Study on Efficient Search in Evolutionary Computation

裴, 岩

https://doi.org/10.15017/1441250

出版情報：九州大学, 2013, 博士（工学）, 課程博士
バージョン：
権利関係：全文ファイル公表済

Doctoral Dissertation of Engineering

Study on Efficient Search in Evolutionary Computation

March, 2014

Yan Pei

Kyushu University

Doctoral Dissertation of Engineering

博士 (工学) 学位論文

Study on Efficient Search in Evolutionary Computation

進化計算の効率的探索に関する研究

2014 年 3 月

Yan Pei

裴 岩

Kyushu University

九州大学

Abstract

Enhancing the search capability of evolutionary computation (EC) and increasing its optimization performance are important but have not completed yet. EC is applicable to high dimensional, non-linear, non-differentiable, and/or other hard problems. However, obtaining an optimal performance is still hard for practical EC applications. For example, user fatigue is a serious issue of applying interactive EC, and reducing fatigue is a practical requirement for its applications. As implementing an efficient search method in EC algorithm is one of the methods for reducing user fatigue, it is valuable to study on the efficient search methods for EC.

In this dissertation, we propose six novel approaches on this subject and discuss them within three research directions. They are: (1) approximating fitness landscape in lower dimensional search space and elite local search, (2) Fourier analysis on fitness landscape and its enhancement methods, (3) Fourier niche method for multi-modal optimization, (4) triple and quadruple comparison-based interactive differential evolution (IDE) and differential evolution (DE), (5) EC acceleration by the accelerating transition from exploration to exploitation, and (6) a new EC algorithm - chaotic evolution.

The first research direction among three directions in this dissertation is the fitness landscape approximation method that tries to obtain the knowledge of the problem structure and search condition in a search space. Once we obtain these kinds of information, we can propose specific search strategies, introducing local search to EC, and others to enhance EC search capability.

The second research direction is developing a new search mechanism. We propose a new triple and quadruple comparison-based IDE and DE, not only to enhance IDE search as well as reducing IDE user fatigue, but also to enhance canonical DE search. By introducing transition from exploration to exploitation, a new EC mechanism is proposed to enhance EC research performance.

The third research direction is developing new EC algorithms. We propose a new EC algorithm based on chaotic ergodicity. This idea is inspired by ergodicity of chaotic systems to combine with EC.

概 要

進化計算の探索能力を向上させて最適化性能を高めることは重要な課題であるが，まだ完成されたとは言えない．進化計算は高次元，非線形，変数間依存，その他の困難なタスクにも適用可能である．しかし，これまで得られている性能では，実用タスクに適用するにはまだ不十分である．例えば，ユーザ疲労は対話型進化計算の大きな課題であり，対話型進化計算の応用のためには疲労軽減が実用的なレベルで求められる．効率的な探索方法を進化計算に組み込むことはユーザ疲労軽減の一方法であり，進化計算のための効率的な探索方法の研究は意義あることである．

本学位論文では，3 つの研究方向における探索効率化のための 6 つのアプローチを提案し議論する．これらは，(1) フィットネス景観をより低い次元で近似する方法と近似で得られたエリート個体による局所探索，(2) フィットネス景観のフーリエ解析とその強調方法，(3) 多峰性最適化のためのフーリエ・ニッチ法，(4) 3 点および 4 点比較ベースの対話型差分進化と差分進化，(5) Exploration から Exploitation への遷移を加速することによる進化計算の高速化，(6) 新しい進化計算アルゴリズムーカオス進化.

本学位論文での 3 つの研究方向における第 1 の研究方向は，タスクの構造と探索空間での探索条件に関する知識を得ようとするフィットネス景観の近似法である．一旦このような情報が得られれば，特定の探索戦略，進化計算への局所探索の導入，その他の進化計算探索能力を向上させる方法などを提案することが可能になる．

第 2 の研究方向は，新しい探索メカニズムの開発である．3 点および 4 点比較ベースの対話型差分進化と差分進化を提案するが，これは，ユーザ疲労を軽減し対話型差分進化の性能を向上させるだけでなく，従来の差分進化法の性能も向上させることを目的にしている．Exploration から Exploitation への遷移の概念を導入することで，進化計算研究を展開させる新しい進化計算メカニズムを提案できた．

第 3 の研究方向は，新しい進化計算アルゴリズムの開発である．カオスのエルゴード性に基づく新しい進化計算を提案する．この考えは，カオスシステムのエルゴード特性を進化計算に組み合わせることから得られたものである．

摘 要

增强进化计算的探索能力和提高其优化性能是一个重要的、但尚未完成的研究课题。 进化计算适用于优化高维的、非线性的、不可微分的和其他一些难以解决的优化问题。 但是对于具体的应用，通过进化计算获得最优的优化结果，仍然是一个非常困难的工作。 举例来说，用户疲劳问题是交互式进化计算需要解决的关键问题， 当应用交互式进化计算解决具体问题时，减轻用户的疲劳是其实用化的具体需求。 由于在进化计算算法中实现更为有效的搜索是解决用户疲劳问题的方法之一， 所以在进化计算中实现更有效率的算法研究是一个非常有价值的研究课题。

在这篇学位论文中，作者从3个研究角度和方向出发，提出了6个创新性的研究方法和新型的进化计算算法。 这些方法和算法包括： (1) 在低维度的探索空间的适应值景观近似和精英个体搜索， (2) 适应值景观的Fourier分析以及应用其原理的进化计算加速方法， (3) 用于解决多峰优化问题的Fourier niche算法， (4) 基于3点和4点比较的交互式差分进化算法和差分进化算法， (5) 通过从全局探索向局部挖掘转化的进化计算加速方法， (6) 新的进化计算算法–混沌进化。

这篇学位论文中的第一个研究方向是适应值景观的近似。 这种方法试图通过适应值景观的近似，在探索空间中，获得被优化问题的知识和进化计算算法的探索状况。 一旦我们获得了这些信息，就可以利用其采取特定的探索策略、局部探索方法等增强进化计算的探索能力。

第二个研究方向是开发新的进化计算算法的探索机制。 作者提出了一个基于3点和4点比较的交互式差分进化算法和差分进化算法。 这些算法不仅可以有效地增强交互式差分进化算法的探索效率，从而减轻交互式差分进化用户的疲劳，而且有效地增强了经典差分进化计算的探索和查找。 通过进化计算的查找从全局探索向局部挖掘的转化，作者提出了一种新的加速进化计算的方法框架。

第三个研究方法是开发新的进化计算算法。 作者提出了一个基于混沌遍历性的进化计算算法。 算法的思想来源于混沌系统的遍历性和进化过程的迭代性。

Contents

Chapter 1

Introduction

1.1 Background and Remaining Problems

Computational intelligence (CI) is a conceptual set of nature inspired computational methodologies and approaches, which primarily includes fuzzy systems, artificial neural networks and evolutionary computation (EC). Many real world problems are mathematically ill posed and complex. They frequently have uncertainties, constraints, multimodal, multi-objectives, noisy and dynamic environments. Nature may provide us many practical solutions for complex real world problems or hints from biological, mathematical, or physical systems for solving them. Nature inspired CI has been applied to complex real world problems, while conventional methods are inefficient or infeasible for them and often fail to solve them.

There are many characteristics of CI. Intelligence is one of them, which is directly linked to reasoning and decision making that usually attribute to products and human life. Adaptivity is another important characteristic of CI, which is also covered by the fields of machine learning and computational neuroscience. CI also includes biology inspired algorithms such artificial immune systems that can be categorized as a part of EC. However, there is a controversial issue about whether swarm intelligence belongs to EC or not. Furthermore other formalisms, such as chaos theory, is used to construct computational models.

Fuzzy logic was originally introduced in 1965 by Zadeh, Lotfi, A [124]. It is a tool to formalize and represent the reasoning process [125]. Fuzzy logic systems are based on fuzzy logic that possesses many characteristics attributed to intelligence. Fuzzy logic can effectively handle the uncertainty that is common for human reasoning, perception and inference.

Artificial neural networks (ANN) were introduced in the 1940s that mimic the human brain [61], and they were further developed in 1980s. It represents a computational mechanism based on a simplified mathematical model of the neurons and signals that they process. ANN is used to solve a variety of tasks that are difficult to solve by conventional rule based programming. The subjects of learning paradigms, such as unsupervised learning, supervised learning and reinforcement learning, are important study topics in ANN.

EC started in the 1970s and has become popular in the late 1980s. It mimics population based evolution through reproduction of generation and handles many optimization problems, such as continuous optimization and combinatorial optimization problems, etc. Its algorithms can be considered as the global optimization methods with a meta heuristic or stochastic optimization characteristic. Most of EC are applied to black box optimization problems, which are often in the context of expensive, large scale, multimodal and multi-objective optimizations. EC uses iterative progress, such as growth or development in a population, and many variety of its algorithms are so called as population based optimization algorithms. EC selects better individuals, which are multiple search points based on a stochastic approach, and finds further better individuals using EC operators iteratively until found individuals reach to a desired stop criterion. Such processes are often inspired by biological mechanisms of evolution. EC has shown its powerful capability to solve complex industrial and engineering optimization problems in many real world applications.

Interactive EC (IEC) is an approach whereby such properties as human knowledge, experience and preference are embedded into an optimization process, and EC solutions are searched using fitness from a human evaluation based system. By embedding a human being itself into an optimization system, EC techniques become applicable to tasks for which it is difficult to construct an evaluation system or to measure their evaluations. For example, hearing aid fitting [112] and cochlear implant fitting [50] are tasks well suited to an IEC approach because there is few ways to measure how a human being hears sounds except the user's subjective responses. IEC has also been applied to artistic areas such as creating music or graphics, engineering areas such as sound and image processing, control and robotics, virtual reality, data mining, media database retrieval, and others, including geoscience, education, games, and many other tasks in various areas [108].

From a framework point of view, IEC can be implemented with any EC algorithm by replacing fitness function with a human user. Several EC techniques are used in IEC, such as interactive genetic algorithms [14], interactive genetic programming [100], interactive evolution strategy [31], human based genetic algorithm [44], interactive particle swarm optimization [56], interactive differential evolution (IDE) [113], as discussed in the next chapter, and others. Evaluation noise due to human subjective evaluations cannot be avoided, and therefore EC algorithms that are sensitive to noise do not show better performance when used as-is in IEC. From a practical point of view, when such noise sensitive EC algorithms are used in IEC, it is necessary to add an algorithm for overcoming noise sensitivity [69].

There are many directions in IEC research, such as expanding applications of IEC; expanding IEC frameworks [109]; applying IEC in a reverse engineering approach to analyze humans and thus advance [110]; accelerating IEC searches and improving IEC interfaces. The major remaining IEC problem is the problem of IEC user fatigue [108]. In addition to the previously mentioned acceleration and improvement of IEC interfaces, many approaches for reducing IEC user fatigue have been conducted. Some of these approaches improve methods of inputting fitness, use a combination of IEC and EC, allow users to intervene in EC searches, introduce IEC user evaluation models, construct new IEC frameworks and introduce paired comparison based fitness evaluation rather than evaluating all individuals at once.

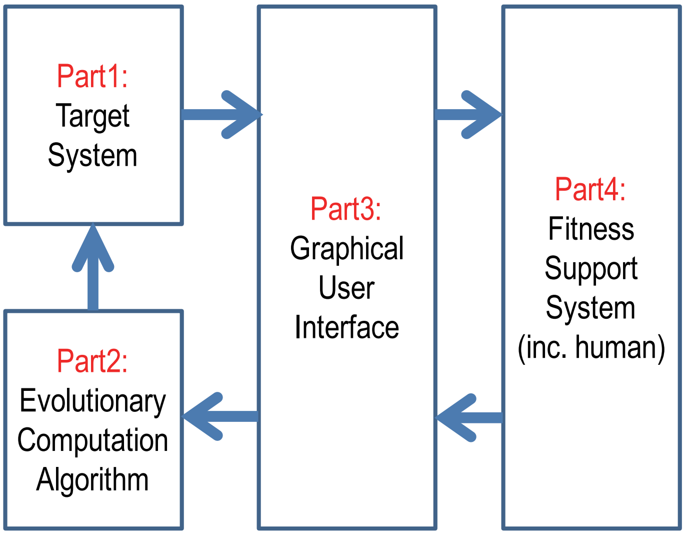

Part1:
Target System
Part2:
Evolutionary Computation Algorithm
Part3:
Graphical User Interface
Part4:
Fitness Support System (inc. human)

Figure 1.1: EC/IEC based optimization system, part 3 is only for IEC.

From a system optimization framework viewpoint, there are four parts in an EC/IEC based optimization system (Figure 1.1), which includes a target system that should be optimized, an EC algorithm that implements concrete optimization operations, a graphical user interface for IEC [108], and one or multiple fitness function(s) (including a human user for IEC). If it is an EC optimization system, there is no part 3 in the framework. These four parts encompass the corresponding four study aspects of an EC/IEC based optimization system, i.e., EC application study, EC algorithm study, IEC interface study and EC fitness function(s) or human science studies by an IEC human user, which involves EC/IEC theoretical research in knowledge discovery.

Although EC has shown powerful optimization capability for many complex and real world problems to which conventional optimization methods are ineffective, EC algorithm optimization capability still needs to be improved further to extend its applicability. How to improve EC and IEC search capabilities and reduce the user fatigue of an IEC application are primary focused contents and study subjects of this dissertation.

1.2 Objectives and Approaches

We focus on improving performance of EC and IEC algorithms, and try to find out efficient search methods and strategies for EC/IEC algorithms, which relate to the subjects of part 2, part 3 and part 4 studies in Figure 1.1.

The objectives of this work are:

1. developing some efficient search approaches and strategies to enhance EC algorithms,
2. designing well posed interfaces for better interaction and communication between IEC algorithms and a human user,

3. constructing a novel EC algorithm framework and fusing multiple schemes inspired by mathematical, physical or biological phenomena to implement a new EC algorithm, and
4. better and deeper understanding EC algorithm search processes and principles of EC algorithm through the study works of (1)-(3) mentioned above.

To achieve these research objectives from aspects of both theoretically understanding EC algorithm optimization principles and practically developing more effective EC algorithm search approaches and strategies, we propose to conduct three primary aspect works of EC algorithm research.

1. approximation of fitness landscape (part 4 in Figure 1.1),
2. development of search strategies and work schemes (part 2 and part 3 in Figure 1.1),
3. development of new algorithms and their analysis (part 2 in Figure 1.1).

Fitness landscape is an important information to assist EC search for a global optimum. Conventional methods are time consuming to approximate a fitness landscape in its original search space. To reduce computational complexity of approximation, a dimensionality reduction technique and a local search method are proposed in approximation process. In a mathematical analysis viewpoint, there is frequency information in any search spaces. If we can obtain the frequency information and use it to implement fitness landscape approximation, it is a novel method to deeply understand search space and to develop an effective approximation method to express fitness landscape. Fourier transform is applied to obtain the frequency information of search space. Based on the obtained principal frequency component, a local search method is applied to enhance EC search. Likewise a Fourier niche method is developed to solve multimodal optimization problems using the same principle.

It is a forever subject in the EC research community to develop better search methods and strategies for obtaining an enhanced EC algorithm. Triple and quadruple based IDE and differential evolution (DE) is developed, which designs an effective interface for interaction and communication between an IDE human user and IDE algorithm. The triple and quadruple points come from opposite points of target and trail vectors in DE based on opposition based learning theory. From theoretical analysis, the proposed search schemes are effective in both IDE and DE. Other novel methods are proposed based on transition from exploration to exploitation, which shows potential enhanced performance of EC convergence.

EC algorithms are originally inspired by the natural phenomena, especially from biological phenomena that genetic algorithm mimics. In recent decades, EC researchers came to focus on optimization schemes from not only biological phenomena, but also physical and mathematical ones. In chaos theory, chaotic ergodicity and output distribution of a chaotic system are a well posed characteristic. If it is fused with evolutionary iterative scheme, an efficient search framework is discovered and further developed. We call this new EC algorithm as *Chaotic Evolution* that is inspired by chaotic ergodicity and evolutionary iteration.

The final objective of these studies is to obtain an accelerated EC optimization performance. The terms “accelerated”, “accelerating” and “acceleration” here refer to obtain the global optimum with less generations, less fitness evaluations, less EC algorithms' computational complexity or computational cost, and so on. If our proposal can obtain a solution with less resources than other methods, or it obtains a better solution than other methods even if it uses more resources, we may say that the proposed method is an accelerated EC algorithm or can obtain a better acceleration performance.

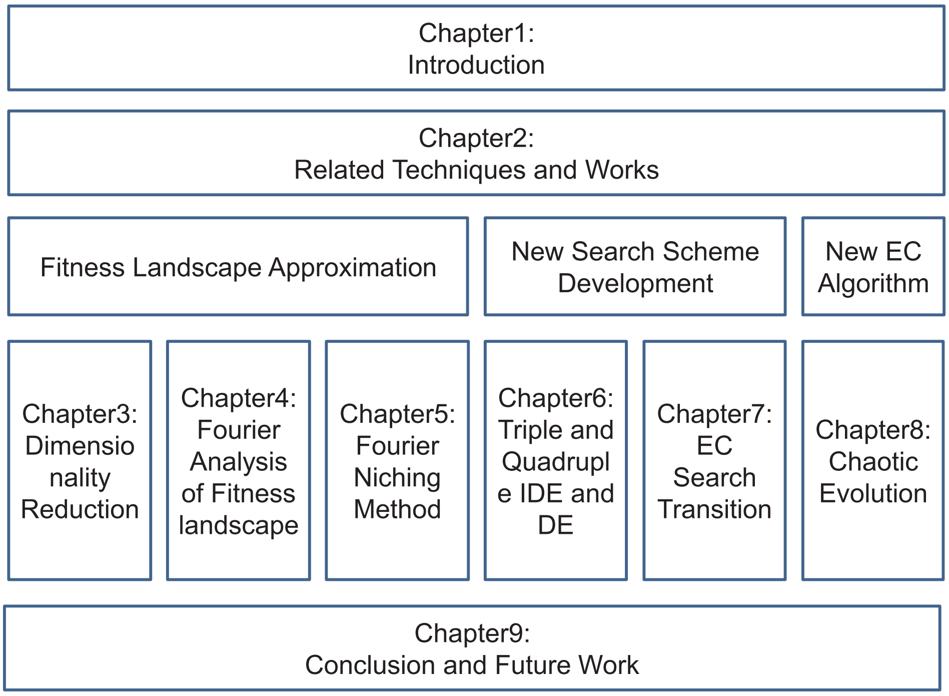

Chapter1:
Introduction
Chapter2:
Related Techniques and Works
Fitness Landscape Approximation
New Search Scheme Development
New EC Algorithm
Chapter3: Dimensionality Reduction
Chapter4: Fourier Analysis of Fitness landscape
Chapter5: Fourier Niching Method
Chapter6: Triple and Quadruple IDE and DE
Chapter7: EC Search Transition
Chapter8: Chaotic Evolution
Chapter9:
Conclusion and Future Work

Figure 1.2: Dissertation's chapter structure.

1.3 Chapter Structure

Following this introductory chapter, an overview of related techniques and approaches on fitness landscape approximation, Fourier analysis and Fourier transform, opposition based learning and chaos theory is presented in chapter 2. Comprehensive survey on conventional EC enhancement methods is reported, and future research directions are discussed. In chapter 3, we introduce several techniques on fitness landscape approximation by dimensionality reduction, theoretically and practically discuss computational complexity of the proposal. The elite combination search approach enabled by a technique for reducing dimensionality of search space is explained. In chapter 4 and chapter 5, a Fourier analysis method on fitness landscape is designed to enhance EC and IEC search, and a Fourier niche method is proposed to solve multimodal optimization problems. A triple and quadruple based IDE and DE is presented and reported in chapter 6. We also discuss the related issues on the human model and interface design. In chapter 7, a new mechanism of transition from exploration to exploitation for accelerating EC is proposed. In chapter 8, we report a chaotic ergodic property based EC algorithm, *Chaotic Evolution*. We describe inspiration and development of the chaotic evolution in detail. Finally, we conclude our primary contributions, limitations and future works in chapter 9.

From the structure viewpoint, chapters 3, 4 and 5 are the studies on fitness landscape approximation and their applications in ordinary EC and IEC acceleration and multimodal optimization. Chapters 6 and 7 are the study on effective search scheme and mechanism in existent EC algorithms. Chapter 8 is a study on the new EC algorithm development. The visual view of chapter structure of this dissertation is shown in Figure 1.2.

Chapter 2

Related Techniques and Works

2.1 Related Techniques

The relationship between related techniques and each chapter is explained as follows. Differential evolution algorithm is described in section 2.1.1, it is a basic test algorithm used in our evaluations of proposals. Approximation techniques are reported in section 2.1.2, which is applied in chapter 3. Fourier analysis in section 2.1.3 is used for obtaining frequency information in chapters 4 and 5. Opposition based learning in section 2.1.4 is the primary tool to implement triple and quadruple comparison mechanisms in chapter 6. Chaos theory in section 2.1.5 is basic knowledge to implement a new EC algorithm, chaotic evolution, in chapter 8.

2.1.1 Differential Evolution

EC comprises bionic optimization algorithms. EC simulates the production and evolution process of all the lives and intelligence agents. It has been developed not only on the basis of Darwin's principles of natural selection and *survival of the fittest*, but also have utilized the theory of genetic by Gregory Mendel to maintain an optimized result, meanwhile attempting to find a better solution. EC is a probabilistic search optimization technique, which uses computational models of evolutionary processes as key elements in design and implementation of computer based problem solving systems [22]. There have been several well defined EC algorithms, which have served as the basis for most activities in the field of EC: genetic algorithms (GA) [32], evolution strategies [96, 97, 98], genetic programming [20, 45], evolutionary programming [64] and differential evolutions (DE) [87, 103].

DE is one of population based EC algorithms. It searches for a global optimum using a differential vector from two individuals which length is in proportion to distribution size of individuals in general. Each parent individual generates its offspring. As a parent's individual is replaced with the generated one only when the fitness of a generated one is better then that of the parent, we may say that DE operations have a similarity to an elite strategy or hill climbing method. The distinctive feature of DE is powerful search capability with quite simple algorithm.

Suppose that an array on the left side of Figure 2.1 means individuals, contour lines at the right side is a fitness landscape, and circles on the landscape are individuals. DE algorithm for one search generation is described in the below and repeats

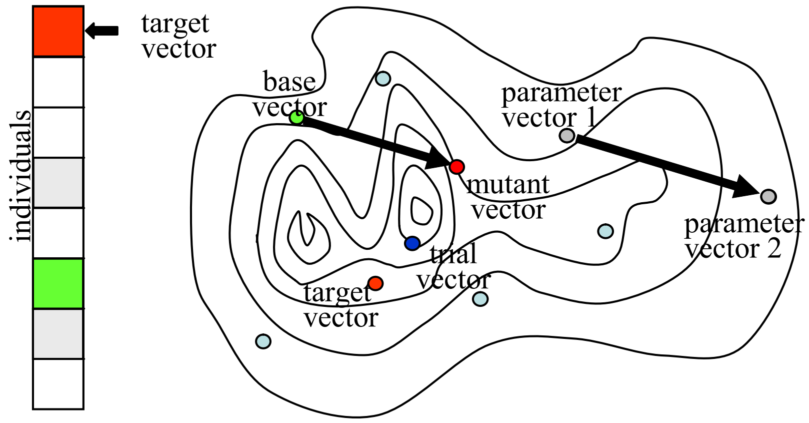


Figure 2.1: Differential evolution algorithm.

until a satisfied solution(s) is(are) found or the search reaches to the maximum generations.

(1) Choose one individual to be the target vector ($x_{i,G}$).
(2) Select two other individuals ($x_{r_2,G}$ and $x_{r_3,G}$) as parameter vectors randomly and derive a differential vector from them.
(3) Select the best individual from the rest of individuals or another individual randomly as the base vector ($x_{r_1,G}$).
(4) Create a mutant vector ($v_{i,G}$) by adding a weighted differential vector to the base vector.
(5) Generate a trial vector by crossing the target vector and the mutant vector.
(6) Compare the fitness of the target vector and the trial vector, and select whichever one is better as the offspring in the next generation.
(7) Go to the (1) and generate other offspring until all individuals are processed using the same operations. Then proceed with the next generation.

$$v_{i,G} = x_{r_1,G} + F * (x_{r_2,G} - x_{r_3,G}) \tag{2.1}$$

The terms of *vector* and *individual* mean the same searching points. The above steps (1) – (4) are summarized as Eq. (2.1), which shows the DE algorithm is easily implemented; where F is called a scale factor. There are several DE variations in the number of differential vectors, selection methods of a base vector in the step (3), crossover methods in the step (5), and others.

DE is an algorithm that can control balance of exploration and exploitation automatically thanks to a differential vector which average length is in proportion to the distribution size of individuals. We can say that DE searches around a base vector by narrowing its search area gradually because differential vectors have different lengths and different directions and are added to the base vector to find search points (in the step (4)). Speaking in more detail, DE biases the search area to each target vector side (in the steps (5) and(6)).

Several strategies for improving DE performance have been proposed, such as SaDE [88], JADE [89], jDE [7] and JASaDE [25]. Their strategies include setting DE parameter randomly, choosing better strategies from search histories, creating new mutation strategy, and fusing those strategies together. Approaches for improving DE performance are roughly categorized into three: parameters tuning methods, strategy setting methods, i.e. strategy pool architecture, and strategy selection methods.

2.1.2 Interpolation and Approximation Approaches for Approximating Fitness Landscape

2.1.2.1 Regression Function Selection

The computation of each EC generation produces a set of discrete individuals and their corresponding fitness $(x_i, y_i), i = 0, 1, ..., m$, but the analytical expression relating the two cannot be known. We make the data regression curve for the required function from the given category Φ. Because there is error in the discrete data, we do not require interpolated or approximated curves pass through all the discrete points accurately, but rather that it approaches the original curve at its discrete points, i.e. near x_i, as much as possible.

We let the function $\varphi(x)$ belong to Φ, and $\delta_i = \varphi(x_i)$ - y_i be the error at $\varphi(x_i)$, such that the error vector is $\delta = (\delta_0, \delta_1, ..., \delta_m)^T$. In the regression calculation process, we let the norm of the vector $||\delta||$ be minimized. For a different norm, we can construct a different regression function. The function set is shown in Eq. (2.2).

$$\Phi = Span\{\varphi_0(x), \varphi_1(x), ..., \varphi_n(x)\} \tag{2.2}$$

The members in Eq. (2.2), i.e. $\varphi_0(x)$, $\varphi_1(x)$, ..., $\varphi_n(x)$, are linearly independent in the interval $[a, b]$, where the nodes x_i are contained. Any type of function $\varphi(x)$ can be used in Eq. (2.3).

$$\varphi(x) = a_0\varphi_0(x) + a_1\varphi_1(x) + ... + a_n\varphi_n(x) \tag{2.3}$$

The regression process brings additional cost to EC, so a better solution is to select a simple function as the regression function to reduce its additional computational cost. The power function is suitable for regression computing, we select it as the regression function in our approximation and interpolation approaches.

2.1.2.2 Lagrange Interpolation Approach

The Lagrange interpolation polynomial has the characteristic of being linear and unique. For one dimensional data from individuals $x_0, x_1, \ldots, x_n$, we can set up an n degree polynomial $l_0(x), l_1(x), \ldots, l_n(x)$. We set its type as $l_i(x_j) = \delta_{ij}$, where the form of δ_{ij} is as shown in Eq. (2.4).

$$\delta_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases} \tag{2.4}$$

From this, we can obtain $p_n(x) = \sum_{k=0}^{n} l_k(x)y_k$, the n degree interpolation polynomial, where $l_i(x)$ is an nth degree polynomial. We can obtain the relationship shown in Eq.s (2.5), (2.6), (2.7) and (2.8).

$$p_n(x) = \sum_{k=0}^{n} l_k(x) y_k \tag{2.5}$$

$$l_k(x) = a(x - x_0)...(x - x_{k-1})(x - x_{k+1})...(x - x_n) \tag{2.6}$$

When $l_k(x) = 1$,

$$a = [(x_k - x_0)...(x_k - x_{k-1})(x_k - x_{k+1})...(x_k - x_n)]^{-1} \tag{2.7}$$

i.e.,

$$l(x) = \prod_{i=1, i \neq k}^{n} \frac{(x - x_i)}{(x_k - x_i)} \tag{2.8}$$

Eq. (2.8) is the n degree Lagrange interpolation benchmark function, and the n degree Lagrange interpolation polynomial is shown in Eq. (2.9).

$$L_n(x) = \sum_{k=0}^{n} l_k(x) y_k, i = 0, 1, ..., n \tag{2.9}$$

We use two degree Lagrange interpolation polynomial as the simplified regression fitness landscape expression, and use the n individuals with relative better fitness as the interpolation points to obtain a concrete regression function form. The concrete interpolation polynomial is as the Eq. (2.10). The Lagrange interpolation approach is used to find the parameters of Eq. (2.10).

$$L(x) = \sum_{k=1}^{n} \left\{ \prod_{i=1, i \neq k}^{n} \frac{(x - x_i)}{(x_k - x_i)} \right\} y_k \tag{2.10}$$

2.1.2.3 Least Squares Approximation Approach

As it was mentioned above in section “Regression Function Selection”, we want to minimize the norm of the error vector $\delta = (\delta_0, \delta, ..., \delta_m)^T$. If we use the 2-norm form vector as the error vector, the calculation process is simplified. When we use the 2-norm as the error vector norm, the approximation approach is called the least squares approach. Given data in form of the function set, i.e., Eq. (2.2), we want to find a function that lets the error 2-norm vector to be minimized, namely, Eq. (2.11).

$$||\delta^*||_2^2 = \sum_{i=0}^{m} \delta_i^{*2} = \sum_{i=0}^{m} [\varphi^*(x_i) - y_i]^2 = \min_{\varphi(x) \in \Phi} ||\delta^*||_2^2 \tag{2.11}$$

The approximation function is Eq. (2.12).

$$\varphi^*(x) = a_0^* \varphi_0(x) + a_1^* \varphi_1(x) + ... + a_n^* \varphi_n(x) \tag{2.12}$$

If we can obtain $\varphi_0(x), \varphi_1(x), \ldots, \varphi_n(x)$, the system will be orthogonal, i.e. $(\varphi_i, \varphi_j) = 0 (i \neq j)$, and the coefficient matrix equations will form a diagonal matrix (Eq. (2.13)).

$$a_k^* = \frac{(f, \varphi_k)}{(\varphi_k, \varphi_k)} \tag{2.13}$$

So the approximation function is Eq. (2.14).

$$\varphi^*(x) = \sum_{i=0}^{n} \frac{(f, \varphi_i)}{(\varphi_i, \varphi_i)} \varphi_i(x) \tag{2.14}$$

We use the two degree Lagrange interpolation polynomial to obtain the regression fitness landscape by interpolation in the interpolation methods. For the approximation in our regression approach, we use the one degree power function as the regression function to approximate the fitness landscape in linear space, with the concrete expression as shown in Eq. (2.15). The approximation process is used to obtain the parameters D and E in Eq. (2.15) such that regression fitness landscape may be defined in a linear space.

$$\begin{pmatrix} (\varphi_0\varphi_0)(\varphi_0\varphi_1) \\ (\varphi_1\varphi_0)(\varphi_1\varphi_1) \end{pmatrix} \begin{pmatrix} D \\ E \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \tag{2.15}$$

2.1.3 Fourier Analysis for Analyzing Fitness Landscape

2.1.3.1 Fourier Transform

Since Jean Baptiste Joseph Fourier issued his classical paper on the distribution of temperature by the trigonometric function in 1807, he proposed that any continuous periodical functions can be expressed by a set of combination of sinusoidal function. Fourier transform have been used in many scientific and industrial societies as a powerful mathematical analysis tool. There are four types of Fourier Transforms, i.e. Fourier transform, Fourier series, discrete time Fourier transform and discrete Fourier transform (DFT), which can process the signals that are continuous and aperiodic, signals that are continuous and aperiodic, signals that are discrete and aperiodic, and signals that are discrete and aperiodic, respectively. Because only the infinite signals can be transferred in Fourier analysis, we have to make the prolongation original signal, which copy its existent part or set zero value in the undefined region, to cope with this restriction.

Due to the computer can only process the discrete signals, the discrete Fourier transform is valuable to the application subject in many areas. Eq.s (2.16) and (2.17) show the discrete Fourier transform and inverse discrete Fourier transform (IDFT), which is the main tool that we use it to analyze the EC fitness landscape.

$$X(k) = DFT[x(n)] = \sum_{n=0}^{N-1} x(n) W_N^{kn} \tag{2.16}$$

$$x(n) = IDFT[X(k)] = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-kn} \tag{2.17}$$

In Eq.s (2.16) and (2.17), W_N^{kn} is $W_N^{kn} = e^{-\frac{2k\pi}{N}}$, and $0 \leq k \leq N-1$. The terms $x(n)$, $X(k)$, n and N are the original signal series, frequency signals by DFT in frequency space, number of original signal and number of transform base frequency, respectively. For one time DFT, its computational complex is up to $O(N^2)$, so it is costly for introducing DTF into EC to analyze fitness landscape complexity and to apply to practical application.

2.1.3.2 Fast Fourier Transform in One Dimension

For reducing the DFT computational cost, in 1965, Cooley and Tukey proposed a fast discrete Fourier transform (FFT) [12]. The principal of FFT is to cut the long series into the shorter ones and to use the periodic and symmetric characteristics to reduce the calculation times. There are two kinds of FFT, one is decimation in time algorithm (DIT) and the other is decimation in frequency algorithm (DIF).

$$X(k) = x(2r)W_N^{2rk} \pm W_N^k x(2r+1)W_N^{(2r+1)k} \tag{2.18}$$

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1}[x(n) + (-1)^{(k)}x(n+\frac{N}{2})]W_N^n k \tag{2.19}$$

Eq.s (2.18) and (2.19) show the fundamental principle of the DIT and DIF, respectively, in Eq. (2.18), mark + when $k \in [0, \frac{N}{2}-1]$, mark - when $k \in [\frac{N}{2}, N-1]$. They can reduce the computational complex from $O(N^2)$ to $O(N \log N)$. We use DIF as the main analysis tool to transfer original EC individual series into frequency space for obtaining frequency information of original fitness landscape.

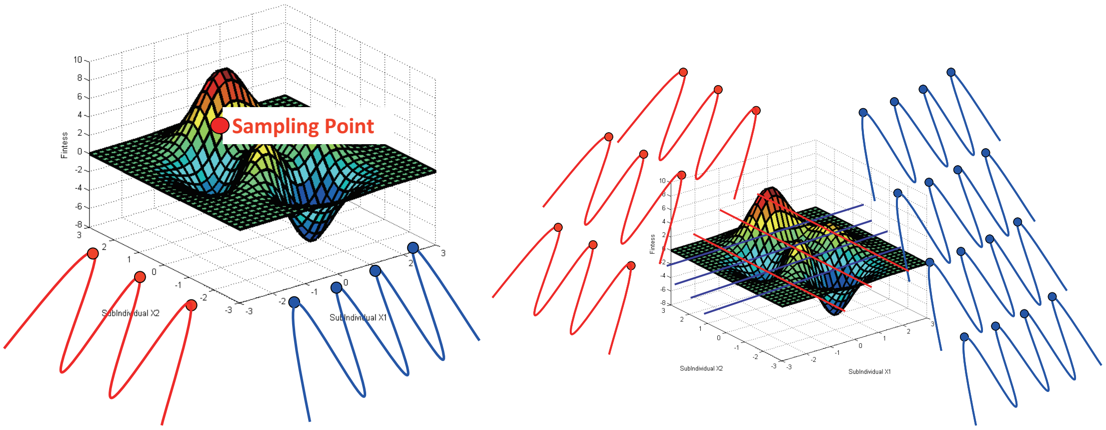


Figure 2.2: 1-D and n-D dimension DFT.

2.1.3.3 Fast Fourier Transform in n Dimension

N dimension fast Fourier transform (n-D FFT) is considered as conducting one time 1-D FFT with M_i points when the other dimensional value is fixed, and this kind of 1-D FFT conducts $\Pi_{k=1,k\neq i}^{n} M_k$ times. M_i is the sampling point in the i-th dimension. For a brief explanation, the 2-D FFT and inverse 2-D FFT are Eq.s (2.20) and (2.21) (see Figure 2.2).

$$X(k,l) = [\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x(m,n)W_M^{km}W_N^{ln}]R_{M,N}(k,l) \tag{2.20}$$

$$x(m,n) = \frac{1}{N}[\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} X(k,l)W_M^{-km}W_N^{-ln}]R_{M,N}(m,n) \tag{2.21}$$

$$R_{M,N}(m,n) = \left\{ \begin{array}{ll} 1 & 0 \leq m \leq M-1, 0 \leq n \leq N-1, \\ 0 & \text{otherwise} \end{array} \right. \tag{2.22}$$

M and N are the ranges of original 2 dimensional range, m and n are the respective sampling data in each dimension, l and k are the FFT base frequency number in each 1 dimensional FFT, respectively. The computational complexity of 2 dimensional FFT is $O((m+n)N \log N)$, i.e. $(m+n)$ times of 1 dimensional FFT.

