广西民族大学

硕士学位论文

进化策略在数值计算中的一些应用研究

姓名：夏慧明

申请学位级别：硕士

专业：计算数学

指导教师：周永权

20080401

进化策略在数值计算中的一些应用研究

摘 要

进化计算是模拟生物在自然环境中的遗传进化机制形成的一种自适应全局优化搜索算法，与传统的数值方法不同，人们愈来愈注重了对进化计算的研究，并把它作为解决问题的一种新型方法，该方法在进化过程中通过重组、突变、选择对个体进行训练学习，向最优解逼近。数值方法是数学的一个分支，它的研究对象是利用计算机求解各种数学问题的数值方法。内容包括函数的数值逼近（插值与拟合），数值积分与数值微分，线性、非线性方程（组）的数值解法，常微分方程组及偏微分方程组的数值解法等。它们在 19 世纪，甚至更早一些时候就已经建立。现代计算机的出现为大规模的数值计算创造了条件，集中而系统地研究适用于计算机的数值方法变得十分迫切和必要。本文将主要采用进化策略来研究数值计算，把数值计算问题转化为函数优化问题以达到解决数值计算问题的目的，同时还将进化策略与差分演化、泛函网络等方法结合，提出新的解决数值问题的混合算法。将该方法分别应用于计算矩阵特征值、特征向量；化学方程式配平；求解复函数方程根等问题。这些问题虽用传统的数值方法可以解决，但存在着初值选取敏感、收敛速度慢、计算精度低、甚至不收敛等缺点。

针对目前传统的数值方法存在的一些问题，本文的主要工作是利用进化策略算法并行搜索、全局收敛、鲁棒性等特性来弥补传统的数值方法存在的不足，同时将进化策略算法与差分演化算法、泛函网络相结合发挥各自的优点，这在处理某些问题时能起到事半功倍的效果。因此用进化策略、差分演化算法、泛函网络来研究数值计算，有较高的理论价值和实际意义。

关键词：数值计算　智能算法　进化策略算法　差分演化算法　泛函网络　改进进化策略　混合进化策略　突变算子

APPLICATION RESEARCH ON NUMERICAL METHODS BASED ON EVOLUTION STRATEGY

ABSTRACT

Evolution computing is an auto-adapted global optimize search algorithms which simulates the genetic and forms of the natural environment. The method is different from the traditional numerical method. People have realized increasingly that the research about soft computing are very important and take it to be a novel method which settles the problems，this method studies the individuals by reorganization, mutation, selection in the evolution process and approaches to the optimal solution.The numerical method is a mathematics branch, whose object of study is to solve the numerical methods of each mathematics questions by using the computer. The content includes digital approximation (interpolation and fitting), numerical integration and numerical differentiation, numerical solution of linear equation(group) function, numerical solution of ordinary differential equation group and partial differential equation group and so on. They had been proposed in 1990s, even earlier. The modern computers have created conditions for large scale numerical computing, it is urgent and necessary to research assembly and systematic which suit computer numerical method. In this object uses Evolution Strategy to research numerical computing, in order to settle the numerical computing problems transfer the numerical computing problems to functional optimization problems and proposed new hybrid algorithm by combining the three intelligent algorithms: Evolution Strategy, Differential Evolution Algorithm and Functional Networks at the same time. Using the method to solve the matrix’s eigenvalues and eigenvectors; balance chemical equations; compute the complex functions and so on. These problems can be settled by the traditional numerical method but there exist some disadvantage such as

select of the initial value sensitively, the speed convergent slowly, the accuracy lowly, even not convergent and so on.

In view of questions about traditional numerical methods, prime task of this article uses the characteristics of Evolution Strategy, such as, parallel search, global convergence and robustness and so on, to solve the problems of traditional numerical methods. Let the three methods Evolution Strategy, Differential Evolution Algorithm and Functional Networks be combined with each other and exert each advantages, it can obtain best answer when deal with some problems. So, researching numerical computing by Evolution Strategy, Differential Evolution Algorithm and Functional Networks have higher theory value and practical significance.

KEYWORDS: numerical computing; intelligence algorithm; evolution strategy algorithm; differential evolution algorithm; functional networks; improved evolution strategy; hybrid evolution strategy; mutation operator

论文独创性声明

本人郑重声明：所提交的学位论文，是本人在导师的指导下，独立撰写完成的。除文中已经注明引用的内容外，本论文不含其他个人或其他机构已经发表或撰写过的研究成果，也没有剽窃、抄袭等违反学术道德规范的侵权行为。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人愿意承担由本声明而引起的法律责任。

研究生签名：　　日期：　　年　　月　　日

论文使用授权声明

本人完全了解广西民族大学有关保留、使用学位论文的规定。学校有权保留并向国家有关部门或机构送交学位论文的复印件和电子文档，可以采用影印、缩印或其他复制手段保存、汇编学位论文。除在保密期内的保密论文外，允许学位论文被查阅和借阅，可以公布（包括刊登）论文的全部或部分内容。

研究生签名：　　日期：　　年　　月　　日

导 师 签 名：　　日期：　　年　　月　　日

1 绪论

1.1 数值方法研究现状与进展

在科学和工程计算中，求解矩阵的特征值及特征向量，是最普遍的问题之一。在许多应用领域，经常使用矩阵的特征值及特征向量，如主成分分析、因子分析、典型相关分析都必须计算相关矩阵的特征值和相应的特征向量。目前，关于特征值、特征向量问题的数值解法有两种：变换法和迭代法。其中，变换法是直接对矩阵进行处理，通过一系列变换，使之变成比较容易求解特征值、特征向量的新矩阵，但是变换法存在存贮量大，计算速度慢等特点；而迭代法是通过一系列矩阵向量乘积而求得特征值和特征向量，常用的方法有：子空间迭代法[1]、Lanczos方法[1]以及 Davidson 方法[2]。虽这些方法在求解矩阵的特征值及特征向量问题中都取得了巨大的成功，但普遍存在着精度低、收敛速度慢及泛化能力弱等缺陷。1993 年 C.Bischof 等提出了不变子空间分解算法求解特征值、特征向量[3]。1999 年罗晓广等提出了利用并行分治算法求解非对称矩阵的特征值[4]。2002 年 Michael Grant 提出了利用 Excel 求解特征向量[5]。2004 年郑敏玲提出了利用并行算法来求解矩阵的特征值[6]。2006 年杨延俊提出了新的同步求解法来求解矩阵的特征值与特征向量[7]。2007 年郭建敏提出了特征值与特征向量同步求解法[8]。

化学反应的计量关系是进行定量分析的基础，配平化学方程式是化学工作者必须具备的基本素质，然而，化学反应成千上万，类型繁多，各具特点，这给配平化学方程式带来困难。不少人对化学方程式的配平进行了广泛的探讨，并根据各类化学反应的特点创造了许多方法[9–10]，常见的有氧化值法、电子法、离子电子法、观察法等。但不论是什么类型的化学反应一定都遵循质量守恒定律。1999 年王进贤等提出了应用代数法配平有机氧化还原反应方程式[11]。2001年吕申壮提出了在WORD中添加“化学方程式自动配平和化学式格式化”工具[12]。2001年孙会霞等提出了利用改进的线性代数法配平化学方程式[13]。2003年杨海文等提出了矩阵解法配平化学方程式[14]。2007年张杨提出了刍议氧化还原方程式反应的配平方法[15]。

在大量的科学和工程计算中都要涉及复函数方程求根问题，研究复函数方程求根的算法有着重要的理论意义和应用价值。求复函数方程根的算法有多种，常用的有迭代法、牛顿法、下山法、转换法、圆盘迭代法等等。但是上述各种算法都存在着一定的局限性。对于迭代法、牛顿法存在收敛性和收敛速度问题，对迭

代初值的选取比较敏感；圆盘迭代法计算很复杂；下山法方法简单，但存在着局部极值问题；转换法是将一般方程求根的问题转换为多项式方程的求根问题，但是，该方法在转换计算多项式方程的系数的过程中要计算多个积分，所求根的精度也不高。现今一些学者提出了利用智能算法来进行复函数方程的求解。1999年，王兰生提出了利用串行和并行算法求解复函数方程的根[16]。2001 年 Lars Kindermann 等提出了利用神经网络来求解复函数方程的解[17]。2001 年陈国良等提出了利用遗传算法来求解复函数方程的根[18]，2002 年他们又提出了一种遗传算法求解方程的根[19]，2004 年他们再次提出了一种利用并行遗传算法求复函数方程的根[20]。

最优化理论是数学的一个分支，它研究的是某些数学问题的最优解，即对给出的实际问题，从众多候选方案中找到最优方案，它具有高度的应用性和技术性等特点。在处理一般单峰函数时二进制蚁群算法有着良好的表现，但是在多峰函数优化问题上，却无法同时找到所有的极值点，但是事实上，无论是数学上还是工程中的许多问题在本质上还是多峰问题，如复杂系统参数及结构辨识问题等。1997 年 Rainer S 等提出了利用差分进化来解决规定区域内的全局最优解[21]。1998 年 Hyun-Kyo Jung 等提出了利用免疫算法与其它启发式算法求解优化问题[22]。2006 年王柳毅等提出了一种利用并行二进制蚁群算法来对多峰值函数进行优化[23]。2006 年蒋忠樟等提出利用演化算法来求解多峰函数优化问题[24]。2007 年靳宗信等提出了一种利用改进的自适应克隆选择算法对多峰值函数进行优化[25]。

函数逼近技术在系统辨识、最优控制等领域有着广泛的应用。对函数逼近技术的研究，已经提出了许多成熟的逼近方法，例如代数多项式逼近、三角多项式逼近、插值方法逼近、样条函数逼近等[26–28]。经典的方法通常是采用回归分析法，即已知样本数据集合，构造满足采样数据点误差等约束条件的函数，但存在着相关却不同的领域问题，如在控制等领域，需要求解特定模型，而该模型是否适当，只有模拟求解该模型后依据其适应性才能评判。回归分析并不适合于求解该类问题，其根本原因在于所采用的样本数据来源往往受到专家领域知识的限制，导致所求解的特定模型与实际模型出现误差，甚至产生错误的结果。机器学习方法的发展为解决该问题提出了一个新的思路，它们不仅可以用于数据回归，而且可用于自适应建模。无论是从函数逼近的方法，还是工程逼近方法，这些都拓展了逼近技术的应用领域。如今在理论上已经证明了，人工神经网络(ANN)可以逼近任意连续函数[29–31]。1989 年 Hornik K 等提出了将多层反馈神经网络用于多维函数逼近[32]。1997 年 Ahmed Moataz A 等将遗传算法用于函数逼近问题中，并取得了较好的结果[33]。1997 年 Jean Gabriel Attali 等提出一种用于函数逼

近问题的新算法[31]。2000 年荔建琦等提出了多维函数的进化逼近[34]。2000 年卢宏涛等提出了基于神经网络的多维离散傅立叶变换函数逼近[35]。2005 年周永权等提出了基于泛函网络的多维函数逼近理论及其学习算法[36]。

1.2 论文的创新性

传统的数值计算方法在求解某些特定数值问题时存在很多不足。本文创新点是：

(1)基于进化策略提出了求解某些数值问题的新算法；

(2)对所需解决的数值问题进行分析，对进化策略算法中的变异个体、变异方式进行改进，这样可使得所求解的结果优于传统的变异策略；

(3)就一些智能算法，如：进化策略、差分演化算法、泛函网络等，综合评价他们的优缺点，取长补短，提出混合算法，这样对于一些特定问题可以提高求解精度、收敛速度。

运用这些算法来研究数值计算，可把数值计算问题转化为函数优化问题以达到解决数值计算问题的目的，本文所提到的这些算法的特性能有效的解决一些数值问题。

1.3 论文工作的重点和难点

本文重点和难点是采用进化策略来研究数值计算，把数值计算问题转化为函数优化问题以达到解决数值计算问题的目的，同时还将进化策略与差分演化、泛函网络等方法结合，提出新的解决数值问题的混合算法。将该方法分别应用于计算矩阵特征值、特征向量；化学方程式配平；求解复函数方程根等问题。并且克服这些问题用传统的数值方法可以解决，但存在着初值选取敏感、收敛速度慢、计算精度低、甚至不收敛等不足。

1.4 论文的主要工作

本文主要有以下几个方面的工作：

(1) 利用进化策略研究特征值、特征向量的计算，化学方程式配平，复函数方程求解。

(2) 进化策略与差分演化算法相结合来研究多峰值函数优化问题。

(3) 进化策略修正泛函网络基函数系数多维函数逼近。

论文的编排结构如下：

第一章主要对数值方法的发展历史和研究现状进行了总结与回顾，并列出本文的主要工作。

第二章主要对进化策略、差分演化算法、泛函网络的产生背景、发展、应用等作一介绍，并详细阐述算法的基本原理、进化过程和特征等。

第三章提出了一种利用标准进化策略算法求解矩阵特征值、特征向量的新算法。

第四、五章对进化策略算法作一定的改进，将经过改进后的进化策略算法应用在化学方程式的配平与复函数方程的求根中。

第六、七章将进化策略算法分别与差分演化算法、泛函网络相结合，提出了新的混合算法及模型，将混合算法应用在多峰值函数优化及多维函数逼近中。

第八章总结与展望。

1.5 本章小结

本章主要阐述数值计算现状与发展，论文的创新点、重难点以及主要工作。

2 智能优化算法

智能计算也有人称之为“软计算”，是受自然(生物界)规律的启迪，根据其原理，模仿求解问题的算法。从自然界得到启迪，模仿其结构进行发明创造，这就是仿生学。这是我们向自然界学习的一个方面；另一方面，我们还可以利用仿生原理进行设计(包括设计算法)，这就是智能计算的思想。这方面的内容很多，如人工神经网络技术、进化策略算法、模拟退火算法、模拟退火技术和群集智能技术等。

2.1 智能优化算法的概述

智能优化算法要解决的一般是最优化问题。最优化问题可以分为(1)求解一个函数中，使得函数值最小的自变量取值的函数优化问题；(2)在一个解空间里面，寻找最优解，使目标函数值最小的组合优化问题。典型的组合优化问题有：旅行商问题(Traveling Salesman Problem,TSP),加工调度问题(Scheduling Problem)，0－1 背包问题(Knapsack Problem)，以及装箱问题(Bin Packing Problem)等。

优化算法有很多，经典算法包括：线性规划、动态规划等；改进型局部搜索算法包括爬山法、最速下降法；模拟退火、进化策略算法、神经网络、泛函网络、系统动态演化方法等。优化思想里面经常提到邻域函数，它的作用是指出如何由当前解得到一个(组)新解，其具体实现方式要根据具体问题分析来定。一般而言，局部搜索就是基于贪婪思想利用邻域函数进行搜索，若找到一个比现有值更优的解就弃前者而取后者。但是，它一般只可以得到“局部极小解”。而模拟退火、进化策略算法、神经网络等从不同的角度和策略实现了改进，取得较好的“全局最小解”。

模拟退火、进化策略算法、神经网络在解决全局最优解的问题上有着独到的优点，并且，它们有一个共同的特点：都是模拟了自然过程。模拟退火思路源于物理学中固体物质的退火过程，进化策略算法借鉴了自然界优胜劣汰的进化思想，神经网络更是直接模拟了人脑。它们之间的联系也非常紧密，比如模拟退火和进化策略算法为神经网络更优良的学习算法提供了思路。把它们有机地综合在一起，取长补短，性能将更加优良。这几种智能算法有别于一般的按照图灵机进行精确计算的程序，尤其是人工神经网络，是对计算机模型的一种新的诠释，跳出了冯·诺依曼机的圈子，按照这种思想来设计的计算机有着广阔的发展前景。

2.2 进化策略

2.2.1 进化策略概述与进展

2.2.1.1 进化策略

20 世纪 60 年代，柏林工业大学的 I.Rechenberg 和 H.P.Schwefel 等在进行风洞实验时，由于设计中描述物体形状的参数难以用传统方法进行优化，因而利用生物变异的思想来随机改变参数值，获得了较好的结果。 随后，他们对这种方法进行了深入的研究和发展，形成了一种新的进化计算方法——进化策略(Evolution Strategies,简称 ES)[37−38]。ES 是专门为求解参数优化问题而设计的，而且在 ES 算法中引进了自适应机制。隐含并行性和群体全局搜索性是它的两个显著特征，而且具有较强的鲁棒性，对于一些复杂的非线性系统求解具有独特的优越性能。目前，这种算法已广泛应用在各种优化问题的处理中，如神经网络的训练与设计、系统识别、机器人控制和机器学习等领域。

2.2.1.2 进化策略的发展过程

1． $(1+1)-ES$

1963 年，I.Rechenberg 最早提出的这种优化算法只有一个个体，并由此衍生同样仅为一个的下一代新个体，故称为 $(1+1)-ES$。进化策略中的个体用传统的十进制实型数表示，即：

$$X^{t+1}=X^{t}+N(0,\sigma) \tag{2.1}$$

式中 X^{t}——第 t 代个体的数值。

$N(0,\sigma)$——服从正态分布的随机数，其均值为 0，标准差为 σ。

因此，进化策略中的个体含有两个变量，为二元组 $\langle X,\sigma\rangle$。新个体的 X^{t+1} 是在旧个体 X^{t} 的基础上添加一个独立随机变量 $N(0,\sigma)$。假若新个体的适应度优于旧个体，则用新个体代替旧个体；否则，舍弃性能欠佳的新个体，重新产生下一代新个体。在进化策略中，个体的这种进化方式称作突变。$(1+1)-ES$ 仅仅使用一个个体，进化操作只有突变一种，亦即用独立的随机变量修正旧个体，以求提高个体素质。显然，这是最简单的进化策略。

2． $(\mu+1)-ES$

早期的(1+1)—ES，没有体现群体的作用，只是单个个体在进化，具有明显的局限性。随后，I.Rechenberg 又提出 $(\mu+1)-ES$，在这种进化策略中，父代有 μ 个个体 $(\mu>1)$，并且引入重组算子，使父代个体组合出新的个体。在执行重组时，从 μ 个父代个体中用随机的方法任选两个个体：

$$\begin{aligned}(X^{1},\sigma^{1})&=((x_1^1,x_2^1,\cdots,x_n^1),(\sigma_1^1,\sigma_2^1,\cdots,\sigma_n^1))\\(X^{2},\sigma^{2})&=((x_1^2,x_2^2,\cdots,x_n^2),(\sigma_1^2,\sigma_2^2,\cdots,\sigma_n^2))\end{aligned} \tag{2.2}$$

然后从这两个个体中组合出如下新个体：

$$(X,\sigma)=((x_1^{q_1},x_2^{q_2},\cdots,x_n^{q_n}),(\sigma_1^{q_1},\sigma_2^{q_2},\cdots,\sigma_n^{q_n})) \tag{2.3}$$

式中$q_i=1$或2。它以相同的概率针对$i=1,2,\cdots,n$随机选取。

对重组产生的个体执行突变操作，突变方式及σ的调整与$(1+1)-ES$相同。

将突变后的个体与父代μ个个体相比较，若优于父代最差个体，则代替后者成为下一代μ个个体新成员；否则，重新执行重组和突变产生另一个个体。

显然，$(\mu+1)-ES$比$(1+1)-ES$有了明显的改进，为进化策略这种新的进化算法奠下良好的基础。

3．$(\mu+\lambda)-ES$及$(\mu,\lambda)-ES$

1975年，H.P.Schwefel首先提出$(\mu+\lambda)-ES$，随后又提出$(\mu,\lambda)-ES$。这两种进化策略都采用含有μ个个体的父代群体，并通过重组和突变产生λ个新个体。它们的差别仅仅在于下一代群体的组成上。$(\mu+\lambda)-ES$是在原有μ个个体及新产生的λ个新个体中(共$\mu+\lambda$个个体)再择优选择μ个个体作为下一代群体。$(\mu,\lambda)-ES$则是只在新产生的λ个新个体中择优选择μ个个体作为下一代群体，这时要求$\lambda>\mu$。总之，在选择子代新个体时若需要根据父代个体的优劣进行取舍，则使用“+”记号，如$(1+1)$、$(\mu+1)$及$(\mu+\lambda)$；否则，改用逗号分隔，如(μ,λ)。

