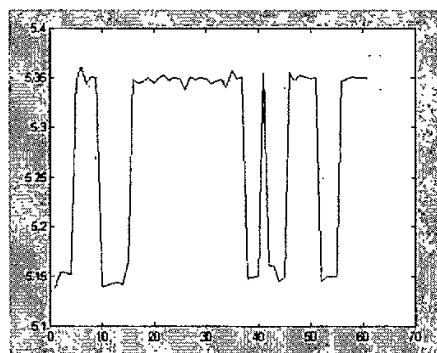
图 5-7 函数值的变化曲线

Fig. 5-6 The path of (x, y) for optimum seeking

Fig. 5-7 The curve of function value

图 5-6 中的线为有效进化代的连线,孤立点为无效子代点。对应二元函数图形(图 5-1),点的分布基本上已经覆盖了每一条“山底”,点的分布规律明显增强;线的路径也具有明显的规律性。这是因为子代的选择有明显父代的优秀特点(即方向性)。其他地方也有少量的点分布,其搜索范围比传统的进化策略更广,搜索侧重点突出,受初值的影响较小。图 5-7 为有效子代对应的测度函数值,即二元极值函数值折线图,收敛的速度很理想。

为了便于对照,将极坐标再转换成直角坐标系,研究有效进化点的所对应的函数变量 x , y 的变化曲线如图 5-8, 图 5-9 所示:

图 5-8 变量 x 变化折线Fig. 5-8 The path of x 图 5-9 变量 y 变化折线Fig. 5-9 The path of y

改进的进化策略是在极坐标系下实现的,因此,函数变量 x , y 的变化是通过极坐标下的极角和极径的变化实现的。极角的变化有效的参考了父代的特性,极径的变异则是通过变异因子,这里仍采用正态分布函数。从图 5-8 和图 5-9 的图形曲线可以看出: x , y 的变化性还是比较有规律的。一方面有利于局部收敛能力较强,另一方面也兼顾了全局最优解的获取。

进化结束时的相邻函数值差为: 0.00042543。效果非常明显。详细数据见附表 1。

5.2 实例 2

对二元多极值函数（香蕉函数）表达式如下：

$$f_2(x_1, x_2) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2 \quad (5-2)$$

函数的实图像如图 5-10 所示，求其最大值，其中 $x \in [-2.048, 2.048]$, $y \in [-2.048, 2.048]$ 。

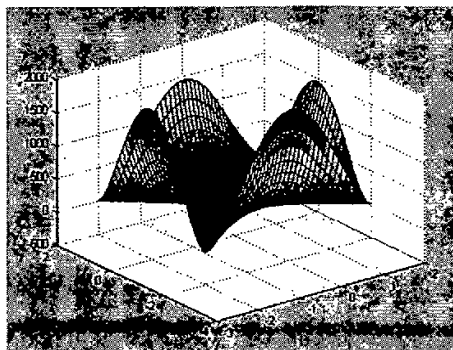


图 5-10 函数 (5-2)

Fig. 5-10 function (5-2)

5.2.1 传统进化策略

采用传统的进化策略所得的函数收敛过程及对应的代数如表 5-3 所示：

迭代次数	对应函数值	迭代次数	对应函数值
1	13.445	13	1737.6
2	119.41	14	1743.4
3	548.56	15	1745
4	635.63	16	1758.2
5	650.7	17	1758.7
6	1025.5	18	1759.3
7	1183.2	19	1759.3
8	1201.2	20	1760.1
9	1485.9	21	1760.1
10	1498.6	22	1760.1
11	1635	23	1760.1
12	1695.5		

表 5-3 传统的进化策略的结果

Table 5-3: Optimum seeking result by traditional evolution strategy

在表 5-3 中所列举的就是有效进化子代，“对应函数值”是在进化过程中所得到的最优解。“迭代次数”指的是有效进化次数，该进化过程一共进行了 78336 次迭代运算，有效进化共 23 代，收敛的最终最优解 1760.1。

寻优数据及其分布状况如图 5-11 中所示：

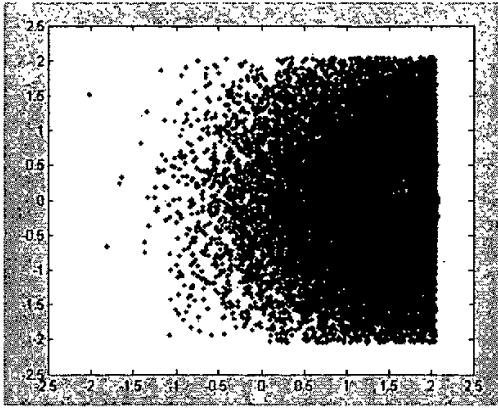


图 5-11 函数寻优过程的变化曲线

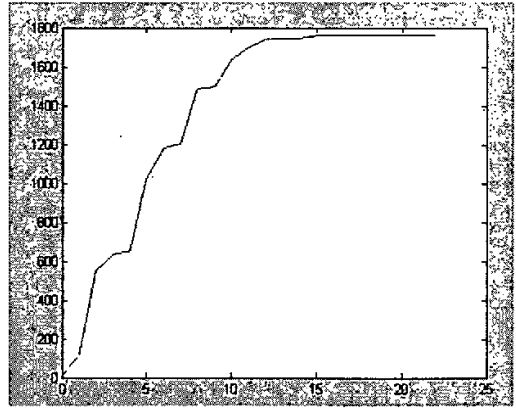


图 5-12 函数值的变化曲线

Fig. 5-11 The path of (X1, X2) for optimum seeking

Fig. 5-12 The curve of function value

图 5-11 中的线为有效进化代的连线，孤立点为无效子代点。对应二元函数图形（图 5-10），点的分布多汇集在香蕉函数的突起部分，点的重复性较强，这是浪费进化代数的主要原因，其他地方有少量的点分布。图 5-12 为有效子代对应的测度函数值，即二元极值函数值折线图。

传统进化策略是在直角坐标系下研究的，有效进化点的所对应的函数变量 x , y 的变化曲线如图 5-13，图 5-14 所示：

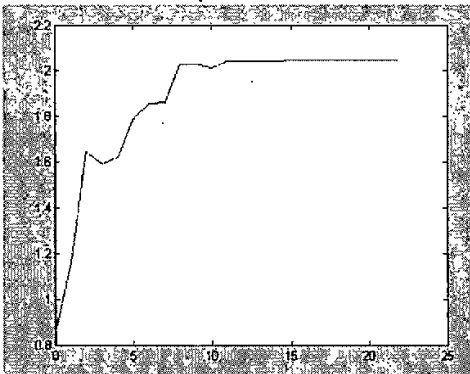


图 5-13 变量 x 变化折线

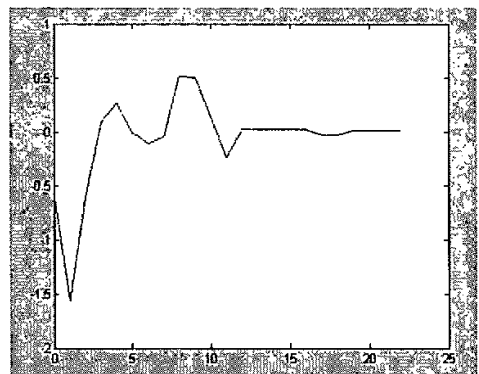


图 5-14 变量 y 变化折线

Fig. 5-13 The path of x

Fig. 5-14 The path of y

由于传统进化策略是在直角坐标系下实现的，因此，函数变量 x , y 的变化可以作为改进算法的一个有利参考方面，由于变异因子是正态分布函数，因此， x , y 的变化还是比较舒缓的。一方面有利于局部收敛，另一方面不利于全局最优解的获取。

进化结束时的相邻函数值差为：0.0027568。

5.2.2 改进进化策略

采用改进的进化策略所得的结果及代数如表 5-4 所示

迭代次数	对应函数值	迭代次数	对应函数值
1	378.37	9	1707.5
2	532.3	10	1733.7
3	663.77	11	1740.6
4	805.52	12	1752.2
5	852.63	13	1754.9
6	1601.5	14	1758
7	1636.3	15	1759.9
8	1683.8	16	1760.1

表 5-4 改进进化策略寻优结果

Table. 5-4: Optimum seeking result by evolution strategy after improving

采用改进进化策略一共进化了 4010 代，有效进化 16 代。在表 5-4 中所列举的就是有效进化子代，“对应函数值”是在进化过程中得到的最优解。“迭代次数”指的是有效进化次数。与传统进化策略对比：改进进化策略的进化速度占有明显的优势，在这里收敛速度提高了接近 20 倍。在达到同样收敛结果的情况下，采用改进算法之所以有这样快的进化速度，是因为子代的选取能够充分利用父代的方向，减少了不必要的重复取值。效果非常明显。