2.1.4 Opposition Based Learning for IDE and DE Enhancement

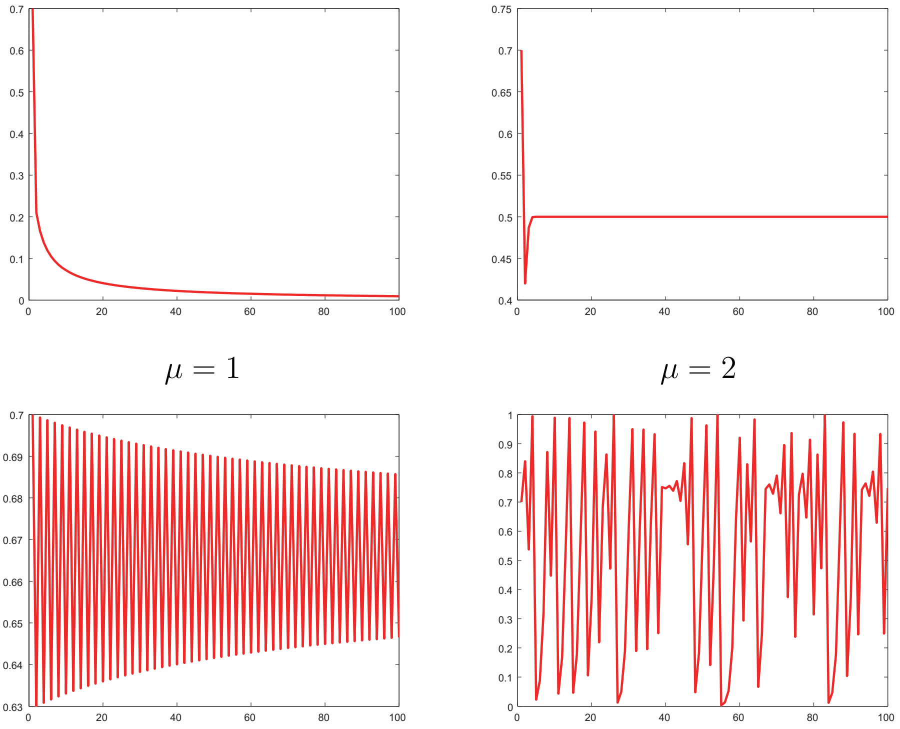
Opposition based learning (OBL) [114] is used in machine learning [115] and to accelerate optimization searches. Suppose that $x \in [a,b]$ is a real number, then the opposition point of x is given by $OP(x) = a + b - x$. By extending this idea to a multi-dimensional space, the opposition point, $OP(X)$, of a point on a n-dimensional real space, X =$(x_1, x_2, ..., x_n)$ $(x_i \in [a_i, b_i], i = 1, 2, ..., n;\ a_i, b_i \in R)$, is given by Eq.s (2.23) and (2.24).

$$\begin{aligned} OP(X) &= \{OP(x_1), OP(x_2), ..., OP(x_n)\} && (2.23) \\ OP(x_i) &= a_i + b_i - x_i && (2.24) \end{aligned}$$

OBL optimization uses the opposite point to accelerate EC search, and two acceleration approaches are widely used: OBL based initialization of individuals and OBL based generation of offspring. The former generates opposite points for randomly generated individuals, and chooses the better of the randomly generated individuals and their oppositions as the initialized individuals for the first generation. When a population size is small, the risk becomes higher that randomly initialized individuals do not cover a search space evenly, and are biased to certain areas, and are trapped in local minima. The OBL based initialization can reduce this risk and help EC to start its search from better sub areas from the global viewpoint.

The latter switches between two strategies for generating offspring with a probability called the *jumping rate*. The one strategy is to generate offspring based on ordinary EC operations and the other is to generate offspring by comparing pairs of parent individuals and their opposite points. Acceleration approaches using opposition points are based on two hypotheses. One is that most population based optimization methods are of a stochastic nature and when the search point is located far from the global optimum, the probability that the opposition point is better becomes high. Another is that the probability of the search point or its opposite being better is usually equivalent.

OBL has been applied to several EC algorithms, and an opposite based differential evolution (OBDE) that embeds the two previously mentioned OBL techniques into a conventional DE was been proposed [90]. Since then, several further variations have also been proposed: OBDE applied to shuffled DE [1], generalized OBDE extended by introducing an opposite search space and four schemes for making the opposition point [116], a new DE applying an opposition operator for a mutant vector and leaving the winner to the next generation [35], and others.



$\mu = 1$ $\mu = 2$

$\mu = 3$ $\mu = 4$

Figure 2.3: Logistic map when initial value is 0.7 and μ is set to 1, 2, 3, and 4; X- and Y- axes are iterations and system outputs, respectively; when μ is 4, the system shows the chaos.

2.1.5 Chaos Theory for New EC Inspiration

2.1.5.1 An Overview of Chaos Theory

The subject beginning of chaos theory can be traced back to the time when Jules Henri Poincare studied on the famous Three-body problem. He found that there are some orbits which are non periodic, and yet not forever increasing up nor approaching to a fixed point [19]. His research results in the production of chaos theory and its studies. The early research of chaos theory involved in the mathematical area and many of them were all directly inspired by physical problems. The modern study of chaos theory is formalized to explain the system behavior, and it is easy to be simulated in computer visibly. The famous application of chaos is in weather prediction by Edward Norton Lorenz [55]. In his research, he had discovered that small changes in initial conditions produced large changes in the long term outcome [105]. This is a simplest explanation of chaos characteristic and phenomenon. Another research topic, Fractal and fractal dimension, was studied by Benoit Mandelbrot [58], which is a classic of chaos theory. It presents that the dimensional number not only is a integrate number, but also can be a fractional number.

It is difficult to define a system as a chaotic system by using a strict language of mathematics. However, there are some explicit properties and characteristics to explain and judge chaotic system and phenomenon. First, the chaotic system is

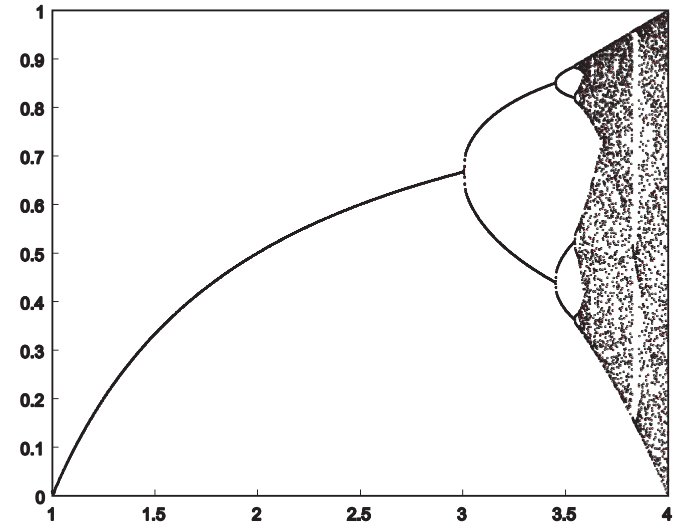


Figure 2.4: Bifurcation diagram of logistic map. X- and Y- axes are μ values and system outputs, respectively.

sensitive to the initial condition. If the initial input value of a chaotic system is changed with a small difference, the outcome of the system can obtain a big change. Second, the chaotic system is topological mixing and its periodic orbits are dense. Third, the chaotic system is a simple system that exhibits complex behavior, which cannot be explained by using a conventional theory. However, it is as well as a complex system that exhibits the behavior, which seems random and unstructured, but it has an underlying order. In the mathematical version, the concept of chaos is famous as sensitive dependence, determinacy and nonlinearity.

2.1.5.2 Logistic Map

Logistic map shows that simple nonlinear equation can result in the chaos, which originally is designed to express the demographic model [60]. Eq. (2.25) shows the mathematical form of the logistic map (x ∈ R), and when the parameter μ is set to 4, the system behaviour will result in the chaos, which exhibits a great sensitivity to initial conditions.

$$x_n = \mu x_{n-1}(1 - x_{n-1}) \tag{2.25}$$

Figure 2.3 sketches the logistic map system outputs when parameter μ is set to 1, 2, 3 and 4, and the initial value is set to 0.7. When μ is set to 1 and 2, the system becomes stable. When μ is set to 3, on a small scale, the system shakes; however, on a large scale, it becomes convergent and stable. When μ is set to 4, the system results in the chaos. The entire system behavior of logistic map is shown in its bifurcation diagram, which has an explanation of population biology when μ is set to a different value (Figure 2.4).

2.2 A Survey of EC Search Enhancement Approaches

2.2.1 Coding Methods

The coding technology is a basic issue for EC algorithm design, and different coding methods influence the algorithm performance and concrete applications. For example, many coding technologies have been proposed in GA. These include messy GA [23], Delta coding GA [120], dynamic parameter encoding GA [95] and real coding GA [17]. These coding techniques develop EC search space and make a basement for a variety of search strategies in corresponding search space. Coding method influences EC search performance.

2.2.2 Initialization and Selection for Reproduction

2.2.2.1 Orthogonal Experimental Design Based Population Initialization

Reference [51] used orthogonal experimental design method in the initialization to generate the population. The problem of this method is selection of the orthogonal experimental factor and level. Whether the pre designed orthogonal experimental parameter is feasible to enhance EC or not, it should be investigated furthermore.

2.2.2.2 Hybrid Population Construction Methods

It is an important EC acceleration method to construct a new population from one generation to the next. In this process, there are two key aspects to be considered. One is to keep population diversity. The objective is to let generated individuals distribute in the whole search space averagely. The other is to generate individuals near the global optimum as much as possible for obtaining that easily.

Reference [122] proposed a virtual population method to reconstruct a new population to achieve these two objectives. A hybrid population construction method using elitist selection and stochastic universal sampling approaches were proposed [9]. A pipeline based hyper population construction was proposed as well [99].

2.2.2.3 Dynamic Fitness Threshold

Fitness threshold method gives higher priority to the fitter individual. It is initialized by the user, and only if the new individual is greater than the fitness threshold one, the new individual is put into the population, otherwise this randomly generated individual is not considered. The fitness threshold value will be tuned by considering the previous generation. The policy improves the fitness values, which will always be increasing from one generation to the next. When the population has lack of diversity, the EC algorithm may fail into the premature trap. Research on how to use the dynamic fitness threshold method and avoid the local optimum could therefore be promising.

2.2.2.4 Fitness Scaling

Fitness scaling technology can be used to tune the individual fitness. It uses some transformations to avoid premature convergence in the selection operation, but this handling destroys the evaluation rule of a certain EC algorithm. For interactive EC application, it may be a good approach to assist human's evaluation, which is sometimes inconsistent.

Fitness scaling technology assumes that best solution in current population is closest to global optimum. If this assumption is true, then searching around the best solution will generate solutions, which are close to the global optimum. Conversely, if the assumption is false, this approach will lead to a local optimum. We will therefore search the opposite direction to avoid the problem (failing into local optimum region), and search beyond the best solution in the current population to find the global optimum and escape the local optimum. References [122] and [121] use an method to generate a population that conducts a local search near the best solution to find a global optimum and avoids the local optimum to solve load flow problem in power system.

2.2.3 Alternative Operations

2.2.3.1 Estimation of Distribution Algorithm

Estimation of distribution algorithms (EDAs) were introduced in the field of EC [2], which is a new area of EC. In EDAs there is neither crossover nor mutation operator. New population is generated by sampling the probability distribution, which is estimated from a database containing selected individuals of previous generation. Different methods have been proposed for the estimation of probability distribution. EDAs learn the structure of the search space and use this knowledge to generate the offspring so as to improve the performance of the algorithm. Accordance with the relationship between the variable, there are three types EDAs, i.e. independent variables [2, 30, 67, 68], bivariate dependencies [3, 15, 84] and multiple dependencies [29, 47, 83].

2.2.3.2 Surrogate Model Based EC

There are two purposes to use surrogate model embedded into EC. Firstly, it is to shorten the evaluation time for obtaining fitness from a surrogate model, because some fitness function evaluations needs more time. Secondly, there are no explicit fitness function for EC in some the real world applications, so a surrogate model needs to be established for fitness evaluation. Several methods were proposed to be used in surrogate model, such as ANN [34, 72], statistical model [40], regression [126], radius based function [93] and Markov model [8].

2.2.3.3 Engineered Conditioning Operation Methods

Engineered conditioning operator [66, 86, 102] is the same conventional optimization search strategy, i.e. *moving to the best adjacent point in the search space.* Engineered conditioning uses the dominant individuals of the current population in a search space and compares their strength. Three local searches are performed, the superset test, the substitute test, and the subset evaluation test. These tests search for better individuals of greater cardinality, equal cardinality, and lesser cardinality.

Reference [86] proposed the engineered conditioning operator based GA to accelerate convergence on multiple fault diagnosis application. Reference [66] extended the research work of [86], and considered engineered conditioning operator as local improvement operators [24] that is specified by domain. A more substitute test was implemented in the engineered conditioning operator (E2C based GA) to accelerate GA convergence.

2.2.3.4 Age Conception Evolution Based Directed Search Method

Age conception evolution based directed search method is based on two conceptions in evolution process. One is the age, the other is direction [42]. The age shows a diversity feature of population, and the direction shows whether the next evolution is forwarding to a global optimum region.

The advantage of this method accelerates EC convergence without decreasing diversity of the individuals by two operations. One is a direction operation that determines search direction according to the fitness. The other is a zero mean Gaussian operation, which is used for a perturbation and is added to a parent to generate an offspring. Reference [42] uses this approach to solve the seven parameters friction model optimization problem in system cybernetics.

2.2.3.5 Elitist Model

Some EC algorithms use elitist model to preserve *the best fitness individual obtained so far* [9]. Reference [6] proposed two methods to keep individual with the best fitness. One makes two copies of the best fitness individual in the last population, and places one of them in the new population. The other compares the child and its parent, the better one survives.

In recent EC research, most of the EC algorithm use the elitist model strategy. Some EC convergence theory is based on the elitist model. For EC application and convergence theory without the elitist model, it may be a challenging topic in future EC research

In single objective GA (SGA), elitism is an operation that ensures the best chromosome found so far exists in the next generation. This can be realized by simply copying the best individual to the next generation. For multi-objective optimization, the elitism strategy no longer remains trivial, as there is no longer a single best design to copy. Because of this, various elitist strategies have been proposed [128]. A simple elitist method is that parents compete with offspring [13]. The offspring population is combined with their parent population, and entire individuals are ranked. The best of these individuals are retained as the parents for the next generation.

2.2.4 Hybrid EC Search Methods

2.2.4.1 Local Search Algorithm

Local search algorithms move from one solution to another solution in a search space, until a global optimum is found or a time bound is elapsed. Adding a randomization step to a gradient search algorithm can improve local search performance. However, for many large and complex search spaces, this method may not be efficient. Many experiments have shown that such randomization has little effect. For hybrid EC with a local search algorithm, different neighbour relation strategy constructs different accelerated EC algorithm [11, 118]. Reference [127] proposed local search and global search with a surrogate model to accelerate EC algorithm.

2.2.4.2 Constraint Satisfaction

Constraint satisfaction is a process of finding a solution to a set of constraints that impose conditions that some variables must satisfy. A solution is therefore a vector of variables that satisfies all constraints. The techniques used in constraint satisfaction depend on the kind of constraints being considered. Most of constraint satisfaction based EC are incomplete in general, because they may prove it unsatisfied, but not always [121]. This method uses some constraint rule, which is based on specialty domain knowledge to modify the unsatisfied individuals after crossover and mutation operation, It conducts the local search near the unsatisfied individual space to check whether the optimum exists.

2.2.4.3 Simulated Annealing

Simulated annealing (SA) is a generic probabilistic meta heuristic method for the global optimization problem, namely locating a good approximation to the global optimum of a given function in a large search space. There is an ability to escape the local optimum by incorporating a probability function in accepting or rejecting a new solution. A cooling schedule has been used to accelerate convergence. It is often used when the search space is discrete.

Reference [59] proposed a SA method, which is used to accelerate convergence of GA by applying a SA test for all populations. The SA test allows the acceptance of any individuals at the initial steps in search, but only the good individuals have the priority to be accepted as the generation increases. The experimental evaluation shows that the SA method is efficient in helping GA to escape from local optimum to prevent the premature convergence.

2.2.4.4 Artificial Neuron Network

The artificial neuron network (ANN) is a mathematical model or computational model that simulates the structure and/or functional aspects of biological neural networks. It consists of an interconnected group of artificial neurons and processes information using a connectionist methodology to compute. An ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase. Modern neural networks are nonlinear statistical data modelling tools. They are usually used to model complex relationships between inputs and outputs or to find patterns in data [107].

There are two promising research topics in ANN based EC. Firstly, EC uses ANN as a directional tool to find the best search direction. It uses landscape information of an EC search space to train the ANN and with this information the search direction is decided [62]. Secondly, EC uses a trained ANN to confirm an input-output relationship for special applications with domain knowledge. Normally, three layer structure ANN, which has an input layer, a hidden layer and an output layer, is considered to be used in these applications. Multilayer feed forward ANN with error back propagation algorithm has been applied to accelerate GA [46].

2.2.5 Stop Criteria and Performance Definitions

An important but always ignored problem that is to design proper stop criteria for EC algorithm. EC convergence concept is a good tool for designing the stop criteria, such as Markov chain, fixed point theory, etc. However, the convergence concept is time unlimited in describing the system behavior. For some complex problem, we cannot use it to design the stop criteria. There are several methods proposed for designing the stop criteria [4, 27, 94]. They include fitness boundary value method, time boundary value method, individual or generation number boundary value method, individual or generation number boundary value method and fitness number boundary value method, etc.

Algorithm performance is based on static relations applied during the execution of an algorithm. There are two types of performance in EC [26]. On-line performance is defined to reflect the character of average fitness of the individuals. During execution of EC algorithm, on-line performance converges to a stable value, i.e. stationary state, and the solutions formed by the algorithm become stable. The off-line performance is similar to the on-line performance, but it gives a lot of importance to the best fitness value. Off-line performance converges to a stable value while the number of evaluations is increasing, and the probability of finding a better solution is decreasing quickly. Therefore, the optimization process can be stopped to avoid any waste of time.

2.2.6 Parameter Setting

An important topic on EC operations is the parameter value setting problem. For example, GA mutation operation adjusts diversity of a population and partially decides GA convergence. If mutation rate is small, but the convergence speed is high, the population has lack of diversity, and GA often converge to a local optimum and lead to the premature convergence. However, mutation with a high rate may result in the loss of the better individuals. Another primary genetic operation is crossover, which is designed to generate offspring in the hope that better fitness is achieved through exchanging partial genetic information of two parents. A valuable research topic is how to balance EC convergence and performance by tuning EC parameter, and design an adaptive EC operation rate or find more efficient operations [118].

2.3 Discussions on Prospective Researches

We have given an overview of accelerating EC convergence methods in the last two decades in section 2.2, and presented the related research works of EC acceleration approaches applications. In this section, we will present some brief proposals for further research on topics of EC acceleration.

EC operations do not directly use the landscape information of search space. If we can use more landscape information directly, EC convergence may be accelerated. For this idea, there are three concrete approaches. The first is the approximation of landscape with a simpler shape to find new elite in the new search space. The second is to project search space to other dimensional search space to obtain easy search information to find the global optimum in the projected space. The third is that we may conduct efficient search strategies in the original space and projected space, it can help accelerating or observing the EC convergence, and adjusting different search strategies adaptively. The three concrete approaches on accelerating EC convergence are the primary works in our further research.

2.3.1 Fitness Landscape Approximation

If we can reduce the complexity of the search space, it can become easier to reach to a global optimum in a approximation search space. It is not the real global optimum but may be a neighbour around the global one. From this viewpoint, we can use the gradient search, local search or some related search algorithms to find the global optimum in the approximation search space. Therefore, it is easy to reach to the real global optimum from the neighbour point. For the concrete approximation landscape methods, there are so many mathematical approaches or computation based that can be used, such as space frequency information, curves, signal processing filtering, differential information, polygons, and clustering, etc.

2.3.2 Search Space Dimensionality Transformation

If we can use some projection algorithms to project EC individuals from its original space to other higher or lower spaces, and conduct some search strategies to search, it should be efficient and EC convergence may be accelerated.

Suppose that original search space is n-D space and projection space is k-D space ($k > n$), support vector machine projects data onto a higher dimensional space and finds linear separation in the k-D space for the nonlinear separation of classification in the n-D space. If we can apply the same way of thinking to global optimization, we can reach to the global optimum in the k-D space quickly, and then search the global optimum in the n-D space from the neighbour point.

Suppose that we project search points, which are in a n-D, onto m-D space ($m < n$), it reduces dimensional complexity though some pieces of information are gone. In the lower dimensional search space, we search the global point in a simpler space, and then search the real global optimum in its original search space. This is the same approach mentioned above.

2.3.3 Search Strategy

When we transfer a search space from A to B (*SpaceA* to *SpaceB*), there are several search or observation strategies, which can provide the information for the algorithm execution. Suppose that the original search space is *SpaceA* (the individuals are all in *SpaceA*), and we use some projection methods to map the *SpaceA*'s individual into another space, and we call it *SpaceB*. In *SpaceB*, we create the new individuals (the neighbours of the old ones), and then three strategies are possible.

The first strategy is to map new neighbour individuals back to *SpaceA* as a new generation, then to search in *SpaceA*, and then to map to *SpaceB* again to find new individuals, looping this process until the global optimum is found. The second strategy is to keep search in space B, and when the some fitness values meet a certain condition, we map the individuals back to *SpaceA* again to judge how to deal with them further, then stop or map to *SpaceB* to continue to search. The third strategy is to search a global optimum in the *SpaceA* as the similar to conventional EC search, but to monitor the search situation of the *SpaceA* in *SpaceB* and feedback search control to the search in the *SpaceA*.

The main research point of the search strategy is to find or design the projection that helps EC search. The choice for strategies mentioned above will be decided after analyzing the characteristics of the projection.

2.3.4 Fusion with Other Soft Computing Methods

Conventional computational methods could model and precisely analyze only relatively simple systems. More complex systems arising in biology, medicine, humanities, management sciences, and similar fields often remained intractable to conventional mathematical and analytical methods. So in the early 1990s, a so called soft computing [125] technology was proposed, which includes EC, neural network (NN), fuzzy system (FS) and other computation intelligence technologies. Ever since their proposal, fusion of these technologies has been an active research direction [106, 107], such as EC+NN, NN+FS, and so on.

Soft computing deals with imprecision, uncertainty, partial truth, and approximation to achieve tractability, robustness and low solution cost. For a different technique in soft computing that has its unique features. It is worth investigating how to use one technique's advantage to compensate the other technique's deficiency. For example, NN has the memorial function, and the GA is a search optimization tool. It might be good to fuse both techniques in such a way that the memorial function of NN is used to direct the search direction of GA so that the GA convergence can be accelerated [62]. Fusion of multiple soft computing techniques is a promising research direction for obtaining an accelerated EC algorithm in EC community.

Chapter 3

Accelerating IEC and EC Searches with Elite Obtained in Projected Lower-Dimensional Spaces

3.1 Introduction

The success of evolution in nature outstrips that of evolutionary computing even though it is based on the same principles. This can be attributed primarily to the abundance of resources in nature, such as time and memory, which are constrained in a computer system. This is especially true in the case of systems that involve human interactivity, such as Interactive Evolutionary Computation (IEC). Consequently, accelerating EC is necessary for many EC applications to improve the performance of their target systems. Consider the case of an IEC that optimizes a target system based on the IEC user's subjective evaluations. User fatigue is a serious problem limiting such a system's practical application. Multiple trials for accelerating EC have therefore been proposed [108].

In this Chapter we introduce interpolation or approximation approaches to obtain the fitness landscape in a dimensionally reduced search space. Reaching the global optimum is easier in the regression space by finding the elite. Although it is not an actual global optimum, it may be close to the global optimum in the original search space [74, 75]. Finding the actual global optimum from the elite is therefore an easier task.

Here, *regression space* refers to a lower dimensional space consisting of k dimensional (k-D) axes of the original n-D axes, where we use $k = 1$ in this chapter. In other words, individuals are projected onto the k-D space to simplify finding the elite. This elitism does not destroy the original EC search space by approximation, but can accelerate the EC convergence with less computational cost by conducting the elite search in the n regression spaces of k-D ($k = 1$). This is the originality contributed by this work.

Following this introduction, we explain in detail the elite collaborative search approach enabled by a technique for reducing the dimensionality of the search space, and we show how the elite can be obtained from the regression search space in section 3.2. In sections 3.3, experimental evaluations are performed using a Gaussian mixture model with various differing dimensions and 34 benchmark functions, the results are compared with the previous work of [111]. Finally, we discuss our proposed methods and obtained results in section 3.4. In section 3.5, we conclude on the performance of our proposed approaches presented here and discuss some open topics, further opportunities and future works.

3.2 Obtaining Elite From Regression Search Spaces

3.2.1 Dimensionality Reduction Method

It is not easy for conventional interpolation or approximation approaches to find an accurate regression search space corresponding to the original multi-dimensional space. An alternative approach is to reduce the dimensionality of the original search space and find approximate curve expressions in the lower dimensional spaces.

However, if the parameter variables of the fitness function are dependent, when they are separated into the lower dimensional space, the dimensionality reduction approach will destroy the original search space information and the dependent parameter variables' relationship. Although there is this risk that some information will be lost when reducing the search space dimensionality, it is easy to interpolate or approximate the search space to obtain landscape information in lower dimensional search spaces with e.g. one or two dimensions of the original search space.

Our approach for reducing the dimensionality of the search space uses only one of the n parameter axes at a time instead of all n parameter axes, and projects individuals onto each 1-D regression space. The landscape of the n-D parameter space is given by a fitness function, $y = f(x_1, x_2, ..., x_n)$, and the fitness value of the m-th individual is given by Eq. (3.1).

$$y_m = f(x_{1m}, x_{2m}, ..., x_{nm}) \; (m = 1, 2, ..., M) \tag{3.1}$$

There are M individuals with n-D parameter variables. We project the individuals onto the n 1-D spaces in i-th dimension as follows.

$$(x_{i1}, y_1) \; (x_{i2}, y_2) \; ... \; (x_{im}, y_m)$$

Each of the n 1-D regression spaces has M projected individuals. The original search space and dimensionally reduced search space are shown in Figure 3.2. The dimensionality reduction approach simplifies the regression computations, and it is easy to obtain a regression search space in lower dimension, which can be helpful in situations which involve a higher dimensional nonlinear search space.

3.2.2 Method for Simplifying Fitness Landscape to Select Elite

We interpolate or approximate the landscape of each 1-D regression space using the projected M individuals and select elite from the n approximated 1-D landscape shapes. In this chapter, we test two approaches for approximating the 1-D regression search spaces; in the first approach we use a Lagrange two-degree polynomial interpolation and in the other linear least squares approximation. Elite are generated from the resulting approximated shapes (see Figure 3.1).

Finding the elite corresponds to a kind of local search in the area where relatively better individuals exist in the original search space. The global optimum is expected to be near this area and we may be able to find it with a probability higher than chance [111]. So the elite selection approach is a critical step in the proposed acceleration processes.

As the elite obtained by the two elite selection approaches is different, it is expected that the acceleration performance will also differ. Further, the regression EC search space obtained by approximation or interpolation has its own characteristics and particularities, and we must use an efficient approach to obtain an elite from this simplified search space after analyzing its characteristics.

Lagrange two-degree polynomial interpolation simplifies a regression space with a non-linear curve, and it is easy to obtain its inflection point from its gradient, using the inflection point as the elite. The linear least squares approximation uses a linear function to approximate the regression space. Its gradient is either descent or ascent. Unlike an inflection point, a safer approach, taking into account both descent and ascent, is to select the average point of the linear approximation line as the elite (see Figure 3.1).

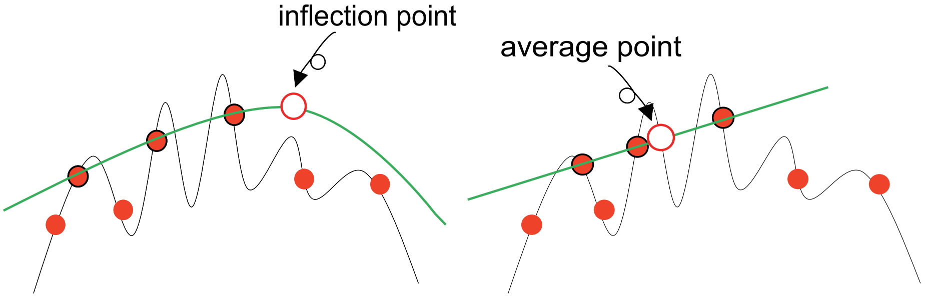


Figure 3.1: New elite selection approaches from a regression search space. Left: by Lagrange interpolation approach; Right: by least squares approximation approach.

Our proposed approaches replace the worst individual in each generation with an elite selected as above. Although we cannot deny the small possibility that the global optimum is located near the worst individual, the possibility that the worst individual will become a parent in the next generation is also low; removing the worst individual therefore presents the least risk and is a reasonable choice.

3.2.3 Method for Synthesizing Elite

The methods in section 3.2.2 select n elite points in n 1-D regression spaces, respectively: $x_{1-elite}$, $x_{2-elite}, ...,$ and $x_{n-elite}$. The n-D elite used for accelerating EC convergence in the next generation is obtained as follows:

$$\text{New Elite} = (x_{1-elite}, x_{2-elite}, .., x_{n-elite}).$$

It is easier to calculate elite in a lower dimensional space than a higher dimensional space. Although we use 1-D as the lower dimensional space in this chapter, in general the method need not be restricted to 1-D. After we obtain the elite, we only use this one elite into the next generation, if its fitness is better than that of worst individual.

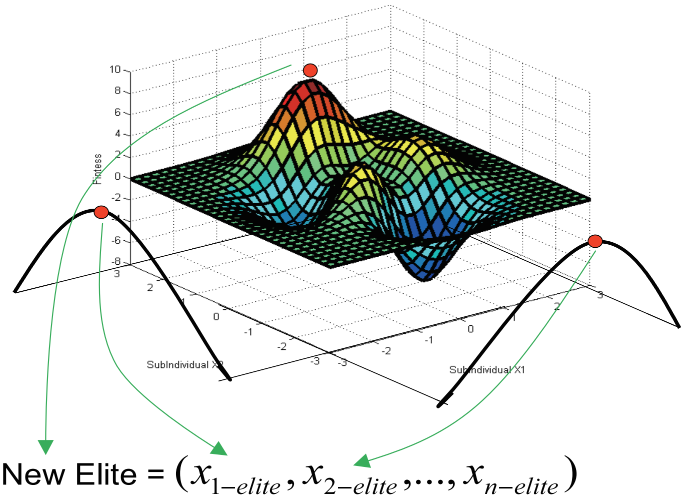


Figure 3.2: Method for synthesizing elite. This proposed method represents a novel local search approach for accelerating EC convergence.

Once a new elite has been obtained, there are two approaches for how it can be handled. In one cautious approach, the fitness value of this elite is calculated to determine whether the new elite is really useful for acceleration, and in the other straightforward approach, the new elite is inserted into the next EC iteration process without any prior consideration or judgement. In our proposed approaches, we choose the first method. If the fitness of elite is better than the worst one, we place it into next generation.

Our proposed method is based on the hypothesis that elite calculated by interpolation or approximation from relatively better individuals will also have good fitness; synthesizing an n-D elite from n elite points in n 1-D regression spaces will also produce a good elite; the probability of the global optimum being located near to the synthesized elite is high. (see Figure 3.2)

In general, this proposed method represents a novel local search approach for accelerating EC convergence, and it is this approach that represents this method's original contribution.

3.3 Experimental Evaluation

3.3.1 Methods for Comparison

Table 3.1 shows our proposed methods and conventional methods for comparison. We use differential evolution (DE) [103] in our experiments, as mentioned above, our proposed methods can be considered as a framework embedded in any IEC and EC algorithms to accelerate their searches, not limited in DE.

Table 3.1: Conventional and our proposed methods for experimental comparisons.

(I)DE-N	Canonical DE. [103]
(I)DE-TB	fitting a single peak function using n best individuals.[111]
(I)DE-TN	fitting a single peak function using n distance nearest best individuals.[111]
(I)DE-TA	fitting a single peak function using all individuals.[111]
(I)DE-LS	fitting a linear function using least square approximation.
(I)DE-LR	fitting a binomial function using Lagrange interpolations.

3.3.2 IEC Experimental Evaluation and Analysis

3.3.2.1 IEC User Model and Experimental Conditions

Experimental evaluations frequently request many repeated experiments under the same conditions, and evaluations with an IEC user model is necessary for this case rather than a real human IEC user. Our IEC user model [69] was designed based on four specifications of (1) a relatively simple fitness landscape, (2) a multimodal fitness landscape, (3) a big valley structure, and (4) parametrically controlled the shape and complexity of a fitness landscape. The rationale of the (1) is that a human IEC user cannot distinguish differences less than the differential threshold of perceptions nevertheless he/she can obtain practical solutions. That of the (2) is the fact that there are graphics, design, music, and others whose fitness values are high but their expressions are quite different. That of the (3) is that an IEC user can reach to the global optimum area easily in spite of the (2).The feature of the (4) is essential to conduct experiments with gradually changed several fitness landscapes.

A Gaussian mixture model (GMM) was established as pseudo-IEC user to simulate the user's evaluation in Reference [69, 113]. The GMM consists of different means, variances and peaks mixed together to express the characteristics of a human user conducting an IEC evaluation experiment. We use GMM for evaluation in this section. Concretely, we combine four Gaussian functions ($k = 4$) and realize the characteristics expressed by F27 in the Appendix in 3 dimensions (3-D), 5-D, 7-D, and 10-D. Its 3-D example of a Gaussian mixture mode is shown in the Appendix as well.

The big difference between an IEC user model and ordinary fitness functions is the implementation of (a) relative and (b) discrete fitness evaluations of a human user. Human IEC user compares given individuals relatively and does not give absolute fitness values unlike fitness functions. He or she also cannot give precise fitness values but discrete ones, e.g. 1 to 5 points, every generation, while ordinary fitness functions give continuous values. When the difference of individuals is less than the minimum discrete fitness range, i.e. an evaluation threshold, a human IEC user cannot distinguish the difference, and it becomes fitness noise that IEC user models should realize.

Table 3.2: IEC experiment parameters setting.

population size	20
search range of parameters	$[-5.12, 5.12]$
scale factor F	0.9
crossover rate	0.8
IDE operation	IDE/best/1/bin
max. search generation, MAX_{NFC}	20
dimensions of Gaussian mixture model, D	3,5,7,10
# of trial runs	50

IDE user simulation in this section randomly chooses either a trial vector or a target vector and leave it as offspring in the next generation when the difference of their fitness values are less than a certain value to simulate an unavailability of a human IDE user's comparison; we set the difference threshold as 1/50 of the difference between the maximum fitness value and the minimum fitness value in population at each generation. IDE parameters are set as the Table 3.2. Population size, 20, is decided to take account of IEC experiments with a real human user.

3.3.2.2 Evaluations of the Proposal

Figure 3.3 shows the average convergence curves of the best fitness values for 50 trial runs of IDE-LS and IDE-LR with their competitors using GMM with 3D, 5D, 7D and 10D. Table 3.3 shows the average fitness value at the 20th generation.

From these results, in general, our proposed methods are able to accelerate all Gaussian mixture models, i.e., our proposed methods are significantly better than normal IDE, except IDE-LR method applied to a 3-D model. It indicates that our proposed acceleration methods can be effectively used in some IEC applications. However, for some cases, our proposed method are not more effective than the previously proposed acceleration method.

To obtain an estimation of the performance of our proposed acceleration methods under real IEC application conditions, we conducted IEC evaluation experiments with 20 individuals per generation. In Table 3.3, it can be seen that the acceleration performance improves with higher dimensions, i.e. the 10D, 7D and 5D GMM with 20 individuals is more efficient than in the 3D case. This indicates that our proposed methods increase the diversity of the population in higher dimensions more significantly than in lower dimensions. When the dimension of the GMM is higher, our proposed methods seem to offer the same performance.

The proposed method IDE-LS and IDE-LR can significantly outperform IDE-TB in lower dimensional problems (3-D and 5-D) and IDE-TN in higher dimensional problems (7-D and 10-D), respectively. These observations indicate that our proposed methods may work better than IDE-(TB, TN, TA) when the distribution of individuals in a search space has less diversity, while IDE-(TB, TN, TA) have better performance even in such cases. Compared with IDE-TA method, all of our proposed methods outperform better than it, except IDE-LS applied in a 7-D modal.

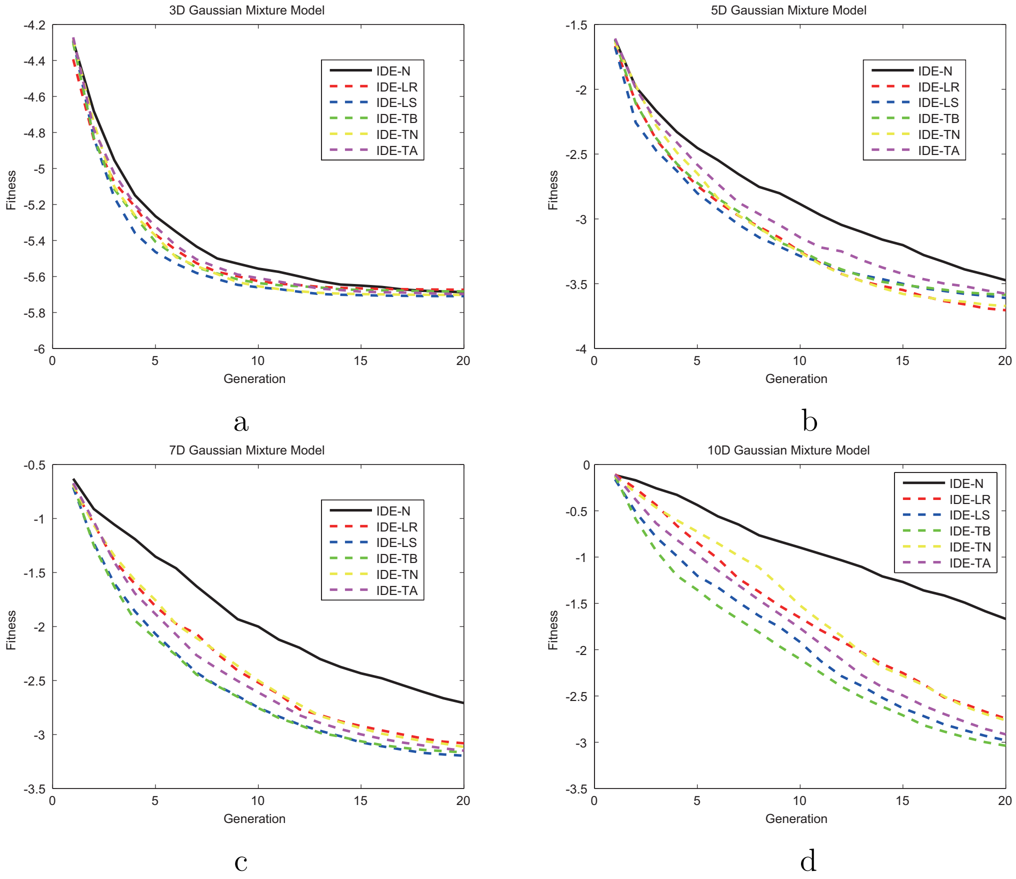


Figure 3.3: Average convergence curves of 50 trial runs for (a) 3D, (b) 5D, (c) 7D and (d) 10D Gaussian mixture model with population size of 20. Average fitness values at 20th generation are shown in Table 3.3.