近年来，$(\mu,\lambda)-ES$得到广泛的应用，这是由于这种进化策略使每个个体的寿命只有一代，更新进化很快，特别适合于目标函数有噪声干扰或优化程度明显受迭代次数影响的课题。

2.2.2 进化策略的基本原理

2.2.2.1 进化策略的基本思想

随机产生一个适用于所给问题环境的初始种群，即搜索空间，种群中的每个个体为实数编码，计算每个个体的适应值；依据达尔文的进化原则，选择遗传算子(重组、突变等)对种群不断进行迭代优化，直到在某一代上找到最优解或近似最优解。

2.2.2.2 进化策略的执行过程

1．确定问题的表达方式。这种表达式中个体由目标变量X和标准差σ两部分组成，每部分又可以有n个分量，即：

$$(X,\sigma)=((x_1,x_2,\cdots,x_i,\cdots,x_n),(\sigma_1,\sigma_2,\cdots,\sigma_i,\cdots,\sigma_n)) \tag{2.4}$$

X和σ之间的关系是：

$$\begin{cases}\sigma_i^{'}=\sigma_i\cdot\exp(r'\cdot N(0,1)+r\cdot N_i(0,1))\\ x_i^{'}=x_i+\sigma_i\cdot N_i(0,1)\end{cases} \tag{2.5}$$

式中：(x_i,σ_i)——父代个体的第i个分量；

$(x_i^{'},\sigma_i^{'})$——子代新个体的第i个分量；

$N(0,1)$——服从标准正态分布的随机数；

$N_i(0,1)$ ——针对第 i 个分量重新产生一次符合标准正态分布的随机数；

r' ——全局系数，等于 $(\sqrt{2\sqrt{n}})^{-1}$，常取 1；

r ——局部系数，等于 $(\sqrt{2n})^{-1}$，常取 1。

上式表明，新个体是在旧个体基础上随机变化而来。

2．随机生成初始群体，并计算其适应度。进化策略中初始群体由 μ 个个体组成，每个个体 (X,σ) 内又可以包含 n 个 x_i、σ_i 分量。产生初始个体的方法是随机生成。为了便于和传统的方法比较，可以从某个初始点 $(X(0),\sigma(0))$ 出发，通过多次突变产生 μ 个初始个体，该初始点可从可行域中用随机方法选取。初始个体的标准差 $\sigma(0)=3.0$。

3．计算初始个体的适应度，如若满足条件，终止；否则，往下进行。

4．根据进化策略，用下述操作产生新群体：

4.1)重组：将两个父代个体交换目标变量和标准差，产生新个体。一般目标变量采用离散重组，标准差采用中值重组。

离散重组：按照式子(2.2)和(2.3)对两个个体实行交叉组合。

中值重组：

从 μ 个父代个体中用随机的方法任选两个个体如式(2.2)，然后将父代个体各分量的平均值作为子代新个体的分量，构成的新个体为：

$$(X,\sigma)=(((x_1^1+x_1^2)/2,(x_2^1+x_2^2)/2,\cdots,(x_n^1+x_n^2)/2),$$
$$((\sigma_1^1+\sigma_1^2)/2,(\sigma_2^1+\sigma_2^2)/2,\cdots,(\sigma_n^1+\sigma_n^2)/2)))$$

4.2)突变：对重组后的个体添加随机量，按照式子(2.5)产生新个体。

4.3)计算新个体适应度。

4.4)选择：按照 (μ,λ) 选择策略，挑选优良个体组成下一代群体。

5．反复执行第 4 步，直到达到终止条件，选择最佳个体作为进化策略的结果。

图 2.2 所示进化策略的工作流程。图中 Gen 表示进化的代次，在第 0 代，根据问题的表达是二元组或是三元组方式，随机产生 μ 个初始个体，并计算它们的适应度。然后依次执行重组和突变操作，产生新个体。图中 j 统计新个体数目，重组和突变执行 λ 次，产生 λ 个新个体。随后计算新个体的适应度，再根据选择策略，从 $(\mu+\lambda)$ 个个体中或 λ 个新个体中选择出优秀的 μ 个个体组成新群体。

这样，便完成一代进化。重复这种进化过程，直至满足终止条件。

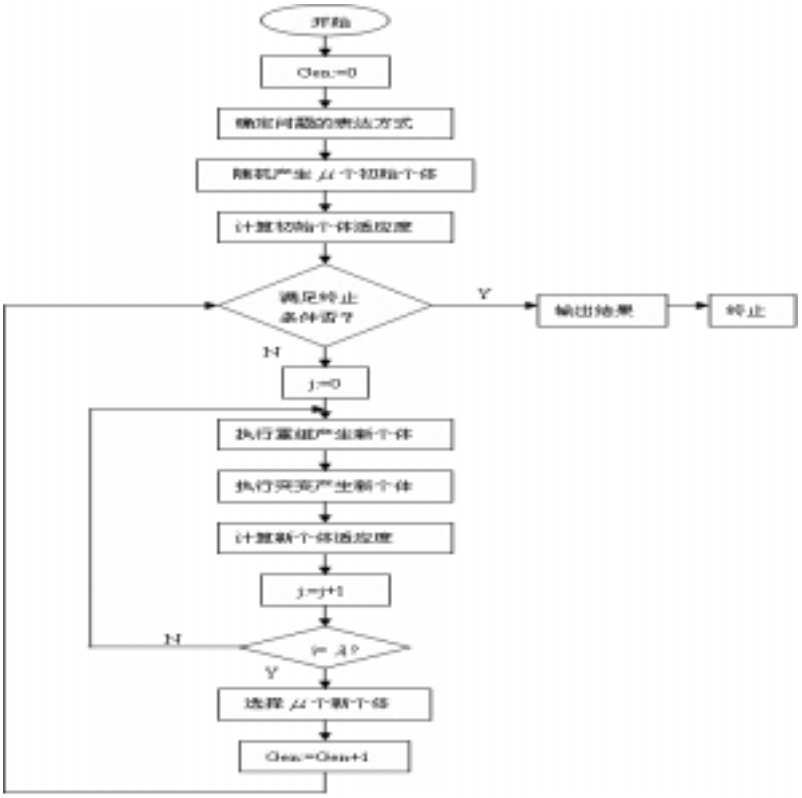

图 2.1 进化策略的工作流程

Fig.2.1 The process of evolution strategy

2.2.3 进化策略的特征

同遗传算法和遗传规划相比，进化策略主要具有下面几个特征：

(1)进化策略从 λ 个新个体或从 $(\mu+\lambda)$ 个个体中挑选 μ 个个体组成新群体，而且选择方法是确定型的。

(2)进化策略直接以问题的可行解作为个体的表现形式，无需再对个体进行编码处理，也无需再考虑随机扰动因素对个体的影响，这样就更便于其应用。

(3)进化策略以 n 维实数空间上的优化问题为主要处理对象。

2.3 差分演化算法

2.3.1 差分演化算法概述

差分演化算法是由 Rainer Storn 和 Kenneth Price[39–40]于 1996 年共同提出的一种采用浮点矢量编码在联想空间中进行随机搜索的优化算法[41–42]。其基本思想在于运用当前种群个体的差来重组得到中间种群，然后运用直接的父子混合个体适应值竞争得到新一代群体。由于其原理简单，受控系数少，易于理解和实现，实施随机、并行、直接的全局搜索。在日本召开的第一届国际进化优化计算竞赛(ICEO)中表现突出[43]，已成为进化算法的一个重要分支。

2.3.2 差分演化算法的基本原理

2.3.2.1 差分演化算法的基本思想

采用实数编码，并利用个体间的差分信息来指导新产生个体的搜索。在演化过程中，群体中的个体通过遗传算子产生新个体，保持群体的多样性，对新的空间进行搜索,以得到问题的最优解。

2.3.2.2 差分演化算法的执行过程

1.(初始化)确定种群规模 N ，杂交概率 $P_c \in [0,1)$ ；随机选取初始群体 $X(0)=(X_1(0),X_2(0),\cdots,X_N(0))$ ，并令 $X(0)$ 中的最优个体为 $X_{best}(0)$ ；置 $t:=0$ 。这里 $X_i(0)$ 为 n 维向量。

2. (种群演化) 对 $X(t)$ 中的每一个个体 $X_i(t)$ 作如下操作：

2.1 变异 令

$$V_i(t)=X_i(t)+\lambda[X_{best}(t)-X_i(t)]+\beta[X_{r2}(t)-X_{r1}(t)]$$
$$\triangleq(\overline{V}_1(t),\overline{V}_2(t),\cdots,\overline{V}_n(t))^T ,$$

其中 r_1 ， r_2 是 $[1,N]$ 中任意两个随机选取的不同整数， λ ， β 为可选参数, Δ 表示定义。

2.2 杂交 随机在 $[1,N]$ 中取整数 m ，并以概率 $p\{L=k\}=p_c^k$ 取 $[1,N)$ 中整数 L 。令

$U_i(t)=(\overline{u}_1,\overline{u}_2,\cdots,\overline{u}_n)^T$ ，其中对 $j=1,2,\cdots,N$ ，

如果 $j=<m>_n,<m+1>_n,\cdots,<m+L-1>_n$ ，则 $\overline{u}_j=\overline{V}_j$ ；否则 $\overline{u}_j=[X_i(t)]_j$ 。

这里 $<\bullet>_n$ 表示关于模 n 的模函数。

2.3 选择 令

如果 $J(U_i(t))>J(X_i(t))$ ，则 $X_i(t+1)=U_i(t)$ ；否则 $X_i(t+1)=X_i(t)$ 。

3. (终止检验) 令由步 2 所产生的新群体为

$$X(t+1)=(X_1(t+1),X_2(t+1),\cdots,X_N(t+1))$$

并记 $X(t+1)$ 中的最优个体为 $X_{best}(t+1)$ 。如果满足精度要求或整个进化已达到进化时限，则停机并输出 $X_{best}(t+1)$ 作为近似解，否则，置 $t:=1$ 转步 2。

2.3.3 差分演化算法的特征

差分演化算法是一种基于群体的启发式搜索算法，它利用群体中的个体在解空间中进行搜索，具有自适应、自学习、自组织和隐并行性等特点。在演化过程中，群体中的个体通过遗传算子产生新个体，保持群体的多样性，对新的空间进行搜索，以得到问题的最优解。它是一种快速的演化算法，采用实数编码，并利用个体间的差分信息来指导新产生个体的搜索。与其它算法相比，它具有结构简单、容易使用、快速和鲁棒性等特点。差分演化算法在求解全局最优化问题时会出现在演化前期收敛较快、后期收敛较慢的问题。

2.4 泛函网络

2.4.1 泛函网络概述

泛函网络[44]是 1998 年由 E.Castillo 提出的。泛函网络处理的是一般泛函模型。泛函网络的各神经元之间没有权值连接，神经元函数是未知的，在学习的过程中根据问题的需要从特定的函数族(多项式，三角函数，Fourier 展开式等)中选取。泛函网络已经被应用于非线性事件，序列预测及逼近，微分、差分和泛函方程的求解。泛函网络是对神经网络的有效推广，这不仅表现在神经网络可以解决的问题泛函网络同样可以解决，还表现在对于某些神经网络不能解决的问题泛函网络也可以解决，同时经过泛函网络训练优化后所得到的解精度高，这是以前人工神经网络的缺点和不足。

2.4.2 泛函网络的基本原理

2.4.2.1 泛函网络的基本思想

给定一组函数族，即学习基函数。通过对基函数的训练学习，来得到一组合适的基函数，然后通过求解方程组的解，确定网络参数的值，从而确定出网络的学习结果。

2.4.2.2 泛函网络的拓扑结构

泛函网络由以下元素组成：(1)输入单元层：这是输入数据的一层单元，输入单元以带有相应的名字的实心圆表示；(2)输出单元层：这是最后一层单元，它输出网络的结果数据，输出单元也是由带有相应的名字的实心圆表示；(3)一层或多层神经元或计算单元：每一个神经元是一个计算单元，它计算一组来自前一层神经元或输入单元的输入值，并给下一层神经元或输出单元提供一组输入数据。计算单元相互连接，每一个神经元的输出可以作为另一个神经元或输出单元数据的一部分，一旦给定输入值，输出便由神经元的类型确定，它由已知函数定义。例如，假设我们有一个神经元具有 s 个输入 $(x_1, x_2, \cdots, x_s)$ 及 k 个函数 $F_j, j = 1,2,\cdots,k$，使得 $y_j = F_j(x_1, x_2, \cdots, x_s), j = 1,2,\cdots,k$。函数 F_j 由网络的结构确定，神经元由带有相应的 F_j 函数名称圆圈来表示；(4)有向连接线：它们将输入层、第一层神经元、第二层神经元、最后一层神经元及输出单元连接起来，箭头表示信息流动的方向，所有这些一起形成网络结构，它确定了网络的泛函能力。网络的结构是指神经元的组织及包含的连接。图2.2给出了泛函网络的拓扑结构图。

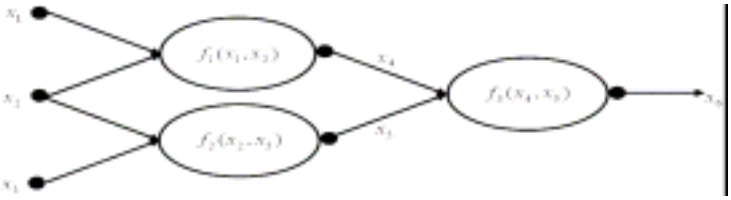

图2.2 泛函网络的拓扑结构

Fig.2.2 The topology configuration of the functional network

2.4.3 泛函网络的特征

标准神经网络与泛函网络之间最明显的区别是：在标准神经网络系统中，给定神经网络的激活函数(常用的激活函数有 Sigmoid 函数、三角波状函数、梯形函数和 RBF 等)，而对激活函数的连接权值进行训练。在泛函网络系统中，没有连接权值，泛函网络训练的是函数 F ，也即对激活函数进行训练。在标准神经网络系统中，输出单元是单个神经元的输出(最后一个单元)，而在泛函网络系统中，输出单元可输出一个或多个神经元，这隐含了这样一种相容性，即：连接到输出单元的神经元个数越多，输出神经元的自由度越少。泛函网络的一个重要特性是能处理连续模型，与初值选举无关等。

2.5 本章小结

本章主要对进化策略、差分演化算法、泛函网络的产生背景、发展及特点作了介绍。根据文章的内容，对进化策略、差分演化算法的原理、特征及工作流程、泛函网络的拓扑结构作了详细的介绍。

3 标准进化策略在求矩阵特征值特征向量中的应用

3.1 引 言

求解矩阵的特征值及特征向量，是科学和工程计算中最普遍的问题之一。在许多应用领域，经常使用矩阵的特征值及特征向量。目前，关于特征值、特征向量问题的数值解法有两种：变换法和迭代法。其中，变换法是直接对矩阵进行处理，通过一系列变换，使之变成较容易求解特征值、特征向量的新的矩阵，但是变换方法常常存贮量较大，计算速度较慢。迭代法是通过一系列矩阵向量乘积而求得特征值和特征向量，常用的方法有：子空间迭代法[1]、Lanczos 方法[1]以及 Davidson 方法[2]。虽这些方法在求解矩阵的特征值及特征向量问题中都取得了巨大的成功，但普遍存在着计算精度低、收敛速度慢及泛化能力弱等缺陷。

本章利用进化策略算法求解矩阵特征值及特征向量。该方法可用于求解任意阶实矩阵 A 的特征值及特征向量。从实验结果可以看出，这种基于进化策略求解矩阵特征值及特征向量的方法，与传统方法相比，具有求解精度高、收敛速度快等特点，能够快速有效地求得任意矩阵的特征值及特征向量。

3.2 特征值与特征向量理论[45]

设 A 是一个 $n\times n$ 方阵， X 是一个 n 维向量，乘积 $Y=AX$ 可以看成是 n 维空间内的线性变换。如果能够找到一个标量 λ，使得存在一个非零向量 X，满足 $AX=\lambda X$，则可以认为线性变换 $T(X)=AX$ 将 X 映射为 λX，此时称 X 是对应于特征值 λ 的特征向量 X。它们形成了 A 的一个特征对(eigenpair) λ, X。通常标量 λ 和向量 X 可以是复数，为了简单起见，特征值考虑在复数范围内，特征向量考虑在实数范围内。

定义 2.1 如果 A 是一个 $n\times n$ 实矩阵，则它存在 n 个特征值 $\lambda_1,\lambda_2,\cdots,\lambda_n$，其中 $\lambda_i(i=1,2,\cdots,n)$ 为实数或复数。

定义 2.2 如果 λ 是 A 的特征值并且非零向量 V 具有如下特性： $AV=\lambda V$，则 V 称为矩阵 A 对应于特征值 λ 的特征向量。

引理：(1) 对于每个惟一的特征值 λ 至少有一个与该特征值相应的特征向量 V。

(2) 如果特征值 λ 的重复度为 r，则至多有 r 个与该特征值相应的线性无关的特征向量 $V_1,V_2,\cdots,V_r$。

3.3 标准进化策略求矩阵特征值的算法

3.3.1 求矩阵特征值的算法步骤

(1) 确定矩阵特征值个体的表达方式：表达式中个体由目标变量 X 和标准差 σ 两部分组成，因为是在复数范围内求特征值，所以每个个体有 2 个分量，分别代表特征值的实部和虚部，即 $(X,\sigma)=((x_1,x_2),(\sigma_1,\sigma_2))$。

(2) 随机生成所求矩阵特征值初始群体：初始群体由 μ 个个体组成，每一个个体 (X,σ) 包含 2 个分量，分别对应复阵特征值的实部和虚部，初始个体是随机生成的，设初始个体的标准差 $\sigma(0)=3.0$。

(3) 计算适应度：特征值是在满足将特征值代入特征多项式后，即多项式 $e=\det(\lambda I-A)$ 的值越小时，则特征值的近似程度越好，取适应度函数为：$f=1/(1+e^2)$，适应度值越接近 1，特征值越优良，其中：$0<f<1$，终止条件选择一个很接近 1 的值 ε，当适应度值大于 ε 时终止。

(4) 如果满足条件，则终止，此时选出最优解。否则，继续向下进行。

(5) 根据进化策略，采用下述操作 5.1)-5.4)产生新群体：

5.1) 重组：从父代个体中随机取出两个个体，交换目标变量和随机因子，产生新个体；目标变量采用离散重组，随机因子采用黄金分割重组。

5.2) 突变：对重组后的个体添加随机变量，按照式子 $\sigma'_i=\sigma_i\cdot\exp(r'\cdot N(0,1)+r\cdot N_i(0,1))$ 与式 $x'_i=x_i+\sigma_i\cdot N_i(0,1)$，其中 $i=1,2$ 产生新个体。其中 r 及 r' 取为 1。

5.3) 计算个体适应度 f。

5.4) 选择：采用 (μ,λ) 选择策略，挑选优良的个体作为进化的结果。

(6) 反复执行第 (5)步，直到满足终止条件，选择最佳的个体作为进化策略的结果，即所求的最优特征值。

3.3.2 算法实现

见附录 1 源程序

3.3.3 仿真实例

为验证算法求解任意矩阵特征值的正确性，适应度函数表达式取为：$f=1/(1+e^2)$，其中 e 为近似特征值代入特征多项式所得的结果。根据算法的思想，当 f 的值越接近1 时，则表示特征值与精确解间的误差越小。以下算例，均采用 (μ,λ) 选择进化策略，其中 $\mu=20,\lambda=7*\mu=140$。在例1中，终止条件取 $\varepsilon=0.9999999999$。在例2中，终止条件取 $\varepsilon=0.999999999$。在例3中，终止条件取 $\varepsilon=0.9999999999$。以下例子表格中精确解均利用数学软件*Matlab*求得。