寻优数据及其分布状况如图 5-15 中所示：

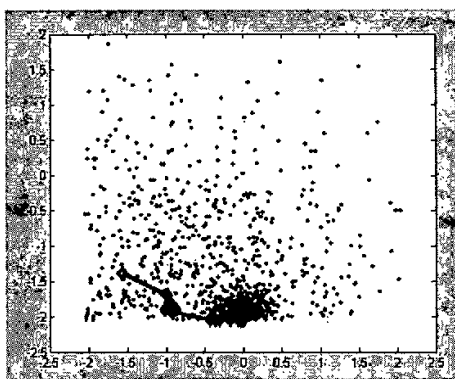


图 5-15 函数寻优过程的变化曲线

Fig. 5-15 The path of (X1, X2) for optimum seeking

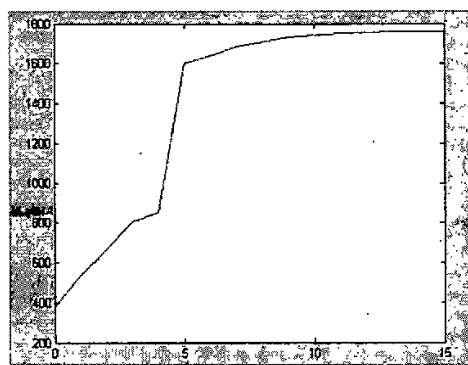


图 5-16 函数值的变化曲线

Fig. 5-16 The curve of function value

为了便于对照，将极坐标再转换成直角坐标系，研究有效进化点的所对应的函数变量 x , y 的变化曲线如图 5-17，图 5-18 所示：



图 5-17 变量 x 变化折线

Fig. 5-17 The path of x

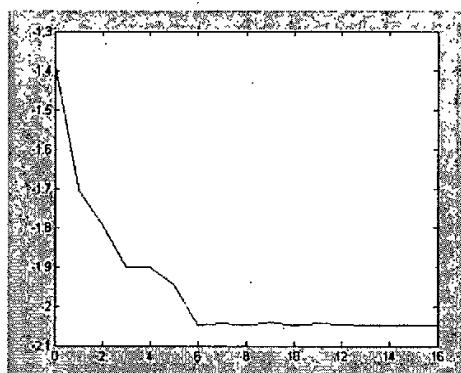


图 5-18 变量 y 变化折线

Fig. 5-18 The path of y

改进的进化策略是在极坐标系下实现的，因此，函数变量 x , y 的变化是通过极坐标下的极角和极径的变化实现的。极角的变化有效的参考了父代的特性，极径的变异则是通过变异因子，这里仍采用正态分布函数。从图 5-17 和图 5-18 的图形曲线可以看出： x , y 的变化性还是比较有规律的。一方面有利于局部收敛能力较强，另一方面也兼顾了全局最优解的获取。

进化结束时的相邻函数值差为：0.00054066。效果非常明显。

5.3 实例 3

对二元多极值函数表达式如下：

$$f_3(x, y) = (4 - 2.1x^2 + \frac{x^4}{3})x^2 + xy + (-4 + 4y^2)y^2 \quad (5-3)$$

函数的实际图像如图 5-19 所示，求其最大值，其中 $x \in [-3, 3]$, $y \in [-2, 2]$ 。

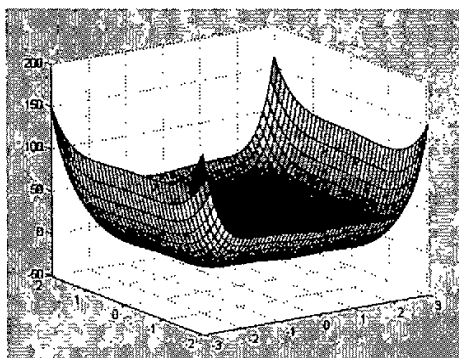


图 5-19 函数 (5-3)

Fig. 5-19 function (5-3)

5.3.1 传统进化策略

采用传统的进化策略所得的函数收敛过程及对应的代数如表 5-5 所示：

迭代次数	对应函数值	迭代次数	对应函数值
1	69.607	9	159.32
2	114.31	10	162.17
3	122.64	11	162.47
4	141.36	12	162.74
5	147.87	13	162.74
6	151.56	14	162.78
7	154.44	15	162.82
8	158.46	16	162.82

表 5-5 传统进化策略的寻优结果

Table 5-5: Optimum seeking result by traditional evolution strategy

在表 5-5 中所列举的就是有效进化子代，“对应函数值”是在进化过程中所得到的最优解。“迭代次数”指的是有效进化次数，该进化过程一共进行了 9148 次迭代运算，有效进化共 16 代，收敛的最终最优解 162.82。

寻优数据及其分布状况如图 5-20 中所示：

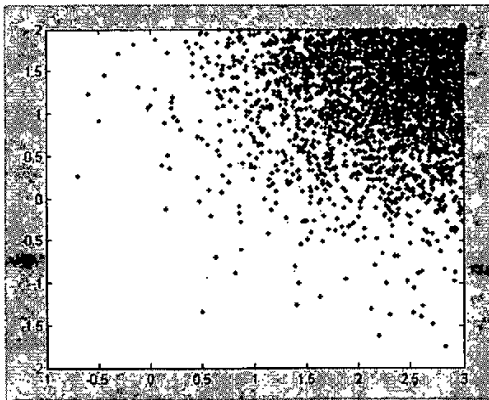


图 5-20 函数寻优过程的变化曲线



图 5-21 函数值的变化曲线

Fig. 5-20 The path of (x, y) for optimum seeking

Fig. 5-21 The curve of function value

图 5-20 中的线为有效进化代的连线，孤立点为无效子代点。对应二元函数图形（图 5-19），点的分布多汇集在函数的突起部分，其他地方有少量的点分布。图 5-21 为有效子代对应的测度函数值，即二元极值函数值折线图。

传统进化策略是在直角坐标系下研究的，有效进化点的所对应的函数变量 x , y 的变化曲线如图 5-22, 图 5-23 所示：

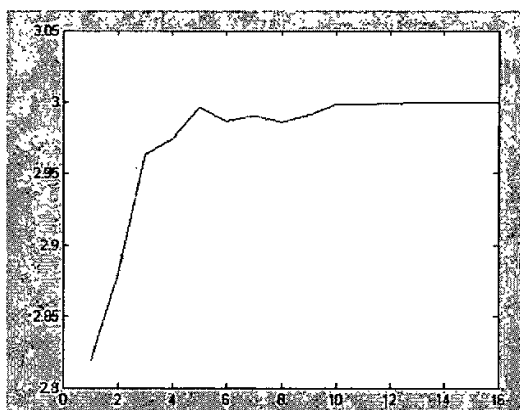


图 5-22 变量 x 变化折线

Fig. 5-22 The path of x

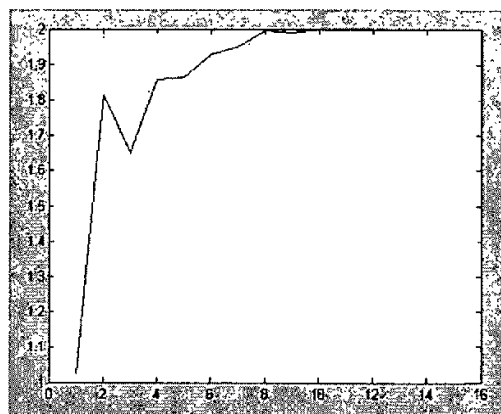


图 5-23 变量 y 变化折线

Fig. 5-23 The path of y

由于传统进化策略是在直角坐标系下实现的, 因此, 函数变量 x , y 的变化可以作为改进算法的一个有利参考方面, 由于变异因子是正态分布函数, 因此, x , y 的变化还是比较疏缓的。一方面有利于局部收敛, 另一方面不利于全局最优解的获取。

进化结束时的相邻函数值差为: 0.0074094。

5.3.2 改进进化策略

采用改进的进化策略所得的结果及代数如表 5-6 所示

迭代次数	对应函数值	迭代次数	对应函数值
1	73.397	6	161.92
2	134.99	7	161.97
3	147	8	162.88
4	153.9	9	162.89
5	161.82		

表 5-6 改进进化策略寻优结果

Table. 5-6: Optimum seeking result by evolution strategy after improving

采用改进进化策略一共进化了 957 代, 有效进化 9 代。在表 5-6 中所列举的就是有效进化子代, “对应函数值”是在进化过程中所得到的最优解。“迭代次数”指的是有效进化次数。与传统进化策略对比: 改进进化策略的进化速度占有明显的优势, 在这里收敛速度提高了接近 100 倍。在达到同样收敛结果的情况下, 采用改进算法之所以有这样快的进化速度, 是因为子代的选取能够充分利用父代的方向, 减少了不必要的重复取值。效果非常明显。

寻优数据及其分布状况如图 5-24 中所示:



图 5-24 函数寻优过程的变化曲线

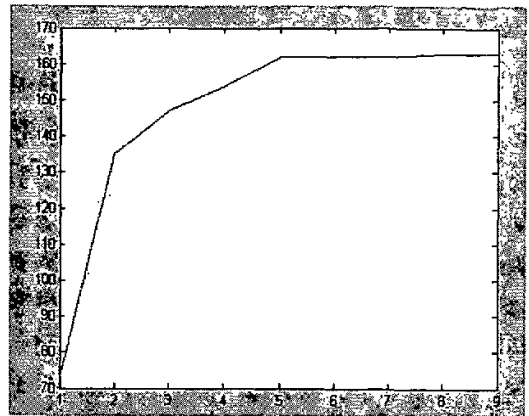


图 5-25 函数值的变化曲线

Fig. 5-24 The path of (X1, X2) for optimum seeking

Fig. 5-25 The curve of function value

为了便于对照，将极坐标再转换成直角坐标系，研究有效进化点的所对应的函数变量 x , y 的变化曲线如图 5-26, 图 5-27 所示：

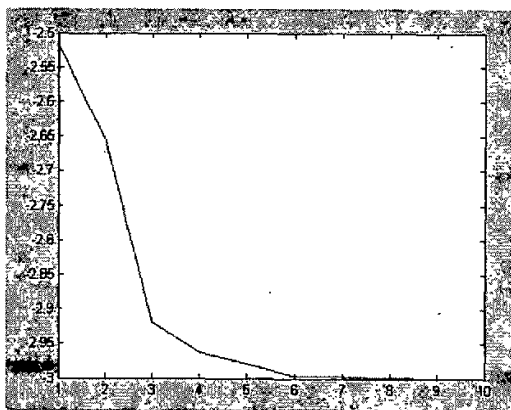


图 5-26 变量 x 变化折线

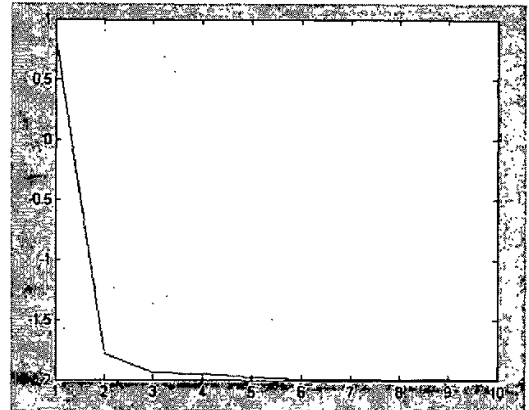


图 5-27 变量 y 变化折线

Fig. 5-26 The path of x

Fig. 5-27 The path of y

改进的进化策略是在极坐标系下实现的，因此，函数变量 x , y 的变化是通过极坐标下的极角和极径的变化实现的。极角的变化有效的参考了父代的特性，极径的变异则是通过变异因子，这里仍采用正态分布函数。从图 5-26 和图 5-27 的图形曲线可以看出： x , y 的收敛特性是非常好的。

