MULTIOBJECTIVE DIFFERENTIAL EVOLUTION BASED ON FUZZY PERFORMANCE FEEDBACK: SOFT CONSTRAINT HANDLING AND ITS APPLICATION IN ANTENNA DESIGNS

By

CHATKAEW JARIYATANTIWAIT

Bachelor of Engineering in Electrical Engineering
Chiangmai University
Chiang Mai, Thailand
1997

Master of Engineering in Electrical Engineering
Chiangmai University
Chiang Mai, Thailand
2002

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
December, 2015

MULTIOBJECTIVE DIFFERENTIAL EVOLUTION

BASED ON FUZZY PERFORMANCE FEEDBACK:

SOFT CONSTRAINT HANDLING AND ITS

APPLICATION IN ANTENNA DESIGNS

Dissertation Approved:

Dr. Gary G. Yen

Dissertation Adviser

Dr. Martin M. Hagan

Dr. Qi Cheng

Dr. R. Russell Rhinehart

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude and appreciation to my advisor Dr. Gary G. Yen for the continuous guidance, mentorship, patience and invaluable help throughout my Ph.D. study. His dedication and experience has helped in refining my attitude toward this research and in future work. I would also like to thank him for helping me out with the utmost patience in writing thesis and guiding me on presentation skills. These learning experiences benefit me all my life.

I would like to thank my committee members, Dr. Martin M. Hagan, Dr. Qi Cheng, and Dr. R. Russell Rhinehart for their timely advice, discussions and suggestions. Dr. Hagan's neural networks and Dr. Rhinehart's optimization applications classes help me very much in my research.

Many thanks to my lab mates at the Intelligent Systems and Control Laboratory (ISCL): Zhenan He, Bin Ha, Weiwei Zhang, Wei-Feng Gao, Dr. Wang Hu, and Jixiang Cheng for the constructive discussions, useful suggestions and benchmark source codes during this research.

I also would like to thank the Ministry of Science and Technology, Royal Thai Government for the financial supports during this study.

Last but not least, I am forever thankful to my mother (Suttinee), father (Surachai), Pailin, Akkasit for their unconditional love and supports. I also thank my mother-in-law and brother-in-law who take care of everything for me in Thailand during my graduate work here. I would like to thank my husband and my daughter for their love, supports and patience.

Name: CHATKAEW JARIYATANTIWAIT

Date of Degree: DECEMBER, 2015

Title of Study: MULTIOBJECTIVE DIFFERENTIAL EVOLUTION BASED ON FUZZY PERFORMANCE FEEDBACK: SOFT CONSTRAINT HANDLING AND ITS APPLICATION IN ANTENNA DESIGNS

Major Field: ELECTRICAL ENGINEERING

Abstract: The recently emerging Differential Evolution is considered one of the most powerful tools for solving optimization problems. It is a stochastic population-based search approach for optimization over the continuous space. The main advantages of differential evolution are simplicity, robustness and high speed of convergence. Differential evolution is attractive to researchers all over the world as evidenced by recent publications. There are many variants of differential evolution proposed by researchers and differential evolution algorithms are continuously improved in its performance. Performance of differential evolution algorithms depend on the control parameters setting which are problem dependent and time-consuming task. This study proposed a Fuzzy-based Multiobjective Differential Evolution (FMDE) that exploits three performance metrics, specifically hypervolume, spacing, and maximum spread, to measure the state of the evolution process. We apply the fuzzy inference rules to these metrics in order to adaptively adjust the associated control parameters of the chosen mutation strategy used in this algorithm. The proposed FMDE is evaluated on the well-known ZDT, DTLZ, and WFG benchmark test suites. The experimental results show that FMDE is competitive with respect to the chosen state-of-the-art multiobjective evolutionary algorithms. The advanced version of FMDE with adaptive crossover rate (AFMDE) is proposed. The proof of concept AFMDE is then applied specifically to the designs of microstrip antenna array. Furthermore, the soft constraint handling technique incorporates with AFMDE is proposed. Soft constraint AFMDE is evaluated on the benchmark constrained problems. AFMDE with soft constraint handling technique is applied to the constrained non-uniform circular antenna array design problem as a case study.

TABLE OF CONTENTS

Chapter Page

Chapter Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

CHAPTER I

INTRODUCTION

1.1 Motivations

In our everyday lives, we encounter problems that demand us to search for the best possible solutions. For example, planning monthly expenditure or buying the maximum amount of food given a limit budget. These problems can be formulated as optimization problems. The goal of the optimization problem can be described by a mathematical model as a number of objective functions. An optimization problem can be categorized by the number of objective functions. If the problem considers only one objective function, it is classified as a single objective optimization problem (SOP). If the problem involves more than one objective function, it is classified as a multiobjective optimization problem (MOP). However, most of our real-world problems are often MOPs. These MOPs' objectives usually conflict with each other. For instance, minimizing the side lobe level while maximizing the gain of an antenna is mandatory to achieve an optimal performance. If we found an optimized solution for the side lobe level, it may come at the cost of degraded gain. Ideally, the optimizers that use to solve MOPs should find a set of trade-off optimal solutions, and the decision maker will choose one solution from the set by using the high-level qualitative consideration.

The classical principle to tackle MOPs is often converting the MOP at hand to a SOP by combining all objective functions into one objective function, and then optimizes that new function as it was a SOP. The single objective optimizer can then be used to solve the problem.

There are some example methods that follow this principle [1-2] such as weighted sum method, ε-constraint method, goal programming, etc. These classical methods require *a priori* knowledge of the problem domain that is usually not available under the real world complications. In addition, we can obtain only one solution from these methods in a given single run. If we want to find *N* solutions (as same as the output of an ideal MOP optimizer), we need to run the optimizer *N* times and change the parameter setting for each single run. Hence, these methods are not efficient for solving MOPs. Hence, researchers have been developing the MOP optimizers that treat each objective equally and produce a set of optimal solutions in a single run. According to Price, Storn, and Lampinen [3], the MOP optimization techniques are broadly classified based on their objective function derivation properties: derivative-based (gradient based) and derivative-free method as shown in Table 1.1.

Table 1.1 A Classification of optimization approaches

Derivative property	**Single-point**	**Multi-point**
Derivative-based (gradient-based)	Steepest descent, Conjugate gradient, Quasi-Newton, etc.	Multi-start and clustering techniques etc.
Derivative-free (direct search)	Random walk, Hooke- Jeeves, etc.	Nelder-Mead, Evolutionary algorithms, Genetic algorithms, Differential evolution, etc.

Most real world MOPs' characteristics are in high dimension, non-differentiation, discontinuity, multimodality and/or NP-complete. The derivative-based approaches cannot be used, and some classical direct search techniques cannot effectively solve these MOPs. Therefore, researchers have been developing optimizers based on nature-inspiration, so called the evolutionary computation.

Evolutionary computation is a stochastic, population-based search algorithm inspired by natural evolution [4]. Natural evolution is mostly determined by natural selection or different individuals competing for resources in the environment. Those fitter individuals of the population are more likely to survive and propagate their genetic materials through the reproduction. The fittest individuals will survive till the end of the evolution process. The evolutionary computation algorithms simulate the nature evolution process in order to search for the fittest individuals which are the optimal solutions of the optimization problems. A few decades ago, Holland [5] proposed Genetic Algorithm (GA) which simulate Darwinian evolution to solve practical optimization problems; Fogel [6] introduced Evolutionary Programming (EP); Evolution Strategies (ES) was proposed by Rechenberg and Schwefel [7-8]; Eberhart and Kennedy [96] introduced the Particle Swarm Optimization (PSO). These algorithms play important roles in solving challenging MOPs. Later on, there are additional nature-inspired algorithms developed such as artificial immune systems [9], harmony search [10], memetic and cultural algorithms [11], etc. These approaches are unified as different representatives of evolutionary computation.

Around a decade ago, Differential Evolution (DE) was proposed by Storn and Price in 1995 as a new evolutionary algorithm (EA) [12]. It is a stochastic population-based search approach for optimization over the continuous space [13-14]. DE is considered one of the most powerful tools for solving optimization problems. DE can handle mixed-type variables, constraints, multimodality, and also multiple-objective in nature. Implementing DE is easier than other EAs such as Genetic Algorithm (GA) even for a beginner in the optimization field. In addition, control parameters in the design of a DE are very few. The powerful performance of DE attracts researchers to develop DE variants for solving optimization problems. Thus, a multiobjective differential evolution algorithm based on fuzzy performance feedback (FMDE) is proposed in this study in order to handle MOPs. Furthermore, FMDE is applied to a practical engineering problem: an antenna design problem specifically 5 by 5 microstrip antenna arrays for 12.5 GHz broadcasting satellite as a case study.

Although the above discussion emphasizes on unconstrained MOPs, there are other category of optimization problem: constrained optimization problems (COPs). In real world applications, many optimization problems are COPs. If a COP involves more than one objective function and a number of constraints, it will be called a constrained multiobjective optimization problem (CMOP). The constraints can be classified into two types: hard and soft constraints. Hard constraints are the constraint that any violations cause the system failure [96-97]. On the other hand, the soft constraints can be relaxed to some extent if the violations of them do not compromise the purpose of the requirements. In some cases, relaxation of the soft constraints can improve some objective functions. Therefore, a soft constraint handling is proposed in this thesis. FMDE incorporated with the soft constraint handling is developed to solve CMOPs. The soft constrained FMDE will be applied to a constrained non-uniform circular antenna arrays design as a case study.

1.2 Statement of Problem

The mutation strategy and the control parameters, namely scaling factor (F), crossover rate (CR), and population size (NP), play the major roles in the success of a DE. Choosing the appropriate mutation operator and parameter values for a particular problem is a difficult task because it is a problem dependent, time-consuming, and trial-and-error process. In addition, balancing the exploration and exploitation throughout the search is the key to the success of an EA. During the evolution process we may need different mutation strategy and parameter values at different stages. In the beginning of the evolution we need a higher degree of exploration than exploitation in order to search larger regions in the space. We may choose the mutation operator that possesses high exploration ability and the control parameter that promote the diversity. However, near the end of the evolution we need to emphasize on the local search that is exploitation. The mutation operator that favors local search should be chosen along with the control parameters that emphasize the exploitation. If we know the state of the evolution process, we may decide whether we should emphasize on exploration or exploitation, and choose suitable parameter values or the mutation strategies.

One possible way that we can observe the status of the evolving process is utilizing the performance metrics. Most of the performance metrics are calculated at the end of the evolution in order to assess the quality of the obtained nondominated front. For instance, generational distance needs the complete knowledge about the true Pareto front in order for calculation. We cannot assume the true Pareto front is available during the evolution search. The quality of the population can be measured by three properties of the obtained nondominated front [15], namely, the convergence, uniform distribution, and extensiveness. Although there are some proposed running performance metrics to measure the quality of the population on the fly, there are very few choices to allow us to measure the convergence, uniform distribution, and extensiveness of the population. Hence, we exploit three performance metrics to measure the three properties of the obtained nondominated solutions. The proposed fuzzy-based multiobjective differential evolution (FMDE) [95] utilizes hypervolume, spacing, and maximum spread as the input to the fuzzy inference rules that adaptively adjust the control parameters for the mutation scheme which is the greedy factor and the diversity factor every generation in order to balance the exploration and exploitation abilities of the population during the search process. Furthermore, the advanced version of FMDE (AFMDE) is introduced. AFMDE incorporate the adaptive *CR* by fuzzy rules to be discussed in Chapter IV.

One challenging practical engineering problem is antenna design. There are various antenna types [16] such as wire antennas, aperture antennas, array antennas, etc. The very popular one among them is microstrip antenna. Microstrip antennas are light weight, low-profile, conformal configuration, easy to manufacture, and low production cost by using printed-circuit technology. They are being used in widespread applications, for example, radars, missiles, aircraft, satellite and mobile communication, etc. [17-18]. Generally, designers will design microstrip antenna configuration, in order to achieve the optimal performance. The goals specified as maximizing antenna gain (*G*) while minimizing side lobe level (*SLL*), and minimized reflection coefficient. Antenna design problems can be formulated as MOPs. The gain, side lobe level, reflection coefficient are possible objectives whereas the antenna configuration is the decision variables. Since AFMDE is a multiobjective optimizer, therefore it can be applied to antenna design problems. The design of 5 by 5 microstrip antenna array for 12.5 GHz broadcasting satellite service using AFMDE is studies as a case study to demonstrate the effectiveness and efficiency of the design process based on FMDE.

Furthermore, the soft constrained AFMDE is developed to solve CMOPs. In order to prove the concept of the proposed soft constrained AFMDE, solving the constrained non-uniform circular antenna arrays design is chosen as a case study. Non-uniform circular antenna arrays are popular in mobile and wireless communications such as air and space navigation, radar, sonar and other applications [16] [93]. The antenna array provides the higher directive radiation pattern than a single element antenna. The design of antenna array is to determine the array geometry and the excitation at each array element. The goals are to minimize the *SLL*, maximizing directivity (D), and minimizing the first null beamwidth (FNBW) amplitude subject to the limitation of the array size (circumference) and FNBW requirement.

1.3 Contributions

The contributions of this dissertation are summarized as the follows:

- Develop a multiobjective differential evolution based on the fuzzy performance metric feedback (FMDE). This algorithm adaptively adjusts the control parameters (i.e., the greedy factor (γ) and scaling factor (F)) of differential evolution and balances the exploration and exploitation of the algorithm throughout the search process.

- The advanced version of FMDE (AFMDE) is proposed by adapting the crossover rate (CR) in concurrent with γ and scaling factor F based on fuzzy inference rules and performance metrics feedback.
- Develop a soft constrained AFMDE (SFMDE) to handle constrained multiobjective optimization problems. The soft constraint handling exploits the relaxation of constraints and preference rule strategy. The proposed soft constraint handling is integrated with AFMDE in order to solve CMOPs.
- Validate the proposed FMDE variants through antenna designs.

1.4 Document Organization

This report comprises of seven chapters. Chapter II presents the background and literature review of differential evolution.

Chapter III provides a comparison between the proposed FMDE and state-of-the-art MOEAs and other DE variants for solving the benchmark multiobjective optimization problems.

Chapter IV provides a comparison between the advanced version of FMDE which adaptively adjust CR and state-of-the-art MOEAs and other DE variants for solving the benchmark multiobjective optimization problems.

In Chapter V, AFMDE was applied to the microstrip antenna design problem in order to prove the concept on a practical engineering problem. Since, the objective evaluation is expensive. Hence, a surrogate model is utilized as the objective model.

In Chapter VI, a soft constraint handling is proposed. Then, AFMDE was integrated with this soft constraint handling and is tested on the benchmark CMOPs. In order to prove the concept, the soft constraint handling AFMDE is applied to the constrained non-uniform circular antenna array design problem.

Conclusions are discussed in Chapter VII. Summary of the main contributions of this report are reviewed and limitations of the proposed works are identified and the future research directions are highlighted.

CHAPTER II

LITERATURE REVIEW

Differential Evolution (DE) was proposed by Storn and Price in 1995 as a new evolutionary algorithm (EA) [12]. DE is a stochastic population-based search approach for optimization over the continuous space [14]. The main advantages of DE are simplicity, robustness and high speed of convergence; make DE one of the most powerful algorithms for global optimization. The fundamentals and advantages of DE, comparison with other optimization algorithms, and the surveys of DE variants are detailed in this chapter.

2.1 Introduction

In human society, everyone is different. They are different lives, different minds, different expertise and so on. When people are together, they form a society and the social behavior will automatically emerge. One of the most influential features of social behavior is *collective intelligence.* That means integration of the differences in a single whole in order to be more powerful, more efficient, and more intelligent, and the more, the better [19].

DE is classified as one type of EAs, DE is unique, since DE utilizes both collective intelligence and evolution, *"the intelligent use of the individual differences."*

The differential mutation is the key to success, dated back in 1994 when K. Price invented Genetic Annealing [20] and soon after that R. Storn cooperated with him in order to solve the Tchebychev polynomial fitting problem by genetic annealing. This problem is formulated within a continuous space, so they changed from bit-string to floating-point encoding, and from logical operators to arithmetic ones. While they were doing some experiments, they discovered the *differential mutation.* They also observed that using the composition of differential mutation, discrete recombination, and pair-wise selection, the annealing mechanism is no longer needed. They removed the annealing factor and then DE was born. They first reported the DE in the ICIS technical report [12] and other publications later [13, 14]. Since then, DE has attracted researchers all over the world as a powerful global optimizer.

Even though DE is first introduced as the global optimizer over the continuous space, DE is also able to handle mixed-type variables, constrained, multimodal and multiobjective optimization problems [21]. In addition, researchers from several areas of science and engineering have been applying DE to solve optimization problems in their own fields. Summary of some applications of DE is shown in Table 2.1. Details on DE are explained in Section 2.2.

2.2 Differential Evolution (DE) Fundamentals

The original version of DE [41] is described in this section. A flowchart of the classical DE algorithm is shown in Figure 2.1.

Like other evolutionary algorithms, it starts with the randomly initializing a population in the search space. In DE community, an individual of the population is called a parameter vector; *NP* parameter vectors form a population. Then the population enters the evolution loop: mutation, crossover, and selection operations. These three operations will be repeated until the stopping criterion is met. The details on each stage of DE will be explained in Subsections 2.2.1 – 2.2.5.

Table 2.1 Summary of some applications of DE

Areas of Applications	**Examples of applications**
Control Systems and Robotics	Controller design and tuning [22,23] Robot motion planning and navigation [24,25] Nonlinear system control [26]
Scheduling	Plant scheduling and planning [27] Traveling salesman problem [28]
Chemical Engineering	Chemical process synthesis and design [29] Parameter estimation of chemical process [30]
Bioinformatics	Gene regulatory networks [31,32] Protein folding [33]
Neural Networks	Training of wavelet neural networks [34] Training of feed forward neural networks [35]
Electromagnetism, Propagation and Microwave Engineering	Electromagnetic inverse scattering [36] Antenna array design [37] Microwave filter design [38]
Image Processing	Automatic clustering [39] Image watermarking [40]

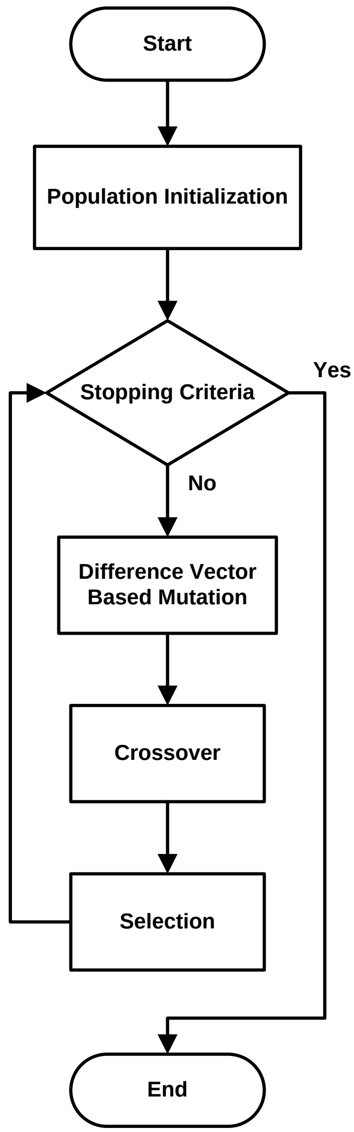


Figure 2.1 The classical DE flowchart.

2.2.1 The population

The DE population contains *NP* *D*-dimensional parameter vectors. These vectors are real numbers. The current population is $\mathbf{P}_{\mathbf{x},G}$ composed of vectors $\mathbf{x}_{i,G}$ as

$$\mathbf{P}_{\mathbf{x},G} = \left\{ \mathbf{x}_{i,G} \right\}, \quad i = 1, 2, ..., NP, \quad G = 1, 2, ..., g_{\max,}$$

$$\mathbf{x}_{i,G} = \left\{ x_{j,i,G} \right\}, \quad j = 1, 2, ..., D \qquad (2.1)$$

where *NP* is the number of population vectors, *G* is the generation number, and *D* is the dimensionality of the vector.

2.2.2 The population initialization

The population is initialized by

$$x_{i,j} = rand_j[0,1).(b_{j,U} - b_{j,L}) + b_{j,L} \qquad (2.2)$$

where $\mathbf{b}_L$ and $\mathbf{b}_U$ is the lower and upper bounds of the vectors $\mathbf{x}_{i,j}$ at generation *G*=0, $rand_j$ [0,1) is uniformly distributed random number within the range [0,1). This number is generated for each element *j* of vectors.

2.2.3 Mutation

After population initialization, DE uses the mutation to generate a candidate vector called the mutant vector $\mathbf{v}_{i,G}$ with respect to the target vector $\mathbf{x}_{i,G}$ by adding the base vector $\mathbf{x}_{r1,G}$ to weighted difference vectors. This key operation of DE is shown in (2.3)

$$\text{DE/rand/1: } \mathbf{v}_{i,G} = \mathbf{x}_{r1,G} + F\left(\mathbf{x}_{r2,G} - \mathbf{x}_{r3,G}\right) \qquad (2.3)$$

The random integers i, $r1$, $r2$ and $r3$ are randomly generated and mutually exclusive within the range [1, NP], used as indices to index the current parent vectors. As can be seen from (2.3), we need at least three vectors in a population. F is a positive scaling factor, which manages the trade-off between exploration and exploitation. Originally, $F \in (0,1+)$ means while there is no upper limit on F, effective values are seldom greater than 1.0 [3]. Storn and Price suggested that F should be 0.8 for general problems [12]. The original mutation strategy is called DE/rand/1 as DE/x/y/z represents DE/base vector/number of difference vectors/crossover. There are two types of crossover operators, namely, binomial and exponential crossovers. Therefore, DE/rand/1/bin is the DE/rand/1 mutation scheme with the binomial crossover and DE/rand/1/exp is the DE/rand/1 mutation scheme with the exponential crossover. A pictorial example of DE/rand/1 is illustrated in Figure 2.2

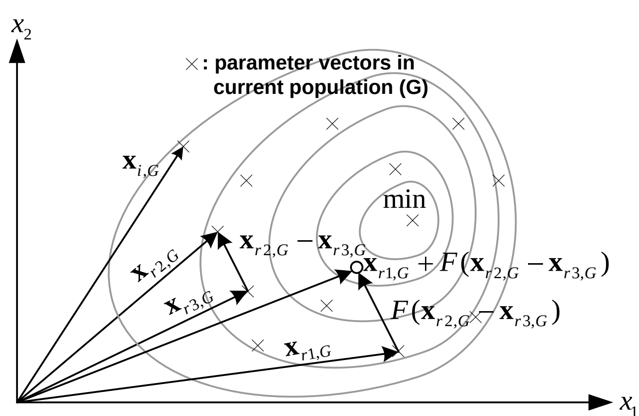


Figure 2.2 Illustration of DE/rand/1 scheme in 2-D space.

There are many mutation strategy proposed by many researchers. The most commonly used mutation operations are [42],

DE/rand/2: $\mathbf{v}_{i,G} = \mathbf{x}_{r1,G} + F(\mathbf{x}_{r2,G} - \mathbf{x}_{r3,G}) + F(\mathbf{x}_{r4,G} - \mathbf{x}_{r5,G})$ (2.4)

DE/best/1: $\mathbf{v}_{i,G} = \mathbf{x}_{best,G} + F(\mathbf{x}_{r1,G} - \mathbf{x}_{r2,G})$ (2.5)

DE/best/2: $\mathbf{v}_{i,G} = \mathbf{x}_{best,G} + F(\mathbf{x}_{r1,G} - \mathbf{x}_{r2,G}) + F(\mathbf{x}_{r3,G} - \mathbf{x}_{r4,G})$ (2.6)

DE/current-to-best/2: $\mathbf{v}_{i,G} = \mathbf{x}_{i,G} + F(\mathbf{x}_{best,G} - \mathbf{x}_{i,G}) + F(\mathbf{x}_{r1,G} - \mathbf{x}_{r2,G})$ (2.7)

DE/rand-to-best/2: $\mathbf{v}_{i,G} = \mathbf{x}_{i,G} + F(\mathbf{x}_{best,G} - \mathbf{x}_{r2,G}) + F(\mathbf{x}_{r3,G} - \mathbf{x}_{r4,G})$ (2.8)

Trigonometric mutation [43]:

$$\begin{aligned} \mathbf{v}_{i,G} = {} & (\mathbf{x}_{r1,G} + \mathbf{x}_{r2,G} + \mathbf{x}_{r3,G})/3 + (p_2 - p_1)(\mathbf{x}_{r1,G} - \mathbf{x}_{r2,G}) \\ & + (p_3 - p_2)(\mathbf{x}_{r2,G} - \mathbf{x}_{r3,G}) + (p_1 - p_3)(\mathbf{x}_{r3,G} - \mathbf{x}_{r1,G}) \end{aligned} \quad (2.9)$$

where $p_1 = \dfrac{|f(\mathbf{x}_{r1,G})|}{p'}, p_1 = \dfrac{|f(\mathbf{x}_{r2,G})|}{p'}, p_1 = \dfrac{|f(\mathbf{x}_{r3,G})|}{p'},\quad p' = |f(\mathbf{x}_{r1,G})| + |f(\mathbf{x}_{r2,G})| + |f(\mathbf{x}_{r3,G})|$ and $f(\mathbf{x})$ is the objective function. The indices $r1$, $r2$, …, $r5$ are randomly generated integers within the range [1,NP] and they are mutually exclusive and different from i. For each mutant vector, we must generate new indices. $\mathbf{x}_{best,G}$ is the individual vector that provides the best objective value in the population in the generation G.