例 1 求矩阵 $A=\begin{bmatrix}12 & 6 & -6\\ 6 & 16 & 2\\ -6 & 2 & 16\end{bmatrix}$ 的全部特征值。

利用本节的算法，我们可得到所求矩阵全部特征值，计算结果见表3.1。

表3.1 复数域内特征值计算结果

Table3.1 The result of the eigenvalue in complex region

精确解	Jacobi法	本文算法
18	18.000000	18.00000000000000
21.54400374	21.543999	21.54400374531753
4.455996255	4.4559898	4.45599625468247

从表3.1可以看出文献[46]中所用的Jacobi法在求解精度上比本章所采用的进化策略算法差。本节算法求解的特征值与精确解相比最大误差为10^{-9}，Jacobi法的最大误差为10^{-3}。图3.1为所求特征值的适应度函数值随迭代次数变化的曲线，从图中可直观地看出所求近似解的变化趋势与收敛速度。

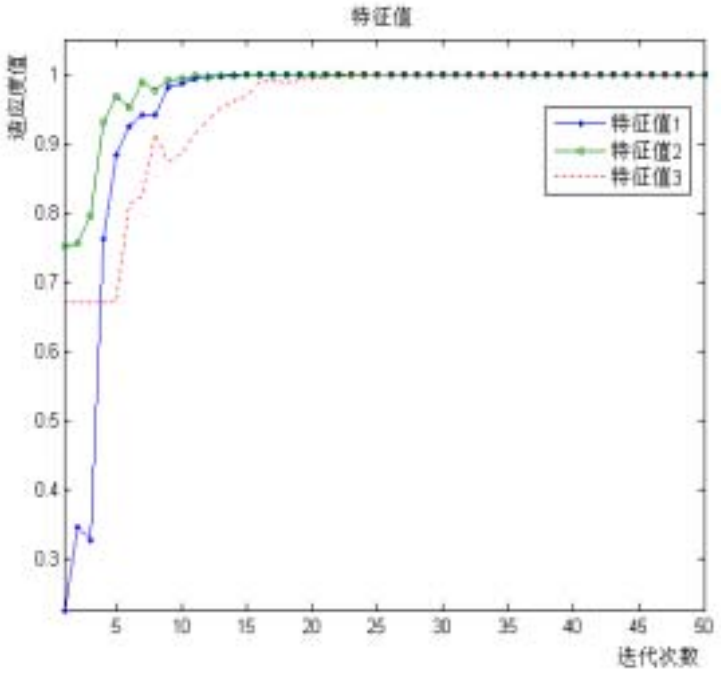

图3.1 特征值的适应度函数变化曲线

Fig.3.1 Fitness value's curve diagram changes of the eigenvalue

例 2 求矩阵 $A=\begin{bmatrix}-3 & 1 & -1\\ -7 & 5 & -1\\ -6 & 6 & -2\end{bmatrix}$ 的全部特征值。

利用本节的算法，对应地求出矩阵的特征值，其计算结果见表3.2。

表3.2 复数域内特征值计算结果

Table3.2 The result of the eigenvalue in complex region

精确解	*Matlab*所求解[45]	同步求解法[7]	本文算法
4	4.00000000000000	4	4.00000000000000 + 0.00000000000000i
-2	-2.00000005285691	-2	-2.00000000000082 + 0.00000000000014i
-2	-1.99999994714309	-2	-1.99999999999918 – 0.00000000000014i

此例中的矩阵含有一个二重特征值-2，从表3.2中可以看出本章算法在求解特征值时与*Matlab*所求解相比最大误差为10^{-8}，与文献[7]中的同步求解法所得解及精确解间的最大误差为10^{-13}，优于*Matlab*法。由此例可看出本节提出的进化算法在求解矩阵特征值含重根的情况下仍然是有效的。图3.2给出为所求特征值的适应度函数值随迭代次数变化的曲线，从图中可以看出所求近似解的变化趋势及收敛速度。

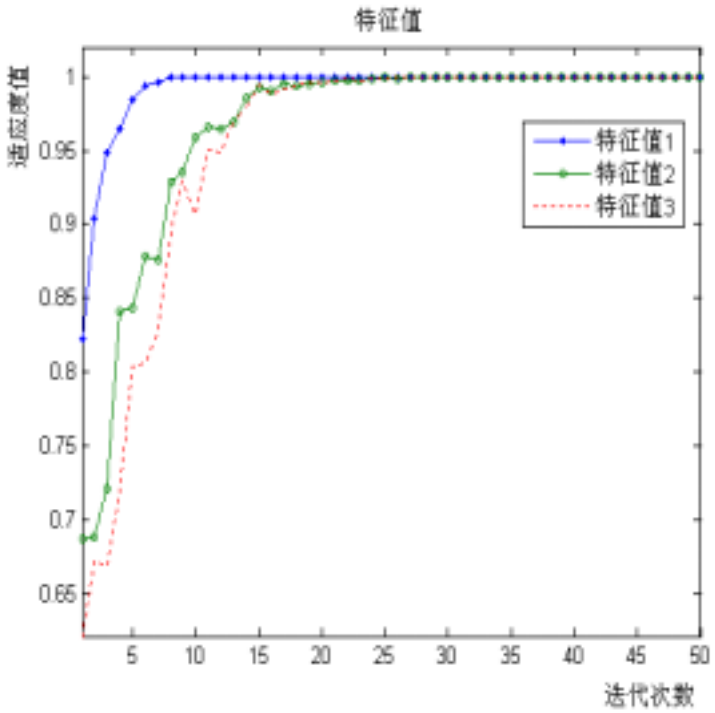

图3.2 特征值的适应度函数变化曲线

Fig.3.2 Fitness value's curve diagram changes of the eigenvalue

例 3 求矩阵 $A=\begin{bmatrix} 10 & 2.5 & 4 & 9 \\ 2.5 & 10 & 3 & 7 \\ 0 & 1 & 6 & 2 \\ 0 & 0 & 2 & 6 \end{bmatrix}$ 的全部特征值。

利用本节的算法，我们可得到所求矩阵全部特征值，其计算结果如表3.3所示。

表3.3 复数域内特征值计算结果

Table3.3 The result of the eigenvalue in complex region

精确解	Languerre迭代分治算法[4]	Newton迭代分治算法[4]	本文算法
13.32279447	13.322793	13.322793	13.32279447344010 - 0.00000000000000i
4.223060030	4.223060	4.223060	4.22306002994501 + 0.00000000000000i
7.227072748 +0.8038188903i	7.227073+0.807819i	4.223060	7.22707274830745 + 0.80381889026742i
7.227072748 -0.8038188903i	7.227073-0.807819i	4.223060	7.22707274830745 -0.80381889026742i

由表3.3可以看出Newton迭代只得到 A 的两个实特征值，一对复特征值被遗漏了，Languerre迭代法虽然将矩阵 A 的四个特征值求出来了，但与本节算法所求得的矩阵特征值相比，精度较低，并且与精确解间的误差大于本文算法与精确解间的误差。图3.3为所求矩阵特征值的适应度函数值随迭代次数增大的变化曲线，从图中可以看出本例含有一组共轭特征值，所求解的精度较高，收敛速度较快。

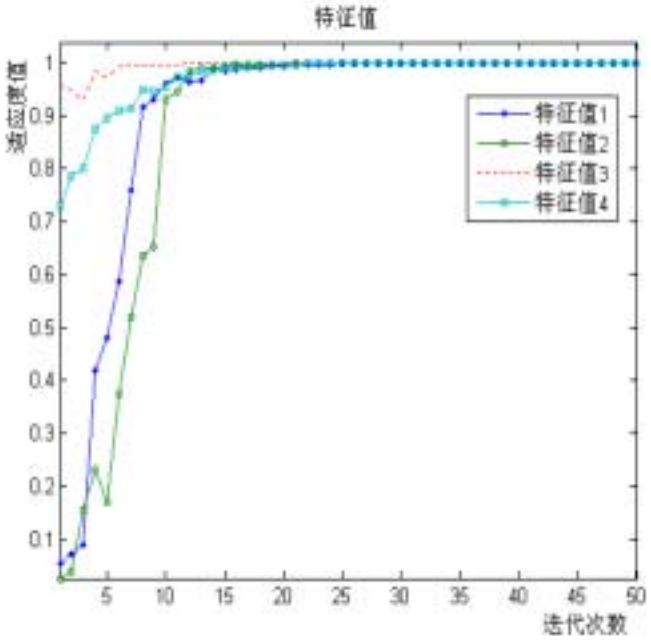

图3.3 特征值的适应度函数变化曲线

Fig.3.3 Fitness value’s curve diagram changes of the eigenvalue

3.4 标准进化策略求实特征值对应的特征向量算法

3.4.1 求矩阵特征向量的算法步骤

(1) 对求矩阵特征向量的算法步骤中所求的特征值进行整理，设误差限 $\eta = 0.0000000001$，若特征值的虚部的绝对值小于 η 时，则记该特征值为实数。从步骤 1 中将得到所有的实特征值，对这些实特征值分别求与其相对应的特征向量。

(2) 随机生成 μ 个初始群体，每一个个体 (X,σ) 包含 n 个分量(n 为矩阵 A 的阶数) 。 即 $(X,\sigma)=((x_1,x_2,\cdots,x_i,\cdots,x_n),(\sigma_1,\sigma_2,\cdots,\sigma_i,\cdots,\sigma_n))$ ， 其 中 $(x_1,x_2,\cdots,x_i,\cdots,x_n)$ 为特征向量个体。初始个体的标准差取 $\sigma(0)=3.0$ 。

(3) 计算适应度：取适应度函数为 $g=1/(1+d)$ ， d 为将特征向量 X 代入线性方程组： $AX=\lambda X$ ，经整理可写成：

$$\begin{cases} f_1(X)=0 \\ f_2(X)=0 \\ \vdots \\ f_n(X)=0 \end{cases} \tag{3.1}$$

然后，令 $d=\sum_{i=1}^{n} f_i^2(X)$，若适应度值越接近 1，则表示特征向量越优，$0<g<1$，终止条件选择一个很接近 1 的值 ε ，当适应度值大于 ε 时终止。

(4) 如果满足条件，则终止，此时选出最优解。否则，继续向下执行。

(5) 根据进化策略，采用下述操作 5.1)-5.4) 产生新群体：

5.1) 重组：从父代个体中随机取出两个个体，交换目标变量和随机因子，产生新个体.目标变量与随机因子均采用黄金分割重组。

5.2) 突变：对重组后的个体添加随机变量，按照式 $\sigma'_i=\sigma_i\cdot\exp(r'\cdot N(0,1)+r\cdot N_i(0,1))$ 与式 $x'_i=x_i+\sigma_i\cdot N_i(0,1)$ 产生新个体。其中 $i=1,2,\cdots,n$ 且 r 及 r' 取为 1。

5.3) 计算个体适应度 g 。

5.4) 选择：采用 (μ,λ) 选择策略，挑选优良的个体作为进化的结果。

(6) 反复执行第(5)步，直到满足终止条件，选择最佳的个体作为进化策略的结果，即为与实特征值相对应的最优特征向量。

3.4.2 算法实现

见附录 2 源程序

3.4.3 仿真实例

求实特征值对应的特征向量时的适应度函数表达式取为： $g=1/(1+d)$ ，其

中 $d=\sum_{i=1}^{n} f_i^2(X)$ 根据上节算法的思想，当 g 的值接近1 时，则表示特征值、特征向量与精确解间的误差越小。以下均采用 (μ,λ) 选择进化策略，其中 $\mu=20,\lambda=7*\mu=140$。在例4中，终止条件取 $\varepsilon=0.9999999999$ ；在例5中，终止条件取 $\varepsilon=0.999999999$ ；在例6中，终止条件取 $\varepsilon=0.9999999999$。算例表格中精确解均利用数学软件*Matlab*求得。

例 4 求与例1中的实特征值相对应的特征向量。

与实特征值相对应的特征向量计算结果见表3.4。

表3.4 实特征值所对应的特征向量计算结果

Table3.4 The result of the eigenvector opposite the real eigenvalue

精确解	Jacobi法	本文算法
$\begin{pmatrix}0\\1\\1\end{pmatrix}$	$\begin{pmatrix}0\\1.000003\\1.000000\end{pmatrix}$	$\begin{pmatrix}0.00000056967248\\1.00002032219228\\1.00000000000000\end{pmatrix}$
$\begin{pmatrix}1\\0.7953336454\\-0.7953336454\end{pmatrix}$	$\begin{pmatrix}1.000000\\0.795408\\-0.795408\end{pmatrix}$	$\begin{pmatrix}1.00000000000000\\0.79535125646788\\-0.79530403822680\end{pmatrix}$
$\begin{pmatrix}1\\-0.6286669788\\0.6286669788\end{pmatrix}$	$\begin{pmatrix}1.000000\\-0.628608\\0.628606\end{pmatrix}$	$\begin{pmatrix}1.00000000000000\\-0.62866756564557\\0.62866157746530\end{pmatrix}$

从表3.4可以看出在求特征向量时，本节方法最大误差为 10^{-5}，Jacobi法最大误差为 10^{-3}。图3.4为所求实特征值所对应的特征向量的适应度函数值随迭代次数变化的曲线，从图中可直观地看出所求近似解的变化趋势与收敛速度。

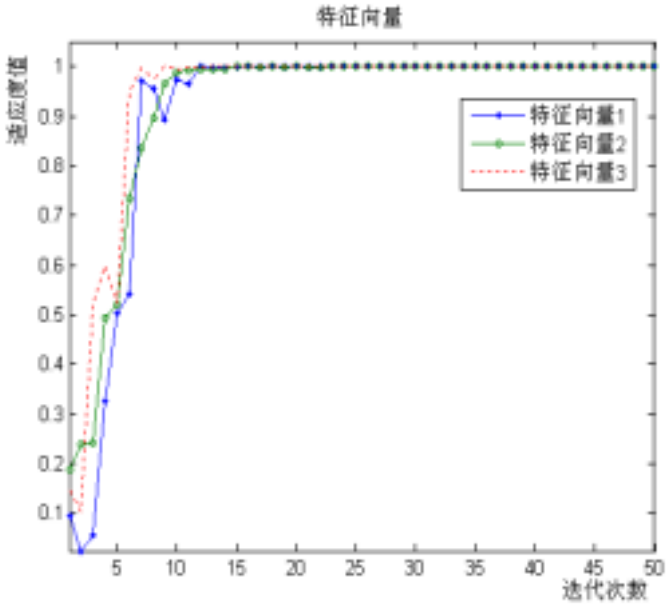

图3.4 特征向量适应度函数变化曲线

Fig.3.4 Fitness value's curve diagram changes of the eigenvector

例 5 求与例2中的实特征值相对应的特征向量。

与实特征值相对应的特征向量计算结果见表3.5。

表3.5 实特征值所对应的特征向量计算结果

Table3.5 The result of the eigenvector opposite the real eigenvalue

精确解	*Matlab*所求解[45]	同步求解法[7]	本文算法
$\begin{pmatrix}0\\1\\1\end{pmatrix}$	$\begin{pmatrix}0\\1\\1\end{pmatrix}$	$\begin{pmatrix}0\\1\\1\end{pmatrix}$	$\begin{pmatrix}0.00000000125825\\0.99999999853348\\1.00000000000001\end{pmatrix}$
$\begin{pmatrix}1\\1\\0\end{pmatrix}$	$\begin{pmatrix}1.00000000000000\\1.00000000000000\\0.00000005285691\end{pmatrix}$	$\begin{pmatrix}1\\1\\0\end{pmatrix}$	$\begin{pmatrix}1.00000000000000\\1.00000000016825\\0.00000005703648\end{pmatrix}$
$\begin{pmatrix}1\\1\\0\end{pmatrix}$	$\begin{pmatrix}1.00000000000000\\1.00000000000000\\-0.00000005285691\end{pmatrix}$	$\begin{pmatrix}1\\1\\0\end{pmatrix}$	$\begin{pmatrix}1.00000000034141\\1.00000000000000\\-0.00000006015755\end{pmatrix}$

此例中所求特征向量与*Matlab*所求解相比最大误差为10^{-9}；文献[7]中的同步求解法所求解及精确解间的最大误差为10^{-8}。图3.5给出实特征值所对应的特征向量的适应度函数值随迭代次数变化的曲线，从图中可以看出所求近似解的变化

趋势及收敛速度。

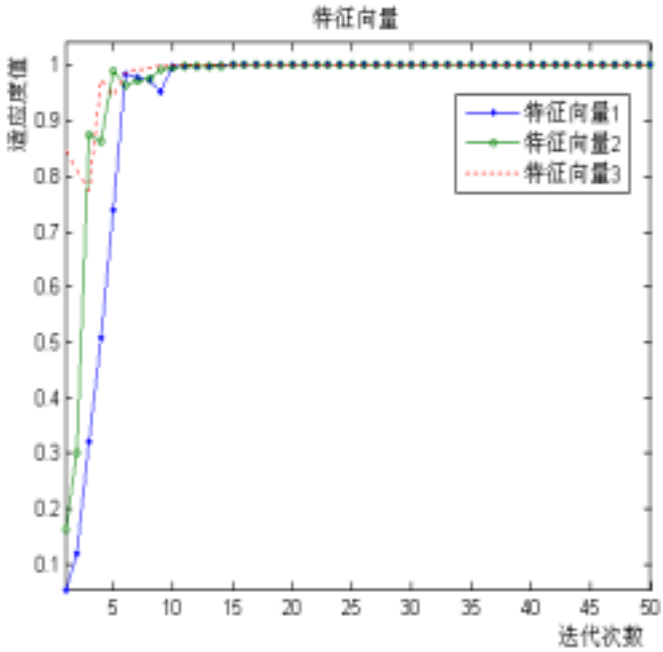

图3.5 特征向量适应度函数变化曲线

Fig.3.5 Fitness value's curve diagram changes of the eigenvector

例 6 求与例3中的实特征值相对应的特征向量。

与实特征值相对应的特征向量计算结果如表3.6。

表3.6 实特征值所对应的特征向量计算结果

Table3.6 The result of the eigenvector opposite the real eigenvalue

精确解	本文算法
$\begin{pmatrix} -0.716542469 \\ -0.6895225072 \\ -0.1017511866 \\ -0.02779026158 \end{pmatrix}$	$\begin{pmatrix} -0.71654246901102 \\ -0.68952240940285 \\ -0.10175114779194 \\ -0.02779027224297 \end{pmatrix}$
$\begin{pmatrix} -0.5969098319 \\ -0.3306555246 \\ -0.6974076342 \\ 0.7849535101 \end{pmatrix}$	$\begin{pmatrix} -0.59690983191003 \\ -0.33065515791063 \\ -0.69740709604496 \\ 0.78495305031909 \end{pmatrix}$

表3.6中利用本节算法所求得的实特征值所对应的特征向量与精确解间的误差为10^{-7}，精度非常高。图3.6为与所求实特征值相对应的特征向量的适应度函数值随迭代次数增大的变化曲线，从图中可以看出所求解的精度较高，收敛速度较快。

