硕士学位论文

基于改进粒子群算法求解常微分方程定解问题

SOLVING DEFINITE PROBLEM OF ORDINARY DIFFERENTIAL EQUATION WITH AN IMPROVED PARTICLE SWARM OPTIMIZATION

张晶晶

哈尔滨工业大学

2018 年 12 月

国内图书分类号：O29　　学校代码：10213
国际图书分类号：51　　密级：公开

理学硕士学位论文

基于改进粒子群算法求解常微分方程定解问题

硕士研究生：张晶晶
导　　师：张新明副教授
申请学位：理学硕士
学　　科：应用数学
所在单位：哈尔滨工业大学（深圳）
答辩日期：2018 年 12 月
授予学位单位：哈尔滨工业大学

Classified Index：O29
U.D.C：51

A dissertation submitted in partial fulfillment of
the requirements for the academic degree of
Master of Science

SOLVING DEFINITE PROBLEM OF ORDINARY DIFFERENTIAL EQUATION WITH AN IMPROVED PARTICLE SWARM OPTIMIZATION

Candidate： Zhang Jingjing
Supervisor： Associate Prof. Zhang Xinming
Academic Degree Applied for： Master of Science
Speciality： Applied Mathematics
Affiliation： Harbin Institute of Technology, Shenzhen
Date of Defence： December, 2018
Degree-Conferring-Institution： Harbin Institute of Technology

摘　要

在自然科学、工程技术以及经济管理等领域中的很多数学模型，其表现形式通常为常微分方程的定解问题，如何有效地进行求解是非常关键的。由于理论方法的局限性，很多方程无法求出解析解，所以从实际应用上来讲，人们需要的往往并不是解在数学理论上的存在唯一性或者具体地求出其解析式，而是在我们关心的某个定义范围内求出对应于精确解的近似值。然而，大部分传统的数值方法普遍存在计算复杂，解的精度低等缺点，近些年，随着计算机的发展，智能算法被广泛使用，为求解常微分方程提供了新的方法。

粒子群算法是一种群智能算法，本文通过对粒子群算法的研究与分析，提出了一种改进的粒子群算法，并利用改进的粒子群算法对常微分方程定解问题进行求解。

本文结合“教与学”优化算法，提出一种新的改进粒子群优化算法。利用函数优化问题来验证改进算法的有效性，数值实验结果表明，改进的粒子群优化算法相比于标准的粒子群优化算法具有求解精度高的优势。在此基础上，采用傅里叶级数构造微分方程的近似解，将微分方程转化为约束优化问题，利用粒子群算法与改进的粒子群算法分别计算此优化问题,最终得到方程的近似解。利用上述思路，通过具体实例进行数值实验，对结果做出相应的误差分析，结果证明，改进后的粒子群算法可以搜索到更精确的解，验证了改进后的粒子群算法求解常微分方程定解问题的可行性。因此，本文不仅拓宽了粒子群算法的应用范围，为常微分方程定解问题求近似解提供了新的思路，还提高了常微分方程定解问题的求解精度。

关键词：常微分方程；“教与学”优化算法；粒子群优化算法；构造函数

Abstract

Many mathematical models in the fields of natural sciences, engineering technology and economic management are usually characterized by the solution of ordinary differential equations. It is very important to solve them effectively. Due to the limitations of theoretical methods, many equations have not analytical solutions. Most traditional numerical methods generally have the disadvantages with regard of computational complexity and low accuracy of the solution. Therefore, from the practical point of view, it is not needed to find the analytical formula specifically. People need to find an approximation corresponding to the exact solution. In recent years, with the development of computers, the method of solving ordinary differential equations by intelligent algorithms has become a new direction.

Particle swarm optimization is a kind of swarm intelligence algorithm. Based on the research and analysis of particle swarm optimization, in this paper an improved particle swarm optimization algorithm is proposed. And then the improved particle swarm optimization algorithm is applied to abtain the approximate solution of ordinary differential equations.

In this paper a new improved particles swarm optimization algorithm is proposed by combining the teaching-learning-based optimization algorithm with the particle swarm optimization algorithm. First of all, the function optimization problem is used to verify the effectiveness of the improved algorithm. The results of numerical experiments show that the improved particle swarm optimization algorithm has better global search ability and higher accuracy compared with the standard particle swarm optimization algorithm. Then, we use the Fourier series to construct the approximate solution of the differential equation, and the differential equation can be transformed into the constrained optimization problem. The standard particle swarm optimization algorithm and the improved particle swarm optimization algorithm are used to calculate the optimization problem respectively, and finally the approximate solutions of the equation are obtained. Based on the above ideas, the error analysis of the results is carried out and the feasibility of the

method is illustrated by solving some specific examples. Therefore, in this paper, we not only broaden the application range of particle swarm optimization algorithm, but also provide a new idea for solving the approximate solution of the ordinary differential equation to improve the solution accuary of definite problem of ordinary differential equation.

Keywords: ordinary differential equation, teaching-learning-based optimization algorithm, particle swarm optimization algorithm, constructed function

目　录

第 1 章 绪　论

1.1 课题背景及研究的目的及意义

自从十七世纪以来，伴随人类的需要与社会生产力的进步，人类面临着许多科学与实际问题有待解决，因此，微积分应运而生。随着微积分的进一步发展，其在物理学、几何学、力学等方面有了广泛的应用，促使了一系列新兴学科的形成，微分方程就是其中一门学科。

人类社会的发展与进步有赖于微分方程的研究。微分方程对于社会科学与数学科学的连接起到了重要的作用，是解决实际问题非常强大的数学工具。在很多领域都具有十分重要的作用，天气预报、量子力学和股市动态都是一些具体例子。简而言之，微分方程正逐渐成为数学科学与实际联系的主要方法之一。在自然科学中，许多现象与问题，本质上即是对微分方程模型的探索与研究。在多数情况下，事物通常有自己的运行规律，单纯依靠实验观察往往无法掌握，而微分方程可以在理论上概括其运行的规律。物体的运动过程可以用微分方程的表达式概括，求出它的解，就可以了解到所有的运动过程。因此，微分方程的求解十分重要。但是，只有少数微分方程有精确解或者是解析解。在无法求出解析解时，可以采取数值分析，求得其数值解，如欧拉、龙格-库塔和亚当斯方法，但是数值技术也有自己的局限。伴随计算机技术的逐步发展，人工智能为我们提供了强大的进化算法来解决复杂问题。因此，利用智能算法对微分方程求解具有一定的研究意义。

伴随计算机技术的逐步发展，人工智能算法的出现为数值研究提供了新的空间。如今，智能优化算法正被逐步探索，虽然在理论方面还有很大的提升空间，但是其拥有的先进性和高效性不可被忽视，在科学研究中，智能优化算法是一个有利的工具，同时也将会成为每一个科研人员需要掌握的基础性技能。智能算法一般有遗传算法（Genetic Alogrithm，启发于物种的自然选择与遗传进化的特征）、人工免疫系统（Artificial Immune Systems，启发于免疫系统中处理复杂信息的特点）、蚁群算法（Ant Colony Opimization，启发于蚁群在觅食过程中信息共享的特点），还有天牛须搜索算法、神经网络等。其中，粒子群优化算法（Particle Swarm Optimization, 简称PSO）也是智能算法。

粒子群优化算法作为一种智能算法，通过迭代进行优化。在种群内进行随

机优化，通过系统初始化，得到一组随机的解，在算法中迭代来搜寻到最优解。它属于人工生命和进化计算的方法，但它又与其他进化算法不尽相同，并不是利用群体解之间的相互竞争来进行迭代，从而产生最优解，相反，利用群体解之间的信息共享不断更新来生成最优值。相比于其它算法，粒子群优化算法有很多优点，比如理论基础相对简单、需要调用的参数不多。利用粒子群算法对常微分方程求解，计算方法简单，得到的解相对较好。然而，标准的粒子群算法也存在一些不足之处。因此，改进粒子群算法，并用改进的粒子群算法求解微分方程具有极大的科学价值与应用意义。

1.2 常微分方程定解问题求解的研究现状

1.2.1 传统算法研究现状

在自然科学和工程技术中的很多实际问题，它们的数学模型很多都是常微分方程，而常微分方程通常很难求得其精确解。对于常微分方程初值问题，想要求其数值解，常用的方法主要是单步法和多步法[1]。

欧拉法是单步法中最简单的数值解法，在运用欧拉法的过程中，只进行了一次斜率的计算。虽然降低了计算量，但是精度并不高。1895 年，龙格对欧拉法做出改进，提出了一种可以提高常微分方程求解精度的方法，即单步法。根据这一思想，龙格-库塔法是一种具有较高的求解精度和应用较为广泛的方法。而后，学者又构造出了新的方法即多步法，其中以 Adams 法为代表[2]。

对于常微分方程边值问题，想要求其数值解，普遍的方法主要包括打靶法和差分法[3]。求解边值问题的方法与求解初值问题相似，打靶法是将边值问题转化为初值问题，即将两点边值化为一阶初值方程组，按照初值问题的解法，通过进行迭代求出初始的斜率，最后求出的解就是此边值问题的最终答案。另一种方法是差分法，它将常微分方程关系式中的导数部分替代成差分公式，以此常微分方程便转化为差分方程，从而求解，其本质是将常微分方程边值问题进行转化，变为代数方程进行求解。

虽然许多学者对传统的数值方法进行了探索与测试，但这些数值解法仍然存在一定的局限性，如计算过程复杂、耗费时间等问题。

1.2.2 新型智能算法研究现状

近年来，随着计算机技术的发展，一些全局性的优化算法，像进化算法中

的遗传算法、群智能算法中的蚁群优化算法等走进了大家的视野。这些优化算法是人们通过观察动物的进化过程并分析研究它们在这种自然选择中表现出来的觅食和交流的独特特征而思考出的优化算法。如今，这些算法已经广泛应用于工程技术、经济金融等领域及一些优化方向。

遗传算法是一种根据遗传学模仿进化的自然进化计算的方法。该算法主要利用达尔文进化论，即适者生存原则，操作对象是染色体（二进制串），在进化过程中，通过染色体的杂交、变异来产生下一代，通过搜索控制策略进行一代一代的演化，直到获得最优解[4]。近年来，大量的研究学者们利用遗传算法，基于遗传算法的这些特征，不断解决常微分方程近似解的计算问题。1993 年，Diver 提出用遗传算法求解常微分方程的边值求解问题[5]。2006 年，文献[6]中提出基于遗传算法程序求解常微分方程，试图将相关错误最小化。2011 年，文献[7]提出一种基于遗传算法的两步遗传算法，利用两步遗传算法解决常微分方程组求解问题。2013 年，文献[8]提出了利用粒子群优化算法来解决线性以及非线性常微分方程求解问题。2017 年，文献[9]提出了一种改进的 RNA 遗传算法，并通过这种算法求解常微分方程。

蚁群算法是由意大利的 Marco 提出的一种群智能优化算法，受启发于蚂蚁觅食行为的高效智能的群体策略[10]。研究表明，在蚂蚁觅食期间，当蚂蚁移回它所发现的食物时，它会在它走过的路径上留下一种化学物质，这种化学物质被称为信息素，当其他的蚂蚁嗅到信息素时，同样会沿着信息素寻找食物，直到找到食物的位置为止。在这个过程中，虽然每个蚂蚁只能不高，但通过协作共享，我们发现蚂蚁总能找到最短的路径寻找到食物。基于这样的群体策略，提出了蚁群算法优化问题的基本思路。如今，在各个领域的优化问题中都有蚁群算法的身影。虽然解决常微分方程定解问题研究已经很多，但是由于蚁群算法的理论基础并不牢固，利蚁群算法求解仍处于基础阶段，一些改进模型的融合有待进一步研究。2013 年，Kanae 等人提出一种调整模糊隶属函数参数的蚁群优化算法，并将其应用在常微分方程中[11]。2015 年，Kamali 等人描述了非传统的改进蚁群规划算法，并利用这种方法对常微分方程求解，实验表明该算法求解具备一致性[12]。

人工蜂群算法由 Karaboga 小组提出，其灵感来自蜂群觅食行为，该算法用以解决多变量函数优化问题[13]。在优化过程中，需要求解的问题的解决答案被视同为人工食物源，食物源越复杂多样，意味着解的整体质量很高，所需招募的雇佣蜂蜜越多；相反，食物源越少，解的质量越差，招募的雇佣蜜蜂越少，

达到一定的阈值，该食物源将被放弃，雇佣蜜蜂将转向丰富的食物源，类似于寻优过程中搜素方向将不断由适应值低的解向适应值高的解逼近。2012 年，Noghrehabadi 等人提出混合动力系统的人工蜂群算法，并将提出的算法应用在非线性微分方程组的求解中，结果具有显著的精度效果[14]。2018 年，Fikri 等人应用人工蜂群算法对微分方程的解进行数值逼近，相比传统方法，可以得到更好的解[15]。

在国内，关于求解微分方程的智能算法研究较少。2007 年，张永恒等人将粒子群算法和蚁群算法应用到打靶法的过程中[16]，结果表明，用这种融合算法对微分方程计算求解，计算过程相对稳定，方法有效。2010 年，马翠等人求解二阶线性常微分方程两点边值问题过程中，引入了粒子群算法，并与传统的有限差分法比较，实验表明，结果优于传统的有限差分法[17]。2015 年，杨鑫等人提出一种改进的遗传算法，并借助 MATLAB 中已有的工具箱对常微分方程求解，结果表明该方法可行[18]。

1.3 粒子群优化算法的研究现状

粒子群算法最早被提出时，将其应用于非线性函数的优化中[19–22]，随后逐渐被应用到各个领域中，解决各种形式的优化问题。粒子群算法的优点是设计简单、程序设计方便、需要控制的参数少，但也存在智能群体算法普遍存在的问题：在全局优化和局部优化的平衡中，易于陷入局部最优，运算后期由于粒子缺乏多样性，以至不能保证算法收敛性和收敛速度。因此在实际问题中，对粒子群优化算法往往需要经过改进才能有较理想的结果。查阅国内外的相关文献，发现对粒子群算法的探索研究主要围绕对算法数学理论方面的基础研究、对算法性能的研究与改进、在算法改进后将其应用在不同的领域之中。国内外的研究工作以后两者为主[23–25]。

1995 年，Eberhart 和 Kennedy 提出粒子群优化算法，通过模拟鸟群搜寻食物行为而建立的计算模型[26]2。在粒子群算法被提出后，又对其进行改进，在更新公式中引入一个参数，即惯性因子，使粒子之前的速度影响现在的速度，从而提高粒子搜索能力。1999 年，Clerc 在速度和位置更新方程中加入收缩因子，实现了算法的收敛效果，同时减轻了算法对于速度的限制，有效提高算法的收敛速度[27]。2001 年，Bergh 提出一种新的粒子群算法，即多启动粒子群算法，根据标准粒子群算法进行迭代指定次数后，保留全局极值，同时重新初始

化种群。该算法虽然可以找到全局极值，却由于多次开始初始化，影响了收敛速度[28]。2006 年，在算法后期的过程中，种群容易陷入单一，从而只能得到局部最优，Meissner 提出了基于生物免疫的粒子群算法，加强了收敛速度与精度，改善粒子群算法容易陷入单一的缺点，使其后期具备多样性，增强全局收敛性[29]。2012 年，Yacine 等人提出了一种依据目标函数值不断变化的情况，进而来调整惯性权重参数的方法，因此更好的在全局与局部优化之间平衡[30]。2016 年，Mahesh 等人提出了一种先进的非主流分选多目标粒子群优化算法，数值实验表明，该算法具有不错的性能[31]。

阅读国内关于粒子群算法的文献表明，有关粒子群算法的研究内容可以分为两大类：基础原理研究和应用改进研究。其中基础原理研究主要包括粒子群算法的自身机制及其数学原理研究、收敛性、鲁棒性的数学证明等；应用改进研究大体为三类，一是充分利用粒子群算法的优点进行实际应用，二是改进它的缺点或不足，三是扩展其应用范围。主要研究方法是采用其它算法结合到粒子群算法中，从而改进出一些新的算法，或者利用一些先进原理引入到粒子群算法中，对其进行改进，或将粒子群算法应用到一些实际系统中，比如离散系统、坐标系统、组合规划等，从而扩大粒子群算法的应用范围。