进化结束时的相邻函数值差为：0.0018515。效果非常明显。

5.4 实例 4

二元多极值函数表达式如下：

$$f(x_1, x_2) = 0.5 + \frac{\sin^2(\sqrt{x_1^2 + x_2^2}) - 0.5 * x_1^2 + x_2^2}{[1 + 0.001 \times (x_1^2 + x_2^2)]^2} \quad (5-4)$$

函数的实际图像如图 5-28 所示, 求其最小值, 其中 $x_1, x_2 \in [-100, 100]$ 。

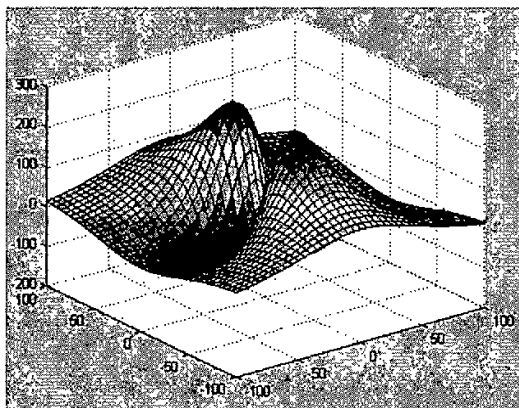


图 5-28 函数 (5-4)

Fig. 5-28 function (5-4)

5.4.1 传统进化策略

采用传统的进化策略所得的函数收敛过程及对应的代数如表 5-7 所示:

迭代次数	对应函数值	迭代次数	对应函数值	迭代次数	对应函数值
0	13.482	64	-25.46	128	-67.918
8	10.715	72	-29.014	136	-71.914
16	6.3872	80	-35.946	144	-82.19
24	3.2679	88	-40.236	152	-88.272
32	-3.4721	96	-42.871	160	-99.116
40	-9.1401	104	-47.594	168	-106.51
48	-14.602	112	-51.987	176	-119.18
56	-20.239	120	-60.594	192	-124.36

表 5-7 传统进化策略寻优结果

Table 5-7: Optimum seeking result by traditional evolution strategy

在表 5-7 中所列举的就是有效进化子代, “对应函数值”是在进化过程中所得到的最优解。“迭代次数”指的是有效进化次数, 该进化过程一共进行了 6327 次迭代运算, 有效进化共 192 代, 收敛的最终最优解-124.36。

寻优数据及其分布状况如图 5-29 中所示:

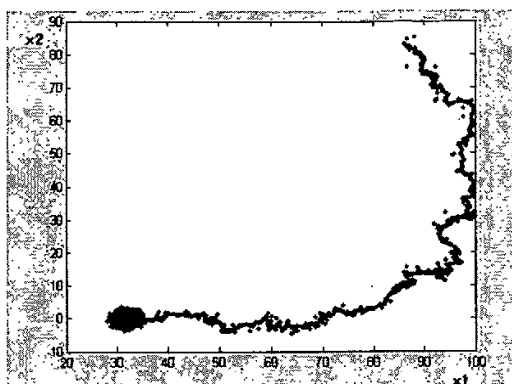


图 5-29 数据分布图

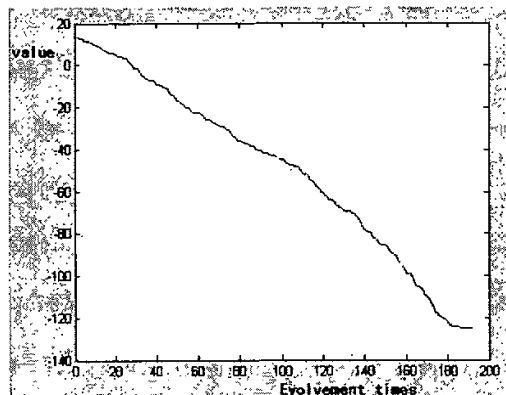


图 5-30 函数值变化曲线

Fig. 5-29 The path of (X1, X2) for optimum seeking

Fig. 5-30 The curve of function value

在图 5-29 中, 给出的是函数中的变量 x_1, x_2 的变化状况, 将其中进化过程中的选中为下一代的用连线连接起来, 寻优过程中不被选为下一代的点则不进行连接, 故用单点表示; 从图中寻优过程的分布状况可以看出该图像点的变化随机性是很大的, 收敛效果也不好。图 5-30 描述的是寻优过程函数值向极小值的变化曲线, 可以看出收敛的速度并不理想。

采用改进后的进化策略

5.4.2 改进进化策略

采用改进的进化策略所得的结果及代数如表 5-8 所示

迭代次数	对应函数值	迭代次数	对应函数值
0	86.024	40	-102.14
5	39.622	45	-110.28
10	16.638	50	-112.32
15	-21.390	55	-120.39
20	-49.197	60	-123.82
25	-59.83	65	-124.31
30	-83.261	68	-124.49
35	-95.346		

表 5-8 改进进化策略寻优结果

Table. 5-8: Optimum seeking result by evolution strategy after improving

采用改进进化策略一共进化了 2164 代, 有效进化 68 代, 收敛到最优解 -124.49。在表 5-8 中所列举的就是有效进化子代, “对应函数值”是在进化过程中所得到的最优解。“迭代次数”指的是有效进化次数。与传统进化策略对比: 改进进化策略的进化速度占有明显的优势, 在这里收敛速度提高了接近 2 倍。在达到同样收敛结果的情况下, 采用改进算法之所以有这样快的进化速度, 是因为子代的选取能够充分利用父代的方向, 减少了不必要的重复取值。效果非常明显。

寻优数据及其分布状况如图 5-31 中所示:

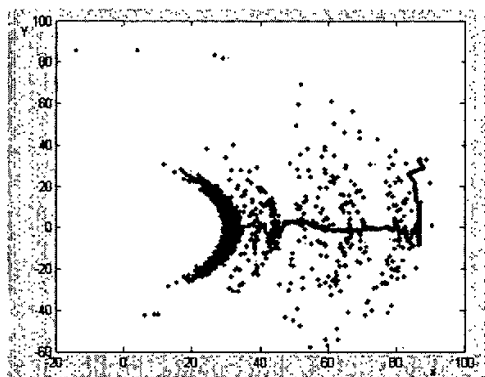


图 5-31 数据分布图

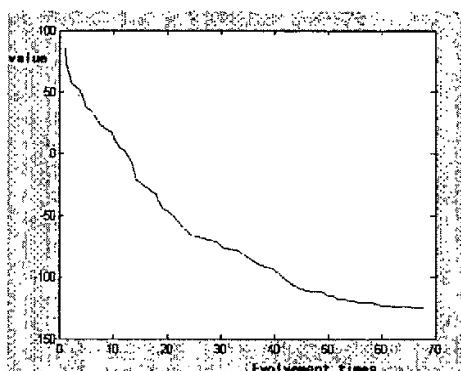


图 5-32 函数值变化曲线

Fig. 5-31 The path of (X1, X2) for optimum seeking Fig. 5-32 The curve of function value

在图 5-31 中, 给出的是函数中的变量 x, y 的变化状况, 将其中进化过程中的选中为下一代的用连线连接起来, 寻优过程中不被选为下一代的点则不进行连接, 故用单点表示; 从图中寻优过程的分布状况可以看出采用改进后的算法, 收敛速度有着显著的提高, 函数在定义域内的搜索范围比传统的方法要宽广得多, 寻优过程方向性也很好。图 5-32 是函数值的变化曲线, 其收敛的速度不仅快, 而且能够收敛到全局最优最优解 -124.49。

5.5 实例 5

二元多极值函数的表达式如下:

$$F(x,y)=\sqrt{x^2+y^2+x*y-y+3}-\sin(x^2+y^3+3)+4 \quad (5-5)$$

函数的实际图像如图 5-33 所示, 求其最小值, 其中 $x,y \in [-8, 8]$ 。

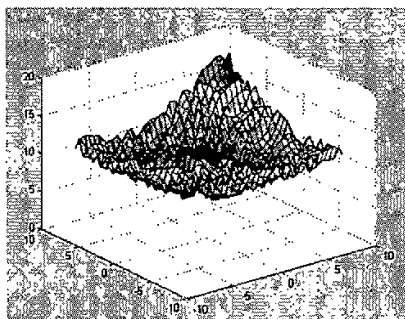


图 5-33 函数 (5-5)

Fig.5-33 function (5-5)

5.5.1 传统进化策略

采用传统的进化策略所得的函数收敛过程及对应的代数如表 5-9 所示:

迭代次数	对应函数值	迭代次数	对应函数值
0	14.604	16	7.498
2	13.973	18	7.1401
4	13.188	20	6.4541
6	12.225	22	5.1252
8	11.092	24	4.8597
10	10.172	26	4.8297
12	9.7144	28	4.824
14	9.2257		