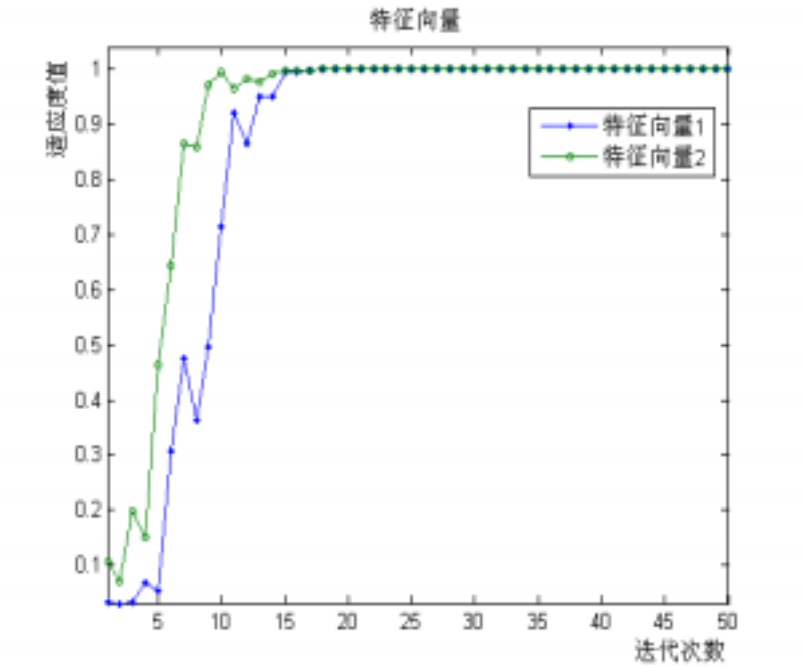

图3.6 特征向量适应度函数变化曲线

Fig.3.6 Fitness value's curve diagram changes of the eigenvector

3.5 本章小结

本章给出的进化策略算法可用于求任意实矩阵特征值、特征向量，从以上算例可以看出该算法所求得的解，精度高、收敛速度快。在求解矩阵的特征值、特征向量时优于经典的 Jacobi 方法；对于虚特征值的情形，由于 Newton 迭代不收敛，故求不出矩阵的虚特征值，而本章算法对于 n 阶的矩阵，能将其在复数域内的所有特征值求出，所求解的精度高于 Languerre 迭代算法及其它一些算法所求的解；对于求解含重特征值的矩阵时，该算法也是行之有效的。总之，提出的基于进化策略的算法能够快速有效地获得任意矩阵对应的特征值及特征向量。

4 改进进化策略在配平化学方程式中的应用

4.1 引 言

化学反应的计量关系是人们进行定量分析的基础，故配平化学方程式是化学工作者必须具备的基本素质。然而，化学反应成千上万，类型繁多，各具特点，这给化学方程式配平带来了困难。不少研究者对化学方程式的配平进行了广泛的探讨和研究，并根据各类化学反应的特点提出了许多配平方法[9−10]，常见的有氧化值法、电子法、离子电子法、观察法等。这些方法虽可用于化学方程式的配平，但当化学方程式比较复杂时，采用这些传统的配平方法，普遍采用一种“试凑”的方法，这种方法往往与人们的所处专业领域的“先验知识”有关。但不论是什么类型的化学反应一定都遵循质量守恒定律。基于此目的，本章研究任意类型的化学方程式的配平问题，根据质量守恒定律和化学方程式左右两边的原子构成来建立一种数学模型，利用改进后的进化策略这一优化算法来确定化学方程式中的各反应物及生成物前的最简整系数，最终完成对任意化学方程式的配平。

本章首先对标准进化策略算法作一改进，利用进化策略的特点，根据质量守恒定律和化学方程式左右两边的原子构成来建立数学模型，将化学方程式配平问题转化为最优化问题，改进后的进化策略算法用于最优求解问题，提出了一种配平化学方程式的进化策略新算法。该算法中的初始群体中的个体为整数，通过对群体的进化，来求化学方程式各物质前的最简系数。最后，实验结果表明，这种改进后的进化策略算法能够有效地确定出任意类型化学方程式各物质前的最简系数，可完成任意类型化学方程式配平。

4.2 数学模型

对任意类型的化学方程式来说，作如下假设：

1) 化学方程式的反应物与生成物各为 p 种、q 种，所有的物质共含有 n 种元素；

2) 第 i 种反应物中所含的第 j 种元素的原子数为 $a_{ij}(i=1,2,3,\cdots p;\ j=1,2,3\cdots n)$；

3) 第 k 种生成物中所含的第 j 种元素的原子数为 $b_{kj}(k=1,2,3,\cdots q;\ j=1,2,3\cdots n)$；

4) p 种反应物分子式的系数分别是：$x_1, x_2, \cdots, x_p$，q 种生成物分子式的系数分别是：$x_{p+1}, x_{p+2}, \cdots, x_{p+q}$；

5) 将化学方程式右端的生成物全部移至左端，此时将生成物中所有的原子数 $b_{kj}(k=1,2,3,\cdots q; j=1,2,3\cdots n)$ 全修改为 $-b_{kj}(k=1,2,3,\cdots q; j=1,2,3\cdots n)$.

那么，根据质量守恒定律，我们可以得到如下的平衡齐次线性方程组：

$$\begin{cases} a_{11}x_1 + a_{21}x_2 + \cdots + a_{p1}x_p - b_{11}x_{p+1} - b_{21}x_{p+2} - \cdots - b_{q1}x_{p+q} = 0 \\ a_{12}x_1 + a_{22}x_2 + \cdots + a_{p2}x_p - b_{12}x_{p+1} - b_{22}x_{p+2} - \cdots - b_{q2}x_{p+q} = 0 \\ \cdots \quad \cdots \quad \cdots \quad \cdots \\ a_{1n}x_1 + a_{2n}x_2 + \cdots + a_{pn}x_p - b_{1n}x_{p+1} - b_{2n}x_{p+2} - \cdots - b_{qn}x_{p+q} = 0 \end{cases} \tag{4.1}$$

置：

$$A_1 = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{p1} & a_{p2} & \cdots & a_{pn} \end{bmatrix}$$ 为反应物的原子数矩阵；

$$A_2 = \begin{bmatrix} -b_{11} & -b_{12} & \cdots & -b_{1n} \\ -b_{21} & -b_{22} & \cdots & -b_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ -b_{q1} & -b_{q2} & \cdots & -b_{qn} \end{bmatrix}$$ 为生成物的原子数矩阵；

$X_1 = (x_1, x_2, \cdots x_p)^T$ 为反应物的系数列矩阵；$X_2 = (x_{p+1}, x_{p+2}, \cdots, x_{p+q})^T$ 为生成物的系数列矩阵。若记 $A = (A_1^T \mid A_2^T)$ 为原子矩阵，$X = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}$ 为系数矩阵，则上面的齐次线性方程组可表示成如下的矩阵形式：

$$AX = 0 \tag{4.2}$$

这样一来，对任意一配平化学反应方程式问题，即可转化为求上述齐次线性方程组的最简整数解问题。一般情况下，我们所得到的方程组(4.1)或(4.2)是一矛盾方程组，用传统的线性方程组求解方法难以得到其准确解。

4.3 进化策略改进及求最简正整系数解的步骤

4.3.1 问题分析及解决方案

① 在重新生成新个体的过程中，可能会出现第 $i+1$ 代的最优个体劣于第 i 代的最优个体。所以我们根据精英策略[47]的选取方法，在进化过程中，将相邻的两代中最优个体进行比较，始终将最优的个体保留下来，将其无条件的作为下一代的一个新个体，参与进化，使得解逐步向最优解逼近；

② 传统的进化策略采用全基因高斯突变算子，一般认为，这样有利于从各个方向进行搜索，但事实上，当问题规模比较大时，由于突变方向的随机性，往往使突变的效果互相抵消，很难得到进化了的后代个体，导致进化策略在求解高维问题时的收敛速度较慢，甚至停顿。为了改变进化策略在收敛性能方面存在的不足，我们对它的变异算子进行一定的改进。根据柯西分布和高斯分布的相似性，以及柯西分布具有较高的两翼概率特性，即柯西分布有一条很长的尾巴。柯西分布容易产生一个远离原点的随机数，它比高斯变异产生的随机数分布的范围大，如果用柯西变异替换原来进化策略中的高斯变异来产生后代，这就意味着柯西变异有可能很快跳出局部极小的区域，得出最优异的解；

③ 因为 Gauss 突变具有很强的局部搜索能力和较弱的全局搜索能力，而 Cauchy 突变全局搜索能力强局部搜索能力弱，所以把二者结合充分发挥各自的优势，使进化速度更快，准确度更高。因此在改进了的算法中，我们对目标变量采用柯西变异，对决策变量采用高斯变异；

④为了避免式(4.2)方程组出现零解、进化过程中系数的波动较大，我们可以预先根据经验给系数一个学习区间$[1,M]$，其中M为一个较大的正整数。

由以上 4 点我们将标准进化策略算法进行改进得如下的新算法。

4.3.2 改进后 ES 算法

根据 4.3.1 的分析，为了解决任意一配平化学反应方程式问题，我们将标准 ES 算法作一改进，改进后的 ES 算法归纳如下：

(1) 确定个体的表达方式：表达式中个体由目标变量X和标准差σ两部分组成，因为反应物与生成物的总个数为$p+q$，所以每部分个体有$p+q$个分量，即$(X,\sigma)=((x_1,x_2,\cdots,x_p,x_{p+1},\cdots,x_{p+q}),(\sigma_1,\sigma_2,\cdots,\sigma_p,\sigma_{p+1}\cdots,\sigma_{p+q}))$。

(2) 随机生成初始群体：进化策略中初始群体由μ个个体组成，每一个个体(X,σ)包含$p+q$个分量，其中$(x_1,x_2,\cdots,x_p,x_{p+1}\cdots,x_{p+q})$为整系数解，$X$内的每一个个体取为 1 至正整数$M$内的整数。初始个体是随机生成的，初始个体的标准差$\sigma(0)=3.0$。

(3) 计算适应度：取适应度函数为$f=1/(1+d)$，其中$d=\sum_{i=1}^{n}g_i^2(X)$为将整系数解X代入式(4.2)所得的n(n为所有的物质中所含不同元素的总个数)个结果的平方和，若适应度值越接近 1，则表示整系数解越优，终止的判定条件为$\varepsilon=1$，当适应度值等于ε时终止。

(4) 如果满足条件，则终止，此时选出最优解。否则，继续向下进行学习。

(5) 根据进化策略，采用下述操作产生新群体：

5.1) 重组：从父代个体中随机取出两个个体，交换目标变量和随机因子，产生新个体。目标变量与随机因子均采用黄金分割重组。

5.2) 突变：对重组后的个体添加随机变量，对于决策变量采用高斯变异，按照式$\sigma_i^{'} = \sigma_i \cdot \exp(r^{'} \cdot N(0,1) + r \cdot N_i(0,1))$进行变异；对于目标变量采用柯西变异，按照式$x_i^{'} = x_i + \sigma_i^{'} \cdot \eta_i$产生新个体，参数$\eta_i$是当$t=1$时的一个柯西分布的随机变量比例参数。其中$r = r^{'} = 1$，$i = 1,2,\cdots, p, p+1,\cdots, p+q$。

5.2.1)将所有的个体里的分量$x_i^{'}(i = 1,2,\cdots, p, p+1,\cdots, p+q)$与1及$M$进行比较；

5.2.2) 如果$x_i^{'} > M$，则$x_i^{''} = 2 * M - x_i^{'}$；

5.2.3) 如果$x_i^{'} < 1$，则$x^{''}{}_i = 2 * 1 - x^{'}{}_i$；

5.2.4) 重复 5.2.2)-5.2.3)的步骤，直到$x_i^{''} \in [1, M]$。

5.3) 计算新个体的适应度。

5.4) 选择：采用(μ, λ)选择策略，挑选出优良的个体作为此代进化的最优结果。

5.4.1) 将此时所得的最优解与上一代的最优解进行比较；

5.4.2) 如果此时的最大适应度值小于上一代最优个体的适应度值，则用上一代的最优个体替换此时的最优个体，否则将此时的最优解传递到下一代进行学习。

(6) 反复执行第(5)步，直到满足终止条件，选择最佳的个体作为进化的结果，即为最优的整系数解。

(7) 将整系数解化为最简正整数系数解。

4.4 算法实现

见附录 3 源程序

4.5 仿真实例

取适应度函数表达式为：$f = 1/(1+d)$，其中$d = \sum_{i=1}^{n} g_i^2(X)$为将整系数解$X$代入式(4.2)所得的$n$($n$为所有的物质中所含不同元素的总个数)个结果的平方和；根据上节算法的思想，当f的值越接近1 时，则表示所求的最优解与精确解间的误差越小。

以下算例，均采用(μ, λ)选择策略，其中$\mu = 30, \lambda = 7 * \mu = 210$，为求得最优的正整系数解，终止条件均取为$\varepsilon = 1$。

例 1 配平化学方程式[48] ：$?FeS_2+?O_2\rightarrow?Fe_2O_3+?SO_2$(其中？表示待确定的配平系数)

解：建立数学模型

由该化学方程式可知 $p=2$、$q=2$、$n=3$，各种物质所含的三种元素的原子数如下表：

物质 / 元素	反应物		生成物	
	FeS_2	O_2	Fe_2O_3	SO_2
Fe	1	0	2	0
S	2	0	0	1
O	0	2	3	2

原子矩阵为：$A=\begin{bmatrix}1&0&-2&0\\2&0&0&-1\\0&2&-3&-2\end{bmatrix}$，进化求解时系数在区间$[1,M]$内进行进化，本例中取正整数 $M=30$。通过迭代计算求得最简正整系数解为：$(4,\ 11,\ 2,\ 8)^T$，从图 4.1 中适应度值随迭代次数变化的曲线可以看出，获得最优解时的迭代次数为：$t=13$。

则配平后的化学方程式可以写为：

$$4FeS_2+11O_2==2Fe_2O_3+8SO_2$$

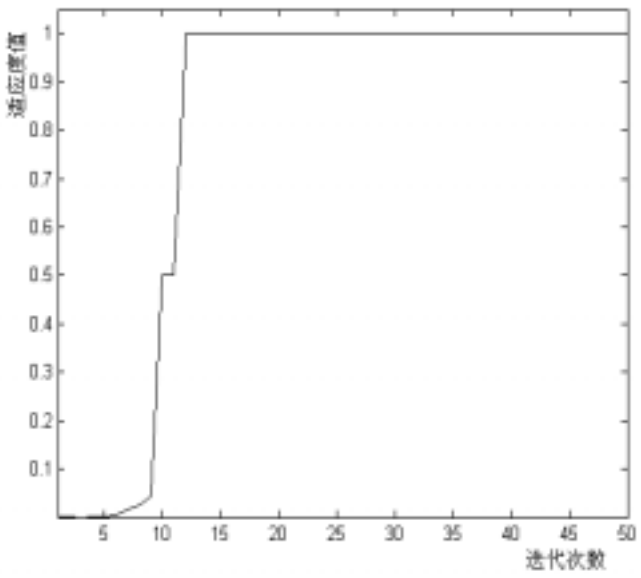

图 4.1 适应度值变化曲线图

Fig.4.1 The curve diagram changes of the fitness value

例 2 配平化学方程式[48] ：$?Cu+?HNO_3\rightarrow?Cu(NO_3)_2+?NO_2\uparrow+?H_2O$

(其中？表示待确定的配平系数)

解：建立数学模型

由该化学方程式可知 $p=2$、$q=3$、$n=4$，各种物质所含的四种元素的原子数如下表：

元素 / 物质	反应物		生成物		
	Cu	HNO_3	$Cu(NO_3)_2$	NO_2	H_2O
Cu	1	0	1	0	0
H	0	1	0	0	2
N	0	1	2	1	0
O	0	3	6	2	1

原子矩阵为：$A=\begin{bmatrix}1 & 0 & -1 & 0 & 0\\ 0 & 1 & 0 & 0 & -2\\ 0 & 1 & -2 & -1 & 0\\ 0 & 3 & -6 & -2 & -1\end{bmatrix}$，求解时系数在区间$[1,M]$内进行进化，本例中取正整数 $M=30$。通过迭代计算求得最简正整系数解为：$(1,\ 4,\ 1,\ 2,\ 2)^T$，从图 4.2 中适应度值随迭代次数变化的曲线可以看出，获得最优解时的迭代次数为：$t=21$。

则配平后的化学方程式可以写成：

$$Cu+4HNO_3==Cu(NO_3)_2+2NO_2\uparrow+2H_2O$$

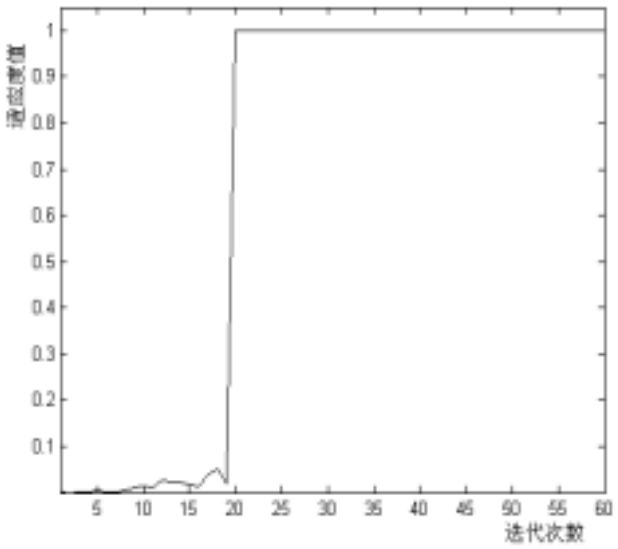

图 4.2 适应度值变化曲线图

Fig.4.2 The curve diagram changes of of the fitness value

例 3 配平化学方程式[49] ：

$?KMnO_4 + ?HCl \rightarrow ?Cl_2 + ?MnCl_2 + ?KCl + ?H_2O$ (其中？表示待确定的配平系数)

解：建立数学模型

由该化学方程式可知 $p=2$、$q=4$、$n=5$，各种物质所含的三种元素的原子数如下表：

元素 / 物质	反应物		生成物			
	$KMnO_4$	HCl	Cl_2	$MnCl_2$	KCl	H_2O
K	1	0	0	0	1	0
Mn	1	0	0	1	0	0
O	4	0	0	0	0	1
H	0	1	0	0	0	2
Cl	0	1	2	2	1	0

原子矩阵为：$A=\begin{bmatrix} 1 & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 0 & -2 \\ 0 & 1 & -2 & -2 & -1 & 0 \end{bmatrix}$，求解时系数在区间$[1, M]$内进行进化，本例中取正整数 $M=40$。通过迭代计算求得最简正整系数解为：$(2,\ 16,\ 5,\ 2,\ 2,\ 8)^T$，从图 4.3 中适应度值随迭代次数变化的曲线可以看出，获得最优解时的迭代次数为：$t=68$。

则配平后的化学方程式可以写成：

$$2KMnO_4 + 16HCl == 5Cl_2 + 2MnCl_2 + 2KCl + 8H_2O$$

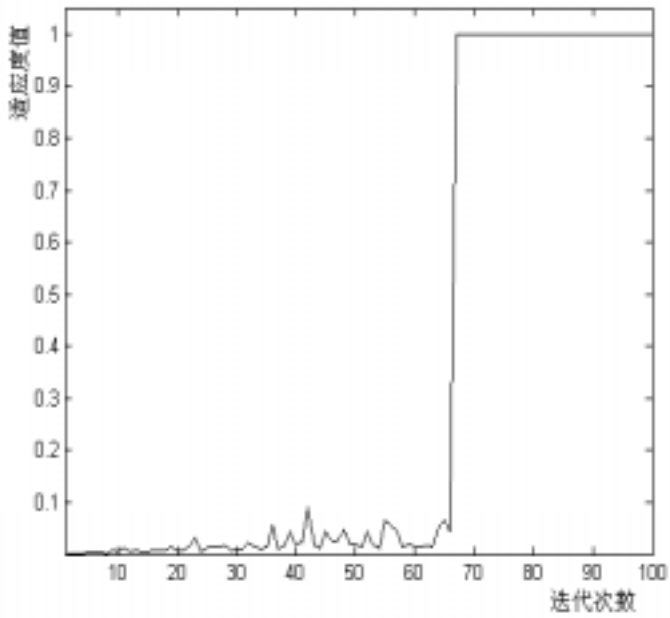

图 4.3 适应度值变化曲线图

Fig.4.3 The curve diagram changes of the fitness value

4.6 本章小结

本章主要对标准进化策略算法进行了改进，对变异算子进行了改进，将高斯与柯西变异的优点集合在一起，同时对进化过程的个体作了限制，采用整数个体参加训练学习，并对每一代中的个体进行限制，使得所求解与精确解间的误差为零。从上面的算例可以看出，本章中所提出的算法在配平化学方程式时，相比以前所采用的算法来说，具有收敛速度较快的优点。