国内对粒子群优化算法的研究始于 2000 年，在第三届国际智能大会中，张春凯等人发表了关于粒子群优化算法的文章[32]，之后众多国内学者开始进入到对粒子群优化算法的研究中。2004 年，谢胜利等人提出了将粒子群优化算法与模拟退火算法进行结合的方法，实验表明，其收敛性能良好[33]。2005 年，刘波等人将混沌算法的思想融合到粒子群优化算法中，实验证明，这个方法提高了粒子群优化算法的搜索性能[34]。由于粒子群算法在非直角坐标描述系统中应用较少，针对此问题，2006 年，孙大刚等人提出了一种基于极坐标的粒子群算法[35]。2010 年，王志刚等人针对粒子群算法在高维函数的优化问题很难寻找到全局最优值的问题，改进了粒子群算法中的基础公式，使得粒子在最优位置做出更新，较大程度增加了算法的寻找最优解的能力[36]。2013 年，徐红洋等人在粒子群算法中融入了混合蛙跳算法，使得粒子在初始化后进行分组，从而有效地避免早熟收敛现象[37]。2015 年，夏学文等人提出了一种新的粒子群优化算法[38]，使其具备局部学习和反向学习的特点，使得在优化高维函数时求解精度变高。

由于数学模型相对简便，需要调整的参数较少，操作方便，编程简单，使得粒子群算法在约束优化、生产车间调度、函数优化以及工程优化等问题中，

均表现出良好的性能，自问世后在很多学科中得到了广泛的应用。比如在电路设计方面、复杂调度的有效工具、流程规划、任务分配、图像分割、多目标优化等方面，均得到了很好的应用。

1.4 本文主要研究内容

在实际的世界中，有许多问题都可归结为常微分方程的定解问题模型，一般情况下，人们无法计算出这些问题的精确解，只能去求它们的近似解。目前普遍采用智能优化算法解决常微分方程数值解的求解问题。本文利用粒子群优化算法及其改进算法求解常微分方程定解问题，并选取几个具体的常微分方程定解问题的数值算例进行实验研究。

本文的主要研究内容如下：

（1）深入地研究粒子群优化算法的基本理论，提出一种新的改进的粒子群优化算法。通过对“教与学”优化算法的研究，将“教与学”优化算法结合到粒子群算法中，提出一种新的算法。

（2）按照提出的算法原理与流程，利用 MATLAB 进行算法实现。选择四个基准函数对所提出的新算法测试，并与标准的粒子群算法进行比较，验证改进算法的可行性。

（3）构造方程的近似解，将常微分方程求解问题转化为函数优化问题，根据粒子群算法的特点，提出利用粒子群优化算法求解常微分方程定解问题的基本思想与过程。

（4）利用粒子群算法与改进粒子群算法分别求解常微分方程定解问题，通过 MATLAB 计算比较两种算法的结果，对真实解与近似解做误差分析。

第 2 章 理论基础

2.1 引言

由于自然物种丰富，拥有各种复杂而有趣的现象，吸引了很多研究人员深入探索解决优化问题。按照昆虫体或其他动物社会行为的机制，科学家们提出了群智能算法。其中，粒子群优化算法就是一种智能算法，该算法为常微分方程的求解问题提供了新的理论空间。

2.2 常微分方程定解问题描述

在自然学科、工程技术等领域中，某些现象的变化过程往往遵循一些基本规律。这些规律可以抽象成为常微分方程这一形式的数学模型，根据实际问题，求出其中满足某种指定条件的解来，求这种解的问题就是定解问题，一般分为初值问题和边值问题。

本文重点求解的方程有以下几类：

（1）一阶常微分方程初值问题形式如下：

$$\begin{cases} y' = f(x,y),\ a \le x \le b \\ y(a) = y_0 \end{cases}$$

（2）二阶常微分方程初值问题形式如下：

$$\begin{cases} y'' = f(x,y,y'),\ a \le x \le b \\ y(a) = y_0, y'(a) = \alpha \end{cases}$$

（3）二阶常微分方程边值问题形式如下：

$$\begin{cases} y'' = f(x,y,y'),\ a \le x \le b \\ y(a) = y_0, y(b) = y_n \end{cases}$$

（4）一阶常微分方程组初值问题形式如下：

$$\begin{cases} y_1' = f_1(x,y_1,y_2),\ a \le x \le b \\ y_2' = f_2(x,y_1,y_2),\ a \le x \le b \\ y_1(a) = \alpha,\ y_2(a) = \beta \end{cases}$$

2.3 最优化理论

最优化理论是数学学科的一个重要分支，是探索和发展自然、工程、经济等领域的数学工具。最优化理论是在研究如何在许多方案中找到最佳的解决方案。优化这一技术，正是在提供解决这些问题的理论基础与解决方案[39]。

最优化问题的一般形式如下：

$$\begin{aligned} &\min\ \sigma = f(\boldsymbol{X}) \\ &\text{s.t.}\ \ \boldsymbol{X} \in S = \{\boldsymbol{X} \mid g_i(\boldsymbol{X}) \le 0, i = 1,2,\cdots,m\} \end{aligned} \tag{2-1}$$

其中，$\sigma = f(\boldsymbol{X})$为目标函数，$g_i(\boldsymbol{X})$为约束函数，$S$为约束域，$\boldsymbol{X}$代表待优化的变量。

由于最优化问题在生活实际中的普遍存在，因此产生了各种解决问题的优化算法，通常可分为局部优化算法和全局优化算法两大类。

（1）局部优化算法

如果存在$\boldsymbol{X}_B^* \in B$，使得对$\forall \boldsymbol{X} \in B$，有

$$f(\boldsymbol{X}_B^*) \le f(\boldsymbol{X}),\ \ \boldsymbol{X} \in B \tag{2-2}$$

成立，其中$B \subset S \subseteq R^n$，$S$是由约束条件限制的寻优空间，则称$\boldsymbol{X}_B^*$为$f(\boldsymbol{X})$在$B$内的局部极小点，$f(\boldsymbol{X}_B^*)$为局部极小值。

大部分优化方法基本上都属于局部优化算法，有一个指定的初始点$\boldsymbol{X}_0 \in S$，从这个点开始，依据某种算法，寻找满足下一个目标函数的并得到改进的解，直到满足某个停止准则。

（2）全局优化算法

如果存在$\boldsymbol{X}^* \in S$，使得对$\forall \boldsymbol{X} \in S$，有

$$f(\boldsymbol{X}^*) \le f(\boldsymbol{X}),\ \ \boldsymbol{X} \in S \tag{2-3}$$

成立，其中$S \subseteq R^n$为由约束条件限制的寻优空间，则称$\boldsymbol{X}^*$为$f(\boldsymbol{X})$在S内的全局极小点，$f(\boldsymbol{X}^*)$为其全局极小值。

目前，大多数发展完善的优化方法都是局部优化算法，求得的结果依赖于选取的初始值。在全局优化问题中，已经发展了许多算法，但与大部分成熟的局部优化问题的方法相比，仍然存在较多差距。为了更好地解决全局优化问题，研究人员正尝试考虑减少对确定性的优化方法的研究，随机性优化算法逐步开始被利用。近十年来，由于模仿自然界生物的行为机制而兴起的智能计算方法，成为了具有有效和普遍适应度的随机全局优化方法。其中，粒子群优化算法是近几年兴起的一种新型智能算法，作为一种仿生的启发式算法，粒子群优化算

法模型简单、操作便捷，有着深刻的智能特征，目前已被广泛应用于各领域。

2.4 粒子群算法的基本理论

2.4.1 粒子群算法的简介

粒子群优化算法这一方法，由 Kennedy 和 Eberhart 在 1995 年提出[26],也称粒子群算法，粒子群算法是基于群体智能的优化算法，其基本想法来源于对鸟类觅食行为的模仿研究，从而建立数学模型，并将其应用于优化问题中。粒子群算法模拟鸟群觅食过程中的迁徙和群聚行为，利用群体中个体间的信息共享和协作，引导整个群体朝着可能解的方向运动，在这个过程中逐渐找到更好的可能解。粒子群算法中，优化问题中的可能解被称为粒子，也就是整个鸟群（搜索空间）中的鸟，所有的粒子都有一个适应度值，这个适应度值由优化函数决定，每个粒子存在一个速度，这个速度决定它们的飞行方位和区间，根据这个速度，粒子们在搜索空间中向最好的粒子逼近。和其它的智能算法相比，粒子群算法模型简单并且容易实现，需要调整的参数相对较少，因此该算法越来越多地被应用于约束优化问题、车间调度、路径规划等其他领域。

2.4.2 基本粒子群算法的原理

与其它群集智能计算方法相似，粒子群算法同样拥有“种群”和“进化”的概念，并依据个体的适应度进行操作。不同之处在于，在粒子群优化算法中，对个体并不使用进化算子，而是将个体视为一个粒子，这个粒子没有质量和体积，在搜索范围内，粒子按照一定的速度飞行。通过速度和位置向量，可以描述粒子飞行状态，根据个体的飞行经验和群体的飞行经验，对粒子的飞行速度进行动态调整。

考虑如下全局优化问题：

$$\begin{aligned} &\min \sigma = f(\boldsymbol{X}) \\ &\text{s.t. } \boldsymbol{X} \in S \subset R^n \end{aligned} \tag{2-4}$$

通过粒子群算法，首先对粒子进行初始化，得到随机粒子，即随机解，然后不断地更新迭代，最终找到最优解。在每一次的迭代中，粒子会跟踪两个极值，从而更新自身位置和速度。粒子跟踪的第一个极值被称为个体极值，即粒子自己搜索到的最优值。另一个极值也叫做全局极值，即在整个种群内，当前搜索到的最优解。此外，对于整个种群中的某一部分，这个范围内存在的极值，

被称为局部极值。

假设在一个 D 维的搜索空间中，有 N 个粒子构成一个种群，其中第 i 个粒子表示一个 D 维的向量：

$$\boldsymbol{X}_i=(x_{i1},x_{i2},\cdots,x_{iD}),\ \ i=1,2,\cdots,N$$

第 i 个粒子的“飞行”速度同样是一个 D 维的向量，记为：

$$\boldsymbol{V}_i=(v_{i1},v_{i2},\cdots,v_{id},\cdots,v_{iD}),\ \ i=1,2,\cdots,N$$

第 i 个粒子迄今为止搜索到的个体极值，记为：

$$\boldsymbol{pBest}=(p_{i1},p_{i2},\cdots,p_{iD}),\ \ i=1,2,\cdots,N$$

整个粒子群迄今为止搜索到的全局极值，记为：

$$\boldsymbol{gBest}=(p_{g1},p_{g2},\cdots,p_{gD})$$

按照追随当前最优粒子的原则，粒子 x_i 将按照（2-5）式和（2-6）式更新自己的速度与位置：

$$v_{id}=v_{id}+c_1r_1(p_{id}-x_{id})+c_2r_2(p_{gd}-x_{id}) \tag{2-5}$$

$$x_{id}=x_{id}+v_{id} \tag{2-6}$$

其中，r_1 和 r_2 是介于[0,1]之间的均匀随机数，c_1 和 c_2 为学习因子，也称加速常数，从更新方程可以看出，c_1 用来调整粒子飞向个体极值方向的步长，c_2 用来调整粒子飞向全局极值方向的步长，v_{id} 是粒子的速度，$v_{id}\in[-v_{\max},v_{\max}]$，$v_{\max}$ 是常数，根据实际情况调整大小，目的是限制粒子的速度。式（2-5）和（2-6）描述了 Eberhart 和 Kennedy 最早提出的粒子群优化模型，该模型被称为基本粒子群优化算法[26]l。

pBest 和 ***gBest*** 在种群每一代上的迭代过程如下公式表示：

$$pBest_i^{d+1}=\begin{cases}pBest_i^{(d)},if & F(x_i^{(d+1)})\geq F(pBest_i^{(d)})\\ x_i^{(d+1)},if & F(x_i^{(d+1)})<F(pBest_i^{(d)})\end{cases} \tag{2-7}$$

$$gBest^{(d+1)}=\min\left\{F(pBest_1^{(d+1)}),F(pBest_2^{(d+1)}),...,F(pBest_D^{(d+1)})\right\} \tag{2-8}$$

其中，$F(\cdot)$ 为适应度函数，即根据适应度函数的值先更新所有粒子的个体最优值 $pBest_i$，然后用最优的个体最优值来更新全局最优值 ***gBest***。

粒子在整个种群内进行搜索，追踪个体极值与全局极值，不断更新自己的速度与位置，直到满足指定的迭代次数或指定的误差标准。在每一个维度中，粒子飞行的速度不能超过最大速度 $v_{\max}$。将 $v_{\max}$ 值设置较大，可以使得粒子种群有较好的全局搜索能力，将 $v_{\max}$ 值设置较小，可以使得粒子种群有较好的局部

搜索能力。

2.4.3 标准粒子群算法的原理

在基本的粒子群算法中，粒子飞行的速度这一概念，就等同于搜索过程中的步长概念，搜索步长的值可以直接影响整个问题的优化性能。当搜索步长过大时，粒子能够快速向目标区域飞行，但是距离最优解更近时，过大的搜索步长很容易使得粒子飞越最优解,转向其他区域继续探索,从而使算法难以收敛；当粒子的搜索步长过小时，尽管能够保证算法局部区域的精细搜索，但却直接导致算法全局搜索能力的降低。由此可知，为达到算法局部搜索和全局搜索之间的有效平衡，必须对粒子的飞行速度采取有效的控制与约束。

为此，Eberhart 和 Shi 等人在基本粒子群算法中加入了惯性权重这一控制参数，目的是有效地控制粒子的飞行速度[26]2 。这个想法来自模拟退火算法，惯性权重类似于温度控制参数。计算模型如下：

$$v_{id} = \omega v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \tag{2-9}$$

$$x_{id} = x_{id} + v_{id} \tag{2-10}$$

其中，ω 为惯性权重。（2-9）式等号的右边部分由三部分组成，第一部分 ωv_{id}，被称为“惯性”部分，代表着粒子具有一定的运动习惯，说明了粒子习惯于保持之前的速度。第二部分是 $c_1 r_1 (p_{id} - x_{id})$，被称为“认知”部分，反映了粒子可以记住自身历史经验，具备一定的历史记忆，说明粒子存在向自身历史经验中的最优位置靠拢的趋向。第三部分是 $c_2 r_2 (p_{gd} - x_{id})$，被称为“社会”部分，体现的是粒子与粒子之间的协同合作与信息共享的群体历史经验，说明粒子存在向群体或邻域历史经验中的最优位置靠拢的趋向。

引入惯性权重的粒子群优化计算模型通常被称为标准粒子群优化算法，本文对粒子群算法的研究与应用以此模型为研究基础。

2.5 本章小结

本章主要介绍了常微分方程定解问题的形式、最优化理论以及粒子群优化算法的基本理论。阐述了基本粒子群算法与标准粒子群算法的主要形式和原理，为接下来对粒子群算法的改进，以及基于改进粒子群算法求解常微分方程定解问题提供了理论支持。

第 3 章 粒子群算法的改进

3.1 引言

近年来，受到很多自然界的生物的觅食过程，交流方法，繁衍等独具特色的生存方式的启发和鼓舞，一些群体智能仿生优化算法被学者们提出并逐步优化。目前，这些优化算法已经解决了很多实际工程中的问题。不仅如此，这类算法也与生活息息相关，为我们的生活提供了更多的便利。粒子群算法就是其中的一种群集智能优化算法，目前在各个领域已经得到了很多应用，学者们也一直在研究和优化此类算法。

3.2 粒子群算法的理论框架

3.2.1 粒子群算法的关键

与其他智能算法相比，粒子群算法的最大优点是无需调整太多参数，但这些参数却能对算法的有效性和收敛性产生直接影响，所以，粒子群算法的关键就在于将参数调试到一个合适的数值。当前，粒子群优化算法的数学理论研究仍处于起步阶段，所以对于参数的取值在很大程度上取决于经验。

粒子群算法的参数包括种群大小 N，粒子范围 $[-x_{\max}, x_{\max}]$，粒子最大速度 $v_{\max}$，惯性权重 ω，学习因子 c_1 和 c_2。以下介绍这些参数的作用和它们的设置经验。