Table 3.3: Average fitness values at 20th generation of 3-D, 5-D, 7-D and 10-d Gaussian mixture models (pseudo-IDE user). Bold font, †, ‡, §marks values mean IDE-LS or IDE-LR significantly better than IDE, IDE-TB, IDE-TN and IDE-TA, respectively, by Wilcoxon sign-ranked tests ($p < 0.05$).

Method	3-D	5-D	7-D	10-D
IDE	-5.59	-3.28	-2.77	-2.76
IDE-TB	-5.59	-3.37	-3.25	-3.14
IDE-TN	-5.76	-3.38	-3.22	-2.74
IDE-TA	-5.58	-3.33	-2.83	-2.98
IDE-LS	**-5.75†§**	**-3.39†‡§**	**-2.78**	**-3.03§**
IDE-LR	-5.58§	**-3.37§**	**-3.27‡§**	**-3.07‡§**

Table 3.4: Benchmark functions used in experimental evaluations, where Range is the scale of the parameters, n is the dimension of the function, C is the function's characteristics, respectively. U=Unimodal, M=Multimodal,Sh=Shifted, Rt=Rotated, GB=Global on Bounds, N=non-separable, and S=separable.

No.	Name	Test function	Range	n	C
F1	Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$	[-5.12,5.12]	3	U-S
F2	Rosenbrock	$f(x) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2$	[-2.048,2.048]	2	U-N
F3	DeJong-Step	$f(x) = \sum_{i=1}^{n} \lfloor x_i \rfloor$	[-5.12,5.12]	5	U-S
F4	Quantic & Noise	$f(x) = \sum_{i=1}^{n} i x_i^4 + Gauss(0,1)$	[-1.28,1.28]	30	U-S
F5	Shekel's Foxholes	$f(x) = [0.02 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}]^{-1}$	[-65.536,65.536]	2	M-S
F6	Rastrigin	$f(x) = (10n) + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i))$	[-5.12,5.12]	5	M-S
F7	Schwefel 2.26	$f(x) = \sum_{i=1}^{n} (-x_i \sin(\sqrt{\lvert x_i \rvert}))$	[-512,512]	5	M-S
F8	Griewank	$f(x) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}})$	[-512,512]	5	M-N
F9	Schaffer 1	$f(x) = 0.5 + \frac{\sin^2(\sqrt{(x_1^2 + x_2^2)}) - 0.5}{[1.0 + 0.001(x_1^2 + x_2^2)]^2}$	[-100,100]	2	M-N
F10	Schaffer 2	$f(x) = (x_1^2 + x_2^2)^{0.25} [\sin^2(50(x_1^2 + x_2^2)^{0.1}) + 1.0]$	[-100,100]	2	M-S
F11	Schwefel 2.22	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	[-10,10]	30	U-N
F12	Schwefel 1.2	$f(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} (x_j))^2$	[-10,10]	30	U-N
F13	Hartman-3	$f(x) = -\sum_{i=1}^{4} c_i \exp[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2]$	[0,1]	3	M-N
F14	Step	$f(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	[-10,10]	30	U-S
F15	Beale	$f(x) = (1.5 - x_1 + x_1 x_2)^2$ $+(2.25 - x_1 + x_1 x_2^2)^2$ $+(2.625 - x_1 + x_1 x_2^3)^2$	[-4.5,4.5]	2	U-N
F16	Kowalik	$f(x) = \sum_{i=1}^{11} [a_i - \frac{x_1(b_i + b_i x_2)}{b_i^2 + b_i x_3 + x_4}]^2$	[-5,5]	4	M-N
F17	Carnel-Back	$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 - 4x_2^4$	[-5,5]	2	M-N
F18	Branin	$f(x) = (x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6)^2$ $+10(1 - \frac{1}{8\pi}) \cos(x_2) + 10$	[-5,10]*[0,15]	2	M-S
F19	Goldstein-Price	$f(x) = [1 + (x_1 + x_2 + 1)^2 *$ $(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2)] *$ $[30 + (2x_1 - 3x_2)^2 *$ $(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2)]$	[-2,2]	2	M-N
F20	Hartman-6	$f(x) = -\sum_{i=1}^{4} c_i \exp[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2]$	[0,1]	6	M-N
F21	Sh Sphere	$same\ as\ F1$	[-100,100]	50	Sh-U-S
F22	Sh Schwefel 1.2	$same\ as\ F12$	[-100,100]	50	Sh-U-N
F23	Sh Rt Elliptic	$f(x) = \sum_{i=1}^{n} (10^6)^{\frac{i-1}{n-1}} x_i^2$	[-100,100]	50	Sh-Rt-U-N
F24	F21 with Noise	$same\ as\ F22\ add\ noise$	[-100,100]	50	Sh-U-N
F25	Sh Schwefel 2.6 GB	$f(x) = Max\lvert A_i x - B_i \rvert$	[-100,100]	50	Sh-U-N
F26	Sh Rosenbrock	$same\ as\ F2$	[-100,100]	50	Sh-M-N
F27	Sh Rt Griewank	$same\ as\ F8$	[0,600]	50	Sh-Rt-M-N
F28	Sh Rt Ackley GB	$f(x) = -20\exp(-0.2(\frac{1}{n}\sum_{i=1}^{n} x^2)^{\frac{1}{2}})$ $-\exp(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)) + 20 + e$	[-32,32] [-32,32]	50	Sh-Rt-M-N
F29	Sh Rastrigin	$same\ as\ F6$	[-5,5]	50	Sh-M-S
F30	Sh Rt Rastrigin	$f(x) = (10n) + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i))$	[-5,5]	50	Sh-Rt-M-N
F31	Sh Rt Weierstrass	$f(x) = \sum_{i=1}^{n} (\sum_{k=0}^{kmax} [a^k \cos(2\pi b^k (x_i + 0.5))]$ $-n\sum_{k=0}^{kmax} [a^k \cos(2\pi b^k * 0.5)])$	[-0.5,0.5]	50	Sh-Rt-M-N
F32	Schwefel 2.13	$f(x) = \sum_{i=1}^{n} (\sum_{j=1}^{n} (a_{ij} \sin x_i + b_{ij} \cos x_i))^2$	$[-100, 100]$	50	Mul-N
F33	Sh Expanded F8F2	$f(x) = F8F2(x_1, x_2, ..., x_n)$	[-3,1]	50	Sh-M-N
F34	Sh Rt Scaffer F6	$f(x) = 0.5 + \frac{\sin^2(\sqrt{(x_1^2 + x_2^2)}) - 0.5}{[1.0 + 0.001(x_1^2 + x_2^2)]^2}$	[-100,100]	50	Sh-Rt-M-N

Table 3.5: EC experiment parameters setting.

population size	100(D=50, D=30)/30(others)
scale factor F	0.9
crossover rate	0.8
DE operations	DE/best/1/bin
max. search generation	50
# of trial runs	50

3.3.3 EC Experimental Evaluation and Analysis

3.3.3.1 Benchmark Functions and Experimental Conditions

We evaluate our proposed methods with 34 benchmark functions. Our proposed methods aim to reduce IDE user fatigue by accelerating IDE search. However, they are not limited to use only for IEC but applicable to any EC searches with fitness functions. The information about some of these functions and their related parameters can be found in the Appendix. See definitions, search ranges of optimization parameters, and characteristics of the benchmark functions in Table 3.4.

Evaluation indexes for comparing our proposed methods applied for EC are average fitness value until convergence reaches to the maximum search generation, Wilcoxon signed-rank test at the maximum search generation and total time cost statistics and analysis, which can show how time-saving proposals are our proposed methods.

EC experimental parameters are set as the Table 3.5. We use differential evolution (DE/best/1/bin) as the optimization method to evaluate the methods. The evaluation is conducted under a hard search condition; only 100 individuals search 50-D/30-D functions whose search parameter ranges are expanded, and other uses 30 individuals.

3.3.3.2 Evaluations of the Proposal

Convergence characteristic of 34 benchmark functions is shown in Tables 3.6 and 3.7. In the tables, Wilcoxon signed-rank tests were applied between our proposed methods with canonical DE and previous acceleration methods [111], which conducts approximation in original search space. It is mentioned that, for the higher dimensional problems (F21-F34), previous acceleration methods require the # of population size must be $2D + 1 = 2 * 50 = 101$, which exceeds the experimental setting, so our proposed methods can be applied to such a higher dimensional problem rather than the previous methods. It shows the advantages of our proposed methods.

Proposed methods converged faster than the conventional methods at the same generation for both lower and higher dimensional problems except a few functions. This effect cannot be observed for F3, F5, F7, F8, F23, F28, F31, F32 and F34 functions because there were no significant difference between normal DE and DE with either of our proposed methods. Our proposed method's acceleration performance

Table 3.6: Average fitness value of F1-F20 benchmark functions. Bold font, †, ‡, §marks values mean IDE-LS or IDE-LR significantly better than canonical IDE, IDE-TB, IDE-TN and IDE-TA, respectively, by Wilcoxon signed-rank tests ($p < 0.05$).

Fun.	DE-N	DE-TB	DE-TN	DE-TA	DE-LS	DE-LR
F1	2.02E-06	9.40E-10	1.65E-11	0.00E+00	**2.97E-08**	4.28E-03
F2	7.12E-07	5.64E-07	2.66E-07	5.99E-07	**2.20E-07†‡§**	**3.81E-07†§**
F3	-3.00E+01	-3.00E+01	-2.99E+01	-3.00E+01	-2.99E+01	-3.00E+01‡
F4	1.33E+01	5.56E-01	1.38E+00	1.86E-01	**9.22E-01‡**	**1.67E+00**
F5	1.10E+00	1.04E+00	1.61E+00	1.10E+00	1.18E+00	1.58E+00
F6	6.85E+00	3.78E+00	3.53E+00	3.79E+00	**4.38E+00**	**4.92E+00**
F7	-1.97E+03	-1.98E+03	-1.94E+03	-1.96E+03	-1.96E+03	-1.95E+03
F8	3.60E-01	3.60E-01	3.60E-01	3.60E-01	3.60E-01	3.60E-01
F9	1.01E-02	8.45E-03	1.13E-02	8.19E-03	**9.10E-03**	**8.11E-03**
F10	7.78E-02	3.59E-03	6.27E-03	1.10E-02	**1.56E-02**	**9.25E-03§**
F11	6.26E+02	2.32E+01	2.54E+01	5.46E+00	**7.72E+00†‡**	**4.60E+01**
F12	3.25E+04	1.70E+03	5.20E+03	4.88E+02	**3.58E+03‡**	**8.51E+03**
F13	6.38E+00	2.70E+00	4.05E+00	1.25E+00	**3.85E+00**	**4.82E+00**
F14	1.93E+02	2.64E+01	4.69E+01	9.86E+00	**3.69E+01‡**	**6.72E+01**
F15	1.78E+01	8.57E+00	1.21E+01	5.46E+00	**1.00E+01‡**	**1.26E+01**
F16	3.49E-03	1.77E-03	4.16E-03	9.54E-04	**1.67E-03†‡**	**2.58E-03**
F17	2.82E-09	5.20E-13	2.00E-13	1.20E-11	**2.65E-11**	**8.15E-06**
F18	3.98E-01	3.98E-01	3.98E-01	3.98E-01	**3.98E-01‡**	3.98E-01
F19	3.00E+00	3.00E+00	3.00E+00	3.00E+00	**3.00E+00‡§**	**3.00E+00**
F20	-3.26E+00	-3.27E+00	-3.27E+00	-3.26E+00	**-3.27E+00‡**	**-3.26E+00**

looks similar, and if there is any difference, the superiority depends on the task being performed. From the Wilcoxon signed-rank tests comparison of our proposed methods with previous acceleration methods, the performance of our proposed methods is better than one of them in F2, F3, F4, F10, F11, F12, F14, F15, F16, F18, F19 and F20 in lower dimension problems. For higher dimensional benchmark functions (F21-F34), our proposed methods also show the capability to accelerate EC search except some cases, even to the problems with shifted and rotation characteristics. The DE-LS method's performance is better than that of DE-LR method in comparison with previous mentioned acceleration methods, i.e. DE-TB, DE-TN and DE-TA.

3.4 Discussions

3.4.1 Discussion on Acceleration Performance

With the exception of F3, F5 F7 and F8, our proposed methods can accelerate all lower dimensional benchmark functions. There were no cases where the proposed methods were significantly poorer than canonical DE. Although our proposed acceleration methods use a dimensionality reduction technique to obtain new elite in a lower dimensional search space, which would seem to imply more efficiency in

Table 3.7: Average fitness value of F21-F34 benchmark functions. Bold font values mean DE-LS or DE-LR significantly better than canonical DE by Wilcoxon signed-rank tests ($p < 0.05$).

Fun.	DE-N	DE-LS	DE-LR
F21	1.58E+05	**1.16E+05**	**1.39E+05**
F22	4.04E+05	**3.81E+05**	3.98E+05
F23	5.07E+09	4.97E+09	5.07E+09
F24	4.95E+05	**4.40E+05**	**4.51E+05**
F25	4.97E+04	**4.65E+04**	**4.91E+04**
F26	1.28E+11	**7.97E+10**	**1.07E+11**
F27	6.01E+03	**4.76E+03**	**5.84E+03**
F28	-1.19E+02	-1.19E+02	-1.19E+02
F29	5.69E+02	**4.78E+02**	**5.52E+02**
F30	1.10E+03	**9.07E+02**	**1.06E+03**
F31	1.68E+02	1.68E+02	1.68E+02
F32	6.86E+06	6.86E+06	6.84E+06
F33	2.62E+03	**1.28E+03**	**1.47E+03**
F34	-2.76E+02	-2.76E+02	-2.76E+02

problems with the characteristic of variable separability (i.e. F1, F4, F6, F10, F14 and F18), the experimental results show that DE-LR and DE-LS also exhibit better performance when the variables are non-separable (i.e. F2, F9, F11-F13, F15-F17, F19 and F20). For uni-modal and multimodal problems, the experimental results indicate that our proposed methods have the same capability to accelerate these two kinds of problems.

As the global optimum of F3 is on the edge of a search space, the elite obtained around the global optimum by function approximation may be located outside of the search range. Our experiment did not use the elite in this case, and DE using our proposed methods became identical to canonical DE. This would explain why there was no significant difference between DE-N, DE-LR and DE-LS.

F4 is a quartic function with Gaussian noise and has its global optimum in the center of the search space. Although it is a multimodal function due to the quartic function and fluctuations due to noise, the influence of the noise is relatively small and its whole shape is close to a quadratic function such as in F1. Elite from an approximated function using individuals that fluctuate slightly should be located in the center, and the performance of DE-LR and DE-LS should be better, as with F1, while that of canonical DE is negatively influenced by the fluctuations. This may explain the good performance of our proposed method with F4.

For higher dimensional problems (F21-F34), our proposed methods are also effective except F23, F28, F31, F32 and F34; most of them have complex fitness landscapes with a rotation characteristic. As rotated fitness landscape changes the real location of global optimum dramatically, our projected one-dimensional landscape can not use other dimensional search space information collaboratively. Obtained elite are not correct in each lower dimension. Maybe this is the reason why our proposed methods are ineffective in those benchmark tasks.

Except in special cases, the performance of DE-LR and DE-LS are better than that of DE-TN. This demonstrates that the original search space landscape fitted by those individuals with the lowest distance from the best individual is not accurate as a regression of the original search space; i.e. individuals near the best individual are not necessarily in regions of better fit in the original search space.

3.4.2 Discussion on Computational Complexity

Reducing computational cost is the greatest feature offered by our methods. This method of approximating landscapes and using the peak of an approximated function belongs to the same category as [111], but we extended the work by introducing projection onto a 1-D space, synthesizing an elite from the elites on multiple 1-D functions, and thus further reducing computational cost. Let's examine the efficiency of the methods by calculating the time cost in [80].

We calculated the run times of 50 trials of DE-N, DE-LR, DE-LS, DE-TB, DE-TN and DE-TA, and obtained the average running time cost of one computation. The system used to run the experiment was powered by a Core2 Duo 2GHz CPU with 1GB RAM running Windows XP (SP2).

The time cost table in [80] shows that the time cost of our proposed methods is more than canonical DE and less than the previous acceleration methods. The time cost of our proposed method is almost the same for a certain benchmark function. The three previous acceleration methods are costly for a certain benchmark function. It concludes that our proposed methods can save the time cost of optimization when compared to previous acceleration methods.

To further evaluate the significance of our proposed methods as compared with the previous methods, we conduct a time cost variance analysis with the time cost data sampled from our tests. Before calculating the F distribution value of the multiple groups' sampled time cost data for significance evaluation, we should check whether the sampled data fit a normal distribution in theory. For a large data set (data $>$ 30 samples), it must fit a normal distribution in accordance with the Central Limit Theorem. We use the Jarque-Bera test to check each time cost data set for goodness-of-fit to a normal distribution, and for abnormal data we smooth it for fitting to a normal distribution. We calculate the F value for each of the lower dimensional benchmark functions (F1-F20) in [80] with $r = 5$, $n = 250$. $F_{0.05}(r-1, n-r) = F_{0.05}(4, 245)$, i.e, $F_{0.05}(4, \infty) = 2.37$. From confidence interval table in [80], it shows that all the F values are greater than 2.37, indicating that our proposed methods time cost means are not equal to those of the previous methods, significantly.

From the above discussion, we can conclude that the proposed methods can significantly save time over previous methods. A comparison of DE-(LR and LS) with DE-(TB, TN and TA) shows that our proposed acceleration methods seem more beneficial than previous ones. However, before the regression and fitting process, it is necessary to select the sampled data. In the selection process, there are more searches and sorting operations, and these take as much as or more time than the regression and fitting processing. This shows how important an issue the sample data selection is when accelerating EC by obtaining the landscape of the search space. Improving the performance of data selection will a subject of our future research.

3.4.3 Regression Function Selection for Fitness Landscape Approximation

Approximation function selection is a critical issue in the approximation of the IEC/EC landscape. From a practical viewpoint, we should select an appropriate approximation function after analyzing the characteristic of the IEC/EC landscape. In this study, we select binomial Lagrange interpolation and linear function least squares approximation as the regression methods for our study.

How to obtain the regression IEC/EC search space is a promising research direction. In our study, we try to use the nonlinear and linear curve expressions to approximate the IEC/EC search space, to obtain the approximation relationship between the approximation and the original IEC/EC search space, and to find the global optimum from the new elite by chance. However, the approximation process influences the performance of the IEC/EC algorithms, so using efficient approximation method is another topic to be considered when designing accelerated IEC/EC algorithms. We should use the lowest possible degree of interpolation and approximation functions in the lowest possible dimensional spaces to reduce the time cost required for the entire IEC/EC algorithm.

3.4.4 Regression Model Establishment and Usage

The experimental results in this study show that both linear (DE-LS) models and nonlinear (DE-LR) models can be used to efficiently accelerate IEC/EC convergence. However, the methods are mostly effective in the initial generations in most of the cases by sign tests, and an alternative solution is to use this kind of acceleration method to speed IEC/EC convergence only in the early generations.

Constructing and selecting different models to better approximate the IEC/EC search space is a worthwhile topic for further future research. To achieve optimal performance, we should choose approximation methods that are suited to the characteristics of the search space. For regression in the discrete domain, a more powerful tool than continuous domain Lagrange interpolation and least squares approximation is required. It is a valuable research topic that to find how best to approximate the search space in the discrete domain.

Another point deserving special attention is the method used for dimensionality reduction. The objective of the dimensionality reduction is to simplify the computation of the search space regression by performing it in a lower dimension. The dimensional reduction loses much search space information, however, such as the relationship between the parameters in non-separable problems. How we can reduce the search space dimension as much as possible while simultaneously preserving the necessary search space information for the next generation of the search is a challenging problem for further research into the dimensionality reduction method.

The new elite selection method is also a key step in the acceleration process. The elite selection method is determined by the different search space regression methods. They can be categorized as linear and nonlinear models, corresponding to the the classification method of the regression model. A new elite selection method should be decided by analyzing the characteristics of the regression model.

3.4.5 Data Sampling Technique

It is necessary to sample the data to obtain the original search space information that is used in the approximation processes. The data sampling technique determines the quality of the approximation model, indirectly influencing the model usage and obtained elite. Several data sampling methods have been proposed [82]. These include best data strategy, distance nearest data strategy, whole data strategy and random data strategy. For comparison in those sample studies, we discuss the advantages and disadvantages of the methods as follows.

First, obtaining suitable data for the approximation of the IEC/EC landscape requires time in searching and sorting operations. With a view to maximizing performance and saving time, we need to select an easy way to obtain the sampled data. Secondly, from the experimental results, distance nearest data strategy is more costly than any of the other sampling data strategies. The acceleration performance of the approximation model combined with distance nearest data is not as good as the others. So with a view to practical application, the distance nearest data strategy exhibits low performance. Third, our proposed acceleration methods that use the best data strategy to approximate the model in the lower search space perform better than the previous acceleration methods in some cases. However, if we select a data sampling strategy that is even more time efficient and does not require searching and sorting, such as the random data strategy, the performance of our proposed methods may be further improved. This is a direction for further research.

3.5 Chapter Summary

We proposed to approximate fitness landscape to accelerate IEC/EC convergence by a dimensionality reduction technique. The novel feature in these acceleration methods was to use elite synthesized from elite points found in lower dimensional spaces. The main contribution of this study was utilizing the notion of dimensionality reduction to approximate fitness landscape for accelerating IEC/EC searches. Our experimental evaluations with a Gaussian mixture model and 34 benchmark functions showed that the proposed methods can accelerate IEC and EC searches with lower computational cost. The acceleration performance can be obtained in both lower and higher dimension problems, especially when the landscape of the tasks takes on a roughly big valley structure. We also analyzed the relationship between the performance of the proposed methods and the landscape shapes.

Chapter 4

Fourier Analysis on Fitness Landscape for Evolutionary Search Acceleration

4.1 Introduction

The Fourier transform is a powerful mathematical tool for analyzing the frequency information of signals using orthogonal trigonometric functions and has been used in signal processing [92], bioinformatics [117], and many other areas. The Fourier transform can also be used in EC as a mathematical tool for studying an EC model, resolving the complexity of a fitness landscape and accelerating EC convergence. It was used to obtain the model parameters of polynomial harmonic models and used for genetic programming in [70]. Taylor series and Fourier series were respectively used for local and global approaches to analyzing the fitness landscape in [119]. EC was also used as an optimization tool to research time-frequency analysis in signal processing [18]. However, little research literature has reported on EC fitness landscape analysis and EC algorithm design where convergence is accelerated using the Fourier transform.

The objective of this work is to propose a method that analyzes a fitness landscape by considering points on it as signal samples, obtaining the frequency characteristics of same, approximating the fitness landscape by filtering its primary frequency component, and finally accelerating the EC search by obtaining elite from the approximated landscape. Concretely speaking, the method re-samples a fitness landscape at uniform intervals, calculates the corresponding fitness values, applies a discrete Fourier transform (DFT) to them, filters only the primary frequency component(s), applies the inverse DFT to the filtered frequency components to obtain an approximate fitness landscape, obtains rough location information for the global optimum from the approximated landscape, and subsequently uses it to accelerate the EC search.

The key points of this proposed method are finding the primary frequency component(s) and filtering it (them) to approximate the original fitness landscape with one or more trigonometric function. We also propose an approach for global approximation, which approximates the whole landscape, and for local approximation, which estimates the landscape around the best individual. Both sampling methods obtain the elite from the approximated surface and use it to accelerate the EC search.

We use a DFT to analyze a fitness landscape on which we can then define primary frequency components. The primary frequency components are used to construct an approximate model for the EC acceleration in section 4.2. We also propose two resampling methods in section 4.2 to obtain corresponding frequency components; one approximates the whole fitness landscape and the other a local area in the landscape. We evaluate the proposed methods by applying differential evolution (DE/best/1/bin) to eight benchmark functions in section 4.3 and section 4.4. We analyze the proposed methods and discuss their future possibilities in section 4.5. Finally, we conclude this research and describe our future research directions in section 4.6.

4.2 Fourier Analysis of a Fitness Landscape

4.2.1 Concept of the Proposed Method

EC search is based on the fitness of the individuals. As we do not use information about the EC search surface but rather the fitness of a limited numbers of individuals distributed in the search space, less obtained search information is a restriction on our ability to extending the EC search capability. In a complex search space, the limited number of fitness values cannot adequately express the search space characteristic and direct the EC interaction process.

If we can obtain EC fitness landscape information during EC search, EC search performance and applicability can be improved and extended. Obtaining EC fitness landscape information is a promising area for research [77]. Jin has investigated fitness landscape approximation approaches and basis evolution management strategies [38].

Fig. 4.1 shows the flow diagram of our proposed method. Frequency characteristics of a fitness landscape are obtained by resampling a search space at regular intervals and applying the DFT to a sequence of fitness values for the resampled points. We can approximate the original fitness landscape with a trigonometric function by filtering a primary frequency component and applying inverse DFT to the filtered frequency components. Our proposed method that aims to accelerate EC search using information of an approximated function obtained by Fourier transform can be considered as a regression model for the trigonometric functions Eq. (4.1).

$$EC(X) = \sum_{i=0}^{N} a_i \sin(2\pi\omega_i X + B_i) \qquad (4.1)$$

4.2.2 Fourier Transform of a Fitness Landscape and Approximation of the Landscape Using Trigonometric Functions

Although individuals generated by EC operations are distributed at irregular intervals within the search space, samples used for a DFT must be at uniformly sampled.

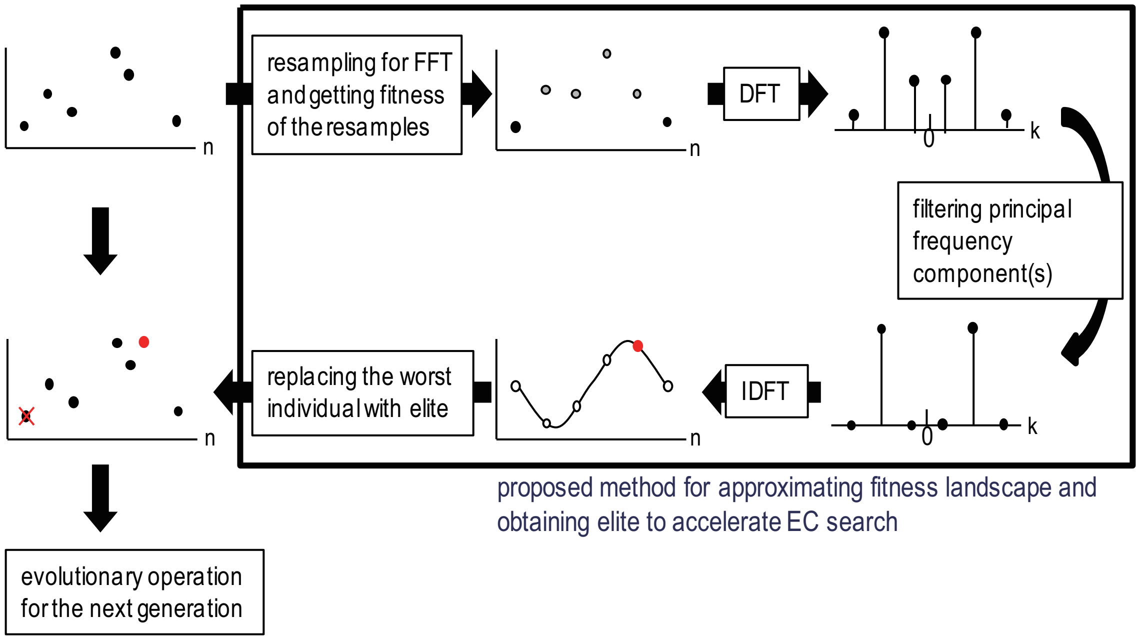

n
resampling for FFT and getting fitness of the resamples
n
DFT
0
k
filtering principal frequency component(s)
k
0
IDFT
n
replacing the worst individual with elite
n
evolutionary operation for the next generation
proposed method for approximating fitness landscape and obtaining elite to accelerate EC search

Figure 4.1: Flow diagram of our proposed method. The search space is resampled at uniform intervals. Fitness values are calculated and their frequency characteristics are calculated by DFT. The primary frequency component is filtered, and an original fitness landscape is approximated with a trigonometric function by applying the inverse DFT to the filtered frequency components. The elite is obtained from the approximated landscape and used to accelerate EC search by substituting it for the worst individual.

The search space must therefore be resampled, and then their fitness values must be recalculated.

In obtaining the frequency characteristics of a fitness landscape, there are two issues we must first decide: (1) the number of resampling points, (2) the sampling period and DFT base frequency points, i.e. the interval between sample points, which decides frequency resolution.

First, we cannot perform a DFT without deciding on the number of sample points (M). The more sample data we use, the higher the frequency resolution we can obtain. However, in deciding upon the number of sample points, we must take into account the balance between computational cost and convergence speed under the conditions of the application task because fitness values must be recalculated for the resampled data and the computational cost ($M \log 2M$) of the Fast Fourier Transform (FFT) depends on the number of sample points (M). In our experiment in this chapter, we set $M = 16$ and applied the FFT to a sequence of these 16 fitness values.

Regarding sampling dimension, there are two possible resampling methods we can use. In one method we resample the search space separately for each dimension, and in the other we resample the whole n-D search space (n-D sampling) at once. Additionally, we have two choices regarding where we choose to resample; in one method we resample the whole search space (global sampling) and in the other we only resample the area around the best individual (local sampling). As a preliminary experiment, we evaluate how these resampling methods influence the approximation of the fitness landscape in the next section.

The second issue, sampling period, depends on the # of sampling points and the sampling area (area to be approximated) which is determined by the the first issue above. Sampling over the entire search range is one approach, and sampling over a range of distributed individuals and narrowing the range according to the narrowing range of its distribution is another. A third technique would limit the sampling area.

Because both global and local properties are valuable to the EC search, we design two resampling methods. In the first method, the entire search space is uniformly sampled in each dimension to obtain resampled data. We call this GLB. In the second, the resample area is centered on the area around the best individual. We call this LOC. These are respectively our global and local sampling approaches.

The third issue to resolve is in which dimension we should approximate the fitness landscape. If each dimension is being resampled separately, there are an additional two options that must be considered. If it is sufficient to resample in just one dimension, we can resample the 1-D space with a uniform sampling interval and then use parameter values for the best individuals in the other corresponding dimensions.

If we are to obtain the full multi-dimensional landscape, we use GLB or LOC. In this case there are a total of 4 ways the sampling methods can be combined ($4 = 2 \times 2$). These combinations are also evaluated below.

4.2.3 Filtering in the Frequency Domain

The trigonometric functions that determine the main structure of a fitness landscape are the most important in the regression model of Eq. (4.1), and they are determined by the amplitude and phase information at the peaks in the power spectrum. Let us define them as *principal frequency components* and call them the principal frequency component, the second, and so on according to their maximum power ranking.

Although the original fitness landscape is not always periodical with 2π , we can universally approximate the local landscape with arbitrary accuracy if we use an arbitrary number of trigonometric functions. In practice, however, too many base functions introduce too much computational complexity, so in this chapter we will approximate the original fitness landscape with just use one trigonometric function, i.e. we use only the first principal frequency component.

4.2.4 Approximation of a Fitness Landscape by Inverse DFT and Acceleration of the EC Search

Once a concrete regression model of trigonometric functions is obtained by applying the inverse DFT to the frequency characteristics containing only the filtered principal frequency components, the regression model is used as an approximation of the local or global EC fitness landscape in 1-D dimension or n-D dimensions. With this approximated landscape characteristic, we can obtain more EC search information than that which is directly provided by the individuals who's population size is limited.

There are several approaches for using the obtained landscape approximation information to accelerate EC search. For example, in one approach we can determine the EC search direction by analyzing the approximated fitness landscape and in another approach can obtain new elites from the approximated landscape and use them for the EC search in the next generation [36]. For optimization problems which involve finding the maximum point, we expect that the global point should be around the points of $k(1/4\omega) + b$ $(x > 0)$ and $k(3/4\omega) + b$ $(x < 0)$; whereas for optimization problems which involve finding the minimum point, we expect that the global point should be around the points of $k(3/4\omega) + b$ $(x > 0)$ and $k(1/4\omega) + b$ $(x < 0)$. We may be able to accelerate the EC search by using the global point as the elite for the next generation. So if we can locate the individuals with this characteristic in the corresponding original search space and put them into the next generation, it should be possible to accelerate the EC. We use this acceleration method using elites obtained from an approximated function in our experimental evaluation in the next section.

4.3 EC Acceleration Experimental Evaluation

4.3.1 Experimental Conditions

We use the De Jong five standard functions (F1 - F5) , Rastrigin function (F6), Schwefel function (F7) and Griewank function (F8) as benchmark functions to evaluate the proposed approaches. The dimensions and search ranges for all parameters are listed in Table 4.1. All these function optimization tasks are posed as minimization problems with the optimal solution being the point with the lowest value. Their fitness landscapes have a variety of characteristics. They include both continuous and discontinuous, convex and non-convex, unimodal and multimodal, and low dimensional, variable separable and non-separable, and high dimensional shapes.

Table 4.1: Benchmark functions used in EC experimental evaluations, where Range, n, and C refer respectively to the ranges of the parameter values, the dimension of the function, and its characteristics. M, U, N, and S refer respectively to multimodal, unimodal, non-separable (non additional) and separable (additional).

No.	Name	Test function	Range	n	C
F1	Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$	[-5.12,5.12]	3	US
F2	Rosenbrock	$f(x) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2$	[-2.048,2.048]	2	UN
F3	DeJong-Step	$f(x) = \sum_{i=1}^{n} \lfloor x_i \rfloor$	[-5.12,5.12]	5	US
F4	Quantic & Noise	$f(x) = \sum_{i=1}^{n} ix_i^4 + Gauss(0,1)$	[-1.28,1.28]	30	US
F5	Shekel's Foxholes	$f(x) = [0.02 + \sum_{j=1}^{25} \frac{1}{j+\sum_{i=1}^{2}(x_i - a_{ij})^6}]^{-1}$	[-65.536,65.536]	2	MS
F6	Rastrigin	$f(x) = (10n) + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$	[-5.12,5.12]	5	MS
F7	Schwefel 2.26	$f(x) = \sum_{i=1}^{n}(-x_i \sin(\sqrt{\lvert x_i \rvert}))$	[-512,512]	5	MS
F8	Griewank	$f(x) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}})$	[-512,512]	5	MN

We compare the EC acceleration method proposed in this chapter with our previously proposed acceleration method [74] and normal non-accelerated EC as

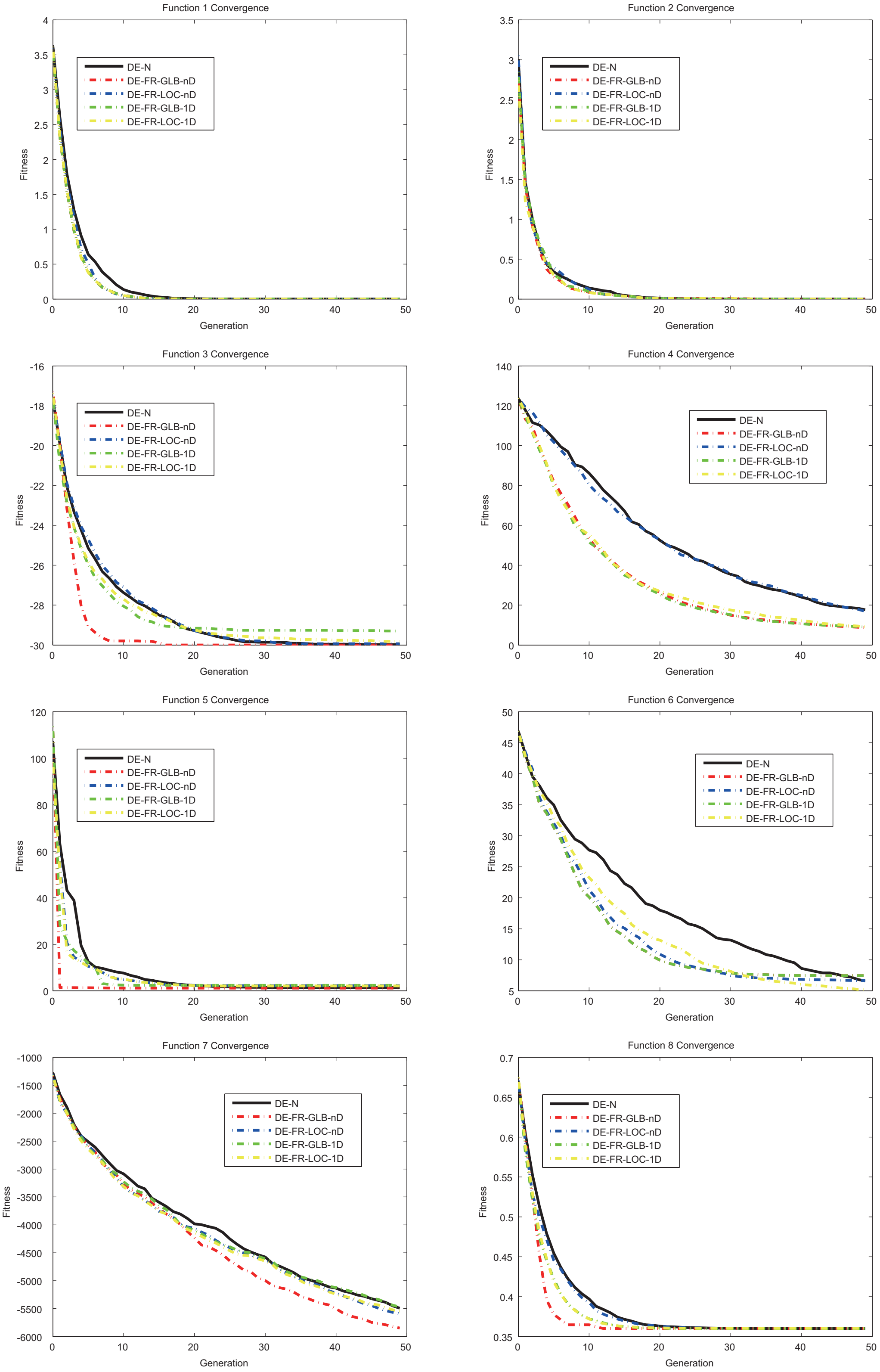


Figure 4.2: Average convergence curves of 50 trial runs. See F1 - F8 in Table 4.1.

references, using the same benchmark functions.