2.2.4 Crossover

After we get the mutant vectors from the mutation operation, we perform the crossover operation in order to increase the potential diversity of the population. Crossover is applied to each pair of the target vector and it’s mutant vector then we obtain a trial vector. The DE family algorithms can use two types of crossover schemes: the binomial (uniform) crossover and the exponential crossover [44].

2.2.4.1 Binomial crossover

The binomial crossover is the discrete recombination operator by (2.10):

$$\mathbf{u}_{i,G} = \{u_{j,i,G}\} = \begin{cases} v_{j,i,G} & \text{if } rand_j(0,1) \le CR \text{ or } j = j_{rand} \\ x_{j,i,G} & otherwise \end{cases}. \tag{2.10}$$

The crossover rate or so called crossover probability $CR \in [0,1]$ is a user-defined constant that controls the fraction of parameter values copied from the mutant vector. *CR* is suggested to be 0.8-1 [20]. Here the crossover operator is uniform in the sense that each parameter of the mutant vector, regardless of its location has the same probability, *CR*, of inheriting its value from a given to the trial vector. For this reason, uniform crossover does not exhibit a representation bias. A pictorial example for binomial crossover is illustrated by Figure 2.3. The target vector $\mathbf{x}_{i,G}$ exchanges its parameters with the mutant vector to $\mathbf{v}_{i,G}$ in order to form the trial vector $\mathbf{u}_{i,G}$.

2.2.4.2 Exponential crossover

DE exponential crossover [44] is functionally equivalent to two-point crossover in GA. In exponential crossover, we begins with randomly choosing an integer *n* from the interval [1, *D*] as a starting point in the target vector, from where the exchange of parameters with the mutant vector starts. Another integer *L* is also randomly chosen from the range [1, *D*] depending on the crossover probability *CR* according to the pseudo code:

```
L=0;
DO
{
 L=L+1;
}WHILE (((rand(0,1)<CR) AND (L<D))
```

After choosing n and L the trial vector is generated by

$$\mathbf{u}_{i,G} = \{u_{j,i,G}\} \\ u_{j,i,G} = \begin{cases} v_{j,i,G}, & j = \langle n \rangle_D, \langle n+1 \rangle_D, \ldots, \langle n+L-1 \rangle_D \\ x_{j,i,G}, & \text{all other } j \in [1, D] \end{cases} \quad (2.11)$$

where $\langle\ \rangle_D$ denote a modulo function with modulus D.

The parameters of the trial vector $\mathbf{u}_{i,G}$ are inherited from the corresponding mutant vector $\mathbf{v}_{i,G}$ starting from index n till the first time that $rand(0,1) > CR$. All the remaining parameters of the trial vector are inherited from the corresponding target vector. An example of exponential crossover is shown in Figure 2.4. In this example, exponential crossover starts with $n = 3$ which is randomly chosen and copied the parameter of the mutant vector to the trial vector until the first occurrence of $rand(0,1) > CR$, the rest of parameters are copied from the target vector to the trial vector.

2.2.5 Selection

The selection stage is to determine whether the target or the trial vector survives to the next generation. The selection operator is a comparison between the trial vector $\mathbf{u}_{i,G}$ and the target vector $\mathbf{x}_{i,G}$. For a minimization problem, if the objective value of the trial vector is lower than the target vector, the trial vector is the winner. Then the trial vector replaces the target vector and enters the next generation. By this method, DE more tightly integrates recombination and selection than other EAs as the following [3]:

$$\mathbf{x}_{i,G+1} = \begin{cases} \mathbf{u}_{i,G}, & \text{if } f(\mathbf{u}_{i,G}) \le f(\mathbf{x}_{i,G}) \\ \mathbf{x}_{i,G}, & \textit{otherwise} \end{cases} \tag{2.12}$$

In (2.12) the target vector is replaced by the trial vector even if both yields the same objective values. It is the feature of DE to move on a flat objective landscape over generations. For clarity, a numerical example of DE algorithm is demonstrated in Section 2.3.

In conclusion, there are three keys to DE success: spontaneous self-adaptability via difference vector mutation, diversity control by crossover, and continuous improvement by selection (elitism).

2.3 An Illustrative Examples of DE

In order to illustrate the steps of the DE algorithm, a numerical example is presented in Figure 2.5 [45] which demonstrated the procedure to generate one vector for the next generation. This problem is to minimize an objective function $f(X) = x_1 + x_2 + x_3 + x_4 + x_5$. The population size is 6. The scaling factor F and the crossover rate CR are 0.8 and 0.6, respectively. The objective value of each individual evaluated by $f(X)$ and is shown in the top cell of the corresponding vector. For the first target vector (individual 1), three randomly chosen vectors are individuals 2, 4 and 6 which produce the mutant vector by (2.3). Consequently, the crossover of the mutant and the target vector occurred. Parameter 1 and 5 are selected from the mutant vector and the remaining from the target vector. Then, the selection is the next step. In the selection stage, objective value (cost value) of the trial vector and the trial vector is compared. Since the target vector has a lower value, it is selected and copied to the next generation.

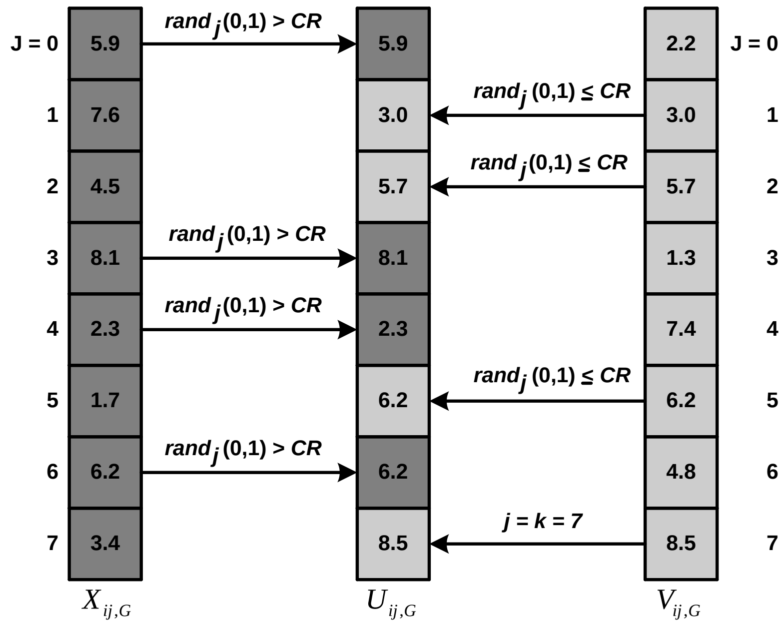


Figure 2.3 A pictorial example for binomial crossover.

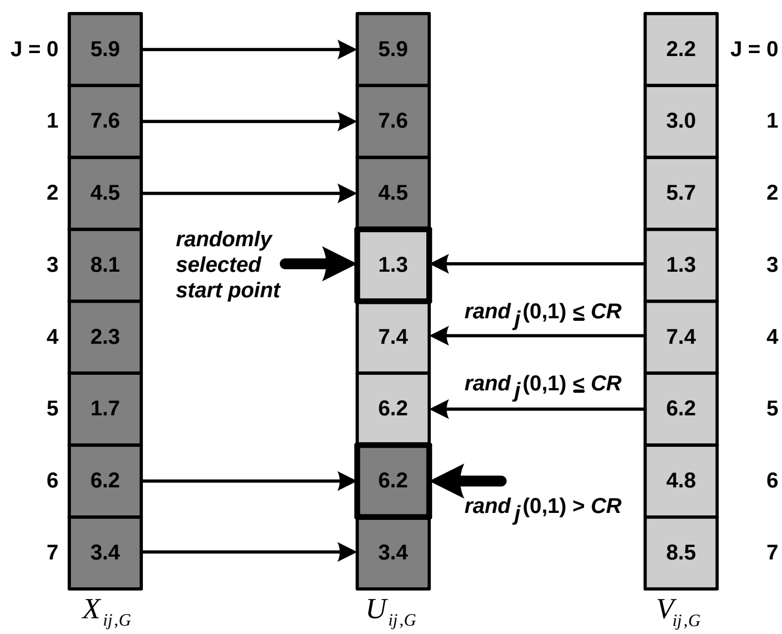


Figure 2.4 A pictorial example of exponential crossover.

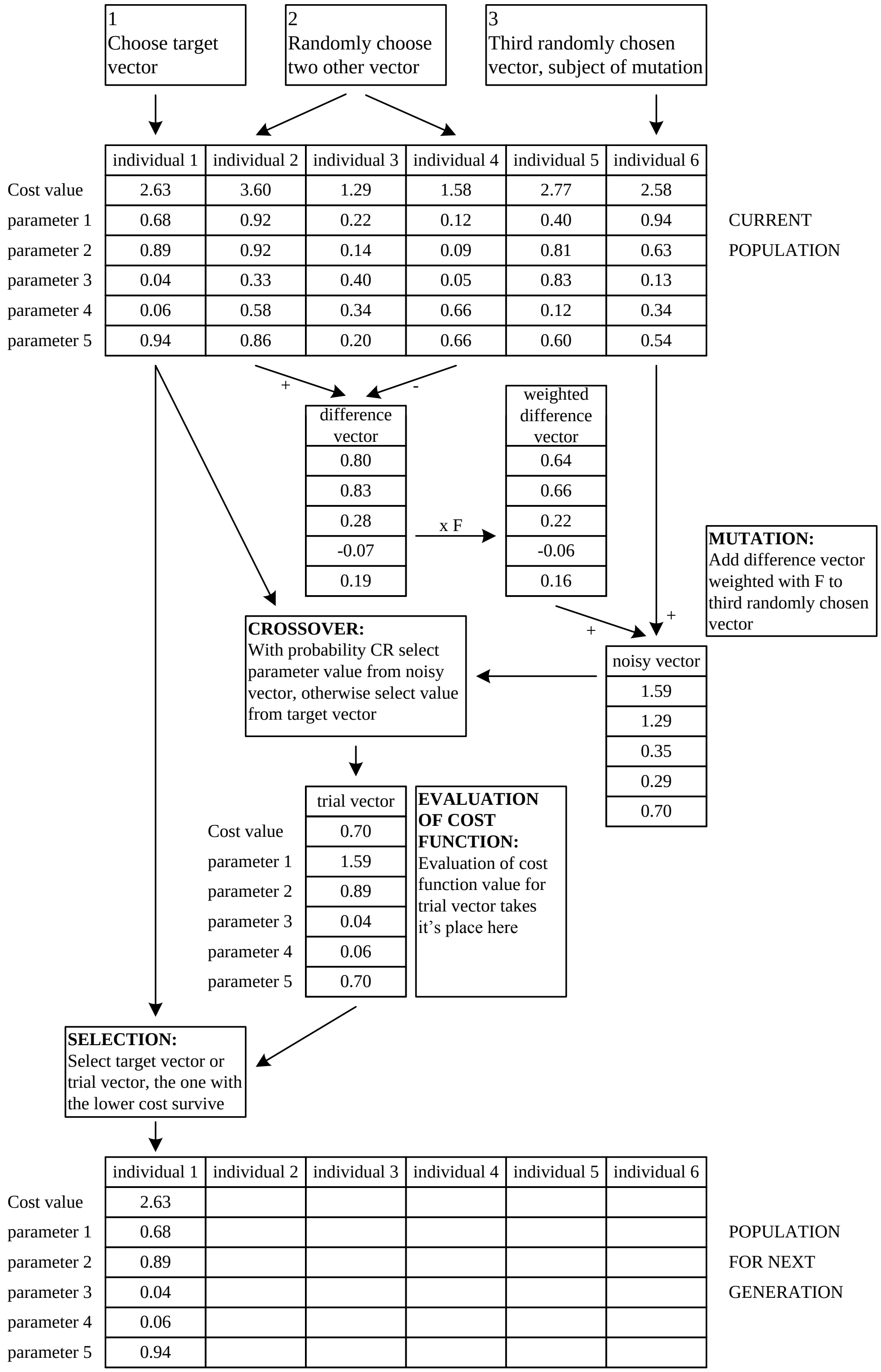


Figure 2.5 A numerical example of utilizing DE/rand/1/bin to minimize

$$f(X) = x_1 + x_2 + x_3 + x_4 + x_5 \, .$$

2.4 Advantages of DE

DE is one of the most powerful global optimizers, according to for the following arguments:

1. DE can handle mixed-type variables and constrained, multimodal and also multiple objective optimization problems. Hence, it is able to handle many types of optimization problems.

2. Its simplicity in implementation compared with other EAs.

3. The number of control parameters is very few. In classical DE, there are only three parameters- *CR*, *F* and *NP*.

4. Mutation operation using the difference vectors makes DE self-adaptive. It adapts the searching process to the objective landscape without the need of *a priori* knowledge about the problem characteristics.

2.5 Reviews of DE for Multiobjective Optimization

Since DE was first introduced to the computational intelligence society around a decade ago, it has been attractive to researchers all over the world to improve and utilize it to solve optimization problems. The review of significant literatures on multiple objective optimization using DE is shown here. The author categorizes the literatures based on the modification of classical DE as the following classes:

2.5.1 Problem formulation

Multiobjective optimization attempts to simultaneously minimize multiple objective functions. Without the loss of generosity, consider the multiobjective minimization problem (MOPs) [46] as:

$$\min_{\mathbf{x}\in\Re^D} \mathbf{F}(\mathbf{x}) = [F_1(\mathbf{x}), F_2(\mathbf{x}), ..., F_k(\mathbf{x})] \tag{2.19}$$

and the *n* decision variable bounds:

$$x_i^L \leq x_i \leq x_i^U, \quad i = 1, 2, ..., D \tag{2.20}$$

where $\mathbf{x} = [x_1, x_2, ..., x_n] \in \Re^D$.

The function F_i is known as the objective function or fitness function, and $F_i(\mathbf{x})$ is called the objective or fitness value of F_i. $\mathbf{x}$ represents a decision vector of n decision variables, where each decision variable is confined by a lower bound x_i^L and an upper bound x_i^U. The n variable bounds constrain a *decision space* or *search space*, $S \subseteq \Re^n$, and the k objective functions constitute an *objective space*, Z.

Decision vectors that minimize $\mathbf{F}(\mathbf{x})$ are also referred to as solutions. By duality principles, any objective function can be converted from minimization to maximization or vice versa as:

$$\left.\begin{array}{l} \max F_i(\mathbf{x}) = \min(-F_i(\mathbf{x})) \\ \min F_i(\mathbf{x}) = \max(-F_i(\mathbf{x})) \end{array}\right\} \tag{2.21}$$

2.5.2 Pareto optimization

In a single objective optimization problem, we search for the best possible solution or the global optimum. However, for MOPs, some objective functions conflict with the others, so we cannot optimize all objective functions simultaneously. Thus, for MOPs, there exist a set of optimal solutions, not a single optimal solution, under different trade-offs. The concepts of "*Pareto Dominance*" and "*Pareto Optimality*" are employed in order to obtain the set of optimal solutions.

Definition 2.1: *Pareto Dominance*

Consider a minimization problem, a decision vector $\mathbf{x}_a$ is said to dominate another decision vector $\mathbf{x}_b$, denoted by $\mathbf{x}_a \prec \mathbf{x}_b$, *iff*

1. $F_i(\mathbf{x}_a) \le F_i(\mathbf{x}_b)$ for all $i = 1, 2, ..., k$ and

2. $F_j(\mathbf{x}_a) < F_j(\mathbf{x}_b)$ for at least one $j = (1, 2, ..., k)$

Definition 2.2: *Nondominated Set*

Let P represent the set of decision vectors in the search space, $P \subseteq S$, the nondominated set are those decision vectors in *P* that are not dominated by any members of the set *P*.

Definition 2.3: Pareto Optimal Set

A decision vector $\mathbf{x}*$ is Pareto optimal if there exist no decision vector $\mathbf{x}_i$ for which $\mathbf{F}(\mathbf{x}_i)$ dominates $\mathbf{F}(\mathbf{x}^*)$. The collection of such decision vectors that is Pareto optimal is known as the *Pareto optimal set*. This means that each solution in this set holds equal importance and is a good compromise among the trade-off objectives. The resulted trade-off curve in the objective space that obtained from Pareto optimal set is called the *Pareto front*.

A pictorial example of bi-objective optimization problem is illustrated in Figure 2.14. From Figure 2.6(a), $\mathbf{x}_a$, $\mathbf{x}_b$, $\mathbf{x}_c$ and $\mathbf{x}_d$ are decision variables stay in the decision space. Their corresponding fitness values are $\mathbf{F}(\mathbf{x}_a)$, $\mathbf{F}(\mathbf{x}_b)$, $\mathbf{F}(\mathbf{x}_c)$ and $\mathbf{F}(\mathbf{x}_d)$ in the objective space are shown in Figure 2.6(b) and the *Pareto front* as well.

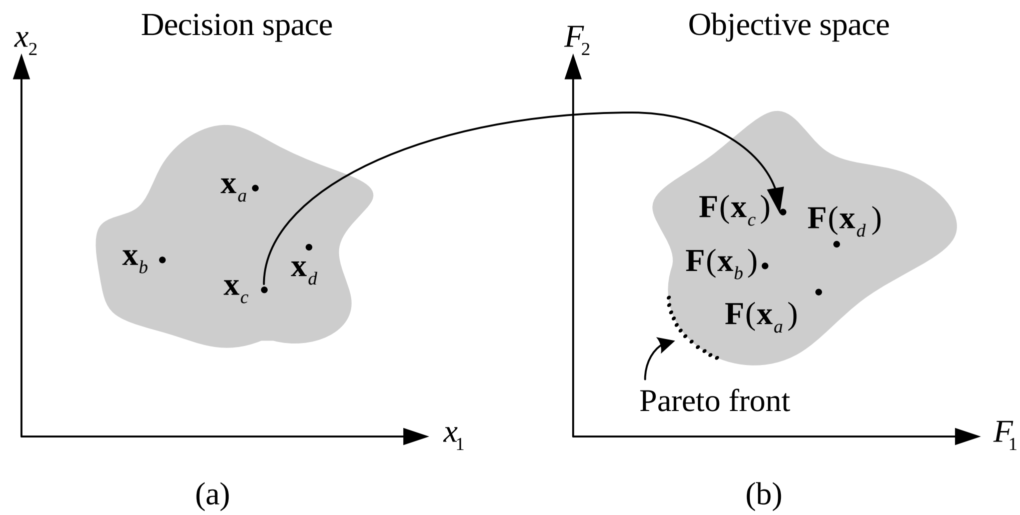


Figure 2.6 A pictorial example of bi-objective optimization problem.

2.5.3 DE for multiobjective optimization

The reviews of literature on multiobjective optimization using DE are shown here. The author categorizes the literatures based on the modification of classical DE as the following classes:

2.5.3.1 Multiobjective DE based on Pareto dominance

Abbass *et al*. [47] proposed the Pareto-frontier differential evolution (PDE) algorithm to solve MOPs. An initial population is randomly generated from a Gaussian distribution with mean 0.5 and standard deviation 0.15. All dominated solutions are removed from the population. The remains are only non-dominated solutions and will be select as the parents. The reproduction is DE/rand/1/bin. PDE also fixes the size of non-dominated solutions. If the size is over the predefined size, the crowded solutions will be removed based on a distance metric. The scale factor *F* is generated from a Gaussian distribution with zero mean and unity standard deviation. *CR* is set by empirical study and they found that in order to obtain a large number of non-dominated solutions, *CR* should be small. Later, Abbass [48] modified PDE to be self-adaptive.

The control parameters, F and CR, are adaptive. They are inherited from the parent in the same way crossover is undertaken for the decision variables. The algorithm is called the self-adaptive Pareto differential evolution algorithm (SPDE).

Xue et al. [49] presented the Pareto-based multi-objective differential evolution (MODE). This algorithm implements the selection of the best individual for the mutation operation. The non-dominated solutions in the population are identified in each generation. For mutation of an individual p, it is identified first if it is a dominated one. If p is a dominated solution, a set of individuals that dominate p will be identified and randomly choose a solution from the set as p_{best}. (p_{best}-p) is the difference vector for mutation operation. Otherwise, p is p_{best} itself and the difference vector will be zero and has no effect. The major difference from single objective DE is that the best individual is varying rather than fixed for the reproduction stage. Also, they adopt $(\mu + \lambda)$ selection (combined parents and offspring population), Pareto ranking and crowding distance in order to produce and maintain diversity. They also applied MODE to the decision support for a design-supplier-manufacturing planning problem [50].

Kukkonen and Lampinen [51] developed the generalized differential evolution version three (GDE3) to solve MOPs with constraints. GDE3 combines the Pareto-based differential evolution with the previous GDE version. If the problem is unconstrained single objective optimization, GDE3 is exactly the same as the original DE. This version uses a growing population and non-dominated sorting as same as NSGA-II [52] to obtain improved diversity and make the algorithm less sensitive to the control parameters. They also studied the effect of control parameters on GDE3 [53] and found that GDE3 is more robust than its previous version. The algorithm performed worse for the rotated multiobjective optimization problems as documented in [54]. Application of GDE3 can also be found in [55].

Robic and Filipic [56] proposed a differential evolution for multiobjective optimization (DEMO). It combines the advantages of DE with the mechanisms of Pareto-based ranking and crowding distance sorting. DEMO maintains only one population and the population size can be extended during each generation. By the end of the generation, the extended population will be truncated to the fixed population size by using non-dominated sorting and crowding distance. Another major mechanism is the immediate replacement of the parent individual with the candidate that dominates it. This mechanism promotes the elitism and makes DEMO converge faster.

Iorio and Li [57] presented the non-dominated sorting DE (NSDE). They modified NSGA-II in the reproduction stage. Crossover and mutation operators of NSGA-II are replaced by crossover and mutation operators of DE, respectively.

2.5.3.2 Non-Pareto based multiobjective DE

Li and Zhang [58] proposed a multiobjective DE based on decomposition (MOEA/D-DE) with variable linkages. They use a decomposition approach for converting approximation of the Pareto front into a number of single objective optimization problems. Then apply the DE/rand/1/bin to generate the trial solutions, and exploits the neighborhood relationship among the subproblems for making its search effectively and efficiently. This method does not employ the Pareto concepts.

2.5.3.3 Self-adaptive multiobjective DE

Huang *et al.* [59] extended the SaDE [60] to solve MOPs. They named the algorithm as the multi-objective SaDE algorithm (MOSaDE). The algorithm automatically adapts the trial vector generation strategies and their associated parameters according to their previous experience of generating promising solutions as same as SaDE. However, MOSaDE uses non-domination sorting and crowding in evaluation process. Later, Huang *et al*. [61] modified

MOSaDE in order to learn the suitable crossover rate and mutation strategies for each objective separately in MOPs.

Zamuda *et al.* [62] proposed differential evolution for multiobjective optimization with self-adaptation (DEMOwSA). They extended the DEMO by incorporating the self-adaptive control parameters *F* and *CR*. *F* and *CR* will be encoded into the decision variables and simultaneously evolved with the population.

Zhang and Sanderson [63] proposed the self-adaptive multiobjective DE with direction information provided by archived inferior solutions (JADE2) which is extended from JADE. JADE2 incorporated the self-adaption of *F* and *CR* and selection scheme based on Pareto dominance and crowding density. Adaptation of *F* and *CR* is based on the principle that the better values of control parameters tend to generate individuals that are more likely to survive and should be propagated.

2.5.3.4 The multiobjective DE based on opposite operation

Dong and Wang [64] proposed the DE based on opposite operation to solve the MOPs. The proposed algorithm utilizes the idea of ODE [65]: population initialization and generation jumping dynamically based on the number of non-dominated solutions generated by DE. Also, the external archive is to store the non-dominated solutions that are sorted by the same mechanism as NSGA-II.

2.6 Previous Works in Indicator-based MOEAs

The performance of MOEAs can also be enhanced through monitoring the progress of evolution process by performance metrics. These designs are often called Indicator-based MOEAs. Some of the most popular ones are summarized below.

HypE [66] is a hypervolume-based multiobjective evolutionary algorithm. It applies computationally efficient Monte Carlo simulation to approximate the exact hypervolume value, and assigns ranks of solutions induced by the hypervolume indicator. These ranks of solutions can be used in fitness evaluation, mating selection, and environmental selection. By design, it balances the accuracy of the estimates and the computation cost of the Hypervolume calculation.

IBEA [67] avoids the dominance ranking and applies a binary performance indicator directly to the selection process. IBEA is combined with arbitrary indicators that are first defined by the optimization goal and can be adapted to the preferences of the user without any additional diversity preservation mechanism such as fitness sharing.