5 进化策略在求复函数方程根中的应用

5.1 引 言

在大量的科学和工程计算中都要涉及复函数方程求根问题，研究复函数方程求根的算法有着重要的理论意义和应用价值。求复函数方程根的算法有多种，常用的有迭代法、牛顿法、下山法、转换法、圆盘迭代法等等。但是上述各种算法都存在着一定的局限性，对于迭代法、牛顿法存在收敛性和收敛速度问题，对迭代初值的选取比较敏感；圆盘迭代法计算很复杂；下山法方法简单，但存在着局部极值问题；转换法是将一般方程求根的问题转换为多项式方程的求根的问题，该方法在转换计算多项式方程的系数的过程中要计算多个积分，所求根的精度也不高。

基于进化策略的特点，本章分析了它在处理非线性函数的优化问题中存在的收敛速度和收敛性的不足，提出了一种基于改进变异算子的双种群进化策略算法。实验结果表明，这种基于改进变异算子的双种群进化策略用于求解复函数方程的根，相比传统方法，具有求解精度高、收敛速度快等特点。

5.2 算法的设计思想及理论基础

5.2.1 把方程求根问题转换为求函数最小值

对于所有的方程求根问题，包括复函数方程求根问题都可以转换为求函数最小值问题。对于复函数方程 $f(z)=0$，可以转换成如下求最小值优化问题：$\min|f(z)|$。

5.2.2 复函数方程根的分布理论

引理1[50]：设函数 $f(z)$ 为在区域 S 内的亚纯函数，r 是 S 内可求长Jordan曲线，且其内部属于 S，$f(z)$ 在 r 上无零点和极点，则 $N-P=\frac{1}{2\pi i}\int_r \frac{f'(z)}{f(z)}dz$，其中 N，P 分别表示 $f(z)$ 在 r 内部零点和极点个数.当 $f(z)$ 在 S 内解析时，上式变为：

$$N=\frac{1}{2\pi i}\int_r \frac{f'(z)}{f(z)}dz=\frac{1}{2\pi i}\Delta r \quad Argf(z)。$$

如果 $f(z)=u+iv$ 在简单闭曲线 C 上和在 C 内解析，且在 C 上不等于零，那么 $f(z)$ 在 C 内零点的个数等于 $1/(2\pi)$ 乘以当 z 沿 C 的正向绕行一周 $f(z)$ 的幅角的改变

量。

推论1[51]：设 $f(z)=u+iv$ 在简单闭曲线 C 上和在 C 内解析，且在 C 上不等于零，点 $z_0=x_0+iy_0$ 沿 C 的正向绕行一周，设向量 (u,v) 作正方向的旋转次数为 N_p，作负方向的旋转次数为 N_n，那么在封闭曲线 C 内 $f(z)=0$ 的根的个数 $N=N_p-N_n$。

如果沿曲线 C 取一系列点 $(Z_1,Z_2,\cdots Z_n)$，对于 Z 计算相应 $u(x,y)$，$v(x,y)$，分配对应象限数给 Z，例如：2，2，3，3，3，4，4，4，4，4，1，1，2，2，…。这时可以将象限数序列压缩为(2, 3, 4, 1, 2, …)。象限序列3，4，4，1，2，2，3，3，4，4，1，1，2，2，3，3，压缩为(3, 4, 1, 2, 3, 4, 1, 2, 3)。在该序列中 $N_p=2$ ((1, 2, 3, 4)出现2次)，$N_n=0$ ((4, 3, 2, 1)出现0次)。所有根的个数 $N=N_p-N_n=2$。

5.3 变异算子的改进[52]

生物在进化过程中，逐渐从简单的低级生物发展成为复杂的高等生物。在这一优化过程中，主要经过重组、变异和选择等作用，依照达尔文“物竞天折，适者生存”这一规则进行演化。进化策略是通过模拟生物进化过程来求解优化问题，在进化过程中主要是用变异算子作为进化算子。

在传统的ES算法中，参与进化的只有一个种群，因此这个唯一的种群中的个体只能遵循唯一的规律进行进化(相同的高斯变异算子、标准差 $\sigma(0)$)，较大的高斯变异算子可以保证种群具有较好的探索能力，即能够以比较大的概率到达解空间的各个地方，种群或者种群中的部分个体能够以比较大的概率落在全局最优解所在的邻域，在每一个局部极值都具有很好的局部逃逸能力。但此种局部逃逸能力的取得是以降低解的精度为代价的，种群总是在全局最优解的附近跳转，无法使用小的变异来得到高精度的逼近最优解，在对于非线性函数进行优化时收敛速度较慢。较小的高斯变异算子能够保证种群在局部具有良好的搜索能力，能够以较高的精度得到局部的极值，但小的变异算子不能产生足够大的变异从而使种群从局部进化到全局最优解，从而早熟收敛。

为了避免早熟收敛，可以从提高变异的作用方面进行考虑，也可以从重组和选择方面来考虑。因为在进化策略中，变异算子作为主要算子起着相当大的作用。所以进化策略的收敛速度与变异有着密切的关系，为了改变其在收敛性能方面的不足，这里主要从变异这一方面进行考虑改进设计。因此，在原来变异算子的基础上提出一种新的变异算子，即基于柯西随机数的变异算子。

5.3.1 柯西（Cauchy）分布

一维柯西密度函数集中在原点附近，其定义为：

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

式中$t>0$为比例参数，相应的分布函数定义为：

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

密度函数 $f_t(x)$ 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，因此柯西分布可以看作是无限的。图 5.1 给出了二者的关系。

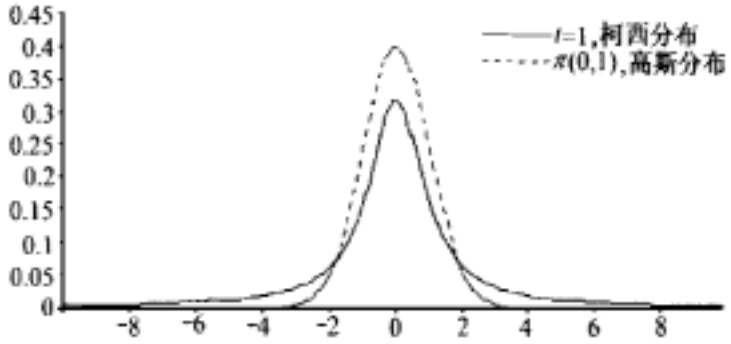

图 5.1 柯西和高斯分布比较

Fig.5.1 The compare distributing of the Cauchy and Gaussian

5.3.2 柯西变异算子

柯西变异算子是用柯西变异替换原来进化策略中的高斯变异。根据柯西分布和高斯分布的相似性以及柯西分布具有较高的两翼概率特性，即柯西分布有一条很长的尾巴；可知，柯西分布容易产生一个远离原点的随机数，它比高斯变异产生的随机数具有更宽的分布范围，如果用柯西变异替换原来进化策略中的高斯变异来产生后代，这就意味着利用柯西变异有可能很快跳出局部极小的区域。本节针对柯西分布和高斯分布的特点提出基于柯西变异的三种方式：即目标变量的柯西变异、策略变量的柯西变异和目标策略变量柯西变异。下面分别介绍这三种方式：

(1) 目标变量的柯西变异

目标变量柯西变异是将进化策略中对目标变量的修改由原来通过高斯分布产生随机数改为由柯西分布所产生的随机数来替代，而对策略变量的修改仍然由高斯分布产生的随机数来修改，即可以用下式表示

$$\sigma_i'(j)=\sigma_i'(j)\cdot\exp(\tau_2\cdot N(0,1)+\tau_1\cdot N_j(0,1)) \tag{5.1}$$

$$x_i'(j)=x_i(j)+\sigma_i'(j)\cdot\eta_j, \quad i=1,2,\cdots,n \tag{5.2}$$

这里η_j是$t=1$时的一个柯西随机变量比例参数，用于更新每一个分量。(5.1) 式保持原来的高斯变异。

这种变异方式中,由于η_j是一个柯西分布的随机变量,它比高斯分布产生的随机变量的范围大，因而对目标变量的修改量较大。

(2) 决策变量柯西变异

决策变量柯西变异是将原进化策略中对决策变量的修改由原来的高斯分布产生的随机数改为由柯西分布所产生的随机数来修改。即可以用下式表示

$$\sigma_i'(j)=\sigma_i'(j)\cdot\exp(\tau_2\cdot\eta+\tau_1\cdot\eta_j) \tag{5.3}$$

$$x_i'(j)=x_i'(j)+\sigma_i'(j)\cdot N_j(0,1)，\ i=1,2,\cdots,n \tag{5.4}$$

这里η，η_j均为$t=1$时的一个柯西随机变量比例参数，其中η_j用于更新每一个分量。(5.4) 式保持原来的高斯变异。实际上,对目标变量的修改也有由柯西分布产生的随机数通过策略变量间接地修改。

这种变异方式中,由于η和η_j是一个柯西分布的随机变量,它比高斯分布产生的随机变量的范围大,因而对策略变量的修改量大,相应地对目标量的修改量也大。

(3) 目标策略变量柯西变异

将原进化策略中对目标变量和决策变量的修改均由原来通过高斯分布产生的随机数改为由柯西分布所产生的随机数来代替。即可以用下式表示

$$\sigma_i'(j)=\sigma_i\cdot\exp(\tau_2\cdot\eta+\tau_1\cdot\eta_j) \tag{5.5}$$

$$x_i'(j)=x_i(j)+\sigma_i'(j)\cdot\eta_j，\ i=1,2,\cdots,n \tag{5.6}$$

这里η，η_j均为$t=1$时的一个柯西随机变量比例参数，其中η_j用于更新每一个分量。

在这种变异方式中,由于η和η_j均为一个柯西分布的随机变量，它比高斯分布产生的随机变量的范围大，同时对目标变量和策略变量进行修改的波动较大。

总而言之,上述对变异算子所进行的三种改进方法都在不同程度上增加了对目标变量的修改量，这样可以加快目标变量向最优解逼近；同时,可使目标变量不易陷入局部最优解,从而加快收敛速度。

5.4 双种群算法的实现过程

在算法实现过程中，其中种群1的目标变量与决策变量均采用高斯(Gaussian)变异，种群2的目标变量采用柯西(Cauchy)变异、决策变量采用高斯(Gaussian)变异。两个种群同时进行进化，求解复函数方程的最优解。

(1) 确定复函数方程根的个体的表达方式：表达式中个体由目标变量 X 和标准差σ两部分组成，其中每个个体有 2 个分量，分别代表复函数方程根的实部和

虚部，即$(X,\sigma)=((x_1,x_2),(\sigma_1,\sigma_2))$。

(2) 随机生成所求复函数方程的根的 2 个初始群体：初始群体 1 与初始群体 2 均由μ个个体组成，每一个个体(X,σ)包含 2 个分量，分别对应于复函数方程的根的实部和虚部，初始个体是随机生成的，初始个体的标准差$\sigma(0)=3.0$。

(3) 计算适应度：将种群 1 与种群 2 的解代入复函数方程中去，我们记所得的值为：e_p $(p=1,2)$，若e_p的值的越接近于 0，则表示根越优异；取适应度函数为：$f_p=1/(1+e_p^2)$，适应度值越接近 1，根就越优良，其中：$0<f_p<1$，终止条件选择一个很接近 1 的值ε，通过训练学习后再比较f_1与f_2的值，从它们中间选择最优的值与ε作比较，当适应度值大于ε时终止。

(4) 如果满足条件，则终止，此时从两个群体中选出最优的解。否则，转步(5)。

(5) 根据进化策略，采用下述操作 5.1)-5.4)产生两个新群体：

5.1) 重组：从两个群体的父代个体中分别随机取出两个个体，交换目标变量和随机因子，产生新个体。目标变量采用离散重组，随机因子采用黄金分割重组。

5.2) 突变：变异产生下一代

5.2.1) 种群 1 按照如下方式进行变异产生新个体：

$$\sigma'_i=\sigma_i\cdot\exp(\tau_2\cdot N(0,1)+\tau_1\cdot N_i(0,1))$$

$$x'_i=x_i+\sigma_i\cdot N_i(0,1)\text{，}$$

其中τ_1及τ_2取为 1，$N(0,1)$与$N_i(0,1)$是服从标准正态分布的随机数，$N_i(0,1)$是针对第i个分量重新产生一次符合标准正态分布的随机数，$i=1,2$；

5.2.2) 种群 2 按照如下方式进行变异产生新个体：

$$\sigma'_i(j)=\sigma'_i(j)\cdot\exp(\tau_2\cdot N(0,1)+\tau_1\cdot N_j(0,1))$$

$$x'_i(j)=x_i(j)+\sigma'_i(j)\cdot\eta_j\text{，}$$

其中η，η_j均为$t=1$时的一个柯西随机变量比例参数，η_j用于更新每一个分量，τ_1及τ_2取为 1，$N(0,1)$与$N_i(0,1)$是服从标准正态分布的随机数，$N_i(0,1)$是针对第i个分量重新产生一次符合标准正态分布的随机数，$i=1,2$。

5.3) 计算个体适应度f_p，$(p=1,2)$。

5.4) 选择：对群体 1 与群体 2 分别采用(μ,λ)选择策略，挑选出优良的个体作为进化的结果，得到新的群体 1 与群体 2。

(6) 反复执行第 (5)步，直到满足终止条件，选择最佳的个体作为进化策略的结果，即所求的复函数方程的最优解。

5.5 算法实现

见附录 4 源程序

5.6 仿真实例

取适应度函数表达式为： $f_p = 1/(1+e_p^2)$ $(p=1,2)$，其中 e_p 为近似解代入复函数方程后所得的结果；以下算例，均采用 (μ,λ) 选择进化策略，其中 $\mu=15, \lambda=7*\mu=105$。误差=| 精确解 — 结果 |，在例 1 中，终止条件取 $\varepsilon=0.999999999$，在例2中，终止条件取 $\varepsilon=0.99999999$；以下各例中的精确解均是由数学软件*Maple*求得。

例 1[20] $f(z)=z^5+(-4+10i)z^4+(7-40i)z^3+(4+70i)z^2+(-8+40i)z-80i$，设 $z=\rho(\cos\theta+i\sin\theta)$，$\rho\in[0,20]$，$\theta\in[0,2\pi)$。

利用上节的双种群变异算法，我们求得实例 1 的解，计算结果如表 5.1。

表 5.1 例 1 计算结果

Table5.1 Results of the example one

精确解	文献[9]结果	本文结果	本文误差
1.000000	1.000001+ 0.000001i	1.00000000000000 + 0.00000000000000i	0.00000000000000
–1.000000	–0.999999+ 0.000000i	-1.00000000000000 - 0.00000000000000i	0.00000000000000
2.000000+ 2.000000i	2.000001+ 1.999999i	2.00000000000000 + 2.00000000000000i	0.00000000000000
2.000000– 2.000000i	1.999999– 1.999998i	2.00000000000000 - 2.00000000000000i	0.00000000000000
–10.00000i	0.000001– 9.999997i	0.00000000000000 -10.000000000000i	0.00000000000000

从图 5.2 中适应度值随迭代次数变化的 5 条曲线可以看出，在求解区间中，算法求得全部根，无一遗漏；并且收敛速度较快。所求结果与精确解间的误差较小，优于文献[20]中利用并行遗传算法所求的解。

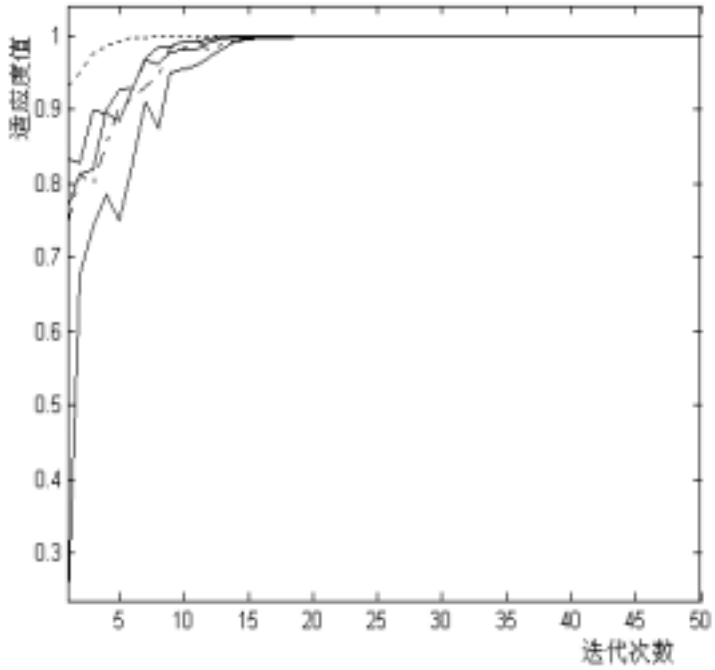

图 5.2 适应度值随迭代次数变化的曲线图

Fig.5.2 Fitness value's curve diagram changes along with iterative time

例 2[19] $f(z)=z^{13}+(1+i)z^{12}+iz^{11}+3iz^{10}+(7+3i)z^{9}+(7+i)z^{8}+(-3+i)z^{7}+(-3+8i)z^{6}+(-3+8i)z^{5}+(-3+7i)z^{4}+7iz^{3}+(-2i)z^{2}+(-8-2i)z-8$，

设 $z=\rho(\cos\theta+i\sin\theta)$，$\rho\in[0,20]$，$\theta\in[0,2\pi)$。

对于实例 2，利用上节的双种群变异算法，我们所求得的解如表 5.2 所示。

表 5.2 例 2 计算结果

Table5.2 Results of the example two

精确解	文献[10]结果	本文结果	本文误差
–1.0000000000+ 0.0000000000i	–1.00001+ 0.000001i	–1.00000000790456+ 0.00000001809236i	0.00000001809236
–1.0000000000+ 1.0000000000i	–1.000000+ 1.000000i	–1.00000000412579+ 0.99999999884065i	0. 00000000412579
1.0000000000+ 1.0000000000i	1.000000+ 1.000000i	1.00000000235773 + 0.99999999394174i	0.00000000605826
0.0000000000+ 1.0000000000i	0.000000+ 1.000000i	0.00000000007672 + 1.00000000125793i	0. 00000000125793
0.0000000000– 1.0000000000i	–0.000003– 0.999998i	–0.00000002175880– 1.00000000912971i	0. 00000000912971
–0.8660254040– 0.5000000000i	–0.866024– 0.499999i	–0.86602542466510– 0.50000000101605i	0.00000002066510
0.8660254040– 0.5000000000i	0.866023– 0.499999i	0.86602535249401– 0.50000009027903i	0. 50000009027903
–1.322875656– 1.500000000i	–1.322876– 1.500001i	–1.32287565577808– 1.50000000031466i	0.00000000031466
1.322875656– 1.500000000i	1.322876– 1.500000i	1.32287565498970– 1.49999999914271i	0.00000000101030
–0.9510565163– 0.3090169944i	–0.951053– 0.309015i	–0.95105651634145– 0.30901699631999i	0.00000000191999
0.9510565163– 0.3090169944i	0.951013– 0.309162i	0.95105659842298– 0.30901694733469i	0.00000004706531
–0.5877852523+ 0.8090169944i	–0.587783+ 0.809023i	–0.58778524352770+ 0.80901701383102i	0.00000101943102
0.5877852523+ 0.8090169944i	0.587782+ 0.809017i	0.58778526576034 + 0.80901701757314i	0.00000002317314