（1）种群规模 N：种群规模即粒子的数量，通常选取 20 到 40 个。对于某些比较复杂的或者情况特殊的问题，可以选取 100 或 200 个粒子甚至更多。选取更多数量的粒子，搜索的范围就会更大，找到全局最优值会更容易。当然，算法运行的时间也越长。

（2）粒子范围 $[-x_{\max}, x_{\max}]$：粒子范围由优化问题的具体情况确定，一般将目标函数中的自变量取值范围设置为粒子的范围。

（3）粒子最大速度 $v_{\max}$：粒子在飞行中可以达到多远的距离，是由粒子最大速度决定的。如果 $v_{\max}$ 太大，粒子可能会飞行过远，从而错过最好的解；如果 $v_{\max}$ 太小，粒子只能在局部优解区间搜索，不能在更大的范围进行足够的搜索，导致陷入局部最优解。通常设定 $v_{\max} = k \cdot x_{\max}$，$0.1 \le k \le 1.0$。

（4）惯性权重ω：ω使粒子具备运动惯性，拥有扩大搜索空间的趋势，有探索新区域的能力。取值范围通常为[0.2,1.2]。最初的实验设定ω为1.0，研究人员发现将惯性权重设置为动态，相比于固定值，可以得到更好的搜寻结果，使算法在全局搜索前期有较高的搜索能力，从而得到最好的粒子。动态惯性权重因子可以根据目标函数值的改变而线性改变，也可以依据算法的搜索过程而动态改变。当前，使用比较广泛的是线性递减权值策略，如（3-1）式所示。

$$\omega = \omega_{\max} - num \cdot (\omega_{\max} - \omega_{\min}) / max\_d \tag{3-1}$$

其中，max_d为最大迭代数，$\omega_{\max}$为最初的惯性值，$\omega_{\min}$为迭代至最大代数时的惯性权重值。经典取值$\omega_{\max} = 0.9$，$\omega_{\min}$=0.4。

（5）学习因子c_1和c_2：c_1和c_2代表每个粒子追踪***pBest***和***gBest***位置时的统计加速项的权重。低的值允许粒子在被拉回之前可以在目标区域外来回搜索，高的值则导致粒子突然超过目标区域。c_1和c_2是固定常数，一般将它们的取值范围设定在[0,2]之内。

3.2.2 粒子群算法的基本步骤

标准的粒子群算法的基本步骤如下：

（1）对粒子种群规模进行初始化，并分别设置粒子位置和速度的临界值，以及迭代最大次数。

（2）计算粒子的适应度值，保存当前种群中各粒子的适应度值和位置，将个体的历史最优位置记为***pBest***，评价所有粒子的***pBest***，将当前种群历史最优位置记为***gBest***，得到粒子初始的个体极值以及全局极值。

（3）根据公式（2-9）和公式（2-10）更新所有粒子自身的速度和位置，从而生成一个新种群。

（4）重新计算粒子的适应度值。

（5）依据当前粒子的适应度值，将其与粒子个体极值和全局极值比较，如果当前适应度值优于个体极值，则将粒子的当前位置保存为***pBest***，若当前适应度值优于全局极值，则当前粒子的位置保存为***gBest***。

（6）判断是否满足终止条件，如果满足，算法终止，输出全局最优值，否则跳转到步骤（3）。

通过上述步骤，我们可以发现粒子群算法的思路相对简单，步骤简便的特点。

3.3 改进的粒子群算法

在标准的粒子群算法中，由于种群中的粒子在位置上缺乏多样性，从而存在粒子易于较早收敛于局部极值的早熟现象以及搜索速度慢的问题。针对此问题，在标准的粒子群算法的基础上，结合了“教与学”算法。

3.3.1 “教与学”优化算法

“教与学”优化算法（Teaching-Learning-Based Optimization，简称 TLBO）模拟教学中的学习方式，以班级作为一个单位，通过教师的“教”来帮助班级中的学生提高水平，同时，学生们通过相互“学习”来加强对知识的学习。相较于其他优化算法，教师和学生就是算法中的个体，教师在所有个体中，适应度值是最好的。每个学生学到的每个科目相当于一个控制变量[40-42]。具体定义如下：

对于优化问题：$z=\min f(\boldsymbol{X})$ 。搜索空间 $S=\{\boldsymbol{X}\,|\,x_i^L\le x_i\le x_i^U, i=1,2,...,d\}$ ，空间中的任意搜索点 $\boldsymbol{X}=(x_1,x_2,...,x_d)$，$d$ 表示空间的维数，x_i^L 和 x_i^U 分别为每一维度的上下界，$f(\boldsymbol{X})$ 为目标函数。设置 $\mathbf{X}^j=(x_1^j,x_2^j,...,x_d^j)(j=1,2,...,NP)$ 为搜索空间中的某个点，NP 为空间搜索点的数量（种群大小）。将其分别对应于 TLBO 算法中：

（1） 班级。在“教与学”优化算法中，搜索空间中所有的点组成一个班级。

（2） 学生。班级中任意一个点 $\boldsymbol{X}^j=(x_1^j,x_2^j,...,x_d^j)$ 称之为一个学生。

（3） 教师。班级中成绩最好的学生 $\boldsymbol{X}_{best}$ 称之为教师，用 $\boldsymbol{X}_{teacher}$ 表示。

一个班级可以表示为如下形式：

$$\left[\begin{array}{c|c} \boldsymbol{X}^1 & f(x^1) \\ \boldsymbol{X}^2 & f(x^2) \\ \vdots & \vdots \\ \boldsymbol{X}^{NP} & f(x^{NP}) \end{array}\right]=\left[\begin{array}{cccc|c} x_1^1 & x_2^1 & \cdots & x_d^1 & f(\boldsymbol{X}^1) \\ x_1^2 & x_2^2 & \cdots & x_d^2 & f(\boldsymbol{X}^2) \\ \vdots & \vdots & & \vdots & \vdots \\ x_1^{NP} & x_2^{NP} & \cdots & x_d^{NP} & f(\boldsymbol{X}^{NP}) \end{array}\right]$$

其中，$\boldsymbol{X}^j(j=1,\cdots,NP)$ 表示班级学员，$\boldsymbol{X}_{teacher}=\arg\min f(\boldsymbol{X}^j)(j=1,\cdots,NP)$。$NP$ 是学生人数，d 是学生学到的科目数。

整个 TLBO 算法分成两个部分：Teaching phase（“教”阶段）和 Learning phase（“学”阶段）。

（1）Teaching phase

算法的第一部分：学生向教师学习的过程。在这个过程中，教师试图将整个班级的平均值提高，将其增加到 $\boldsymbol{X}_{teacher}$，实际情况不可能将整个班级的平均值增加到 $\boldsymbol{X}_{teacher}$，但是，班级的平均值可以从 $\boldsymbol{M}_1$ 提高到 $\boldsymbol{M}_2$，两个平均值之间存在一定的距离，这个差值的计算通过如下公式得到：

$$\boldsymbol{difference} = r_i \times (\boldsymbol{X}_{teacher} - TF \times \boldsymbol{mean}) \tag{3-2}$$

其中，r_i 为在[0,1]之间的随机数，$\boldsymbol{mean} = \frac{1}{NP}\sum_{i=1}^{NP}\boldsymbol{X}^i$ 是所有学生的平均值，TF 是教学因子，决定着平均值的改变，有以下公式决定：

$$TF = round[1 + rand(0,1)] \tag{3-3}$$

在 *difference* 的基础上，就可以更新当前解：

$$\boldsymbol{X}^i_{new} = \boldsymbol{X}^i_{old} + \boldsymbol{difference} \tag{3-4}$$

（2）Learning phase

算法的第二部分：通过学生之间的相互交流来促使自身知识的提升，同时，每个学生可以随机选择其他学生进行相互沟通，从而提升自身的水平。在每次迭代更新期间，假设第 i 次迭代是学生 x^i 和 x^j，并且 $i \neq j$。

$$\boldsymbol{X}^i_{new} = \boldsymbol{X}^i_{old} + r_i(\boldsymbol{X}^i - \boldsymbol{X}^j) \quad if \quad f(\boldsymbol{X}^i) < f(\boldsymbol{X}^j) \tag{3-5}$$

$$\boldsymbol{X}^i_{new} = \boldsymbol{X}^i_{old} + r_i(\boldsymbol{X}^j - \boldsymbol{X}^i) \quad if \quad f(\boldsymbol{X}^j) < f(\boldsymbol{X}^i) \tag{3-6}$$

如果 $\boldsymbol{X}_{new}$ 比 $\boldsymbol{X}_{old}$ 好，则接受新解 $\boldsymbol{X}_{new}$。

整个算法的步骤如下：

（1）对种群进行初始化，设定优化问题的参数变量；

（2）将种群中当前的最优解视为教师，计算学生（即种群）的平均值；

（3）根据公式（3-2）计算当前解与最优的平均值之间的差距；

（4）根据公式（3-4）更新当前解；

（5）根据公式（3-5）和（3-6）更新当前解；

（6）重复步骤（2）至（5），直至满足终止条件。

3.3.2 基于“教与学”优化算法的改进粒子群算法

在标准的粒子群算法中，由于种群中的粒子在位置上缺乏多样性，从而存在粒子易于较早收敛于局部极值的早熟现象以及搜索速度慢的问题。针对此问题，在标准粒子群算法的基础上，结合了“教与学”算法。新提出的算法，使得各个粒子向最好的粒子靠拢聚集，加快了搜索速度。同时，通过粒子间的相互

比较，使其相互取长补短，保证了粒子的多样性，从而对算法在搜索空间中的全局探索能力有了保障。

以下是提出的算法的步骤：

（1)对粒子种群规模进行初始化,并分别设置粒子位置和速度的区间范围，以及最大迭代次数。

（2)计算各个粒子的适应度值,保存当前种群中各粒子的适应度值和位置，将个体的历史最优位置记为 ***pBest*** ，评价所有粒子的 ***pBest*** ，将当前种群历史最优位置设定为 ***gBest*** ，获得粒子初始的个体极值和全局极值。

（3）计算粒子的平均值 $\boldsymbol{mBest}=\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{pBest}$ 。

（4）将各个粒子的位置向 *gBest* 位置移动

$$\boldsymbol{X}_{new}^{i}=\boldsymbol{X}_{old}^{i}+r_i(\boldsymbol{gBest}-\boldsymbol{mBest})$$

其中，r_i为在[0,1]之间的随机数。

（5）根据公式（2-9）和公式（2-10）对粒子的速度和位置进行更新。

（6）根据公式（3-5）和（3-6）比较粒子间的位置。

（7）计算当前粒子适应度值，将其与粒子个体极值和全局极值比较，如果当前适应度值优于个体极值，则将粒子的当前位置保存为 ***pBest*** ，若当前适应度值优于全局极值，则当前粒子的位置保存为 ***gBest*** 。

（8）判断是否满足终止条件，如果满足，算法终止，输出全局最优值，否则跳转到（3）。

3.3.3 实验结果比较

为了验证所提算法的有效性，并进行全面的分析，本文选取了四个标准的测试函数对算法进行检验，如下所示：

Sphere 函数的表达式为：

$$f_1(x)=\sum_{i=1}^{n}x_i^2$$

其中，$-10\le x_i\le 10$，$n=10$，全局最优为：$f(x)=0$，$x_i=0$。MATLAB 仿真三维示意图如图 3-1 所示。

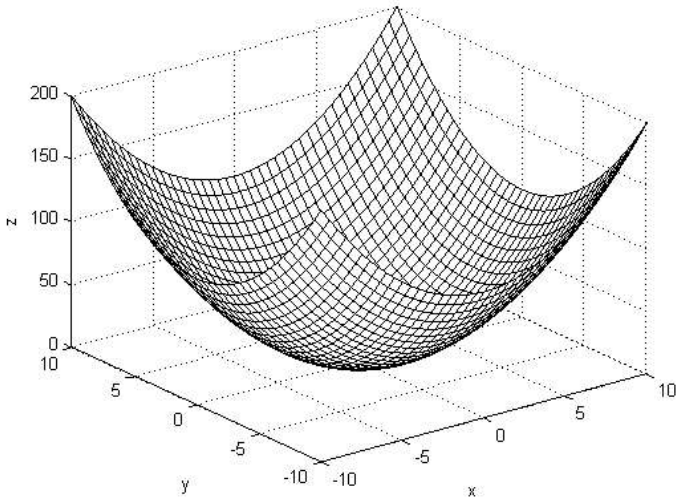

图 3-1 Sphere 函数 MATLAB 三维示意图

Sum Squares 函数的表达式为：

$$f_2(x)=\sum_{i=1}^{n} ix_i^2$$

其中，$-10\le x\le 10$，$n=20$，全局最优为：$f(x)=0$，$x_i=0$。MATLAB 仿真三维示意图如图 3-2 所示。

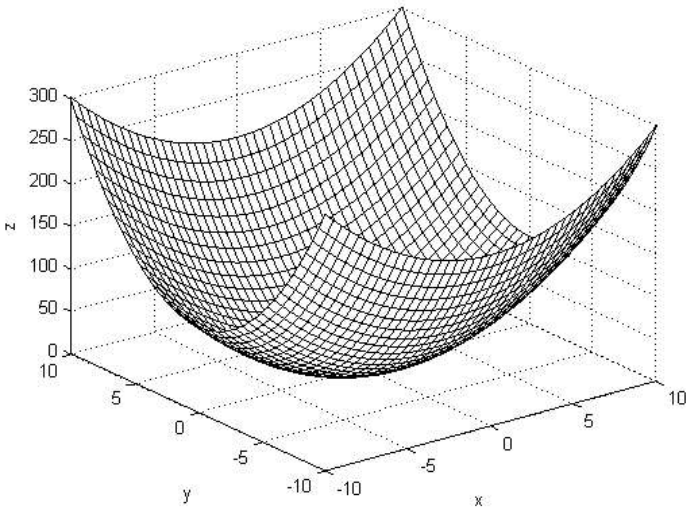

图 3-2 Sum Squares 函数 MATLAB 三维示意图

Matyas 函数的表达式为：

$$f_3(x)=0.26\sum_{i=1}^{n} x_i^2-0.48\sum_{i=2}^{n} x_{i-1}x_i$$

其中，$-10 \le x_i \le 10$，$n = 2$，全局最优为：$f(x) = 0$，$x_i = 0$。MATLAB 仿真三维示意图如图 3-3 所示。

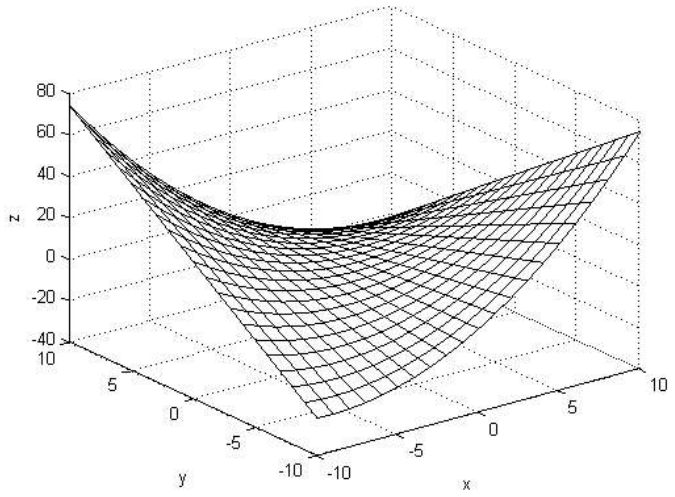

图 3-3 Matyas 函数 MATLAB 三维示意图

Ackley 函数的表达式为：

$$f_4(x) = 20 + \mathrm{e} - 20\mathrm{e}^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - \mathrm{e}^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}$$

其中，$-10 \le x_i \le 10$，$n = 2$，全局最优为：$f(x) = 0$，$x_i = 0$。MATLAB 仿真三维示意图如图 3-4 所示。