表 5-9 传统进化策略收敛结果

Table 5-9: Optimum seeking result by traditional evolution strategy

在表 5-9 中所列举的就是有效进化子代,“对应函数值”是在进化过程中所得到的最优解。“迭代次数”指的是有效进化次数,该进化过程一共进行了 3165 次迭代运算,有效进化共 28 代,收敛的最终最优解 4.824。

寻优数据及其分布状况如图 5-34 中所示:

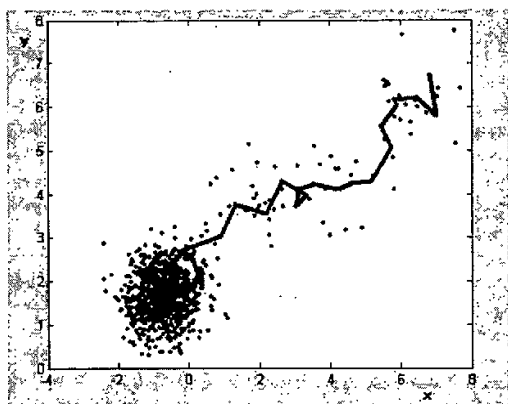


图 5-34 数据分布图

Fig. 5-34 The path of (x, y) for optimum seeking

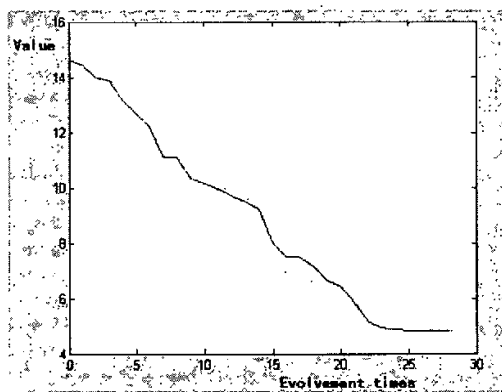


图 5-35 函数值变化曲线

Fig. 5-35 The curve of function value

在图 5.34 中,给出的是函数中的变量 x, y 的变化状况,将其中进化过程中的选中为下一代的用连线连接起来,寻优过程中不被选为下一代的点则不进行连接,故用单点表示;从图中寻优过程的分布状况可以看出该图像点虽然能够收敛,但是方向性很差。图 5-35 描述的是寻优过程函数值向极小值的变化曲线,可以看出收敛的速度并不理想。

5.5.2 改进进化策略

采用改进的进化策略所得的结果及代数如表 5-10 所示

迭代次数	对应函数值	迭代次数	对应函数值
0	10.668	6	5.9
1	8.2761	7	5.7088
2	7.6344	8	5.3007
3	7.4536	9	4.9361
4	6.5752	10	4.832
5	6.1201	11	4.8218

表 5-10 改进进化策略收敛结果

Table. 5-10: Optimum seeking result by evolution strategy after improving

采用改进进化策略一共进化了 1023 代，有效进化 11 代，收敛到最优解 4.8218。在表 5.10 中所列举的就是有效进化子代，“对应函数值”是在进化过程中所得到的最优解。“迭代次数”指的是有效进化次数。与传统进化策略对比：改进进化策略的进化速度占有明显的优势，在这里收敛速度提高了接近 2 倍。在达到同样收敛结果的情况下，采用改进算法之所以有这样快的进化速度，是因为子代的选取能够充分利用父代的方向，减少了不必要的重复取值。效果非常明显。

寻优数据及其分布状况如图 5-36 所示：

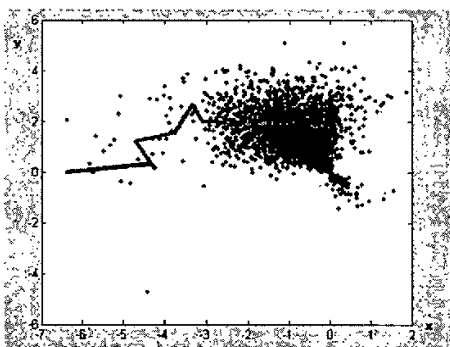


图 5-36 数据分布图

Fig. 5-36 The path of (X1, X2) for optimum seeking

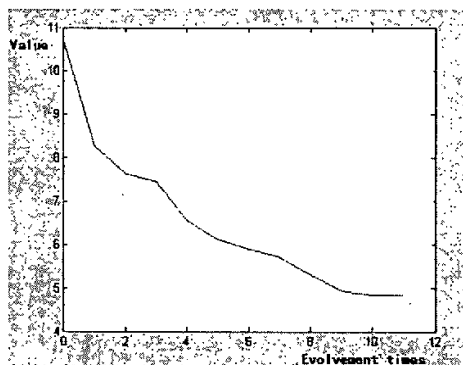


图 5-37 函数值变化曲线

Fig. 5-37 The curve of function value

在图 5-36 中，给出的是函数中的变量 x, y 的变化状况，将其中进化过程中的选中为下一代的用连线连接起来，寻优过程中不被选为下一代的点则不进行连接，故用单点表示；从图中寻优过程的分布状况可以看出该寻优过程是可取的，较好的利用了父代的方向，用较短的时间、较短的距离达到最优解。图 5-37 是函数值的变化曲线，其收敛的速度很快，而且能够收敛到全局最优最优解 4.8218。

5.6 本章小结

通过上面 5 个典型函数的极值寻优过程可以看出传统的进化算法存在着如下缺点：

- A. 收敛速度慢，进化的代数较多。
- B. 定义域内值的搜索范围小，较容易趋向局部最优解而错过全局最优解。
- C. 得到的最优解需要通过验证。

改进后的进化算法有如下的优点：

- A. 收敛速度快，能够充分利用父代提供的方向指导，最快的达到目标点。
- B. 极坐标中的角度变化对点的位置变化影响非常大，因此能够做到在收敛局部最优解的情况下，也能够跳出局部最优解继续搜索全局最优解。
- C. 收敛的最终解为全局最优解。

在采用极坐标方向寻优的进化策略时，应该考虑到从直角坐标系到极坐标系的变换过程变量定义域的变化，注意极径和极角的取值范围。

经过大量的试验证明，本文提出的用极坐标方向寻优方法改进传统的进化策略是非常明显、有效的，能够有效的缩短寻优时间。

第六章 改进算法思想在汽车电子导航方面的应用

进化策略以其优秀的特点在越来越多的领域得到了广泛的应用,具有非常大的发展空间,可以说能够有效地找到进化策略中的变异方式和测度函数的问题,就可以利用进化策略来解决,采用本文提出的利用方向性能够有效地来提高算法的收敛时间。本章将进化策略思想应用到汽车电子导航,是利用进化策略思想的一个突破。

6.1 路径规划简介

路径规划是帮助司机在旅行前或旅行中规划行驶路线的过程,被广泛认为是车辆导航领域中的一个基本问题。路径规划可分为多车辆路径规划和单车辆路径规划,即在一个特定的公路网上为所有车辆规划各自通向的目标路径,或者根据一个车辆当前的位置和目标给出单个路径规划[27]。

在计算机文献中,人们通常把求从 A 点到 B 点的一条路径问题称为最短路径问题。人们已经提出了许多算法解决单节点源最短路径问题和每一节点对最短路径问题,这些算法可视为单车辆路径规划和多车辆路径规划的情形。这里主要讨论解决单车辆路径规划问题的算法。多车辆路径规划比单车辆路径规划更复杂,但用于解决单车辆路径规划问题的背景知识将有利于研究解决多车辆路径规划情形。

在路径规划中,可采用各种路径优化标准。一条路径的好坏取决于许多因素[28],例如距离、旅行时间、旅行速度、拐弯和交通灯的数目和动态交通信息。我们把所有这些因素称为旅行费用。有些司机愿意选择最短路径,而其他司机愿意选择最短旅行时间等等。因此,选择的估价函数取决于系统的设计和用户的选择。路径选择标准可由设计决定或通过用户界面修改。为了求最小旅行距离(路段长度),距离可存储在数字地图数据库以便路径规划算法进行最小化时使用。如果最小化的是旅行时间,可以把路段长度和速度限制存储在数字地图数据库中以便计算每条路段的旅行时间时使用。线段长度和速度限制通常在数字地图数据库中被定义为路段的属性。总之,求最佳路径通常要用到数字地图数据库来选择使诸如时间和距离最小的一条路径。

最短路径算法主要代表之一是迪杰斯特拉(Dijkstra)算法[28]。该算法采用了在优化问题中常用的贪心技巧。贪心算法在每一步都选择局部最优解以期望产生一个全局最优解。给定一个加权有向图,如图 6-1 所示,迪杰斯特拉算法保存一个节点集合(即带有数字的小

圆圈), 节点相对于原节点的最小费用(即节点间的权值)是已知的。开始时, 这个集合仅包含原节点, 其他各节点为“剩余节点”。在每一步, 向集合加入一个“剩余节点”, 它的费用相对于原节点尽可能小。假设所有线段的费用都是非负的, 那么, 总能找到一条从原节点到“剩余节点”的最短路径, 这条路径经过的节点都是集合中的节点。我们称这样的路径为“特殊的”。在算法的每一步, 用一个数组记录每一节点的最短路径长度。一旦该集合包含所有节点, 所有路径都是“特殊的”, 该数组记录了从原节点到各节点的最短距离。我们很容易将一个道路网络转化为一个类似于图 6-1 的有向图。

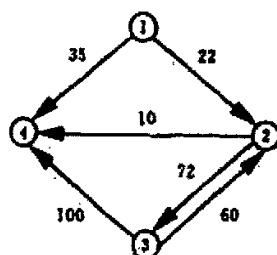


图 6-1 路径有向图

Fig. 6-1 Direction picture of route

用这个算法的一个必要条件是对每个有向线段来说费用必须是非负的, 这在车辆导航中总是真的。一个有向线段意味着每个路段有一个和它相关联的旅行方向。若非负的费用值的条件成立, 则 Dijkstra 算法将会求出从原节点到其他各节点的最优解。在这个例子中, 这意味着该算法分别求出从节点 1 到节点 2, 节点 1 到节点 3, 节点 1 到节点 4 的最小费用路径。