SMS-EMOA [68] is a hypervolume-based evolutionary multiobjective optimization algorithm. The non-dominated sorting is used as a ranking method and the hypervolume contribution is applied as a selection strategy. The solutions from the worst rank front which contributes the least hypervolume will be removed from the population.

2.7 Selected State-of-the-Art Multiobjective Optimization Algorithms

There are some popular MOEAs often chosen as state-of-the-art competitors for comparison in literature. Some of the most prevalent ones are outlined in the following.

The advanced version of nondominated sorting genetic algorithm (NSGA-II) [52] was improved from its original version. A fast nondominated sorting method is employed to Pareto rank individuals and a crowding distance measurement provides the density estimation for each individual. In fitness assignment, NSGA-II prefers the one with the lower rank, or the one that is located in a less crowded region if both solutions are in the same front. The crowding comparison method preserves the diversity of the population and no sharing parameter is required. The elitism mechanism does not allow an already found nondominated solution to be deleted. Therefore, NSGA-II combines a fast nondominated sorting approach, a parameterless sharing method, and an elitism scheme in order to produce a better spread of solutions in a given MOP. However, the nondominated sorting needs to be performed on a population size of 2*NP*.

SPEA2 [69] assigns a strength value to each individual in both main population and elitist archive which incorporates both dominated and density information. To avoid individuals dominated by the same archive members having identical fitness values, both dominating and dominated relationships are taken into account. The final rank value of a current individual is generated by the summation of the strengths of the individuals that dominate it. The density value of each individual is obtained by the nearest neighbor density estimation. The final fitness value is the sum of rank and density values. In addition, the number of elitists in elitist archive is maintained to be constant.

A multiobjective evolutionary algorithm based on decomposition (MOEA/D) [58] decomposes a multiobjective optimization problem into a number of scalar single optimization subproblems and optimizes them simultaneously. Later, the new version of MOEA/D so called MOEA/D-DE [70] was introduced. MOEA/D-DE employs a DE operator and polynomial mutation. The simulation study of MOEA/D-DE shows that it is less sensitive to *F* and *CR* setting.

Since the associated control parameters of DE play important roles to DE performance. The parameter tuning for a particular problem is a challenging task because it is problem dependent and time-consuming trial and error process. Therefore, among multiobjective DE algorithms proposed, we are interested in a self-adaptive multiobjective DE development. The proposed algorithm utilizes the feedback information from the population to adaptively adjust the control parameters of the proposed multiobjective DE. The feedback information is gathered by the specified performance metrics and then feed to the fuzzy inference rules to adaptively adjust the control parameters. Details on the proposed algorithm are presented in Chapter III.

CHAPTER III

FUZZY MULTIOBJECTIVE DIFFERENTIAL EVOLUTION USING PERFORMANCE METRICS FEEDBACK

Since balancing exploration and exploitation of the population throughout the search process is a key ingredient of any evolutionary algorithms, how to trade-off those two abilities is a challenging task of any EA development. For a multiobjective DE, the mutation strategy and the associated control parameters play important roles to DE performance. To address the issue for those two tasks at hand, a fuzzy multiobjective differential evolution based on performance metrics feedback is proposed in this chapter. The performance of the proposed algorithm is quantified by the well-known ZDT, DTLZ, and WFG test suites.

3.1 Introduction

The performance of the multiobjective optimization algorithms can be quantified by three design goals [15]. First, the distance of the resulting nondominated set to the true Pareto-optimal front should be minimized. Second, a good (in most cases uniform) distribution of the solutions to be found is desirable. Last, the extent of the obtained nondominated front should be maximized, i.e., for each objective, a wide range of values should be covered by the nondominated solutions. This understanding has motivated the idea to exploit the performance metrics, specifically hypervolume, spacing, and maximum spread which address the three optimization goals respectively, as inputs to fuzzy rules. The designed fuzzy inference rules adapt the control parameters of the proposed Fuzzy-based Multiobjective Differential Evolution (FMDE) algorithm in order to dynamically emphasize on the convergence or the diversity in balance of exploration and exploitation throughout the evolution process. The flowchart of the proposed FMDE is shown in Figure 3.1.

The FMDE begins with a randomly generated population and associated control parameters. The population will undergo the mutation and crossover processes. Afterward we combine the offspring and parent population together and identify the nondominated solutions of the combined population. The obtained nondominated front will be measured by the chosen performance metrics, specifically hypervolume, spacing, and maximum spread.

These three measurements become inputs to a fuzzy rule based inference system. Outputs of the fuzzy rule based system determine the control parameters of DE, namely, scaling factor F and greedy factor γ for the mutation strategy that is used in FMDE. The fuzzy rules will be evaluated at every generation in order to adaptively adjust the parameters of mutation strategy for the next generation. The combined population size of $2NP$ will be truncated to the size of NP. Then we update the archive by adding the nondominated solutions found from the combined population. In order to maintain the archive size, FMDE used the crowding comparison method from NSGA-II [52] as the diversity preservation in the archive. The new population undergoes the whole process until the stopping criterion is met. The stopping criterion is the preset maximum number of generations.

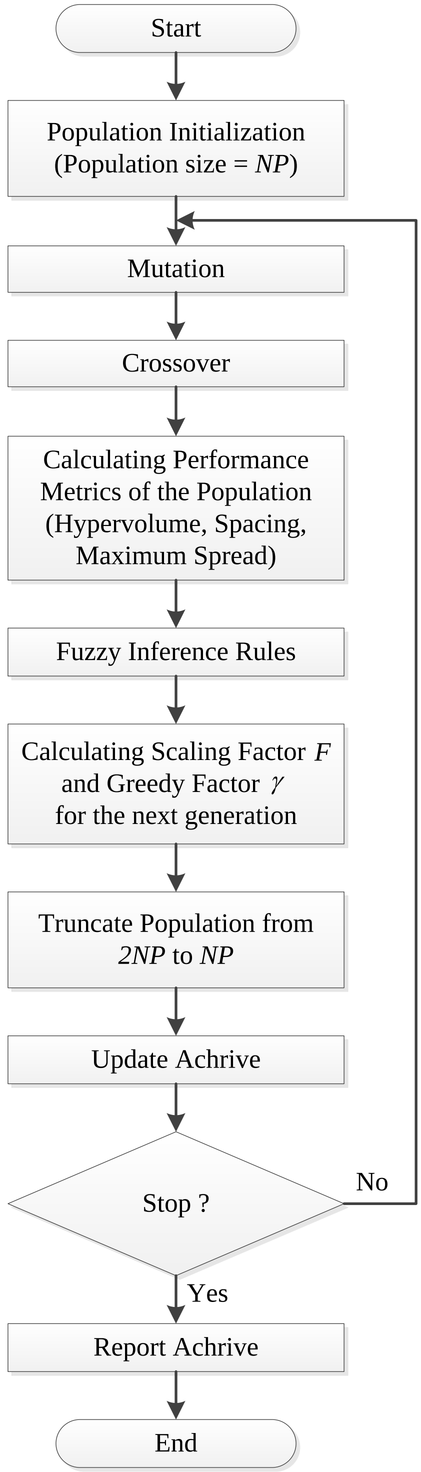


Figure 3.1 Fuzzy Multiobjective Differential Evolution flowchart

3.2 Mutation Strategy

Joshi and Sanderson [71] proposed the mutation operator as

$$\mathbf{v}_{i,G} = \gamma \mathbf{x}_{best,G} + (1-\gamma)\mathbf{x}_{i,G} + F\sum_{k=1}^{K}(\mathbf{x}_{i_a^k} - \mathbf{x}_{i_b^k}), \quad (3.1)$$

where $\gamma \in [0,1]$ is the greediness of the operator, $F \in [0,2]$ is the scaling factor, $\mathbf{x}_{best,G}$ is the best individual found in the parent population at generation *G*, i_a^k, $i_b^k \in [1, NP]$, and *K* is the number of differentials used to generate the perturbation. The control parameter γ represents the degree of exploitation and greediness of the mutation operator. If γ is larger, the mutation strategy is greedier. Consequently, mutant vectors will be generated near the best vectors in the parent population and emphasize the exploitation ability of the algorithm. The scaling factor *F* controls the diversity and exploration ability of the mutation. If *F* is larger, the degree of exploration is higher, and more diversity will be promoted around the mutant vectors. Choosing the appropriate values for γ and *F* is often a trial-and-error, time-consuming, and problem-dependent task. Knowing the state of the current population through performance metrics, we can adjust the values of these parameters without a prior knowledge of the underlying problem, even though Joshi and Sanderson suggested that *F* should be set between 0 and 2. In this study, we follow the suggestion in [72] that *F* should be set even tighter between 0.4 and 1 because from the primary research of *F* indicated that if *F* is set between 0 and 2, but outside of 0.4 and 1, it slows the convergence speed and cannot converge for multiple local fronts problem. Given this consideration, our algorithm sets the range of *F* between 0.4 and 1.

3.3 Crossover Strategy

As stated in Chapter II, there are two types of crossover strategies often employed in DE: binomial and exponential crossovers. Zaharie [73] analyzed the influence of crossover type on the behavior of the DE designs, and found that the exponential crossover was more sensitive to the problem size than the binomial crossover does. Wang *et al.* [72] suggested that *CR* should be a low value near 0 or high value near 1. Given our experiment, we choose the binomial crossover and set *CR* to 0.3. We decide to fix *CR* throughout the evolution process because we would like to investigate the effects of balancing the exploration and exploitation via mutation operation alone.

3.4 Performance Metrics

During the evolution process, we do not know the true Pareto front in *a priori*. We have to resort to choose the performance metrics that can measure the three design goals of optimization without a prior knowledge of the true Pareto front. Even there exist some running metrics such as those in [74], there are very few choices to allow us to measure the convergence, uniform distribution, and extensiveness of the population. Hence, we choose three performance metrics, namely hypervolume, spacing, and maximum spread to quantify the three properties of the obtained nondominated front. In order to evaluate convergence, the metric we choose is hyperarea ratio (hypervolume indicator) [75]. It calculates the size of the hypervolume enclosed by the obtained nondominated front PF_{known} and a reference point. For instance, an individual $\mathbf{x}_i$ in PF_{known} for a two-dimensional MOP defines a rectangle area, $a(\mathbf{x}_i)$, bounded by the chosen reference point and $f(\mathbf{x}_i)$. The union of such rectangle areas is referred to as hyperarea of PF_{known},

$$H(PF_{known}) = \left\{ \bigcup_i a(\mathrm{x}_i) \middle| \forall \mathrm{x}_i \in PF_{known} \right\}. \tag{3.2}$$

It measures both convergence and distribution of a nondominated set, and reference points are set as discussed in [76]. Reference point can also be chosen as the anti-ideal of the worst possible performance in all objectives [77]. If hypervolume value is larger, we can interpret the status of the population as is converging and/or with good distribution. However, it is not clear that the increased value is due to converging, or better distribution, or both. Therefore, we need another metric that can measure the degree of uniform distribution, i.e., spacing. Spacing (*S*) [78] is a metric measuring how the obtained nondominated solutions are evenly distributed:

$$S = \sqrt{\frac{1}{\bar{n}} \sum_{i=1}^{\bar{n}} \left(d_i - \bar{d} \right)^2}, \bar{n} \to \left| PF_{known} \right| \tag{3.3}$$

where d_i is the Euclidean distance in the objective space between individual x_i and the nearest solution of the obtained nondominated front, $\bar{d}$ is the mean Euclidean distance of d_i, and $\bar{n}$ is the number of solutions in the obtained nondominated front. If *S* is zero, it indicates that all solutions of the nondominated front are equally spaced.

Maximum spread (*MS*) [15] measures the length of diagonal hyperbox formed by the extreme solutions observed in the nondominated sets. But it does not reveal the distribution of solutions. A normalized version of *MS* [1] is defined as

$$MS = \sqrt{\frac{1}{M} \sum_{m=1}^{M} \left(\frac{\max_i f_m^i - \min_i f_m^i}{F_m^{\max} - F_m^{\min}} \right)^2}, \tag{3.4}$$

where $F_m^{\max}$ and $F_m^{\min}$ are the maximum and minimum values of the *m*-th objective in the chosen set of Pareto optimal solutions. f_m^i is the value of the *m*-th objective function of the *i*th member of the obtained nondominated solutions and *M* is the number of the objective functions.

3.5 Fuzzy Membership Functions and Fuzzy Inference Rules

The membership functions of hypervolume, spacing, and maximum spread (i.e., $\mu_H, \mu_{Spacing}$, and μ_{MS}) are shown in Figure 3.2. All three performance metrics use the same shape of membership functions. The input is the percentage change of performance metrics calculated every two successive generations and fuzzified to the decreasing and increasing membership values. The chosen fuzzification is "and" method. The output membership function for γ and F are the same shape. There are three status, namely, decrease, no change, and increase for γ and F value. The centroid defuzzification is used, and then we get the percent change of γ and F value. The output membership function is shown in Figure 3.3

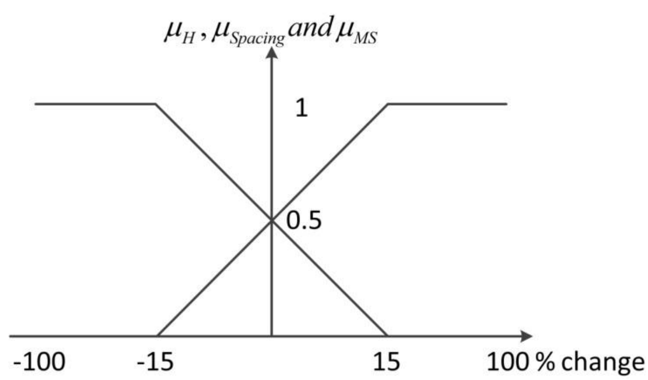


Figure 3.2 Input membership functions

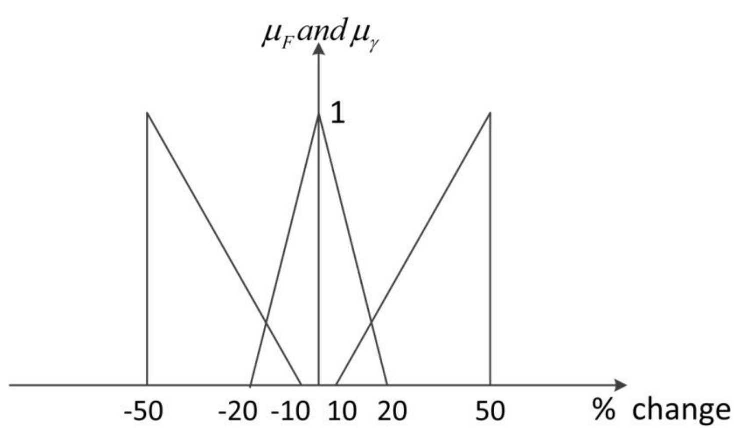


Figure 3.3 Output membership functions

The fuzzy inference rules are shown in Table 3.1. These rules are used for adjusting the values of γ and F in order to emphasize the exploitation (greedy) or exploration (diversity) of the mutation strategy for the next generation. Once we receive the quality feedback of the nondominated front through hypervolume, spacing, and maximum spread, we can design how we emphasize the exploitation or exploration abilities of the proposed algorithm. If we need to place strong emphasis on exploitation, we increase γ and decrease *F*. On the other hand, if we need to encourage strong emphasis on exploration, we decrease γ and increase *F*. However, if we need to place a mild emphasis on the exploitation, we can accomplish it by two approaches: increase γ and keep *F* unchanged, or keep γ unchanged and decrease *F*. To place a mild emphasis on the exploration, we also can have two options as the same manner as exploitation: keep γ unchanged and increase *F* or decrease γ and keep *F* unchanged.

Rule number 3 describes the best status quo of the population because hypervolume is increasing while spacing is decreasing, that implies the population is converging and more uniformly distributed. The extensiveness of the obtained front is larger by increasing *MS* value. Thus, we will not change γ and *F* because they should remain appropriate values for the current status. Compared to Rule 3, in Rule number 4 we need mild exploration in order to increase the extensiveness of the population. As a result, we do nothing with γ, but increase *F* slightly.

Rule number 6 on the other hand reveals the worst case scenario, because all three metrics indicate that the obtained front is diverging and losing diversity and extensiveness in the obtained nondominated front. We need strong exploration so to decrease γ and increase *F*.

In the case of increasing hypervolume, it means the population is converging but we do not know whether the nondominated solutions are uniformly distributed or not. Thus, we turn our attention to spacing metric. If it is increasing, so the solutions are not well distributed and we need a mild exploitation. Rules 1 and 2 are under this scenario, but the maximum spread for Rule 1 is increasing thus *F* is kept unchanged but γ is increased accordingly. Maximum spread for rule number 2 is decreasing, the stage of population is converging but not well distributed and the searched area is shrinking. Thus we need to increase a mild degree of exploration. Since the spacing is worsening, we decrease γ, and keep F unchanged.

If the hypervolume is decreasing, it implies that the search direction of the population is incorrect. As a result, we will increase the exploration ability. Rules 5 to 8 are under this case. Rules 5 and 7 actions are to increase a mild degree of exploration. Spacing of rule 5 is increasing; it implies that the solutions are crowded then we decrease γ accordingly, while keep γ unchanged for rule 7. Rule 8 states that the population is not converging and the search area of the population is shrinking even though the distribution is good. To respond to this scenario, we add a mild degree of exploration by keeping γ unchanged and increasing F slightly.

3.6 Experiments and Results

3.6.1 The Benchmark functions

The proposed Multiobjective Differential Evolution (MODE), namely FMDE, is tested on the ZDT and DTLZ test suites [15, 79]. Specifically, ZDT benchmark functions (i.e., ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6) are bi-objective optimization problems. On the other hand, DTLZ1 to DTLZ7 benchmark functions are all three-objective. ZDT1 is with a convex Pareto-optimal front, while ZDT2 is the nonconvex counterpart to ZDT1. ZDT3 represents the discontinuous Pareto-optimal front. It consists of several discontinuous convex parts. ZDT4 contains many local Pareto-optimal fronts. The search space of ZDT6 is non-uniformity. Thus, it introduces two challenges: first, the Pareto-optimal front is non-uniformly distributed, and the density of the solutions is lowest near the Pareto front and highest away from the front. ZDT5 is usually not included in the experiment because the decision variable is a binary string. Additionally, DTLZ1 is a linear hyper-plane and has many local Pareto fronts where an MOEA can be attracted to them before reaching the global Pareto front. DTLZ2 is the spherical Pareto optimal front. DTLZ3 is with the concave and multimodality front. The local optimal fronts are

Table 3.1 Fuzzy inference rules

Rules	**Inputs**			**Outputs**		**Actions**
	Hypervolume	***Spacing***	***MS***	γ	***F***	
1	Increase	Increase	Increase	Increase	No Change	Mild exploitation
2	Increase	Increase	Decrease	Decrease	No Change	Mild exploration
3	Increase	Decrease	Increase	No Change	No Change	Do nothing
4	Increase	Decrease	Decrease	No Change	Increase	Mild exploration
5	Decrease	Increase	Increase	Decrease	No Change	Mild exploration
6	Decrease	Increase	Decrease	Decrease	Increase	Strong exploration
7	Decrease	Decrease	Increase	No Change	Increase	Mild exploration
8	Decrease	Decrease	Decrease	No Change	Increase	Mild exploration

parallel to the global Pareto optimal front. DTLZ4 is the modification of DTLZ2 to allow the solutions to be crowded near the plane. DTLZ5 is to test the ability of an MOEA to converge to a curve. DTLZ6 is modified from DTLZ5 to make the problem even harder. DTLZ7 has four disconnected Pareto optimal regions in the search space.

In addition, FMDE is also tested on WFG test suite [80]. WFG1 to WFG9 benchmark functions are scalable and include some properties that are different from DTLZ test suite. These properties involve non-separable problems, a truly degenerate problem, deceptive problems, a mixed shape Pareto front problem, and bias problems. We tested WFG1 to WFG9 in three-objective functions.

3.6.2 Experimental setup

The proposed FMDE is compared with three state-of-the-art MOEAs, i.e., NSGA-II [52], SPEA2 [69], MOEA/D-DE [58], and three indicator-based MOEAs, i.e., HypE [66], IBEA [67], SMS-EMOA [68]. Each algorithm is tested on the two-objective (five ZDT) test functions and three-objective (seven DTLZ) test functions with 30 independent runs. We also investigated the performance of FMDE, MOEA/D-DE, and the self-adaptive JADE2 on a wide range of different problems (i.e., three-objective WFG1 to WFG9 problems). For each trial, an algorithm will stop if it reaches the maximum number of function evaluation at 250,000 for bi-objective and 300,000 for three-objective problems. The population size is 100 for bi-objective and 300 for three-objective problems. The FMDE uses the external archive and is updated as in [52]. The archive size for bi-objective and three objective problems are 100 and 300, respectively. γ and F are both set at 0.5 for the first generation. K is 1 which implies we use only one differential vector. Even though CR offers the diversity control mechanism for DE as well, however, if we adaptively adjust γ, F, and CR, it can be overly used for the diversity effect. As a result, we set CR = 0.3 and F should be limit within [0.4, 1] for the whole experiments [72].

The other parameter settings for the competing algorithms are the same as what suggested in the respective original papers. The reference points we set for calculating hypervolume indicator for all test function are specified in [76]. We choose the reference point (3,100) for ZDT1 and ZDT4. The other reference points for ZDT2, ZDT3, and ZDT6 are (3/2, 4/3), (100, 5.446), and (1.497, 4/3), respectively. Additionally, the reference point for DTLZ1 is (1, 1, 1) and (1.180, 1.180, 1.180) for DTLZ2, DTLZ3, and DTLZ4. DTLZ7 uses (13.3725, 5.3054, 5.3054) as the reference point. Since [76] does not state the reference points for DTLZ5 and DTLZ6, we choose the reference point as (5, 5, 5) for both benchmark functions. The reference points for WFG1 to WFG9 are all set at (100, 100, 100).

The code for NSGA-II is available at http://www.iitk.ac.in/kangal. Similarly, MOEA/D-DE can be found at http://dces.essex.ac.uk/staff/zhang/webofmoead.htm, while SPEA2, HypE, and IBEA are available at http://www.tik.ee.ethz.ch/pisa. On the other hand, the code of SMS-EMOA is available at http://ls11-www.cs.uni-dortmund.de/rudolph/hypervolume/start.

3.6.3 Metrics

In comparing the performance among MOEAs, the performance metric used in this experiment is the inverted generational distance (*IGD*) [81]. Let PF_{true} be the uniformly distributed true Pareto front. Let P_A be the obtained approximated one. *IGD* is defined as

$$IGD(P_A, PF_{true}) = \frac{\sum_{v \in PF_{true}} d(v, P_A)}{|PF_{true}|} , \tag{3.5}$$

where $d(v, P_A)$ is the minimal Euclidean distance between every $v \in PF_{true}$ and the set P_A. *IGD* measures both the convergence and diversity of the obtained approximation front. If *IGD* = 0, it means that all the approximation solutions are in the true Pareto solutions and they cover all the extension of the true Pareto front. The uniformly distributed true Pareto fronts for calculating

IGD for all test problems are taken from [82]. $|PF_{true}|$ for ZDT1, ZDT2, ZDT3, and ZDT6 is 1,001 whereas ZDT4 is 269. Also $|PF_{true}|$ for DTLZ1 and DTLZ2 is 10,000, DTLZ3 and DTLZ4 is 4,000, DTLZ5 is 166,500, DTLZ6 is 28,000 and DTLZ7 is 676. On the other hand, $|PF_{true}|$ for WFG1 is 2,000, WFG2 is 2900. $|PF_{true}|$ for WFG3, WFG5, WFG6, and WFG7 is 10,000. $|PF_{true}|$ for WFG4 is 9,997, for WFG8 is 10,100, and for WFG9 is 10,201.

3.6.4 Experimental results

An example of the obtained nondominated front and the associated performance metrics, and control parameters are shown in Figure 3.4 for ZDT1. At the beginning of the evolutionary process, γ and *F* start at 0.5. At approximately the 20th generation, the spacing reaches the highest while *MS* is the lowest. This implies the obtained approximate front is crowded and the distance between extreme solutions is short, even though the hypervolume is continuously increasing. The performance assessment implies that the population can find more nondominated solutions but they are crowded together in some part of the approximation front. Hypervolume is continuously improved and goes to steady state around the 100th generation. After that the hypervolume has very small fluctuation which means FMDE continuously improves its convergence and distribution. When the algorithm converges, we can see that the spacing went to near zero which is the ideal value for evenly distribution. The maximum spread of the algorithm is growing to approximately one which indicates that the algorithm reaches its maximum extent of the extreme solutions. Later on γ is decreasing while *F* is increasing. This infers that the search process detects the promising region and then fast converges toward the direction that makes γ value even higher in order to facilitate the exploitation ability. When the population converges, the exploration becomes prominent because every individual will be near or at the true Pareto front and we need to emphasize the local search then. As can be seen, the FMDE continuously improves its performance: hypervolume is increasing which demonstrates that the algorithm is converging but we observe that the distribution is not good due to fluctuating spacing values. Meanwhile the fluctuation of the maximum spread indicates that the extreme nondominated solutions occupy a smaller space. After the 120th generations, γ decreased to the lowest value near zero, but F is closed to one. This implies that the degree of exploration is higher than exploitation. What it means is that the algorithm converges, the number of the nondominated solutions found is fairly high, and the algorithm tries to do the local search to make the obtained front evenly distributed. Sample plots of the obtained approximate fronts for ZDT2, ZDT3, ZDT4, ZDT6, and DTLZ test suite (DTLZ1-7) are shown in Figure 3.5 for reference. It can be seen that FMDE performs very competitively on all test instances with various problem characteristics.