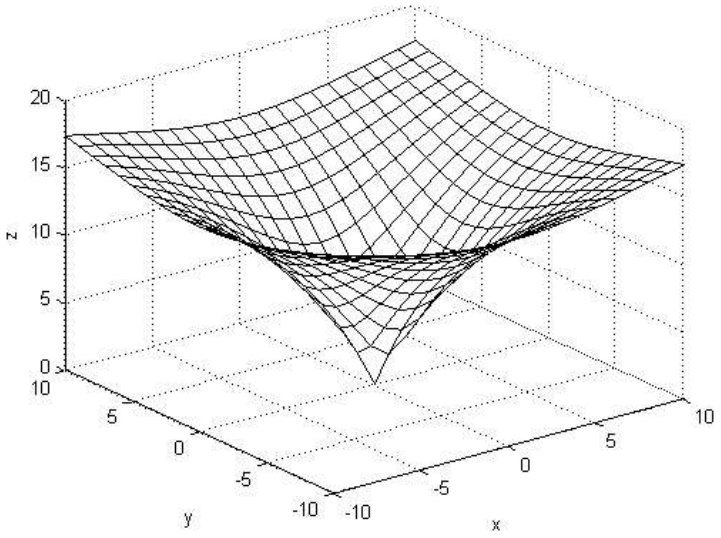

图 3-4 Ackley 函数 MATLAB 三维示意图

本文在算法参数保持一致的条件下进行优化，基本参数设置如表 3-1 所示。

表 3-1 基本参数设置

参数	值/区间
种群大小 N	[10,60]
学习因子 c_1	0.5
学习因子 c_2	1.5
惯性权重 ω	[0.4,0.9]
随机数 r_1, r_2	[0,1]
迭代次数 G	500

为了对算法的性能进行评估，对于上述函数优化问题，对函数在相同参数设置下独立运行 30 次，获取统计数据，得到 30 个最好值、最差值、平均值以及标准差，并与标准的粒子群算法比较，结果如下表 3-2 所示。

表 3-2 两种算法的优化结果

测试函数	算法	最好值	最差值	平均值	标准差
f_1	PSO	6.9306e-04	1.0515e+00	1.2692e-01	2.2234e-01
	TLPSO	6.5272e-07	3.3530e-02	1.4406e-03	6.1892e-03
f_2	PSO	1.2025e-01	7.3757e+00	2.2096e+00	1.7249e+00
	TLPSO	6.4844e-07	2.6027e-04	6.7282e-05	7.8069e-05
f_3	PSO	9.9882e-80	2.0841e-69	6.9786e-71	3.8045e-70
	TLPSO	1.4063e-129	7.2031e-119	3.0269e-120	1.3422e-119
f_4	PSO	-8.8818e-16	2.5799e+00	8.5998e-02	4.7103e-01
	TLPSO	-8.8818e-16	2.6645e-15	-5.3291e-16	1.0840e-15

表 3-2 给出了 PSO 和 TLPSO 对于四个测试函数的计算结果，可以看出，改进的粒子群算法在最好值、最差值、平均值、标准差四个评价指标上都优于标准的粒子群算法。因此，无论是从搜索精度上，还是从算法的稳定性上，改进的粒子群算法的优化效果都比标准的粒子群算法好。

近年来，布谷鸟算法由于其求解精度高，收敛速度快受到广泛应用，下面将布谷鸟算法与改进的粒子群算法对四个基准函数的寻优结果进行比较[43]，结果如表 3-3 所示。

表 3-3 两种算法的优化结果[43]14–15

测试函数	算法	最好值	最差值	平均值
f_1	CS	1.1000e-03	8.1800e-02	2.5600e-02
	TLPSO	6.5272e-07	3.3530e-02	1.4406e-03
f_2	CS	8.8000e-03	1.2440e-01	3.2700e-02
	TLPSO	6.4844e-07	2.6027e-04	6.7282e-05
f_3	CS	2.0919e-32	1.4729e-23	5.1809e-25
	TLPSO	1.4063e-129	7.2031e-119	3.0269e-120
f_4	CS	3.2203e-11	2.9417e-08	3.5574e-09
	TLPSO	-8.8818e-16	2.6645e-15	-5.3291e-16

由表 3-3 可以看出改进的粒子群算法的求解结果均优于布谷鸟算法，通过基准函数优化问题的应用，表明了改进的粒子群算法的有效性，这为我们接下来研究常微分方程定解问题求解情况提供了可能。

为了给出更直观的比较，图 3-5 给出了粒子群算法与改进粒子群算法在 Sphere 函数优化问题中的随机一次运行的收敛图。

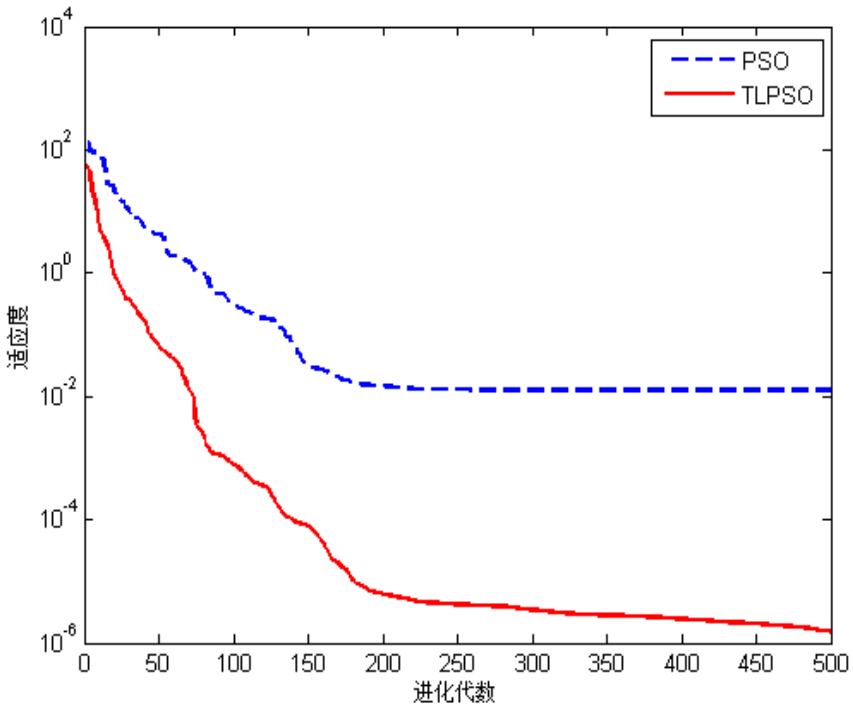

图 3-5 Sphere 函数的收敛曲线

图 3-6 给出了粒子群算法与改进粒子群算法在 Sum Squares 函数优化问题中的随机一次运行的收敛图。

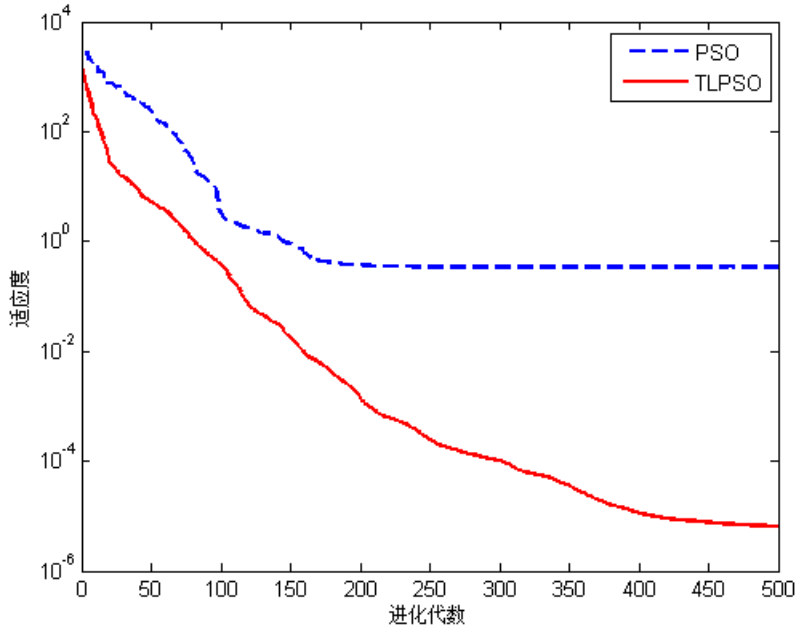

图 3-6　Sum Squares 函数的收敛曲线

图 3-7 给出了粒子群算法与改进粒子群算法在 Matyas 函数优化问题中的随机一次运行的收敛图。

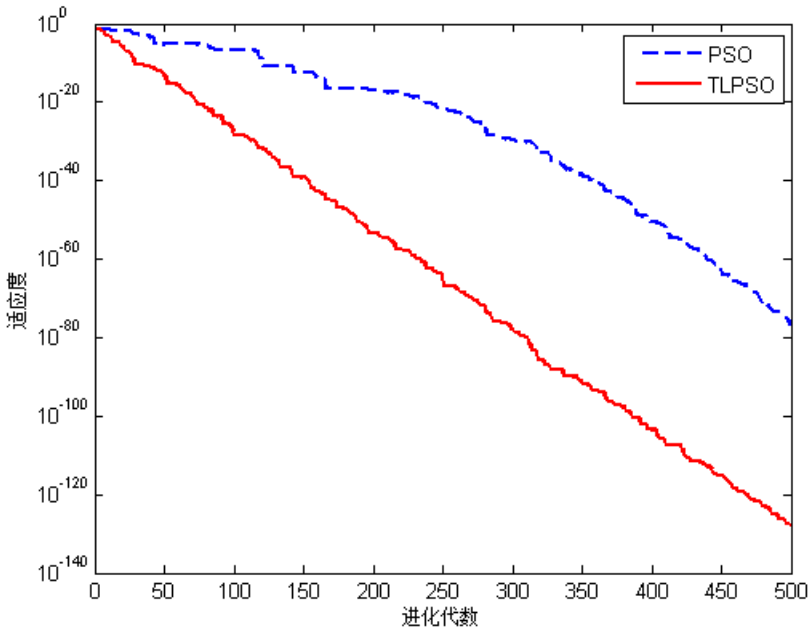

图 3-7　Matyas 函数的收敛曲线

图 3-8 给出了粒子群算法与改进粒子群算法在 Ackley 函数优化问题中的随机一次运行的收敛图。

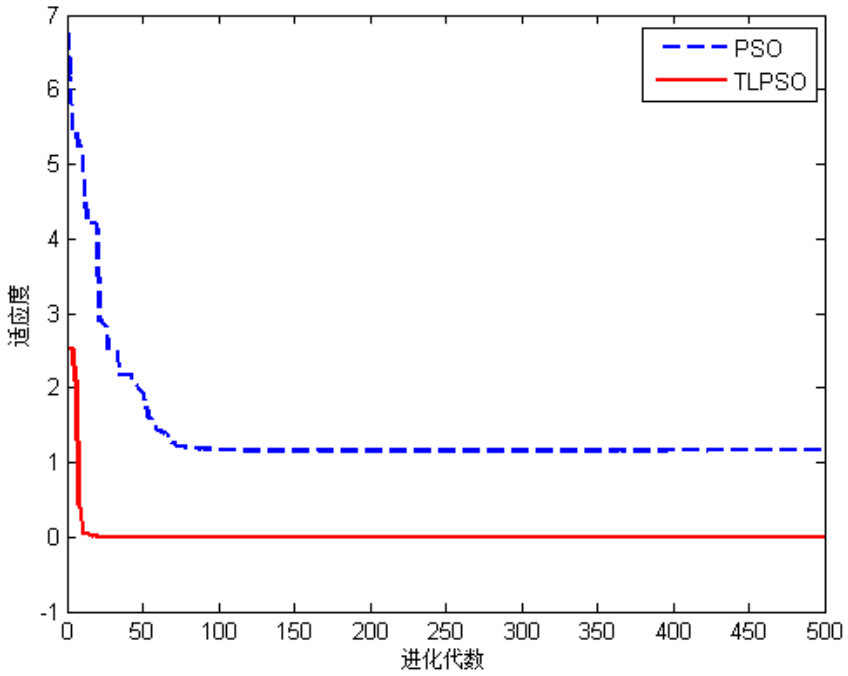

图 3-8　Ackley 函数的收敛曲线

从以上四幅图可以看出，改进的粒子群算法能更快地寻找最优解，加速了优化过程。通过引入“教与学”算法，使得各个粒子向最好的粒子靠拢聚集，加快了搜索速度。同时，保证粒子的多样性特征，从而保证算法不会过早的进入停滞状态，从而更好地搜索最优解。

3.4 本章小结

本章首先介绍了粒子群算法的基本理论框架，然后基于“教与学”优化算法对粒子群算法进行改进，提出了基于“教与学”优化算法的改进粒子群算法。为了验证改进算法的性能，通过四个标准测试函数对该算法的有效性进行检验，实验结果表明该算法搜索精度高，具有优越性。

第 4 章 改进粒子群算法求解常微分方程

4.1 引言

对于大多数的常微分方程，想要求解出其精确的解析表达式是很困难的。随着智能算法的发展，可以利用智能算法求解方程的近似解。

本章通过改进的粒子群算法，对常微分方程定解问题的近似解进行求解，基本思路是，将常微分方程定解问题转化为优化问题，然后再利用改进的粒子群算法求解该约束优化问题。

约束优化问题的一般形式可表达如下：

$$\begin{aligned} \min \quad & f(\boldsymbol{X}) \\ \text{s.t.} \quad & g_i(\boldsymbol{X}) \le 0, \ \ i = 1,2,\ldots,m_1 \\ & h_i(\boldsymbol{X}) = 0, \ \ i = m_1+1,\ldots,m \\ & \boldsymbol{X} \in D \end{aligned} \tag{4-1}$$

其中，$\boldsymbol{X}$是决策变量，$f(\boldsymbol{X})$是目标函数，$g_i(\boldsymbol{X}) \le 0$是不等式约束，$h_i(\boldsymbol{X}) = 0$是等式约束，$D = \left\{ \boldsymbol{X} \in R^n \right\}$，即为可行解空间。

4.2 理论研究

（1）近似解的构造

解区间为x_0到x_n的常微分方程定解问题一般表达形式如下：

$$f\left(x, y, y', \ldots, y^{(n)}\right) = 0 \tag{4-2}$$

边界值条件形式如下：

$$y(x_0) = y_0, \ \ y'(x_0) = y_0', \ldots, y(x_n) = y_n, \ \ y'(x_n) = y_n' \tag{4-3}$$

或初值条件形式如下：

$$y(x_0) = y_0, \ \ y'(x_0) = y_0', \ldots, y^{(n-1)}(x_0) = y_0^{(n-1)} \tag{4-4}$$

对于将求解常微分方程近似解问题转化为约束优化问题，首先要构造方程的近似解。本文采用傅里叶级数的部分和来构造常微分方程的近似解，对于构造近似解，傅里叶级数有以下两个优点。第一，根据傅里叶级数收敛定理，傅里叶级数的收敛性适用于各种连续函数。也就是说，可以将连续函数写为傅里

叶级数的展开式。第二，傅里叶级数包含 sin 和 cos 项，它们可以不断求导，使用有限量的项数来满足高阶微分方程的近似解，不必担心在微分过程中会减少它的导数。将具有待定系数的傅里叶级数部分和作为近似解，通过算法来寻找这些系数。

对于求解区间为 x_0 到 x_n 的常微分方程，我们以 x_0 为中心的傅里叶级数展开式作为近似解，形式如下：

$$y(x) \approx Y_{nat}(x) = a_0 + \sum_{m=1}^{nat}[a_m \cos(\frac{m\pi(x-x_0)}{L}) + b_m \sin(\frac{m\pi(x-x_0)}{L})] \tag{4-5}$$

其导数如下：

$$\begin{aligned} y'(x) &\approx Y'_{nat}(x) = \sum_{m=1}^{nat}[-\frac{m\pi}{L}a_m \sin(\frac{m\pi(x-x_0)}{L}) + \frac{m\pi}{L}b_m \cos(\frac{m\pi(x-x_0)}{L})] \\ y''(x) &\approx Y''_{nat}(x) = \sum_{m=1}^{nat}[-(\frac{m\pi}{L})^2 a_m \cos(\frac{m\pi(x-x_0)}{L}) - (\frac{m\pi}{L})^2 b_m \sin(\frac{m\pi(x-x_0)}{L})] \\ &\vdots \\ y^{(n)}(x) &\approx Y^{(n)}_{nat}(x) = \sum_{m=1}^{nat}[\cdots] \end{aligned} \tag{4-6}$$