6.2 路径引导简介

路径引导是指司机沿着由路径规划模块计算出的路线来行驶的过程。该引导过程可以在旅行前或在途中以实时方式进行。行驶前的引导信息可以用打印机打印出来, 提供给司机。这类似于旅行代办处提供的旅行提示, 但打印出的信息包括详细的车辆行驶中依次转向的指令; 或者更象汽车出租柜台附近的小亭打印出的输出一样。这些指令可以包括转向、街道名称、行驶距离和路标。另一方面, 行进中的路径引导需要向司机提供实时的依次转向行驶指令, 这非常有用, 但需要一个导航地图数据库、准确的定位模块和具有实时计算能力的软件。

路径引导模块利用路径规划模块和定位子系统引导车辆行驶。除地图数据库模块外,

定位子系统可以仅包括定位模块，或者包括定位模块和地图匹配模块。一旦由路径规划模块计算出特定的路径且定位子系统已经确定了车辆的位置，路径引导模块与这些子系统协同工作以向司机提供适当的引导。引导模块与其他模块相互作用的简化功能框图如图 6.2 所示。注意在某些系统中行驶指令的产生是路径规划模块的主要任务。

随着车辆行驶，实时路径导航要求车辆的位置作为时间的函数不断地与路径规划模块产生的最佳路径相比较。根据车辆当前的位置、走向及行驶的道路信息，实时路径引导系统不断地更新这些信息。当转弯或者行驶指令到达时，路径引导系统可视信号、声频信号或者行驶指令向驾驶员发出报警信号。

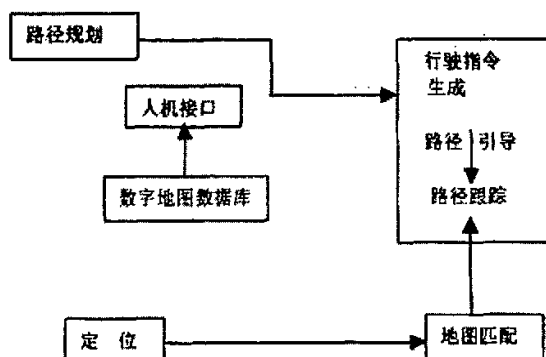


图 6-2 路径引导模块和其他模块相互作用的简化框图

Fig. 6-2 Route leading module and interaction

正如图 6-2 所示，路径引导包括两个任务：一是行驶指令的产生；二是规划路径的跟踪。注意对于路径引导模块不只是图 6-2 所示的一种方案，还有许多不同的设计和实现方案。例如，行驶指令生成任务可以是路径规划模块的主要部分。行驶指令生成任务和规划路径跟踪任务可直接处理由规划模块产生的数据。然而为了表达基本思想，本文利用图 6-2 所示的功能框图描述路径引导模块的原理和实际问题。

汽车导航系统中[29][30]，为了保证系统数据的实时性和数据量的最小化，通常在数字电子地图数据库中存储了关于道路的基本属性值，道路的当前拥塞信息则是实时进行传递的。为了保证实时性，电子地图系统不会对地图上的路线事先标记出具体值；也不会将任意两点的路径信息事先存储在电子地图数据库中。这就需要在驾驶员选择了起点和终点之后能够采用算法规划出一条“最短”路线来指引驾驶员。这里的最短路线是指到达指定位置花费时间最短的路线，这就是路径寻优问题。平常研究的路径寻优问题多是基于简单网络，出发点和终止点都是边缘点，例如图 6-3 所示：

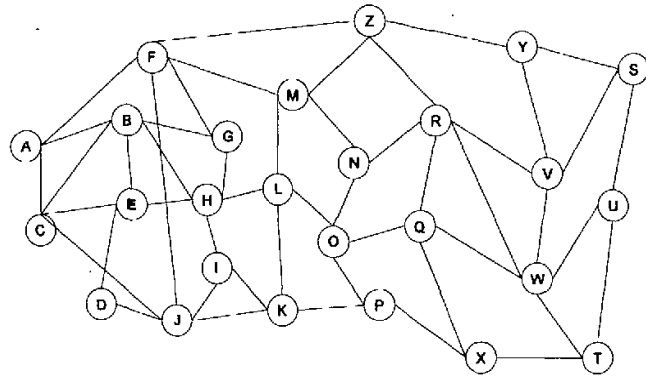


图 6-3 路径图

Fig. 6-3 Routing track

其中点 A, C, D, J, K, P, X, T, U, S, Y, Z, F 为边缘点，即位置处于图形边缘的点，在实际问题里出现边缘点的概率是非常低的，大多数都是处于非边缘位置，如点 B, E, H, G, L……。

采用改进进化策略解决这样的问题，其方法是利用改进进化策略的方向特点首先找到一条路径，然后以其作为测度尺度衡量所出现的路径优劣，下面首先说明该应用所采用的进化策略变异方式和测度函数。

6.3 利用改进进化策略实现路径寻优

汽车的行驶路线实际上就是由许多节点连接起来的路线，所以最优路线的寻找也就转变成节点的挑选和节点顺序排列问题。采用改进进化策略实现路径寻优的基本思想如下：

- (1) 首先在从起点到终点的大致方向上，初步随机产生一条路径，当然这条路径可能是比较长的路径，把这条路径作为测度函数的初始值，用来检验进化的优劣性。
- (2) 产生二维数据库，用于记录所有出现过的任意两个节点之间的最小“旅行费用”。
- (3) 从起点 Point(0)开始，对起始点的连接点进行排序，并且按照概率密度函数 $f(k)=2(n+1-k)/(n*n+n)$ 随机产生下一个连接点 Point(1)。
- (4) 从产生的点 Point(k)开始，对 Point(k)点的所有连接点（除 Point(k-1)点外）进行“排序”，序号顺序以 1, 2, ……n 表示。
- (5) 按照概率密度函数 $f(k)=2(n+1-k)/(n*n+n)$ 随机产生下一个连接点 Point(k+1)。
- (6) 将出现过的点放入数据库中作为横坐标和纵坐标，二维数据库中的值填充为节点的距离。如果节点在数据库中没有记录，则添加节点；否则，比较原来数据库中的值，如果当前的值小于数据库中的值则进行替换，如果当前的值大于数据库中的记录

值，则用数据中的值替换当前值。

- (7) 如果未到达终点，判断当前累计的“旅行费用”是否大于测度函数的值，如果已经超过测度函数值则返回至 (3)。
- (8) 如果未达到终点，判断当前累计的“旅行费用”是否大于测度函数的值，如果未超过测度函数值则返回至 (4)。
- (9) 如果已经达到终点，判断“旅行费用”与测度函数值，如果大于测度函数的“旅行费用”，则返回至 (3)。
- (10) 如果连续 n 次出现旅行费用相同，则结束路径寻优过程。打印节点顺序。

在图 6-4 中，假如汽车现在处于 B 点，目的地是 V 点，要寻找一个最优路径，分别从以下几个问题具体说明如何采用进化策略：

6.3.1 进化代的选取方式

这里，进化过程中的每一代所包含的节点并不是一个固定数目的量，进化代的规模也不固定。有些类似家谱，把每一条路径比喻成一个支流血脉，路线长度的比喻成所支流血脉的“旅行费用”。那么路线图可以解释 B 点是最初始的代—第 1 代；然后发展为第 2 代：B-A (2)、B-C (4)、B-E (1)、B-H (6)、B-G (4)，其中括号中的数字表示支流存在的旅行费用；接着发展为第 3 代……。最终目的就是要找到其“旅行费用”最小的一个支流。

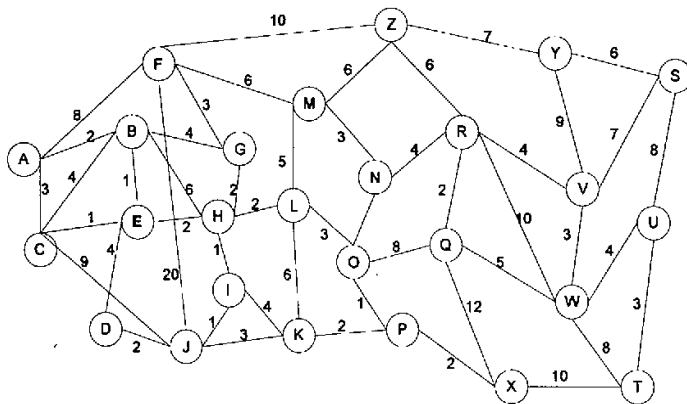


图 6-4 路径轨迹图

Fig. 6-4 Routing track

每一次进化都会增加新的支流，同时也增加了每个支流的旅行费用，这时就需要利用测度函数对支流进行检测，及时的抛弃适应性较差的支流。

6.3.2 测度函数的获得

为了衡量进化的优劣,在进行选择之前需要确定一个测度函数,利用改进算法中提到的进化策略的方向性从 B 到 V 点找一系连通的节点作为一个测度值,如果在进化过程中发现有比这个值更小的,那么这个测度值也相应发生变化,因此称为测度函数而不是测度值,用公式表示为:

$$f(i) = \sum_{l=1}^n a_l(n). \quad (6-1)$$

上式中 n 的值为所有经过 (节点) 路径个数; a 为节点; i 为第 i 个路径; $f(i)$ 为第 i 条路径的“旅行费用”值。

为了获取第一个测度函数值,要用一条线连接 B 和 V, 如图 6.3 所示

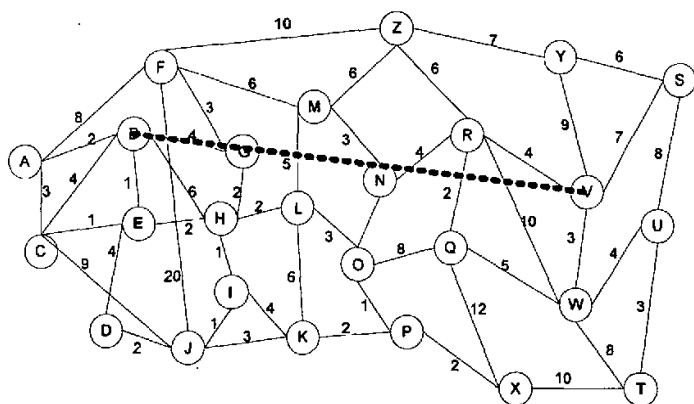


图 6-5 路径轨迹图

Fig. 6-5 Routing track

然后从 B 点出发点,在与 B 点相连接的节点中选择距离 V 最近的点且距离 BV 直线较近的点,根据图 6-5 可以得到 G 点为较为合适的点,然后在以 G 为出发点继续寻找直到到达终点 N。得到的初始路径为 B—G—H—L—M—N—R—W,测度值为 $4+2+2+5+3+4+10+3=33$ 。33 作为测度函数的初始值,如果经过几代进化发现有更小的从 B 到 G 的值时,这个测度函数值将被取代。