We use differential evolution (DE/best/1/bin) as our optimization method to compare the proposed approach with the reference conventional ones. The approaches are applied to 8 benchmark functions for up to 50 generations with 50 trial runs, and a sign test is used to determine if there a significant difference.

Here, we abbreviate the four variations of our DFT-based methods proposed in this chapter as DE-FR-GLB-1D, DE-FR-GLB-nD, DE-FR-LOC-1D, and DE-FR-LOC-nD. These refer, respectively, to DE with elite obtained by global sampling in 1-D, global sampling in n-D, local sampling in 1-D, and local sampling in n-D. We use the same abbreviations in the other EC acceleration approaches, i.e. DE-LR and DE-LS mean DE with elite obtained by a two-degree Lagrange interpolation and a line power function least squares approximation. Finally, canonical DE without any acceleration method is referred to as DE-N. These abbreviations are also used in Figures 4.2 and 4.3.

Figure 4.2 shows the average convergence curves for the best fitness values over 50 trial runs of these methods, and Figure 4.3 shows the sign test results between DE-N and DE with each acceleration method at each generation.

4.3.2 Experimental Results

The following observations can be made from these experimental results.

1. Our proposed methods were able to accelerate the EC well for all benchmark functions except F2.
2. The proposed methods did not accelerate DE convergence well for F2.
3. The performances of the four proposed methods look similar and their relative superiority depends on the task if there even is a difference between their performances.
4. The proposed method, DE-FR-GLB-nD, demonstrated better performance than all the other proposed methods, i.e. global sampling is the best method for obtaining the whole fitness landscape characteristic and efficiently finding the global optimum.
5. Our proposals in this chapter have shown better performance for F3, F5, F6, F7, F8 than the acceleration approaches which we previously outlined in [74].

4.4 Comparative Evaluation of EC Acceleration by 1-D FFT and n-D FFT

4.4.1 Experimental Conditions

We use the same benchmark function described in Table 4.1. All these function optimization tasks are posed as minimization problems with the optimal solution being the point with the lowest value. For comparing the performance with the previous acceleration methods, we use the EC acceleration approaches of works [74, 79, 111] to test those benchmark functions and make a comparative evaluation.

Differential evolutionary (DE/best/1/bin) is as well as used as an optimization method to evaluate the proposed approaches. We test each benchmark function

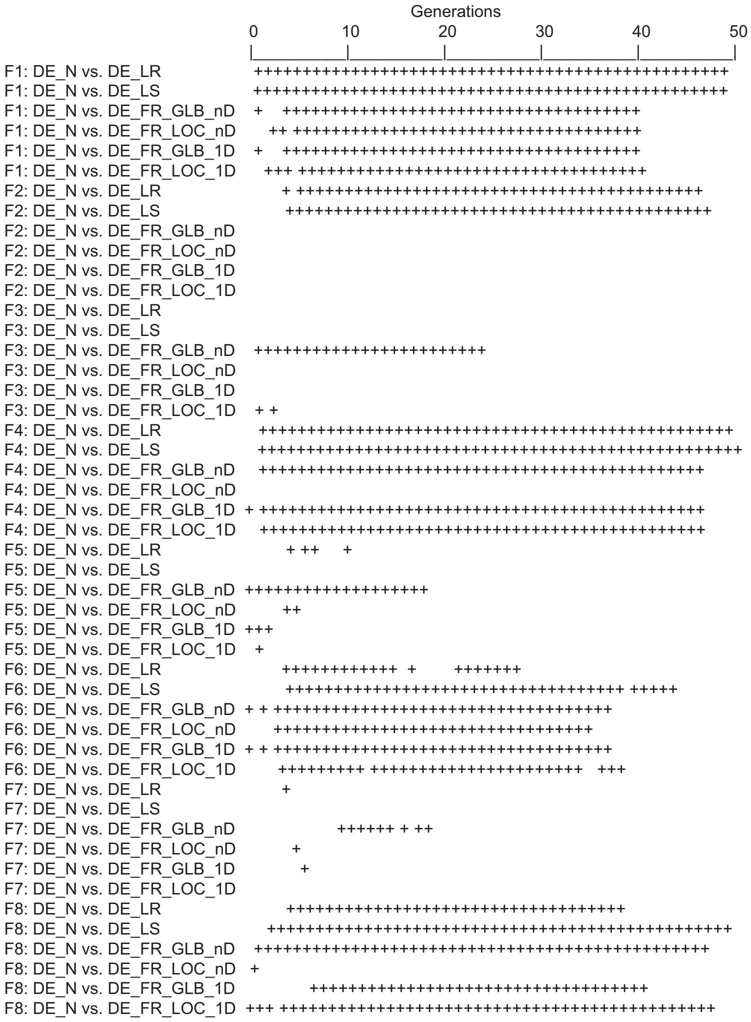


Figure 4.3: Sign test results for 50 trial runs of (DE-N vs. DE-LR), (DE-N vs. DE-LS), (DE-N vs. DE-FR-GLB-1D), (DE-N vs. DE-FR-GLB-nD), (DE-N vs. DE-FR-LOC-1D), and (DE-N vs. DE-FR-LOC-nD) per generation. See F1 - F8 in Figure 4.2. The $(+, -)$ marks show whether our proposed methods converge significantly better or poorer than canonical DE, respectively, ($p \leq 0.05$).

up to 100 generations with 50 trial runs. Figure 4.4 shows the average convergence curves of the best fitness values of 50 trial runs for all eight benchmark functions, and Figures 4.5 and 4.6 show their sign test results at each generation. Abbreviations used in Figures 4.4, 4.5 and 4.6 are given in Table 4.2.

4.4.2 Experimental Results

Accordance with those test results, we can conclude that:

1. Our proposed n-D DFT approximation approach can accelerate all eight benchmark functions, except F3, F4, and F5.
2. Our proposed n-D DFT approximation approach has the better acceleration performance than that of 1-D DFT approximation approach, except F3, F4,

Table 4.2: Abbreviations used in Figures 4.4, 4.5 and 4.6.

abbreviations	DE whose fitness landscape is regressed by
DE-N	no any method (canonical DE)
DE-F1D, DE-F2D	1-D or 2-D FFT approach
DE-LR	a two-degree Lagrange interpolation
DE-LS	a line power function least squares approximation
DE-TB	a two-degree power function least squares approximation in original search space with the best sampling [74, 75]

and F5.

3. N-D DFT approximation approach can accelerate F7's search, but 1-D DFT approximation approach cannot.

4.5 Discussions

4.5.1 EC Acceleration Performance

We obtained information about a fitness landscape using four proposed methods (DE-FR-GLB-nD, DE-FR-GLB-1D, DE-FR-LOC-nD or DE-FR-LOC-1D) and approximated the landscape by filtering its principal frequency component. From the comparison between global and local sampling methods, we can say that DE-FR-GLB-nD and DE-FR-GLB-1D are suitable for global exploration over a whole search space and DE-FR-LOC-nD and DE-FR-LOC-1D are suitable for local exploration in the local search space near the best individual, because different sampling methods obtain different frequency and phase information from the global/local and 1 dimension/ n dimensions.

From the experimental results, the global sampling methods, DE-FR-GLB-nD and DE-FR-GLB-1D, demonstrated better acceleration performance, i.e obtained a better final solution. This is because they can obtain more accurate frequency information about a given fitness landscape than the other methods. Even though the current best individuals are in local optimum areas, we can acquire information about the global optimum from DE-FR-GLB-nD and DE-FR-GLB-1D.

From average convergence curves and sign test results of F1, F4, F6, F7, and F8, the local sampling methods, DE-FR-LOC-nD and DE-FR-LOC-1D, are useful for problems in which the fitness landscape has local valleys.

The shape of the global optimum area of F2 is a long, narrow, parabolic, and flat valley. As each of its dimensions has this similar shape, our proposed methods are unavailable for problems with this kind of shape. Although this valley shape looks trivial, convergence to the global optimum is difficult; our proposed methods seem inefficient and leave room to be improved.

F4 is a function with Gaussian noise. As noise made by a Gaussian random number is white Gaussian noise, its spectrum is flat and therefore does not influence

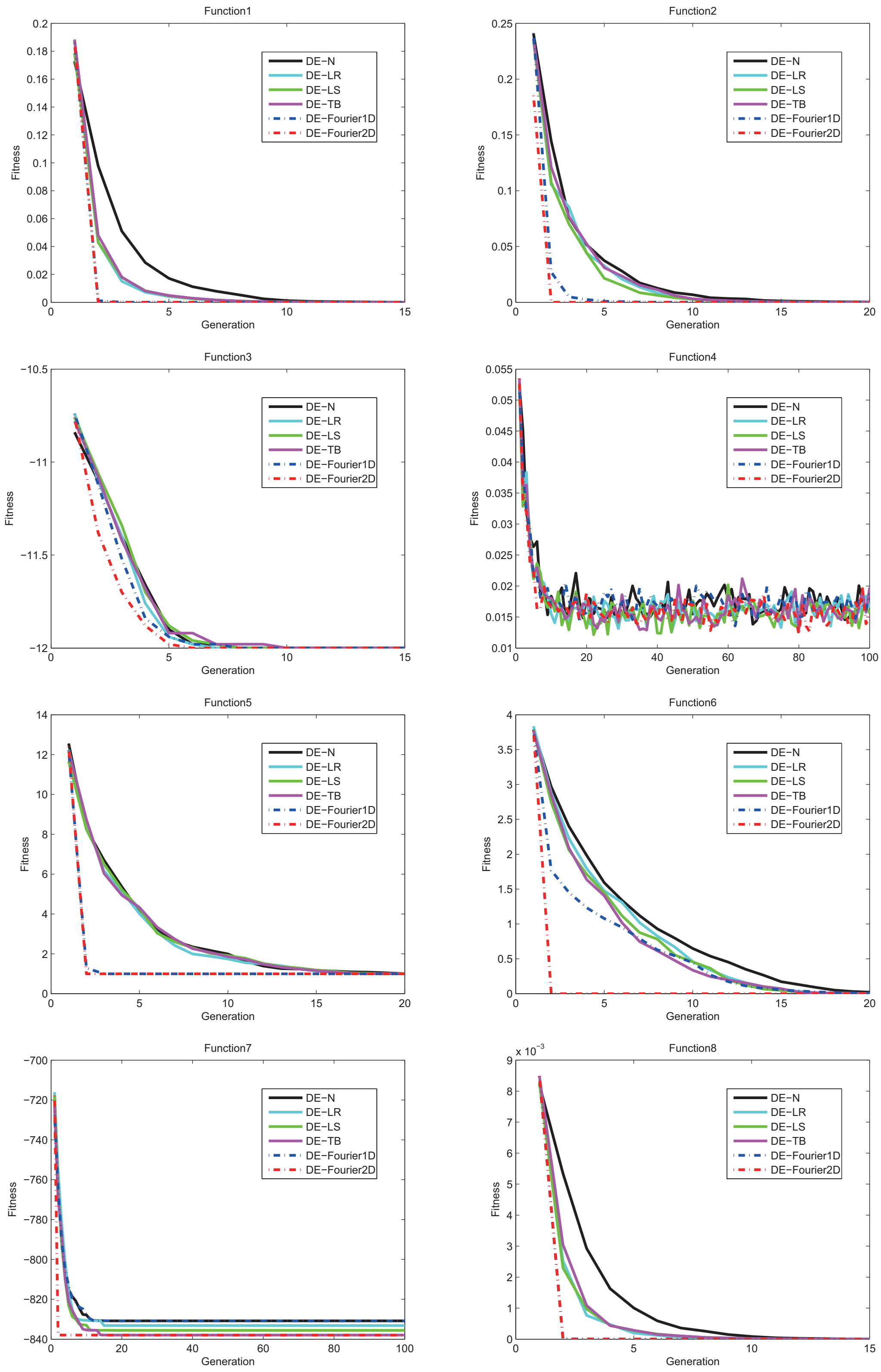


Figure 4.4: Average convergence curves of 50 trial runs for F1-F8.

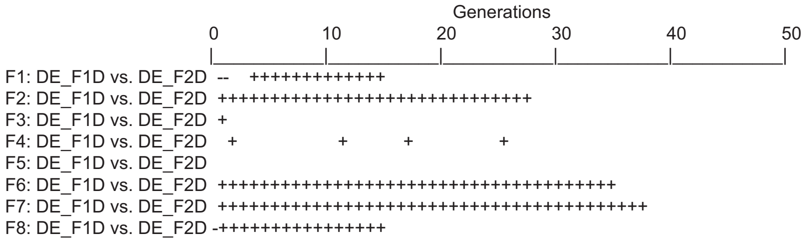


Figure 4.5: Sign test results for 50 trial runs of (DE-F1D vs. DE-F2D) per generation. The + and − marks mean that A is significantly better and poorer than B, respectively, for "A vs. B" ($p < 0.05$).

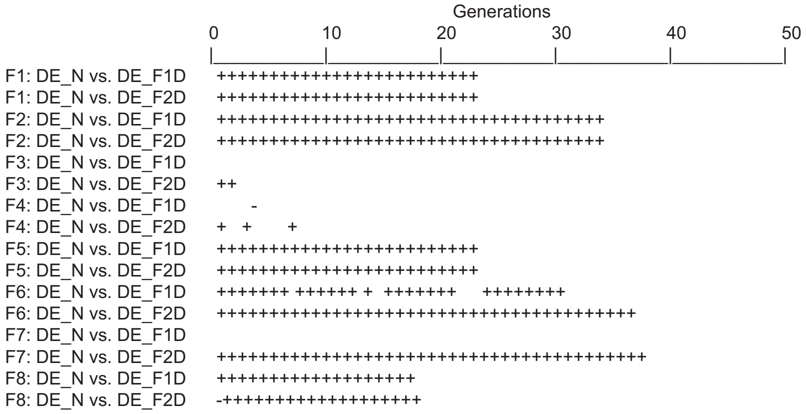


Figure 4.6: Sign test results for 50 trial runs of (DE-N vs. DE-F1D), (DE-N vs. DE-F2D) per generation. The + and − marks' meaning as in Figure 4.5.

the principal frequency components. Consequently, our proposed method can reduce the effect of the noise significantly. Our experimental results indicate that our proposed methods can significantly accelerate F4 in most generations. Local areas of F4 are not approximated well by a cosine curve and significant performance improvement was not obtained for F4 when DE-FR-LOC-nD was used.

For multimodal problems, such as F5, our methods can accelerate the convergence in the initial generations. However, when individuals migrate towards a single local optimum and their diversity is decreased, the performance of the elite acceleration drops. When individuals gather at one point over the generations, the frequency spectrum obtained from the area covered by the individuals no longer has enough information to approximate the fitness landscape.

The objective of this work is to reduce the complexity of a fitness landscape by obtaining an approximation of it from the frequency characteristics of the fitness landscape. Simplifying a fitness landscape by reducing the noise that makes a fitness landscape complex realizes this objective, and the subsequent global and local exploration is the biggest contribution of our proposed method for accelerating EC convergence. Experimental results have shown that our proposed methods can accelerate EC.

4.5.2 Acceleration Performance by 1-D and n-D FFT

Our proposed n-D DFT approach can obtain the better acceleration performance than 1-D DFT approach from the evaluation results, except F3, F4, and F5, which the global optima locates in bound. N-D DFT approach can obtain more frequency information and transfer to relative full-scale fitness landscape thanks to conducting 1-n DFT in each dimensional several time. However, the 1-D DFT approach just conduct 1-D DFT with certain point one time in each dimension. This is the reason why the acceleration performance by n-D DFT approach better than 1-D DFT approach.

Although n-D DFT approximation approach can obtain relatively enough frequency information than that obtained by 1-D DFT approximation approach, its performance depends on the location of sampling points for n-D DFT conducts. If each location is around the global optimum, n-D DFT approximation approach can obtain the better acceleration performance, but when they are far from the global optima, its performance may not be great. The acceleration performance obtained by n-D DFT approximation approach also depends on base frequency setting and sampling data number.

Theoretically, the computational complexities of n-D FFT and 1-D FFT are $(\Pi m_i)N\log N$ and $nN\log N$, respectively. It means that n-D FFT computational complexity is $(\Pi m_i)/n$ times of that of 1-D FFT; n-D FFT approximation approach is costly approach. For practical applications, if the 1-D FFT approximation approach has the relatively acceptable acceleration performance, which is similar to that of n-D FFT approximation approach, we should consider to use 1-D FFT approximation approach, of course, it depends on benchmark tasks and real world applications.

4.6 Chapter Summary

We proposed a family of landscape approximation methods that analyze the frequency characteristics of a fitness landscape using a DFT, obtain principal frequency components, filter the components, and apply the inverse DFT to the filtered frequency characteristics. Our methods accelerate EC by substituting an elite obtained from the approximated landscape for the worst EC individual. We evaluated the methods with eight benchmark functions and demonstrated that this strategy of approximating fitness landscapes is effective for accelerating EC search. We also analyzed the performance of our proposed methods and discussed them in detail.

It is difficult to applied this acceleration method in a real IEC application, because the real human user would give fitness when the re-sampling the data in a search space. This process increases the human fatigue even though it can obtain a better final result. So using an established human model or other method should be investigated for this method in a real IEC application. We will study this topic in the future.

The original contribution of this work is to direct our attention to the frequency characteristics of fitness landscapes, to use this observation to simplify the landscapes, and to apply our simplification so that we can accelerate EC search.

Chapter 5

Fourier Niche Method for Multimodal Optimization

5.1 Introduction

There are several reasons that we need to obtain not only a global optimum but also some local ones in real world EC applications. On the one hand, the global optimum solution may be difficult to be implemented in the actual system for some EC applications. On the other hand, we need to obtain a variety of optimal solutions, so the alternative solutions should be found in the EC results. This is the original motivation of solving multimodal optimization problems, which have multiple modal in their search space including global and local optima.

The conventional methods for solving multimodal optimization problems are point-to-point methods. They find one optimum in one time and conduct the same operation several times to obtain all the solutions. EC has a niche in its population to solve multimodal optimization problems. Niche methods conduct an optimization once to find all the solutions rather than applying the conventional methods with multiple times.

All the existing niche methods focus on either fitness tuning or restricted operations. Most of them ignore to consider a fitness landscape in their search algorithms. If the number of multimodal peaks or local region information can be obtained, optimization performance of niche methods can be enhanced using these information.

In this chapter, we propose a Fourier niche method that uses frequency information of a fitness landscape. It can support information about potential multimodal regions and their number. After we obtain the frequency, phase and amplitude information of a fitness landscape using fast Fourier transform (FFT), approximate multimodal regions and the number of the local peaks are calculated. We can use this information to implement a novel niche method - Fourier niche method.

This chapter is structured as follows. An overview of niche method is given in section 5.2. The fundamental of the Fourier niche method is introduced in section 5.3. In section 5.4, we use differential evolution (DE/best/1/bin) as an evaluation tool to analyze the proposed algorithms' performance with six multimodal benchmark functions. In section 5.5, we discuss the proposed algorithm in detail. An outlook on open topics and potential opportunities are presented. Finally, we present the further research direction and conclusion (section 5.6).

5.2 State-of-the-Art on Niche Method

EC has capability to find a global optimum quickly and easily, but lacks the capability to find all peaks of multimodal optimization problems, i.e., global and local optima. Several so-called niche preserving techniques therefore were proposed.

The breaking work of the multimodal optimization problems was fitness sharing method [21]. Its mechanism is shown in Eq. (5.1), where $F_{shared}(i)$ is a shared fitness and $F_{original}(i)$ is a original fitness. $Sh(d_{ij})$ presents a sharing function, which is defined in Eq. (5.2), where d_{ij} is a distance between two individuals, σ_{share} is a constant parameter which regulates the shape of the sharing function.

$$F_{shared}(i) = \frac{F_{original}(i)}{\sum_{j=i}^{N} SH(d_{ij})} \tag{5.1}$$

$$SH(d_{ij}) = \begin{cases} 1 - (\frac{d_{ij}}{\sigma_{share}})^{\alpha} & \text{if } d_{ij} > \sigma_{share} \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

A clustering method divides a population into niches, and fitness calculations are based on the distance between an individual and niche centroid (i.e., the centroid point of niches) [123]. The concept of crowding was originally proposed in [41], which was motivated by phenomena on competition for limited natural resource among the species. Two crowing techniques, deterministic crowing and probabilistic crowing, were proposed by replacement with restriction in [57, 63]. Restricted tournament selection for multimodal optimization was introduced in [28]. A clearing procedure simulates a process of sharing the limited resource by characterized individuals [85]. Species conserving GA was proposed to solve multimodal optimization problems in [52].

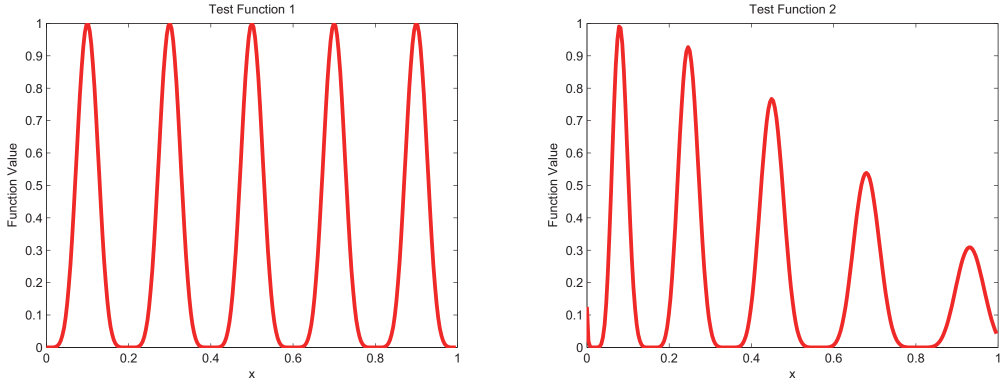
5.3 Fourier Niche Method

5.3.1 Obtaining Frequency Information

In order to obtain frequency information, we should re-sample the data in a search space with equal interval and calculate their fitness value. Some data sampling method were proposed in [79]. In this study we choose the global 1 dimension sampling method in our experimental part. One of FFT method, i.e. decimation in frequency algorithm, is shown in Eq. (5.3), where W_N^{kn} is $W_N^{kn} = e^{-\frac{2k\pi}{N}}$, and $0 \leq k \leq N-1$. The terms $x(n)$, $X(k)$, n and N are the original signal series, frequency signals by DFT in frequency space, the number of sample points of original signals and the number of transform base frequency, respectively.

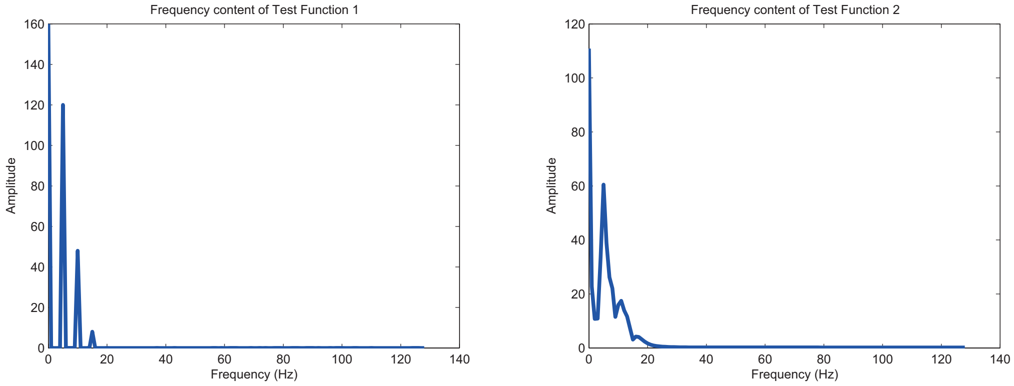
$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} [x(n) + (-1)^k x(n + \frac{N}{2})] W_N^{nk} \tag{5.3}$$

Figure 5.1 shows two multimodal benchmark functions, F1 and F2, which expressions are shown in Table 5.1. F1 has five global and local optima located equally, and F2 has those with different amplitude located unequally. Their power spectrum are shown in Figure 5.2.



(a) (b)

Figure 5.1: Power spectrum of F1 and F2.



(a) (b)

Figure 5.2: Frequency content of F1 and F2 by FFT.

5.3.2 Calculating Peak Range and Number

After we obtain the frequency (F_i), phase (b_i) and amplitude (A_i) information in each dimension, principal frequency components [79] with relative larger amplitude are filtered. We use these principal frequency components to calculate peak regions and their number of original fitness landscape.

Inverse FFT of the principal frequency components can approximate the original fitness landscape as shown in Eq. (5.4). In our proposed method, we only use the powerful amplitude as the approximation model, so it can be expressed as $a_pcos(\omega_px) + c_psin(\omega_px)$, i.e. $\sqrt{a_p^2 + c_p^2}cos(\omega_px + b_p)$, and $b_p = arctan(\frac{a_p}{c_p})$. For the cos($x$) function, the minimum should local at $(\frac{T_i}{2}) + b_i$, $T_i = \frac{1}{F_i}$ ($F_i = \frac{\omega_p}{2*\pi}$), so in each dimension, the peak appears in locations within the search range around points $k_i(\frac{1}{2*F_i}) + b_i$. $k_i = 1, 2, ..., n$ are the number of peaks in the ith dimension calculated by Eq. (5.5). The total number of peaks in a search space is calculated by Eq. (5.6). In experimental part, we use a maximum value of the total number of peaks.

Table 5.1: Benchmark functions used in experimental evaluations. Range is a range of parameter, n is dimension of the benchmark function. C means function's characters. M, U, N, and S mean multimodal, uni-modal, non-separable, and separable, respectively.

No.	Name	Test function	Range	n	C
F1	Niching1	$f(x) = sin^6(5\pi x)$	[0,1]	1	MN
F2	Niching2	$f(x) = e^{-2(ln2)(\frac{x-0.01}{0.8})^2} sin^6(5\pi(x^{0.75} - 0.05))$	[0,1]	1	MN
F3	Shekel's Foxholes	$f(x) = [0.02 + \sum_{j=1}^{25} \frac{1}{j+\sum_{i=1}^{2}(x_i - a_{ij})^6}]^{-1}$	[-65.536,65.536]	2	MS
F4	Rastrigin	$f(x) = (10n) + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$	[-5.12,5.12]	2	MS
F5	Schwefel 2.26	$f(x) = \sum_{i=1}^{n}(-x_i \sin(\sqrt{\lvert x_i \rvert}))$	[-512,512]	2	MS
F6	Griewank	$f(x) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}})$	[-20,20]	2	MN

$$f(x) = a_0 + \sum_{i=1}^{n}(a_i cos(\omega_i x) + c_i sin(\omega_i x) + b_i) \tag{5.4}$$

$$k_i = Interity \llcorner \frac{Range_i}{T_i} \lrcorner \tag{5.5}$$

$$\sum_{i=1}^{n} k_i < N < \prod_{i=1}^{n} k_i \tag{5.6}$$

5.3.3 Inserting Elite and Setting Search Radius

With the peak region and number information, we can roughly obtain elite in original search space. We insert these elite into a population and replace with individuals with relative worse fitness value. However, inserting those elites cannot ensure obtaining the final peak. Because in some of EC algorithms, such as DE used in experimental part, a parent is replaced by new offspring with better fitness value, which can locate anywhere. So we should restrict the change range of the elite, the concept of elite search radius is proposed.

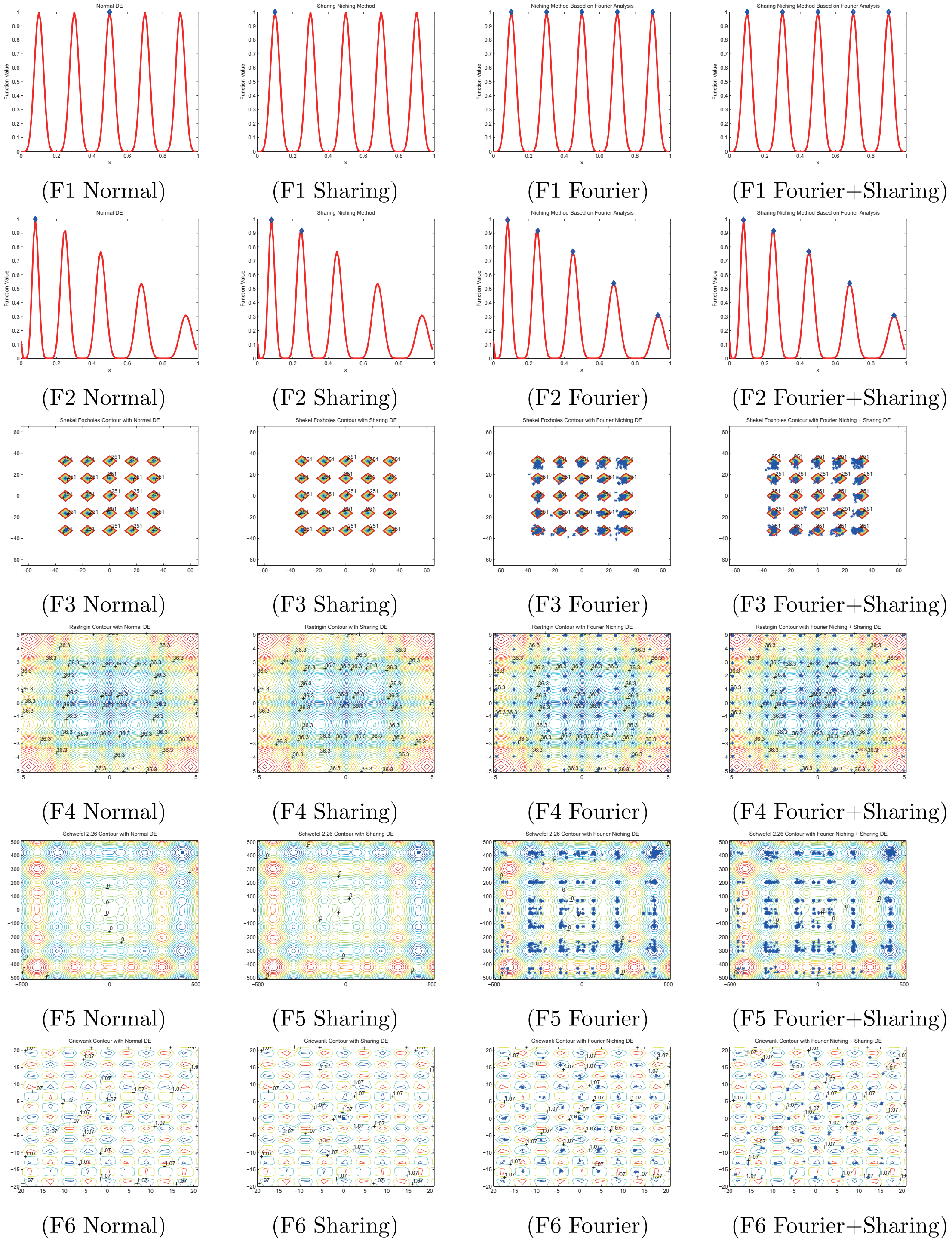
Search radius restricts search range of elite to ensure it searches for a optimum within a peak region. In this work, we set the search radius as shown in Eq. (5.7), m is a tuned parameter, in our study, we set it to 2.

$$SearchRadius_i = \frac{Range_i}{m * k_i} \tag{5.7}$$

5.4 Experimental Evaluation

5.4.1 Multimodal Benchmark Functions

We use six multimodal benchmark functions to evaluate our proposed Fourier niche method. F1 has local optima locating equally with the same amplitude, F2 has those locating unequally with different amplitude, and F3, F4, F5 and F6 are Shekel's Foxholes function, Rastrigin function, Schwefel function and Griewank function,



(F1 Normal) (F1 Sharing) (F1 Fourier) (F1 Fourier+Sharing)

(F2 Normal) (F2 Sharing) (F2 Fourier) (F2 Fourier+Sharing)

(F3 Normal) (F3 Sharing) (F3 Fourier) (F3 Fourier+Sharing)

(F4 Normal) (F4 Sharing) (F4 Fourier) (F4 Fourier+Sharing)

(F5 Normal) (F5 Sharing) (F5 Fourier) (F5 Fourier+Sharing)

(F6 Normal) (F6 Sharing) (F6 Fourier) (F6 Fourier+Sharing)

Figure 5.3: Final results of F1 and F2 with 100 population size at the 50th generation and F3, F4, F5 and F6 with 1000 population size at the 50th generation.

respectively. Their shape curves are shown in Figure 5.1 and in the Appendix (F5, F6, F7, F8). Their mathematical expressions are shown in Table 5.1.

5.4.2 Experimental Conditions

We use differential evolution (DE/best/1/bin) with a proposed Fourier niche method to evaluate multimodal benchmark functions mentioned above. For the F1 and F2, the parameter range is $0 \leq x \leq 1$, and population size is 100. For F3, F4, F5 and F6 in Table 5.1, the parameter range is as in Table 5.1. The dimensions of F3, F4, F5 and F6 are set to 2, and population size is 1000. We conduct each evaluation until the 50th generation.

Fourier niche method uses the search radius that is set as Eq. (5.7) accordance with the number of obtained peaks. We compare the performance of the proposed Fourier niche method with the sharing method and a hybrid method (sharing + Fourier method), which fuses fitness sharing method and Fourier niche method. The parameter setting of fitness sharing method is $\sigma_{sharing} = 0.3$ in Eq. (5.2). We set base frequency number of FFT to 256, and sampling data number to 256 for F1, F2 and F4; 64 for F3 and F6; and 8 for F5. Since the number of optima is known in benchmark functions, we use the number of peaks (N_{Peak}) and computational time as performance metrics. We test each function with 50 trail runs and obtained the result in Table 5.2 and Figure 5.3.

Table 5.2: Computational time of one time optimization in F1 and F2. The experiment is conducted by a Core2 Duo 2GHz CPU with 1GB RAM running Windows XP (SP2).

function.	method	time(ms)	N_{Peak}
F1	Normal	14.09	1
F1	Sharing	99.99	2
F1	Fourier	17.83	5
F1	Sharing+Fourier	107.46	5
F2	Normal	8.11	1
F2	Sharing	100.33	2
F2	Fourier	16.24	5
F2	Sharing+Fourier	106.12	5

5.4.3 Experimental Results

Table 5.2 shows the number of peaks and computational time of each method. Figure 5.3 shows the results with the 1000 population size at the 50th generation. Accordance with these test results, we can conclude as follows.

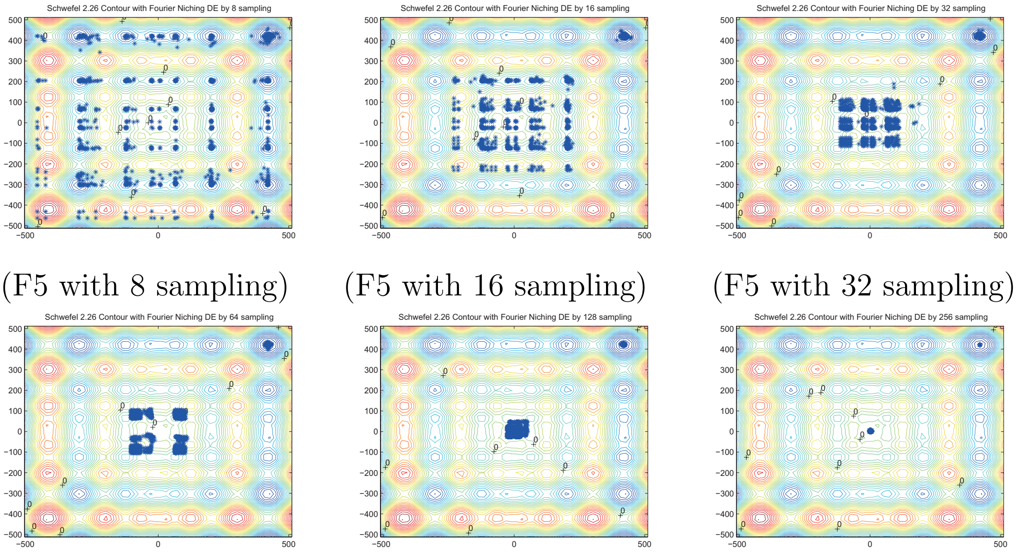
1. the Fourier niche method can obtain all of the peaks in F1 and F2,
2. the Fourier niche method can obtain more peaks of F3, F4, F5 and F6 than the ordinary DE and fitness sharing method,

3. fitness sharing method can obtain two peaks once for F1 and F2, and only one peak for F3, F4, F5 and F6, so our proposed method is better than the fitness sharing method in aspect of peak number,
4. for F3, F4, F5 and F6, the Fourier niche method can obtain almost the same number of peak as that obtained by sharing + Fourier method, and
5. the computational time of the Fourier niche method is less than the fitness sharing method and sharing+Fourier method.

5.5 Discussions

We use six multimodal benchmark functions to evaluate the Fourier niche method. The method could obtain more peaks than the fitness sharing method in all the cases. Our method could obtain all peaks of F1 and F2 with less computational cost as shown in Table 5.2. For F3, F4, F5 and F6, our method could obtain almost the peaks but not all. It is because we use the 1-D FFT for obtaining the rough location information of multi-dimensional fitness landscape, and restrict the number of elite by consideration of performance. We must ensure that the population size is set to more than the number of peaks when applying our Fourier niche method to obtain a better final result for larger scale multimodal problems.

The main purpose of applying Fourier transform is to find rough regions from frequency information, where the global or local optima in original fitness landscape locate. Obtaining accuracy frequency information is a important step for the success of Fourier niche method. Several data sampling method were therefore proposed in [79]. In order to achieve better performances for concrete multimodal optimization problems, we should select a proper sampling method and parameter setting of FFT for obtaining the frequency.



(F5 with 8 sampling) (F5 with 16 sampling) (F5 with 32 sampling)

(F5 with 64 sampling) (F5 with 128 sampling) (F5 with 256 sampling)

Figure 5.4: Final results of F5 with different sampling number.

Frequency resolution is an important concept to obtain the correct frequency and influences the parameter setting of FFT. Eq. (5.8) shows the definition of frequency resolution, where Δf is the frequency resolution that is the minimum difference between two neighbor frequencies, f_s is the sampling number, N is the number of the base frequency in the FFT that should be an integer, which can be expressed as 2^n usually.

$$\Delta f = \frac{f_s}{N} \tag{5.8}$$

We evaluated F5 with different sampling numbers of 8, 16, 32, 64, 128, and 256 (see Figure 5.4). The evaluation results implies that more sampling data do not always obtain higher accurate frequency information from the search space, which depends on the frequency resolution, i.e. the proportion of the sampling number and the base frequency number. In Figure 5.4, the number of the peaks obtained by the Fourier niching method becomes fewer accordance with the increasing sampling number. It is because the frequency resolution value becomes lower and hard to distinguish the lower frequency, and with the restriction of elite number limitation (it can not more than the population size), all the elite points search within a massive area in a search space.