其中，*nat* 是 sin 和 cos 的个数，L 是区间长度，也就是 $L=x_n-x_0$。

这样，常微分方程的近似解求解问题就转化为利用粒子群算法求解具有待定系数的傅里叶级数展开式的部分和问题。

对于由几个微分方程组成的微分方程组，也可以通过构造近似解来求解。本文考虑由两个微分方程组成的方程组，解区间为 t_0 到 t_n 的微分方程组的隐形式如下：

$$\begin{cases} f_1(t,x,y,x',y',\ldots,x^{(n)},y^{(n)}) = 0 \\ f_2(t,x,y,x',y',\ldots,x^{(n)},y^{(n)}) = 0 \end{cases} \tag{4-7}$$

边界值条件形式：

$$\begin{cases} x(t_0) = x_0 \\ y(t_0) = y_0 \end{cases}, \begin{cases} x'(t_0) = x'_0 \\ y'(t_0) = y'_0 \end{cases}, \cdots, \begin{cases} x(t_n) = x_n \\ y(t_n) = y_n \end{cases}, \begin{cases} x'(t_n) = x'_n \\ y'(t_n) = y'_n \end{cases}, \cdots \tag{4-8}$$

或初值条件形式：

$$\begin{cases} x(t_0) = x_0 \\ y(t_0) = y_0 \end{cases}, \begin{cases} x'(t_0) = x'_0 \\ y'(t_0) = y'_0 \end{cases}, \begin{cases} x^{(n-1)}(t_0) = x_0^{(n-1)} \\ y^{(n-1)}(t_0) = y_0^{(n-1)} \end{cases} \tag{4-9}$$

根据以上讨论，方程组的近似解的构造是以 t_0 为中心的傅里叶级数部分和：

$$\begin{cases} x(t) \approx X_{nat}(t) = a_0 + \sum_{m=1}^{nat}[a_m \cos(\frac{m\pi(t-t_0)}{L}) + b_m \sin(\frac{m\pi(t-t_0)}{L})] \\ y(t) \approx Y_{nat}(t) = c_0 + \sum_{m=1}^{nat}[c_m \cos(\frac{m\pi(t-t_0)}{L}) + d_m \sin(\frac{m\pi(t-t_0)}{L})] \end{cases} \tag{4-10}$$

（2）加权残差函数作为收敛标准

为了检验得到的近似解的准确度，并对误差进行数值评估，本文采用加权残差函数作为近似解可接受性的数值标准。加权残差函数是对方程的近似解的积分，近似解 Y_{nat} 以残差积分的形式来满足微分方程，形式如下：

$$WRF = \int_D |W(x)| \cdot |R(x)| \mathrm{d}x \tag{4-11}$$

其中，$W(x)$ 是权重函数，$R(x)$ 是残差函数，即用近似解 $Y(x)$ 和它的导数替换微分方程（4-2）中的 $y, y', \ldots, y^{(n)}$ 得到，形式如下：

$$R(x) = f(x, Y(x), Y'(x), \ldots, Y^{(n)}(x)) \tag{4-12}$$

加权残差函数即是优化问题的目标函数，它的值可以通过梯形法或辛普森法在求解过程中得到。WRF 的值应该越接近零越好，否则该算法不收敛。在当前的公式中，权重函数对近似解的求解有很大的影响。在粒子群算法的执行中，$W(x)$ 的绝对值变大，近似解中出现的误差也会被相应的放大。最终解的准确性将通过 WRF 值进行检查，这个值越小，就越准确。

（3）初值或边界值条件作为约束条件

处理微分方程问题时，我们需要找到同时满足方程本身和其初值或边值条件的解。通过优化算法求解微分方程，我们将初值或边界值条件作为约束条件。对于齐次的条件，有如下形式：

$$\begin{aligned} &y(x_0) = 0 \Rightarrow g_1(x_0) = |y(x_0)| \approx |Y(x_0)| \\ &y'(x_0) = 0 \Rightarrow g_2(x_0) = |y'(x_0)| \approx |Y'(x_0)| \\ &\vdots \\ &y^{(n)}(x_0) = 0 \Rightarrow g_n(x_0) = |y^{(n)}(x_0)| \approx |Y^{(n)}(x_0)| \end{aligned} \tag{4-13}$$

对于非齐次条件，可以将它进行标准化，形式如下：

$$
\begin{aligned}
&y(x_0)=y_0 \Rightarrow g_1(x_0)=\left|\frac{y(x_0)}{y_0}-1\right| \approx \left|\frac{Y(x_0)}{y_0}-1\right| \\
&y'(x_0)=y_0' \Rightarrow g_2(x_0)=\left|\frac{y'(x_0)}{y_0'}-1\right| \approx \left|\frac{Y'(x_0)}{y_0'}-1\right| \\
&\vdots \\
&y^{(n)}(x_0)=y_0^{(n)} \Rightarrow g_1(x_0)=\left|\frac{y^{(n)}(x_0)}{y_0^{(n)}}-1\right| \approx \left|\frac{Y^{(n)}(x_0)}{y_0^{(n)}}-1\right|
\end{aligned}
\tag{4-14}
$$

其中，$g_1, g_2, \ldots, g_n$代表了优化问题的约束条件。

（4）罚函数和适应度函数

粒子群算法本身是处理无约束优化问题，在本文中，想要粒子群算法来求解约束优化问题，其关键在于怎样处理约束条件，即解的可行性。如果约束条件处理不当，其优化的结果可能会出现收敛不够好或者结果不存在的情况。因此，我们采用罚函数法，构造某种罚函数，把它加到目标函数中，将求解函有约束优化问题变成求解无约束优化问题。最后，通过将罚函数 PFV 的值加到加权残差函数WRF，以此来获得适应度函数：

$$FFV = WRF + PFV \tag{4-15}$$

罚函数 PFV 的计算方法如下：

$$PFV = WRF \cdot \sum_{m=1}^{n_{BVs}+n_{IVs}} K_m g_m \tag{4-16}$$

其中，g_m是约束条件，n_{BVs}和n_{IVs}分别是边值条件和初值条件的数量。k_m是惩罚因子，它的值太大或太小，都会影响迭代计算的正常进行。许多实验表明，令$k_m=1$常常可以取得满意的效果。

（5）最小二乘解

在粒子群算法中的一个关键参数是粒子的范围$[-x_{\max}, x_{\max}]$，本文中的粒子即傅里叶级数中的待定系数，它的范围选取，本文采取最小二乘法。

首先构造方程的近似解，将其带入原方程以及初值或边值条件中，得到几个等式。其中待定系数作为未知变量，将x所在区间内选取 20 个等分节点，等式将被离散为 20 个等式。于是，我们构造出了一个形式为 $\boldsymbol{Ax=b}$ 的线性方程组，此方程的方程数多于未知数个数，对应的系数矩阵 $\boldsymbol{A}$ 的行数大于列数，此方程组被称为超定方程组。

定理[44] $\boldsymbol{x}^*$是$\boldsymbol{Ax}=\boldsymbol{b}$的最小二乘解的充要条件为：$\boldsymbol{x}^*$是$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{Ax}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}$的解。

根据以上的定理，通过 MATLAB 可以求出近似解中的待定系数，即粒子的初始搜索范围。

4.3 求解问题的算法

根据上一节讨论的概念，我们将求解常微分方程定解问题转化为在可行域D上求解残差函数的最小化问题，边界值或初值作为其约束条件，形式如下：

$$\min \quad WRF=\int_D |W(x)|\cdot|R(x)|\mathrm{d}x \tag{4-17}$$

$$\begin{aligned}\text{s.t.}\quad & g_i^{(1)}=0,\quad i=1,2,\ldots,n_{BVs}\\ & g_j^{(2)}=0,\quad i=1,2,\ldots,n_{IVs}\end{aligned} \tag{4-18}$$

$W(x)$是权重函数，$R(x)$是残差函数，$g_i^{(1)}$和$g_i^{(2)}$分别代表了边界值和初值条件的约束。算法流程图如图 4-1 所示。

算法的具体步骤如下所示：

（1）将常微分方程化为（4-2）的隐形式形式，解区间从x_0到x_n。

（2）将边值条件或初值条件转化为（4-13）到（4-14）的约束条件形式，以便用于罚函数中。

（3）取适当数量的傅里叶级数的展开项，形式如（4-5）。

（4）将优化变量$a_0,a_1,b_1,a_2,\ldots,a_{nat},b_{nat}$赋给近似函数中的每个系数，并将其引入到粒子群算法中。

（5）调用粒子群优化算法，将最小二乘解作为粒子初始搜索位置，在近似函数$Y_{nat}(x)$中找到待定系数。

（6）计算以$\triangle x$为步长的x的各个点的值

$$x_i=x_0+i\cdot\triangle x \tag{4-19}$$

$$y_i\approx Y_{nat}(x_i) \tag{4-20}$$

（7）计算点x_i处的导数的近似值。

（8）构造形如（4-11）的加权残差函数，将近似函数$Y_{nat}(x)$及其导数值带入残差函数进行数值计算，形式如（4-21）。

$$WRF\approx\int_D\left|W(x)\right|\left|f(x,Y_{nat}(x),Y'_{nat}(x),\ldots,Y_{nat}^{(n)}(x))\right|\mathrm{d}x \tag{4-21}$$

其中，用梯形或辛普森积分法对该积分进行数值计算，将加权函数$W(x)$单位化，

取值为 1。

（9）控制边界条件和初始条件的满意度，在（4-16）中计算违反解的惩罚值。

（10）如（4-15），将加权残差值和惩罚值相加，为每一个粒子获得适应度函数值 *FFV* 。

（11）重复（6）至（11），直到在粒子群算法中达到停止标准。

（12）重复（6）至（12），直到达到预期的精确度。

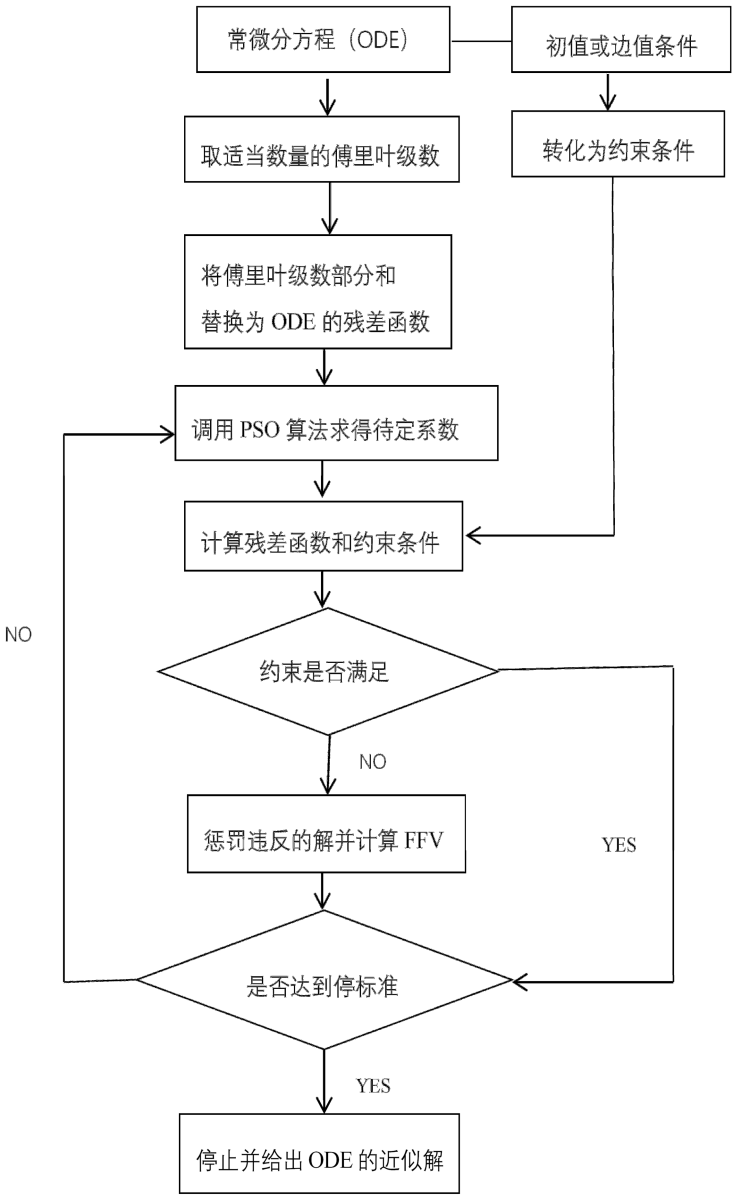

图 4-1 算法流程图

4.4 数值实验与分析

4.4.1 数值实验

本节选取四个具体的常微分方程实例，通过数值实验求得它们的近似解，并做出结果分析。

例 1 $y' = -\frac{1}{5}y + \exp(-\frac{x}{5})\cos x$， $y(0) = 0$， $x \in [0,1]$

该方程的精确解如下：

$$y(x) = \exp(-\frac{x}{5})\sin x \tag{4-22}$$

这是一个一阶线性常微分方程，首先构造如下形式的近似解：

$$y(x) \approx Y_{nat}(x) = a_0 + \sum_{m=1}^{nat}[a_m \cos(\frac{m\pi(x-x_0)}{L}) + b_m \sin(\frac{m\pi(x-x_0)}{L})] \tag{4-23}$$

近似解的一阶导函数如下：

$$y'(x) \approx Y'_{nat}(x) = \sum_{m=1}^{nat}[-\frac{m\pi}{L}a_m \sin(\frac{m\pi(x-x_0)}{L}) + \frac{m\pi}{L}b_m \cos(\frac{m\pi(x-x_0)}{L})] \tag{4-24}$$

其中，$L=1$，$x_0=0$，$nat=5$。

首先将常微分方程转化为约束优化问题，形式如下：

$$\begin{aligned} &\min \quad WRF = \int_0^1 |y' + \frac{1}{5}y - \exp(-\frac{x}{5})\cos x| \mathrm{d}x \\ &\text{s.t.} \quad g_1(0) = a_0 + a_1 + a_2 + a_3 + a_4 + a_5 = 0 \end{aligned} \tag{4-25}$$

根据算法流程，调用粒子群算法，种群大小为 80，进行 1000 次迭代，运行 20 次，得到方程最好的近似解形式如下：

$$\begin{aligned} y(x) \approx\ & (3.2654\mathrm{e}-01) + (-3.8645\mathrm{e}-01)\cos(\pi x) + (1.8742\mathrm{e}-02)\cos(2\pi x) \\ & + (4.3769\mathrm{e}-02)\cos(3\pi x) + (-7.8484\mathrm{e}-04)\cos(4\pi x) \\ & + (-1.8284\mathrm{e}-03)\cos(5\pi x) + (1.2261\mathrm{e}-01)\sin(\pi x) \\ & + (1.2725\mathrm{e}-01)\sin(2\pi x) + (-4.1215\mathrm{e}-03)\sin(3\pi x) \\ & + (-1.1763\mathrm{e}-02)\sin(4\pi x) + (8.8695\mathrm{e}-05)\sin(5\pi x) \end{aligned} \tag{4-26}$$

近似值与真实值的对比图如图 4-2 所示。

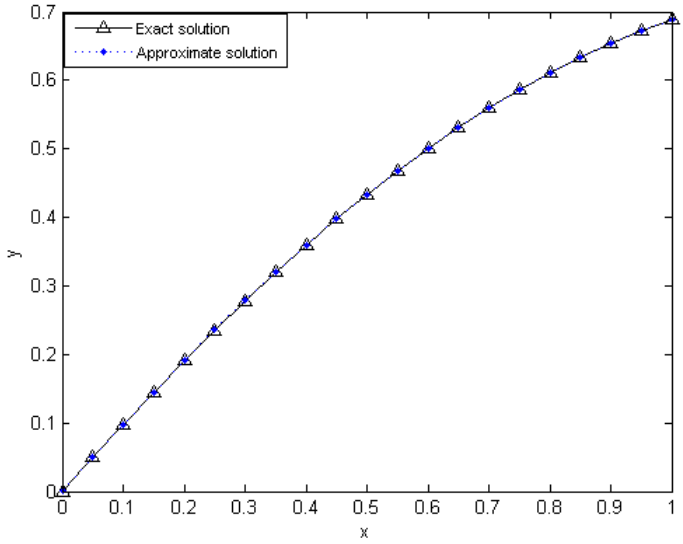

图 4-2 近似解与真实解比较示意图

调用改进的粒子群算法，得到方程最好的近似解如下：

$$\begin{aligned} y(x) \approx\ & (3.2654\mathrm{e}-01)+(-3.8644\mathrm{e}-01)\cos(\pi x)+(1.8759\mathrm{e}-02)\cos(2\pi x) \\ & +(4.3779\mathrm{e}-02)\cos(3\pi x)+(-7.9567\mathrm{e}-04)\cos(4\pi x) \\ & +(-1.8251\mathrm{e}-03)\cos(5\pi x)+(1.2262\mathrm{e}-01)\sin(\pi x) \\ & +(1.2725\mathrm{e}-01)\sin(2\pi x)+(-4.1401\mathrm{e}-03)\sin(3\pi x) \\ & +(-1.1763\mathrm{e}-02)\sin(4\pi x)+(9.2983\mathrm{e}-05)\sin(5\pi x) \end{aligned} \tag{4-27}$$