6.3.3 变异方式

本章利用自适应的变异方式来实现路径的寻优,路径寻优过程的变异首先对节点的路径进行排序,排序的方法为:

(1) 首先,比较与当前节点连接的所有节点 (除父代节点) 的下一个节点距离终点

的距离, 距离最近的排序为 1, 其次为 2, ……。

(2) 如果存在两个或者两个以上节点距离终点相同距离, 则再按照节点距离起点、终点连线的距离, 由小到大进行排序。

在完成排序工作之后, 根据排序的结果通过变异随机产生一个进化子代节点。这里的随机是按照一定规律产生的, 向着终点的方向的节点有着较高的变异概率, 背离终点方向的有较低的概率, 所用到的概率密度函数为:

$$f(k)=2(n+1-k)/(n*n+n)$$

式中 n 为所有可变化的节点数目; k 为排序的序号; $f(k)$ 为对应的概率。 n, k 为正整数。变异概率密度函数图像如图 6-6 所示:

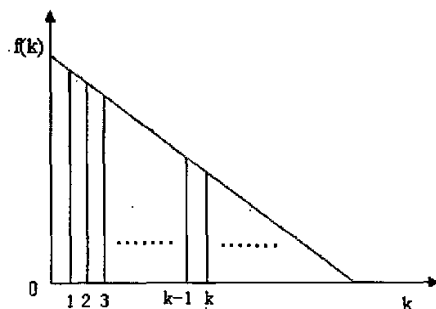


图 6-6 变异概率密度函数

Fig. 6-6 Variation probability function

整个变异空间所有变异节点的概率累加和为 1, 具有较好方向的变异概率较大, 占有明显优势。例如图 6.3 中从起始点 B 开始到达 H 点之后, H 点的变异子代排序为:

变异子代排序	1	2	3	4
对应节点	L	G	I	E

表 6-1 变异子代排序

Table. 6-1 compositor of variational generation

表 6-1 是根据上述排序方式得到, 然后利用变异概率密度函数随机产生下一个节点, 这里表示 L 的变异概率大于 G, G 的变异概率大于 I, I 的变异概率大于 E。

6.3.4 选择方式

在由第二代进化第三代的时候会出现图 6.4 所示的问题:

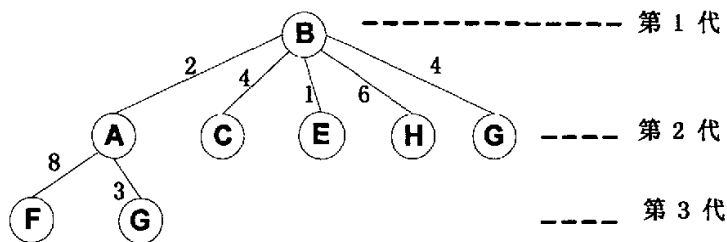


图 6-7 进化过程

Fig. 6-7 Evolution process

第 3 代出现了和第二代相同的节点“G”，这时要进行一次比对，B—A—G 的旅行费用为： $2+3=5$ ，B—G 的旅行费用为：4，因此看到支脉 B—A—G 旅行费用较大应该删除，删除时候注意删除的是支脉 B—A—G 的 G 点。同理，如果在两条支脉中存在相同的两个节点，但节点之间的旅行费用不同，这时候删除旅行费用较大的支脉。

上面所阐述的删除支脉的操作严格来说并不是选择，真正意义上的选择是指利用测度函数值来衡量支脉的生存能力，如果某一个支脉的旅行费用大于测度函数，则不管这个支脉是否达到终点、是否已经非常接近终点，都会被删除。例如图 6.3 所示采用路线 B, J, L, M, F 的旅行费用为 29，达到 Z 时旅行费用为 39，与测度函数值（当前为 34）比较，已经大于 34，因此删除该路径。

6.3.5 通过数据库修正路径

在数据库中记录了所有曾经走过的点的相互之间的距离关系，并且用较小的值随时对数据库进行更新，这种操作在路径的延伸过程被完成。当成功的完成了从起点到终点的寻优过程后，需要利用已经获得的数据库信息来修正当前路径，使得尽可能的找到最短路径，例如从 B 到 V 的变化过程找到这样一条路径 B-H-I-K-P-O-Q-W-V，通过查询数据库发现从 Q 到 V 有一条更好的路径 Q-R-V，可以节省 2 个旅行费用。这样，就可以利用数据库来修正已经找到的路径。此次进化的路径为：B-H-I-K-P-O-Q-R-V

6.3.6 路径寻优的终止方式

随着支脉节点的不断推进，会有一条旅行费用率小于测度函数的支脉到达目标点，这时将测度函数值改为当前的旅行费用，并记录下这个支脉。如果连续 n 次有效进化值都停留在同一个旅行费用上，则进化过程结束。最后记录下来的支脉就是最优解。汽车路径寻优的最后结束条件可以描述为所有可行支脉已经校验完成时，终止条件发生。

6.4 电子地图的标记方式

正如前面所说,电子地图上的任意两点之间的坐标距离并不是上面说的旅行费用,这是由于汽车从地图上的一点到另外一点之间的行驶时间不仅有坐标距离决定,而且和道路状况有直接的关系,比如红灯的多少,道路拥塞程度,是否高峰,是否限制速度……。所有这些因素是不能够简单从地图上的坐标距离能够描述的,在汽车行驶过程中路况是一个变化的量,比如某个路段的拥塞程度是随时间变化的,所有这些数据通过当地的交通信息系统传输到汽车内,对这些数据进行加工,并且标记到每条路段上,这就是上面所说的旅行费用,每条路段都有个旅行费用。如果无法与当前交通信息系统联系,则把路段的实际距离看成旅行费用。

下面讲述一下对电子地图的一般处理过程:

在讨论算法的图 6.3 中,有明确的节点和路径,对实际电子地图处理的时候也需要确定这样的节点和路径,其一般方式是:

1. 节点

将地图上交叉路口表示成“节点”,这里的交叉路口可以是十字交叉路口、“T”型路口,例如图 6-8 所示

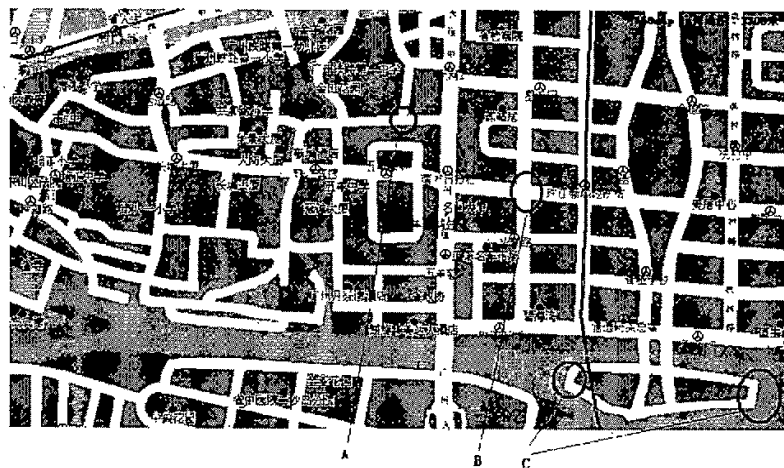


图 6-8 道路交通图

Fig. 6-8 Traffic path

图 6-8 中 A 点为“T”型路口; B 点为十字交叉路口, A, B 定义为节点 A 和节点 B。C 点虽然也是转弯的地方,但在 C 点并没有分岔口,因此 C 点不能看成节点。后台处理的并不是图 6.5 所示的图形文件,而是简单的节点和路径的折线图。

2. 路径

按照上述原则将电子地图上所有的节点标记出来, 然后根据实际电子交通图, 将连在一起的节点用直线或者折线连接, 构成类似图 6-5 所示简单节点路线图。为了保证实时性, 图中并不固定每一条路径的“旅行费用”, 只有当汽车驾驶员对路线开始查询时才通过 GPRS 网络向交通中心获取实时的交通数据, 这里的交通数据是已经处理后的节点之间的“旅行费用”, 不需要汽车上的车载终端进行换算。汽车上的车载终端将获得的旅行费用直接标注在每一条路线上, 以供接下来的路线寻优使用。

6.5 电子地图的路线显示

经过上面一系列的步骤得到了一条最优路径, 接下来的工作就是将路线显示在汽车控制台上, 为了显示画面生动需要将所获得的节点折线图转换成电子地图上的路线, 如图 6-9 所示:

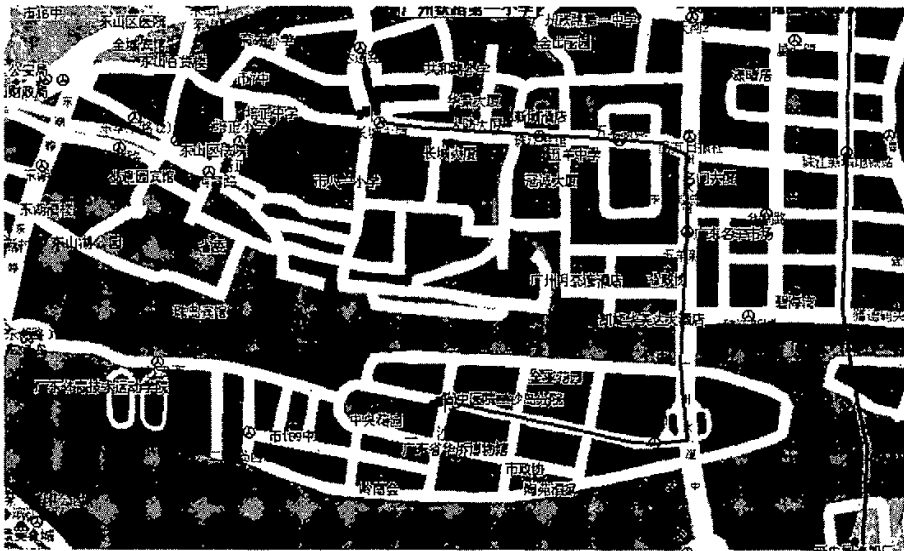


图 6-9 交通路径图

Fig. 6-9 Traffic path

上面的图是事先输入汽车的车载终端内的, 显示动态路径时候根据上面算法得到的最优路径平移到电子地图上并且画出一条路径, 这就是汽车的行驶曲线。

6.6 本章小结

本应用是以改进进化策略的思想为指导, 虽然变异的过程并不严格满足变异因子的随机性, 但是进化策略的思想在这里得到了较好的应用, 能够很好的解决路径优化问题, 这

是进化策略在导航系统中的第一次尝试性应用。

车载终端的产品已经飞速的发展,利用 GPS 导航系统可以高效、及时的更新路况数据,给人们的生活带来极大的方便。正如人们通过发明罗盘拉开了航海时代的帷幕使其后的世界焕然一新那样, GPS 导航系统这一现代的罗盘。将为我们的机动化社会带来巨大的变化。

结 论

在自然科学、工程设计与现代管理中提出了许多大型复杂的全局优化问题，有效的全局优化方法已经成为影响这些领域发展的关键之一。进化策略算法恰恰满足了这类问题的需要，逐渐成为研究的热点。

本文提出的基于极坐标方向寻优的进化策略，改进了传统进化策略在收敛速度和收敛结果方面的不足，充分考虑并利用了父代提供的方向指引，实现了更快、更好的收敛特性。

本文的论述是针对初始种群规模为“1”来研究的，这只是为了叙述和证明方便，通常情况下，为了达到更好的收敛速度和更加稳定的全局收敛能力，种群的初始规模应该适当增加。传统的进化策略为了避免出现错过全局最优解，都是采用较大规模的初始种群，分别进化然后最终对比判断得出全局最优解的。在比较严格的情况下，建议初始种群的规模不小于“3”。

现有的研究让我们看到进化策略在解决多目标优化问题中大有可为，但是我们看到这里还有很多问题有待进一步研究，总结起来有以下几点：

- 1) 改进后的进化策略的收敛速度比传统进化策略快 2 倍。虽然大量的实例已经说明上述问题，有的实例甚至是以 100 倍的进化速度收敛。但是数学理论上的证明还没有完成。
- 2) 进化策略的变异因子对收敛速度有明显的作用，变异因子也是提高收敛速度的一个较好的入手点。
- 3) 在解决实际问题时，如何准确确定测度函数有待研究。
- 4) 运用进化策略解决实际问题有待进一步研究。

参考文献

- [1] 余有明等,进化计算的理论与算法,计算机应用研究 2005.09
- [2] Salomen R. Evolutionary Algorithms and Gradient Search: Similarities and Difference. IEEE Trans on Evolutionary Computation,1998,2(2):45~55
- [3] 邢文训, 谢金星.现代优化计算方法.北京:清华大学出版社,1999.
- [4] 王凌.智能优化算法及其应用.北京:清华大学出版社, 2001.
- [5] Thomas Back, Fogel, D.B., Zbigniew Michalewicz, Evolutionary Computation I Basic Algorithms and Operators, Bristol and Philadelphia: Institute of Physics Publishing, 2000.
- [6] Darwin,物种起源 (舒德干等), 西安: 陕西人民出版社, 2001 年.
- [7] 遗传学编写组. 遗传学. 北京: 中国大百科全书出版社, 1983 年.
- [8] Friedberg R.M., A learning machine: part IBM J.2 1958,2-13.
- [9] Bremermann H.J., Optimization through evolution and recombination Self-Organizing Systems, Washington, DC: Spartan, 1962.
- [10] Rechenberg, I., Cybernetic Solution Path of an Experimental Problem,Royal Aircraft Establishment Library Translation 1965,1122.
- [11] Schwefel,H-P., Kybernetische Evolution als Strategie der experimentellen Forschung in der Stromungstechnik Dipl.-Ing. Thesis,Technical University Berlin, 1965.
- [12] Bienert, P. aufbsu einer Optimierungsautomatik for drei parameter Dipl.-Ing. Thesis, Technical University Berlin, 1967.
- [13] Rechenberg, I., Evolutionsstrategie:Optimierung trchnischer Systeme nach Prinzipien der biologischen Evolution.Stuttgart: Frommann-Holzboog,1973.
- [14] Schwefel, H-P., Numerical Optimization of Computer Models,Chichester: Wiley, 1981.
- [15] 王云诚,唐焕文 单峰函数最优化问题的进化策略 计算数学 2000.11
- [16] 徐宗本, 李国, 解全局优化问题的仿生类算法(I)一模拟进化算法, 运筹学杂志, 14:2 (1995),1-13.
- [17] 欧阳柳波,李学勇,吴克寿 基于进化策略的抽题算法设计 计算技术与自动化 2004.3
- [18] 王战权,赵朝义,云庆夏 进化策略中基于柯西分布的变异算子改进探讨 系统工程 1999.7
- [19] 王楠 变权重多目标进化算法研究 当代经济管理 2005.12
- [20] 于建伟 多目标进化算法研究综述 海南大学学报自然科学版 2005.12
- [21] Beyer, Hans-Georg, The Theory of Evolution Strategies, Berlin Heidelberg: Springer-Verlag, 2001.
- [22] 李宏,唐焕文,郭崇慧 一类进化策略的收敛性分析 运筹学学报 1999.12
- [23] 徐宗本, 李国.解全局优化问题的仿生类算法(I)一模拟进化算法.运筹学杂志, Vol.14,No.2,1995
- [24] David B.Fogel, Asymptotic convergence properties of genetic ary programming: analysis and experiments.Cybernetics and Journal,25:389-407,1994
- [25] 梁之舜等.概率论及数理统计, 高等教育出版社, 1988
- [26] Bian Zengyuan, Yu Yongquan, Zeng Bi, WangMinghui, Mao Huifeng "A Novel Evolution Strategy

Algorithm Based On the Selected Direction By the Polar Coordinates",1st International Symposium on Systems and Control in Aerospace and Astronautics Harbn,China January 19-21,2006

- [27] 孙小荣, 徐爱功, 刘玉华 车辆导航中一种改进的路径优化算法 辽宁工程技术大学学报 2005.4
- [28] 张歆奕,吴今培,张其善 车载导航仪中路径规划算法及其实现 2001.9
- [29] 王宗原,郝燕玲,徐兆新 基于电子地图的路径规划的分析解决方案 应用科技 2006.2
- [30] 仲欣, 吕恬生 基于遗传算法的汽车式移动机器人路径规划方法 上海交通大学学报 1999.7
- [31] Yifeng Zhang,Zhenya He,; Chengjian Wei, Luxi Yan."A self-organizing evolvi- ng algorithm combined with a transient chaotic neural network".Intelligent Multimedia, Video and Speech Processing, 2001. Proceedings of 2001 International Symposium on , 2-4 May 200 Pages:239 - 242
- [32] Dellaert F.,and Beer R.D., "Toward an evolvable model of development for autonomous agent synthesis",Artificial Life IV,Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems,edited by Brooks R.and Maes P.,MIT Press,Cambridge,MA,USA,1994
- [33] Back,T."Optimal Mutation Rates in Genetic Search".In S.Forrest(Ed).Fifth International Conference on Genetic Algorithms.(pp.1-9).San teo,CA;Morgan Kaufmann.1993.
- [34] Qing Zhou, Yanda Li "Directed variation in evolution strategies"Evolutionary Computation, IEEE Transactions on , Volume: 7 , Issue: 4 , Aug. 2003 Pages:356 - 366
- [35] Sang-HwanLee; Hyo-ByungJun; Kwee- Bo Sim; "Performance improvement of evolution strategies using reinforcement learning"Fuzzy Systems Conference Proceedings, 1999. FUZZ-IEEE '99. 1999 IEEE International , Volume: 2 , 22-25 Aug. 1999 Pages:639 - 644 vol.2
- [36] Krasnogor N."Genetic Algorithms for Protein Folding Problem,a Critical View".Proc Engr of Intell Sys,1998,98: 345-352.
- [37] Kampen A H,Buydens"The Ineffectiven- ess of recombination in a Genetic Algori- thm for the Structure Etucidation of a Heptapeptide in Torsion Angle Space": a Comparison to Simulated Annealin g.Chem&Intell lab Sys,1997,36:141-152