The mean value and standard deviation of IGD on ZDT and DTLZ test instances for FMDE, MOEA/D-DE, NSGA-II, SPEA2, HypE, IBEA, and SMS-EMOA are listed in Table 3.2. The WFG test results are shown in Table 3.3. We compare the performance between any two algorithms in terms of statistics by utilizing the nonparametric Mann-Whitney-Wilcoxon rank sum test on *IGD*. The performance of FMDE with respect to *IGD* on each test instance could be noted as better ("+"), same ("="), or worse("-") than/as that of one chosen competitor if the p-value generated by FMDE and its competitor is larger than, equal to, or smaller than a standard tabulated value of U-test at a significance level of 0.05 by a two-tailed test. The best value of *IGD* among those algorithms is highlighted by boldface in each test instance.

A row "Score" in Table 3.2 and Table 3.3 shows the difference between the number of "+" and the number of "-", which gives an overall comparison between FMDE and one competing algorithm over all test problems considered. For example, comparing FMDE and HypE on ZDT and DTLZ test suites, FMDE outperforms HypE on nine problems (i.e., ZDT1 to ZDT6, DTLZ2 to DTLZ4, and DTLZ7), does almost the same on one problem (i.e., DTLZ6) and underperforms on two problems (DTLZ1 and DTLZ5). As a result, the score listed in the last row and the column of HypE is 7 (i.e., 9 – 2 = 7).

Table 3.2 shows that the proposed FMDE algorithm outperforms MOEA/D-DE on eight out of twelve benchmark functions. In case of bi-objective, FMDE is competitive with MOEA/D-DE. The performance of FMDE is statistically better than MOEA/D-DE on ZDT1, ZDT2, and ZDT3, but worse on ZDT4 and ZDT6. However, FMDE outperforms MOEA/D-DE on five out of seven three-objective test functions. FMDE outperforms NSGA-II on ten out of twelve functions. It outperforms NSGA-II on all ZDT test functions, but underperforms on DTLZ1 and DTLZ5. FMDE is competitive with SPEA2 on most bi-objective and three-objective problems. The comparison between FMDE and the indicator based algorithms shows that FMDE outperforms HypE on the ZDT suite and underperforms on the DTLZ1 and DTLZ5. FMDE underperforms IBEA on ZDT1, DTLZ1, DTLZ5, and DTLZ6 and does the same on DTLZ3 and DTLZ4. FMDE outperforms SMS-EMOA on ZDT6 and five DTLZ test instances except for DTLZ1 and DTLZ4, and underperforms on ZDT1 to ZDT3, and does the same on ZDT4. Table 3.3 shows the results for WFG test suite. FMDE outperforms MOEA/D-DE on all nine test problems. FMDE outperforms JADE2 on seven problems except WFG8 and does the same performance on WFG6. The self-adaptive mechanism of FMDE can help improving the performance of DE algorithm on the bias problems such as WFG test instances.

Overall, FMDE is competitive with respect to the chosen competing MOEAs on all three test suites popularly used in literature. It is interesting to observe that the most difficult problems for FMDE are DTLZ1 and WFG8. The problem characteristics presented in DTLZ1 is the linear Pareto front, while WFG8 is the bias Pareto front. In summary, FMDE perform very well on the convex, nonconvex, multimodality and discontinuous problems. But it faces difficulties on the linear and bias problems. The preservation of diversity in FMDE may not be sufficient for solving a higher degree of multimodality. It should be improved by adaptively adjust *CR* in our future research.

3.7 Remarks

This chapter presents a MODE which utilizes performance metrics, namely hypervolume, spacing, and maximum spread to estimate the state of evolution in order to dynamically adapt the greedy and distribution parameters of a DE-based mutation strategy over the course of evolution. The direction of change for each parameter is determined through fuzzy inference rules. The effect of dynamically adjust these parameters is that we can emphasize the exploitation or exploration ability due to the status of the search process. The experimental results show that the proposed FMDE performs better than those chosen state-of-the-art competing MOEAs. This research demonstrates that we can integrate performance metrics observed and expert knowledge of optimization process together through fuzzy inference rules. Therefore, it is one credible approach to automatically adjust the control parameters without a prior knowledge on the landscape of the Pareto front. The advanced version of FMDE will be introduced in Chapter IV. It will adaptively adjust *CR* in concurrent with γ and F.

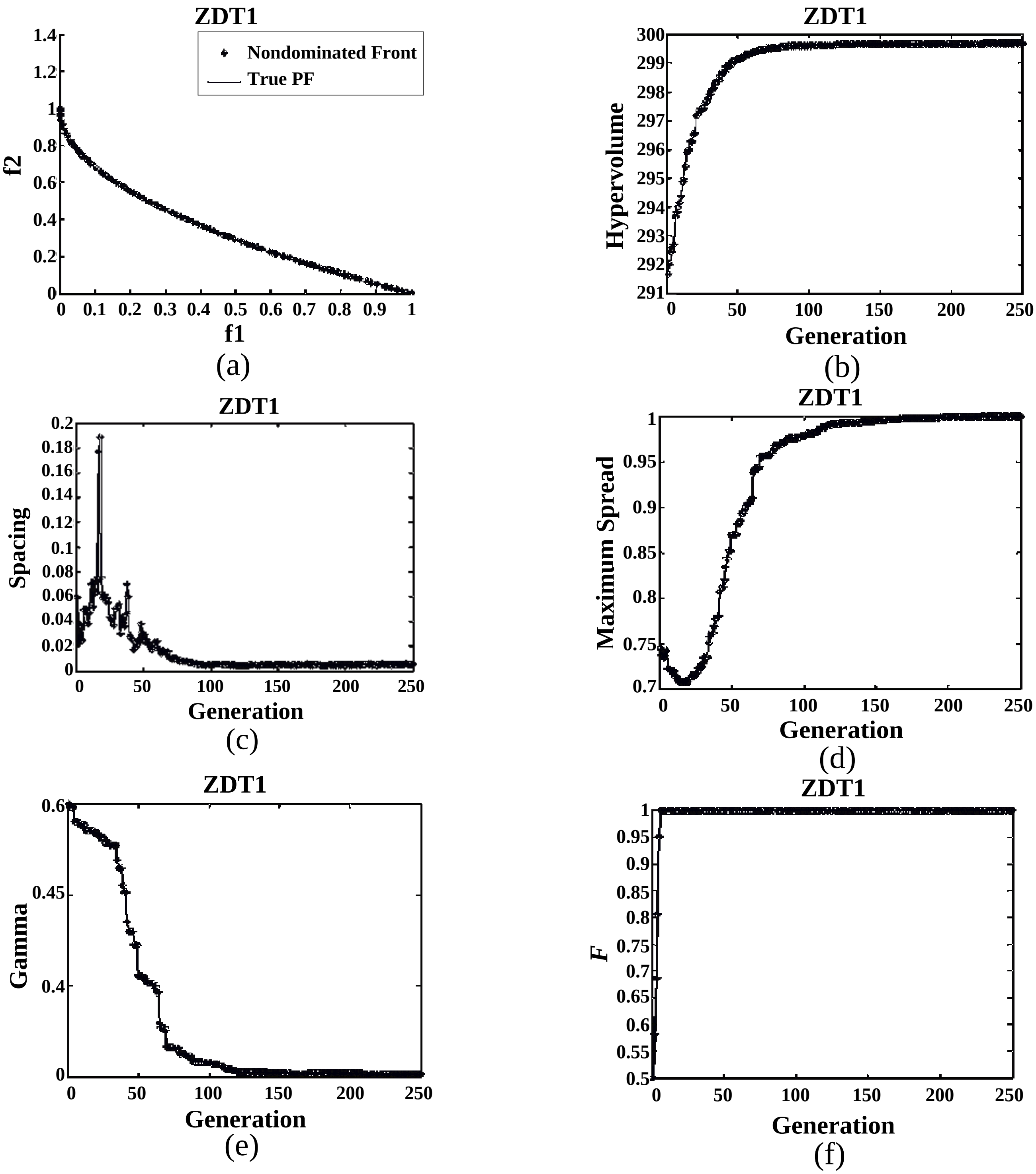


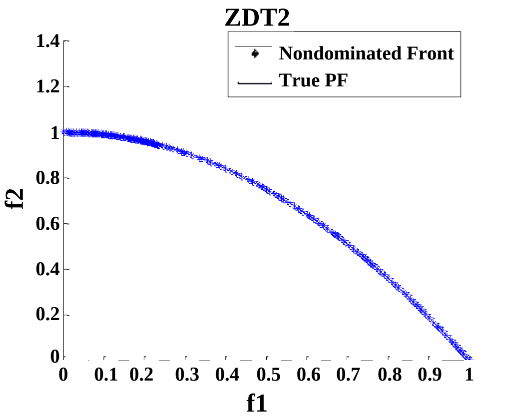
Figure 3.4 A sample run of FMDE on ZDT1. (a) The approximated and true Pareto fronts, (b) hypervolume performance metric (c) spacing performance metric , (d) maximum spread performance over course of evolution (generations), (e) Gamma and (f) *F* control parameters of DE operator

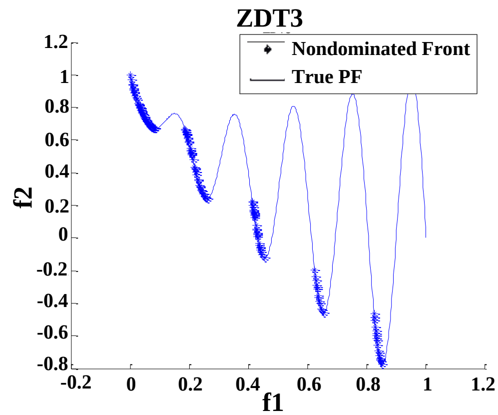
Table 3.2 Comparison of IGD for FMDE and Other MOEAS on Rank Sum Tests

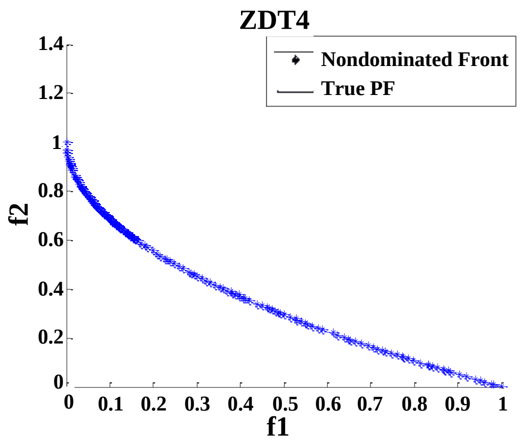
Test Functions		FMDE	MOEA/D-DE	NSGA-II	SPEA2	HypE	IBEA	SMS-EMOA
ZDT1	Mean	4.800E-3	1.460E-1	9.400E-3	8.500E-3	3.410E-2	4.000E-3	**3.600E-3**
	Std.	2.560E-4	5.293E-18	8.775E-4	8.8214E-4	1.050E-2	3.534E-5	1.4955E-5
	p-value		3.012E-11	3.012E-11	3.012E-11	3.012E-11	3.012E-11	3.0199e-11
	u-test		+	+	+	+	-	-
ZDT2	Mean	4.700E-3	6.400E-3	3.926E-1	2.923E-1	3.974E-1	4.692E-1	**4.400E-3**
	Std.	1.9826E-4	8.8219E-19	2.954E-1	3.056E-1	2.840E-1	2.590E-1	9.723E-5
	p-value		3.012E-11	3.012E-11	3.012E-11	3.012E-11	3.018E-11	1.070e-9
	u-test		+	+	+	+	+	-
ZDT3	Mean	3.500E-3	1.230E-2	4.078E-1	4.087E-1	4.376E-1	3.997E-1	**2.600E-3**
	Std.	2.3176E-4	7.058E-18	1.020E-2	0.0067	2.360E-2	6.800E-3	2.2173e-5
	p-value		1.212E-12	3.012E-11	3.012E-11	3.012E-11	3.012E-11	3.012e-11
	u-test		+	+	+	+	+	-
ZDT4	Mean	**6.949E-1**	7.000E-3	1.691E+0	1.189E+0	1.9488E+0	1.930E+0	1.930E+0
	Std.	6.792E-1	2.500E-3	8.934E-1	6.130E-1	8.659E-1	2.368E+0	2.368E+0
	p-value		3.160E-1	3.571E-6	1.200E-3	3.012E-11	9.500E-3	5.895E-1
	u-test		-	+	+	+	+	=
ZDT6	Mean	3.700E-3	**2.100E-3**	8.423E-1	9.057E-1	5.391E-1	5.821E-1	3.106E+0
	Std.	6.545E-4	0.000E+0	1.183E-1	1.125E-1	5.970E-2	8.760E-2	5.996E+0
	p-value		3.012E-11	3.012E-11	3.012E-11	4.444E-7	3.012E-11	3.012e-11
	u-test		-	+	+	+	+	+
DTLZ1	Mean	2.384E+2	**5.104E-1**	4.203E+0	2.279E+0	2.063E+0	1.430E+0	5.384E-1
	Std.	2.678E+0	3.876E-16	2.973E+0	1.447E+0	8.387E-1	3.360E+0	1.200E-3
	p-value		1.212e-12	3.012E-11	3.012E-11	3.012E-11	3.012E-11	1.440E-2
	u-test		-	-	-	-	-	-
DTLZ2	Mean	**6.900E-2**	6.248E-1	8.670E-2	5.460E-2	3.879E-1	8.910E-2	7.410E-2
	Std.	2.600E-3	2.258E-16	6.800E-3	1.300E-3	1.850E-2	2.700E-3	9.936E-4
	p-value		1.212E-12	3.6897E-11	3.012E-11	3.012E-11	3.012E-11	3.497E-9
	u-test		+	+	-	+	+	+
DTLZ3	Mean	**7.205E+0**	5.596E-1	3.455E+1	2.496E+1	1.946E+1	5.739E+0	1.034E+1
	Std.	6.492E+0	4.517E-16	1.044E+1	7.6415E+0	1.160E+1	5.372E+0	7.011E+0
	p-value		1.334E-8	6.696E-11	1.695E-9	4.745E-6	5.592E-1	3.150E-2
	u-test		-	+	+	+	=	+
DTLZ4	Mean	7.240E-2	2.947E-1	1.275E-1	1.0701E-1	1.914E-1	8.970E-2	**4.590E-2**
	Std.	5.800E-3	2.258E-16	1.012E-1	8.100E-2	8.310E-2	8.120E-2	1.600E-3
	p-value		1.212E-12	3.500E-3	0.186E-1	9.919E-11	3.988E-4	3.012E-11
	u-test		+	+	=	+	=	-
DTLZ5	Mean	6.115E-1	7.934E-1	7.100E-3	**5.500E-3**	1.035E-1	1.260E-2	6.234E-1
	Std.	1.220E-2	3.3876E-16	5.718E-4	2.359E-4	2.420E-2	1.100E-3	3.500E-2
	p-value		1.212E-12	3.012E-11	3.012E-11	3.012E-11	3.012E-11	1.170E-2
	u-test		+	-	-	-	-	+
DTLZ6	Mean	6.091E-1	3.4069E+0	3.406E+0	1.978E+0	7.844E-1	**1.141E-1**	6.927E-1
	Std.	9.100E-3	1.355E-15	3.650E-1	1.731E-1	3.885E-1	9.800E-3	2.565E-1
	p-value		1.212E-012	3.012E-11	3.012E-11	9.352E-1	3.019E-11	2.366E-12
	u-test		+	+	+	=	-	+
DTLZ7	Mean	**4.980E-2**	8.747E-1	1.500E-1	1.262E-1	1.960E-1	1.162E-1	5.340E-2
	Std.	1.900E-3	4.517E-16	9.210E-2	1.423E-1	7.130E-2	1.265E-1	4.620E-2
	p-value		1.212E-12	3.012E-11	3.012E-11	3.012E-11	1.206E-10	5.573E-10
	u-test		+	+	+	+	+	+
	Better (+)		8	10	8	9	6	6
	Same (=)		0	0	1	1	2	1
	Worse (-)		4	2	3	2	4	5
	Score		4	8	5	7	2	1

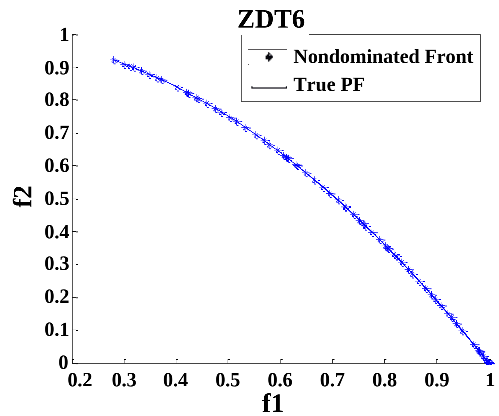
Table 3.3 Comparison of IGD for FMDE and other MODEs on Rank Sum Tests

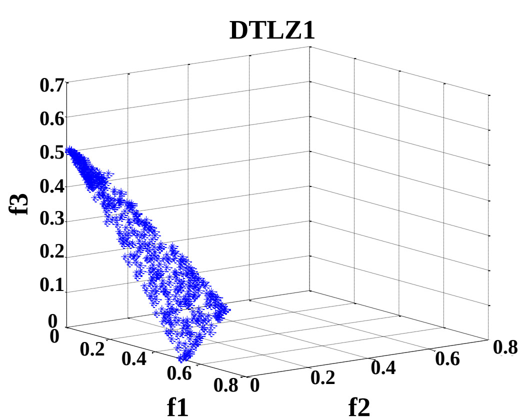
Functions		FMDE	MOEA/D-DE	JADE2
WFG1	Mean Std. p-value u-test	**5.330E-1** **2.700E-3**	6.571E-1 2.258E-16 1.212E-12 +	5.594E-1 2.700E-3 3.012E-11 +
WFG2	Mean Std. p-value u-test	**4.292E-1** **4.859e-4**	5.822E-1 0.000E+0 1.212E-12 +	4.316E-1 1.000E-3 6.066E-11 +
WFG3	Mean Std. p-value u-test	**5.254E-1** **6.1244E-4**	8.315E-1 3.388E-16 1.212E-12 +	5.307E-1 1.300E-3 3.012E-11 +
WFG4	Mean Std. p-value u-test	**4.011E-1** **4.300E-3**	1.1176E+0 9.034E-16 1.212E-12 +	4.231E-1 6.900E-3 3.012E-11 +
WFG5	Mean Std. p-value u-test	**1.854E-1** **1.200E-3**	5.695E-1 1.1292E-16 1.2118E-12 +	1.955E-1 4.900E-3 3.338E-11 +
WFG6	Mean Std. p-value u-test	**3.854E-1** **1.000E-3**	5.271E-1 1.129E-16 1.212E-12 +	3.863E-1 6.300E-3 7.506E-1 =
WFG7	Mean Std. p-value u-test	**8.360E-2** **1.060E-2**	6.404E-1 0.000E+0 1.2118E-12 +	3.0007E-1 2.430E-2 3.012E-11 +
WFG8	Mean Std. p-value u-test	4.570E-1 2.000E-3	5.345E-1 3.3876E-16 1.2118E-12 +	**4.436E-1** **3.100E-3** **3.012E-11** **-**
WFG9	Mean Std. p-value u-test	**1.064E-1** **4.300E-3**	4.617E-1 2.8230E-16 1.2118E-12 +	1.560E-1 1.750E-2 3.012E-11 +
	Better (+) Same (=) Worse (-) Score		9 0 0 9	7 1 1 6

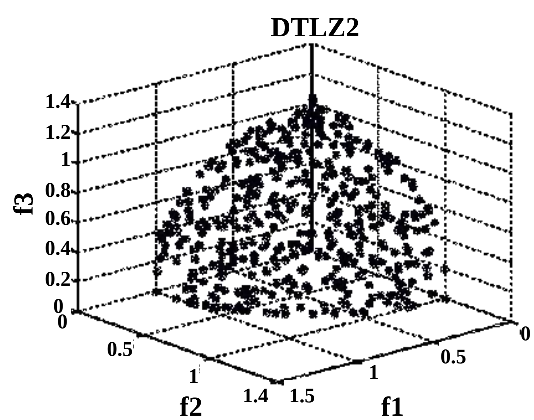

ZDT2
Nondominated Front
True PF
f2
f1
(a)

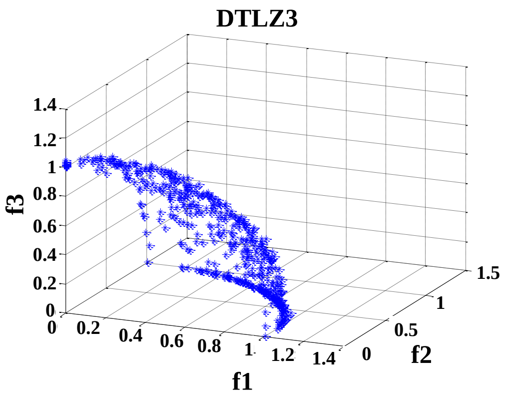

ZDT3
Nondominated Front
True PF
f2
f1
(b)


ZDT4
Nondominated Front
True PF
f2
f1
(c)

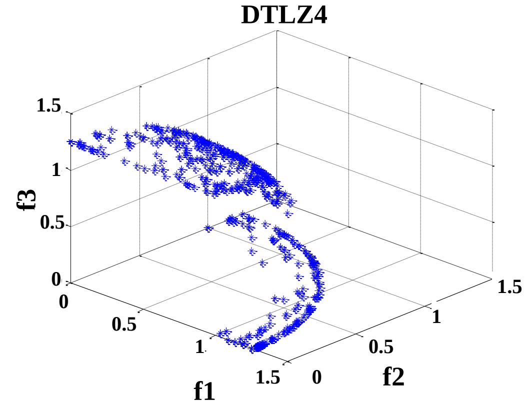

ZDT6
Nondominated Front
True PF
f2
f1
(d)


DTLZ1
f3
f1
f2
(e)

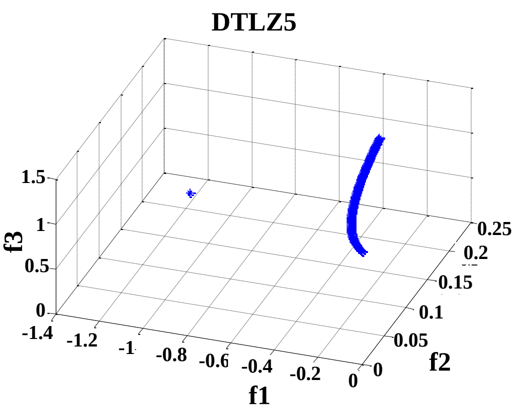

DTLZ2
f3
f2
f1
(f)



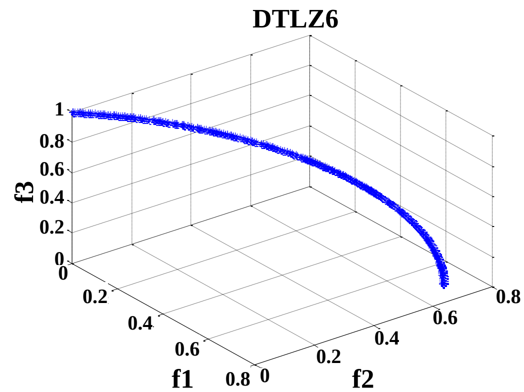
(g)



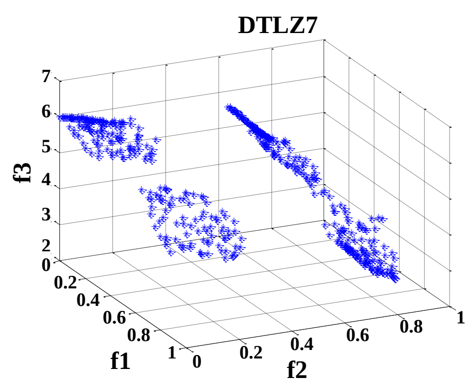
(h)



(i)



(j)



(k)

Figure 3.5 Sample approximated fronts by FMDE for benchmark function (a) ZDT2, (b) ZDT3, (c) ZDT4, (d) ZDT6, (e) DTLZ1, (f) DTLZ2, (g) DTLZ3, (h) DTLZ4, (i) DTLZ5, (j) DTLZ6, and (k) DTLZ7.