近似值与真实值的对比图如图 4-3 所示。

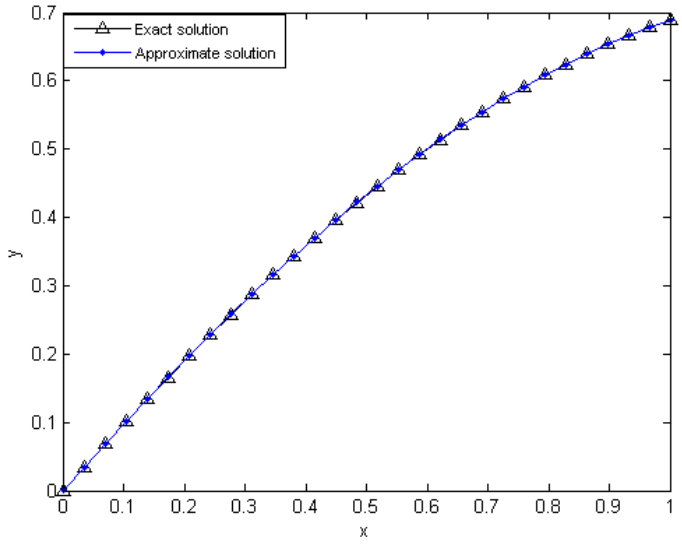

图 4-3 近似解与真实解比较示意图

为了方便比较，求得点 0,0.1,0.2,……,1.0 处的近似值与真实值，并计算出绝对误差，各项结果如表 4-1 所示。

表 4-1 粒子群算法与改进粒子群算法解的绝对误差比较

x值	真实值	粒子群算法的近似值	改进粒子群算法的近似值	粒子群算法的绝对误差	改进粒子群算法的绝对误差
0	0	-1.2240e-05	1.7230e-05	1.2240e-05	1.7230e-05
0.1	9.7857e-02	9.7902e-02	9.7920e-02	4.5768e-05	6.3893e-05
0.2	1.9088e-01	1.9088e-01	1.9089e-01	1.9623e-06	5.8655e-06
0.3	2.7831e-01	2.7837e-01	2.7837e-01	6.3824e-05	6.1754e-05
0.4	3.5948e-01	3.5949e-01	3.5949e-01	1.3102e-05	1.4746e-05
0.5	4.3380e-01	4.3383e-01	4.3384e-01	3.1189e-05	3.6247e-05
0.6	5.0079e-01	5.0084e-01	5.0085e-01	4.8453e-05	5.3496e-05
0.7	5.6006e-01	5.6005e-01	5.6006e-01	9.9047e-07	4.2051e-06
0.8	6.1129e-01	6.1135e-01	6.1135e-01	5.8337e-05	5.8840e-05
0.9	6.5429e-01	6.5430e-01	6.5429e-01	1.3962e-05	1.3105e-06
1.0	6.8894e-01	6.8901e-01	6.8899e-01	6.8387e-05	5.1257e-05

改进粒子群算法解的绝对误差与文献[45]中的遗传算法得到的绝对误差作对比，如表 4-2 所示。

表 4-2 文献[45]与改进粒子群算法解的绝对误差比较

x 值	文献[45]的绝对误差	改进粒子群算法的绝对误差
0.1	1.8831e-04	6.3893e-05
0.2	9.0994e-05	5.8655e-06
0.3	5.1974e-04	6.1754e-05
0.4	8.6643e-04	1.4746e-05
0.5	9.8966e-04	3.6247e-05
0.6	8.4054e-04	5.3496e-05
0.7	4.6405e-04	4.2051e-06
0.8	6.6266e-07	5.8840e-05
0.9	3.2086e-04	1.3105e-06
1.0	1.7182e-04	5.1257e-05

例 2 $\begin{cases} y_1' = y_2 \\ y_2' = -y_1 \end{cases}$，$y_1(0)=1$，$y_2(0)=0$，$x\in[0,1]$

该方程的精确解如下：

$$\begin{cases} y_1 = \cos x \\ y_2 = -\sin x \end{cases} \tag{4-28}$$

这是一个线性方程组，首先构造如下形式的近似解：

$$\begin{cases} y_1(x) \approx Y_{1_{nat}}(x) = a_0 + \sum_{m=1}^{nat}[a_m \cos(\frac{m\pi(x-x_0)}{L}) + b_m \sin(\frac{m\pi(x-x_0)}{L})] \\ y_2(x) \approx Y_{2_{nat}}(x) = c_0 + \sum_{m=1}^{nat}[c_m \cos(\frac{m\pi(x-x_0)}{L}) + d_m \sin(\frac{m\pi(x-x_0)}{L})] \end{cases} \tag{4-29}$$

近似解的一阶导函数如下：

$$\begin{cases} y_1'(x) \approx Y_{1_{nat}}'(x) = \sum_{m=1}^{nat}[-\frac{m\pi}{L}a_m \sin(\frac{m\pi(x-x_0)}{L}) + \frac{m\pi}{L}b_m \cos(\frac{m\pi(x-x_0)}{L})] \\ y_2'(x) \approx Y_{2_{nat}}'(x) = \sum_{m=1}^{nat}[-\frac{m\pi}{L}a_m \sin(\frac{m\pi(x-x_0)}{L}) + \frac{m\pi}{L}b_m \cos(\frac{m\pi(x-x_0)}{L})] \end{cases} \tag{4-30}$$

其中，$L=1$，$x_0=0$，$nat=4$。

首先将常微分方程转化为约束优化问题，形式如下：

$$\begin{aligned} \min \quad & WRF = \int_0^1 |y_1' - y_2| + |y_2' + y_1| \mathrm{d}x \\ \text{s.t.} \quad & g_1(0) = a_0 + a_1 + a_2 + a_3 + a_4 + a_5 - 1 = 0 \\ & g_2(0) = c_0 + c_1 + c_2 + c_3 + c_4 + c_5 = 0 \end{aligned} \tag{4-31}$$

根据算法流程，调用粒子群算法，种群大小为 200，进行 1000 次迭代，运行 20 次，得到方程最好的近似解形式如下：

$$\begin{aligned} y_1 \approx & (-5.1223\mathrm{e}-02) + (2.4763\mathrm{e}-01)\cos(\pi x) + (9.1010\mathrm{e}-01)\cos(2\pi x) \\ & + (-1.7924\mathrm{e}-02)\cos(3\pi x) + (-8.8623\mathrm{e}-02)\cos(4\pi x) \\ & + (1.5357\mathrm{e}+00)\sin(\pi x) + (-6.9338\mathrm{e}-02)\sin(2\pi x) \\ & + (-3.8950\mathrm{e}-01)\sin(3\pi x) + (2.4836\mathrm{e}-03)\sin(4\pi x) \end{aligned} \tag{4-32}$$

$$\begin{aligned} y_2 \approx & (-1.5526\mathrm{e}-02) + (4.5357\mathrm{e}-01)\cos(\pi x) + (-4.4857\mathrm{e}-01)\cos(2\pi x) \\ & + (-3.2638\mathrm{e}-02)\cos(3\pi x) + (4.3661\mathrm{e}-02)\cos(4\pi x) \\ & + (-7.6324\mathrm{e}-01)\sin(\pi x) + (-1.2684\mathrm{e}-01)\sin(2\pi x) \\ & + (1.9171\mathrm{e}-01)\sin(3\pi x) + (4.6292\mathrm{e}-03)\sin(4\pi x) \end{aligned} \tag{4-33}$$

近似解与真实解的对比图如图 4-4 所示。

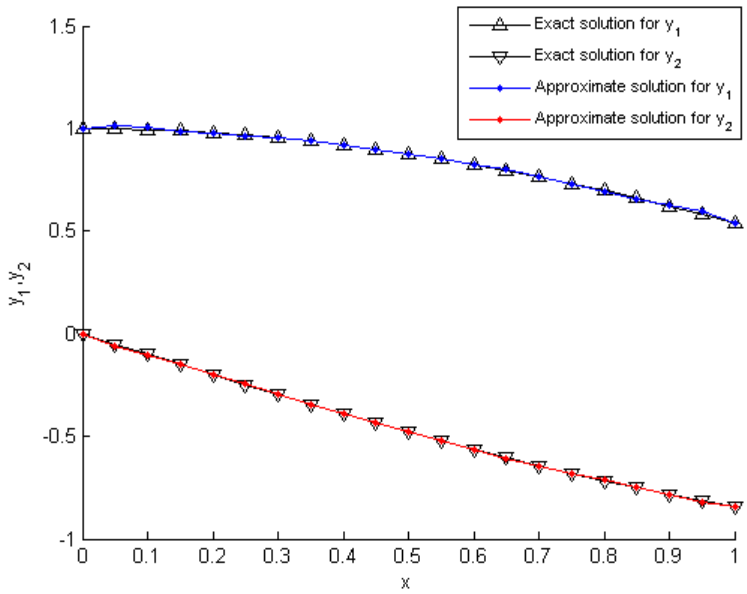

图 4-4 近似解与真实解比较示意图

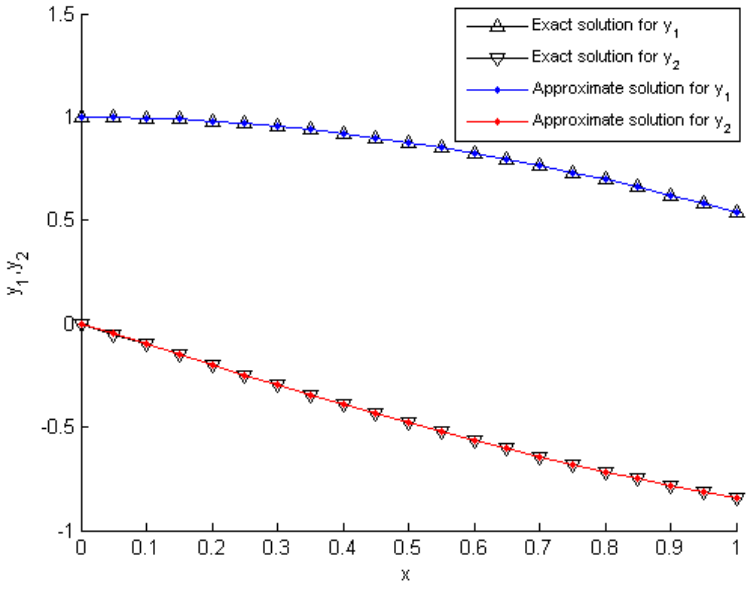

图 4-5 近似解与真实解比较示意图

调用改进的粒子群算法，得到方程最好的近似解如下：

$$\begin{aligned} y_1 \approx\ & (7.5105\mathrm{e}-01)+(2.4846\mathrm{e}-01)\cos(\pi x)+(1.9423\mathrm{e}-02)\cos(2\pi x) \\ & +(-1.8608\mathrm{e}-02)\cos(3\pi x)+(-3.2514\mathrm{e}-04)\cos(4\pi x) \\ & +(1.4309\mathrm{e}-01)\sin(\pi x)+(-7.0231\mathrm{e}-02)\sin(2\pi x) \\ & +(-3.1902\mathrm{e}-03)\sin(3\pi x)+(2.8383\mathrm{e}-03)\sin(4\pi x) \end{aligned} \tag{4-34}$$

$$\begin{aligned} y_2 \approx\ & (-4.1030\mathrm{e}-01)+(4.5480\mathrm{e}-01)\cos(\pi x)+(-1.0611\mathrm{e}-02)\cos(2\pi x) \\ & +(-3.4061\mathrm{e}-02)\cos(3\pi x)+(1.7763\mathrm{e}-04)\cos(4\pi x) \\ & +(-7.8169\mathrm{e}-02)\sin(\pi x)+(-1.2856\mathrm{e}-01)\sin(2\pi x) \\ & +(1.7428\mathrm{e}-03)\sin(3\pi x)+(5.1955\mathrm{e}-03)\sin(4\pi x) \end{aligned} \tag{4-35}$$

近似值与真实值的对比图如图 4-5 所示。

为了方便比较，求得点 0,0.1,0.2,……,1.0 处的近似值与真实值，并计算出绝对误差，y_1 的各项结果如表 4-3 所示。

表 4-3 粒子群算法与改进粒子群算法解的绝对误差比较

x 值	真实值	粒子群算法的近似值	改进粒子群法的近似值	粒子群算法的绝对误差	改进粒子群算法的绝对误差
0	1.0000e+00	9.9996e-01	1.0000e+00	4.0000e-05	1.1249e-04
0.1	9.9500e-01	1.0037e+00	9.9529e-01	8.6993e-03	2.8115e-04
0.2	9.8007e-01	9.7533e-01	9.8025e-01	4.7392e-03	1.8505e-04
0.3	9.5534e-01	9.5648e-01	9.5557e-01	1.1428e-03	2.3562e-04
0.4	9.2106e-01	9.2249e-01	9.2124e-01	1.4283e-03	1.7584e-04
0.5	8.7758e-01	8.7525e-01	8.7777e-01	2.3286e-03	1.8393e-04
0.6	8.2534e-01	8.2668e-01	8.2552e-01	1.3441e-03	1.8817e-04
0.7	7.6484e-01	7.6609e-01	7.6496e-01	1.2454e-03	1.1911e-04
0.8	6.9671e-01	6.9255e-01	6.9687e-01	4.1617e-03	1.6528e-04
0.9	6.2161e-01	6.3054e-01	6.2163e-01	8.9318e-03	1.6480e-05
1.0	5.4030e-01	5.4055e-01	5.4041e-01	2.4569e-04	1.0618e-04

为了方便比较，求得点 0,0.1,0.2, ……,1.0 处的近似值与真实值，并计算出绝对误差，y_2 的各项结果如表 4-4 所示。

表 4-4 粒子群算法与改进粒子群算法解的绝对误差比较

x 值	真实值	粒子群算法的近似值	改进粒子群法的近似值	粒子群算法的绝对误差	改进粒子群法的绝对误差
0	0.0000e+00	4.9700e-04	8.6300e-06	4.9700e-04	8.6300e-06
0.1	-9.9833e-02	-1.0366e-01	-9.9576e-02	3.8243e-03	2.5697e-04
0.2	-1.9867e-01	-1.9664e-01	-1.9864e-01	2.0314e-03	3.1944e-05
0.3	-2.9552e-01	-2.9618e-01	-2.9540e-01	6.5566e-04	1.1919e-04
0.4	-3.8942e-01	-3.9009e-01	-3.8942e-01	6.7538e-04	5.5311e-06
0.5	-4.7943e-01	-4.7824e-01	-4.7943e-01	1.1805e-03	6.3140e-07
0.6	-5.6464e-01	-5.6531e-01	-5.6464e-01	6.6769e-04	5.1749e-06
0.7	-6.4422e-01	-6.4475e-01	-6.4433e-01	5.3689e-04	1.1559e-04
0.8	-7.1736e-01	-7.1488e-01	-7.1738e-01	2.4772e-03	2.8206e-05
0.9	-7.8333e-01	-7.8773e-01	-7.8357e-01	4.3999e-03	2.4215e-04
1.0	-8.4147e-01	-8.4137e-01	-8.4147e-01	1.0398e-04	1.6148e-06