We propose a search radius concept in Fourier niche method to restrict search range of elite. It is a crucial parameter for certain elite to find a global optimum or several local optima, so tuning its value for each elite is a promising research topic to improve performance of our proposal.

Compared with conventional fitness sharing method, Fourier niche method can save computational cost because it conducts special operations (FFT, etc.) only one time after initializing a population. Meanwhile fitness sharing method tunes fitness value of each individual every generations or (after) several generations. Reducing computational complexity presents our proposal's advantage. Compared with sharing + Fourier method, it still spends more computational time than that of the Fourier niche method. The number of the final peaks found by Fourier niche methods is more than that of sharing + Fourier methods. Form Figure 5.3, the Fourier niche method has better convergence performance than sharing + Fourier method.

5.6 Chapter Summary

In this chapter, we propose a Fourier niche method for solving multimodal optimization problem. There are three main steps to implement this method:

1. obtaining frequency information,
2. calculating the potential peaks' locations and their number, and
3. putting elite and setting their search radius for restricting search range.

From the evaluation results, we conclude that our proposed Fourier niche method is feasible and practical, especially with the less computational complexity.

Chapter 6

Triple and Quadruple Comparison Based Interactive Differential Evolution and Differential Evolution

6.1 Introduction

This chapter studies a method of reducing IEC user fatigue by accelerating the EC search. There have been several EC acceleration approaches proposed in the past, including a gradient method; a hill-climbing method; combining other meta-heuristic approaches [77]; and approximating the EC fitness landscape [38]. There are also several approaches of landscape approximation that have been investigated, including a method approximating the landscape with a unimodal function and estimating the global optimum area that is defined as the area located around the global optimum [36, 111], accelerating the approximation by using a lower dimensional function [74, 75, 76, 82] and landscape approximation using the Fourier transform and estimating the global optimum area [78, 79].

In this chapter, we propose and evaluate IEC acceleration methods that embed opposition-based learning (OBL) into differential evolution (DE). OBL is embedded into two stages of the DE: initial value determination and offspring generation. The conventional paired comparison-based IDE is as discussed later in section 6.2. We propose the new methods for embedding OBL into IDE in section 6.3. The proposed methods demand triple or quadruple comparisons, and the IDE user fatigue for a single comparison stage is therefore greater than in the case of conventional paired comparison-based IDE. However, the proposed methods are based on the hypothesis that IDE user fatigue may not increase drastically when the number of comparing individuals is less than a memorable number, such as three individuals, and the thought that the total fatigue of a human user can be decreased by accelerating the IDE search. We evaluate the methods using an IDE simulation in section 6.4. The proposed methods can be used not only for IDE but also for DE, and we evaluate their performance as DE accelerators using 24 benchmark functions in section 6.5. Finally, we conclude this research and describe our future research directions in section 6.6.

6.2 Paired Comparison Based Interactive Differential Evolution

In most instances of IEC, as typified by interactive genetic algorithm (IGA), all individuals are presented to the IEC user and the user is then required to input a fitness evaluation for each of them. In the case where the individuals are still images, it is easy for an IEC user to compare them spatially and evaluate them. For this reason, most IEC approaches use this method of display and evaluation.

However, when the individuals are sounds or movies, an IEC user has to compare each individual with others held in their memory and the mental load and fatigue increase. It was been pointed out that humans possess limited memory and cannot process more than five to nine different items simultaneously [65]. The population sizes of many IEC systems frequently exceed this memory limitation, and displaying 10 to 20 sounds or movies to an IEC user is not practical.

Paired comparison-based IEC solves this problem by replacing the comparison of all individuals with paired comparisons and is thus expected to reduce IEC user fatigue. The first proposed approach is a tournament IGA [39, 53]. $N-1$ paired comparisons are iterated for N individuals per generation, and fitness values are calculated using the number of winnings and fitness differences between each pair. The disadvantage of tournament GA is that the obtained fitness necessarily includes noise because the tournament is not a round robin competition against the original GA algorithm. The noise influences the GA selection operation and results in reduced GA search performance.

Differential evolution (DE) is a type of population-based optimization algorithm [87, 103]. It searches for the global optimum using a differential vector between two individuals for which the length is in proportion to the distribution size of the individuals in general and for which each parent individual generates its offspring. The following description outlines the main steps of a DE implementation.

(1) Chose one individual to be the target vector. Select two other individuals as parameter vectors at random and derive a difference vector from them; the best individual from the rest of the individuals or another individual selected at random is the base vector.

(2) Create a mutant vector by adding a weighted differential vector to the base vector.

(3) Generate a trial vector by crossing the target vector and the mutant vector.

(4) Compare the target vector with the trial vector, and select whichever one is better as the offspring for the next generation.

(5) Go to (1) and generate other offspring until all individuals are processed using the same operations,. Then proceed with the next generation.

Paired comparison-based IDE [113] does not modify any parts of its algorithm because the algorithm already includes paired comparison by nature in step (4) of the above algorithm. Since pairs of individuals are presented to the IDE user for comparison without modifying the DE algorithm, the IDE is expected to be a promising IEC method.

Table 6.1: Conventional paired comparison-based DE/IDE and proposed triple and quadruple comparison-based DE/IDE. Target-OB and trial-OB refer to the opposition vectors for target vectors and trial vectors. We use the center point of an individual distribution to calculate an opposition point.

conventional method #1	Ordinary DE [87] / paired comparison-based IDE [113] (pair comparison of a target vector and a trial vector).
conventional method #2	Switching between ordinary DE and paired comparison of a target vector and target-OB according to a jumping rate [90].
proposed method #1	Triple comparison-based DE among target, trial, and target-OB.
proposed method #2	Triple comparison-based DE among target, trial, trial-OB.
proposed method #3	Quadruple comparison-based DE among target, trial, target-OB, trial-OB.
proposed method #4	Triple comparison-based DE among target, trial, random.

6.3 Proposal of a New Opposition Based Differential Evolution

Table 6.1 shows our proposed OBDE along with conventional methods for comparison. As the first OBL technique, using OBL for the initialization of individuals as mentioned, is applicable to all the methods compared in this chapter, we only use the second OBL technique – opposition-based individual selection with jumping rate – for comparing with the proposed methods.

It is necessary to consider (a) the computational cost of a fitness value, (b) that of an introduced acceleration method, and (c) convergence speed when we use or develop EC acceleration methods. Since (c) has higher priority than (b) for tasks where (a) takes the most time, total time until convergence reaches the goal is frequently used as an evaluation index.

On the other hand, user fatigue is an important factor in the evaluation of IEC. When the mental demand for evaluating individuals is otherwise the same, we may say that the IEC user fatigue is in proportion to the total time until the IEC user finds a satisfactory individual. However, when the mental load for evaluating individuals is different due to different IEC interfaces, this relation is not always true. There are cases where IEC user fatigue becomes low thanks to easy evaluation even if total evaluation time until the goal is reached is long. Likewise there are opposite cases where IEC user fatigue can be lowered thanks to short total evaluation time even if the mental load for a single evaluation is high. We need to evaluate acceleration methods by analyzing the load of a single evaluation along with the convergence characteristics through IEC simulation, and after that we must conduct a human subjective evaluation to confirm the simulation results. This chapter deals with the IEC simulation of the first stage.

Our idea is to use not only a target vector and a trial vector but also their opposition vectors at every comparison in the DE search. Our proposed method #1 functionally includes conventional method #2 because it conducts the comparison of a trial vector and its opposition vector as in the conventional method #2. For the same reason, we may say that proposed method #3 includes the proposed methods #1 and #2 functionally. In order to compare the effectiveness of the opposite mechanism, i.e., the proposed methods #1, #2 and #3, we also propose to use a random vector as a third vector to compare the optimization performance with proposed methods #1, #2 and #3, we refer to it as proposed method #4 in this chapter.

When the proposed methods are applied to non-interactive DE, whereas conventional paired-comparison-based DE had twice the number of calculations, the proposed methods require three or four times the number of calculations. In other words, the calculation time of (a) mentioned above is 1.5 or 2.0 times that is required for the conventional paired comparison DE case. The crux of our study into the application of the proposed methods for ordinary non-interactive DE is whether the acceleration performance gains of the proposed methods exceed the increased time required for their fitness calculations, allowing the total calculation time to be reduced.

The number of fitness calculations is not the final evaluation index for IDE because IDE user fatigue is not in proportion to it. Consider the case of user fatigue from choosing the best still image between two images or from among three or four images. The former must be less than the latter, but the mental load in the latter case is not 1.5 or 2.0 times that of the former. Even when the IDE requires the comparison of sounds or movies that we cannot compare spatially and simultaneously, the IDE user's mental load must increase, but the ratio will not necessarily increase by 1.5 or 2.0 times. Generally speaking, when the number of individual comparisons is within the number that an IDE user can memorize, IEC user fatigue is lower; when it exceeds the maximum memory capacity, user fatigue increases drastically. We focus on this fact in developing our proposed methods requiring triple and quadruple comparisons, and aim to reduce the total user fatigue by accelerating IDE search even if the user fatigue for each single comparison increases.

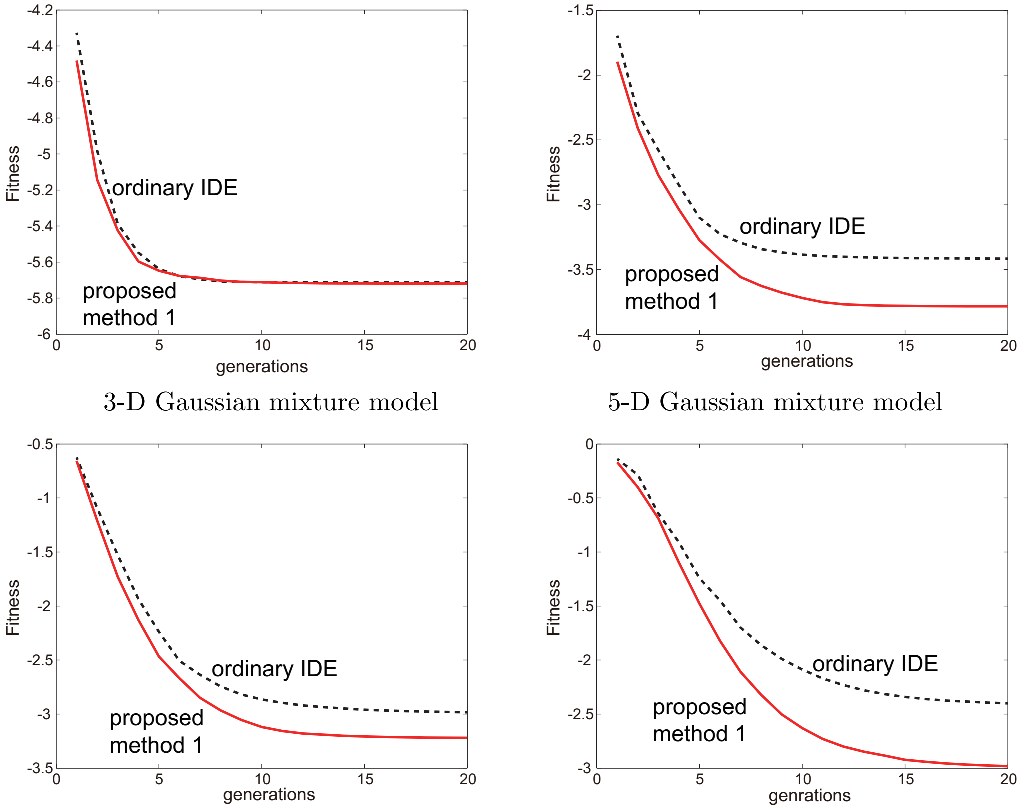
One mirror point for calculating opposition points is used in this chapter. Reference [90], i.e. conventional method #2, uses the center gravity point of an individual distribution as the mirror point. This appears reasonable for narrowing the distribution of the opposition vectors according to the DE individual distribution.

6.4 IDE Experimental Evaluation and Analysis

6.4.1 IDE User Model and Experimental Conditions

We use Gaussian mixture models for evaluation in this section. Concretely, we combine four Gaussian functions ($k = 4$) and realize the characteristics expressed by F27 in the Appendix with 3 dimensions (3-D), 5-D, 7-D, and 10-D. The IDE user simulation in this section randomly chooses either a trial vector or a target vector and leaves it as offspring in the next generation when the difference of their fitness values are less than a certain value to simulate the unavailability of a human IDE user's comparison; we set the difference threshold as 1/50 of the difference between the minimum and maximum fitness value in the population at each generation.

Experimental evaluations are conducted as follows. The four dimensional Gaussian mixture models are run for 1,000 generations. 30 trial runs of these searches with different initial search points are conducted. A Wilcoxon signed-rank test is applied at the 20th generation and the 1,000th generation to test for a significant difference between them. IDE parameters are set as described in Table 6.2. Population size, 20, is decided upon to take match the sort of population size used in IEC experiments with a real human user.



3-D Gaussian mixture model

5-D Gaussian mixture model

7-D Gaussian mixture model

10-D Gaussian mixture model

Figure 6.1: Average convergence curves for 30 trial runs with 3-D, 5-D, 7-D and 10-D Gaussian mixture models. Dot line and solid lines represent the conventional method #1 (ordinary IDE) and the proposed method #1, respectively.

6.4.2 Experimental Evaluation of the Proposed Methods

Comparisons between ordinary IDE and proposed method #1 are shown in Figure 6.1. They show how the proposed method converges faster than normal IDE at each generation.

Convergence comparisons with practical conditions, i.e. at the 20th generation with a population size of 20, are shown in Table 6.3(a). There was no significant

Table 6.2: IDE experiment parameters setting.

population size	20
search range of parameters	$[-5.12, 5.12]$
scale factor F	0.3
crossover rate	0.7
DE operation	DE/best/1/bin
max. search generation, MAX_{NFC}	1,000
dimensions of Gaussian mixture model, D	3,5,7,10
# of trial runs	30

difference between ordinary IDE and others for a 3-D Gaussian mixture model. However, convergence speed of the proposed methods became significantly faster than ordinary IDE as the task complexity increases, i.e. the dimensions of model functions increase. The proposed methods were superior to both conventional methods.

We can easily imagine the reason for the superiority from Figure 6.1. That is, for the simple 3 dimensional case, all methods converged sufficiently by the 20th generation that no significant difference could be observed. However, convergence at the 20th generation was still ongoing for the higher dimensional Gaussian mixture models, and the convergence acceleration of the proposed methods became effective.

The evaluation results in Table 6.3(b) showed the same tendency as with Table 6.3(a) though the evaluation at the 1,000th generation is not realistic for IDE but should be considered in the evaluation conducted in section 6.5.

Comparisons using only Table 6.3(a) cannot be fair because proposed methods #1, #2, #3 and #4 are triple, triple, quadruple and triple comparison-based searches, respectively, while the two conventional methods are paired comparison-based ones. Table 6.4 is another evaluation that compares convergence at the same number of fitness calculations. Since conventional methods #1 and #2 calculate fitness 40 times (= 20 individuals × 2 for each paired comparison) per generation, 1200 and 1600 fitness calculations equate to the evaluations at the 30th and the 40th generations, respectively. Likewise, they correspond to the 20th, the 20th and the 28th generation for he triple comparison-based proposed methods #1, #2 and #4, and the 16th generation for the quadruple comparison-based proposed method #3.

The results here are the same as in Table 6.3; there is no significant difference between ordinary IDE and the others when the tasks are simple, such as with a 3-D Gaussian mixture model, and they have converged in early generations. On the other hand, the proposed methods converge significantly faster than conventional methods when the complexity of tasks increases, such as for the higher dimensional Gaussian mixture model.

6.4.3 Discussion on Convergence Characteristics

We applied a Wilcoxon-signed rank test for comparison of the proposed methods #1, #2 and #3 with #4 at 20th generation, 1000th generation, 1200th fitness calculation and 1600th fitness calculation. The results show only proposed method

#3 outperforms significantly proposed method #4 in all cases with the 10-D Gaussian mixture model. There is not a significant difference between the three triple-comparison methods (proposed methods #1, #2 and #4) for the entire 4 Gaussian mixture models in all the cases.

The same statistical test is applied to compare proposed method #3 with proposed method #1 and #2. From the results, it is demonstrated that proposed method #3 performs significantly better than proposed method #1 and #2 in the 10-D Gaussian mixture model in 20th generation, 1000th generation, and 1600th fitness calculation.

We can draw some conclusions from the above observations. First, the triple-comparison mechanism using the random method offers the same enhancement to performance as the opposite method. Second, for the IDE case, the triple-comparison mechanism implemented by opposition does not show any advantages over randomness. Third, for the higher dimension problem (10-D Gaussian mixture model), proposed method #3, i.e., the quadruple-comparison method, significantly shows better performance than any other proposed methods.

6.5 DE Experimental Evaluation and Analysis

6.5.1 Test Functions and Experimental Conditions

We evaluate our proposed methods with 24 benchmark functions. Our proposed methods aim to reduce IDE user fatigue by accelerating IDE search even if user fatigue for a single comparison is increased. However, their use is not limited to IDE but rather they are applicable to any DE searches with a fitness function. We compare them with conventional methods of ordinary DE and OBDE using minimum optimization problems made by 24 benchmark functions [104]. Definitions, search ranges for optimization parameters, the global optimum values, and characteristics of the benchmark functions can be seen in Table 6.5.

Because there is no need to evaluate IDE user fatigue, as was done in section 6.4, the evaluation indices used in this section for comparing our proposed method applied to DE are the number of fitness calculations (NFC) until the convergence threshold (CT) is reached and the success ratio (SR) for those that reach the CT. The lower NFC is, the faster convergence is.

Evaluations trial runs are 30, and the maximum generation is MAX_{NFC} =1,000. A successful convergence is counted when a convergence reaches the CT defined by Eq. (6.1). All benchmark functions are evaluated by an acceleration ratio, AR, too, to evaluate convergence speed. It is defined using NFC at the maximum generation, the 1,000th generation, in Eq. (6.4), and $AR > 1$ means that the proposed method converges faster than ordinary DE. Success ratio, SR, is defined by the number of trials that reached the convergence threshold, CT, in Eq. (6.3). Furthermore, average acceleration ratio and average success ratio are calculated and used for the final evaluation results.

Table 6.3: Average fitness values at the 20th and 1000th generation for 3-D, 5-D, 7-D, and 10-D Gaussian mixture models (pseudo-IDE users). Conventional method #2 is opposition-based generation jump with a jumping rate of 37% [90]. See details for the conventional and proposed methods in Table 6.1. Bold values and daggers mean that Wilcoxon sign-ranked tests showed significant differences between conventional method #1 and the proposed method and between conventional method #2 and the proposed method, respectively (p <0.05).

(a) Average fitness at the 20th generation.

method	3-D	5-D	7-D	10-D
conventional method #1	-5.71	-3.42	-2.98	-2.40
conventional method #2	-5.70	-3.46	-3.03	-2.48
IDE-proposed method #1	-5.72	**-3.78**†	**-3.22**†	**-2.98**†
IDE-proposed method #2	-5.73	**-3.57** †	**-3.19** †	**-2.87** †
IDE-proposed method #3	-5.72	**-3.73**†	**-3.25**†	**-3.14**†
IDE-proposed method #4	-5.70	**-3.66**†	**-3.12**	**-2.93**†

(b) Average fitness at the 1,000th generation.

method	3-D	5-D	7-D	10-D
conventional method #1	-5.71	-3.42	-3.00	-2.44
conventional method #2	-5.70	-3.47	-3.04	-2.49
IDE-proposed method #1	-5.72†	**-3.78**†	**-3.22**†	**-3.00**†
IDE-proposed method #2	-5.73 †	**-3.62**†	**-3.24**†	**-2.93**†
IDE-proposed method #3	-5.72	**-3.73**†	**-3.25**†	**-3.18**†
IDE-proposed method #4	-5.70	**-3.66**†	**-3.13**	**-2.95**†

Table 6.4: Average fitness values when the number of fitness calculation reaches 1200 and 1600. For details, see the values indicated in bold or with daggers in Table 6.3 and those of the others in Table 6.1.

(a) Average fitness at the 1200th fitness calculation.

mrthod	3-D	5-D	7-D	10-D
conventional method #1	-5.71	-3.42	-2.99	-2.43
conventional method #2	-5.70	-3.47	-3.04	-2.49
IDE-proposed method #1	-5.72	**-3.78**†	**-3.22**†	**-2.99**†
IDE-proposed method #2	-5.73	**-3.57**†	**-3.20**†	**-2.89**†
IDE-proposed method #3	-5.72	**-3.73**†	**-3.22**†	**-3.08**†
IDE-proposed method #4	-5.70	**-3.66**†	**-3.13**	**-2.93**†

(b) Average fitness at the 1600th fitness calculation.

method	3-D	5-D	7-D	10-D
conventional method #1	-5.71	-3.42	-3.00	-2.43
conventional method #2	-5.70	-3.47	-3.04	-2.49
IDE-proposed method #1	-5.72	**-3.78**†	**-3.22**†	**-3.00**†
IDE-proposed method #2	-5.73	**-3.61**†	**-3.22** †	**-2.92**†
IDE-proposed method #3	-5.72	**-3.73**†	**-3.25**†	**-3.14**†
IDE-proposed method #4	-5.70	**-3.66**†	**-3.13**	**-2.95**†

Table 6.5: Twenty-four benchmark functions adopted from [104] are used in the experimental evaluations. (Uni=Unimodal, Mul=Multimodal, Rt=Rotated, GB=Global on Bounds, HC=Hybrid Composition, NM=Number Matrix, NS=Non-Separable, and S=Separable.)

No.	Name	Range	Optima	Characters
F1	Sphere	[-100,100]	-450	Uni-S
F2	Schwefel 1.2	[-100,100]	-450	Uni-NS
F3	Elliptic	[-100,100]	-450	Uni-NS
F4	F2 with Noise	[-100,100]	-450	Uni-NS
F5	Schwefel 2.6 GB	[-100,100]	-310	Uni-NS
F6	Rosenbrock	[-100,100]	390	Mul-NS
F7	Griewank	[0,600]	-180	Mul-NS
F8	Ackley GB	[-32,32]	-140	Mul-NS
F9	Rastrigin	[-5,5]	-330	Mul-S
F10	Rt Rastrigin	[-5,5]	-330	Rt-Mul-NS
F11	Weierstrass	[-0.5,0.5]	90	Mul-NS
F12	Schwefel 2.13	[-100,100]	-460	Mul-NS
F13	Expanded F8F2	[-3,1]	-130	Mul-NS
F14	Scaffer F6	[-100,100]	-300	Mul-NS
F15	HC Function	[-5,5]	120	HC-S
F16	Rt HC F1	[-5,5]	120	Rt-HC-NS
F17	F16 with Noise	[-5,5]	120	Rt-HC-NS
F18	Rt HC F2	[-5,5]	10	Rt-HC-NS
F19	F18 with Basin	[-5,5]	10	Rt-HC-NS
F20	F18 with GB	[-5,5]	10	Rt-HC-NS
F21	Rt HC F3	[-5,5]	360	Rt-HC-NS
F22	F21 with NM	[-5,5]	360	Rt-HC-NS
F23	NC Rt F2	[-5,5]	360	HC-NS
F24	Rt HC F4	[-5,5]	260	Rt-HC-NS

$$converg.\,threshold,\ CT = average\,fitness\,of\,each\ method\,at\,MAX_{NFC} - th\ generation. \tag{6.1}$$

$$NFC = average\,\#\,of\,fitness\ calculation\,until\,convergence\ reaches\,CT. \tag{6.2}$$

$$success\,ratio,\ SR = \frac{\#\,of\,reached\,to\,CT}{\#\,of\,MAX_{NFC}} \tag{6.3}$$

$$acceleration\,ratio,\ AR = \frac{NFC_{ordinaryDE}}{NFC_{proposal}} \tag{6.4}$$

The DE experimental parameters are set as shown in Table 6.6. The evaluation is conducted under difficult search conditions; only 50 individuals are used to search

10-D functions for which search parameter ranges have been expanded.

Table 6.6: DE experiment parameters setting.

population size	50
scale factor F	0.3
crossover rate	0.7
DE operations	DE/best/1/bin
max. search generation, MAX_{NFC}	1,000
convergence threshold, CT	CT in Table 6.7
dimensions of benchmark functions, D	10
# of trial runs	30

6.5.2 Experimental Evaluation of the Proposed Methods

Convergence characteristics for the 24 benchmark functions are shown in Table 6.7. Numerical values in the table are the average fitness values over 30 trial runs at the 1,000th generation. Convergence thresholds, CT's, for convergence success ratio, SR, in Table 6.8 are also listed in this table. Comparisons of computational cost are listed in Table 6.8.

The proposed methods converged faster than the conventional methods at the same generation for all but a few functions. This effect cannot be observed for the F12 function, where all methods failed to approach its global optimum. The global optimum of the F12 function is -460, but the fitness values of all methods at the 1,000th generation are still large positive values, which means there was no convergence at all. It must be too difficult to search the 10-D F12 function with only 50 individuals. These methods have not reached the global optima of F15 - F24 functions at the 1,000th generation, but their average fitness values are close to the global optima and their convergence are largely better than that of the ordinary DE. As a general remark, the acceleration effectiveness of the proposed methods is significant except in the case of F12 and F18-F20, which proved to be too difficult for all the methods under the given experimental conditions.

Comparative evaluations of the computational costs, i.e. the number of fitness calculations, and the success ratios, SR, of reaching a convergence threshold, CT, are shown in Table 6.8. It is hard for the ordinary DE to search 10-D benchmark functions with only 50 individuals; the average SR for the 24 benchmark functions was 17%, and the SR of only 3 of the 24 functions exceeded 50%. The average SR of conventional method #2 was better than that of #1 and was 60%, but our proposed methods achieved around 60%-80%. They showed better performance for all benchmark functions except for the F12 and F18-F20 functions where none of the methods converged with the difficult experimental conditions given in this section.

The average number of fitness calculations, NFS, before reaching the convergence threshold, CT, was lower using our proposed methods than when using the conventional methods. Not only convergence evaluation at the same generation as shown in Table 6.7, but also NFS in Table 6.8 showed that the proposed methods reached the CT quickly with less computational cost. From these results, we can say that the proposed methods can realize an effective acceleration of convergence.

The generation number for when a DE search reaches the convergence threshold, CT, is calculated as:

$$generation\, \# = NFC/(population\ size \times p),$$

where p is the number of individuals being compared per comparison, and p = 2, 3, and 4 for paired comparison-based conventional methods #1 and #2, triple comparison-based proposed methods #1 #2 and #4, and quadruple comparison-based proposed method #3, respectively.

As a summary of the experimental results, all four kinds of our proposed methods #1 – #4 are better than or equal to the conventional methods for all 24 benchmark functions by number of fitness calculations, the success ratio for reaching a convergence threshold, acceleration ratio, and fitness value at each generation.

6.5.3 Discussion on Functional Inclusion and Convergence Characteristics

Although proposed method #1 includes the conventional method #2 and proposed method #3 includes the proposed methods #1 and #2 functionally as mentioned in section 6.3, methods that included the functionality of one or more methods did not always perform better than the method(s) they included, though they outperformed them in general. We now discuss this point in more detail.

The reason for this result is that it is not guaranteed that the continuous selection of the best individual at each comparison as *best-best-best* will become the total best; in other words, the principle of optimality [5] does not hold in this case. Probabilistically speaking, it is expected that areas around better individuals are more likely to reach the global optimum than other areas. However, monotonicity, which is a requirement of the principle of optimality, does not stand up in multimodal tasks, and it is not guaranteed that the areas where opposite vectors are chosen are advantageous to reach the global optimum even if the opposite vectors are the best among the triple or quadruple individuals used for comparison. Additionally, if a non-advantageous individual is chosen, subsequent selections will also be affected by this choice.

From these two points, when method A includes method B functionally, we can say that the method A is expected to converge faster than the method B probabilistically but this expectation is not guaranteed. For further investigation of the enhancement to performance among our proposed methods, we applied Wilcoxon signed rank test to compare the performance between proposed method #3 and proposed methods #1 and #2. The results show that proposed method #3 is significantly better than proposed methods #1 and #2 for F3, F6-F9, F11 and F24, and only worse than proposed methods #1 and #2 in F3. The experimental results in Tables 6.7 and 6.8 are consistent with this discussion.

The effectiveness of the third vector from the random mechanism and the opposition mechanism is compared by applying Wilcoxon signed rank test on the proposed method #1, #2, #3 and #4. The results show the proposed method #4 is signifi-

Table 6.7: Average fitness values for the benchmark function, F1 – F24 at the 1,000th generation over 30 trial runs. Bold values and daggers mean that Wilcoxon sign-ranked tests showed significant differences between conventional method #1 and the proposed method and between conventional method #2 and the proposed method, respectively (p <0.05). *Conventional* and *proposal* mean respectively *conventional method* and *proposed method.*

method	F1	F2	F3	F4	F5	F6
conventional #1	-354.71	-250.18	192113.06	-141.32	2359.79	733669.47
conventional #2	**-413.53**	**-359.01**	308946.65	**-448.03**	26.70	287810.81
DE-proposal #1	**-440.88†**	**-438.45†**	67357.98	**-416.28†**	**-39.89**	**10528.84†**
DE-proposal #2	**-444.75†**	**-441.15†**	**26538.41†**	**-433.12†**	**-127.78†**	**7668.20†**
DE-proposal #3	**-447.59†**	**-440.02†**	**12532.29†**	**-443.62†**	**-120.31†**	**1042.11†**
DE-proposal #4	**-440.47†**	**-377.29**	70371.76†	**-418.67†**	**179.05**	**13491.41†**
CT	-423.66	-384.35	112976.69	-383.51	379.59	175701.81

method	F7	F8	F9	F10	F11	F12
conventional #1	-170.87	-119.60	-313.48	-302.79	95.21	3374.60
conventional #2	**-179.41**	**-136.56**	**-320.56**	**-320.56**	**91.16**	18078.54
DE-proposal #1	**-179.74†**	**-137.68†**	**-323.89†**	**-323.89†**	**90.89**	12610.21
DE-proposal #2	**-179.84†**	**-138.29†**	**-323.44†**	**-323.44†**	**90.92**	13725.32
DE-proposal #3	**-179.87†**	**-138.72†**	**-324.02†**	**-324.02†**	**90.73†**	14801.29
DE-proposal #4	**-179.76†**	**-138.23†**	**-321.11**	**-321.11†**	**90.63**	15473.06
CT	-178.25	-134.85	-321.08	-319.30	91.59	13010.50

method	F13	F14	F15	F16	F17	F18
conventional #1	-127.92	-296.71	498.21	299.84	306.39	990.61
conventional #2	-128.18	**-297.98**	**158.22**	**158.22**	**157.13**	992.86
DE-proposal #1	**-128.99†**	**-298.90†**	**148.68†**	**148.68†**	**147.75†**	959.60
DE-proposal #2	**-129.09†**	**-298.97†**	**147.01†**	**147.01†**	**145.43†**	962.36
DE-proposal #3	**-129.06†**	**-299.01†**	**143.99†**	**143.99†**	**144.87†**	966.31
DE-proposal #4	**-129.23†**	**-298.85†**	**153.54**	**153.54**	**154.76**	936.51†
CT	-128.75	-298.40	208.27	175.21	176.05	968.04

method	F19	F20	F21	F22	F23	F24
conventional #1	990.25	992.80	1463.61	1253.51	1546.75	1077.70
conventional #2	1006.69	1005.34	**414.45**	**414.45**	**400.56**	**911.72**
DE-proposal #1	958.27	959.23	**403.18†**	**403.18†**	**385.85†**	**354.11†**
DE-proposal #2	970.57	965.20	**400.62†**	**400.62†**	**386.21†**	**339.78†**
DE-proposal #3	965.93	964.67	**398.10†**	**398.10†**	**384.86†**	**310.60†**
DE-proposal #4	942.82†	935.39 †	**400.61†**	**400.61†**	**387.76†**	**381.16†**
CT	972.42	970.44	580.09	545.08	582.00	562.51

Table 6.8: Evaluation indices for 30 trial runs of benchmark functions, F1 – F24. See Eq.s (6.2), (6.3), and (6.4) for the # of fitness calculations, NFC, convergence success ratio, SR, and acceleration ratio, AR, respectively. *conventional* and *proposal* are defined as in the caption of Table 6.7.

	conventional #1 (DE)			conventional #2 (OBDE)			DE-proposal #1		
function	NFC	SR	AR	NFC	SR	AR	NFC	SR	AR
F1	60897	0.39		51644	0.62	1.18	16750	0.89	3.64
F2	70757	0.29		65527	0.52	1.08	3040	0.98	23.28
F3	38357	0.62		69391	0.49	0.55	31590	0.79	1.21
F4	79197	0.21		8741	0.94	9.06	23085	0.85	3.43
F5	86620	0.13		11047	0.92	7.84	7315	0.95	11.84
F6	47300	0.53		15390	0.89	3.07	1520	0.99	31.12
F7	83683	0.16		10791	0.92	7.75	1455	0.99	57.51
F8	100000	0.00		28802	0.79	3.47	6455	0.96	15.49
F9	86913	0.13		65710	0.52	1.32	29415	0.80	2.95
F10	100000	0.00		38812	0.72	2.58	14655	0.90	6.82
F11	100000	0.00		47352	0.65	2.11	36685	0.76	2.73
F12	43357	0.57		92557	0.32	0.47	120095	0.20	0.36
F13	67763	0.32		88169	0.36	0.77	37470	0.75	1.81
F14	100000	0.00		101910	0.26	0.98	62850	0.58	1.59
F15	100000	0.00		2069	0.98	48.34	2050	0.99	48.78
F16	100000	0.00		7234	0.95	13.82	7680	0.95	13.02
F17	100000	0.00		20600	0.85	4.85	3330	0.98	30.03
F18	70373	0.30		105422	0.23	0.67	105405	0.30	0.67
F19	67060	0.33		100891	0.26	0.66	95620	0.36	0.70
F20	70370	0.30		105390	0.23	0.67	95640	0.36	0.74
F21	100000	0.00		1238	0.99	80.80	800	0.99	125.00
F22	100000	0.00		1375	0.99	72.75	955	0.99	104.71
F23	100000	0.00		1196	0.99	83.58	695	1.00	143.88
F24	96700	0.03		51489	0.62	1.88	6845	0.95	14.13
average		0.17			0.60	10.65		0.80	18.34

	DE-proposal #2			DE-proposal #3			DE-proposal #4		
function	NFC	SR	AR	NFC	SR	AR	NFC	SR	AR
F1	11585	0.92	5.26	2273	0.99	26.79	21595	0.86	2.82
F2	2915	0.98	24.27	3993	0.98	17.72	56860	0.62	1.24
F3	11665	0.92	3.29	2253	0.99	17.02	41370	0.72	0.93
F4	13405	0.91	5.91	4040	0.98	19.60	34880	0.77	2.27
F5	6985	0.95	12.40	3033	0.98	28.56	37010	0.75	2.34
F6	1305	0.99	36.25	1593	0.99	29.69	6420	0.96	7.37
F7	1380	0.99	60.64	1753	0.99	47.73	1510	0.99	55.42
F8	1430	0.99	69.93	1880	0.99	53.19	1495	0.99	66.89
F9	33680	0.78	2.58	39620	0.80	2.19	62675	0.58	1.39
F10	18920	0.87	5.29	20167	0.90	4.96	47920	0.68	2.09
F11	17260	0.88	5.79	22833	0.89	4.38	17320	0.88	5.77
F12	119970	0.20	0.36	163673	0.18	0.26	136445	0.09	0.32
F13	27450	0.82	2.47	49427	0.75	1.37	12735	0.92	5.32
F14	38135	0.75	2.62	47420	0.76	2.11	49055	0.67	2.04
F15	2050	0.99	48.78	3260	0.98	30.67	2225	0.99	44.94
F16	7785	0.95	12.85	4580	0.98	21.83	8030	0.95	12.45
F17	3260	0.98	30.67	5720	0.97	17.48	8540	0.94	11.71
F18	95735	0.36	0.74	121227	0.39	0.58	85925	0.43	0.82
F19	95580	0.36	0.70	127660	0.36	0.53	85935	0.43	0.78
F20	90775	0.39	0.78	127753	0.36	0.55	85860	0.43	0.82
F21	755	0.99	132.45	947	1.00	105.63	875	0.99	114.29
F22	930	0.99	107.53	1200	0.99	83.33	1045	0.99	95.69
F23	590	1.00	169.49	807	1.00	123.97	735	1.00	136.05
F24	11765	0.92	8.22	9060	0.95	10.67	16810	0.89	5.75
average		0.80	21.98		0.60	17.11		0.71	16.76

cantly worse than the proposed method #1, #2 and #3 for most of the benchmark functions. This shows that the the third vector is more effective when selected from the opposition than at random, which also confirms the previous work in [91].

6.6 Chapter Summary

We proposed new OBDE methods embedding opposition-based learning into DE and evaluated them. The proposed methods are triple and quadruple comparison-based methods. Fitness calculation cost per comparison is higher than ordinary DE and ordinary OBDE based on paired comparison. However, experimental evaluations using an IDE simulation and 24 benchmark functions showed better acceleration performance than conventional methods on converged fitness values at same generations, the number of fitness calculations, the success ratio for reaching a convergence threshold, and acceleration ratio.

There are many IDE tasks where the IDE user can memorize three or four individuals. In these cases, we expect that IDE user fatigue for our proposed methods will not become 1.5 or 2.0 times of that of paired comparison-based conventional methods. Taking account of their fast convergence performance and the non-proportional characteristics of IDE user fatigue, we conclude that the proposed methods are effective, especially for IDE.

Chapter 7

EC Acceleration by Accelerating Transition from Exploration to Exploitation

7.1 Introduction

The objective of this work is to propose an acceleration method for EC by accelerating transition from exploration to exploitation, observe their behavior, and obtain hints to further accelerate EC. The proposed acceleration method deletes poor inactive individuals for several generations and generates their alternative individuals around the top individuals. To generalize this method, we observe its behavior by combining it with canonical DE.