CHAPTER IV

FUZZY MULTIOBJECTIVE DIFFRENTIAL EVOLUTUION WITH ADAPTIVE CROSSOVER RATE USING PERFORMANCE METRICS FEEDBACK

A classic DE involves three operators, i.e., mutation, crossover, and selection. FMDE focuses on adaptation of the associated control parameters of the mutation strategy alone. There are a few studies on the influence of the crossover strategy and the crossover rate to DE performance [72-73]. The conclusions from those crossover rate studies state that there is not a single crossover rate that is suitable for all types of optimization problems. The tuning of crossover rate usually needs *a priori* knowledge of the problem at hand. Therefore, the advanced version of FMDE (AFMDE) is proposed. AFMDE adaptively adjusts the crossover rate in concurrence with mutation parameters. The performance of AFMDE is quantified by the well-known ZDT, DTLZ, and WFG test suites. The influence of the initial crossover rate and fuzzy membership function parameters to AFMDE performance is investigated as well.

4.1 Introduction

The mutation operator plays an important role in DE and it promotes diversity in the population. However, only the mutation strategy alone cannot provide sufficient diversity for the search process. Therefore, DE adopts the crossover operation in order to further enhance the diversity of the population. This chapter introduces the advanced version of FMDE (in short called AFMDE) by adaptively adjust CR in concurrence with mutation parameters: γ and F.

There are literatures that proposed adaptively adjusting *CR* mechanism in DE optimization but most of them were applied to single objective DE variants. The various methods to adapt *CR* can be classified in the following:

4.1.1 Encoding CR into each individual

Abbass [48] proposed the self-adaptive Pareto DE algorithm for MOPs by encoding *CR* into each individual. Then *CR* will be simultaneously evolved with other parameters. Brest *et al.* [83] proposed a self-adaptive DE (jDE) in which *F* and *CR* are encoded into each vector. The better values of these encoded parameters that produce the better offspring are more likely to survive. Zamuda *et al.* [62] proposed differential evolution for multiobjective optimization with self-adaptation (DEMOwSA). *F* and *CR* are encoded into the decision variables and simultaneously evolved with the population.

4.1.2 Success rate

Huang *et al.* [59] proposed MOSaDE. The algorithm automatically adapts the trial vector generation strategies and their associated parameters according to their previous experience of generating promising solutions as same as SaDE [60]. However, MOSaDE uses non-domination sorting and crowding in evaluation process. Later, Huang *et al*. [61] modified MOSaDE in order to learn the suitable crossover rate and mutation strategies for each objective separately in MOPs. Zhang and Sanderson [63] proposed the self-adaptive multiobjective DE with direction information provided by archived inferior solutions (JADE2). The self-adaption of *F* and *CR* based on the principle that the better values of control parameters tend to generate better offspring are more likely to survive and should be propagated through the evolution process.

4.1.3 CR pool

Mallipeddi and Suganthan [84] proposed a DE with an ensemble of mutation and crossover strategies and its associated parameters (EPSDE). A pool of mutation and crossover strategies along with a pool of each control parameter (*F* and *CR*) values are initialized. Each vector is randomly assigned mutation strategy and associated parameter values taken from the respective pools. The mutation operator and associated parameters that produce the better offspring (trial vector) are retained with that trial vectors, and will be used to generate the new vector in the next generation. The combination of these successful mutation and crossover strategies and associated parameters are stored in a "successful pool". If the combination of the mutation operator and associated parameters fail to generate the better offspring (trial vector), the parents (target vector) will randomly assigned the mutation and crossover operators and parameters from the initialized mutation and crossover operators and parameter pools, or from the successful pool.

4.1.4 Randomly generated CR

Jingqiao and Sanderson [85] proposed a single objective DE with adaptive parameter control called JADE. They incorporate a new mutation strategy along with the adaptive control parameter *F* and *CR*. *F* and *CR* will be generated at each generation according to a Cauchy distribution. The location parameter of the Cauchy distribution will be updated at the end of each generation based on the set of successful *F* and *CR* values.

4.2 Fuzzy Membership Functions and Fuzzy Inference Rules

Since γ and *F* parameters in FMDE have an impact on balancing exploration and exploitation abilities of the population, if *CR* keeps changing in spite of population's status, it may compromise the exploration and exploitation balance. Besides, adjusting *CR* for different mutation strategies has different effects to the exploration and exploitation abilities. For instance, when DE/rand/1 is adopted, increasing *CR* will increase diversity and emphasize exploration ability of the population. However, if DE/best/1 is used, increasing *CR* will emphasize the exploitation ability of the population because the trial vectors will be generated near the best vector.

Therefore, the advanced version of FMDE (AFMDE) which incorporates adaptive *CR* is proposed. Consider FMDE mutation strategy (3.1) which is shown below again for convenience,

$$\mathbf{v}_{i,G} = \gamma \mathbf{x}_{best,G} + (1-\gamma)\mathbf{x}_{i,G} + F\sum_{k=1}^{K}(\mathbf{x}_{i_a^k} - \mathbf{x}_{i_b^k}). \tag{3.1}$$

If *CR* is higher, the trial vector $\mathbf{u}_{i,G}$ elements are more likely to inherit the mutant vector $\mathbf{v}_{i,G}$ elements. That means the diversity or exploration ability of the population is stronger than exploitation. On the other hand, if *CR* is lower, the trial vector $\mathbf{u}_{i,G}$ elements are more likely to inherit the target vector $\mathbf{x}_{i,G}$ elements. That means the diversity or exploitation ability of the population is stronger than exploration. Therefore, *CR* will be only adapted by two rules: rule 1 and rule 6 as shown in Table 4.1. Rule 1 needs an emphasis on mild exploitation, thus, *CR* will be decreased. Rule 6 is the worst case scenario thus it needs strong exploration, hence, *CR* will be increased.

Table 4.1 Fuzzy inference rules

Rules	**Inputs**			**Outputs**			**Actions**
	Hypervolume	**Spacing**	**MS**	γ	F	CR	
1	Increase	Increase	Increase	Increase	No Change	Decrease	Mild exploitation
2	Increase	Increase	Decrease	Decrease	No Change	No Change	Mild exploration
3	Increase	Decrease	Increase	No Change	No Change	No Change	Do nothing
4	Increase	Decrease	Decrease	No Change	Increase	No Change	Mild exploration
5	Decrease	Increase	Increase	Decrease	No Change	No Change	Mild exploration
6	Decrease	Increase	Decrease	Decrease	Increase	Increase	Strong exploration
7	Decrease	Decrease	Increase	No Change	Increase	No Change	Mild exploration
8	Decrease	Decrease	Decrease	No Change	Increase	No Change	Mild exploration

The input and output fuzzy membership functions for AFMDE are shown in Figure 4.1 and Figure 4.2, respectively.

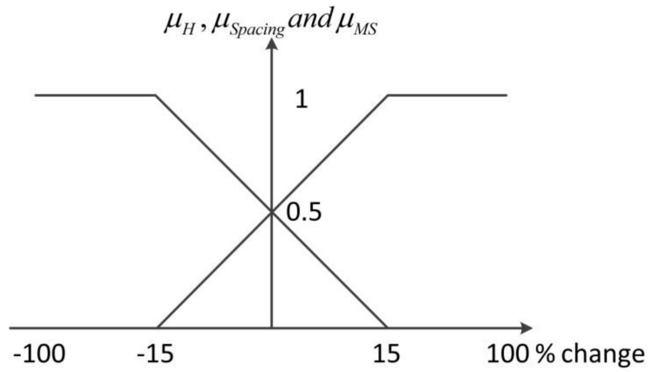


Figure 4.1 Input membership functions

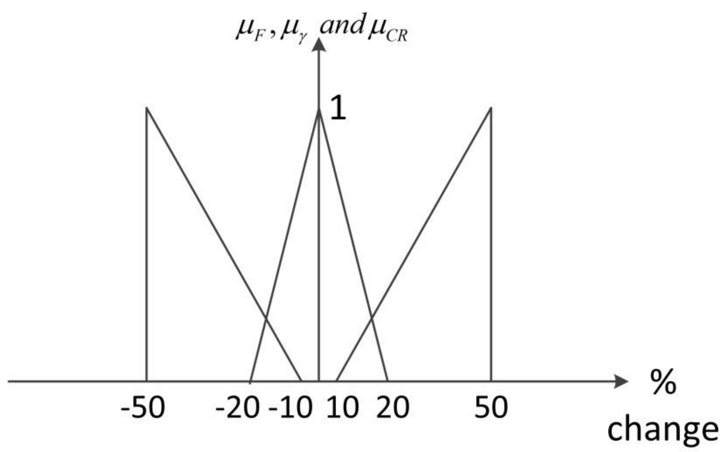


Figure 4.2 Output membership functions

4.3 Experiments and Results

AFMDE will be tested on the ZDT, DTLZ, and WFG test instances. The parameter settings are the same as FMDE experiments in Section 3.6. The sensitivity of initial value of *CR* and output membership function parameters is investigated as well. *IGD* will be used for measuring the performance of a MOEA. Mann-Whitney-Wilcoxon rank sum test on *IGD* is used to compare any two algorithms in terms of statistics with a two-tail test at the significance level of 0.05.

4.3.1 AFMDE with various initial CR values

FMDE and AFMDE with various initial *CR* values (i.e., 0.1, 0.3, 0.5, and 0.9) are tested on the ZDT, DTLZ, and WFG test suites. The results from Table 4.2 shows that AFMDE with initial *CR* = 0.1 (AFMDE0.1) is comparable with FMDE. AFMDE with other initial *CR* values underperform FMDE. In case of bi-objective problems (ZDT test suites), AFMDE with initial *CR* 0.1 (AFMDE0.1) is comparable with FMDE. AFMDE0.1 outperforms FMDE on ZDT1 and ZDT3, does the same on ZDT6, but underperforms on ZDT2 and ZDT4. AFMDE0.1 underperforms FMDE on DTLZ2, DTLZ3 and DTLZ7, but does the same on DTLZ1, DTLZ4, DTLZ5, and DTLZ6. Overall, AFMDE is considered statistically better than FMDE on ZDT1 and ZDT3.

Table 4.3 shows results for WFG test suites. AFMDE0.1 and AFMDE with initial *CR* 0.9 (AFMDE0.9) have the same score. However, AFMDE0.1 outperforms FMDE on four problems (WFG1, WFG2, WFG7, and WFG8). It does the same on WFG6 and WFG9. AFMDE0.1 underperforms FMDE on WFG3 to WFG5 whereas AFMDE0.3 outperforms FMDE on WFG5 and does the same on the other problem. Overall AFMDE0.1 is comparable to FMDE, while AFMDE with other initial *CR* values underperform FMDE.

Overall, the initial *CR* = 0.1 is the most promising performance among other initial *CR* values. The parameters of *CR* fuzzy membership function will be investigated in the next section.

Table 4.2 FMDE compared with AFMDE on ZDT and DTLZ test suites

Test Functions		**FMDE**	**AFMDE initial *CR* = 0.1**	**AFMDE initial *CR* = 0.3**	**AFMDE initial *CR* = 0.5**	**AFMDE initial *CR* = 0.9**
ZDT1	Mean	4.800E-3	**4.300E-3**	5.300E-3	6.113E-1	2.751E-1
	Std.	2.560E-4	**1.2663E-4**	4.443E-4	1.517E-1	4.230E-2
	p-value		4.0772E-11	3.770E-4	3.020E-11	3.020E-11
	u-test		-	+	+	+
ZDT2	Mean	**4.700E-3**	5.200E-3	1.401E-1	9.311E-1	8.812E-1
	Std.	**1.9826E-4**	5.414E-04	1.985E-1	1.956E-1	2.701E-1
	p-value		4.943E-05	3.020E-11	3.020E-11	3.020E-11
	u-test		+	+	+	+
ZDT3	Mean	3.500E-3	**3.200E-3**	1.100E-2	5.074E-1	4.946E-1
	Std.	2.3176E-4	**1.5503E-04**	8.000E-3	8.740E-2	8.310E-2
	p-value		1.1937E-06	3.012E-11	3.020E-11	3.020E-11
	u-test		-	+	+	+
ZDT4	Mean	**6.949E-1**	2.548E+0	4.134E+0	4.020E+1	3.934E+1
	Std.	**6.792E-1**	3.217E+0	4.634E+0	5.966E+0	6.3135E+0
	p-value		2.800E-3	2.959E-5	3.020E-11	3.020E-11
	u-test		+	+	+	+
ZDT6	Mean	3.700E-3	**3.600E-3**	3.800E-3	5.000E-3	4.300E-3
	Std.	6.545E-4	**5.9401E-4**	8.6097E-4	1.200E-3	8.738E-4
	p-value		5.201E-1	9.234E-1	7.739E-6	4.900E-3
	u-test		=	=	+	+
DTLZ1	Mean	2.384E+2	**2.375E+2**	2.376E+2	2.384E+2	2.400E+2
	Std.	2.678E+0	**1.887E+0**	2.983E+0	3.272E+0	2.939E+0
	p-value		2.062E-1	1.809E-1	9.792E-5	4.210E-2
	u-test		=	=	+	+
DTLZ2	Mean	**6.900E-2**	7.360E-2	8.39E-2	9.320E-2	9.620E-2
	Std.	**2.600E-3**	3.700E-3	8.500E-3	7.600E-3	7.100E-3
	p-value		1.286E-6	6.696E-11	3.020E-11	3.020E-11
	u-test		+	+	+	+
DTLZ3	Mean	**7.205E+0**	1.833E+1	1.617E+1	1.090E+1	1.684E+1
	Std.	**6.492E+0**	1.518E+1	1.064E+1	8.792E+0	1.372E+1
	p-value		8.147E-5	3.988E-4	3.020E-11	3.182E-4
	u-test		+	+	+	+
DTLZ4	Mean	**7.240E-2**	8.530E-2	8.360E-2	8.760E-2	8.630E-2
	Std.	**5.800E-3**	6.280E-2	9.400E-3	1.110E-2	1.160E-2
	p-value		3.953E-1	6.736E-6	5.265E-5	5.859E-6
	u-test		=	+	+	+
DTLZ5	Mean	6.110E-1	6.171E-1	6.085E-1	6.218E-1	**6.077E-1**
	Std.	1.220E-2	9.900E-3	1.720E-2	2.820E-2	**1.970E-2**
	p-value		1.624E-1	6.627E-1	1.224E-1	4.464E-1
	u-test		=	=	=	=
DTLZ6	Mean	**6.091E-1**	6.113E-1	6.135E-1	6.134E-1	6.107E-1
	Std.	**9.100E-3**	9.100E-3	8.900E-3	9.600E-3	9.300E-2
	p-value		3.555E-1	9.620E-2	4.376E-1	7.062E-1
	u-test		=	=	=	=
DTLZ7	Mean	**4.920E-2**	5.410E-2	5.290E-2	5.590E-2	6.990E-2
	Std.	**1.900E-3**	5.000E-3	4.500E-3	7.100E-3	3.180E-2
	p-value		9.211E-5	5.300E-3	3.200E-11	1.429E-8
	u-test		+	+	+	+
	Better (+)		5	8	10	10
	Same (=)		5	4	2	2
	Worse (-)		2	0	0	0
	Score		3	8	10	10

Table 4.3 FMDE compared with AFMDE on WFG test suites

Test Functions		FMDE	AFMDE initial = 0.1	AFMDE initial = 0.3	AFMDE initial = 0.5	AFMDE initial = 0.9
WFG1	Mean	5.330E-1	5.384E-1	5.342E-1	5.348E-1	**5.329E-1**
	Std.	2.700E-3	3.300E-3	2.800E-3	3.100E-3	**3.000E-3**
	p-value		1.473E-7	1.494E-1	3.390E-2	6.414
	u-test		+	=	+	**=**
WFG2	Mean	4.292E-1	4.309E-1	**4.291E-1**	4.328E-1	4.308E-1
	Std.	4.859E-4	9.300E-3	**4.322E-4**	7.484E-4	8.297E-4
	p-value		6.736E-6	3.871E-1	3.020E-11	8.1014E-10
	u-test		+	=	+	+
WFG3	Mean	5.254E-1	**5.240E-1**	5.255E-1	5.286E-1	5.281E-1
	Std.	6.1244E-4	**3.355E-04**	4.275E-4	1.300E-3	1.200E-3
	p-value		7.389E-11	3.953E-1	6.066E-11	7.389E-11
	u-test		-	=	+	+
WFG4	Mean	4.011E-1	**3.952E-1**	4.113E-1	4.370E-1	4.350E-1
	Std.	4.300E-3	**5.400E-3**	7.500E-3	6.900E-3	6.100E-3
	p-value		7.199e-5	2.783E-7	3.020E-11	3.020E-11
	u-test		-	+	+	+
WFG5	Mean	1.854E-1	**1.844E-1**	1.851E-1	1.880E-1	1.876E-1
	Std.	1.200E-3	**1.400E-3**	1.300E-3	1.100E-3	1.300E-3
	p-value		5.874E-4	3.112E-1	1.070E-9	9.833E-8
	u-test		-	=	+	+
WFG6	Mean	**3.854E-1**	3.857E-1	3.853E-1	3.857E-1	3.856E-1
	Std.	**1.000E-3**	9.991E-4	9.307E-4	1.100E-3	1.200E-3
	p-value		5.895E-1	4.370E-1	7.506E-1	8.303E-1
	u-test		=	=	=	=
WFG7	Mean	**8.360E-2**	9.730E-2	7.990E-2	1.011E-1	9.220E-2
	Std.	**1.060E-2**	1.290E-2	1.030E-2	1.090E-2	1.840E-2
	p-value		8.147E-5	1.669E-1	3.256E-7	8.240E-2
	u-test		+	=	+	=
WFG8	Mean	**4.570E-1**	4.585E-1	4.563E-1	4.571E-1	4.565E-1
	Std.	**2.000E-3**	1.600E-3	1.700E-3	3.300E-3	2.900E-3
	p-value		4.714E-4	6.570E-2	6.843E-1	4.376E-1
	u-test		+	=	=	=
WFG9	Mean	**1.064E-1**	1.065E-1	1.076E-1	1.139E-1	1.107E-1
	Std.	**4.300E-3**	5.800E-3	5.800E-2	9.600E-3	6.400E-3
	p-value		6.100E-1	4.119E-1	5.561E-4	8.700E-3
	u-test		=	=	+	+
	Better (+)		4	1	7	5
	Same (=)		2	8	2	4
	Worse (-)		3	0	0	0
	Score		1	1	7	5

4.3.2 AFMDE and various membership function parameters

The impact of initial *CR* value was investigated in Subsection 4.3.1. The experimental results from Subsection 4.3.1 show that AFMDE with the initial *CR* 0.1 delivered the best results among other initial *CR* values. Hence, AFMDE0.1 membership function parameters will be investigated in this Subsection.

The input membership functions are the same as Figure 4.1. The output membership function for γ and *F* are the same as Figure 4.2. In order to investigate the influence of the range of the percentage change of *CR*, the parameters a1, a2, b1, b2, c1, and c2 are varied for μ_{CR} but kept constant for μ_F and μ_γ as shown in Table 4.4. The other parameter settings such as population size, maximum number of function evaluations are the same as the previous experiments in Subsection 3.6.

The experimental results for FMDE and AFMDE1 to AMFMDE4 on ZDT and DTLZ test instances are shown in Table 4.5. AFMDE4 outperforms FMDE on seven test instances, i.e., ZDT1, ZDT2, ZDT3, ZDT6, DTLZ1, DTLZ2, and DTLZ4. AFMDE underperforms FMDE on ZDT4, and DTLZ3 and does the same on DTLZ5, DTLZ6, and DTLZ7. Overall score of AFMDE4 for ZDT and DTLZ suites shows that AFMDE outperforms the others.

Table 4.6 shows the results on WFG test suite. AFMDE3 and AFMDE4 scores are the same and better than AFMDE1, and AFMDE2, but comparable to FMDE. They outperform FMDE on four problems, i.e., WFG2, WFG3, WFG4, and WFG5. They do the same on one WFG9 and underperform on WFG1, WFG6, WFG7, and WFG8 over FMDE.

Considering overall performance for all test instances, AFMDE4 shows the best performance among AFMDE1 to AFMDE4 and FMDE.

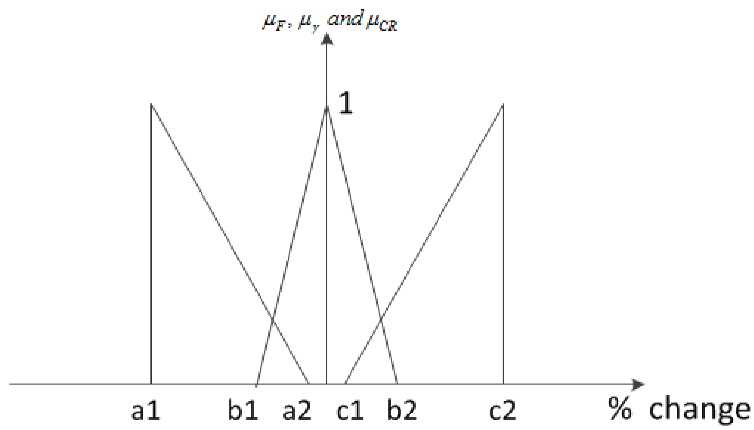


Figure 4.1 Output membership functions

Table 4.4 Output fuzzy membership function parameters

Parameters	μ_γ and μ_F	μ_{CR}			
		AFMDE1	AFMDE2	AFMDE3	AFMDE4
a1	-50	-50	-30	-20	-30
a2	-10	-10	-10	-10	-15
b1	-20	-20	-15	-15	-20
b2	20	20	15	15	20
c1	10	10	10	10	15
c2	50	50	30	20	30

Table 4.5 FMDE compared with AFMDE1 to AFMDE4 on ZDT and DTLZ test suites

Test Functions		FMDE	AFMDE1	AFMDE2	AFMDE3	AFMDE4
ZDT1	Mean	4.800E-3	4.300E-2	4.300E-2	4.200E-2	4.300E-2
	Std.	2.560E-4	1.266E-4	1.863E-4	**1.566E-4**	3.566E-4
	p-value		4.077E-11	1.464E-10	8.993E-11	3.020E-11
	u-test		-	-	-	-
ZDT2	Mean	4.700E-3	5.200E-3	6.200E-3	9.200E-3	**4.500E-3**
	Std.	1.9826E-4	5.414E-4	9.100E-3	2.580E-2	**1.115E-4**
	p-value		4.943E-5	5.100E-3	2.433E-5	3.020E-11
	u-test		+	+	+	-
ZDT3	Mean	3.500E-3	3.200E-3	**3.100E-3**	**3.100E-3**	**3.100E-3**
	Std.	2.318E-4	1.550E-4	**1.149E-4**	**1.058E-4**	**1.515E-4**
	p-value		1.194E-6	2.034E-9	6.722E-10	2.390E-8
	u-test		-	-	-	-
ZDT4	Mean	**6.949E-1**	2.5479E+0	3.0841E+0	5.2346E+0	3.891E+0
	Std.	**6.792E-1**	3.2171E+0	2.953E+0	3.3792E+0	3.669E+0
	p-value		2.800E-3	1.385E-6	3.825E-9	1.254E-7
	u-test		+	+	+	+
ZDT6	Mean	3.700E-3	3.600E-3	3.400E-3	**3.300**E-3	3.400E-3
	Std.	6.545E-4	5.940E-4	4.093E-4	**3.463E-4**	3.566E-4
	p-value		5.201E-1	3.150E-2	8.000E-3	4.900E-3
	u-test		=	-	-	-
DTLZ1	Mean	2.384E+2	2.375E+2	2.368E+2	2.366E+2	**2.365E+2**
	Std.	2.678E+0	1.887E+0	1.949E+0	1.774E+0	**1.776**E+0
	p-value		2.062E-1	1.990E-2	4.600E-3	3.300E-3
	u-test		=	-	-	-
DTLZ2	Mean	6.900E-2	7.360E-2	6.520E-2	**6.230E-2**	6.330E-2
	Std.	2.600E-3	3.700E-3	2.000E-3	**1.800E-3**	2.300E-3
	p-value		1.286E-6	2.028E-7	4.504E-11	8.101E-10
	u-test		+	-	-	-
DTLZ3	Mean	**7.205E+0**	1.833E+1	2.084E+1	1.916E+1	1.570E+1
	Std.	**6.492E+0**	1.518E+1	1.431E+1	1.348E+1	1.155E+1
	p-value		8.147E-5	4.943E-5	4.943E-5	5.874E-4
	u-test		+	+	+	+
DTLZ4	Mean	7.240E-2	8.530E-2	6.850E-2	6.520E-2	**6.450E-2**
	Std.	5.800E-3	6.280E-2	6.800E-3	7.400E-3	**4.500E-2**
	p-value		3.953E-1	1.990E-2	8.663E-5	2.678E-6
	u-test		=	-	-	-
DTLZ5	Mean	**6.110E-1**	6.171E-1	6.176E-1	6.167E-1	6.137E-1
	Std.	**1.220E-2**	9.900E-3	9.200E-3	8.900E-3	1.010E-3
	p-value		1.624E-1	0.0963	1.494 E-1	6.843E-1
	u-test		=	=	=	=
DTLZ6	Mean	**6.091E-1**	6.113E-1	6.136E-1	6.157E-1	6.133E-1
	Std.	**9.100E-3**	9.100E-3	8.600E-3	7.800E-3	9.900E-3
	p-value		3.555E-1	5.190E-2	5.300E-3	7.980E-2
	u-test		=	=	+	=
DTLZ7	Mean	**4.920E-2**	5.410E-2	6.360E-2	6.430E-2	7.280E-2
	Std.	**1.900E-3**	5.000E-3	6.050E-2	6.020E-2	7.220E-2
	p-value		9.211E-5	3.000E-3	7.730E-2	6.627E-1
	u-test		+	+	+	=
	Better (+)		5	4	5	2
	Same (=)		5	2	1	3
	Worse (-)		2	6	6	7
	Score		5	-2	-1	-5