改进粒子群算法解的绝对误差与改进欧拉法得到的绝对误差作对比[46]28，

如表 4-5 所示。

表 4-5 改进粒子群算法解与改进欧拉法的绝对误差比较[46]28

x 值	改进欧拉法的 y_1 的绝对误差	改进粒子群算法的 y_1 的绝对误差	改进欧拉法的 y_2 的绝对误差	改进粒子群算法的 y_2 的绝对误差
0	0.0000e+00	1.1249e-04	0.0000e+00	8.6300e-06
0.1	4.0000e-06	2.8115e-04	1.6700e-04	2.5697e-04
0.2	4.2000e-05	1.8505e-04	3.3100e-04	3.1944e-05
0.3	1.1100e-04	2.3562e-04	4.8800e-04	1.1919e-04
0.4	2.1300e-04	1.7584e-04	6.3200e-04	5.5311e-06
0.5	1.4090e-04	1.8393e-04	7.5900e-04	6.3140e-07
0.6	5.0200e-04	1.8817e-04	8.6500e-04	5.1749e-06
0.7	6.8300e-04	1.1911e-04	9.4500e-04	1.1559e-04
0.8	1.1158e-03	1.6528e-04	9.9700e-04	2.8206e-05
0.9	1.1020e-03	1.6480e-05	1.0170e-04	2.4215e-04
1.0	1.3310e-03	1.0618e-04	1.0020e-03	1.6148e-06

例 3 $y''-y=x$， $y(0)=0$， $y(1)=1$， $x\in[0,1]$

该方程的精确解如下：

$$y(x)=\frac{2(\exp(x)-\exp(-x))}{\mathrm{e}-\mathrm{e}^{-1}}-x \tag{4-36}$$

这是一个二阶线性常微分方程，首先构造如下形式的近似解：

$$y(x)\approx Y_{nat}(x)=a_0+\sum_{m=1}^{nat}[a_m\cos(\frac{m\pi(x-x_0)}{L})+b_m\sin(\frac{m\pi(x-x_0)}{L})] \tag{4-37}$$

近似解的一阶和二阶导函数如下：

$$\begin{aligned} y'(x)&\approx Y'_{nat}(x)=\sum_{m=1}^{nat}[-\frac{m\pi}{L}a_m\sin(\frac{m\pi(x-x_0)}{L})+\frac{m\pi}{L}b_m\cos(\frac{m\pi(x-x_0)}{L})] \\ y''(x)&\approx Y''_{nat}(x)=\sum_{m=1}^{nat}[-(\frac{m\pi}{L})^2a_m\cos(\frac{m\pi(x-x_0)}{L})-(\frac{m\pi}{L})^2b_m\sin(\frac{m\pi(x-x_0)}{L})] \end{aligned} \tag{4-38}$$

其中， $L=1$， $x_0=0$， $nat=6$。

首先将常微分方程转化为约束优化问题，形式如下：

$$\begin{aligned} \min\quad & WRF=\int_0^1|y''-y-x|\mathrm{d}x \\ \text{s.t.}\quad & g_1(0)=a_0+a_1+a_2+a_3+a_4+a_5+a_6=0 \\ & g_2(0)=a_0-a_1-a_2-a_3-a_4-a_5-a_6-1=0 \end{aligned} \tag{4-39}$$

根据算法流程，调用粒子群算法，种群大小为 100，进行 1000 次迭代，运

行 20 次，得到方程最好的近似解形式如下：

$$\begin{aligned} y(x) \approx\ & (5.0442\mathrm{e}-01)+(\text{-}5.7286\mathrm{e}-01)\cos(\pi x)+(-1.1999\mathrm{e}-02)\cos(2\pi x) \\ & +(9.6440\mathrm{e}-02)\cos(3\pi x)+(-4.1455\mathrm{e}-03)\cos(4\pi x) \\ & +(-7.6743\mathrm{e}-03)\cos(5\pi x)+(4.3861\mathrm{e}-04)\cos(6\pi x) \\ & +(-1.4099\mathrm{e}-01)\sin(\pi x)+(2.4153\mathrm{e}-01)\sin(2\pi x) \\ & +(-4.2789\mathrm{e}-03)\sin(3\pi x)+(-3.2219\mathrm{e}-02)\sin(4\pi x) \\ & +(2.6646\mathrm{e}-04)\sin(5\pi x)+(1.0558\mathrm{e}-04)\sin(6\pi x) \end{aligned} \tag{4-40}$$

近似值与真实值的对比图如图 4-6 所示。

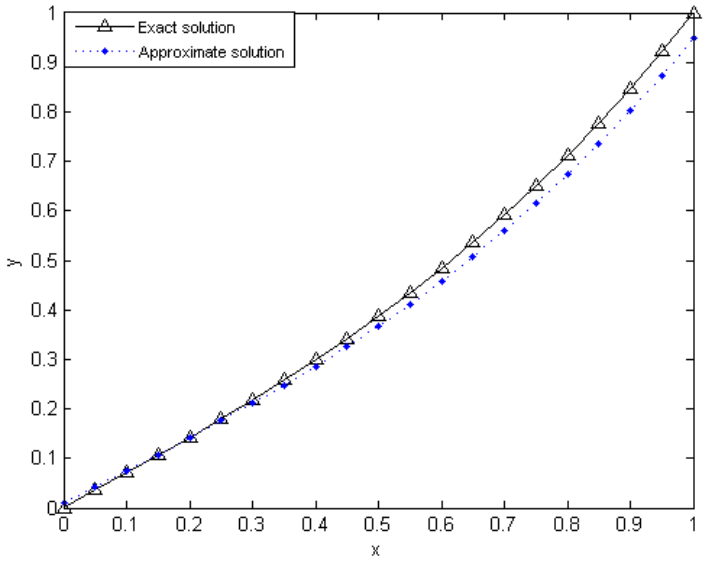

图 4-6 近似解与真实解比较示意图

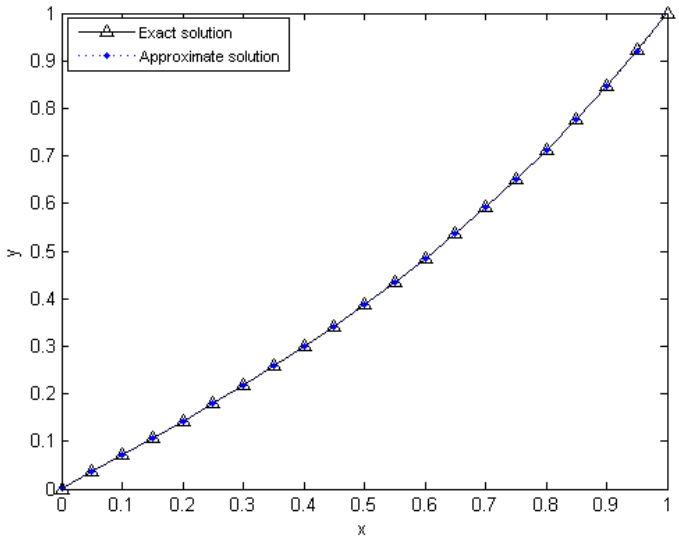

图 4-7 近似解与真实解比较示意图

调用改进的粒子群算法，得到方程最好的近似解如下：

$$\begin{aligned} y(x) \approx\ & (5.0000\mathrm{e}-01)+(-5.9024\mathrm{e}-01)\cos(\pi x)+(9.7283\mathrm{e}-03)\cos(2\pi x) \\ & +(9.7635\mathrm{e}-02)\cos(3\pi x)+(-1.0315\mathrm{e}-02)\cos(4\pi x) \\ & +(-7.4715\mathrm{e}-03)\cos(5\pi x)+(6.7757\mathrm{e}-04)\cos(6\pi x) \\ & +(-1.1384\mathrm{e}-01)\sin(\pi x)+(2.4729\mathrm{e}-01)\sin(2\pi x) \\ & +(-1.7685\mathrm{e}-02)\sin(3\pi x)+(-3.1977\mathrm{e}-02)\sin(4\pi x) \\ & +(3.7192\mathrm{e}-03)\sin(5\pi x)+(9.2809\mathrm{e}-04)\sin(6\pi x) \end{aligned} \tag{4-41}$$

近似值与真实值的对比图如图 4-7 所示。

为了方便比较，求得点 0,0.1,0.2,……,1.0 处的近似值与真实值，并计算出绝对误差，各项结果如表 4-6 所示。

表 4-6 粒子群算法与改进粒子群算法解的绝对误差比较

x 值	真实值	粒子群算法的近似值	改进粒子群法的近似值	粒子群算法的绝对误差	改进粒子群算法的绝对误差
0	0	4.6180e-03	1.4370e-05	4.6180e-03	1.4370e-05
0.1	7.0467e-02	7.0727e-02	7.0568e-02	2.5922e-04	1.0026e-04
0.2	1.4262e-01	1.4134e-01	1.4270e-01	1.3003e-03	6.1312e-05
0.3	2.1824e-01	2.1577e-01	2.1825e-01	2.4732e-03	6.7165e-06
0.4	2.9903e-01	2.9230e-01	2.9907e-01	6.7316e-03	3.8969e-05
0.5	3.8682e-01	3.7539e-01	3.8684e-01	1.1427e-02	2.4446e-05
0.6	4.8348e-01	4.7051e-01	4.8348e-01	1.2968e-02	4.1195e-06
0.7	5.9099e-01	5.7660e-01	5.9096e-01	1.4390e-02	2.7029e-05
0.8	7.1141e-01	6.9220e-01	7.1144e-01	1.9209e-02	2.7113e-05
0.9	8.4696e-01	8.2234e-01	8.4685e-01	2.4624e-02	1.1763e-04
1.0	1.0000e+00	9.7281e-01	1.0002e+00	2.7193e-02	1.6737e-04

改进粒子群算法解的绝对误差与文献[47]中的遗传算法得到的绝对误差做对比，如表 4-7 所示。

表 4-7 文献[47]与改进粒子群算法解的绝对误差比较

x 值	文献[47]方法的绝对误差	改进粒子群算法的绝对误差
0.1	2.1220e-05	1.0026e-04
0.2	4.1320e-05	6.1312e-05
0.3	5.9510e-05	6.7165e-06
0.4	7.4030e-05	3.8969e-05
0.5	8.3610e-05	2.4446e-05
0.6	8.6750e-05	4.1195e-06
0.7	8.1370e-05	2.7029e-05
0.8	6.6750e-05	2.7113e-05
0.9	4.0350e-05	1.1763e-04
1.0	0.0000e+00	1.6737e-04

例 4 $y''-(x-\frac{1}{2})^2 y=-(x^2-x+\frac{5}{4})\sin(x)$， $y(0)=0$， $y'(0)=1$， $x\in[0,\frac{\pi}{2}]$

该方程的精确解如下：

$$y(x)=\sin(x) \tag{4-42}$$

这是一个二阶线性常微分方程，首先构造如下形式的近似解

$$y(x)\approx Y_{nat}(x)=a_0+\sum_{m=1}^{nat}[a_m\cos(\frac{m\pi(x-x_0)}{L})+b_m\sin(\frac{m\pi(x-x_0)}{L})] \tag{4-43}$$

近似解的一阶和二阶导函数如下：

$$\begin{aligned} y'(x)&\approx Y'_{nat}(x)=\sum_{m=1}^{nat}[-\frac{m\pi}{L}a_m\sin(\frac{m\pi(x-x_0)}{L})+\frac{m\pi}{L}b_m\cos(\frac{m\pi(x-x_0)}{L})] \\ y''(x)&\approx Y''_{nat}(x)=\sum_{m=1}^{nat}[-(\frac{m\pi}{L})^2 a_m\cos(\frac{m\pi(x-x_0)}{L})-(\frac{m\pi}{L})^2 b_m\sin(\frac{m\pi(x-x_0)}{L})] \end{aligned} \tag{4-44}$$

其中，$L=\frac{\pi}{2}$， $x_0=0$， $nat=6$。

首先将常微分方程转化为约束优化问题，形式如下：

$$\begin{aligned} \min\quad & WRF=\int_0^{\frac{\pi}{2}}|y''-(x-\frac{1}{2})^2 y+(x^2-x+\frac{5}{4})\sin x|\mathrm{d}x \\ \text{s.t.}\quad & g_1(0)=a_0+a_1+a_2+a_3+a_4+a_5+a_6=0 \\ & g_2(0)=2b_1+4b_2+6b_3+8b_4+10b_5+12b_6-1=0 \end{aligned} \tag{4-45}$$

根据算法流程，调用粒子群算法，种群大小为 100，进行 1000 次迭代，运行 20 次，得到方程最好的近似解形式如下：

$$\begin{aligned} y(x)\approx\ & (4.9409\mathrm{e}-01)+(-5.6911\mathrm{e}-01)\cos(2x)+(9.8970\mathrm{e}-03)\cos(4x) \\ & +(6.5246\mathrm{e}-02)\cos(6x)+(6.1262\mathrm{e}-03)\cos(8x) \\ & +(-5.4883\mathrm{e}-03)\cos(10x)+(-6.1933\mathrm{e}-04)\cos(12x) \\ & +(2.3599\mathrm{e}-01)\sin(2x)+(1.7015\mathrm{e}-01)\sin(4x) \\ & +(8.3942\mathrm{e}-03)\sin(6x)+(-2.2101\mathrm{e}-02)\sin(8x) \\ & +(-2.3785\mathrm{e}-03)\sin(10x)+(7.1181\mathrm{e}-04)\sin(12x) \end{aligned} \tag{4-46}$$

近似值与真实值的对比图如图 4-8 所示。

调用改进的粒子群算法，得到方程最好的近似解如下：

$$\begin{aligned} y(x) \approx\ & (5.0001\mathrm{e}-01)+(-5.6406\mathrm{e}-01)\cos(2x)+(-1.2096\mathrm{e}-02)\cos(4x) \\ & +(7.3240\mathrm{e}-02)\cos(6x)+(1.2425\mathrm{e}-02)\cos(8x) \\ & +(-8.8223\mathrm{e}-03)\cos(10x)+(-6.8729\mathrm{e}-04)\cos(12x) \\ & +(2.0809\mathrm{e}-01)\sin(2x)+(1.7562\mathrm{e}-01)\sin(4x) \\ & +(2.2012\mathrm{e}-02)\sin(6x)+(-2.8441\mathrm{e}-02)\sin(8x) \\ & +(-4.1793\mathrm{e}-03)\sin(10x)+(1.5520\mathrm{e}-03)\sin(12x) \end{aligned} \tag{4-47}$$

近似值与真实值的对比图如图 4-9 所示。

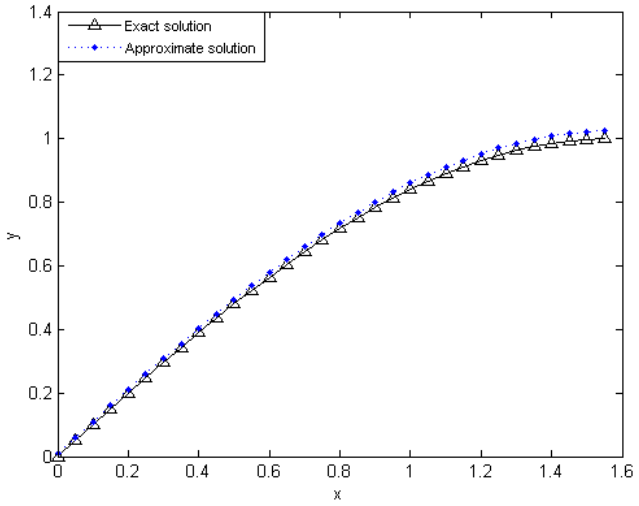

图 4-8 近似解与真实解比较示意图

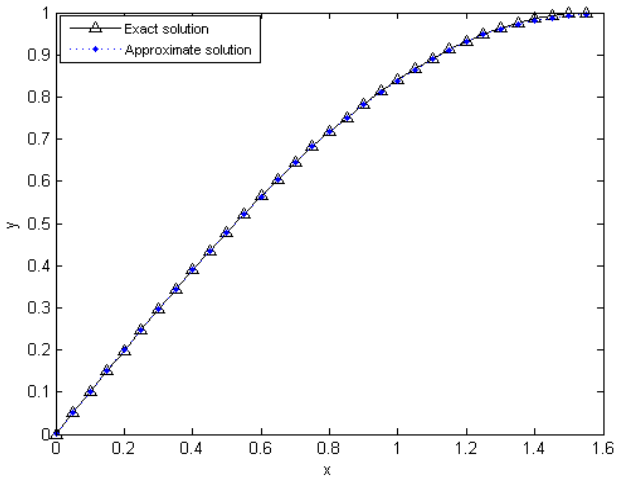

图 4-9 近似解与真实解比较示意图

为了方便比较，求得点 $0,\frac{\pi}{20},\frac{2\pi}{20},\ldots\ldots,\frac{\pi}{2}$ 处的近似值与真实值，并计算出绝对误差，各项结果如表 4-8 所示。