由图 5.3 中适应度值随迭代次数变化的 13 条曲线可以看出，在求解的区间上，由算法求得全部根，无一遗漏；并且收敛速度较快。所求得的解与精确解间的误差较小，并且优于文献[19]中利用遗传算法所求得的结果。

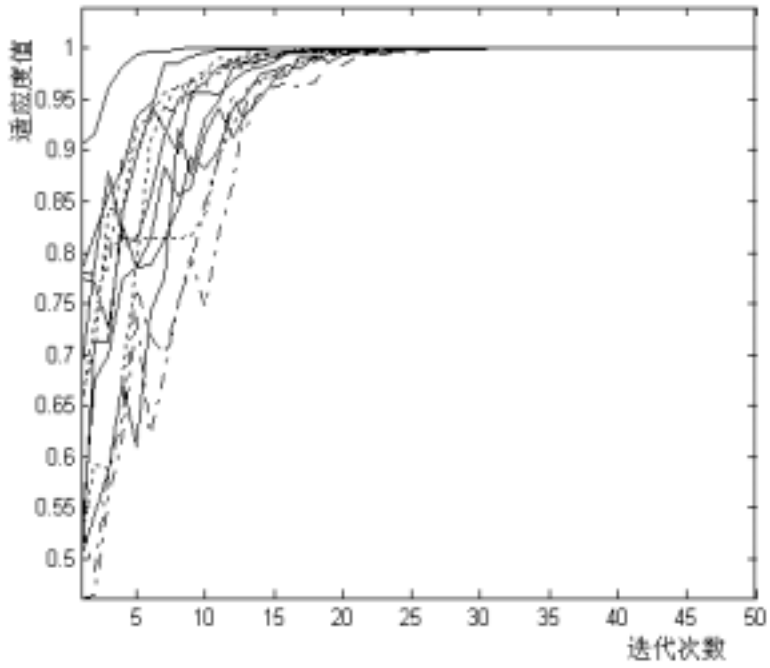

图 5.3 适应度值随迭代次数变化的曲线图

Fig.5.3 Fitness value's curve diagram changes along with iterative time

5.7 本章小结

本章主要是提出了一种新的求解复函数方程全部解的双种群进化策略，对各种群中的变异个体作了改进，两个种群采取不同的变异算子。从上面的算例可以看出，该算法具有求解精度高，收敛速度快等优点，是求解复函数全部根的好算法。

6 混合进化策略在多峰值函数优化中的应用

6.1 引 言

多峰值函数的优化问题在实践中大量存在，是函数优化问题的一个重要方面。如神经元的结构及权重优化，模糊系统结构和参数优化，最优控制律设计，复杂系统及结构辨识等,归根到底都是一些多峰值函数的优化问题。因此，研究有效快速的函数优化算法，具有重要的理论意义和应用价值。

在对多峰目标函数进行全局寻优时，多数优化算法容易陷入局部最优；一些算法若初始值选择不当,很容易陷入局部最优，从而影响优化效果。多峰优化即是寻找优化目标函数定义域空间的前 N 个或所有的局部最优值。对多峰函数，常用的寻优方法，如梯度法、模拟退火法(SA)、禁忌搜索(TS)等，往往会陷入局部最优，难以找到全局最优值。比较成功的多峰函数优化算法有遗传算法(GA)[53–54]、人工免疫算法(AIA)[55–58]、蚁群算法(ACA)[59–62]和微粒群算法(PSO)[63–65]等。

为了提高算法的全局搜索能力和收敛速度，本章基于 ES 算法，结合 DEA 算法两种变异方式的特点，引入模拟退火策略，将两种变异方式进行自适应的线性组合，提出了一种基于进化策略的改进差分退火因子双重变异寻优新算法(A Novel Improved Bi-mutation Differential Evolution Strategy Algorithm, IBDESA)。仿真结果表明，该算法计算简捷，寻优效果良好，可有效地应用于多峰函数的优化问题中。

6.2 改进的差分演化算法

DEA 的基本操作包括变异、交叉和选择三种操作，但与其它进化算法不同。DEA 由 N 个 D 维的参数矢量 $X_i^t(i=1,2,\cdots,N)$ 构成种群在搜索空间进行寻优，其中 t 表示第 t 代。首先由父代两个不同随机个体相减得到的差分矢量加到第三个个体上，生成一变异个体，接着按照一定的概率，父代个体与变异个体之间进行交叉操作，生成一试验个体，然后在父代个体与试验个体之间根据适应值的大小进行选择操作，选择适应度更优的个体作为子代。

DEA 最基本的变异成分是父代的差分矢量，每个矢量对包括父代两个不同的个体 (X_{r1}^t, X_{r2}^t)。根据变异个体的生成方法不同，形成了多种不同的差分演化算法方案[39]，其中 DEA/rand/1 和 DEA/best/1 方案个体变异操作的方程为：

$$X_m = X_{r3}^t + F*(X_{r1}^t - X_{r2}^t)\,, \tag{6.1}$$

$$X_m = X_{gbest}^t + F*(X_{r1}^t - X_{r2}^t)\,, \tag{6.2}$$

(6.1)式和(6.2)式中 X_{r1}^t、X_{r2}^t、X_{r3}^t 为互不相同的随机个体，X_{gbest}^t 为种群中适应值最好的个体，$F \in [0,2]$，为缩放因子。由(6.1)式可知，变异个体 X_m 由三个互不相同的随机个体组成，无需任何适应值信息，有利于保持种群的多样性，因而全局搜索能力强，但收敛速度慢；由(6.2)式可知，变异个体 X_m 由 X_{gbest}^t 作引导，因而局部搜索能力强，精度高，收敛速度快，但会加大算法陷入局部最优点的可能性。结合这两种不同变异方式的特点，在变异方程中同时考虑随机个体 X_{r3}^t 和最优个体 X_{gbest}^t 的作用，本文提出一种新的变异方案，其变异操作方程为：

$$X_m = \alpha*X_{r3}^t + \beta*X_{gbest}^t + F*(X_{r1}^t - X_{r2}^t)\,, \tag{6.3}$$

式中 $\alpha+\beta=1$，$\alpha \in [0,1]$。若 $\alpha=1$，则(6.3)式等价于(6.1)式，变成 DEA/rand/1 方案；若 $\alpha=0$，则(6.3)式等价于(6.2)式，变成 DEA/best/1 方案。对于一个良好的算法来说，要求在初始阶段有较强的全局搜索能力，尽可能发现较多的可能全局最优点，而在后阶段则应该有较强的局部搜索能力，这样可以提高算法的求解精度和收敛速度。因此，在本节中引入模拟退火策略，将 α 设置为退火因子，如下式所示：

$$\alpha = \frac{T-t}{T}\,, \tag{6.4}$$

式中 T 表示最大迭代次数，t 表示当前迭代次数。α 在搜索过程中由 1 逐渐变化为 0，使得 X_{r3}^t 的权重逐渐减小而 X_{gbest}^t 的权重逐渐增大。因此在进化过程中，前期进行全局搜索，后期进行局部搜索，从而保证算法既有较强的全局搜索能力又有较快的收敛速度和搜索精度。

6.3 改进差分进化策略算法多峰值函数优化流程

(1) 确定个体的表达方式：表达式中个体由目标变量 X 和标准差 σ 两部分组成，因为所考虑的函数为 N 维，所以每部分个体有 N 个分量，即 $(X,\sigma)=((x_1,x_2,\cdots,x_N),(\sigma_1,\sigma_2,\cdots,\sigma_N))$。

(2) 随机生成初始群体：进化策略中初始群体由 μ 个个体组成，每一个个体 (X,σ) 包含 N 个分量，其中 $(x_1,x_2,\cdots,x_N)$ 为与多维函数相对应的自变量的取值，X 内的每一个分量 $x_i(i=1,2,\cdots,N)$ 为给定定义域 $[A,B]$ 内的数。初始个体是随机生成的，初始个体的标准差 $\sigma(0)=3.0$。

(3) 计算适应度：取适应度函数为 $f=1/(1+d)$，其中 $d=abs(V-g(X))$，V 为精确值、$g(X)$ 为多维函数的近似值，若适应度值越接近 1，则表示与函数值相对应的自变量 X 越优，即函数值越优。其中：$0<f<1$，终止条件选择一个很接

近 1 的值ε，当适应度值大于ε时终止。

(4) 如果满足条件，则终止，此时选出最优的自变量取值，从而得出最优的函数值。否则，继续向下进行学习。

(5) 根据进化策略，采用下述操作产生新群体：

5.1) 重组：从父代个体中随机取出两个个体，交换目标变量和随机因子，产生新个体。目标变量采用离散重组，随机因子采用黄金分割重组。

5.2) 突变：对重组后的个体添加随机变量，按照如下方式进行变异产生新个体：

$$\sigma'_i = \sigma_i \cdot \exp(\tau_2 \cdot N(0,1) + \tau_1 \cdot N_i(0,1))$$

$$x'_i = x_i + \sigma_i \cdot N_i(0,1)\ ,$$

其中τ_1及τ_2取为 1，$N(0,1)$与$N_i(0,1)$是服从标准正态分布的随机数，$N_i(0,1)$是针对第i个分量重新产生一次符合标准正态分布的随机数，$i = 1,2,\cdots,N$；

5.2.1) 将所有个体里的分量$x'_i(i = 1,2,\cdots,N)$与定义域$[A,B]$的两个区间端点A、B进行比较；

5.2.2) 如果$x'_i > B$，则$x''_i = 2 * B - x'_i$；

5.2.3) 如果$x'_i < A$，则$x''_i = 2 * A - x'_i$；

5.2.4) 重复 5.2.2)-5.2.3)，直到$x''_i \in [A,B]$。

5.3) 计算新个体的适应度。

5.4) 选择：采用(μ,λ)选择策略，挑选出λ个优良的个体作为此代进化得到的最优自变量取值。

(6) 利用差分演化算法，采用突变操作产生λ个新个体；

6.1) 突变：对上面由进化策略算法所得到的λ个最优自变量按照如下方式再次进行变异产生新个体：

$$X_m = \alpha * X^t_{r3} + \beta * X^t_{gbest} + F * (X^t_{r1} - X^t_{r2})$$

其中$\alpha = \dfrac{T-t}{T}$(T为最大迭代次数)，$\alpha + \beta = 1$，$\alpha \in [0,1]$，$F \in [0,2]$，$m = 1,2,\cdots,\lambda$，$r = 1,2,\cdots,\lambda$。

6.1.1) 将λ个个体里的各个分量$x'_i(i = 1,2,\cdots,N)$与定义域$[A,B]$的两个区间端点A、B进行比较；

6.1.2) 如果$x'_i > B$，则$x''_i = 2 * B - x'_i$；

6.1.3) 如果$x'_i < A$，则$x''_i = 2 * A - x'_i$；

6.1.4) 重复 6.1.2)-6.1.3)，直到$x''_i \in [A,B]$。

6.2) 计算新个体的适应度。

(7) 反复执行第(5)-(6)步，直到满足终止条件，选择最佳的个体作为进化的结

果，即为最优的自变量取值。

(8) 将最优自变量代入函数表达式得出最优的函数值。

6.4 算法实现

见附录 5 源程序

6.5 仿真实例及结果分析

为了验证改进的差分进化策略双重变异新算法在多峰值函数优化问题中的正确性，适应度函数表达式取为： $f=1/(1+d)$，其中 $d=abs(V-g(X))$，V 为精确值、$g(X)$ 为多维函数的近似值；以下算例，均采用 (μ,λ) 选择策略。由上面算法的思想知，当 f 的值越接近1时，则表示最终所求的最优解与精确解间的误差越小，误差=|精确解—结果|。

6.5.1 测试函数和运行参数

选取典型的Benchmark优化问题作为测试算例。

算例1 求解 n 维的 *Sinc* 函数的最大值，其目标函数为：

$$\max f(X)=\frac{\sin(\sum_{i=1}^{n}|x_i-5|)}{\sum_{i=1}^{n}|x_i-5|}, \qquad x_i\in[1,10]$$

该算例中取 $\mu=15,\lambda=7*\mu=105$，最大迭代次数 $T=80$，缩放因子 $F=1.5$，终止条件 $\varepsilon=0.9999999999$。

算例2 求解 n 维的 $Multi\bmod al$ 函数的最大值，其目标函数为：

$$\max g(X)=900-\sum_{i=1}^{n}[(x_i-5)^2-10\cos(2\pi(x_i-5))], \qquad x_i\in[1,10]$$

该算例中取 $\mu=30,\lambda=7*\mu=210$，最大迭代次数 $T=150$，缩放因子 $F=1.5$，终止条件 $\varepsilon=0.9999$。

6.5.2 测试结果与分析

利用改进的差分进化策略双重变异新算法对 *Sinc* 与 $Multi\bmod al$ 函数进行寻优，为了便于与文献[22]、[66]的结果做比较，对 7 维的 *Sinc* 函数和 10 维 $Multi\bmod al$ 函数进行寻优，试验多次，每次都能够收敛到全局最优点。算法运行 30 次，给出与文献[22]、[66]相同的统计指标来衡量新算法的寻优性能。所涉及的统计指标为：最优解(best value, BV)、最差解(worst value, WV)、平均解(average value, AV)。算例 1 的统计结果如表 6.1 所示：

表 6.1 7 维 *Sinc* 函数的优化求解统计指标结果及相互比较

Table6.1 The optimized solution statistics target result of the seven dimension *Sinc* function and mutual comparison

算法	*Sinc*		
	BV	WV	AV
IBDESA	0.99999999999949	0.99999999990340	0.99999999996081
SFIA	1.000	0.998	0.999
IA	1.000	0.128	0.563
GA	0.128	0.071	0.094

Sinc 函数的全局最优值 $f_{best}=1$，文献[66]中对于 *Sinc* 函数目标函数值达到 0.999 才求解成功，而利用改进算法求最优解时，*Sinc* 函数的目标函数值 f^* 与其全局最优解 f^{best} 之间的差值达到10^{-10} 算求解成功。由表 6.1 中的各个算法所求得的结果可知改进算法在求解多峰值问题时性能较好，优于其他三种算法。文献[25]所求最优解为 0.9999999999350313，最差解为 0.9999999883908636；由此可知，改进算法在求解 *Sinc* 的最大值时优于文献[25]中所求的解。

算例 2 的统计结果如表 6.2 所示：

表 6.2 10 维 $Multi\bmod al$ 函数的优化求解统计指标结果及相互比较

Table6.2 The optimized solution statistics target result of the ten dimension $Multi\bmod al$ function and mutual comparison

算法	$Multi\bmod al$		
	BV	WV	AV
IBDESA	999.9999936868053	999.9999141469248	999.9999464209776
SFIA	999.87	995.19	998.46
IA	999.92	994.85	996.67
GA	999.95	985.56	995.31

$Multi\bmod al$ 函数的全局最优值 $g_{best}=1000$，文献[66]中对于 $Multi\bmod al$ 函数目标函数值达到 995 算求解成功，而利用改进算法求最优解时，$Multi\bmod al$ 函数的目标函数值 g^* 与其全局最优解 g^{best} 之间的差值达到10^{-4} 算求解成功。由表 6.2 中的各个算法所求得的结果可知本节算法在求解多峰值问题时性能较好。文献[25]所求得的最优解为 999.9991228003522，最差解为 999.9964969663567；从计算结果可看出，提出的改进算法在求解 $Multi\bmod al$ 的最大值时优于文献[25]中所求的解。

6.6 本章小结

本章主要是在对进化策略算法进行了分析后，针对它所存在的缺陷，将进化策略算法与差分演化算法相混合，同时引入模拟退火算子，提出一种基于进化策略的改进差分演化双重变异算法，这种混合算法在多峰值函数优化问题中表现出求解精度较高、收敛速度较快等特点。

7 多维函数逼近

7.1 引 言

函数逼近问题可以描述为：如何确定一个函数集合Φ，使得特定空间A上的任意函数f可由Φ中函数近似地构造出来。存在的基本问题是：(1)如何选择Φ，(2)如何基于Φ中函数构造近似函数,使其与目标函数有尽可能小的偏差[26]。函数逼近技术在系统辨识、最优控制等领域有着广泛的应用。因此，探索函数逼近的理论与技术具有重要的现实意义。对函数逼近论的研究，已经产生了许多成熟的函数逼近方法，例如代数多项式逼近、三角多项式逼近、插值方法逼近、样条函数逼近等[26−28]。经典的方法通常是采用回归分析方法，即已知采用样本数据集合$\{(x_i, y_i) \mid x_i \in R^n, y_i \in R, i \in N\}$，构造满足采样数据点误差等约束条件的函数，但存在着相关却不同的其它问题，如在控制等领域，需要求解特定模型，而该模型是否适当，只有当模拟执行该模型后依据其适合性才能评判。回归分析并不适合于求解该类问题,其根本原因在于所采用的样本数据来源往往受到专家领域知识的限制，导致所求解的特定模型与实际模型出现偏差，甚至产生错误的结果。机器学习方法的发展为解决该问题提供了一个新的思路，它们不仅可以用于数据回归，而且还可用于自适应建模，使逼近技术的应用领域得以拓广。

理论上已经证明,人工神经网络(ANN)可以逼近任意连续实函数[29−30][67]。尽管在解决许多具体问题时取得了一定成功，但ANN作为通用函数逼近方法仍存在一些问题。在设计特定类型的ANN时，网络的结构和规模需要人工确定，另外还须合理地选择网络的初始权值，且在学习过程中容易陷入局部极小，而导致逼近能力有限等问题，因此在一些复杂系统建模时，常存在较大的建模误差。

在文献[35]提出了一种多维函数的通用进化逼近方法，在函数逼近方面取得了一定的成功。但作为进化算法本身，编码是进化算法应用时所要解决的首要问题。针对一个具体的应用问题，如何设计一种完美的编码方案，一直是进化算法的应用难点之一。此外，目前还没有一套既严密又完整的指导理论及评价准则能够帮助设计编码方案。基于此，提出了一种新的基于泛函网络的进化策略修正基函数系数的多维函数逼近理论及学习算法。

根据特定的问题，选择不同的基函数族(如多项式、三角函数等)，来满足不同问题的建模，并利用进化策略来训练学习基函数的系数，实现函数的逼近。这种基于泛函网络的进化策略修正基函数系数的多维函数逼近方法，相比目前所使用的函数逼近方法，具有收敛速度快，逼近能力、泛化能力强等优点，这为多维

函数逼近提供了一种新方法。最后的仿真试验表明了这种方法的有效性。

7.2 泛函网络与进化策略

泛函网络与标准神经网络之间最明显的区别是：在标准神经网络系统中，给定神经网络的激活函数(常用的激活函数有Sigmoid函数、三角波状函数、梯形函数和RBF等)，而对激活函数的连接权值进行训练。在泛函网络系统中，没有连接权值，泛函网络训练的是函数 F ，也即对激活函数进行训练。在标准神经网络系统中，输出单元是单个神经元的输出(最后一个单元)，而在泛函网络系统中，输出单元可输出一个或多个神经元，这隐含了这样一种相容性，即：连接到输出单元的神经元个数越多，输出神经元的自由度越少。泛函网络的一个重要特性是能处理连续模型，与初值选举无关等。这些特性为研究多维函数的逼近奠定了理论基础。

进化策略与遗传算法、遗传规划相比，主要有以下几个特征：①进化策略从 λ 个新个体或从 $(\mu+\lambda)$ 个个体中挑选 μ 个个体组成新群体，而且选择方法是确定型的；②进化策略直接以问题的可行解作为个体的表现形式，无需再对个体进行编码处理，也无需再考虑随机扰动因素对个体的影响，这样就更便于其应用；③进化策略以 n 维实数空间上的优化问题为主要处理对象。这些特性为研究多维函数逼近中的系数优化奠定了理论基础。

7.3 进化泛函网络函数逼近模型

泛函网络与神经网络一样，有许多结构，不可能用一个传统的通用的结构来描述所有的泛函网络，也不可能用一个统一的函数来表示所有的泛函网络。在泛函网络研究中，应用最广泛的是可分离泛函网络[68]。本节基于可分离泛函网络的模型，构建了一种用于函数逼近的可分离进化泛函网络，如图7.1所示(此图给出的是一个二维拓扑逼近模型，可以推广到多维的情形)。