Table 4.6 FMDE compared with AFMDE1 to AFMDE4 on WFG test suite

Test Functions		FMDE	AFMDE1	AFMDE2	AFMDE3	AFMDE4
WFG1	Mean Std. p-value u-test	**5.330E-1** **2.700E-3**	5.384E-1 3.300E-3 1.473E-7 +	5.389E-1 2.900E-3 1.011E-08 +	5.380E-1 3.100E-3 1.254E-7 +	5.379E-1 2.800E-3 1.873E-7 +
WFG2	Mean Std. p-value u-test	4.292E-1 4.859E-4	4.309E-1 9.300E-3 6.736E-6 +	**4.286E-1** **4.710E-4** 2.154E-06 -	4.298E-1 6.700E-3 5.265E-5 +	4.287E-1 4.088E-4 8.663E-5 -
WFG3	Mean Std. p-value u-test	5.254E-1 6.1244E-4	**5.240E-1** **3.355E-4** 7.389E-11 -	5.241E-1 3.170E-4 8.153E-11 -	5.240E-1 3.618E-4 8.153E-11 -	5.241E-1 3.155E-4 9.919E-11 -
WFG4	Mean Std. p-value u-test	4.011E-1 4.300E-3	3.952E-1 5.400E-3 7.1988E-5 -	3.910E-1 5.000E-3 8.485E-9 -	3.880E-1 4.400E-3 1.957E-10 -	**3.878E-1** **4.700E-3** 1.777E-10 -
WFG5	Mean Std. p-value u-test	1.854E-1 1.200E-3	**1.844E-1** **1.400E-3** 5.874E-4 -	1.846E-1 1.200E-3 1.120E-2 -	1.847E-1 1.300E-3 3.030E-2 -	1.847E-1 1.300E-3 1.910E-2 -
WFG6	Mean Std. p-value u-test	**3.854E-1** **1.000E-3**	3.857E-1 9.991E-4 5.895E-1 =	3.858E-1 7.7364E-4 1.055E-1 +	3.858E-1 7.718E-4 1.413E-1 =	3.858E-1 6.399E-4 1.910E-2 +
WFG7	Mean Std. p-value u-test	**8.360E-2** **1.060E-2**	9.730E-2 1.290E-2 8.147E-5 +	9.990E-2 1.210E-2 9.514E-6 +	1.012E-1 1.310E-2 2.317E-6 +	1.000E-1 1.570E-2 9.792E-5 +
WFG8	Mean Std. p-value u-test	**4.570E-1** **2.000E-3**	0.4585E-1 0.0016E-3 4.7138E-4 +	4.583E-1 1.800E-3 5.561E-4 +	4.582E-1 7.7751E-4 1.585E-4 +	4.583E-1 1.000E-3 2.531E-4 +
WFG9	Mean Std. p-value u-test	**1.064E-1** **4.300E-3**	1.065E-1 5.800E-3 6.100E-1 =	1.060E-1 4.800E-3 9.941E-1 =	1.059E-1 4.900E-3 7.394E-1 =	1.061E-1 4.600E-3 8.766E-1 =
	Better (+) Same (=) Worse (-) Score		4 2 3 1	4 1 4 0	4 2 3 1	4 1 4 0

4.4 Remarks

The advanced version of FMDE incorporates the adaptive *CR* mechanism. The experimental results show that the initial *CR* value in the range of 0.3 to 0.9 is not sensitive to performance of AFMDE on ZDT and DTLZ test suites but sensitive to WFG test suite. The suitable initial *CR* value is 0.1. The investigation on membership function parameters shows that if the maximum percentage change of *CR* value is 30, AFMDE produces the most promising results and is not sensitive to the algorithm performance as well. If the maximum percentage change of *CR* value is greater than 30, AFMDE performance will be worsen.

CHAPTER V

5 BY 5 MICROSTRIP ANTENNA ARRAY SYNTHESIS FOR 12.5 GHZ BROADCASTING SATELLITE SERVICE

The concept of AFMDE was presented in the previous chapter. AFMDE performance is quantified by conducting the experiments on the well-known benchmark test suites. The results show the competitive performance. In order to prove its practical usage, AFMDE is applied to a microstrip antenna array design as a case study. The microstrip antenna design is formulated as a three-objective optimization problem. Searching for a set of the trade-off optimal solutions can be done by AFMDE. Furthermore, the objective evaluation of a 5 by 5 microstrip antenna array is computationally expensive. Hence, a radial basis function neural network is developed as a surrogate model for the objective function evaluations.

5.1 Introduction

Microstrip antennas are low-profile lightweight antennas and are being use in various wireless communication systems [17-18]. A 5x5 microstrip patch antenna array synthesis for operating at 12.5 GHz broadcasting satellite service (BSS) is formulated as a MOP. The design goals for the antenna array involve three competing objectives- maximizing gain, minimize side lobe level and minimize the reflection coefficient. Since, the microstrip antenna array design is formulated as a MOP, and then AFMDE, a multiobjective optimizer, can be used to tackle this design problem. AFMDE searches for the optimal antenna array configurations which are the optimal trade-off solutions. Therefore, the end users (the manufacturers) will make their own decision to choose an optimal solution from the optimal set based upon the higher level information such as the manufacturing cost and the radiation pattern.

Generally, the evaluation of the objective values (gain, side lobe level and the reflection coefficient) can be computed by the commercial software such as COMSOL, FEKO, or IE3D. They are expensive, and the analysis of the radiation pattern is high computational expensive and time consuming. Hence, we create a surrogate model [94] by utilizing a radial basis neural network (RBF) in order to overcome these problems.

The rest of this Chapter is organized as follows. Section 5.2 states the problem formulation and radial basis neural network as a surrogate model. Section 5.3 presents the experimental setup and results. Remarks will be given in Section 5.4.

5.2 The Microstrip Antenna Array Synthesis Formulation

The proposed AFMDE incorporates with a radial basis neural network is an optimizer to tackle a 5x5 microstrip antenna array operates at 12.5 GHz satellite broadcasting services [89] synthesis. In this section, we will describe the problem formulation, i.e., array configuration, the objective functions, a surrogate model and the framework of the design and optimization process.

5.2.1 5 by 5 microstrip antenna array architecture and design goals

The 5x5 microstrip antenna configuration is shown in Figure 5.1. Spasos *et al.* [89] proposed this configuration. For simplicity, the array was formed as a planar array with one feed point and each patch are connected by a serial feed line. Each patch size is equal in size and in a square shape.

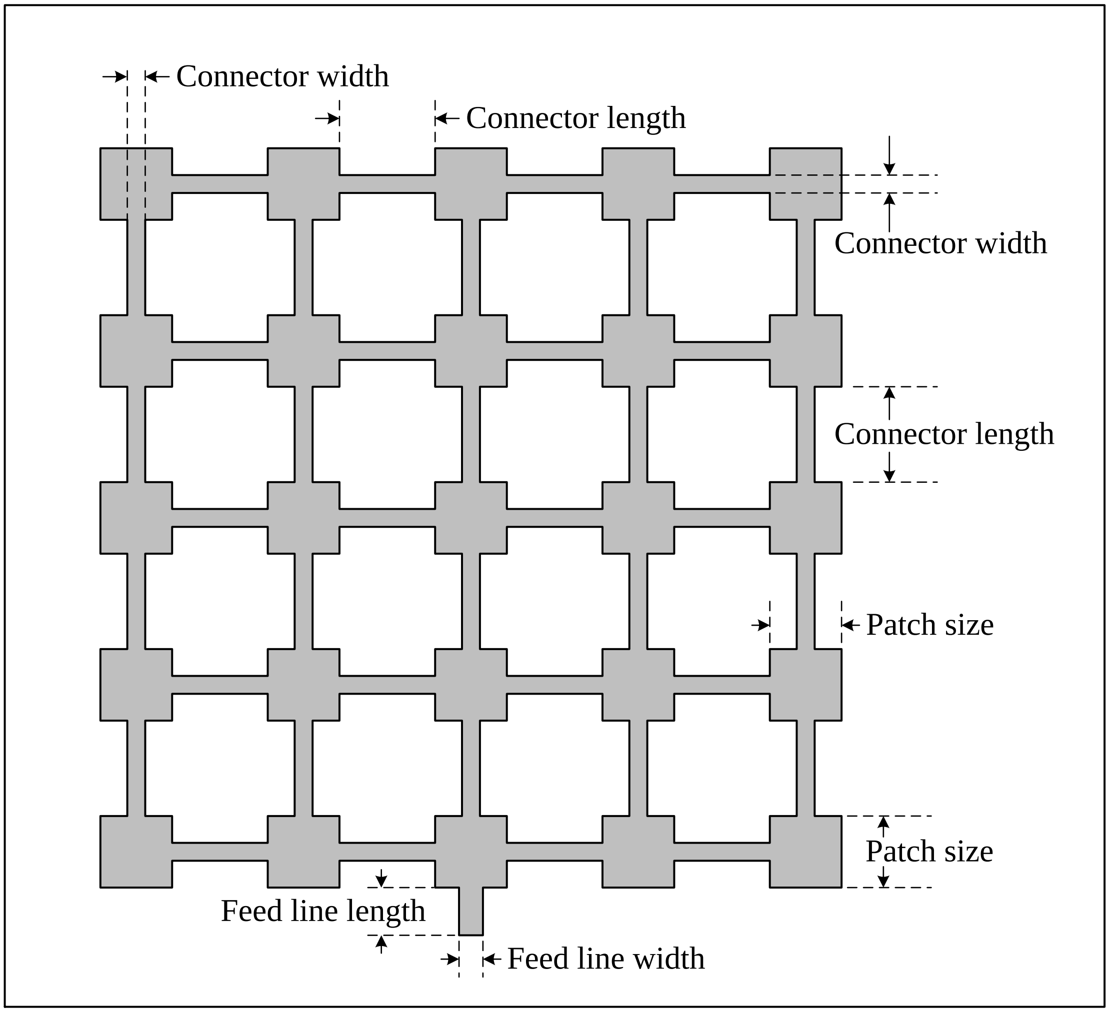


Figure 5.1 5 by 5 Microstrip antenna array

AFMDE decision variables consist of patch side, feed line width, feed line length, connector width, and connector length. Therefore, the dimension of each decision vector is five.

There are three objectives of this optimization problem, i.e., maximizing gain (*G*), while minimizing side lobe level (*SLL*) and the reflection coefficient ($S11_{dB}$).

The flowchart for the AFMDE that is applied to 5x5 microstrip antenna design is shown in Figure 5.2. It starts with the same process as AFMDE except for the objective evaluation. RBF will be used as objective evaluation.

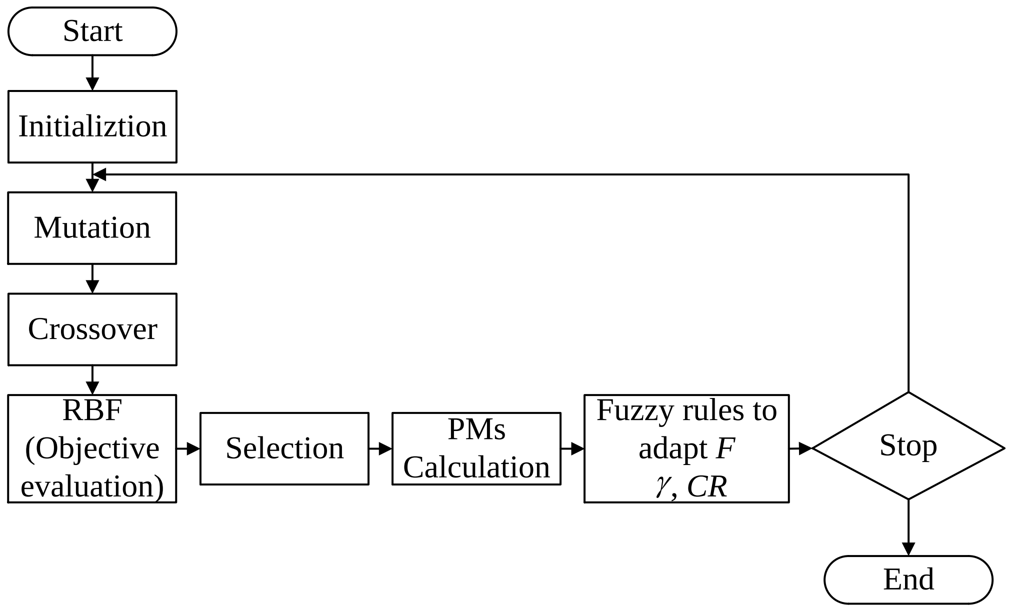


Figure 5.2 5x5 Microstrip antenna array synthesis by AFMDE

5.2.2 *A surrogate model by Radial Basis Neural Network*

Most contemporary engineering field, computer simulations are used extensively on both design verification and during design process in order to improve the system reliability or reduce manufacturing cost. The design goals can be modelled as objective functions as in optimization problems. Then those objective values are obtained by computer simulations. However, those objective functions are often analytically intractable, sensitivity information is unavailable or too expensive to compute [94]. Objective evaluation can be highly computationally expensive, time-consuming task (several hours, days or weeks per objective function evaluation) despite the increase of available computing power. In order to overcome this difficulty, the expensive computation will be replaced by a surrogate model. The surrogate model should be at least locally accurate representation of the real one. There are various types of surrogate modelling techniques. For instance, polynomial regression, Kriging, and support vector regression are a few popular choices. Among the surrogate modelling approaches, the radial basis neural network (RBF) is one of the most widely used technique. RBF was proposed by Broomhead and Lowe in 1988 [99] as an approximation technique. The good generalization ability, simple network architecture, and fast training make RBF popular in many applications, for instance, pattern classifications, function approximation, signal processing and control [90]. The architecture of the RBF is shown in Figure 5.3. RBF performs a mapping from an *m*-dimensional input space to an *n*-dimensional output space.

The *n*-dimensional input vector $\mathbf{x} = \left[x_i\right]^T$ being passed directly to a hidden layer, assuming there are *m* neurons. Each of these neuron in the hidden layer applies a radial basis function which is the Gaussian function vector $\mathbf{h} = \left[h_j\right]^T$ when h_j is the Gaussian function value for the neuron *j* in the hidden layer [91],

$$h_j = \exp\left(-\frac{\left\| \mathbf{x} - c_j \right\|^2}{2b_j^2} \right), \tag{5.1}$$

where $\mathbf{c} = \left[c_{ji}\right]$ represents the center point of the Gaussian function of the *j*th neuron from the *i*th input, *i* = 1,2,…,*n*, *j* = 1,2,…,*m*. $\mathbf{b} = \left[b_1, ..., b_m\right]^T$, b_j represents the width value of Gaussian function for the *j*th neuron. The weight value of RBF is given in (5.2)

$$\mathbf{w} = \left[w_1, ..., w_m\right]^T \tag{5.2}$$

The output of RBF neural network consists of sums of the weighted hidden layer neurons as the following:

$$\hat{y}_m = w_1 h_1 + w_2 h_2 + ... + w_m h_m \,. \tag{5.3}$$

Therefore, the center and the width of the Gaussian in (5.1) are the control parameters for the network. There are many ways to choose the center and the width of a Gaussian function. We randomly choose the center from the training data set, and calculate the width from (5.4) [92]

$$b = \frac{d_{\max}}{\sqrt{2m}} \,, \tag{5.4}$$

where d_{max} is the maximum distance between the selected center, m is the number of the hidden neurons.

The gradient descent method is used to train RBF. The parameter can be updated as the following:

$$E(t) = \sum_{l=1}^{N} (y_l - \hat{y}_l)^2 \tag{5.5}$$

$$\Delta w_j(t) = -\eta \frac{\partial E}{\partial w_j} = \eta \sum_{l=1}^{N} e_l h_j \tag{5.6}$$

$$e_l = y_l - \hat{y}_l \tag{5.7}$$

where $\eta \in (0,1)$ is the learning rate, $\hat{y}_l$ is the training output, y_l is the target output, and l = 1, 2,…, N, N is the number of a training data point.

Our chosen RBF architecture is 5-10-3. The empirically rules-of-thumb for choosing the number of neurons in the hidden layer in a feed forward neural network should be between the size of input and the size of the output layer [140]. However, our primary experiment on training the RBF shows that ten neurons in the hidden layer are the most suitable for our RBF. The training method is the gradient descent. Once the RBF is trained, we will use the network for the objective function evaluation in AFMDE. The RBF weights training process is shown in Figure 5.4. The input data are the antenna configuration (decision variables) and the target data are the objective values i.e., G, SLL, and $S11_{dB}$.

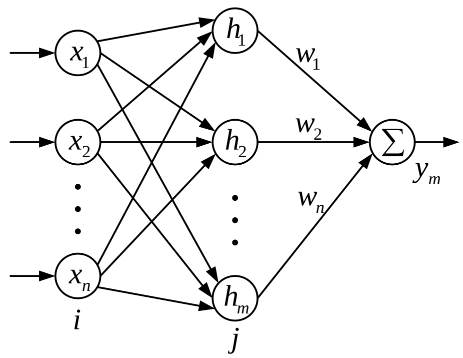


Figure 5.3 Radial Basis Neural Network

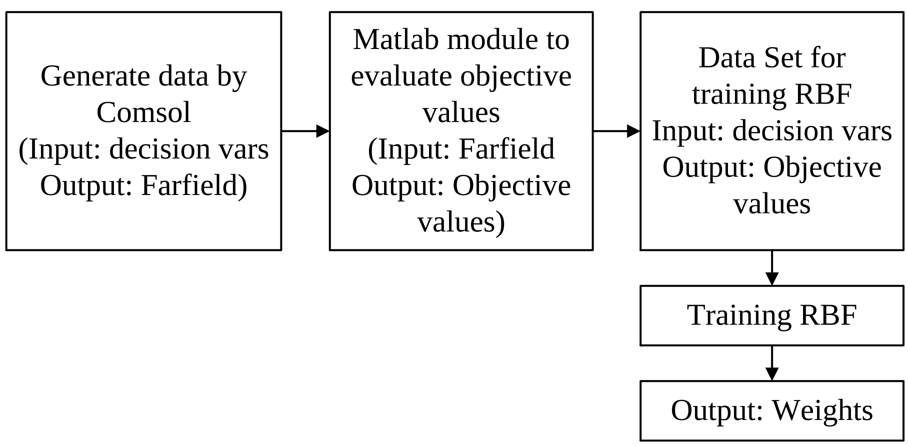


Figure 5.4 Training RBF

There is no deterministic approach to set the number of training data. At the beginning, 30 data points are used for training and measuring the average minimum distance between population and data these data points. If the distance is small, it means RBF approximates the objective values near the training data throughout the evolution process. Otherwise we increase the number of training data points and monitor the average minimum distance. The average minimum distance for 70 data points has provided the best approximation given the limited amount of time. Therefore, we stop collecting data point and use 70 data points to train RBF. The average minimum distance for 70 data points is shown in Figure 5.5. The training was stopped at 1,252 iterations. The error is 2.142E+3. However, the error remains high. This implies we may need more data points and different training approach or network architecture in order to improve the accuracy of the RBF which will be include in our future work.

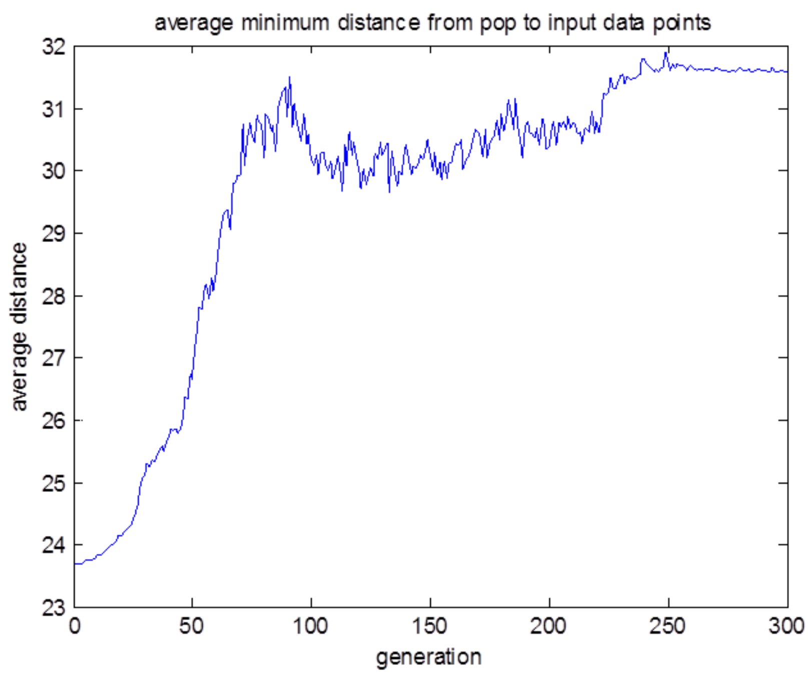


Figure 5.5 Average minimum distance from population to input data points

5.3 Experimental Setup and Results

5.3.1 Experimental setup

Spasos *et al.* [89] proposed a 12.5 GHz microstrip antenna array synthesis using Taguchi's method, with interconnected elements in order to achieve good matching (S11 < -10 dB) without using any matching networks and high gain, suitable for the 12.5 GHz broadcasting satellite service (BSS) frequency bands. It was claimed to be the current state-of-the-art in this technical field. As a result, it has been chosen as a competitor with respect to the approach proposed in this dissertation, AFMDE.

This antenna array consists of equally spaced rectangular patches joined together with microstrip lines and is fed through a simple microstrip line without any matching network.

The parameters for antenna:

1. The chosen substrate is the microwave laminate RT/duroid 5880 from Rogers with relative permittivity $\varepsilon_r = 2.2$, loss tangent tan δ = 0.0009, and thickness = 1.575 mm.

2. Operating frequency is 12.5 GHz.

3. Free air wavelength at 12.5 GHz = 24 mm.

The objective functions:

1. f1: maximizing *G*

2. f2: minimizing *SLL*

3. f3: minimize $S11_{dB}$

Decision vector:

The dimension of decision vector is 5, including patch side (patch_s), feed width (feed_w), feed line (feed_l), connector width (con_w), and connector length (con_l),

patch_s (patch side, [mm]) (9.6 - 14.4)	feed_w (feed line width, [mm]) (4.8 - 7.2)	feed_l (feed line length, [mm]) (9.6 - 14.4)	con_w (connector width, [mm]) (2.4 - 3.6)	con_l (connector, length [mm]) (14.4 - 21.6)

Parameter setting for AFMDE: population size *NP*=100, the maximum number of generation is 300, and the external archive size is 100. AFMDE will be run for 30 independent trials. The extreme solutions of each run will be compared. The solution with the highest gain will be the winner, and compared with results from Spasos *et al.* [89].

5.3.2 Results

The obtained Pareto front from AFMDE is shown in Figure 5.6. It can be seen from Figure 5.7 that the population becomes fast converging, but loss its diversity and the extensiveness of extreme solutions. Around 250th generation, γ is decreasing, but F and CR are increasing. This infers that the search process detects the promising region and fast converges to that direction. After approximately 255th generation, γ is closed to zero, but F is still high and near one. CR is decreasing. This implies that the degree of exploitation is higher than exploration. It means that the algorithm is converging and tries to perform the local search to make the obtained front evenly distributed.