Following this brief introductory, a new proposal of EC acceleration by transition from exploration to exploitation is reported in section 7.2. A series of experimental evaluation is conducted and discussed to analyze the proposal in section 7.3. We discuss the experimental results and some related issues in section 7.4. Finally, we conclude the whole work in section 7.5.

7.2 Acceleration from Exploration to Exploitation

It is important for generic EC algorithms to search better solutions widely in early generations, i.e. exploration, and gradually shift to exploitation. Important but difficult point is how to transfer from the exploration to the exploitation or change their balance. Simple GA does not have this kind of mechanism, and it is difficult to control the balance. Sometimes EC algorithm convergence is slow and sometimes a premature convergence happens. Keeping statistical characteristics of parent population to the next generation [43] is one of indexes or guidelines for adequate control of the balance. DE has this mechanism in its algorithm. Average length of differential vectors is in proportion to the distribution size of population and becomes shorter gradually according to the search convergence. DE controls the transition speed from exploration to exploitation taking account of the shrinking speed of the distribution size.

Basic idea behind our proposed method is that EC search can be accelerated by eliminating useless individuals for exploration and accelerating the shift to exploitation. The proposed method is to delete poor individuals that do not evolve for several generations and generate the same number of new individuals around better individuals as shown in Figure 7.1. This idea can be express in the below rule parametrically.

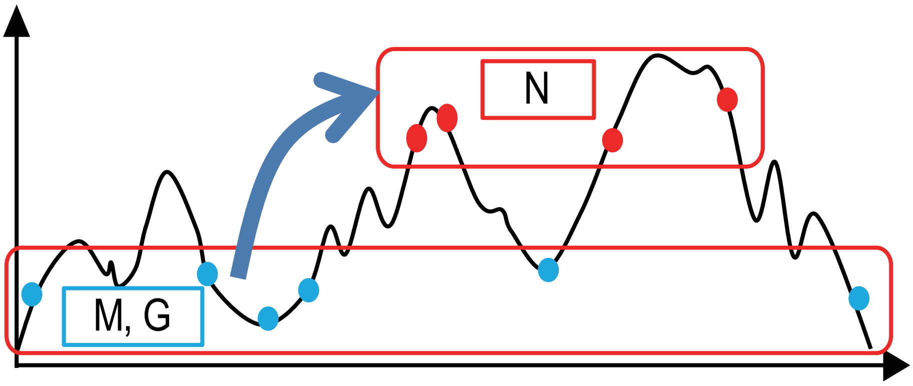


Figure 7.1: The concept of our proposed method. Poor individuals whose fitness values are low and that do not change for several continuous generations are deleted, and the same number of new individuals are generated around better individuals randomly.

IF	the fitness values of individuals among the M worst individuals do not change in continuous past G generations,
THEN	delete them and generate the same number of new individuals around the N best individuals.

There are several variations of generating new individuals around the N best individuals. As one of them, we generate new individuals by adding normally distributed random values to the best individuals selected by a roulette wheel selection among the N best individuals in section 7.3.

Our proposed method cannot be applied to GA and other EC approaches that generate an offspring set from a parent set, and its applicability is limited to DE, PSO, and other EC approaches that each of parent individuals generates one offspring.

7.3 Experimental Evaluations of the Proposed Method

7.3.1 Experimental Conditions

We evaluate our proposed method using DE under the experimental condition shown in Table 7.1.

There are several realizations of our proposed method because of parameters (M, N, G) described in section 7.2, a method for selecting individuals among the N best individuals (we use a roulette wheel selection in this chapter), and a parameter that

Table 7.1: DE conditions used in experimental evaluations.

population size	500
scale factor (F)	1.0
crossover rate	0.9
DE operations	DE/best/1/bin
max. generations	200 (F_1 – F_7), 400 (F_8 – F_{14})
# of trial runs	30
tasks	14 10-D benchmark functions in Table 7.2

decides neighbor positions around the selected best individuals (we use Gaussian distributed noise with a standard deviation σ_i: σ_i = search range of the i-th variable × k. See Table 7.3.).

Task characteristics and these parameters must influence to the performance of the proposed method, too. As the first stage of this research, we observe its performance by changing these parameters.

Benchmark functions used in this evaluations are 10 dimensional (10-D) F_1 – F_{14} in the CEC2005 benchmark function set [104] (see Table 7.2.) They are minimization tasks for finding the global optimum whose fitness is 0.

Table 7.2: Fourteen benchmark functions for experimental evaluations. The below symbols express their characteristics: Uni=Unimodal, Mul=Multimodal, Sh=Shifted, Rt=Rotated, GB=Global on Bounds, NS=non-separable, and S=separable.

function No.	benchmark functions	search ranges	function characteristics
F_1	Sh Sphere	$[-100,100]^{10}$	Sh-Uni-S
F_2	Sh Schwefel 1.2	$[-100,100]^{10}$	Sh-Uni-NS
F_3	Sh Rt Elliptic	$[-100,100]^{10}$	Sh-Rt-Uni-NS
F_4	F_2 with Noise	$[-100,100]^{10}$	Sh-Uni-NS
F_5	Schwefel 2.6 GB	$[-100,100]^{10}$	Uni-NS
F_6	Sh Rosenbrock	$[-100,100]^{10}$	Sh-Mul-NS
F_7	Sh Rt Griewank	$[0,600]^{10}$	Sh-Rt-Mul-NS
F_8	Sh Rt Ackley GB	$[-32,32]^{10}$	Sh-Rt-Mul-NS
F_9	Sh Rastrigin	$[-5,5]^{10}$	Sh-Mul-S
F_{10}	Sh Rt Rastrigin	$[-5,5]^{10}$	Sh-Rt-Mul-NS
F_{11}	Sh Rt Weierstrass	$[-0.5,0.5]^{10}$	Sh-Rt-Mul-NS
F_{12}	Schwefel 2.13	$[-100,100]^{10}$	Mul-NS
F_{13}	Sh extended F_8,F_2	$[-3,1]^{10}$	Sh-Mul-NS
F_{14}	Sh Rt Scaffer F_6	$[-100,100]^{10}$	Sh-Rt-Mul-NS

7.3.2 Preliminary Experiments

Firstly, we observed convergences of DE with/without the proposed method with the parameter values of the preliminary experiment, Exp1-1, in Table 7.3. Its results were: the performance of (DE + the proposed method) is almost similar to that of normal DE or slightly better at the final generation for five benchmark functions, F_3, F_8, F_{11}, F_{12}, and F_{14}; critical acceleration effect were observed for other nine functions.

Table 7.3: Parameter conditions of two preliminary experiments. PS and SR mean a population size and DE search ranges of optimization variables, respectively. See section 7.2 for M, N, and G and section 7.3.1 for k.

experiment No.	M	N	k	G
Exp1-1	PS×0.25	PS ×0.01	SR×0.05	5
Exp1-2	PS ×0.30	PS ×0.30	SR×0.37	6

Secondly, we tried to optimize rule parameters themselves using DE. Unfortunately, the computational cost of this optimization is quite high, and its calculations do not finish within several days under the condition of Table 7.1. There is nothing for it but to do it under the simplified condition of Table 7.4; it is based on the thought that we should be satisfied if better parameters than those of the preliminary experiment, Exp1-1, are obtained. The obtained average values of (M, N, k, and G) of 14 benchmark functions are shown as the preliminary experiment, Exp1-2, in Table 7.3.

Table 7.4: Simplified experimental conditions for optimizing the parameters of the proposed method.

population size	100
scale factor (F)	1.0
crossover rate	0.9
DE operations	DE/best/1/bin
max. generations	20
# of trial runs	30
tasks	14 5-D functions in Table 7.2

Contrary to our expectation, an acceleration effect of the preliminary experiment, Exp1-2, was clearly poorer than that of the Exp1-1, and its performance is almost similar to that of the normal DE (see Figure 7.2). Of course, it is hard to say that the parameter values of Exp1-2 is really optimized due to fewer population size, fewer generations, different dimensions of used benchmark functions, and averaged values for 14 functions. Even though, the performance difference between two preliminary experiments with Exp1-1 and Exp1-2 is big. To analyze the difference, we observe the relationship between parameter values of the proposed method and convergence characteristics in the next section.

7.3.3 Parameter Values of Our Proposal and Convergence Characteristics

We observe convergence performances by replacing each of four Exp1-2 parameter values with those of Exp1-1 whose performance was better than Exp1-2 one by one.

These experimental conditions are shown in Table 7.5.

Table 7.5: Experimental conditions that combine parameters of two preliminary experiments. PS and SR mean a population size and DE search ranges of optimization variables, respectively.

experiment No.	M	N	k	G
Exp2-1	PS×**0.25**	PS×0.30	SR×0.37	6
Exp2-2	PS×0.30	PS×**0.01**	SR ×0.37	6
Exp2-3	PS×0.30	PS×0.30	SR ×**0.05**	6
Exp2-4	PS×0.30	PS×0.30	SR ×0.37	**5**
Exp1-1	PS×**0.25**	PS×**0.01**	SR ×**0.05**	**5**
Exp1-2	PS×0.30	PS×0.30	SR ×0.37	6

Their results are shown in Figure 7.2. Significances between normal DE and (DE + proposed method with different Exp conditions in Table 7.5) at the 10th, 100th, and 200th generation are tested using the Wilcoxon's signed-rank test and shown in Table 7.6.

7.4 Discussions

7.4.1 Performance of the Proposed Method

From Table 7.6 and Figure 7.2, there is a tendency that an acceleration performance becomes better when N and k are small, i.e. Exp1-1, Exp2-2, and Exp2-3. The methods of DE + the proposed method with bigger values of N and k are better than normal DE from Table 7.6, but their effectiveness is not big as you see in Figure 7.2. One of its reasons would be that the performance of normal DE is not bad and converges relatively fast.

When the complexities of given tasks increase, their convergences become slow and the number of inactive individuals increases. It results to increase the number of applying a proposed rule that deletes poor inactive individuals and generates new individuals around the best individuals. According to its increase, we can expect that the effect of proposed method increases, too.

To confirm this point, we evaluate the proposed method with the same benchmark functions with higher complexity. Although there are several ways to increase a complexity, such as decreasing population size or increasing task dimensions, we increase the complexities of 14 benchmark functions by expanding search ranges. Concretely speaking, initial population is randomly generated within the search ranges in Table 7.2, but restricting search ranges is not applied during DE search. Non-use of the range restriction causes the individuals to be outside of the range in Table 7.2, expands the distribution of individuals, and makes DE search difficult.

The convergence results under this condition are shown in Figure 7.3. The discussion points mentioned in the above became prominent.

Our observation was that it was better to generate new individuals quite near (normal distribution noise of standard deviation was 5% of search range in our experiment in section 7.3) around the quite top individuals (top 1% among all individuals in our experiment). It means that quicker transition from exploration to exploitation accelerates search convergences well if we delete poor individuals and generate new individuals as their alternatives.

In general, this policy looks risky and to have a tendency of premature convergence. However, it is also true that keeping poor individuals for long generations is not a solution of the premature convergence. Nevertheless, we need further risk assessments of our proposed method by observing whether it causes premature convergence and how it frequently cause the premature convergence if it surely causes.

Table 7.6: Wilcoxon's signed-rank test results for the differences between convergence of DE and (DE + our proposed method) at the 10th, 100th, and 200th generation in Figure 7.2. See Exp numbers in Table 7.5. ** and * mean significance of ($p < 0.01$) and ($p < 0.05$), respectively.

test generation	DE vs.	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}
at the 10th	Exp1-1	**				*	**	**			**			**	
	Exp1-2														
	Exp2-1														
	Exp2-2	**	*				**	**		**	**	**		**	
	Exp2-3	**					**	*		*			*	**	
	Exp2-4														
at the 100th	Exp1-1	**	**		**	**	**	**		**	**			**	
	Exp1-2	**					**	**				*	*	**	
	Exp2-1	**	**		**		**	**					*	**	
	Exp2-2	**	**		**	**	**	**		**	**			**	
	Exp2-3	**	**	**	**	**	**	**		**	**	**	*	**	
	Exp2-4	**	**		*		**	**					*	**	
at the 200th	Exp1-1	**	**		**	**	**	**		**	**	*		**	
	Exp1-2	**	**	*	**		**	**				**		*	
	Exp2-1	**	**	**	**		**	**					*	**	
	Exp2-2	**	**		**	**	**	**		**	**		*	**	**
	Exp2-3	**	**	**	**	**	**	**		**	**	**	*	**	**
	Exp2-4	**	**	*	**	*	**	**					*		
at the 300th	Exp1-1									**	**	**	**	**	**
	Exp1-2									**	*	*			
	Exp2-1									**					
	Exp2-2									**	**	*	*	**	**
	Exp2-3									**					
	Exp2-4									**			*		
at the 400th	Exp1-1								**		**		**	**	**
	Exp1-2									*					
	Exp2-1														
	Exp2-2										**		**	**	
	Exp2-3									*					
	Exp2-4									**			*		

7.4.2 Distribution of Individuals

Next, we observe how distributions of all individuals change according to convergence. The below average value of standard deviations for 10-D and 30 trials is used as its index at the i-th generation.

$$\sigma_{i\text{-th generation}} = \frac{1}{\text{dimensions}} \sum_{d=1}^{\text{dimensions}} \left(\frac{1}{\text{trials}} \sum_{t=1}^{\text{trials}} \sigma_{idt} \right)$$

Figure 7.4 shows the transition of the standard deviation of Figure 7.2. A unique feature in this figure is the following; when $\sigma_{i\text{-th generation}}$ becomes small, i.e. a distribution of whole individuals becomes small, DE search works well and our proposed method is effective, too; when $\sigma_{i\text{-th generation}}$ becomes big, DE convergence is poor and the proposed method is not effective or a few. Although the same figures for Figure 7.3 are not posted in this chapter, this feature appeared in the figures clearly. As $\sigma_{i\text{-th generation}}$ is an average standard deviation over dimensions at the i-th generation of 30 trial runs, the increase of this simplified index does not tell us whether it happens in whole dimension or in only certain dimension. Through further analyses in the next step, we may obtain hints to know the difficulty of tasks and switch search strategies though the analyses

7.5 Chapter Summary

We proposed an idea to accelerate EC search by deleting poor individuals and generating their alternatives around the best individuals. In this chapter, we observe the convergence characteristics and a distribution of population by changing parameters of the rule. Through the observations, we (1) found that the proposed method accelerated DE well for almost tasks and did not become worse than normal DE for other few cases, (2) found the relationship between parameters of the proposed method and their performance, (3) pointed out the possibility that we can further improve an acceleration performance by optimizing the parameters and observing convergence with the optimized parameters, and (4) may be able to obtain new findings that can improve acceleration performance from the observation of distribution of population.

We would like to continue to analyze the behaviors as described in section 7.4 since there are possibilities of obtaining hints to further improve our proposed method and new acceleration approaches.

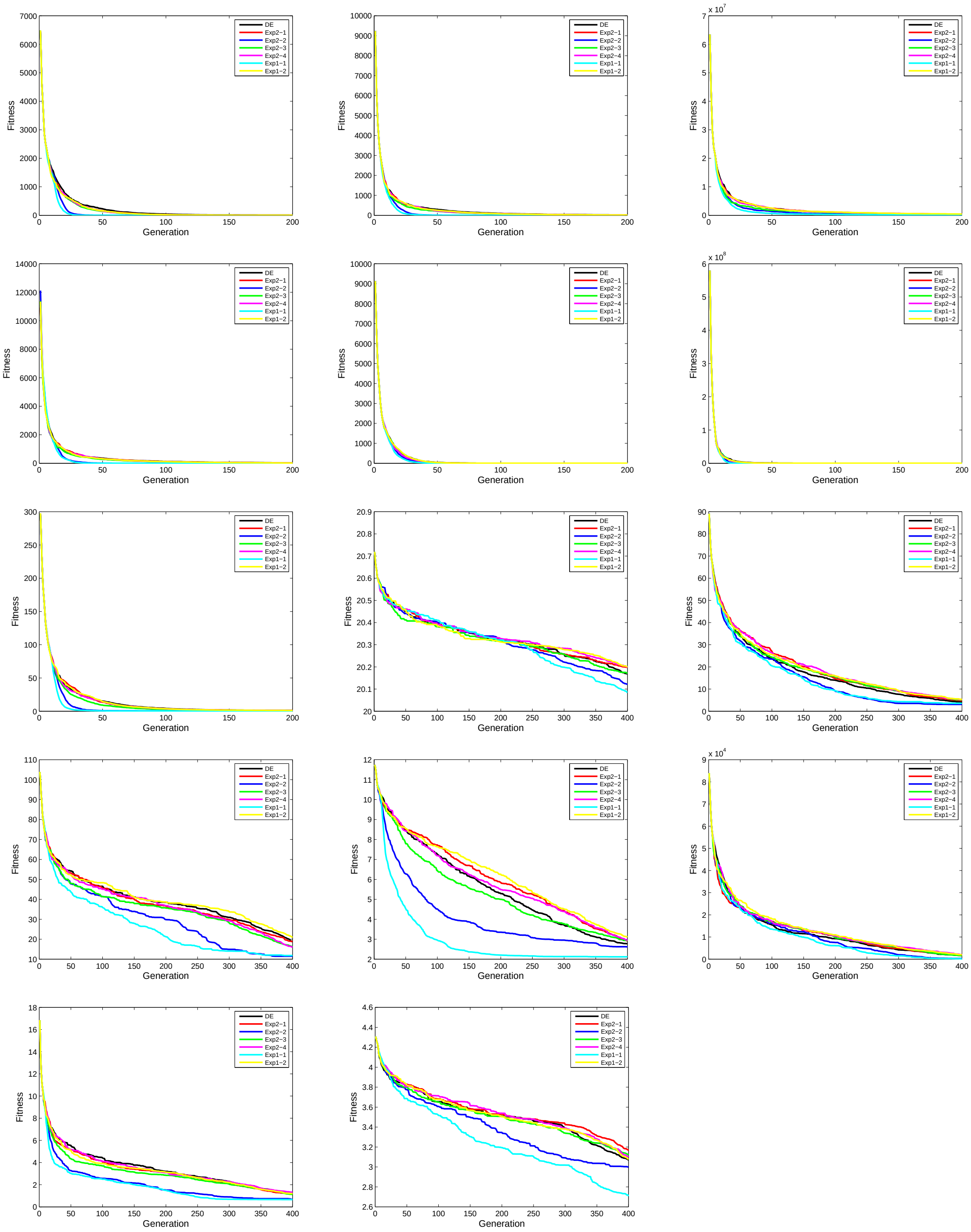


Figure 7.2: Convergence characteristics of normal DE and (DE+proposed method); search ranges in Table 7.2 are used. See Table 7.5 for Exp numbers in these graphs. Used benchmark functions are $F_1 - F_3$, $F_4 - F_6$, $F_6 - F_9$, $F_{10} - F_{12}$, and F_{13} and F_{14} from left to right and from the top to the bottom.

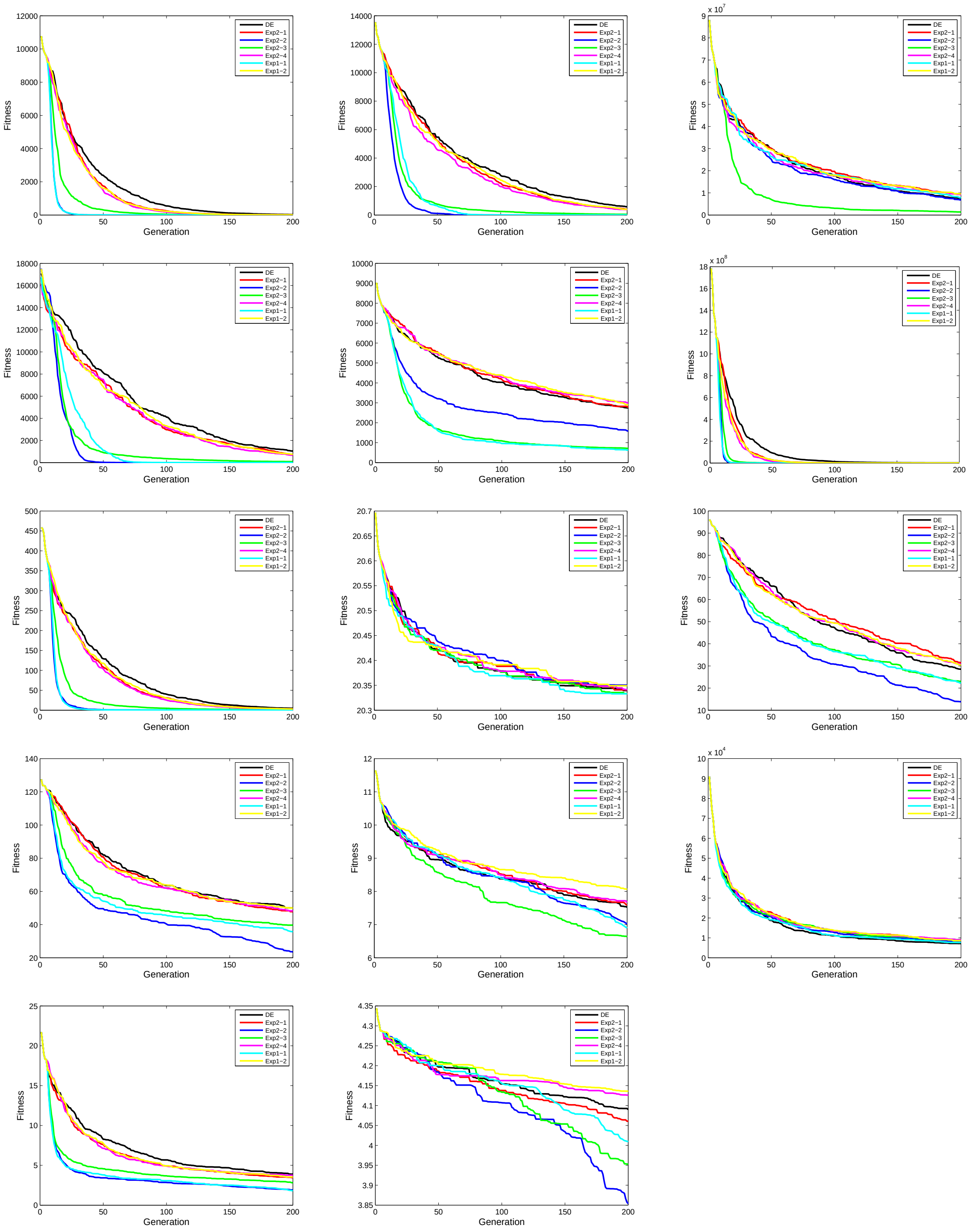


Figure 7.3: Convergence characteristics of normal DE and (DE+proposed method); search ranges in Table 7.2 are NOT used except initialization. See Table 7.5 for Exp numbers in these graphs. Used benchmark functions are $F_1 - F_3$, $F_4 - F_6$, $F_6 - F_9$, $F_{10} - F_{12}$, and F_{13} and F_{14} from left to right and from the top to the bottom.

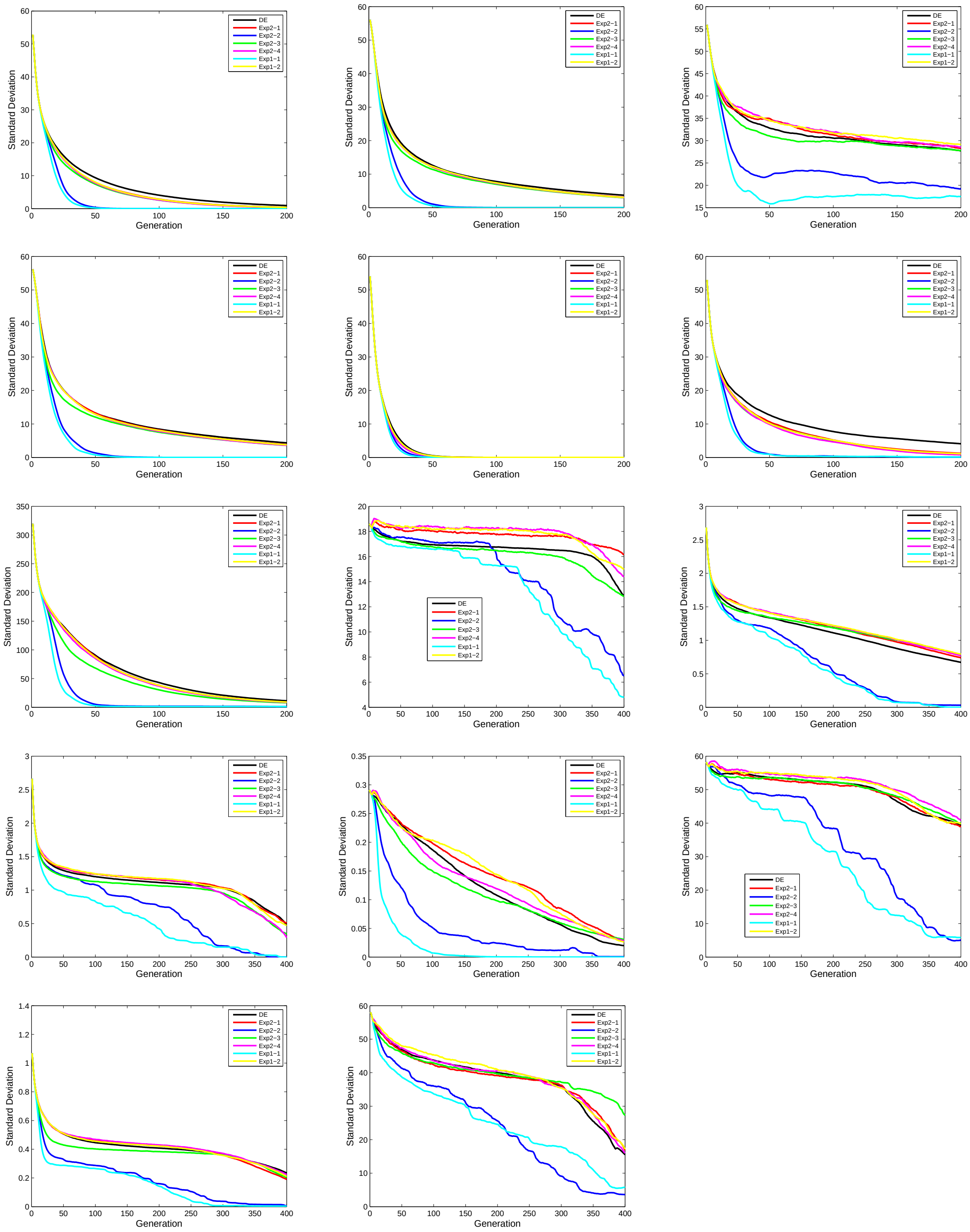


Figure 7.4: Transitions of standard deviations of all individuals when parameters of the proposed method are changed. See Table 7.5 for Exp numbers in these graphs. Used benchmark functions are $F_1 - F_3$, $F_4 - F_6$, $F_6 - F_9$, $F_{10} - F_{12}$, and F_{13} and F_{14} from left to right and from the top to the bottom.

Chapter 8

Chaotic Evolution: A New EC Algorithm and Framework

8.1 Introduction

Conventional EC algorithms simulate biological phenomena or behaviors to implement a biological process for optimization. These algorithms encompass two components in their search mechanism. One is a search method by a variety of implementations, and the other is an iterative process. Most of EC algorithms do not require the optimization problem to have some specific characteristics, and few of them do not use any mathematical characteristics or mechanisms to make sure algorithm convergence. If we introduce mathematical property or mechanism, such as chaotic ergodicity, into an optimization iteration, we may be able to implement new EC algorithms that make sure their convergence partially. It helps to improve the global convergence characteristics of an algorithm.

In this chapter, we propose a novel EC algorithm that fuses an evolutionary iteration of EC and an ergodic property of chaos. We call it as *Chaotic Evolution* (CE). The CE adopts a mutation operation with simulating chaotic behaviour of every individual in each dimension. Only when fitness of offspring is better than that of its parent, the parent will be replaced. To direct search directions of an individual in each dimension, a new control parameter, direction factor rate (DR) is proposed. Compared with differential evolution (DE), our proposed CE algorithm can optimize most of the benchmark problems with a higher convergence speed and better final solution quality.

For properties of a great practical optimization algorithm, CE has its robust search capability, parallelizability, simplicity and consistent convergence. First, CE is designed as a stochastic directed search method. The stochastic directed search is implemented by simulating chaotic motion with the new parameter DR, which can be set to a certain rate, a random rate or an adaptive rate, etc. The design of stochastic directed search and the simulation of chaotic ergodic motion ensure robust search capability of the proposal. Second, CE uses a population where chaotic perturbation of the population can be conducted independently to implement parallelizability of the proposal. The perturbation is conducted by every individual in a chaotic way. Third, to optimize a certain system, the conventional EC algorithm must select pertinent algorithm parameters. However, there are population size (PS), chaotic system parameter (CP), crossover rate (Cr) and direction rate (DR) in

CE. Cr can be set to 100% and DR can be randomly set within the range of $(0, 1]$, so only PS and CP need to be considered. These characteristics ensure the proposal's simplicity. The originality of this work is not only to introduce a new EC algorithm by simulating chaotic ergodic motion for optimization, but also to propose a new algorithm design philosophy that is to implement a mathematical principle (i.e. chaotic ergodicity) into a search algorithm to make sure the convergence of an algorithm.

Following this introductory section, an overview of chaos theory used in EC is reported in section 8.2. We introduce the novel EC algorithm, *Chaotic Evolution* in section 8.3. In section 8.4, experimental evaluations are performed using 12 benchmark functions, the results are compared with DE. Finally, we discuss our proposed CE and obtained evaluation results in section 8.5. In section 8.6, we conclude the whole work and some open topics, further opportunities and future works are presented.

8.2 Chaos Theory Meets Evolutionary Computation

EC is a field of computational intelligence. It can solve optimization problems that are difficult to obtain the optimum by using conventional methods. Chaos theory is a tool and method for describing a nonlinear system, which has many great properties, such as chaotic ergodicity. Evolution and chaos have more same features in common. First, the two words, evolution and chaos, are used to refer simultaneously both to phenomena that need to be explained and to theories that are supported to do the explanation [101]. Second, there must be an iterative process in both evolution and chaos. Third, the concept of evolution contributes to discussion of chaos and vice versa.

There are three directions in current research that chaos is used in EC. They are local search method, parameter tuning mechanism and new EC algorithm inspiration. Chaos is used as a local search method in the memetic algorithm that benefits from the local search. A chaos search method is used in memetic algorithm to improve the performance of multi-objective optimization [71]. A chaotic system is used as a generator to replace the random one in conventional EC [33, 49, 54]. Reference [10] compares optimization performance of random and chaotic generators by benchmark functions in GA. Numerous examples and statistical tests show an improvement of GA, when chaotic generators are used to replace the random one. The scale factor of DE is tuned by the logistic map to improve its performance [16, 48]. Some new EC algorithms are inspired by the chaotic property and phenomenon [37, 73].

Algorithm 1 Chaotic Evolution. PS: population size; Dim: dimension; D: direction; DR: direction rate; CP: chaotic parameter; G: generation; maxIter: maximum generation.

```
Generate an initial population.
Evaluate the fitness for each individual.
/*D and CP initialization*/
for $i = 1$ to $PS$ do
    for $j = 1$ to $Dim$ do
        /*DR as a random value*/
        if $rand[0,1) < DR$ then
            $D_{i,j} = -1$
        else
            $D_{i,j} = +1$
        end if
        $CP_{i,j}$ = rand(0,1)
    end for
end for
/*Chaotic Search*/
for $G = 1$ to $maxIter$ do
    for $i = 1$ to $PS$ do
        $k$=rand(1,Dim)
        for $j = 1$ to $Dim$ do
            if $rand[0,1) < C_r$ or $j == k$ then
                $mutant_{i,j} = target_{i,j} * (1 + D_{i,j} * CP_{i,j})$
                $chaotic_{i,j} = mutant_{i,j}$
            else
                $chaotic_{i,j} = target_{i,j}$
            end if
        end for
        /*Selection*/
        for $i = 1$ to $PS$ do
            if $f(chaotic_i) < f(target_i)$ then
                replace $target_i$ with $chaotic_i$
            end if
        end for
    end for
    /*D and CP update*/
    for $i = 1$ to $PS$ do
        for $j = 1$ to $Dim$ do
            $CP_{i,j} = ChaoticSystem(CP_{i,j})$
            if $rand[0,1) < DR$ then
                $D_{i,j} = -1$
            else
                $D_{i,j} = +1$
            end if
        end for
    end for
end for
return the optimum
```

8.3 A Chaotic Ergodicity Based Evolutionary Algorithm

8.3.1 Concept of the Proposal

Exploitation and exploration functions of DE depend on differential vector production, population distribution, crossover rate and the scale factor parameter setting. It is easy to fall into premature convergence and the evolution speed becomes slow, when it has a lack of population diversity. This is one of the drawbacks of DE. Several strategies for improving DE performance have been proposed, such as SaDE [88], JADE [89], jDE [7] and JASaDE [25]. Their strategies include: setting DE parameters randomly [7], choosing better strategies from search histories [88], creating new mutation and crossover strategy [89], and fusing those strategies together [25]. Approaches for improving DE performance are roughly categorized into three: parameters tuning methods, strategy setting methods, i.e. strategy pool architecture, and strategy selection methods.

We propose to use chaotic vector to solve the low population diversity problem mentioned above and establish a new EC search framework and an optimization principle by simulating chaotic ergodic motion in a search space. We call this novel population-based algorithm as *Chaotic Evolution* (CE). The principle of CE is to simulate chaotic ergodic motion for implementing search. CE generates a mutant vector from a target vector with a chaotic system and a chaotic vector by crossing the mutant vector and the target vector in its algorithm framework. CE reserves the improved chaotic vector into the next generation by greedy selection method. There are three new concepts, i.e., direction factor, chaotic parameter and chaotic vector in CE.

Suppose there is a best point in a closed bottle (a global optimum in a search space), an air molecule (an individual) randomly moves in the closed bottle for a long time. Because of the molecular motion with ergodic property, it may visit the best point in the closed bottle at least once (finding the global optimum). In the process, the motion with ergodic property guarantees the search convergence. This is an ideal model and simple explanation of the optimization process of our proposal.

Table 8.1: 12 benchmark functions used in experimental evaluations. (Uni=Unimodal, Mul=Multimodal, GB=Global on Bounds, NS=non-separable, and S=separable.)

No.	Name	Form	Range	Optimum	Characters
F1	Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$	[-100,100]	-450	Uni-S
F2	Schwefel 1.2	$f(x) = \sum_{i=1}^{n}(\sum_{j=1}^{i}(x_j))^2$	[-100,100]	-450	Uni-NS
F3	Elliptic	$f(x) = \sum_{i=1}^{n}(10^6)^{\frac{i-1}{n-1}} x_i^2$	[-100,100]	-450	Uni-NS
F4	F2 with Noise	$f(x) = \sum_{i=1}^{n}(\sum_{j=1}^{i}(x_j))^2 * (1 + 0.4\vert N(0,1)\vert$	[-100,100]	-450	Uni-NS
F5	Schwefel 2.6 GB	$f(x) = Max\vert A_i x - B_i\vert$	[-100,100]	-310	Uni-NS
F6	Rosenbrock	$f(x) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2$	[-100,100]	390	Mul-NS
F7	Griewank	$f(x) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}})$	[0,600]	-180	Mul-NS
F8	Ackley GB	$f(x) = -20\exp(-0.2(\frac{1}{n}\sum_{i=1}^{n} x^2)^{\frac{1}{2}}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$	[-32,32]	-140	Mul-NS
F9	Rastrigin	$f(x) = (10n) + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$	[-5,5]	-330	Mul-S
F10	Weierstrass	$f(x) = \sum_{i=1}^{n}(\sum_{k=0}^{kmax}[a^k\cos(2\pi b^k(x_i + 0.5))] - n\sum_{k=0}^{kmax}[a^k\cos(2\pi b^k * 0.5)])$	[-0.5,0.5]	90	Mul-NS
F11	Schwefel 2.13	$f(x) = \sum_{i=1}^{n}(\sum_{j=1}^{n}(a_{ij}\sin x_i + b_{ij}\cos x_i))^2$	$[-100, 100]$	-460	Mul-NS
F12	Expanded F8F2	$f(x) = F8F2(x_1, x_2, ..., x_n) = F8(F2(x_1, x_2) + ... + F8(F2(x_n, x_1))$	[-3,1]	-130	Mul-NS

8.3.1.1 Chaotic Vector

Generating a chaotic vector is the crucial operation in CE. A mutant vector ($mutant_i$) is generated from a target vector ($target_i$) by simulating the chaotic motion in a search space (Eq. (8.1)). A chaotic vector ($chaotic_i$) is generated by crossing the target vector and the mutant vector (Eq. (8.2)). In Eq. (8.1), D_i is the direction factor, which value is either $+1$ or -1; CP_i is the chaotic parameter, which is initially set to random value within the range of $(0, 1]$ and is updated every generation by a chaotic system. A chaotic vector implements the actual search function of CE.

$$mutant_i = target_i * (1 + D_i * CP_i) \tag{8.1}$$

$$chaotic_i = Crossover(target_i; mutant_i) \tag{8.2}$$

There are two strategies to generate a chaotic vector, one is with the crossover operator, on the contrary, the other is without it (i.e. $mutant_i = chaotic_i$). It means we have two CE implementation methods, i.e., with and without a crossover operation.

8.3.1.2 Direction Factor

The direction factor (D_i) is a new control parameter in our proposal. It decides the search direction of every individual in each dimension. We define a direction factor rate (DR) to control the D_i percentage of positive and negative values. For example, if the $DR = 0.9$, it means 90% of D_i is $+1$ and 10% of D_i is -1, when generating a mutant vector from a target vector.

In this chapter, we set the DR as a random value, however, the direction factor rate setting method is not limited to that. A static rate or an adaptive rate according to the search condition, or other setting methods, can be as well as used by considering the output distribution of a chaotic system.

Although it is difficult to make a proper setting of DR, CE keeps the DR concept for its further investigation in two perspectives. One is that we can obtain some potential knowledge about the optimized problems from the statistical results of DR. The other is that we can develop a variety of CE algorithms by considering the different DR setting.

8.3.1.3 Chaotic Parameter from Chaotic System

The chaotic parameter (CP) from a chaotic system decides the CE search range in a search space. It is set initially to random value within range of $(0, 1]$, and is updated by a chaotic system in every generation. When the output of a chaotic system is beyond the range of $(0, 1]$, we should consider to project this value within $(0, 1]$. Chaotic parameter value influences the exploration and exploitation functions of CE algorithm.