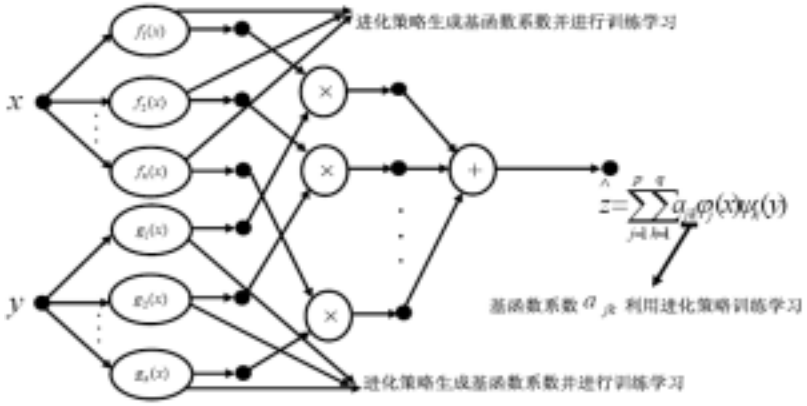

图7.1 可分离的进化泛函网络函数逼近模型

Fig.7.1 The function approach model of the separable evolutionary functional network

图7.1所示的是一个两个输入 x ， y ，一个输出 $\hat{z}$ 的二维可分离进化泛函网络模型(此网络模型可以推广到多维的情形)。该网络的输出表达式如下所示：

$$\begin{aligned}\hat{z} &= \hat{F}(x,y) = \sum_{i=1}^{n} f_i(x) g_i(y) \\ &= \sum_{i=1}^{n}((\sum_{j=1}^{p} b_{ij} \cdot \varphi_j(x)) \cdot (\sum_{k=1}^{q} c_{ik} \cdot \psi_k(y))) \\ &= \sum_{i=1}^{n}(\sum_{j=1}^{p}\sum_{k=1}^{q} b_{ij} \cdot c_{ik} \cdot \varphi_j(x) \cdot \psi_k(y)) \\ &= \sum_{j=1}^{p}\sum_{k=1}^{q} a_{jk} \varphi_j(x) \psi_k(y)\end{aligned} \tag{7.1}$$

其中：

$$f_i(x) = \sum_{j=1}^{p} b_{ij} \varphi_j(x) \tag{7.2}$$

$$g_i(y) = \sum_{k=1}^{q} c_{ik} \psi_k(y) \tag{7.3}$$

$i = 1,2,\cdots,n$ ， $j = 1,2,\cdots,p$ ， $k = 1,2,\cdots,q$ ， p 为与变量 x 相对应的基函数的个数，q 为与 y 相对应的基函数的个数。

图7.1的逼近模型所得到的表达式(7.1)，它是一个典型的分片逐元插值函数的形状。通过泛函网络与进化策略算法对基函数 $\varphi_j(x)$ 与基函数 $\psi_k(y)$ 及基函数前的系数进行学习，可使 $\hat{F}(x,y)$ 尽可能地逼近目标函数 $F(x,y)$ ，从而实现对多维函数的逼近。

7.4 多维函数的进化泛函网络逼近算法流程

7.4.1 问题分析

上节给出了基于函数逼近的进化泛函网络模型，其模型输出表达式的几何意义表示是一个典型的分片逐元插值函数。另一方面，对于任意一分片逐元插值函数，可用图7.1的进化泛函网络来逼近。因此，在进化泛函网络逼近的意思下，函数的逼近问题可以重新描述如下：如何通过学习来确定一组基函数集 Φ 及最优基函数系数，使得特定空间 S 上的任意函数 f 可由 Φ 中函数近似地构造出来。但还存在一些基本问题：①如何选择基函数集 Φ ；②选用何种突变因子来对系数进行学习；③如何基于基函数集 Φ 中的函数构造近似函数，使其与目标函数之间的误差尽可能的小，利用进化泛函网络来实现近似函数与目标函数之间的误差。

对于有教师训练学习算法来讲，给定一组学习样本数据：

$\{((x_i, y_i), z_i) \mid i = 1,2,\cdots,m\}$，此时考虑的是一个二维(可以推广到多维的情形)的逼近，m 个学习样本数据。

适应度函数取为：

$$\hat{f} = \frac{1}{1+\sum_{i=1}^{m}\frac{(z_i - \hat{z}_i)^2}{20}} \tag{7.4}$$

其中：

$$\hat{z}_i = \hat{F}(x_i, y_i) = \sum_{j=1}^{p}\sum_{k=1}^{q} a_{jk}\varphi_j(x_i)\psi_k(y_i) \tag{7.5}$$

为进化泛函网络的输出。

本章在利用进化策略对基函数的系数进行学习时，假设对应于 x 与 y 的基函数的个数分别为 p 与 q 个；在变异过程中采用高斯单基因突变与柯西单基因突变相结合的方法，将柯西变异的全局搜索性和高斯变异的局部搜索性有机地结合起来，这样可以加快收敛的速度，使得基于进化泛函网络所求得的逼近函数与目标函数间的误差较小。

7.4.2 算法流程

根据算法的模型及分析，给出进化泛函网络二维(可以推广到多维的情形)函数逼近算法，步骤如下：

Step1: 利用进化泛函网络对基函数进行学习，同时生成基函数系数个体，个体由目标变量 X 和标准差 σ 两部分组成，每部分个体含有 $p \cdot q$ 个分量；初始群体由 μ 个个体组成，标准差 $\sigma = 3.0$。

Step2: 计算适应度 $\hat{f}$ 的值，$0 < \hat{f} < 1$。取一个很接近 1 的值 ε，当 $\hat{f}$ 大于 ε 时程序终止，选出最优解，跳出；否则，向下进行学习。

Step3: 对基系数采用黄金分割重组；利用单基因高斯、柯西突变，生成 2λ 个个体，高斯突变按照式 $\sigma'_i = \sigma_i \cdot \exp(r' \cdot N(0,1) + r \cdot N_i(0,1))$ 与 $x'_i = x_i + \sigma_i \cdot N_i(0,1)$ 产生 λ 个新个体；柯西突变按照式 $\sigma'_i = \sigma_i \cdot \exp(r' \cdot N(0,1) + r \cdot N_i(0,1))$ 与 $x'_i = x_i + \sigma'_i \cdot \eta_i$ 产生 λ 个新个体，参数 η_i 是当 $t = 1$ 时的一个柯西分布的随机变量比例参数，其中 r 及 r' 取值 1，$i = 1,2,\cdots, p \cdot q$；计算适应度值 $\hat{f}$；利用 $(\mu, 2\lambda + \mu)$ 选择策略，挑选出最优良的个体作为进化的结果。

Step4: 反复执行 **Step3**，同时对基函数进行学习，直到满足终止条件，此时选择出最佳的基函数系数 $\hat{a}_{jk}$，将最优系数向量与基函数向量求内积，则可得目

标函数 $F(x,y)$ 的最佳逼近表达式为：$\hat{F}(x,y)=\sum_{j=1}^{p}\sum_{k=1}^{q}\hat{a}_{jk}\,\varphi_j(x)\psi_k(y)$。

7.5 算法实现

见附录 6 源程序

7.6 仿真实例及结果分析

为了验证算法的有效性，对下面的二维函数[68] $F:R^2\to R$ 实施逼近，对基系数采用 $(\mu,2\lambda+\mu)$ 选择策略。

二维函数为：$F(x,y)=1+x-y-x^2y$，随机地产生一组训练样本数据，如表7.1所示。

表7.1 用于逼近 $z=F(x,y)$ 的样本数据

Table7.1 The swatch data to approach $z=F(x,y)$

z	x	y	z	x	y
0.880	0.384	0.439	1.072	0.136	0.062
0.299	0.309	0.922	1.449	0.457	0.006
0.359	0.449	0.907	0.604	0.818	0.727
0.681	0.673	0.682	0.760	0.663	0.627
1.019	0.387	0.320	0.989	0.405	0.357
1.117	0.976	0.440	1.369	0.801	0.263
0.873	0.725	0.558	1.214	0.377	0.143
0.935	0.058	0.122	0.669	0.799	0.689
1.336	0.697	0.242	1.228	0.862	0.364
0.556	0.662	0.762	0.814	0.028	0.214

(1) 选取基函数 $\varphi(x)=[1,x,x^2]$、$\psi(y)=[1,y,y^2]$，$\mu=10,\lambda=7*\mu=70$，终止条件为 $\varepsilon=0.99999$，利用上面的学习算法，可以得到 $F(x,y)$ 的近似函数 $\hat{F}(x,y)$ 表达式如下所示：

$$\begin{aligned}\hat{F}(x,y)=&\,0.99759899900021+1.00906999001003x-0.01061900900101x^2-\\&0.98359990500004y-0.06859991430010xy-0.92849985000206x^2y-\\&0.01659996040030y^2+0.07306999980006xy^2-0.07764000010020x^2y^2\end{aligned}$$

此时适应度函数值为 $f=0.99999538507894$。

(2) 若选取基函数 $\varphi(x)=[1,x,x^2]$、$\psi(y)=[1,y]$，$\mu=10,\lambda=7*\mu=70$，终止条件为 $\varepsilon=0.99999$，利用上面的学习算法，可以得到 $F(x,y)$ 的近似函数 $\hat{F}(x,y)$ 表达式如下所示：

$$\hat{F}(x,y)=0.999240908009+1.000100011234x+0.000995000101x^2-0.996199998908y-0.010851299986xy-0.993399898609x^2y$$

此时适应度函数值为 $f=0.99999538333906$。

(3) 当选取基函数 $\varphi(x)=[1,x,x^2,x^3]$、$\psi(y)=[1,y,y^2,y^3]$，$\mu=10,\lambda=7*\mu=70$，终止条件为 $\varepsilon=0.9$，利用上面的学习算法，可以得到 $F(x,y)$ 的近似函数 $\hat{F}(x,y)$ 表达式如下所示：

$$\begin{aligned}\hat{F}(x,y)=&0.83490748218401+1.87545131868690x-0.04123578355078x^2-\\&1.10697926721172x^3+0.45948141921084y-1.25434759240175xy-\\&1.61036321083995x^2y+1.01741312158433x^3y-0.75864696715894y^2-\\&0.63868943457644xy^2+0.03739253262153x^2y^2+0.56536941877224x^3y^2+\\&1.45902382399360xy^3-1.38927833969981x^2y^3-1.85463480706022x^3y^3\end{aligned}$$

此时适应度函数值为 $f=0.94591277989646$，与(1)、(2)中的结果相比，此时的逼近效果较差，原因在于基函数选取的不合适。

从上面的实例讨论可以看出，相比以前的逼近方法，基于进化泛函网络的函数逼近的主要优点是：①计算原理和机制不同,将泛函网络与进化策略两者的优点有机地结合在一起；②进化泛函网络中基函数类与基函数的系数是通过学习来完成的；③进化泛函网络具有自学习、隐含并行性、自适应、全局搜索性等特点；④将进化泛函网络的自适应性与实际问题的“ 经验知识 ”相结合,可以避免因采样数据不足而导致的逼近函数与目标函数间的误差过大,以及因采样数据过多而出现的过配现象；⑤若基函数选得恰当合理,利用进化泛函网络所得到的逼近效果会更好。

7.7 本章小结

本章就进化策略所存在的缺陷，将泛函网络与进化策略相结合，集它们的优点于一起，提出了一种用于多维函数逼近的进化泛函网络逼近方法，并设计出了一种新的进化泛函网络逼近模型，该种新的逼近方法简单可行,具有较快的收敛速度和良好的逼近性能。

8 结束语

数值计算是数学的一个重要分支，它的研究对象是利用计算机求解各种数学问题的数值方法及有关理论。内容包括非线性方程(组)的数值解法，函数的数值逼近(插值与拟合)，数值积分与数值微分等。众所周知，在传统的数值计算方法中存在许多问题，如：计算矩阵特征值、特征向量时，传统的方法在计算时存贮量较大，计算速度较慢，普遍存在计算精度低、收敛速度慢及泛化能力弱等缺陷；在化学方程式的配平中，传统的方法普遍采用一种“试凑”的方法，这种方法往往与人们的所处专业领域的“先验知识”相关，有较大的局限性；求解复函数方程根时，常用的有迭代法、牛顿法、下山法、转换法、圆盘迭代法等，但是上述各种算法都存在着一定的局限性，有时会存在漏解的情形；在处理多峰函数优化问题时，无法同时找到所有的极值点；函数逼近，经典的方法通常是采用回归分析方法，即已知样本数据集合，构造满足采样数据点误差等约束条件的函数，但存在着相关却不同的领域问题，所采用的样本数据来源往往受到专家领域知识的限制，导致所求解的特定模型与实际模型出现误差，甚至产生错误的结果。本文主要采用进化策略，同时将进化策略算法与差分演化算法、泛函网络相结合来研究数值计算，把数值计算问题转化为函数优化问题以达到解决数值计算问题的目的。从文中可以看出，利用进化策略并行搜索和全局收敛的特性能有效的解决上面提到的问题。所以采用进化策略、泛函网络、差分演化算法来研究数值计算，有很大的理论和实际意义。

本文重点研究了进化策略在数值计算中的应用，主要做了以下几个方面的工作：

(1) 利用标准进化策略算法求解矩阵的全部特征值及与实特征值相对应的特征向量；

(2) 利用改进的进化策略新算法配平化学方程式、求解复函数方程的所有解；

(3) 利用混合进化策略算法研究多峰值函数优化问题及函数逼近问题。

受笔者学术水平、资料和时间等方面的限制，本文的工作和成果只是初步的，恳请老师、专家、同事们提出批评，以改进本论文的研究，展望今后进一步研究的方向。本文是把进化策略算法、差分演化算法、泛函网络应用到数值计算中，由于算法的理论还不太成熟，所以还要在理论证明方面进行深入的研究，这样对于应用来说更有充分的理论支持。另外，本文只是利用进化策略对一些数值问题

进行了研究，关于把进化策略算法应用到数值计算的其它方面有待进一步研究。

参考文献

[1] 蒋尔雄. 对称矩阵计算[M]. 上海: 上海科学技术出版社, 1984, 33-111.

[2] Davidson E R.The iterative calculation of a new of the lowest eigenvalues and corresponding eigenvectors of large real symmetric matrices[J]. J compute Phys, 1975, (17): 87-94.

[3] C.Bischof, S. Huss-Lederman, X. Sun, A. Tsao, and T. Turnbull. Parallel performance of a sym-metric eigensolver based on the invariant subspace decomposition approach. Technical report, Supercomputing Reserch Center, 1993.

[4] 罗晓广, 李晓梅. 解非对称矩阵特征值问题的一种并行分治算法. 高等学校计算数学学报, 1999(2): 140-149.

[5] Michael Grant. Eigenvectors via Excel. MSOR Connections, 2002, 2(2), May.

[6] 郑敏玲. 求解矩阵特征值问题的一种新的并行算法. 安徽师范大学学报, 2004, 27(1): 13-16.

[7] 杨廷俊. 矩阵特征值与特征向量的同步求解. 甘肃联合大学学报, 2006, 20(3): 20-22.

[8] 郭建敏. 特征值与特征向量同步求解[J]. 山西大同大学学报, 2007, 23(1): 15-18.

[9] 曹锡章, 宋天佑, 王杏乔, 等. 无机化学(上册)[M]. 北京: 高等教育出版社, 1998, 415-418.

[10] 叶斌. 化学反应方程式的配平方法[M]. 长沙: 湖南科技出版社, 1983.

[11] 王进贤, 张玉梅, 胡雨来. 应用代数法配平有机氧化还原反应方程式[J]. 西北师范大学学报, 1999, 35(2): 95-98.

[12] 吕申壮. 在 WORD 中添加“化学方程式自动配平和化学式格式化”工具[J]. 计算机与应用化学, 2001, 18(6): 591-595.

[13] 孙会霞, 职桂珍. 线性方程组的基本理论在配平化学方程式中的应用——改进的线性代数法[J]. 郑州轻工业学院学报, 2001, 16(4): 68-71.

[14] 杨海文, 高兴驰. 化学方程式配平的矩阵解法[J]. 延安大学学报, 2003, 22(3): 16-17.

[15] 张杨. 刍议氧化还原方程式反应的配平[J]. 科学教学研究, 2007, 6: 70.

[16] 王兰生, 程锦松. 求解复函数方程的根的串行和并行算法[J]. 安徽大学学报,

1999, 23(1): 53-57.

[17] Lars Kindermann, Achim Lewandowski, Peter Protzel. A framework for solving functional equations with neural networks[J]. Neural Information Processing, ICONIP2001 Proceedings, Fudan University, Shanghai, 2001(2): 1075-1078.

[18] 刘锋，陈国良，吴昊．求复函数方程根的遗传算法[J]．计算机工程与应用，2001(24): 18-19.

[19] 刘锋，陈国良，吴昊．基于遗传算法的方程求根算法的设计和实现[J]．控制理论与应用, 2002, 21(3), 467-469.

[20] 刘锋，陈国良，吴昊．用并行遗传算法求复函数方程根的设计和实现[J]．系统工程理论与实践, 2004(6): 61-66.

[21] Rainer S, Price K. Differential evolution---a simple and efficient heuristic for global optimization over Continuous Spaces[J]. J of Global Optimization, 1997, 11(4): 341-359.

[22] Jang-Sung Chun, Hyun-Kyo Jung, Song-Yop Hahn. A Study on Comparison of Optimization Performances Between Immune Algorithm and Other Heuristic Algorithms[J]. IEEE Transactions on Magnetic, 1998, 34(5): 2972-2975.

[23] 王柳毅，熊伟清．并行二进制蚁群算法的多峰函数优化[J]．计算机工程与应用, 2006(22): 42-45.

[24] 蒋忠樟，成浩．一种求解多峰函数优化问题的演化算法[J]．武汉大学学报(理工版), 2006, 52(3): 335-339.

[25] 靳宗信，刘光远，温万惠，贺一．一种改进的用于多峰值函数优化的自适应克隆选择算法[J]．西南大学学报(自然科学版), 2007, 29(3): 164-168.

[26] Lorentz G G. Approximation of functions[M]. Austin: Holt, Rinehart and Winston, 1996.

[27] 徐利治，王仁宏，周蕴时．函数逼近的理论与方法[M]．上海：上海科学技术出版社, 1983.

[28] 王仁宏，梁学章．多元函数逼近[M]．北京：科学出版社, 1988.

[29] Hornik K, Stinchcombe M, White H. Multiplayer feedforward networks are universal approximators [J]. Neural Networks, 1989, 2(3): 359-366.

[30] Hornik K, Stinchcombe M, White H. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks[J]. Neural Networks, 1990, 3(5): 551-560.

[31] Jean Gabriel Attali, Gilles Pages. Approximations of functions by a multilayer perception: a new approach[J]. Neural Networks, 1997, 10(6): 1069-1081.

[32] Hornik K, Stinchcombe M, White H. Multiplayer feedforward networks are universal approximators. Neural Networks, 1989, 2(3): 359-366.

[33] Ahmed Moataz A, De Jong Kenneth A. Function approximator design using genetic algorithms. In: Proceedings of the 1997 IEEE International Conference on Evolutionary Computation(ICEC'97), Indianapolis, 1997. 519-524.

[34] 荔建琦, 陈火旺, 王兵山. 多维函数的进化逼近[J]. 计算机学报, 2000, 23(6): 593-601.

[35] 卢宏涛, 戚飞虎. 多维离散傅立叶变换神经网络函数逼近[J]. 上海交通大学学报, 2000, 34(7): 956-959.

[36] 周永权, 赵斌, 焦李成. 基于泛函网络的多维函数逼近理论及学习算法[J]. 系统工程与电子技术, 2002(5): 906-909.