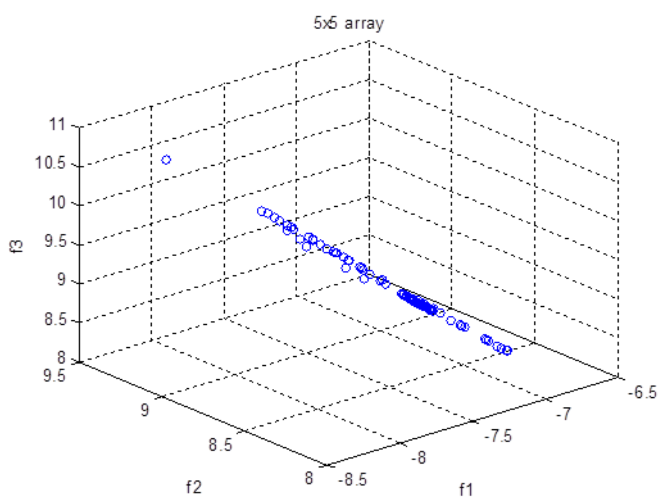
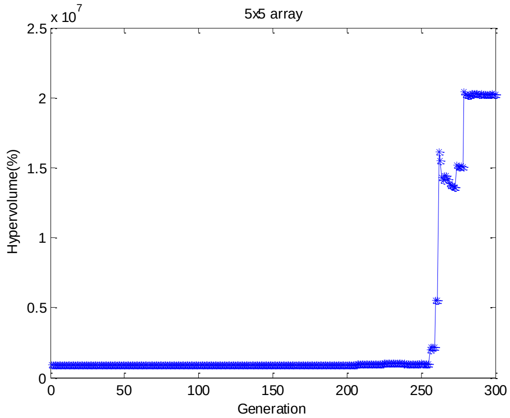
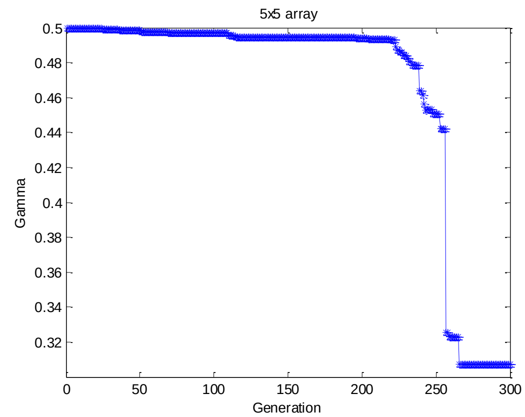


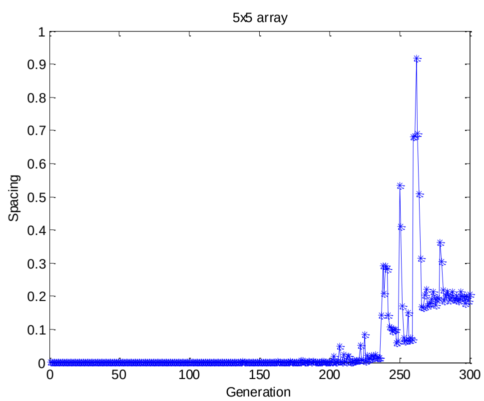
Figure 5.6 The obtained Pareto front



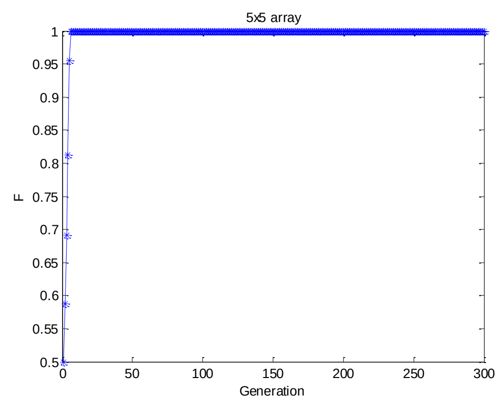
(a)



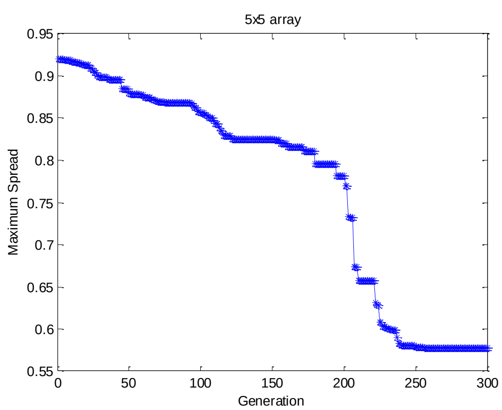
(b)



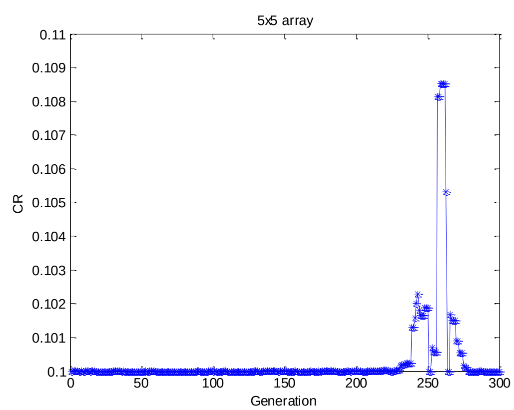
(c)



(d)



(e)



(f)

Figure 5.7 The AFMDE associated control parameters

Since there are multiple optimum solutions from the approximation Pareto front, the extreme solutions in f1, f2 and f3 dimension is selected as the representatives to be compared with Spasos *et al.* [89] which use Taguchi's method to design the microstrip antenna array. One main advantage of the proposed MODE method should be made clear. It provides a set of Pareto optimal solutions, not only including those extreme solutions, but compromised solutions among the objectives. These solutions provide numerous Pareto optimal choices for the decision-maker under various trade-offs under real-world complications. The objective values with respect to the decision variables are shown in Tables 5.1 and 5.2. We can see that the extreme solution in f2 dimension demonstrates AFMDE's ability to search for a solution better than Taguchi's method in terms of side lobe level reduction. In addition, the extreme solution in f3 dimension shows that AFMDE can find a solution that better than Taguchi's method solution in term of the reflection coefficient.

Figures 5.8 to 5.10 show the 5 by 5 microstrip antenna array and the radiation pattern of the AFMDE extreme solution in f1 dimension. Figure 5.8 presents the antenna configuration in 3D. The square patch at the center is the microstrip antenna array. The half circle represents the perfectly matched layer (PML) which is used to truncate the computational domain. The PML has its function to absorb the outgoing wave and suppress the reflected wave. The propagation of electric field along the microstrip line and on the patch antenna is shown in Figure 5.9. The electric field energy from feed point propagates from the feed point to the center of the antenna. The microwave does not propagate throughout the whole patches. Consequently, the obtained G is lower than Taguchi's method. Figure 5.10 shows the far field radiation pattern in three dimension space. It can be seen that the major lobe has its direction pointed in z axis which is reasonable with the configuration of the array that is shown in Figure 5.8. Figure 5.11 represents a y plane (x-z cut) of the far field radiation pattern of Figure 5.10.

The electric field propagation on 5 by 5 microstrip antenna array, radiation pattern in 3D and 2D (y-plane) of the extreme solution in f2 and f3 dimensions are shown in Figure 5.12, 5.13 and 5.14 respectively.

Table 5.1 Objective values

	f1 (G)	f2 (*SLL*)	f3 ($S11_{dB}$)
Taguchi's method	15	-13	-7
AFMDE extreme solution in f1 dimension	9.2678	-8.2131	-10.7473
AFMDE extreme solution in f2 dimension	6.8500	-13.5501	-11.4501
AFMDE extreme solution in f3 dimension	7.1330	-10.2214	-12.7460

Table 5.2 Decision variables

	patch_s [mm]	feed_w [mm]	feed_l [mm]	con_w [mm]	con_l [mm]
Taguchi's method	10.823	6.875	11.196	2.498	16.569
AFMDE extreme solution in f1 dimension	10.8506	6.9598	15.3574	2.6114	21.9665
AFMDE extreme solution in f2 dimension	0.8984	5.4224	4.8421	0.3340	8.9841
AFMDE extreme solution in f3 dimension	2.4386	5.4941	7.2262	2.5037	8.5808

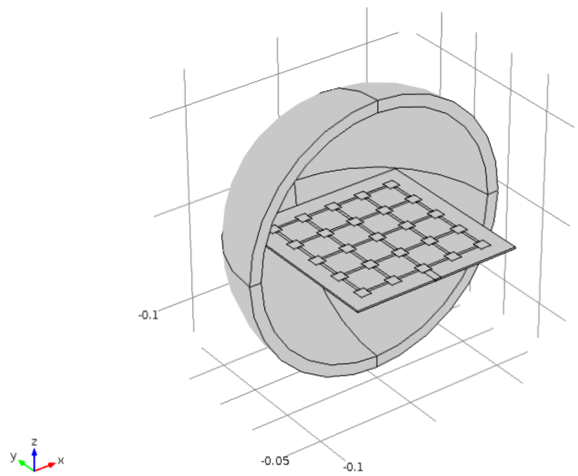


Figure 5.8 5 by 5 microstrip antenna array configuration

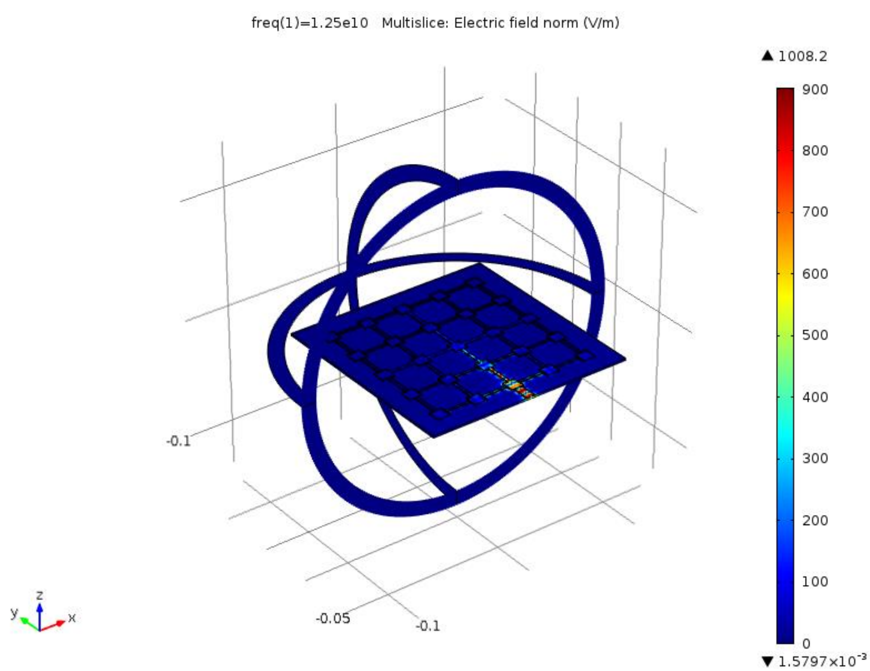


Figure 5.9 Electric field propagation on 5 by 5 microstrip antenna array

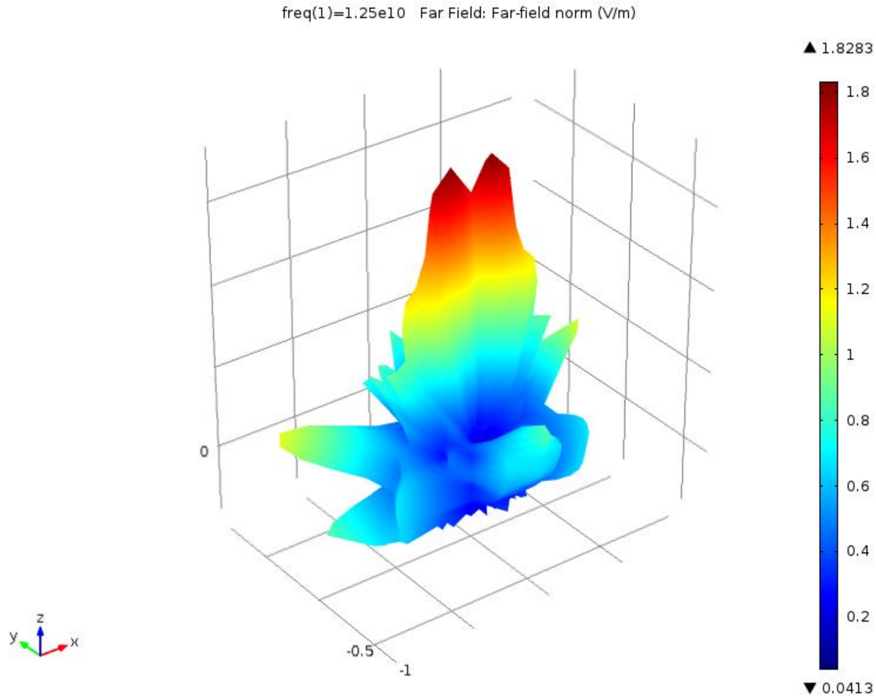


Figure 5.10 5 by 5 microstrip antenna array pattern in 3D

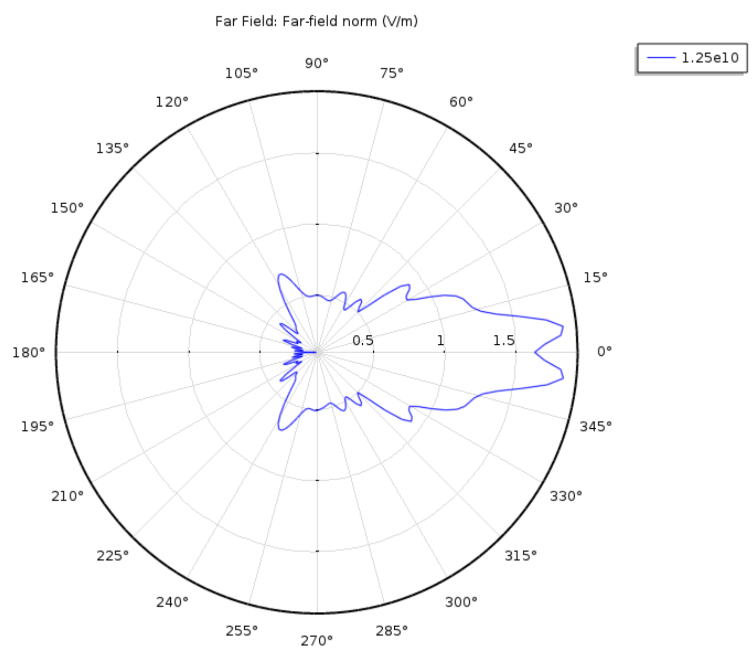
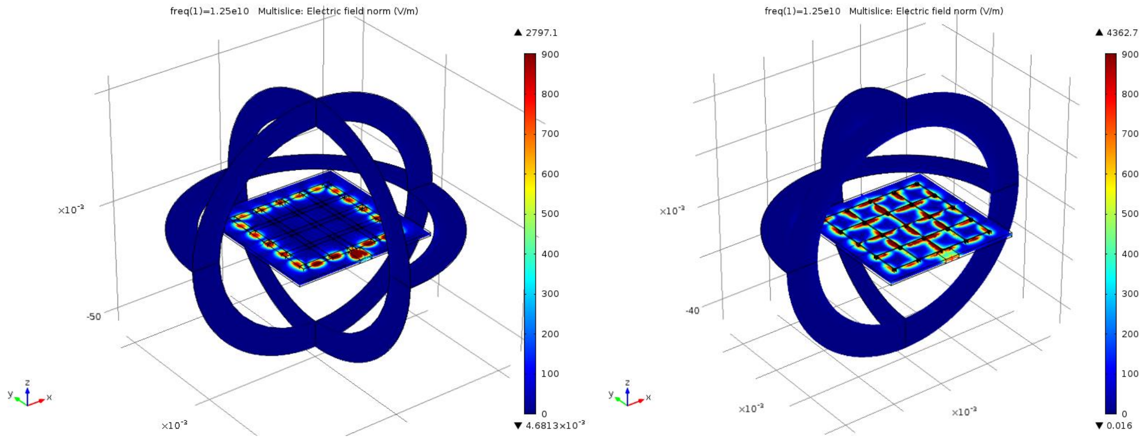
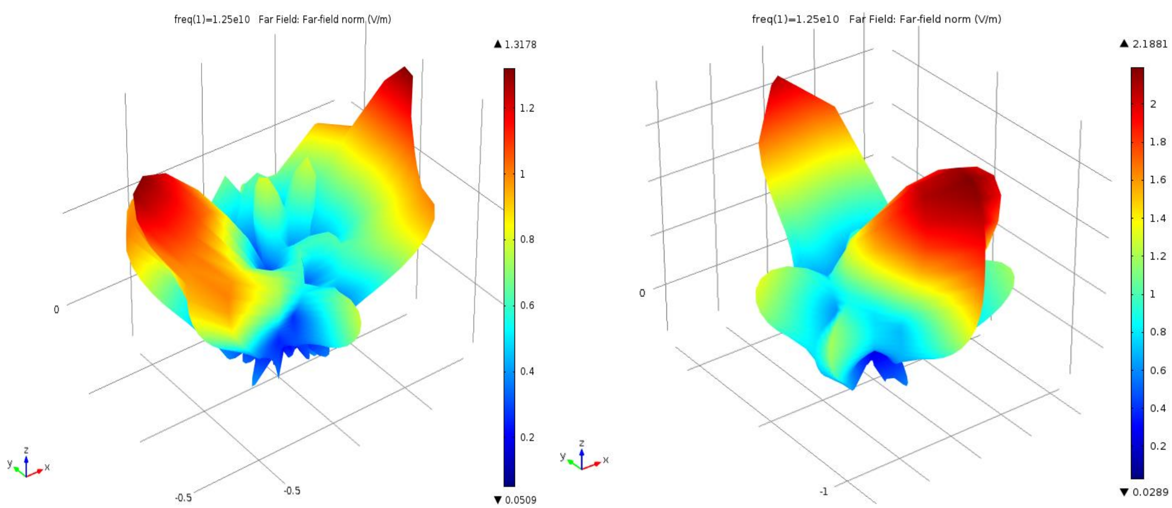


Figure 5.11 Far field radiation pattern in 2D on y plane (x-z cut)



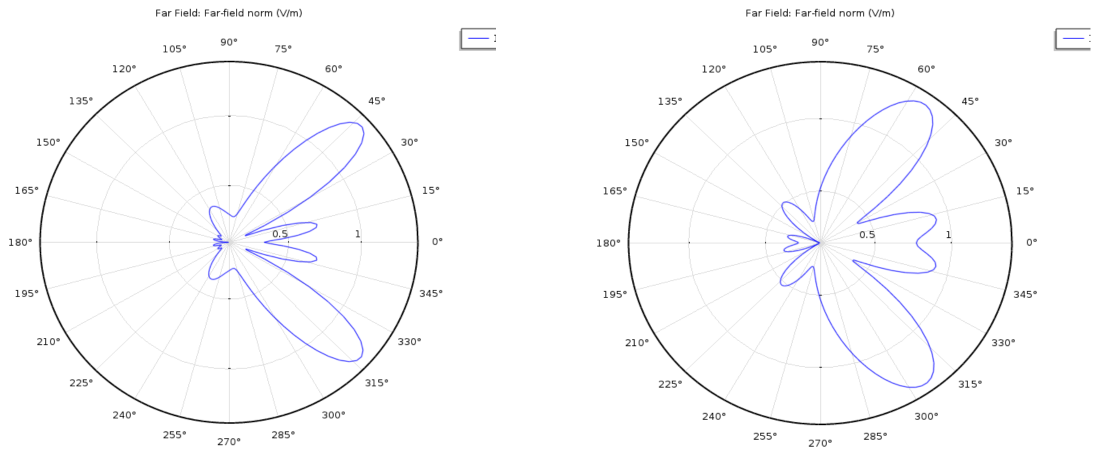
(a) (b)

Figure 5.12 Electric field propagation on 5 by 5 microstrip antenna array for the extreme solution in (a) f2 dimension (b) f3 dimension



(a) (b)

Figure 5.13 5 by 5 microstrip antenna array pattern in 3D for the extreme solutions in (a) f2dimension (b) f3 dimension



(a) (b)

Figure 5.14 Far field radiation pattern in 2D on y plane (x-z cut) for the extreme solutions in f2 and f3 dimension

5.4 Remarks

The 5 by 5 microstrip antenna array synthesis can be formulated as a MOP. AFMDE can be applied to tackle this problem. Radial basis function network is used as a surrogate model for the objective function evaluation because the computational cost is much lower than that of the commercial software. The design of RBF surrogate model may demand a higher number of data points or different training approach in order to improve the performance of the RBF. Asides, the experimental results demonstrate the ability of finding not only one, but a set of quality solutions that were never made available before specifically in terms of side lobe level and reflection coefficient. To sum up, AFMDE provides a set of Pareto optimal designs for the 5 by 5 microstrip antenna array. The end users (e.g. manufacturers) can choose an optimal design that meets their different requirements such as different main lobe direction, specific side lobe reduction, etc.

CHAPTER VI

SOFT CONSTRAINT HANDLING FUZZY MULTIOBJECTIVE DIFFERENTIAL EVOLUTION AND ITS APPLICATION IN A CONSTRAINED NON-UNIFORM CIRCULAR ANTENNA ARRAYS DESIGN PROBLEM

Optimization problems can be free from any constraints. However, there are many real world optimization problems that involve various types of constraints. These problems are known as constrained optimization problems (COPs). If a COP involves more than one objective and a number of constraints, it is called a constrained multiobjective optimization problem (CMOP). In order to handle COPs, we search for the optimal solution within the feasible region. The search for feasible solutions is time-consuming and often leads to computational difficulties. The constraints can be classified into the hard and soft constraints. The hard constraints cannot be violated because it can cause the critical failures [96-97] while the soft ones can be relaxed to some extent if the violations of them do not compromise the purpose of the requirements. Under the soft constraint circumstances, the decision makers often prefer solutions with compromised tradeoff between solution quality and constraint violation. It is implying that some constraints may be relaxed within acceptable ranges to gain the performance improvement in some objective functions. Therefore, a soft constraint handling approach is proposed in this chapter. AFMDE is integrated with the proposed soft constraint handling method. It is tested on benchmark functions and applied to a constrained non-uniform circular antenna array designs as a case study.

6.1 Introduction

In order to illustrate the soft constraint handling concept, we use a single objective benchmark problem, g6 [100] as an example. The formulation of g6 is the following:

Minimize

$$f(x) = (x_1 - 10)^3 + (x_2 - 20)^3,$$

subject to

$$g_1(x) = (x_1 - 5)^2 + (x_2 - 5)^2 - 100 \geq 0$$

$$g_2(x) = -(x_1 - 6)^2 - (x_2 - 5)^2 + 82.81 \geq 0$$

$$x_1 \in [13,100], \quad x_2 \in [0,100].$$

The optimization results are shown in Table 6.1. The overall constraint violation is the sum of the two constraints violation degrees. The feasible optimal solution of g6 is (14.095, 0.84296) and the optimal objective value is -6961.8149. We can see from the infeasible solution (14.03915, 0.72594) that after relaxing two constraints at 0.1295% (the overall violation is 0.1117), we gain 1.9018% improvement in the objective value (the objective value decreases from -6961.8149 to -7,094.2109). If the highest relaxation is 0.17%, the objective value is improved by 2.4972%. In conclusion, at the higher relaxation extent, we can obtain a higher degree of objective improvement.

As can be seen from the above example, it clearly indicates that relaxing the constraint violation at some extent can significantly improve the objective quantity.

Table 6.1 Results of g6 with relaxed constraints

	x_1	x_2	*f(x)*		overall violation	
			objective value	% objective improvement	violation degree	%violation
feasible solution	14.09500	0.84296	-6961.8149	-	0.0000	0.0000
infeasible solutions	14.03915	0.72594	-7094.2109	1.9018	0.1117	0.1295
	14.03863	0.72667	-7093.4224	1.8904	0.1127	0.1275
	14.07684	0.80787	-7001.4263	0.5690	0.0363	0.0377
	14.06692	0.79146	-7020.0762	0.8369	0.0562	0.1067
	14.07900	0.85129	-6993.1219	0.4497	0.0320	0.0937
	14.07603	0.80674	-7002.7129	1.5875	0.03794	0.0475
	14.0690	0.80664	-7003.1743	0.5941	0.0518	0.3065
	14.01431	0.68990	-7135.6621	2.4972	0.1614	0.1700

The above example involves a single objective COP, was demonstrated for clarity of the soft constraint handling concept. However, we are interested in the soft constraint handling for CMOPs. Typically, a mathematical model for a CMOP can be formulated as follows:

$$\min_{\mathbf{x}\in\Re^n} f(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_k(\mathbf{x})]^T \quad (6.1)$$

subject to

$$g_j(\mathbf{x}) \le 0; \qquad j = 1,2,3,...,l \tag{6.2}$$

$$h_j(x) = 0; \qquad j = l+1, l+2,...,m \tag{6.3}$$

$$x_i^L \le x_i \le x_i^U; \quad i = 1,2,3,...,n \tag{6.4}$$

where $$\mathbf{x} = [x_1, x_2, x_3,..., x_n]^T \in \Re^n \tag{6.5}$$

The function $f(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}),..., f_k(\mathbf{x})]^T$ denote the objective vector function to be minimized. **x** is a decision vector of *n* decision variables, where each decision variable x_i is bounded by a lower bound x_i^L and upper bound x_i^U. The search is performed in the decision space. The feasible region is defined by satisfying all constraints (6.2)-(6.3). A solution in the feasible region is called a feasible solution; otherwise it is an infeasible solution. All Pareto-optimal solutions must also be feasible solution.

Constraints can be classified in two types: inequality constraints, (6.2), and equality constraints, (6.3). The number of inequality constraints is *l* while the number of equality constraints is *m-l*. However, the idea of treating all constraints equally and driving the population towards feasibility regardless of the importance level is usually not the best practical choice. In reality a decision-maker is often interested in the solutions that satisfy the hard constraints while marginally violate the soft constraints in order to gain additional benefits in objective quantity. For a highly constrained optimization problem that is difficult to decide which constraints are hard or soft, it will be very helpful to provide decision-makers with analysis on the importance level of each soft constraint and the comprehensive results (solutions) based on different violation level of constraints.