We use the logistic map (Eq. (2.25)) as the chaotic system to update the chaotic parameter. Actually, the rule for updating the CP is not limited to the logistic map.

All the nonlinear systems that can show the chaotic output characteristic can be as the update rule. This strategy shows the scalability of our proposal. It is another original contribution of this method.

8.3.2 Proposed Algorithm

As mentioned above, our proposed CE algorithm is shown in Algorithm 1. The following description outlines the main steps of a CE implementation.

1. Line 1 is for generating initial population and line 2 is for evaluating for each individual of the population.
2. From lines 4 to 14, chaotic parameter (CP) and search direction (D) are initialized with $rand(0, 1]$ and $\{+1, -1\}$, respectively.
3. It is the main search part of CE algorithm from lines 16 to 33. Line 21 shows the mutant vector production. From the lines 19 to 26, it is the crossover and mutation operations of CE. It is the selection operation from line 28 to 32, and only a new generated chaotic vector needs a fitness evaluation.
4. Lines 35 to 44 shows the process of updating parameters CP and D in each generation.
5. Line 46 returns the optimum.

8.4 Experimental Evaluation

8.4.1 Experimental Conditions

To investigate our proposals' performance, we evaluate CE using 12 benchmark functions comparing with DE. The definition of the benchmark functions, range, the global optimum and characteristics are listed in Table 8.1. Some of them are adopted from [104].

We test each benchmark function up to 10^5 generations with 50 trial runs. Figures 8.1 and 8.2 show the average convergence curves of the best fitness values of 50 trial runs for all 12 benchmark functions. Table 8.2 shows their means and variances, the bold font shows that our proposed chaotic evolution is significantly better than DE by Wilcoxon signed rank test ($p < 0.05$).

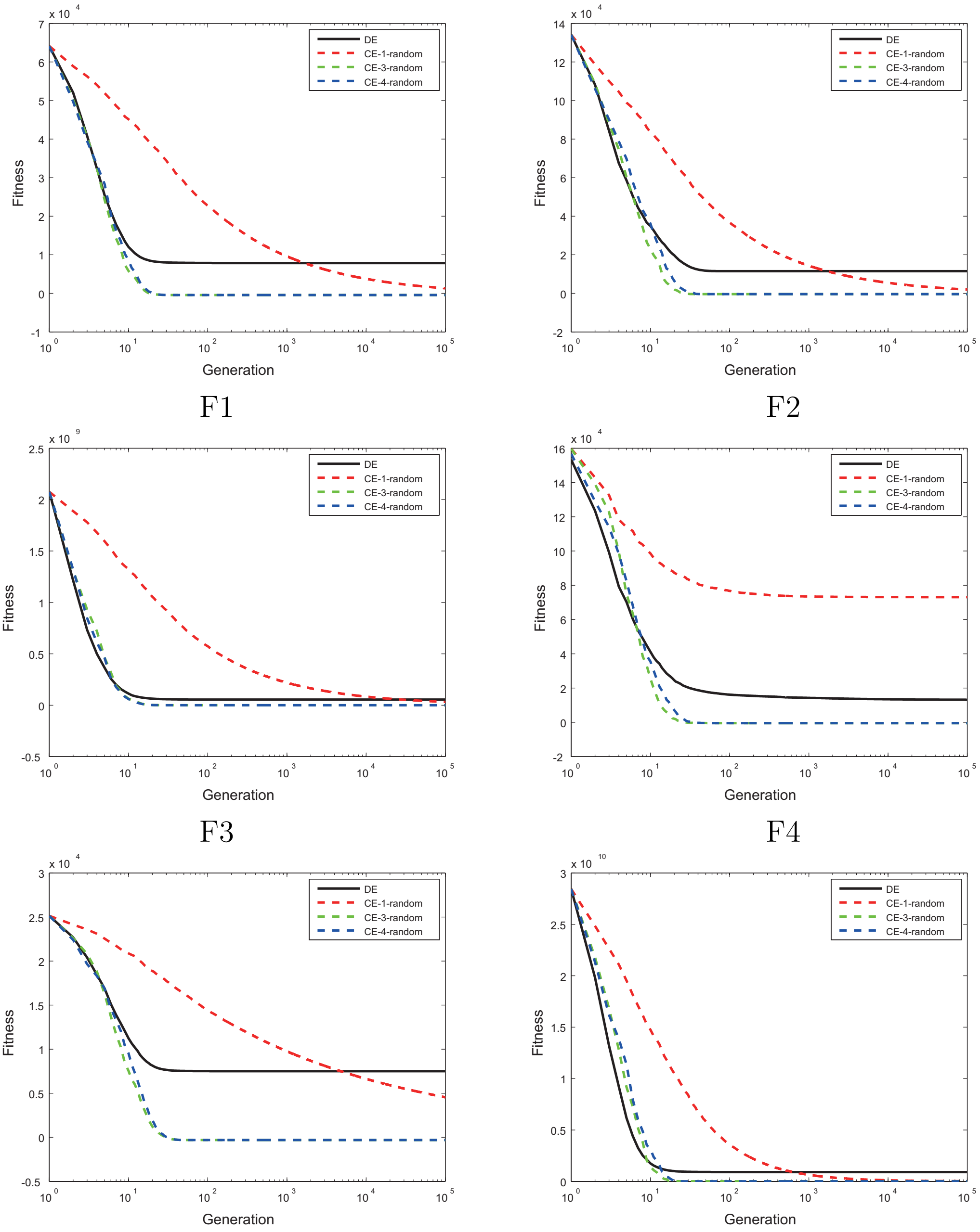
To investigate the chaotic evolution optimization performance by the same chaotic system setting with the different system output behaviours, we set the parameter of Eq. (2.25) as $\mu = 1$, 3, and 4, i.e., convergence output, analogical chaotic output and chaotic output. Abbreviations used in Figures 8.1, 8.2 and Table 8.2 are given in Table 8.3.

Experimental parameters are set in the Table 8.4. Those value were used as better DE parameter setting for some of the benchmarks in [81]. The evaluation is conducted under a hard search condition; only 50 individuals search 30-D functions.

Table 8.2: Means at the 10^5th generation of 50 trail runs of 12 benchmark functions. The bold font numbers show that CE is better than DE by Wilcoxon signed rank test ($p < 0.05$).

Methods	F1	F2	F3	F4	F5	F6
DE	7.85E+03	1.15E+04	5.45E+07	1.32E+04	7.50E+03	8.97E+08
CE-1-01	**-4.35E+02**	**-2.10E+02**	**1.47E+05**	**4.07E+04**	**-1.68E+02**	**3.32E+05**
CE-3-01	**-4.47E+02**	**-4.45E+02**	**5.65E+03**	**-4.48E+02**	**-2.89E+02**	**5.98E+03**
CE-4-01	**-4.48E+02**	**-4.47E+02**	**4.08E+03**	**-4.42E+02**	**-2.90E+02**	**1.23E+04**
CE-1-03	3.53E+04	6.45E+04	3.37E+08	8.01E+04	1.91E+04	1.17E+10
CE-3-03	2.57E+04	4.99E+04	1.59E+08	6.16E+04	1.64E+04	8.88E+09
CE-4-03	2.56E+04	4.54E+04	1.16E+08	4.99E+04	1.52E+04	8.82E+09
CE-1-05	5.56E+04	9.32E+04	1.08E+09	1.09E+05	2.27E+04	2.16E+10
CE-3-05	4.87E+04	8.70E+04	7.56E+08	1.02E+05	2.20E+04	1.86E+10
CE-4-05	4.70E+04	7.95E+04	6.63E+08	9.01E+04	2.08E+04	1.81E+10
CE-1-07	6.32E+04	1.06E+05	1.66E+09	1.26E+05	2.41E+04	2.72E+10
CE-3-07	6.06E+04	1.08E+05	1.44E+09	1.30E+05	2.39E+04	2.61E+10
CE-4-07	5.87E+04	1.05E+05	1.32E+09	1.17E+05	2.30E+04	2.48E+10
CE-1-09	6.41E+04	1.23E+05	1.99E+09	1.38E+05	2.46E+04	2.84E+10
CE-3-09	6.39E+04	1.25E+05	1.93E+09	1.42E+05	2.47E+04	2.81E+10
CE-4-09	6.34E+04	1.17E+05	1.88E+09	1.33E+05	2.38E+04	2.76E+10
CE-1-random	**1.30E+03**	**1.94E+03**	**3.23E+07**	7.31E+04	**4.55E+03**	**1.82E+07**
CE-3-random	**-4.50E+02**	**-4.50E+02**	**-4.50E+02**	**-4.50E+02**	**-3.10E+02**	**4.18E+02**
CE-4-random	**-4.50E+02**	**-4.50E+02**	**-4.50E+02**	**-4.50E+02**	**-3.10E+02**	**4.18E+02**

Methods	F7	F8	F9	F10	F11	F12
DE	-1.04E+02	-1.26E+02	-2.25E+02	1.12E+02	1.54E+05	4.77E+01
CE-1-01	**-1.79E+02**	**-1.32E+02**	-4.20E+01	1.17E+02	1.58E+06	**-8.06E+01**
CE-3-01	**-1.80E+02**	**-1.40E+02**	**-3.30E+02**	**9.00E+01**	1.03E+06	**-1.16E+02**
CE-4-01	**-1.80E+02**	**-1.40E+02**	**-3.30E+02**	**9.00E+01**	6.70E+05	**-1.16E+02**
CE-1-03	1.42E+02	-1.20E+02	2.05E+01	1.26E+02	1.54E+06	2.82E+03
CE-3-03	5.61E+01	-1.22E+02	-1.24E+02	9.10E+01	7.26E+05	1.78E+03
CE-4-03	5.50E+01	-1.23E+02	-1.77E+02	9.01E+01	5.17E+05	1.70E+03
CE-1-05	3.25E+02	-1.20E+02	4.91E+01	1.31E+02	1.53E+06	9.34E+03
CE-3-05	2.63E+02	-1.20E+02	3.45E+00	9.58E+01	6.01E+05	6.48E+03
CE-4-05	2.48E+02	-1.20E+02	-5.41E+01	9.30E+01	4.65E+05	6.50E+03
CE-1-07	3.94E+02	-1.20E+02	6.81E+01	1.37E+02	1.52E+06	1.40E+04
CE-3-07	3.71E+02	-1.20E+02	7.61E+01	1.05E+02	6.15E+05	1.18E+04
CE-4-07	3.54E+02	-1.20E+02	3.16E+01	1.01E+02	4.91E+05	1.12E+04
CE-1-09	4.02E+02	-1.20E+02	7.48E+01	1.39E+02	1.53E+06	1.67E+04
CE-3-09	4.00E+02	-1.20E+02	8.80E+01	1.19E+02	6.81E+05	1.60E+04
CE-4-09	3.96E+02	-1.20E+02	7.25E+01	1.11E+02	5.45E+05	1.57E+04
CE-1-random	**-1.63E+02**	-1.22E+02	-1.82E+02	1.30E+02	**-1.63E+02**	6.95E+01
CE-3-random	**-1.80E+02**	**-1.40E+02**	**-3.30E+02**	**9.00E+01**	6.33E+05	**-1.17E+02**
CE-4-random	**-1.80E+02**	**-1.40E+02**	**-3.30E+02**	**9.00E+01**	2.71E+05	**-1.18E+02**



F1

F2

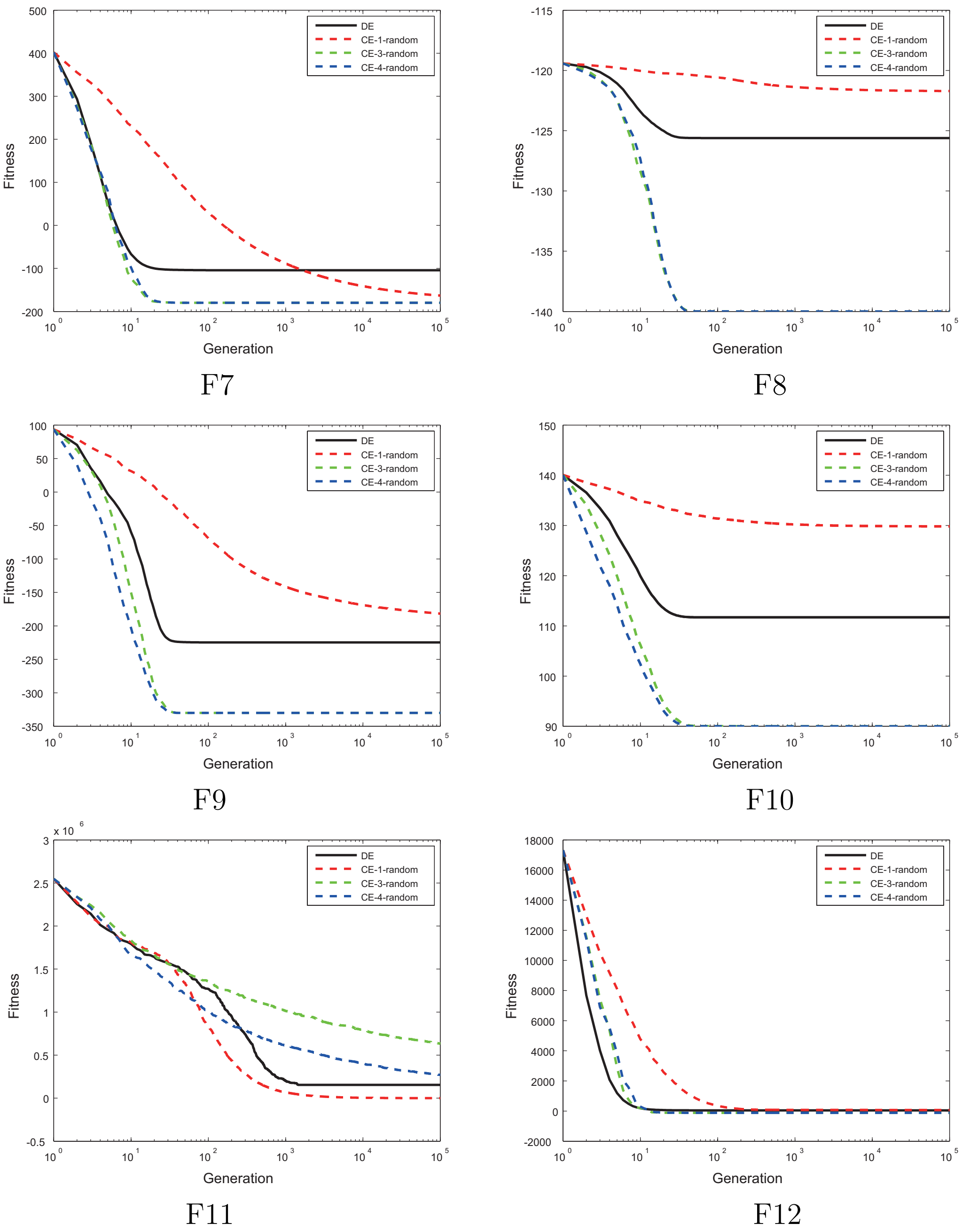
F3

F4

F5

F6

Figure 8.1: Average convergence curves of 50 trial runs for F1-F6.



F7

F8

F9

F10

F11

F12

Figure 8.2: Average convergence curves of 50 trial runs for F7-F12.

Table 8.3: Abbreviations used in Figures 8.1 and 8.2, and Table 8.2.

abbreviations	Meaning
DE	standard DE
CE-1	chaotic evolution with CP updated by Eq. (2.25), $\mu = 1$
CE-3	chaotic evolution with CP updated by Eq. (2.25), $\mu = 3$
CE-4	chaotic evolution with CP updated by Eq. (2.25), $\mu = 4$

Table 8.4: DE and CE experiment parameters setting.

population size	50
max. search generation	10^5
dimensions of benchmark functions, D	30
# of trial runs	50
DE	DE/best/1/bin
scale factor F	0.3
crossover rate	0.7
CE direction factor rate DR	0.1,0.3,0.5,0.7,0.9,random
CE crossover rate	0.7,1.0

8.4.2 Experimental Results

From Figures 8.1 and 8.2, and Table 8.2, we can conclude as follows: (1) CE-3 and CE-4 methods are better than CE-1 method, (2) CE-3 and CE-4 performances are almost the same for most of benchmark functions, and (3) our proposed CE is significantly better than DE in the final results, except by all methods in F11, and the CE-1 method in F4, F9 and F10.

8.5 Discussions

8.5.1 Chaotic Evolution Work Mechanism

Combination of chaotic ergodicity and evolutionary iteration is a characteristic of CE algorithm's optimization principle and framework. There are three techniques to support CE works well. First, the mutation operation of CE is based on a chaotic system. That means CE optimization performance highly depends on the chaotic system output and its distribution. The ergodicity supports that the CE individuals can visit any locations of a search space, which makes sure the convergence of CE algorithm. Second, a parent individual is replaced by its offspring only if the fitness of offspring is better than that of its parent. It is a greedy criterion. So the best solution is kept and the whole population is towards to the better evolution. However, other selection methods can be as well as used in CE. Third, we introduce a new control parameter, DR, to guide its search direction for every individual in each dimension. CE optimization performance also depends on the DR setting.

The offspring of CE (chaotic vector) is generated by crossing the target vector and the mutant vector, and the mutant vector depends on a chaotic system. CE optimization performance depends on the chaotic parameter from a chaotic system in the view points of global exploration and local exploitation. When its value is larger, the chaotic vector searches beyond the target vector and conducts the global exploration. On the contrary, when its value is smaller, the chaotic vector searches around the target vector and conducts the local exploitation. However, the offspring of DE depends on the base vector and the differential vector, which comes from two random vectors. Its search performance of global exploration and local exploitation depends on the population distribution. This is the main mechanistic difference between CE and DE.

8.5.2 Optimization Performance of Chaotic Evolution

The total number of fitness evaluations in CE is the same as that in DE. CE algorithm calls a fitness function at two places. One is after population initialization, and all individuals need to be evaluated by a fitness function. The other is when the chaotic vector is compared with the target vector; the chaotic vector needs to be evaluated by a fitness function (Algorithm 1).

Our evaluation results have shown that our proposed CE outperformed DE except some cases. First, as these results were obtained from only 12 benchmark functions, it cannot strongly prove that CE must outperform DE though the best performance was obtained at least for the 11 benchmark functions. Second, the CE-3 and CE-4 methods were better than the CE-1 methods, which shows that the logistic map with the chaotic outputs and the analogical chaotic outputs can obtain better optimization performances.

Table 8.5: Distribution characteristics of a logistic map ($\mu = 4$), a quadratic-like distribution ($N(0, 0.15^2) + N(1, 0.15^2)$) and a uniform distribution.

Interval	logistic	quadratic-like	uniform
(0, 0.1]	18.60%	22.80%	11.20%
(0.1, 0.2]	7.50%	16.80%	9.70%
(0.2, 0.3]	8.60%	6.10%	9.60%
(0.3, 0.4]	7.70%	2.30%	9.30%
(0.4, 0.5]	5.10%	0.40%	11.30%
(0.5, 0.6]	6.20%	0.80%	11.80%
(0.6, 0.7]	7.00%	1.60%	9.80%
(0.7, 0.8]	8.90%	6.70%	9.10%
(0.8, 0.9]	9.70%	15.90%	9.40%
(0.9, 1]	20.70%	26.60%	8.80%

We conduct an additional experiment to investigate the relationship between an output distribution of a chaotic system and CE optimization performance. The output distribution of the logistic map has a characteristic that most of the outputs cover the intervals $[0, 0.1]$ and $[0.9, 1]$ (Figure 8.5). From the exploration and exploitation viewpoint, this characteristic decides the CE's exploration and exploitation functions. To confirm this hypothesis, we compare the optimization performance by CE with chaotic parameter CP from a logistic map ($\mu = 4$), a quadratic-like distribution ($N(0, 0.15^2) + N(1, 0.15^2)$) and a uniform distribution. We abbreviate the CE algorithms with CP set by these distribution as CE-logistic, CE-quadratic and CE-uniform, respectively.

The experiment uses 10-D functions from Table 8.1 with 50 individuals, 100 generations and 30 running trails. Because most of the benchmark functions obtain the global optimum by the 100th generation, we conduct as Wilxocon signed rank test at the 10th generation. Table 8.6 shows means of 12 benchmark functions and results of Wilxocon signed rank test ($p < 0.05$). Most of the CE-logistic method significantly outperforms CE-uniform, and significant difference could not found between CE-logistic method and CE-quadratic method. It indicates that output distribution characteristic of the logistic map decides optimization performance of CE algorithm. CE-quadratic and CE-uniform outperform DE in some benchmark tasks, it indicates that evolution strategy-like algorithm (CE-quadratic and CE-uniform) can obtain better optimization performance.

Table 8.6: Means of CE algorithm with CP generated by a logistic map ($\mu = 4$) (CE-logistic), a quadratic distribution ($N(0, 0.15^2) + N(1, 0.15^2)$) (CE-quadratic), a uniform distribution (CE-uniform) and DE. The marks ‡ and ‡ mean CE-logistic method significantly outperforms CE-quadratic and CE-uniform, respectively, by Wilcoxon signed test ($p < 0.05$). The marks △ and □ mean CE-quadratic and CE-uniform outperform DE, respectively, by Wilcoxon signed test ($p < 0.05$).

Fucntion	CE-logistic	CE-quadratic	CE-uniform	DE
F1	-342.18 ‡	-134.26	-74.69	-127.17
F2	20.41 ‡	340.60 △	547.68 □	934.95
F3	234550.49 ‡	446507.86 △	1485224.95	2223488.16
F4	-13.31†‡	107.20△	299.30 □	1288.93
F5	335.98 ‡	865.73 △	1146.03 □	1641.78
F6	1950824.34	2589855.58 △	4794977.98	3731426.17
F7	-178.12 ‡	-176.27△	-175.62	-176.09
F8	-135.68 ‡	-134.63△	-132.87	-133.00
F9	-320.12 ‡	-316.09△	-306.51 □	-280.64
F10	92.09 ‡	92.94 △	93.81 □	94.90
F11	85301.58	91033.36	88215.97	89324.51
F12	-124.42 ‡	-124.61	-123.47	-124.01

8.5.3 Chaotic Evolution Parameters

In CE, there are some parameters and a chaotic system should be set and selected; one is direction factor rate, one is crossover rate, and one is a chaotic system equation. Those settings decide the optimization performance of CE.

Direction factor rate is a new control parameter in our proposal, and it guides the search direction in the whole search space. From the Table 8.2, the proposed methods with $DR = 0.1$ and $DR = random$ can significantly better than DE. The optimum of benchmark functions in our evaluation are vector $\bar{0}$, i.e., $\{0, 0..., 0\}$. We set DR to 0.1, 0.3, 0.5, 0.7, 0.9, and *random*, which means that the 10, 30, 50, 70, 90 and *random* percent values in the whole dimension of an offspring will become smaller than its parent in one generation. This is the reason why CE performance with the less DR value setting becomes better.

From the observation above, the random direction rate is a robust setting for all the benchmark tasks. However, for an unknown problem, which DR value is better, it should depend on the landscape and problem complexity. We need to investigate this issue in the future. Using an adaptive DR setting is also a valuable study in our future work.

Crossover rate borrows from the DE, and whether it is a necessary parameter for CE is an issue of further investigation. Crossover operation is borrowed from GA originally, and its work principle is to simulate crossover phenomenon of the biological gene. However, investigation of crossover operation for CE is a valuable topic for deeply and better understanding the working mechanism of CE.

Chaotic system selection shows the scalability of chaotic evolution algorithm. Chaotic system selection is a key step to make a better CE algorithm, and the chaotic system should be with the following properties. First, it must be with the chaotic output or analogy the chaotic output. Second, it must be easy to obtain its output, i.e., with the less computational complexity. The different chaotic system selection must result in the different optimization performance. It is a valuable work to study which chaotic system can obtain the best optimization performance to a certain landscape, or a category of benchmarks, or real world applications. Our future work will involve this topic.

8.6 Chapter Summary

In this chapter, a new EC algorithm, *Chaotic Evolution* has been introduced and investigated preliminarily. A chaotic vector is generated from a chaotic system to implement exploration and exploitation functions of CE algorithm based on the chaotic ergodic property. By introducing this property into an evolutionary iterative optimization process, some new control parameters and operating principles of CE algorithm were proposed, analyzed and studied.

The main motivation for the current work is introducing a mathematical mechanism (chaotic ergodic property) into an evolutionary optimization process to design a new EC algorithm. Our proposed CE outperforms DE and some of its variants in the convergence speed and solution quality from the experimental results. Direction factor rate, crossover rate, chaotic system and its parameter are the related parameter setting problems that we have discussed in this chapter. Further studies are still required to investigate their benefits, weaknesses, and limitations. In the search mechanism of CE, individuals search for the optimum independently. That means CE algorithm does not use the population information, such as population distribution, fitness landscape, etc., to enhance its optimization performance. We will consider this drawback to improve CE performance in the future.

The main contribution of this chapter is establishing a novel search framework that fuses the chaotic ergodicity and evolutionary iteration. We point out that the chaos theory is used not only to describe and explain a nonlinear system, but also to implement a variety of optimization algorithms based on its ergodic property. Possible directions for future work include chaotic system selection, setting method investigation of the direction factor rate and theoretical study of crossover rate necessity, etc. We will conduct these works in the future.

Chapter 9

Conclusion and Future Works

9.1 Main Contributions

Enhancing and improving search capability and optimization performance of evolutionary computation (EC) algorithms is a promising research subject in EC community. In this dissertation, we proposed six novel approaches on this subject within three research directions and discussed them. They include:

1. approximating fitness landscape in lower dimensional search space and elite local search,
2. Fourier analysis on fitness landscape and its enhancement methods,
3. Fourier niche method for multimodal optimization,
4. triple and quadruple comparison based interactive differential evolution (IDE) and differential evolution (DE),
5. EC acceleration by accelerating transition from exploration to exploitation, and
6. a new EC algorithm – chaotic evolution.

Fitness landscape approximation, new search mechanism development and new EC algorithm design are primary three research directions in this dissertation.

The fitness landscape approximation method tries to obtain knowledge of problem structure and EC search condition in a search space. Once we obtain these kinds of information, we can design special search strategies and local search methods to enhance search capability of EC algorithms. In this dissertation, we originally proposed two methods to achieve this objective. The one is to approximate fitness landscape in a lower dimensional search space, the other is to use Fourier analysis to obtain frequency information in a search space for approximating fitness landscape. A novel niche method for multimodal optimization, Fourier niche method, was proposed and initially investigated by the same principle of obtaining frequency information in a search space. The philosophy of these two approaches presents originality and innovation of this dissertation. Proposed methods and search strategies are not limited within any special EC algorithm, but it is general to all of them. This is also a feature of this work.

Developing a new search mechanism is the second research direction of this dissertation. Since user fatigue is a serious issue of applying interactive evolutionary computation (IEC), reducing the user fatigue is a practical requirement for its applications. In this dissertation, we developed two new comparison mechanism in canonical DE algorithm, we call the first one as a triple and quadruple comparison method that can not only improve IDE performance as well as reducing user fatigue, but also DE search capability. The second one is an EC acceleration method by accelerating transition from exploration to exploitation by monitoring inactive poor individuals. These two acceleration methods present the originality of this dissertation in the second research direction.

It is a feature that inspiration of all EC algorithms comes from physical, mathematical or biological phenomena and work schemes. To obtain philosophies behind these phenomena and work schemes is a forever and not yet a completed subject. Chaotic ergodicity is a great property to be used for implementing a search mechanism in an iterative EC process. We used it to develop a new EC algorithm. we call it as Chaotic Evolution that fuses chaotic ergodicity and evolutionary iteration. The design philosophy of chaotic evolution presents the main originality and contribution in this dissertation. We hope that it can enrich researches of EC community by further studying and investigating the chaotic evolution algorithm principle and philosophy behind ergodicity and iteration.

9.2 Limitations

As our proposed acceleration methods need additional time to approximate an IEC/EC fitness landscape, their computational complexities increase. The performances of our acceleration methods is a biggest issue for their applications. Although our proposed methods show better performance than previously proposed acceleration methods, their performances are remained critical issues for their applications. When we apply them to concrete applications, we must consider their performances taking account of a balance between their computational complexities and their convergence speed. It restricts applications of these methods. Especially in the Fourier analysis method, we have to re-sample individuals and conduct DFT to obtain frequency information. It needs more computational time for high dimensional problems. Computational time is a primary key that may restrict practical applications of these methods.

Population diversity is a key consideration for obtaining the final global optimum in IEC/EC. Acceleration methods may lead to a premature convergence to a local optimum under special landscape conditions. When the almost same elite is inserted generation by generation, our proposed methods may lead to premature convergence. Selection of an elite strategy is a considerably important topic when our proposed acceleration methods is applied to concrete IEC/EC applications.

When we introduce some new mechanisms to extent EC algorithms, additional fitness evaluations or statistical metrics calculations need more computational time. Even though the proposed methods, such as triple and quadruple comparisons and search transition from exploration to exploitation, can significantly enhance the performance of the original algorithm, the additional time used in some process is a main problem for their applications, too, especially for IEC applications. Their real world applications will be involved for further investigation.

Chaotic evolution is a new proposed EC algorithm in this dissertation; however, its work principles are still not well clear. Distribution of the outputs from a chaotic system decides the optimization performance of chaotic evolution. We do not know whether there are some hidden philosophy and effective mechanism behind the work scheme of chaotic evolution; it should be studied and investigated in the future.

Other limitation of our proposals is practicability and effectiveness in a real world applications. We evaluated our proposed approaches in some benchmark functions. They are well explored and suitable for algorithm performance assessment. However, there are more challenging characteristics in real world problems, such as dynamical systems, large-scale problems, constraint problems, etc. How about the optimization performance when we apply our proposed six approaches in real world applications. We will conduct this work in the future.

9.3 Future Works

We proposed three research directions and six concrete approaches to study the subject on EC and IEC algorithm enhancement and acceleration. We should develop more concrete approaches to enrich the contents of these three research directions. We should especially concentrate on the following works and research topics for concrete methods reported in this dissertation.

It is an initial theoretical work to approximate fitness landscape in lower dimensional search space. Theoretical research is but one part of this scientific research. The ultimate objective of our research is to apply this novel method and technique to actual applications with societal benefit. We conducted simulation evaluations to compare characteristics of several methods with multiple different initializations under the same experimental conditions. In the next step, we should quantitatively evaluate IEC user fatigue and collect basic experimental data together with convergence experiments' results of this work. After then, we need to conduct human subjective tests with human IEC users, evaluate user fatigue and acceleration performance synthetically, and conclude the evaluations of the proposed methods. It is important to obtain conclusions through its application and directions as to how we can improve our work. We expect that these developments will occur on the topics such as selection of linear or nonlinear model, actual interpolation or approximation methods used and expression of regression space.

In the Fourier analysis on fitness landscape study, we can increase precision with which we approximate fitness landscapes and improve acceleration performance by using elite that is located closer to the global optimum by using multiple primary frequency components, although we used only one primary frequency component in our experiments of this work. Using multiple primary frequency components will be a topic of future research. We expect that it may be possible to extend the proposed approach into evolutionary multi-objective optimization. That is, unlike with conventional Pareto concept, we may be able to handle multiple objectives in one domain, i.e, the frequency domain, by projecting each objective landscape into its corresponding frequency domain. A further investigation of this idea will form the basis of another future study in multi-objective optimization. Another topic is wavelets transform can be applied in this analysis method, we will investigate it in the future.

The frequency resolution setting is a successful factor for Fourier niche method. The lower frequency resolution cannot distinguish high frequency in an original search space. We have to apply different frequency resolution settings in Fourier niche approach for discovering a correct frequency for an unknown landscape. It is the best way to set a proper frequency resolution to obtain a wide frequency scale. This is a topic in our future work. The search radius is an important parameter for success of the proposal. If the search radius is large, elite may escape from peak location by chance, on the contrary, it will lead a slow convergence when it is too small. Adaptive tuning search radius value for each elite may be the best way to solve this problem.

In the triple and quadruple comparison based DE and IDE study, we conducted simulation evaluations as well to compare the characteristics of several methods with multiple different initializations under the same experimental conditions. In the next step, we should quantitatively evaluate IDE user fatigue with a largest population size that is hard for a human being to memorize and collect basic experimental data together with convergence experiment results in this work. After that, we need to conduct human subjective tests with human IDE users, evaluate user fatigue and acceleration performance synthetically, and thus conclude our evaluation of methods proposed here.

In the study of EC acceleration method by accelerating transition from exploration to exploitation, parameter setting of the proposal is an important factor to obtain a better acceleration performance. Development of an adaptive tuning method for parameters by observing search condition is an effective way to improve this method. Other possibility is to obtain new acceleration methods by observing a population distribution.

The main contribution of chaotic evolution is introducing a novel evolutionary optimization algorithm that fuses chaotic ergodicity and evolutionary iteration. Possible directions for future work include chaotic system equation selection, direction factor rate setting method investigation and crossover rate necessity issue. We will conduct these works in the future.

Bibliography

[1] M. Ahandani and H. Alavi-Rad. Opposition-based learning in the shuffled differential evolution algorithm. *Soft Computing*, 16(8):1–35, 2012.

[2] S. Baluja. Population based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Technical Report CMU-CS-94-163, Carnegie Mellon University, 1994.

[3] S. Baluja and S. Davies. Using optimal dependency trees for combinatorial optimization: Learning the structure of search space. Technical Report CMU-CS-97-107, Carnegie Mellon University, 1997.

[4] D. Beasley, D. R. Bull, and R. R. Martin. *An Overview of Genetic Algorithms*, chapter Part 1: Fundamentals, pages 58–69. University Computing, 1993.

[5] R. Bellman. *Dynamic Programming.* Princeton University Press, 1957.

[6] D. Bhandaarl and C. A. Murthy. Genetic algorithm with elitist model and its convergence. *Int. J. Pattern Recognition and Artificial Intelligence*, 10(5):731–734, 1996.

[7] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Umer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *IEEE Trans. on Evolutionary Computation.*, 10(6):107–125, 2008.

[8] A. Brownlee, O. Regnier-Coudert, J. McCall, and S. Massie. Using a markov network as a surrogate fitness function in a genetic algorithm. In *IEEE Congress on Evolutionary Computation*, pages 1–8, 2010.

[9] O. Buchtala, M. Klimek, and B. Sick. Evolutionary optimization of radial basis function classifiers for data mining applications. *IEEE Trans. on System Man and Cybenetics, Part B*, 35(5):928–947, 2005.

[10] R. Caponetto, L. Fortuna, and S. Fazzino. Chaotic sequences to improve the performance of evolutionary algorithms. *IEEE Trans. on Evolutionary Computation*, 7(3):289–304, 2003.

[11] J. A. Carretero and M. A. Nahon. Solving minimum distance problems with convex or concave bodies using combinatorial global optimization algorithms. *IEEE Trans. on System Man and Cybenetics, Part B*, 35(6):1144–1155, 2005.

[12] J. W. Cooley and W. T. John. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[13] S. Cui, A. Mohan, and D. S. Weile. Pareto optimal design of absorbers using a parallel elitist nondominated sorting genetic algorithm and the finite element-boundary integral method. *IEEE Trans. on Antennas and Propagation*, 53(6):2099–2107, 2005.

[14] R. Dawkins. *The Blind Watchmaker.* Essex: Longman, 1986.

[15] J. S. De Bonet, C. L. Isbell, P. Viola, et al. Mimic: Finding optima by estimating probability densities. *Advances in Neural Information Processing Systems*, pages 424–430, 1997.

[16] L. Dos S Coelho, T. Bora, and L. Lebensztajn. A chaotic approach of differential evolution optimization applied to loudspeaker design problem. *IEEE Trans. on Magnetics*, 48(2):751–754, 2012.

[17] L. Eshelman and J. D. Shaffer. *Foundations of Genetic Algorithm 2*, chapter Real-coded Genetic Algorithms and Interval Schemata, pages 187–202. Morgan Kaufmann, 1993.

[18] A. R. Ferreira da Silva. Evolutionary time-frequency analysis. In *IEEE congress on evolutionary computation 2000*, volume 2, pages 1102–1109, 2000.

[19] D. Florin and H. Philip. *Celestial Encounters: The Origins of Chaos and Stability.* Princeton Uni. Press, 1996.

[20] L. J. Fogel. Autonomous automata. *Industrial Research*, 4(2):14–19, 1962.

[21] D. Goldberg and J. Richardson. Genetic algorithms with sharing for multi-modal function optimization. In *Genetic Algorithms and Their Applications: The Second Int. Conf. on Genetic Algorithms*, pages 41–49, 1987.

[22] D. E. Goldberg. *Optimization Machine Learning*, chapter Genetic Algorithms in Search. Addison-Wesley, 1989.

[23] D. E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and first result. *Complex System*, 3(5):493–530, 1988.

[24] E. Goldberg, David. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.

[25] W. Gong, Z. Cai, C. X. Ling, and H. Li. Enhanced differential evolution with adaptive strategies for numerical optimization. *IEEE Trans. on Systems, Man and Cybernetic*, 41(2):397–413, 2011.

[26] J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Trans. on System Man and Cybenetics*, 16(1):122–128, 1986.

[27] O. Hajji, S. Brisset, and P. Brochet. A stop criterion to accelerate magnetic optimization process using genetic algorithms and finite element analysis. *IEEE Trans. on Magnetics*, 39(3):1297–1300, 2003.

[28] G. Harick. Finding multi-modal solutions using restricted tournament selection. In *The Sixth Int. Conf. on Genetic Algorithms*, pages 24–31, 1995.

[29] G. Harik. Linkage learning via probabilistic modeling in the ecga. Technical Report 99010, Illinois Genetic Algorithm Laboratory, University of Illinois, 1999.

[30] G. R. Harik, F. G. Lobo, and D. E. Goldberg. The compact genetic algorithm. In *IEEE Conf. on Evolutionary Computation*, pages 523–528, 1998.

[31] M. Herdy. Evolutionary optimisation based on subjective selection-evolving blends of coffee. In *5th Eourpean Congress on Itellengt Techniques and Soft Computing*, pages 640–644, 1997.

[32] J. Holland. *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, 1975.

[33] X. Huang, P. Jia, and B. Liu. Chaotic particle swarm optimization for synchronization of finite dimensional henon dynamical system. In *2010 Sixth Int. Conf. on Natural Computation*, volume 5, pages 2600 – 2604, 2010.