攻读学位期间发表的论文

1. 边增远、刘建伟、曾碧, PIC 汇编编程技巧, 顺德职业技术学院学报 2004. 2
2. 边增远, 曾碧, 单片机多中断处理技术研究, 单片机技术应用 2004. 5
3. 边增远、曾碧, 模糊控制实现万能充电器, 电子技术 2004. 12
4. 边增远、曾碧、鞠仪静、李雷, 一种通用的 LCD 显示屏驱动程序, 电子世界 2005. 2
5. 边增远、曾碧、陈海金、李雷, Lonwork 网络实现安全防盗门, 电子世界 2005. 6
6. Bian Zengyuan, Yu Yongquan, Zeng Bi, A Novel Evolution Strategy Algorithm Based On the Selected Direction By the Polar Coordinates, isscaa 2006 1st international Symposium on Systems and Control in Aerospace and Astronautics IEEE 收录 2006. 1. 19
7. Jie qionghan, Bi Zeng, GuangChang Liu Zengyuan Bian Extensional Methods and Application in the Robot Navigation, isscaa 2006 1st international Symposium on Systems and Control in Aerospace and Astronautics IEEE 收录 2006. 1. 19
8. 刘建伟, 边增远, 基于神经元芯片的远程水温监控系统, 单片机与嵌入式系统应用 2005. 8

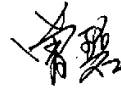
独创性声明

秉承学校严谨的学风与优良的科学道德，本人声明所呈交的论文是我个人在导师的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的地方外，论文中不包含其他人已经发表或撰写过的研究成果，不包含本人或其他用途使用过的成果。与我一同工作的同志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。

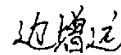
本学位论文成果是本人在广东工业大学读书期间在导师的指导下取得的，论文成果归广东工业大学所有。

申请学位论文与资料若有不实之处，本人承担一切相关责任，特此声明。

指导教师签字：



论文作者签字：



2006年5月26日

学位论文版权使用授权书

本学位论文作者完全了解 有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印、或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后适应本授权书）

学位论文作者签名：边增远

签字日期：2006年5月26日

导师签名：



签字日期：2006年5月26日

学位论文作者毕业后去向：

工作单位：

通讯地址：

电话：

邮编：

致 谢

本文是在我的导师曾碧教授和余永权教授的悉心指导下完成的。三年来,导师曾碧教授在各方面给予了我很大的帮助。在科研和学习方面,曾碧教授不仅在实验室为我提供最好的科研条件,同时也从项目筹备、项目实施、论文发表等方面身体力行地指导我,为我提供了非常难得的学习和见识的机会;在生活方面,曾碧教授给予了我无微不至的关心。

同时,还要衷心感谢三年来给我许多帮助和指导的余永权教授。余老师无论在学术上还是在生活上都给了我无微不至的关怀,点点滴滴我都将铭记在心。

同时,我也要感谢计算机学院何振炎书记、林伟副教授、汤荣江副教授、汪明慧老师、彭海霞老师、张灵老师、谢光强老师、朱清华老师、陈云华老师、曾安老师,他们在学习和生活上都给了我很多帮助,使我受益颇多。

感谢智能工程研究所王增强、叶伟琼、林勇、刘志煌、陈智斌、鲁庆、刘建伟、陈海金、陈贺青、李雷、鞠仪静、韩洁琼、朱钦华、李治龙、李磊、张衡、帅知春以及 05 师弟师妹给予我的热情帮助和支持。

三年的学习生活是让人难忘的,和教研室的老师同学一起度过的美好的时光留在了我的心中。再一次感谢各位师长的关心和爱护!

感谢我敬爱的父母给予我博大无私的关爱与支持,感谢他们辛劳一生供我上学,感谢他们因我的离家求学度过的无数寂寞的日子!感谢我的女朋友给予我的鼓励和关怀!

最后感谢支持关心我的老师、同学、朋友、亲人以及所有支持关心我的人!向所有帮助过我的老师、同学、朋友和家人致谢!

附录 1

实例 1 部分数据:

传统进化策略相关数据:

有效点的函数值:

27.224 20.812 19.706 17.833 17.324 15.985 15.508 15.053 15.032 15.016 14.912 14.843 14.836
14.818 14.817 14.816

x 值:

6.571 6.6005 7.1001 8.5456 7.827 9.1935 9.1611 9.842 9.8248 9.824 9.8197 9.8306 9.8356
9.8327 9.8335 9.8347

y 值:

5.4145 5.3528 5.1346 5.3309 5.3365 5.154 5.3485 5.148 5.3583 5.3578 5.3505 5.3474 5.3475
5.3506 5.3499 5.3499

一共迭代次数: 9768

有效进化代数: 16

相邻函数函数误差: 0.0012943

有效点的函数值:

24.048 23.853 22.112 21.758 21.695 21.53 21.31 20.995 20.941 20.869 20.753 20.596 20.587
20.547 20.383 20.26 20.177 19.899 19.79 19.641 19.576 19.507 19.503 19.501 19.5 19.486
19.242 18.922 18.846 18.824 18.819 17.264 17.224 17.143 17.117 16.964 16.821 16.819 16.375
16.368 16.354 16.205 16.02 15.97 15.781 15.724 15.58 15.519 15.501 15.496 15.483 15.483
15.078 15.017 15.016 15.016 14.87 14.84 14.821 14.82 14.817 14.816

有效进化点转换成直角坐标情况后的 x 值:

3.0498 3.0454 3.1179 3.1538 3.1724 3.7704 3.8653 3.8578 3.8094 3.8185 4.4748 4.4823 4.4793
4.4788 5.1186 5.1738 5.224 5.2086 5.2028 5.1952 5.1792 5.1793 5.1708 5.1708 5.1601 5.1664
5.8214 5.8554 5.845 5.841 5.838 7.8674 7.8677 7.8043 7.8194 7.838 7.8386 7.8354 8.4946
8.5081 8.5032 8.5027 9.1652 9.1654 9.1785 9.178 9.1615 9.1703 9.166 9.1725 9.1675 9.1685
9.8298 9.836 9.8352 9.8338 9.8259 9.8278 9.8329 9.8314 9.8345 9.8343

有效进化点转换成直角坐标情况后的 y 值:

5.135 5.1399 5.1549 5.1541 5.1524 5.3487 5.3618 5.344 5.3514 5.3502 5.1402 5.1421 5.143
5.1443 5.1429 5.1653 5.3485 5.3456 5.3461 5.3504 5.3451 5.3502 5.3529 5.3475 5.3507 5.3492
5.3382 5.351 5.3486 5.3502 5.3496 5.3447 5.3466 5.3484 5.341 5.3576 5.35 5.3513 5.1484
5.1494 5.1513 5.3548 5.1617 5.1608 5.1456 5.1503 5.3557 5.3466 5.3527 5.3514 5.3497 5.3503
5.1456 5.15 5.1504 5.1503 5.3473 5.3489 5.3515 5.3502 5.3507 5.3499

一共迭代次数: 3470

有效进化代数: 62

相邻函数函数误差: 0.00042543

附录 2

实例1 传统进化策略源程序

```

function []=jhsf(x1,y1)
x1=rand*6.5+3.5;
y1=rand*0.25+5.25;
sigema=1;
count=0;
for i=1:10000;
    a(i)=0;
    b(i)=0;
    x(i)=0;
    y(i)=0;
    long_x(i)=0;
    long_y(i)=0;
end
i=1; j=1;
while sigema>=0.001 | count>=20
    x2=x1+randn;
    y2=y1+randn;
    kk=1;
    while x2>=10 | x2<=-3 | y2>=5.5 | y2<=-5
        x2=x1+randn/kk;
        y2=y1+randn/kk;
    end
    answer1=x1*sin(3*pi*x1)+y1*sin(10*pi*y1)+30;
    answer2=x2*sin(3*pi*x2)+y2*sin(10*pi*y2)+30;
    if answer2<answer1
        x1=x2;
        y1=y2;
        x(i)=x2;
        y(i)=y2;
        sigema=abs(answer1-answer2);
        b(i)=answer2;
        i=i+1;
    end
    count=count+1;
    a(count)=answer2;
    if mod(count,3)==0
        long_x(j)=x2;
        long_y(j)=y2;
        j=j+1;
    end
end
a=1;

```

附录 3

实例1 改进进化策略源程序

```
function []=jhsf(r,q)
r1=rand*13; q1=rand*1.57;
while r1*cos(q1)>=13 | r1*cos(q1)<=0 | r1*sin(q1)>=0.5 | r1*sin(q1)<=0
    r1=rand*13;
    q1=rand*1.57;
end
r2=rand*13; q2=rand*1.57;
while r2*cos(q2)>=13 | r2*cos(q2)<=0 | r2*sin(q2)>=0.5 | r2*sin(q2)<=0
    r2=rand*13;
    q2=rand*1.57;
end
sigema=1; count=0;
pi=3.14159;
while sigema>=0.01
    answer1=(r1*cos(q1)-3)*sin(3*pi*(r1*cos(q1)-3))+(5+r1*sin(q1))*sin(10*pi*(5+r1*sin(q1)))+30;
    answer2=(r2*cos(q2)-3)*sin(3*pi*(r2*cos(q2)-3))+(5+r2*sin(q2))*sin(10*pi*(5+r2*sin(q2)))+30;
    if answer1>answer2
        r=r2;
        q=q2;
        r2=r+randn;
        q2=(q+q1)/2;
        kk=1;
        while r2*cos(q2)>=13 | r2*cos(q2)<=0 | r2*sin(q2)>=0.5 | r2*sin(q2)<=0
            r2=r+randn/(2*kk);
            q2=(q+q1)/2;
            kk=kk+1;
        end
        q1=q;
        r1=r;
        sigema=abs(answer1-answer2);
    end
    r2=r1+randn;
    q2=q1+randn;
    kk=1;
    while r2*cos(q2)>=13 | r2*cos(q2)<=0 | r2*sin(q2)>=0.5 | r2*sin(q2)<=0
        r2=r1+randn/(kk*5);
        q2=q1+randn/(kk*10);
        kk=kk+1;
        x=r2*cos(q2);
        y=r2*sin(q2)
    end
    count=count+1;
end
```