[37] Schwefel H P, Back T. Evolution Strategies I: Variants and Their Computetional Implementation. In: Genetic Algorithms in Engineering and Computer Science, Winter G(ed), Wiley, 1995, 111-126.

[38] Schwefel H P, Back T. Evolution Strategies II: Theoretical Aspects. In:Genetic Algorithms in Engineering and Computer Science, Winter G(ed), Wiley, 1995, 127-140.

[39] Storn R and Price K. Differential evolution---a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, ICSI, 1995.

[40] Storn R and Price K. Minimizing the real functions for the ICEC'96 contest by differential evolution. Processing of IEEE International Conference on Evolutionary computation, 1996: 842-844.

[41] Price K. Differential evolution a fast and simple numerical optimizer[C]//1996 Biennial Conference of the North American Fuzzy Information Processing Society. New York, 1996. 524-527.

[42] Rainer S, Price K. Differential evolution---a simple and efficient heuristic for global optimization over Continuous Spaces [J]. J of Global Optimization, 1997, 11(4): 341-359.

[43] Price K. Differential evolution vs. the functions of the 2nd ICEO[C]//IEEE International Conference on Evolutionary Computation. Indianapolis, 1997. 153-157.

[44] Castillo E. Functional Networks[J]. Neural Processing, Letters, 1998(7): 151-159.

[45] John H. Mathews, Kurtis D.Fink. Numerical Methods Using MATLAB (Fourth Edition)[M]. Beijing: Publishing House of Electronics Industry, 2005(4): 450-455.

[46] 同济大学计算数学教研室编著. 现代数值数学和计算[M]. 上海: 同济大学出版社, 2004, 221-224.

[47] 徐宗本. 进化智能（第一册）——模拟进化计算[M]. 北京: 高等教育出版社 2004(2): 41-43.

[48] 邵学俊，董平安，魏益海编著.无机化学下册[M]. 武汉: 武汉大学出版社，1996, 107-141.

[49] 刘新锦，朱亚光，高飞. 无机元素化学[M]. 北京: 科学出版社，2005, 164-166.

[50] 西安交通大学高等数学教研室. 复变函数[M]. 科学出版社, 1994.

[51] 程锦松. 求多项式全部根的遗传算法[J]. 微机发展, 2001, (1): 1-2.

[52] 王战权, 赵朝义, 云庆夏等. 进化策略中变异算子的改进研究[J]. 计算机仿真, 1999, 16(3): 8-11.

[53] Vose M D. The simple genetic algorithms: foundations and theory [M]. Boston: The MIT Press, 1999.

[54] 李敏强, 寇纪淞, 林丹等.遗传算法的基本理论与应用[M]. 北京: 科学出版社, 2002, 17-66.

[55] Dasgupta D. Artificial Immune Systems and Their Applications[M]. Berlin: Springer- Verlag, 1999.

[56] 葛红, 毛宗源.免疫算法几个参数的研究[J]. 华南理工大学学报(自然科学版) , 2002, 30(12) : 15-18.

[57] 葛红, 毛宗源. 免疫算法的实现[J]. 计算机工程, 2003, 29(5) : 62-63.

[58] Tmmis J. Artificial Immune Systems: A Novel Data Analysis Technique Inspired by the Immune Network theory[D]. Ceredigion, Walesmeyr, Aberystwyth: Department of Computer Science, University of Walse, 2000.

[59] Colorni A, Dorigo M, Maffioli Fetal. Heuristics from nature for hard combinatorial optimization problems[J]. Int Trans in Operational Research, 1996; 3(1): 1-21.

[60] Colorni A, Dorigo M, Maniezzo V. Distributed optimization by ant colonies [C]. In: Proc of 1st European Conf Artificial Life, Paris, France: Elsevier, 1991: 134-142.

[61] 张纪会, 高齐圣, 徐心和. 自适应蚁群算法[J]. 控制理论与应用, 2000,

17(1) : 1-8.

[62] 马良. 基于蚂蚁算法的函数优化[J]. 控制与决策, 2002, 17(Suppl): 719-726.

[63] Kennedy J, Eberhart R. Particle Swarm Optimization[C]. In: Proc IEEE, Int Conf on Neural Networks, Perth, 1995, 1942-1948.

[64] Eberhart R, Kennedy J. A new optimizer using particle swarm theory[C]. In: Proc 6th Int Symposium on Micro Machine and Human Science, Nagoya, 1995, 39-43.

[65] 谢晓锋, 张文俊, 杨之廉. 微粒群算法综述[J]. 控制与决策, 2003, 18(2): 129-134.

[66] Yan-jun Li, Tie-jun Wu. A Novel Immune Algorithm for Complex Optimization Problems[A]. Proceedings of the 5th World Congress on Intelligent Control and Automation[C]. Hang Zhou: IEEE, 2004: 2279-2283.

[67] White H. Connectionist nonparametric regression: Multilayer feed forward networks can learn arbitrary mappings. Neural Networks, 1990, 3(5): 535-549.

[68] Enrique Castillo, Angel Cobo, JosèManuel Gutièrrez, etal. Functional networks with applications[M]. Kluwer Academic Publishers, 1999.

附 录

附录 1：进化策略算法求矩阵特征值的主要源代码

```
X=2*rand(u,1)-1+(2*rand(u,1)-1)*i;    %生成初始群体
sigma(:,:)=3.0;
v=polyval(p,X);
f=1./(1+sqrt(abs(v)));
  while (m<epsilon)
       for k=1:lenda
       k1=randperm(u);
       XX(k)=X(k1(1));
     Sigma(k,:)=sigma(k1(1),:)*0.618034+sigma(k1(2),:)*0.381966; %黄金分割重组
     end
  Sigma=Sigma.*exp(r1*ra1+r*ra);
  XX=(real(XX)+Sigma(:,1).*ra(:,1))+(imag(XX)+Sigma(:,2).*ra(:,2))*i;    %突变
  v=polyval(p,XX);
  f=1./(1+sqrt(abs(v)));
  [f,index]=sort(f);
  X=XX(index(lenda-u+1:end));                                    %选出最优的一组解
  sigma=Sigma(index(lenda-u+1:end),:);
  g(t)=X(end);
  end
g(t)=-(p(end)/p(1));
```

附录 2：进化策略算法求矩阵特征向量的主要源代码

```
for i=1:n                              %找出实根
  if abs(imag(a(i)))<0.00001
     b(i)=real(a(i));
     c(i)=1;
  end
end
zhen=[b c];
```

```
shunxu=find(zhen(:,2)==1);
d=b(shunxu,1);
fcj=d(t)*eye(n)-A;
X=2*rand(u,n)-1;                    %生成初始群体
sigma(:,:)=3.0;
temp=X*fcj';
v=sum(temp.^2,2);
f=1./(1+v);
  while(m<epsilon)
        k1=floor(u*rand(lenda,2))+1;
        XX(:,:)=X(k1(:,1),:)*0.618034+X(k1(:,2),:)*0.381966;     %重组
        Sigma(:,:)=sigma(k1(:,1),:)*0.618034+sigma(k1(:,2),:)*0.381966;
        Sigma=Sigma.*exp(r1*ra1+r*ra);
        XX=XX+Sigma.*ra;                                          %突变
       temp=XX*fcj';
       vv=sum(temp.^2,2);
       f=1./(1+vv);
     [f,index]=sort(f);
     X=XX(index(lenda-u+1:end),:);                    %选出最优的一组解
     sigma=Sigma(index(lenda-u+1:end),:);
     m=f(end);
     g(t,:)=X(end,:);
   end
end
```

附录 3：改进进化策略算法配平化学方程式的主要源代码

```
X=floor(m0)+1;        %生成初始群体
sigma(:,:)=2.0;
temp=X*A;
v=sum(temp.^2,2);
f=1./(1+v);
   while(m<epsilon)
      M1=m;
      X0=X(end,:);
```

```
k1=floor(u*rand(lenda,2))+1;
      XX(:,:)=X(k1(:,1),:)*0.618034+X(k1(:,2),:)*0.381966;   %黄金分割重组
      Sigma(:,:)=sigma(k1(:,1),:)*0.618034+sigma(k1(:,2),:)*0.381966;
      Sigma=Sigma.*exp(r1*ra1+r*ra)+1;%对决策变量采用高斯突变
      for i=1:lenda            %对目标变量采用柯西突变
     ra23(i)=-1./tan(pi.*rand(1,1)+eps);
     XX(i,:)=floor(XX(i,:)+ Sigma(I,:)*ra23(i));
   end
   for i=1:lenda                %对个体的区间范围进行限制
    for j=1:n
       if XX(i,j)>M
       XX(i,j)=2*M-XX(i,j);;
       end
       if XX(i,j)<1
       XX(i,j)=2*1-XX(i,j);;
       end
    end
   end
    temp=XX*A;
    vv=sum(temp.^2,2);
    f=1./(1+vv);
    [f,index]=sort(f);
    m=f(end) ;
    X=XX(index(lenda-u+1:end),:);                %选出最优的一组解
   sigma=Sigma(index(lenda-u+1:end),:);
   g(1,:)=X(end,:);
   if    m<M1         %进行精英选择
   X(end,:)=X0;
   m=M1;
   end
   end
  g=g
```

附录 4：双种群进化策略算法求解复函数方程根的主要源代码

```
X1=2*rand(u,1)-1+(2*rand(u,1)-1)*i;          %生成初始群体1
X2=2*rand(u,1)-1+(2*rand(u,1)-1)*i;          %生成初始群体2
sigma1(:,:)=3.0;
sigma2(:,:)=3.0;
v1=polyval(p,X1);
f1=1./(1+sqrt(abs(v1))); %群体1的适应度值
v2=polyval(p,X2); %群体2的适应度值
f2=1./(1+sqrt(abs(v2)));
   while(m<epsilon)
   k1=floor(u*rand(lenda,2))+1;
   XX1(:,:)=X1(k1(:,1),:)*0.618034+X1(k1(:,2),:)*0.381966; %群体1黄金分割重组
   Sigma1(:,:)=sigma1(k1(:,1),:)*0.618034+sigma1(k1(:,2),:)*0.381966;
   k2=floor(u*rand(lenda,2))+1;
   XX2(:,:)=X2(k1(:,1),:)*0.618034+X2(k1(:,2),:)*0.381966; %群体2黄金分割重组
   Sigma2(:,:)=sigma2(k2(:,1),:)*0.618034+sigma2(k2(:,2),:)*0.381966;
   Sigma1=Sigma1.*exp(r1*ra12+r*ra11);
XX1=(real(XX1)+Sigma1(:,1).*ra11(:,1))+(imag(XX1)+Sigma1(:,2).*ra11(:,2))*i;
                                                  %群体1进行高斯突变
   v1=polyval(p,XX1);
   f1=1./(1+sqrt(abs(v1)));
   [f1,index1]=sort(f1);
   X1=XX1(index1(lenda-u+1:end));%群体1的一组最优解
   sigma1=Sigma1(index1(lenda-u+1:end),:);
   m1=f1(end);
    for i=1:lenda          %群体2进行柯西突变
    ra23(i)=-1./tan(pi.*rand(1,1)+eps);
    end
   ra23=[ra23 ra23];
    Sigma2=Sigma2.*exp(r1*ra22+r*ra21);
XX2=(real(XX2)+Sigma2(:,1).*ra21(:,1))+(imag(XX2)+Sigma2(:,2).*ra21(:,2))*i;
    v2=polyval(p,XX2);
    f2=1./(1+sqrt(abs(v2)));
    [f2,index2]=sort(f2);
    X2=XX2(index2(lenda-u+1:end)); %群体 2 的一组最优解
```

```
sigma2=Sigma2(index2(lenda-u+1:end),:);
    m2=f2(end);
    if m2<m1                %采用精英策略选取最优个体
     m=m1;
     g(t)=X1(end);
    end
  if m1<m2                  %采用精英策略选取最优个体
    m=m2;
    g(t)=X2(end);
  end
 end
end
```

附录 5：差分演化进化策略算法多峰函数优化的主要源代码

```
X=9*rand(u,wei)+1;
sigma(:,:)=3.0;
f=1./(1+abs(1-v));
  while(m<epsilon)
    k2=floor(u*rand(lenda,2))+1;
    XX(:,:)=X(k2(:,1),:)*0.618034+X(k2(:,2),:)*0.381966;%黄金分割重组
    Sigma(:,:)=sigma(k2(:,1),:)*0.618034+sigma(k2(:,2),:)*0.381966;
    Sigma=Sigma.*exp(r1*ra1+r*ra);     %突变
    XX=XX+Sigma(:,:).*ra(:,:);
      for i=1:lenda             %对个体的区间范围进行限制
        for j=1:wei
          if XX(i,j)>10||XX(i,j)<1
            XX(i,j)=9*rand+1;
          end
        end
      end
  vv=sin(HE)./HE;
  f=1./(1+abs(1-vv));)
  X1=XX(index(lenda-u+1:end),:);        %选出最优的一组解
  sigma=Sigma(index(lenda-u+1:end),:);
```

```
g=X(end,:);
   m=f(end);
   lem=(T-t)/T;          %退火算子
   k1=floor(u*rand(u,3))+1;
   G=g(ones(u,1),:);
   X(:,:)=lem*X1(k1(:,3),:)+(1-lem)*G+1.5*(X1(k1(:,1),:)-X1(k1(:,2),:));
    for i=1:u            %对个体的区间范围进行限制
      for j=1:wei
        if X(i,j)>10||X(i,j)<1
          X(i,j)=9*rand+1;
        end
      end
    end
end
```

附录 6：进化泛函网络算法多维函数逼近的主要源代码

```
Y=[0.880 1.117 0.299 0.873 0.359 0.935 0.681 1.336 1.019 0.556 1.072 1.369 1.449
1.214 0.604 0.669 0.760 1.228 0.989 0.814];
X=[0.384 0.439;0.976 0.440;0.309 0.922;0.725 0.558;0.449 0.907;0.058 0.122;0.673
0.682; 0.697 0.242;0.387 0.320;0.662 0.762;0.136 0.062;0.801 0.263;0.457
0.006;0.377 0.143;0.818 0.727;0.799 0.689;0.663 0.627;0.862 0.364;0.405
0.357;0.028 0.214]; %给定的数据
ji1=ones(20,1);
ji2=X(:,1);
ji3=X(:,1).^2;
ji4=X(:,2);
ji5=X(:,1).*X(:,2);
ji6=X(:,1).^2.*X(:,2);
ji7=X(:,2).^2;
ji8=X(:,1).*X(:,2).^2;
ji9=X(:,1).^2.*X(:,2).^2;
ji34=X(:,1).^3;
ji68=X(:,1).^3.*X(:,2);
ji12=X(:,1).^3.*X(:,2).^2;
```

```
ji13=X(:,1).*X(:,2).^3;
ji14=X(:,1).^2.*X(:,2).^23;
ji15=X(:,1).^3.*X(:,2).^3;
ji=[ji1 ji2 ji3 ji34 ji4 ji5 ji6 ji68 ji7 ji8 ji9 ji12 ji13 ji14 ji15]'; %基函数
Xishu=2*rand(u,n)-1;      %生成初始个体
sigma(:,:)=3.0;
Shuchuzhi=Xishu*ji;
temp=Zhi-Shuchuzhi;
v=sum((temp).^2,2)./20;
f=1./(1+v);
  while(m<epsilon)
  %黄金分割重组
    k1=floor(u*rand(lenda,2))+1;
   XXishu1(:,:)=Xishu(k1(:,1),:)*0.618034+Xishu(k1(:,2),:)*0.381966;
   Sigma1(:,:)=sigma(k1(:,1),:)*0.618034+sigma(k1(:,2),:)*0.381966;
     %单基因柯西突变
       ra1=randn(lenda,1);
       ra11=randn(lenda,1);
       K1=floor(n*rand(lenda,1))+1;
       for i=1:lenda
         ra=-1./tan(pi.*rand(1,1)+eps);
         XXishu1(i,K1(i))=Xishu1(i,K1(i))+Sigma1(i,K1(i))*ra;
       end
     %单基因高斯突变
       ra2=randn(lenda,1);
       ra22=randn(lenda,1);
       K2=floor(n*rand(lenda,1))+1;
       for i=1:lenda
         Sigma2(i,K2(i))=Sigma2(i,K2(i)).*exp(r1*ra22(i)+r*ra2(i));
         XXishu2(i,K2(i))=XXishu2(i,K2(i))+Sigma2(i,K2(i))*ra22(i);
       end
       XXXishu=[XXishu1;XXishu2];
       Sigma=[Sigma1;Sigma];
       ShuChuZhi=XXXishu*ji;
```

```
temp=ZZZhi-ShuChuZhi;
      vv=sum((temp).^2,2)./20;
      f=1./(1+vv);
      [f,index]=sort(f);
      m=f(end) ;
 %选取新个体
Xishu=XXXishu(index(2*lenda-u+1:end),:);
sigma=Sigma(index(2*lenda-u+1:end),:);
end
```

致 谢

研究生三年的学习生活转眼就要过去了，回想这三年的学习生活感触颇多。在此，特别感谢我的导师周永权博士教授，本文是在周老师的悉心指导下完成的，从开展课题到论文发表，都无不凝聚着导师的心血和汗水。回忆这三年，周老师严谨的治学态度、一丝不苟的工作作风、细致敏锐的洞察力、开阔的思路都给我留下了深刻的印象，还有对我的谆谆教诲都将让我受益终身，没有周老师的辛勤培养，就没有我今天取得的成绩。

同时我也要感谢数学与计算机科学学院的何登旭教授、宣士斌教授、刘晓冀教授、王勇博士、黄敬频教授、曹敦虔老师、刘美玲老师、刘桂青老师等学院老师们，感谢他们多年来在学习和生活上给予我的指导和帮助！

感谢陪伴我度过三年愉快硕士生活的 2005 级全体硕士研究生，感谢他们对我的热忱帮助！

在此我要特别感谢我的父母，感谢他们给予我博大无私的关爱与支持，是他们无私的资助，不断的鼓励，才使我完成学业，顺利地走上工作岗位。我为有这样的父母而骄傲，没有他们，就不会有我的今天。感谢我所有的亲人，是他们的大力支持我才能完成自己的学业，再次对家人致以最衷心的感谢！

感谢广西民族大学三年来对我的培养。感谢曾经教育和帮助过我的所有老师。感谢百忙之中抽出时间参加论文评阅和评议的各位专家学者，感谢你们为审阅本文所付出的辛勤劳动！

最后再次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！

攻读硕士期间参与的科研项目

1.国家自然科学基金项目《变参数自适应代数神经网络及应用》项目号: 60461001

2．广西自然科学基金项目《变参数自适应代数神经网络及应用》项目号: 0542048

3．广西民族大学重大科研项目《一类自适应回归神经计算模型及其应用》项目号: 0609013

4．广西研究生教育创新计划资助项目《进化策略在化学方程式配平中的应用》项目号: 2007106080701M18

攻读硕士学位期间发表的学术论文

[1] 夏慧明, 梁华, 周永权. 用双种群进化策略算法求解复函数方程的根. 计算机工程与应用(中文核心期刊) , 2008, 44(7): 78-81.

[2] 夏慧明, 梁华, 周永权. 进化策略算法在矩阵特征值求解中的应用. 计算机工程与设计(中文核心期刊) , 2008, 29(8): 2093-2095, 2098.

[3] 夏慧明, 周永权. 求解矩阵特征值及特征向量的进化策略新方法. 计算机工程(中文核心期刊) , 已录用.

[4] 夏慧明, 梁华, 周永权. 一种基于进化策略的化学方程式配平新方法. 数学的实践与认识(中文核心期刊) , 已录用.

[5] 夏慧明, 周永权. 用于多峰值函数优化的改进差分进化策略算法. 已投稿.

[6] Xia hui-ming, Zhou yong-quan. A Novel Method for Balancing Chemical Equation Based on Evolution Strategy. 已投稿.