[34] J. Hunger and G. Huttner. Optimization and analysisof force field parameters by combination of genetic algorithm and neural networks. *J. of Computational Chemistry*, 20(4):455–471, 1999.

[35] G. Iacca, F. Neri, and E. Mininno. Opposition-based learning in compact differential evolution. In *Lecture Notes in Computer Science*, volume 6624 of *Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*, pages 264–273, 2011.

[36] T. Ingu and H. Takagi. Accelerating a ga convergence by fitting a single-peak function. In *IEEE Int. Conf. on Fuzzy Systems*, pages 1415–1420, 1999.

[37] R. Jahani, N. H. Chahkandi, and Z. H. Ghiasi. Applying chaotic optimization algorithm for optimal power distribution. *Australian J. of Basic and Applied Sciences*, 5(9):917–922, 2011.

[38] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9(1):3–12, 2005.

[39] B. Johanson. Automated fitness raters for the gp-music system. Master's degree final project, Univ. of Birmingham, 1997.

[40] D. Jones. A taxonomy of global optimization methods based on response surface. *J. of Global Optimization*, 21(4):345–383, 2001.

[41] K. A. D. Jong. *An analysis of the behavior of a class of genetic adaptive systems.* PhD thesis, University Michigen, 1975.

[42] J. H. Kim, H. K. Chae, J. Y. Jeon, and S. W. Lee. Identification and control of systems with friction using accelerated evolutionary programming. *IEEE Control Systems Magazine*, 16(4):38–47, 1996.

[43] H. Kita, I. Ono, and S. Kobayashi. Theoretical analysis of the unimodal normal distribution crossover for real-coded genetic algorithms. In *IEEE Int. Conf. on Evolutionary Computation*, pages 529–534, 1998.

[44] A. Kosorukoff. Human-based genetic algorithm. In *2001 IEEE Int. Conf. on Systems, Man and Cybernetic*, pages 3464–3469, 2001.

[45] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* The MIT Press, 1992.

[46] A. Lahiri and S. Chakravorti. Electrode-spacer contour optimization by ann aided genetic algorithm. *IEEE Trans. on Dielectrics and Electrical Insulation*, 11(6):964–975, 2004.

[47] P. Larranaga and J. A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Optimization.* Kluwer Academic Publishers, 2001.

[48] d. S. C. Leandro and C. M. Viviana. Combining of chaotic differential evolution and quadratic programming for economic dispatch optimization with valve-point effect. *IEEE Trans. on Power Systems*, 21(2):989–996, 2006.

[49] d. S. C. Leandro and C. M. Viviana. A novel chaotic particle swarm optimization approach using hennon map and implicit filtering local search for economic load dispatch. *Chaos, Solitons & Fractals*, 39(2):510–518, 2009.

[50] P. Legrand, C. Bourgeois-Republique, V. Pean, E. H. Cohen, J. Levy-Vehel, B. Frachet, E. Lutton, and P. Collet. Interactive evolution for cochlear implants fitting. *Genetic Programming and Evolvable Machines*, 8(4):319–354, 2007.

[51] Y. W. Leung and Y. Wang. An orthogonal genetic algorithm with quantization for global numerical optimization. *IEEE Trans. on Evolutionary Computation*, 5(1):41–53, 2001.

[52] J. P. Li, M. E. Balazs, T. P. Geoffrey, and P. J. Clarkson. A species conserving genetic algorithm for multi-modal function optimization. *Evolutionary Computation*, 10(3):207–234, 2002.

[53] I. S. Lim and D. Thalmann. Tournament selection for browsing temporal signals. In *ACM Symp. on Applied Computing*, pages 570–573, 2000.

[54] B. Liu, L. Wang, Y.-H. Jin, F. Tang, and D.-X. Huang. Improved particle swarm optimization combined with chaos. *Chaos, Solitons & Fractals*, 25(5):1261–1271, 2005.

[55] E. N. Lorenz. Deterministic non-periodic flow. *J. of the Atmospheric Sciences*, 20(2):130–141, 1963.

[56] J. Mader, J. Abonyi, and F. Szeifert. Interactive particle swarm optimization. In *Int. Conf. on Intelligent Systems Design and Applications*, pages 2314–319, 2005.

[57] S. W. Mahfoud. Crowding and preselection revisited. *Parallel Problem Solving from Nature*, 2:27–36, 1992.

[58] B. B. Mandelbrot. *The Fractal Geometry of Nature.* W. H. Freeman, 1982.

[59] A. H. Mantawy, Y. L. Abdel-Magid, and S. Z. Selim. Integrating genetic algorithms, tabu search, and simulated annealing for the unit commitment problem. *IEEE Trans. on Power Systems*, 14(3):829–836, 1999.

[60] R. M. May. Simple mathematical models with very complicated dynamics. *Nature*, 261(5560):459–467, 1976.

[61] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

[62] H.-A. Mehrdad, H. G. Seyyed, and K. Reza. Multi-objective genetic local search algorithm using kohonen's neural map. *Computers and Industrial Engineering*, 56(4):1566–1576, 2009.

[63] O. Mengsheol and D. Goldberg. Probabilistic crowding: Deterministic crowding with probabilistic replacement. In *Genetic and Evolutionary Computation Conf.*, pages 409–416, 1999.

[64] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, 1996.

[65] G. A. Miller. The magical mumber seven, plus or minus two: Some limits on our capacity for processing information. *The Psychological Review*, 63(2):81–97, 1956.

[66] J. A. Miller, W. D. Potter, R. V. Gandham, and C. N. Lapena. An evaluation of local improvement operators for genetic algorithms. *IEEE Trans. on System Man and Cybenetics*, 23(5):1340–1351, 1993.

[67] H. Muehlenbein. The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5(3):303–346, 1998.

[68] H. Muehlenbein and T. Mahnig. The factorized distribution algorithm for additively decomposed functions. In *IEEE Congress on Evolutionary Computation*, pages 752–759, 1999.

[69] Y. Nakano and H. Takagi. Influence of quantization noise in fitness on the performance of interactive pso. In *IEEE Congress on Evolutionary Computation*, pages 2146–2422, 2009.

[70] N. Nikolaev and H. Iba. Genetic programming of polynomial harmonic models using the discrete fourier transform. In *IEEE Congress on evolutionary computation 2001*, volume 1, pages 267–274, 2001.

[71] A. Paranya and M. Phayung. A chaos search for multi-objective memetic algorithm. In *2011 Int. Conf. on Information and Electronics Engineering*, pages 140–144, 2011.

[72] B. K. Pathak, S. Srivastava, and K. Srivastava. Neural network embedded multiobjective genetic algorithm to solve non-linear time-cost tradeoff problems of project scheduling. *J. of Scientific & Industrial Research*, 67(2):124–131, 2008.

[73] Y. Pei. A chaotic ergodicity based evolutionary computation algorithm. In *The 2013 9th Int. Conf. on Natural Computation*, pages 454–459, 2013.

[74] Y. Pei and H. Takagi. Accelerating evolutionary computation with elite obtained in projected one-dimensional spaces. In *5th Int. Conf. on Genetic and Evolutionary Computing*, pages 89–92, 2011.

[75] Y. Pei and H. Takagi. Comparative evaluations of evolutionary computation with elite obtained in reduced dimensional spaces. In *3rd Int. Conf. on Intelligent Networking and Collaborative Systems*, pages 35–40, 2011.

[76] Y. Pei and H. Takagi. Novel traveling salesman problem solution by accelerated evolutionary computation with approximated cost matrix in an industrial application. In *3rd Int. Conf. on Soft Computing and Pattern Recognition*, pages 39–44, 2011.

[77] Y. Pei and H. Takagi. A survey on accelerating evolutionary computation approaches. In *3rd Int. Conf. on Soft Computing and Pattern Recognition*, pages 201–206, 2011.

[78] Y. Pei and H. Takagi. Comparative study on fitness landscape approximation with fourier transform. In *Sixth Int. Conf. on Genetic and Evolutionary Computation*, pages 400–403, 2012.

[79] Y. Pei and H. Takagi. Fourier analysis of the fitness landscape for evolutionary search acceleration. In *IEEE Congress on Evolutionary Computation*, pages 2934–2940, 2012.

[80] Y. Pei and H. Takagi. Accelerating iec and ec searches with elite obtained by dimensionality reduction in regression spaces. *Evolutionary Intelligence*, 6(1):27–40, 2013.

[81] Y. Pei and H. Takagi. Triple and quadruple comparison-based interactive differential evolution and differential evolution. In *The twelfth workshop on foundations of genetic algorithms XII*, pages 173–182, 2013.

[82] Y. Pei, S. Zheng, Y. Tan, and H. Takagi. An empirical study on influence of approximation approaches to enhance fireworks algorithm. In *2012 IEEE Int. Conf. on Systems, Man and Cybernetics*, pages 1322–1327, 2012.

[83] M. Pelikan, D. E. Goldberg, and E. Cantu-paz. Linkage problem, distribution estimation and bayesian networks. *Evolutionary Computation*, 8(3):311–340, 2000.

[84] M. Pelikan and H. Muehlenbein. The bivariate marginal distribution algorithm. *Advances in Soft Computing Engineering Design and Manufacturing*, pages 521–535, 1999.

[85] A. Petrowski. A clearing procedure as a niching method for genetic algorithms. In *Third IEEE Int. Conf. on Evolutionary Computation*, pages 798–803, 1996.

[86] W. D. Potter, J. A. Miller, B. E. Tonn, R. V. Gandham, and C. N. Lapena. Improving the reliability of heuristic multiple fault diagnosis via the ec-based genetic algorithm. *J. of Applied Intelligence*, 2(1):5–23, 1992.

[87] K. Price, R. Storn, and J. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization.* Springer-Verlag, 2005.

[88] A. K. Qin, V. L. Huang, and P. N. Suganthan. Differential evolution algorithm with strategy adaptition for global numberical optimization. *IEEE Trans. on Evolutionary Computation*, 13(2):398–417, 2009.

[89] A. K. Qin, V. L. Huang, and P. N. Suganthan. Jade: Adaptive differential evolution with external achive. *IEEE Trans. on Evolutionary Computation*, 13(5):945–958, 2009.

[90] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. S. Salama. Opposition-based differential evolution algorithms. *IEEE Trans. on Evolutionary Computation*, 12(1):64–79, 2008.

[91] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. S. Salama. Opposition versus randomness in soft computing techniques. *Apply Soft Computing*, 8(2):906–918, 2008.

[92] S. Raungrong, F. C. Luis, and A. Aydin. Discrete evolutionary transform for time frequency signal analysis. *J. of the Franklin Institute*, 337(4):347–364, 2000.

[93] R. G. Regis and C. A. Shoemaker. Local function approximation in evolutionary algorithms for the optimization of costly functions. *IEEE Trans. on Evolutionary Computation*, 8(5):490–505, 2004.

[94] G. Rudolph. Convergence analysis of canonical genetic algorithms. *IEEE Trans. on Neural Networks*, 5(1):96–101, 1994.

[95] N. N. Schraudolph and R. K. Belew. Dynamic parameter encoding for genetic algorithms. *Machine Learning*, 9(1):9–22, 1992.

[96] H. P. Schwefel. *Evolutionsstrategie und Numerische Optimierung.* PhD thesis, Technische Universitat Berlin, 1975. in German.

[97] H. P. Schwefel. *Numerical Optimization of Computer Models.* Wiley Chichester, 1981.

[98] H. P. Schwefel. *Evolution Optimum Seeking.* Sixth Generation Computer Technology Series. John Wiley and Sons, 1995.

[99] S. T. Sheu and Y. R. Chuang. A pipeline-based genetic algorithm accelerator for time-critical processes in real-time systems. *IEEE Trans. on Computers*, 55(11):1435–1448, 2006.

[100] K. Sims. Ariticial evolution for computer graphics. *Computer Graphics*, 25(4):319–328, 1991. Proc. of Siggraph 91.

[101] L. Smith. *Chaos: a very short introduction.* Oxford University Press, 2007.

[102] Y. Song and C. S. V. Chou. Advanced engineered-conditioning genetic approach to power economic dispatch. *IEE Proceedings on Generation, Transmission and Distribution*, 144(3):285–292, 1997.

[103] R. Storn and K. Price. A simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, 11(4):341–359, 1997.

[104] P. Suganthan, N. Hansen, J. Liang, K. Deb, Y. Chen, A. Auger, and S. Tiwari. Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. Technical report, Nanyang Technological University, 2005.

[105] G. Sugihara and R. May. Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series. *Nature*, 344(6268):734–741, 1990.

[106] H. Takagi. *Intelligent Systems: Fuzzy Logic, Neural Networks, and Genetic Algorithms*, chapter Introduction to fuzzy systems, neural networks, and genetic algorithms, pages 1–33. Kluwer Academic Publishers, 1997.

[107] H. Takagi. Fusion technology of neural networks and fuzzy systems: A chronicled progression from the laboratory to our daily lives. *Int. J. of Applied Mathematics and Computer Science*, 10(4):647–673, 2000.

[108] H. Takagi. Interactive evolutionary computation: Fusion of the capabilities of ec optimization and human evaluation. *Proceedings of the IEEE*, 89(9):1275–1296, 2001.

[109] H. Takagi. *New IEC Research and Framkworks*, chapter in Aspects of Soft Computing, Intelligent Robotic and Control, pages 65–78. Springer-Verlag, 2009.

[110] H. Takagi. Interactive evolutionary computation for analyzing human aware mechanism. *Applied Computational Intelligence and Soft Computing*, Article ID: 694836, 2012.

[111] H. Takagi, T. Ingu, and K. Ohnishi. Accelerating a ga convergence by fitting a single-peak function. *J. of Japan Society for Fuzzy Theory and Intelligent Informatics*, 15(2):219–229, 2003. in Japanese.

[112] H. Takagi and M. Ohsaki. Interactive evolutionary computation-based hearing-aid fitting. *IEEE Trans. on Evolutionary Computation*, 11(23):414–427, 2007.

[113] H. Takagi and D. Pallez. Paired comparison-based interactive differential evolution. In *The First World Congress on Nature and Biologically Inspired Computing*, pages 375–480, 2009.

[114] H. Tizhoosh. Opposition-based learning: A new scheme for machine intelligence. In *Int. Conf. on Computational Intelligence for Modelling Control and Automation*, volume I, pages 695–701, 2005.

[115] H. Tizhoosh. Reinforcement learning based on actions and opposite actions. In *ICGST Int. Conf. on Artificial Intelligence and Machine Learning*, 2005.

[116] H. Wang, Z. Wu, and S. Rahnamayan. Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems. *Soft Computing*, 15(11):2127–2140, 2011.

[117] S. Y. Wang, F. C. Tian, X. Liu, and J. Wang. A novel representation approach to dna sequence and its application. *IEEE signal processing letters*, 16(4):275–278, 2009.

[118] Y. Wang, Z. X. Cai, G. Q. Guo, and Y. R. Zhou. Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems. *IEEE Trans. on System Man and Cybenetics, Part B*, 37(3):560–575, 2007.

[119] E. Weinberger. Fourier and taylor series on fitness landscapes. *Biological Cybernetics*, 65(5):321–330, 1991.

[120] D. Whitley, K. Mathias, and P. Fitzhorn. Delta coding: an interative search strategy for genetic algorithms. In *4th Int. Conf. on Genetic Algorithms*, pages 77–84, 1991.

[121] K. P. Wong, A. Li, and T. M. Y. Law. Development of constrained genetic algorithm load flow method. *IEE Proceeding on Generation, Transmission and Distribution*, 144(2):91–99, 1997.

[122] K. P. Wong, A. Li, and T. M. Y. Law. Advanced constrained genetic algorithm load flow method. *IEE Proceeding on Generation, Transmission and Distribution*, 146(6):609–616, 1999.

[123] X. Yin and N. Germay. Investigations on solving load flow problems by genetic algorithms. *Electric Power System Research*, pages 151–163, 1991.

[124] L. A. Zadeh. Fuzzy sets. *Information and control*, 8(3):338–353, 1965.

[125] L. A. Zadeh. Fuzzy logic, neural networks, and soft computing. *Communications of the ACM*, 37(3):77–84, 1994.

[126] Z. Zhou, Y. S. Ong, M. H. Nguyen, and D. Lim. A study on polynomial regression and gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm. In *IEEE Congress on Evolutionary Computation*, volume 3, pages 2832–2839, 2005.

[127] Z. Z. Zhou, Y. S. Ong, P. B. Nair, J. Keane, and Y. K. Lum. Combining global and local surrogate models to accelerate evolutionary optimization. *IEEE Trans. on System Man and Cybenetics, Part C*, 37(1):66–76, 2007.

[128] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.

Acknowledgements

I would like to thank Professor Hideyuki Takagi, my supervisor, for his many suggestions and stronger supports during my doctoral research. I obtained so much knowledge on scientific methodology and research philosophy when I was working with him. Especially, his research ethic and attitude do and will do influence my future research work as an eligible researcher.

I would like to thank four referees, Professors Hideyuki Takagi, Mario Köppen, Hajime Kimura, and Kiichi Urahama for their helpful comments for polishing up this dissertation.

I would like to thank Yoshida Scholarship Foundation for their supports to my study in Japan. I should also mention that my doctoral research was supported in part by Grants-in-Aid of The Japan Society for the Promotion of Science (No. 23500279).

Of course, I am grateful to my parents (Lin PEI and Huijuan JIANG) for their patience and love. Without them, I cannot finish my whole study and research work in my life.

Yan PEI

Fukuoka, Japan
December 17, 2013

Publications Related to Each Chapter

This is a list of publications during doctoral research in each chapter.

Chapter 1: Introduction

Chapter 2: Related Techniques and Works

1. **Yan Pei** and Hideyuki Takagi, "*A Survey on Accelerating Evolutionary Computation Approaches*", Third Int. Conf. on Soft Computing and Pattern Recognition (SoCPaR2011), pp.201-206, Dalian, China (14-16, Oct., 2011).

Chapter 3: Accelerating IEC and EC Searches with Elite Obtained in Projected Lower-Dimensional Spaces

1. **Yan Pei** and Hideyuki Takagi, "*Accelerating IEC and EC searches with elite obtained by dimensionality reduction in regression spaces*", Evolutionary Intelligence, Springer, Vol.6, No.1, pp.27-44, DOI: 10.1007/s12065-013-0088-9(2013).

2. **Yan Pei** and Hideyuki Takagi, "*Accelerating Evolutionary Computation with Elite Obtained in Projected One-Dimensional Spaces*", Fifth Int. Conf. on Genetic and Evolutionary Computation (ICGEC2011), pp.89-92, Kinmen Taiwan / Xiamen China (29, Aug., - 1, Sept., 2011).

3. **Yan Pei** and Hideyuki Takagi, "*A Novel Traveling Salesman Problem Solution by Accelerated Evolutionary Computation with Approximated Cost Matrix in an Industrial Application*", Third Int. Conf. on Soft Computing and Pattern Recognition (SoCPaR2011), pp.39-44, Dalian, China (14-16, Oct., 2011).

4. **Yan Pei** and Hideyuki Takagi, "*Comparative Evaluations of Evolutionary Computation with Elite Obtained in Reduced Dimensional Spaces*", Third Int. Conf. on Intelligent Networking and Collaborative Systems (INCoS2011), pp.35-40, Fukuoka, Japan (30, Nov., - 2, Dec., 2011).

5. **Yan Pei**, Shaoqiu Zheng, Ying Tan and Hideyuki Takagi, "*An Empirical Study on Influence of Approximation Approaches to Enhancing Fireworks Algorithm*", 2012 IEEE Int. Conf. on Systems, Man, and Cybernetics (IEEE SMC 2012), pp.1322-1327, Seoul, Korea (14-17, Oct., 2012).

6. **Yan Pei** and Hideyuki Takagi, "*Accelerating Evolutionary Computation with Elite Obtained by Dimensionality Reduction*", Joint meeting of 7th Evolutionary Computation Frontier (SIG-ECF) meeting and 1st Evolutionary Computation Meeting, Tokyo University Hongo Campus, Tokyo, Japan, pp.25-31 (9-10, Sep., 2011). (in Japanese)

Chapter 4: Fourier Analysis on Fitness Landscape for Evolutionary Search Acceleration

1. **Yan Pei** and Hideyuki Takagi, "*Fourier Analysis of the Fitness Landscape for Evolutionary Search Acceleration*", 2012 IEEE Congress on Evolutionary Computation (IEEE CEC 2012), pp.2934-2940, Brisbane, Australia (10-15, Jun., 2012).

2. **Yan Pei** and Hideyuki Takagi, "*Comparative Study on Fitness Landscape Approximation with Fourier Transform*", Sixth Int. Conf. on Genetic and Evolutionary Computation (ICGEC2012), pp.400-403, Kitakyushu, Japan (25-28, Aug., 2012).

3. **Yan Pei** and Hideyuki Takagi, "*Fourier Analysis of Fitness Landscape to Accelerate Evolutionary Search*", Evolutionary Computation Symposium 2011, Iwanuma, Miyagi Perfecture, Japan, pp.167-173 (17-18, Dec., 2011). (in Japanese)

Chapter 5: Fourier Niche Method for Multimodal Optimization

1. **Yan Pei** and Hideyuki Takagi, "*Fourier Niching Approach for Multi-modal Optimization*", Joint meeting of 8th Evolutionary Computation Frontier (SIG-ECF) meeting and 2nd Evolutionary Computation Meeting, Osaka University Toyonaka Campus, Osaka, Japan, pp.193-199 (9-10, Mar., 2012). (in Japanese)

Chapter 6: Triple and Quadruple Comparison-Based Interactive Differential Evolution and Differential Evolution

1. **Yan Pei** and Hideyuki Takagi, "*Triple and Quadruple Comparison-Based Interactive Differential Evolution and Differential Evolution*", Transaction of the Japanese Society for Evolutionary Computation, Vol.3, No.2, pp.98-108 (2012) (in Japanese).

2. **Yan Pei** and Hideyuki Takagi, "*Triple and Quadruple Comparison-Based Interactive Differential Evolution and Differential Evolution*", Foundations of Genetic Algorithms (FOGA2013) Workshop XII, pp.173-182, Adelaide, Australia (16-20, Jan., 2013)

3. **Yan Pei** and Hideyuki Takagi, "*Triple and Quadruple Comparison-Based Interactive Differential Evolution and Differential Evolution*", 3rd Evolutionary

Computation Meeting, Hiroshima University Higashi-Hiroshima Campus, Hiroshima, Japan, pp.74-84 (9-10, Sep., 2012). (in Japanese)

Chapter 7: EC Acceleration by Accelerating Transition from Exploration to Exploitation

1. Hideyuki Takagi and **Yan Pei**, "*Proposal of a Method for Accelerating Transition from Exploration to Exploitation*", 4rd Evolutionary Computing meeting, Yokosuka, Japan, pp.96-101 (18-19, Mar., 2013) (in Japanese).

Chapter 8: Chaotic Evolution: A New EC Algorithm and Framework

1. **Yan Pei**, "*A Chaotic Ergodicity Based Evolutionary Computation Algorithm*", The 2013 9th International Conference on Natural Computation (ICNC'13), pp.454-459, Shenyang, China (23-25, Jul., 2013).

2. **Yan Pei**, "*Chaotic Evolution: Fusion of Evolutionary Iteration Property and Chaotic Ergodicity Mechanism*", Japanese Society for Evolutionary Computation Symposium 2012, Karuizawa, Nagano, Japan, pp.256-263 (15-16, Dec., 2012).

Chapter 9: Conclusion and Future Work

Other Publications

1. **Yan Pei** and Hideyuki Takagi, "*Fitness Landscape Approximation by Adaptive Support Vector Regression with Opposition-Based Learning*", 2013 IEEE International Conference on Systems, Man, and Cybernetics (IEEE SMC2013), Manchester, UK (13-16, Oct., 2013).

2. **Yan Pei** and Hideyuki Takagi, "*Approximating and Analyzing Fitness Landscape for Evolutionary Search Enhancement*", 14th Japan Society for Fuzzy Theory and Intelligent Informatics Kyushu Chapter Annual Conference (SOFT-Kyushu2012), pp.21-24, Saga, Japan (8, Dec., 2012). (in Japanese)

3. **Yan Pei** and Hideyuki Takagi, "*Approximating and Analyzing Fitness Landscape for Evolutionary Search Enhancement*", Satellite Workshop on Problem, Landscape Analysis, Automated Algorithm Selection and Adaptation in Optimization at Foundations of Genetic Algorithms (FOGA2013) Workshop XII, Adelaide, Australia (16-20, Jan., 2013)

4. **Yan Pei** and Hideyuki Takagi, "*Proposal of a method for determining search states of Markov chain practically and its application to predicting EC convergence*", Japanese Society for Evolutionary Computation Symposium 2013, pp.124-127, Kirishima, Japan (14-15, Dec., 2013).

Appendix: Benchmark Functions

F1:Sphere Function
$f(x) = \sum_{i=1}^{n} x_i^2$
Properties: $x \in [-5.12, 5.12]$, uni-modal, separable

F2: Rosenbrock Function
$f(x) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2$
Properties: $x \in [-2.048, 2.048]$, multi-modal, non-separable

F3: Step Function
$f(x) = \sum_{i=1}^{n} \lfloor x_i \rfloor$
Properties: $x \in [-5.12, 5.12]$, uni-modal, separable

F4: Quantic Function
$f(x) = \sum_{i=1}^{n} ix_i^4 + Gauss(0, 1)$
Properties: $x \in [-1.28, 1.28]$, uni-modal, separable

F5: Shekel's Foxholes Function
$f(x) = [0.02 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}]^{-1}$
Properties: $x \in [-65.536, 65.536]$, multi-modal, separable

F6: Rastrigin Function
$f(x) = (10n) + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i))$
Properties: $x \in [-5.12, 5.12]$, multi-modal, separable

F7: Schwefel 2.26 Function
$f(x) = \sum_{i=1}^{n} (-x_i \sin(\sqrt{|x_i|}))$
Properties: $x \in [-512, 512]$, multi-modal, separable

F8: Griewank Function
$f(x) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}})$
Properties: $x \in [-512, 512]$, multi-modal, non-separable

F9: Schaffer 1 Function
$f(x) = 0.5 + \frac{\sin^2(\sqrt{(x_1^2 + x_2^2)}) - 0.5}{[1.0 + 0.001(x_1^2 + x_2^2)]^2}$
Properties: $x \in [-100, 100]$, multi-modal, non-separable

F10: Schaffer 2 Function
$f(x) = (x_1^2 + x_2^2)^{0.25} [\sin^2(50(x_1^2 + x_2^2)^{0.1}) + 1.0]$
Properties: $x \in [-100, 100]$, multi-modal, separable

F11: Schwefel 2.22 Function
$f(x) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i|$
Properties: $x \in [-10, 10]$, uni-modal, non-separable

F12: Schwefel 1.2 Function
$f(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} (x_j))^2$
Properties: $x \in [-10, 10]$, uni-modal, non-separable

F13: Hartman-3 Function

$f(x) = -\sum_{i=1}^{4} c_i \exp[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2]$

Properties: $x \in [0, 1], n = 3$, multi-modal, non-separable

$a[4][3] = \{3, 10, 30, 0.1, 10, 35, 3, 10, 30, 0.1, 10, 35\}$

$c[4] = \{1, 1.2, 3, 3.2\}$

$p[4][3] = \{0.3689, 0.1170, 0.2673, 0.4699, 0.4387, 0.7470,$
$0.1091, 0.8732, 0.5547, 0.038150, 05743, 0.8828\}$

F14: Step Function

$f(x) = \sum_{i=1}^{n}(\lfloor x_i + 0.5 \rfloor)^2$

Properties: $x \in [-10, 10]$, uni-modal, separable

F15: Beale Function

$f(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$

Properties: $x \in [-4.5, 4.5], n = 2$, uni-modal, non-separable

F16: Kowalik Function

$f(x) = \sum_{i=1}^{11}[a_i - \frac{x_1(b_i+b_ix_2)}{b_i^2+b_ix_3+x_4}]^2$

Properties: $x \in [-5, 5], n = 4$, multi-modal, non-separable

F17: Carnel-Back Function

$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 - 4x_2^4$

Properties: $x \in [-5, 5]$, multi-modal, non-separable

F18: Branin Function

$f(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10$

Properties: $x \in [-5, 10] * [0, 15]$, multi-modal, non-separable

F19: Goldstein-Price Function

$f(x) = [1 + (x_1 + x_2 + 1)^2 * (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] *$
$[30 + (2x_1 - 3x_2)^2 * (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$

Properties: $x \in [-2, 2]$, multi-modal, separable

F20: Hartman-6 Function

$f(x) = -\sum_{i=1}^{4} c_i \exp[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2]$

Properties: $x \in [-2.048, 2.048], n = 6$, multi-modal, non-separable

F21: Elliptic Function

$f(x) = \sum_{i=1}^{n}(10^6)^{\frac{i-1}{n-1}}x_i^2$

Properties: $x \in [0, 1]$, uni-modal, non-separable

F22: Schwefel 1.2 Function with Noise

$f(x) = \sum_{i=1}^{n}(\sum_{j=1}^{i}(x_j))^2 * (1 + 0.4|N(0, 1)|)$

Properties: $x \in [-100, 100]$, uni-modal, non-separable

F23: Schwefel 2.6 Function

$f(x) = \max\{|x_1 + 2x_2 - 7|, |2x_1 + x_2 - 5|\}$

Properties: $x \in [-100, 100]$, uni-modal, non-separable

F24: Ackley Function

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

Properties: $x \in [-32, 32]$, multi-modal, non-separable

F25: Weierstrass Function

$f(x) = \sum_{i=1}^{n}(\sum_{k=0}^{kmax}[a^k \cos(2\pi b^k(x_i + 0.5))] - n\sum_{k=0}^{kmax}[a^k \cos(2\pi b^k * 0.5)])$

Properties: $x \in [-0.5, 0.5], a = 0.5, b = 3, Kmax = 20$, multi-modal, non-separable

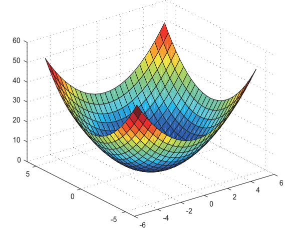
F26: Schwefel 2.13 Function
$f(x) = \sum_{i=1}^{n}(\sum_{j=1}^{n}(a_{ij}\sin x_i + b_{ij}\cos x_i))^2$
Properties: $x \in [-\pi, \pi]$, multi-modal, non-separable
F27: Gaussian Mixture Function
$GMM(x) = \sum_{i=0}^{k} a_i \exp(-\sum_{j=0}^{n}\frac{(x_{ij}-\mu_{ij})^2}{2\sigma_{ij}^2})$
Properties: $x \in [-5.12, 5.12]$, multi-modal, non-separable

$$\sigma = \begin{pmatrix} 1.5 & 1.5 & 1.5 & 1.5 & 1.5 & 1.5 & 1.5 & 1.5 & 1.5 & 1.5 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \end{pmatrix}$$

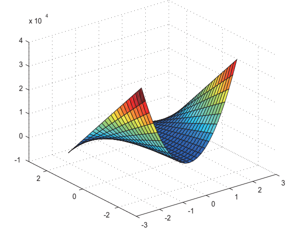
$$\mu = \begin{pmatrix} -1 & 1.5 & -2 & 2.5 & -1 & 1.5 & -2 & 2.5 & -1 & 1.5 \\ 0 & -2 & 3 & 1 & 0 & -2 & 3 & 1 & 0 & -2 \\ -2.5 & -2 & 1.5 & 3.5 & -2.5 & -2 & 1.5 & 3.5 & -2.5 & -2 \\ -2 & 1 & -1 & 3 & -2 & 1 & -1 & 3 & -2 & 1 \end{pmatrix}$$

$$a_i = \begin{pmatrix} 3.1, & 3.4, & 4.1, & 3 \end{pmatrix}^T$$

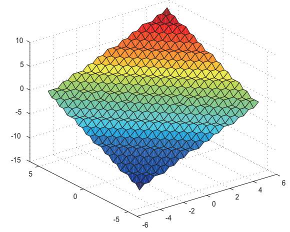
Sphere Function

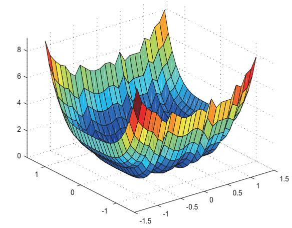


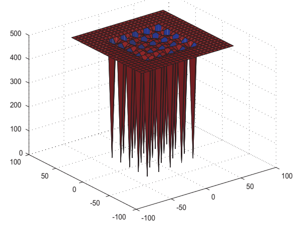
DeJong Function 2



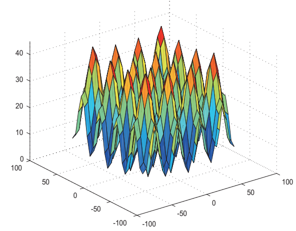
Quantic Function



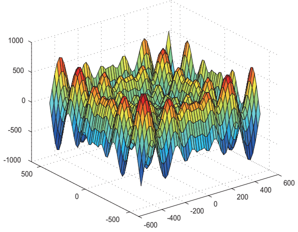




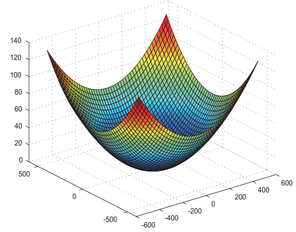
Rastrigin Function



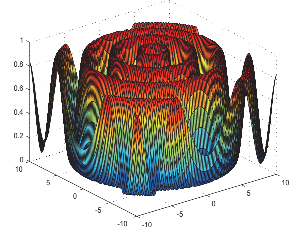
Schwefel Function



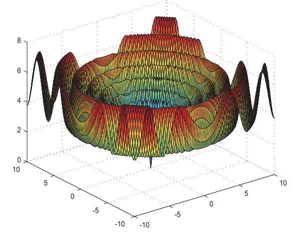
Griewank Function



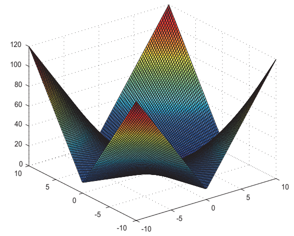
Schaffer 1 Function



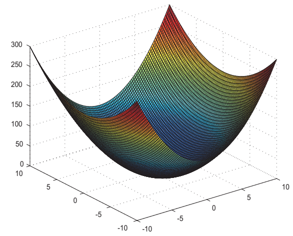
Schaffer 2 Function



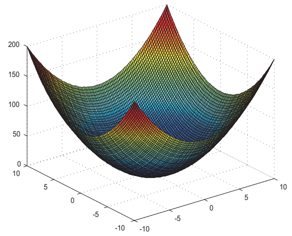
Schwefel 2.22 Function



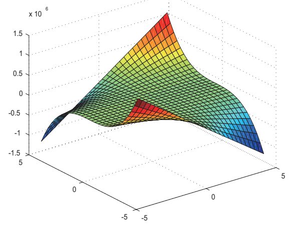
Schwefel 1.2 Function



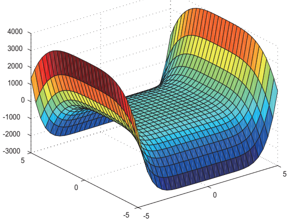
Step Function



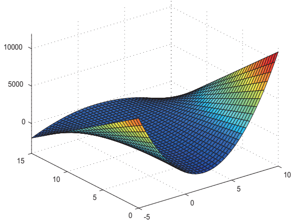
Beale Function



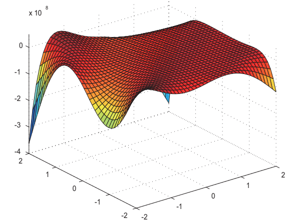
Camel-Back Function



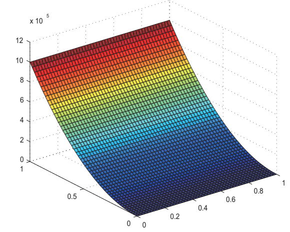
Branin Function

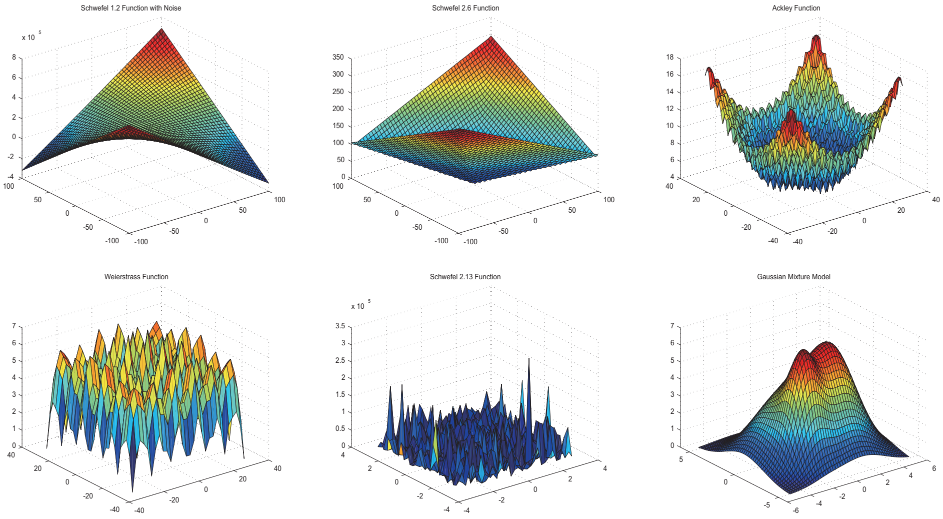


Goldstein-Price Function



Elliptic Function




Schwefel 1.2 Function with Noise
x 10 5
8
6
4
2
0
-2
-4
100
50
0
-50
-100
-100
-50
0
50
100
Schwefel 2.6 Function
350
300
250
200
150
100
50
0
100
50
0
-50
-100
-100
-50
0
50
100
Ackley Function
18
16
14
12
10
8
6
4
40
20
0
-20
-40
-40
-20
0
20
40
Weierstrass Function
7
6
5
4
3
2
1
0
40
20
0
-20
-40
-40
-20
0
20
40
Schwefel 2.13 Function
x 10 5
3.5
3
2.5
2
1.5
1
0.5
0
4
2
0
-2
-4
-4
-2
0
2
4
Gaussian Mixture Model
7
6
5
4
3
2
1
0
5
0
-5
-6
-4
-2
0
2
4
6