表 4-8 粒子群算法与改进粒子群算法解的绝对误差比较

x 值	真实值	粒子群算法的近似值	改进粒子群法的近似值	粒子群算法的绝对误差	改进粒子群法的绝对误差
0	0	1.4157e-04	9.4100e-06	1.4157e-04	9.4100e-06
$\pi/20$	1.5643e-01	1.5828e-01	1.5646e-01	1.8485e-03	2.4055e-05
$2\pi/20$	3.0902e-01	3.1271e-01	3.0927e-01	3.6907e-03	2.5689e-04
$3\pi/20$	4.5399e-01	4.5929e-01	4.5410e-01	5.3037e-03	1.0701e-04
$4\pi/20$	5.8779e-01	5.9486e-01	5.8776e-01	7.0753e-03	2.2156e-05
$5\pi/20$	7.0711e-01	7.1616e-01	7.0712e-01	9.0490e-03	1.0209e-05
$6\pi/20$	8.0902e-01	8.1972e-01	8.0902e-01	1.0704e-02	5.0292e-09
$7\pi/20$	8.9101e-01	9.0364e-01	8.9084e-01	1.2633e-02	1.6497e-04
$8\pi/20$	9.5106e-01	9.6607e-01	9.5077e-01	1.5012e-02	2.8438e-04
$9\pi/20$	9.8769e-01	1.0048e+00	9.8796e-01	1.7066e-02	2.6950e-04
$\pi/2$	1.0000e+00	1.0188e+00	9.9929e-01	1.8846e-02	7.0599e-04

改进粒子群算法解的绝对误差与差分法得到的绝对误差作对比[46]34，如表 4-9 所示。

表 4-9 改进粒子群算法与差分法解的绝对误差比较[46]34

x 值	差分法的绝对误差	改进粒子群算法的绝对误差
$\pi/8$	4.1273e-04	2.3062e-04
$\pi/4$	6.3923e-04	1.0209e-05
$3\pi/8$	5.2906e-04	2.9979e-04
$\pi/2$	0.0000e+00	7.0599e-04

4.4.2 结果分析

通过以上四个算例，可以看出，相比于标准的粒子群算法以及其他方法求解常微分方程定解问题，基于改进粒子群算法求解常微分方程定解问题，能够有更高的求解精度，近似解与精确解之间的绝对误差更小，有效说明了改进粒子群算法求解的可行性。

4.5 本章小结

本章主要是利用改进的粒子群算法求解具体的常微分方程定解问题，为常微分方程定解问题的求解提供新的思路。首先将微分方程的求解问题转化为约束优化问题，进而将粒子群算法应用于求解该优化问题。为了验证算法的有效性与准确性，将精确解与近似解加以比较，进行了误差分析。可以看出，精确解与近似解的绝对误差很小，相比于标准的粒子群算法求出的解，更加精确，说明了改进的粒子群算法的可行性。

结　论

本文主要解决常微分方程定解问题的求解问题。多年来，求解常微分方程数值解的问题一直是人们关注的焦点。传统的数值方法普遍存在耗费时间，计算复杂等不足。智能算法中，遗传算法及蚁群算法等智能算法存在搜索时间长，精度不高且编程实现比较复杂等局限性。粒子群优化算法作为近几年发展起来的一种新的智能算法，具备参数少，计算速度快和更好的全局搜索能力。在对粒子群优化算法和现有的常微分方程求解方法的研究基础上，本文提出了一种求解常微分方程定解问题的改进粒子群算法，并验证了改进的粒子群算法在求解常微分方程定解问题上的效率。

本文通过 MATLAB 编程进行计算与实验，对标准的粒子群算法进行了深入的研究和探讨，并进行了改进与应用，主要工作总结如下：

（1）介绍了常微分方程定解的一般形式、优化算法的本质以及粒子群算法的理论框架，阐述了标准粒子群算法的原理。

（2）针对粒子群算法容易早熟收敛陷入局部最优极值的问题，结合了“教与学”优化算法，提出了一种改进的粒子群算法。选取四个基准函数进行测试，改进粒子群算法的结果精度明显优于标准粒子群算法。

（3）阐述了粒子群算法应用于常微分方程定解问题的基本思路，通过采用傅里叶级数构造方程近似解的方法，实现了将求解常微分方程定解问题转化为求解无约束优化问题，从而将粒子群算法应用在优化问题的求解中，最终完成了对常微分方程定解问题的求解。选取四个具体的常微分方程为例，进行数值实验，在求解精度上，改进的粒子群算法优于标准的粒子群算法。

综上所述，利用改进的粒子群算法求解常微分方程定解问题的方法可行且相对可靠。未来，研究者们可就提高粒子群算法求解高阶常微分方程所得到的解的精度进行研究，力求在高阶问题中可以得到高精度的解。相信随着对粒子群算法工作原理的深入研究，粒子群算法将拥有更广阔的应用领域。

参考文献

[1] 李忠杰. 常微分方程初值问题 RK 法和多步法[J]. 科学技术创新，2010，197(18)：199-200.

[2] Rodriguez-Fernandez M, Iii F J D. Ordinary Differential Equation (ODE)[J]. Proenvironment Promediu, 2013, 187(1): 72-77.

[3] Meeter D. Introduction to Numerical Analysis[J]. Texts in Applied Mathematics, 2012, 8(4): 719-720.

[4] 袁明珠. Matlab 遗传算法工具箱在约束非线性惩罚函数中的应用[J]. 软件工程，2017，20（1）：37-39.

[5] Diver D A. Applications of Genetic Algorithms to the Solution of Ordinary Differential Equations[J]. Journal of Physics A General Physics, 1993, 26(14): 3503.

[6] Tsoulos I G, Lagaris I E. Solving Differential Equations with Genetic Programming[J]. Genetic Programming & Evolvable Machines, 2006, 7(1): 33-54.

[7] Jovanovski J, Jakimovski B, Jakimovski D. Parallel Genetic Algorithms for Finding Solution of System of Ordinary Differential Equations[J]. Italian Journal of Animal Science, 2011, 6(1): 619-621.

[8] Babaei M. A General Approach to Approximate Solutions of Nonlinear Differential Equations using Particle Swarm Optimization[J]. Applied Soft Computing, 2013, 13(7): 3354-3365.

[9] Si X Y, Lu J W, Zhang X. An Improved RNA Genetic Algorithm for the Parameter Estimation Multiple Solutions of Ordinary Differential Equations [J]. Procedia Engineering, 2017, 174(5): 477-481.

[10] Zhong D, Guo Q, Lin Q. Research of Ant Colony Algorithm Model of Road Selection[J]. Engineering of Surveying & Mapping, 2017, 2017(4): 47-52.

[11] Kanae S, Nakamichi M. An Ant Colony Optimization Method for Fuzzy Membership Functions and Its Application to Estimate the Pulmonary Elastance[C]// Proceedings of the 32nd Chinese Control Conference. Xian, Peoples R China, 2013: 8056-8060.

[12] Kamali M Z M, Kumaresan N, Ratnavelu K. Solving Differential Equations with Ant Colony Programming[J]. Applied Mathematical Modelling, 2015, 39(10-11): 3150-3163.

[13] Subbotin S A, Oleinik A A. Multiagent Optimization Based on the Bee-colony Method[J]. Cybernetics & Systems Analysis, 2009, 45(2):

177-186.

[14] Noghrehabadi A, Ghalambaz M, Ghalambaz M, et al. A Hybrid Power Series Artificial Bee Colony Algorithm to Obtain a Solution for Buckling of Multiwall Carbon Nanotube Cantilevers Near Small Layers of Graphite Sheets[J]. Applied Computational Intelligence & Soft Computing, 2012, 2012(6): 19.

[15] Fikri F F, Nuraini N. Use of Artificial Bee Colonies Algorithm as Numerical Approximation of Differential Equations Solution[C]// Proceedings of the Symposium on BioMathematics, Inst Teknologi Bandung, Bandung, Indonesia, 2018: 1063-1071.

[16] 张永恒, 金花, 王良璧. 基于粒子群和蚁群算法的边界层微分方程求解[J]. 兰州交通大学学报，2007，26（6）：1-4.

[17] 马翠, 周先东, 宋丽娟. 二阶线性常微分方程的两点边值问题的新解法[J]. 西南师范大学学报(自然科学版)，2010，35（4）：74-78.

[18] 杨鑫, 代鹏. 改进遗传算法的常微分方程求解及应用[J]. 辽宁工程技术大学学报，2015，2015（9）：1099-1104.

[19] 张建科, 王晓智, 刘三阳. 求解非线性方程及方程组的粒子群算法[J]. 计算机工程与应用，2006，42（7）：60-62.

[20] Mo Y, Liu H, Wang Q. Conjugate Direction Particle Swarm Optimization Solving Systems of Nonlinear Equations[J]. Computers & Mathematics with Applications, 2009, 57(11): 1877-1882.

[21] Abdelwahed W F, Mousa A A, Elshorbagy M A. Integrating Particle Swarm Optimization with Genetic Algorithms for Solving Nonlinear Optimization Problems[J]. Journal of Computational and Applied Mathematics, 2011, 235(5): 1446-1453.

[22] 李昌兵, 杜茂康, 付德强. 基于层次粒子群算法的非线性双层规划问题求解策略[J]. 系统工程理论与实践，2013，33（9）：2292-2298.

[23] 李爱国，覃征，鲍复民. 粒子群优化算法[J]. 计算机工程与应用，2002，38（21）：1-3.

[24] Zhou Z, Jiao B. The Improvement of Particle Swarm Optimization[C]// Proceedings of the 3rd International Conference on Systems and Informatics. Shanghai, China, 2017: 373-377.

[25] 苏运，毛宇晗，张焰. 基于改进量子粒子群算法的备自投投退策略优化方法[J]. 水电能源科学，2018，2018（3）：192-195.

[26] Eberhart R C, Shi Y. Comparison between Genetic Algorithms and Particle Swarm Optimization[J]. Lecture Notes in Computer Science, 998, 1447(7): 611-616.

[27] Angeline P J. Using Selection to Improve Particle Swarm Optimization[C]// Proceedings of the IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence. Anchorage, AK, USA, 2002: 84-89.

[28] Frans V D B. An Analysis of Particle Swarm Optimizers[D]. Pretoria: University of Pretoria, 2001: 3-14.

[29] Meissner M, Sehmuker M, Sehneider G. Optimized Particle Swarm Optimization and its Application to Artifical Neural Network Training[J]. BMC Bioinformatics, 2006(7): 125-130.

[30] Morsly Y, Aouf N, Djouadi M S. Particle Swarm Optimization Inspired Probability Algorithm for Optimal Camera Network Placement[J]. IEEE Sensors Journal, 2012, 12(5): 1402-1412.

[31] Mahesh K, Nallagownden P, Elamvazuthi I. Advanced Pareto Front Non-Dominated Sorting Multi-Objective Particle Swarm Optimization for Optimal Placement and Sizing of Distributed Generation[J]. Energies, 2016, 9(12): 982.

[32] Zhang C, Yu L, Shao H. A New Evolved Artificial Neural Network and Its Application[C]// Proceedings of the 3rd World Congress on Intelligent Control and Automation, Hefei, China, 2000: 1065-1068 .

[33] 高鹰，谢胜利. 基于模拟退火的粒子群算法[J]. 计算机工程与应用，2004，40（1）：47-50.

[34] Liu B, Wang L, Jin Y H. Improved Particle Swarm Optimization Combined with Chaos[J]. Chaos, Solitons and Fractals: 2005, 25(5): 1261-1271.

[35] 刘蜀阳，张学良，孙大刚. 基于极坐标复运算的粒子群优化算法[J]. 系统仿真学报，2006，18（7）：1794-1798.

[36] 郭广寒，王志刚. 一种改进的粒子群算法[J]. 哈尔滨理工大学学报，2010，15（2）：31-34.

[37] 黄太安，生佳根，徐红洋. 一种改进的简化粒子群算法[J]. 计算机仿真，2013，30（2）：327-335.

[38] 夏学文，刘经南，高柯夫. 具备反向学习和局部学习能力的粒子群算法[J]. 计算机学报，2015，2015（7）：1397-1407.

[39] 樊骥，吴红梅. 基于最优化理论几种算法的比较[J]. 才智，2011，2011（9）：53.

[40] Yu K, Wang X, Wang Z. An Improved Teaching-Learning-Based Optimization Algorithm for Numerical and Engineering Optimization Problems[J]. Journal of Intelligent Manufacturing, 2016, 27(4): 1-13.

[41] 拓守恒，雍龙泉，邓方安. “教与学”优化算法研究综述[J]. 计算机应用研

究，2013，30（7）：1933-1938.

[42] Chen D, Zou F, Li Z, et al. An Improved Teaching-Learning-Based Optimization Algorithm for Solving Global Optimization Problem[J]. Information Sciences An International Journal, 2015, 297(3): 171-190.

[43] 刘延龙. 布谷鸟算法的应用研究及算法性能度量[D]. 哈尔滨：东北林业大学，2016：14-15.

[44] 李宝家, 刘昊阳. 超定方程组的一种解法[J]. 沈阳工业大学学报，2002，24（1）：76-77.

[45] Zhong M L. Analysis on Application of Genetic Algorithm upon Solution of Ordinary Differential Equation[J]. Journal of Bohai University, 2009, 30(2): 158-161.

[46] 华冬英, 李祥贵. 微分方程的数值解法与程序实现[M]. 北京：电子工业出版社，2016：28-34.

[47] 张秋杰, 赵文星. 遗传算法在二阶微分方程定解问题中的应用[J]. 中国科技信息，2013，32（15）：52-53.

哈尔滨工业大学学位论文原创性声明和使用权限

学位论文原创性声明

本人郑重声明：此处所提交的学位论文《基于改进粒子群算法求解常微分方程定解问题》，是本人在导师指导下，在哈尔滨工业大学攻读学位期间独立进行研究工作所取得的成果，且学位论文中除已标注引用文献的部分外不包含他人完成或已发表的研究成果。对本学位论文的研究工作做出重要贡献的个人和集体，均已在文中以明确方式注明。

作者签名：张晶晶　　日期：2018 年 12 月 27 日

学位论文使用权限

学位论文是研究生在哈尔滨工业大学攻读学位期间完成的成果，知识产权归属哈尔滨工业大学。学位论文的使用权限如下：

（1）学校可以采用影印、缩印或其他复制手段保存研究生上交的学位论文，并向国家图书馆报送学位论文；（2）学校可以将学位论文部分或全部内容编入有关数据库进行检索和提供相应阅览服务；（3）研究生毕业后发表与此学位论文研究成果相关的学术论文和其他成果时，应征得导师同意，且第一署名单位为哈尔滨工业大学。

保密论文在保密期内遵守有关保密规定，解密后适用于此使用权限规定。

本人知悉学位论文的使用权限，并将遵守有关规定。

作者签名：张晶晶　　日期：2018 年 12 月 27 日

导师签名：张新明　　日期：2018 年 12 月 27 日

致　谢

在两年半的研究生生活中，首先要感谢我的导师张新明副教授对我的悉心指导。从确定课题方向，开题报告的确定到毕业论文的完成，张老师在每一步都耐心指导，为我指明方向，提供思路，帮助我克服困难。科研中，张老师治学严谨的态度深深地影响着我，使我对科学研究有了更严谨的态度，让我受益终生。生活中，张老师平易近人，和蔼幽默，这使我的科研生活倍感温馨，在此表示由衷的感谢和最诚挚的敬意！

同时，我还要感谢实验室的同学们，大家互相鼓励，共同进步，创造了一个良好的科研氛围，在学习中对我也有很大的帮助。

另外，感谢朝夕相处的室友刘雨和王梦杰，感谢她们在生活中对我的关心和照顾。硕士生活，不仅收获了知识，也收获了友谊，这些都将成为我人生中永不褪色的珍藏！

最后，感谢父母一直以来的支持与鼓励，感谢他们无私的付出，让我顺利完成论文。感谢在论文撰写期间给予过我帮助、支持与鼓励的朋友们。