6.2 Previous Works on Constraint Handling Approaches of Multiobjective Optimization Problems

MOEAs were originally designed as unconstrained multiobjective optimizers. Later, when CMOPs has received a lot of attention, methods for constraint handling have been gradually developed. Most of DE algorithms that were proposed to handle constrained optimization problems involved constrained single objective optimization problems. There are fewer number of the proposed MODE algorithms that devoted to solve the CMOPs. The literature review on constraint handling approaches is demonstrated in this Section. The literature survey on MODE for CMOPs is presented as well.

6.2.1 Hard-constraint based approaches for multiobjective optimization

There are various approaches that were proposed to tackle CMOPs. The hard constraint handling techniques can be mainly classified as the following

6.2.1.1 Penalty functions

Penalty functions are the simplest and most commonly used methods for handling constraints using EAs. The principle idea is transforming a constrained problem into an unconstrained one by penalizing the objective function with the constraint violation. The penalty function pushes the solutions toward feasible region. The constraint handling methods based on penalty function can broadly be classified into three categories:

1) The death penalty such as [105-106] is the first research on penalty function. The death penalty methods reject any individuals that violate any constraint and no information is extracted from infeasible individuals. If the added penalty does not depend on the current generation number and remain constant during the entire evolution process, then the penalty function is called static penalty function.

2) The dynamic penalty methods was first proposed by Mu *et al.* [107] use comparison between the infeasibility degree of a solution and the threshold to accept or reject the solution, whereas Hadj-Alouane [108] use the generation number in determining the penalty factor. The dynamic penalty methods do not use the information gathered from the search process to control the amount of penalty.

3) To overcome the problem of dynamic penalty, the adaptive penalty methods are proposed. Tessama and Yen [109] designed their adaptive penalty function based on the number of feasible solutions in the current population to determine the amount of penalty added to infeasible individuals for constrained SOPs. The rank of each individual is determined by the sum of distance (i.e., a normalized variant of fitness) and the penalty. Later, Woldesenbet *et al.* [110] extended the idea to the more complicated multiobjective optimization problems. The related work on this category also proposed by [111-114].

6.2.1.2 Feasibility rules

These methods based on preference of feasible solutions over infeasible solutions using specific rules. Coello Coello [115] used the Pareto non-dominance concept to deal with constrained multi-objective optimization problems. Feasible solutions are always ranked higher than infeasible solutions, and infeasible solutions are ranked by the ascending order of constraint violations. Ray *et al.* [116] suggested a more elaborated constraint-handling technique based on non-domination check of constraint violations. Three different non-dominated rankings of the population are performed, using objective function values, the constraint violation values of all constraints, and a combination of objective functions and constraint-violation values, respectively. Domination check is based on individual comparison between these ranks. Deb *et al*. [117] proposed the NSGA-II for multi-objective optimization problems. To extend its utility to handle the constrained problems, a constraint handling approach is designed and incorporated.

Constrained-dominate relationship between two solutions is defined in which any feasible solution has a better non-domination rank than any infeasible solutions. All feasible solutions are ranked according to their non-domination level based on the objective function values. Among two infeasible solutions, the solution with a smaller constraint violation is assigned a better rank.

6.2.1.3 Separation of objective and constraints

The design principle is to deal with the objectives and constraints separately. Surry and Radcliffe [118] measured the degree of constraint violation for each constraint and treated each of them as an objective in a multi-criterion problem. The approach views a constrained optimization problem alternatively as a constraint satisfaction problem (i.e., ignoring the objective function) and as an unconstrained optimization problem (i.e., ignoring the constraints). A population-based adaptive method is used to decide which view to take. Venkatraman and Yen [119] proposed a generic framework for constrained optimization problems. In the first phase, the objective function is completely disregarded and the search is directed solely toward finding a feasible solution. In the second phase, the simultaneous optimization of the objective function and the satisfaction of the constraints are treated as a bi-objective optimization problem. Yen and Leong [120] transformed the bi-objective constrained optimization problem into an unconstrained tri-objective optimization problem, where the third objective is the overall constraint violation. The rank value and constraint violation are combined to update the personal best of PSO population for the infeasible case. Constraint violation and the feasibility ratio are used to guide the particles towards feasibility first and then influence them to search for global optimal solution.

The common characteristic of current constraint handling methods is that a feasible solution wins over an infeasible one in almost all cases, although some methods keep a certain number of infeasible solutions [121]. Details on constraint handling can be found in the well-documented survey papers [134-136].

6.2.2 Soft handling of constraints-rationality and basic idea

The soft handling concept can be found in many fields. Berrada *et al.* [122] studied the nurse scheduling problem with a multi-objective mathematical programming approach. In the model, administrative and union contract specifications are expressed as hard constraints while the constraints related to days off, the number of consecutive working days, and other specific nurse's wishes in scheduling are formulated as soft constraints. Wang and Fang [123] considered the nondeterministic nature of the business environment of a manufacturing enterprise and described the single-objective production planning by using fuzzy mathematical programming model. The proposed improved genetic algorithm finds a family of inexact solutions within an acceptable level and can find a family of preferred solutions which provide more candidates than the exact approach for choice. A decision maker can select a preferred solution via the human-computer interaction. Fargier and Lamothe dealt with soft constraints in hoist scheduling problems in chemical treatment line electroplating using the fuzzy approach [124]. Infeasibility Driven Evolutionary Algorithm is used in constrained optimization problems to search for optimum solutions near the constraint boundary [125-126]. A small proportion of infeasible solutions is allowed. The original constrained minimization problem with k objectives is reformulated as an unconstrained minimization problem with $k+1$ objectives, where the additional objective is calculated based on the relative amount of constraint violation among the population members. The proposed approach provides a set of marginally infeasible solutions for trade-off studies. Later the algorithm was modified to quantify the amount of constraint violation by ranking the infeasible solutions according to the violation levels [127-128]. The performance of the algorithm with infeasibility consideration is demonstrated through a lot of mathematical and engineering problems.

6.2.3 Constraint handling in constrained multiobjective DE

Even though there are many researches on constrained single objective DE, there are fewer proposed multiobjective DE algorithms to handle CMOPs.

Kukkonen and Lampinen [51] developed the generalized differential evolution version three (GDE3) to solve MOPs with constraints. GDE3 combines the Pareto-based differential evolution with the previous GDE version. If the problem is unconstrained single objective optimization, GDE3 is exactly the same as the original DE. This version uses a growing population and non-dominated sorting as same as NSGA-II [52] to obtain improved diversity and make the algorithm less sensitive to the control parameters. They also studied the effect of control parameters on GDE3 [53] and found that GDE3 is more robust than its previous version. The algorithm performed worse for the rotated multiobjective optimization problems as documented in [54]. Application of GDE3 can also be found in [55].

Zielinski *et al.* [132] extend a single objective DE to handle CMOPs by modifying the dominance principle and the crowding distance of NSGA-II to handle the constraints.

Zhang *et al.* [133] proposed a hybrid of DE and GA algorithm for CMOPs. The search biases strategy is introduced by selection of the current best solution (for mutation in MODE) based on constraint Pareto dominance and crowding distance. Then a hybrid of MODE and GA with the (N+N) framework is given. The offspring will be generated by both MODE and NSGA-II.

Zamuda *et al.* [137] proposed the DE with self-adaptation and local search for CMOPs. The algorithm uses the self-adaptation mechanism from [62] and a sequential quadratic programming local search. The constraint handling is done by controlling the ε level constraint violation and altering the domination principle.

Santana-Quintero *el al.* [129] proposed DEMORS to handle CMOPs by using a two-stage hybrid DE approach. In the first stage they use a multiobjective DE to generate an initial approximation of the Pareto front. Then, in the second stage, rough set theory is used to improve the spread and quality of this initial approximation.

Qu and Suganthan [130] proposed a diversity enhanced constrained multiobjective DE (DE-CMODE) to overcome the premature convergence problem. DE-CMODE combines the current population with a diversified memory based on the crowding DE concept to increase the diversity of the differential vectors and thereby the diversity of the newly generated offspring. Later, they proposed an ensemble of constraint handling methods (ECHM) [131] which integrated with a MODE. Since no single state-of-the-art constraint handling technique can outperform all others on every problem. Therefore, an ensemble of three constraint handling techniques (self-adaptive penalty, superiority of feasible solution, and ε-constraint) is used.

Min-Nan *et al.* [138] proposed a hybrid constraint handling mechanism with DE. They combines the ε-comparison and penalty method together. Each constraint is assigned its own ε-value and is controlled the value by the amount of violation. The penalty method deals with the region where constraint violation exceeds the ε-value and guides the search toward the ε-feasible region. The new mutation strategy for DE was proposed as well.

Liang *et al.* [139] uses the information of infeasible solutions to help the multiobjective DE improve the convergence and diversity of solutions. The proposed method is to ensure that a certain number of good infeasible solutions will be kept in the evolution process to guide the search.

The previous work on multiobjective DE to handle CMOPs is all about hard constraint handling approach. These constraints are strictly to be satisfied without any exception. The proposed constraint handling techniques mainly based on the modified constrained Pareto dominance rule. The performance of the optimizers for constrained optimization largely depends on the mechanism of constraint handling.

Since the constraints that are soft handled can improve some objectives, therefore it will be very useful for decision makers if a set of solutions are provided with good trade-off among objectives and trade-off between objectives and constraint violation. We proposed a soft constraint handling technique which is integrated within AFMDE. Details on the proposed concept are described in Section 6.3.

6.3 Soft Constraint Handling AFMDE

In this Section, the proposed soft constraint handling method is described below.

6.3.1 Definition of constraints violation

To measure the constraint violation, a common scalar value is used. Let G_j be the violation of constraints *j*, then

$$G_j = \begin{cases} \max(0, g_j(x)), & j = 1, 2, ..., l \\ \max(0, |h_j(x)| - \delta), & j = l+1, l+2, ..., m \end{cases} \tag{6.6}$$

6.3.2 Constraints violation degree-based nondominated sorting

Violation degree will be used to bound acceptable infeasible region. To express the degree of violation of a soft constraint, a Gaussian function is used to quantify the satisfaction degree of constraint violation:

$$\mu(G_j) = \begin{cases} 0, & G_j \le \delta_j \\ 1 - e^{-\left(\frac{G_j - \delta_j}{\alpha}\right)}, & G_j \ge \delta_j \end{cases} \tag{6.7}$$

Figure 6.1 shows the satisfaction degree curve of a soft constraint, where G_j is the violation magnitude, $\mu(G_j)$ is the violation degree, δ_j is the threshold (tolerance of constraint violation). The shape of the curve is controlled by a parameter α. Solid line in this figure is the case of hard constraint when α approaches zero. As a result, hard constraint is considered a special case under this soft constraint formulation. The dotted lines correspond to the situations of various constraint relaxations. The degree of violation increases with α along the direction of arrow. Hard constraints must be satisfied by the solid line. Soft constraints can be relaxed within a certain range whose violation degree is showed by the dotted lines.

Let $G_j(x)$ be the violation of constraints *j* (*j* = 1, 2, ..., *m*) of an individual *x*, δ_j be the tolerance value of constraint *j*, $\mu(G_j)$ be the violation degree of constraint *j* on individual *x*.

The overall violation degree of an individual is defined as:

$$\mu = \sum_{j=1}^{m} \mu(G_j) \tag{6.8}$$

Based on relaxation of constraints, the proposed *preference rule strategy* between two individuals is designed as follows:

1) If both individuals are feasible solutions, the one with the better fitness value wins.

2) If one solution is feasible and the other one is infeasible when the overall violation degree of the infeasible solution is less than a predefined tolerance level β, the solution with the better fitness value wins. Otherwise the feasible solution wins.

3) When both solutions are infeasible,

- If both individuals' overall violation degrees are less than β, the one with the better fitness value wins, in the case of non-dominated fitness value the solution with smaller violation wins;
- If the overall violation degree of one solution is less than β, while the other is greater than β, the one with smaller violation value wins;
- If both individuals' overall violation degrees are greater than β, the one with smaller value of violations is preferred, in the case of equal violation degree, the solution with better fitness wins.

AFMDE incorporates the proposed soft constraint handling approach in which both feasible and infeasible solutions are nondominated sorted based on preference rule strategy. The diversity preservation in the soft constraint handling AFMDE is the same as crowding distance sorting in NSGA-II.

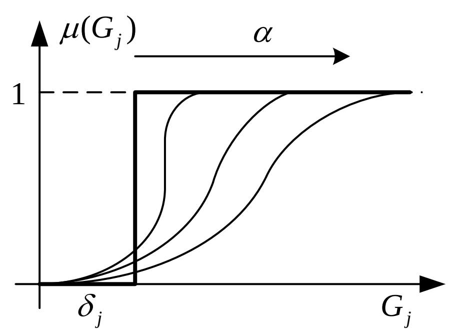
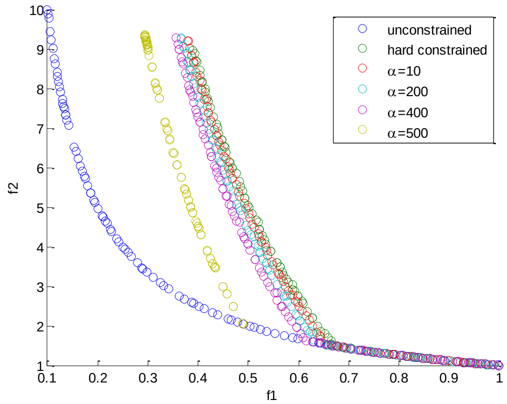


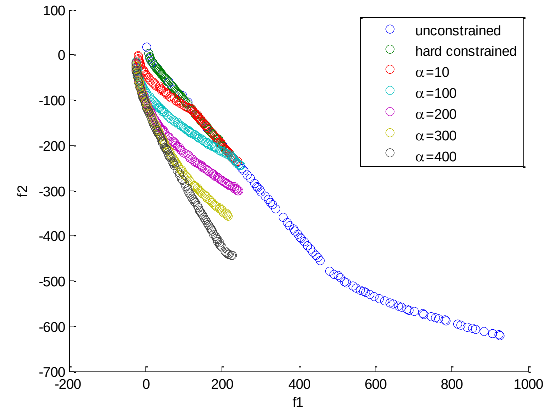
Figure 6.1 Satisfaction degree curve of a constraint

6.4 Experiments and Results for Benchmark Functions

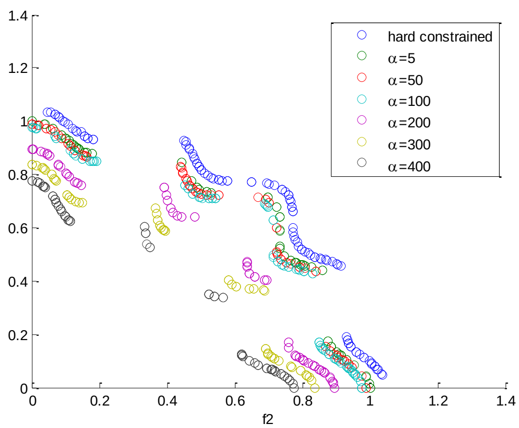
The proposed soft constraint handling approach is integrated with AFMDE to handle CMOPs. The proposed concept is quantified by testing on benchmark functions: CONSTER [101], SRN [102], TNK [103], OSY [104], Welded beam [141] and CTP (CTP1 to CTP8) test suites [1] for 10 independent runs. The parameters for CTP2 to CTP7 are shown in Table 6.2. For each trial, the algorithm will stop if it reaches the predefined maximum number of function evaluation at 300,000. The population size for all test instances is 100. In order to quantify the performance, the hypervolume is computed for each run. We compare the performance between the hard constrained and the soft constrained solutions by utilizing the nonparametric Mann-Whitney-Wilcoxon rank sum test at the significance level of 0.05 by two-tailed test. The *p*-values are shown in Table 6.3. As can be seen, the proposed soft constraint handling approach has shown statistically improvements from hard-constrained approach unless stated. There are only CTP3 results for $\alpha = 50$ and $\alpha = 100$ that the hypervolume distributions are not different from the hard- constrained approach. The example Pareto fronts found for unconstrained, hard constrained, and a number of soft constrained optimization for all test instances are demonstrated in Figure 6.2. It is clearly indicated that when constraints are relaxed within a relatively small value, the obtained Pareto fronts are very close to the hard constraint one. If the constraints are further relaxed, the approximation Pareto fronts are improved consequently. It should be noted that the different α values for each test problems also lies in the consideration that some front will be very close to each other. In the case of the welded beam problem which aims at welding a beam on another beam and must carry a certain load, the violation of the constraints will make the design unacceptable. Therefore, the result obtained by relaxation of any constraints has no specific meaning although some extended solutions have been found as shown in Figure 6.2(e).



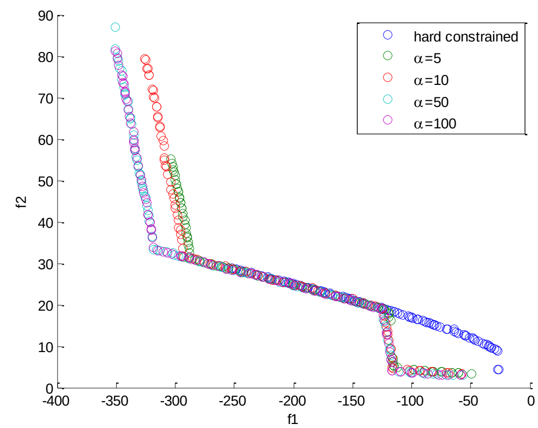
(a) CONSTR



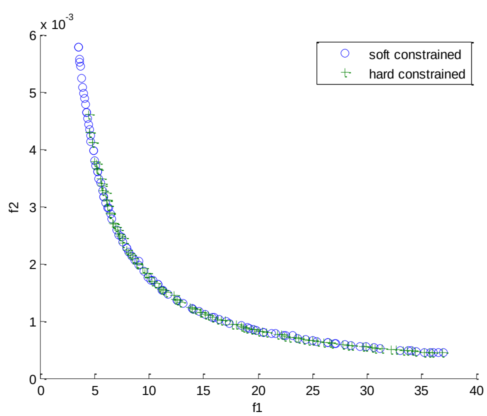
(b) SRN



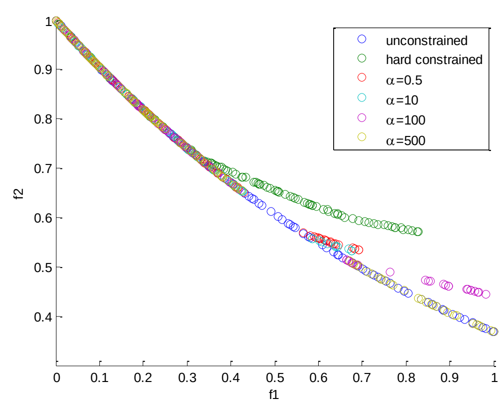
(c) TNK



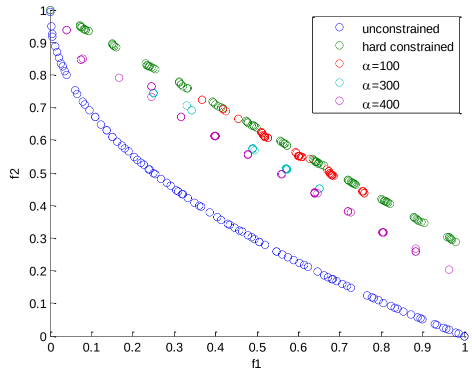
(d) OSY



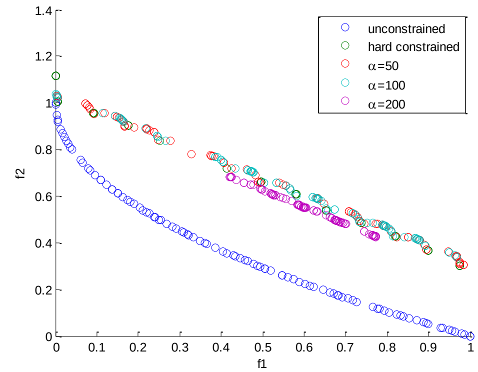
(e) Welded beam



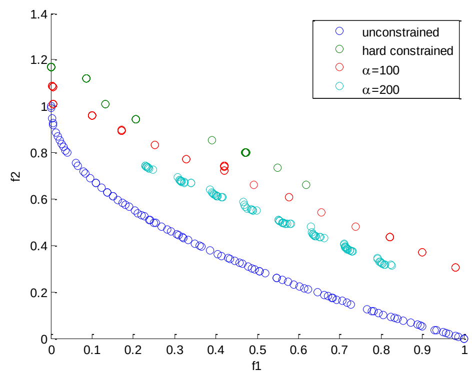
(f) CTP1



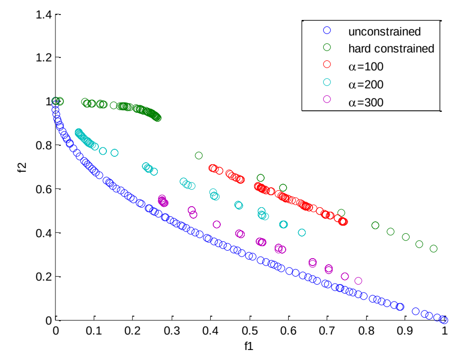
(g) CTP2



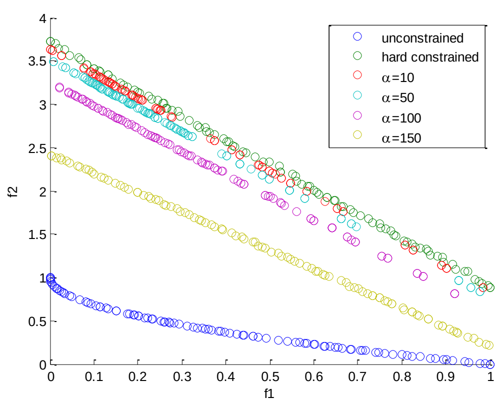
(h) CTP3



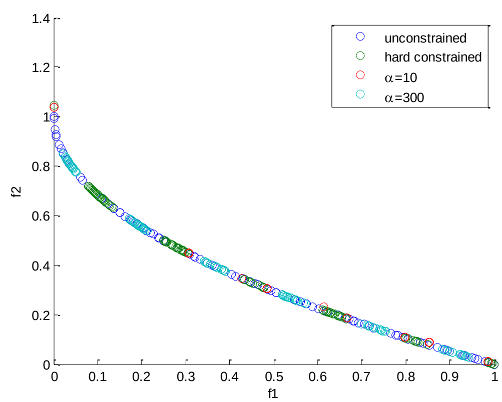
(i) CTP4



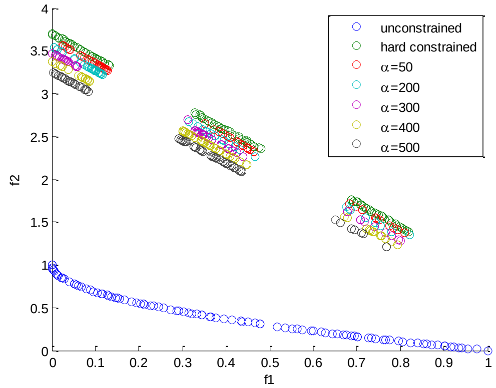
(j) CTP5



(k) CTP6



(l) CTP7



(m) CTP8

Figure 6.2 Examples of obtained Pareto front for (a) CONSTR (b) SRN (c) TNK (d) OSY (e) Welded beam (f) CTP1 (g) CTP2 (h) CTP3 (i) CTP4 (j) CTP5 (k) CTP6 (l) CTP7 (m) CTP8

Table 6.2 Parameters of CTP test suites

	θ	a	b	c	d	e
CTP2	-0.2π	0.2	10	1	6	1
CTP3	-0.2π	0.1	10	1	0.5	1
CTP4	-0.2π	0.75	10	1	0.5	1
CTP5	-0.2π	0.1	10	2	0.5	1
CTP6	0.1π	40	0.5	1	2	-2
CTP7	-0.05π	40	5	1	6	0
CTP8	0.1π -0.05π	40 40	0.5 2	1 1	2 6	-2 0

Table 6.3 The distribution of hypervolume values using Mann-Whitney-Wilcoxon rank sum test

TNK

$\alpha = 5$	$\alpha = 50$	$\alpha = 100$	$\alpha = 200$	$\alpha = 300$	$\alpha = 400$
1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4

SRN

$\alpha = 10$	$\alpha = 100$	$\alpha = 200$	$\alpha = 300$	$\alpha = 400$	$\alpha = 500$
5.8284E-4	1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4

OSY

$\alpha = 5$	$\alpha = 10$	$\alpha = 50$	$\alpha = 200$	$\alpha = 300$	$\alpha = 500$
1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4

CONSTER

$\alpha = 100$	$\alpha = 200$	$\alpha = 400$	$\alpha = 500$
1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4

CTP1

$\alpha = 10$	$\alpha = 100$	$\alpha = 300$	$\alpha = 500$
1.8063E-4	1.8063E-4	1.8063E-4	1.8063E-4

CTP2

$\alpha = 10$	$\alpha = 100$	$\alpha = 300$	$\alpha = 500$
7.9258E-4	2.4905E-4	1.1387E-4	1.2292E-4

CTP3

$\alpha = 50$	$\alpha = 100$	$\alpha = 200$
6.776E-1 (no difference)	7.337E-1 (no difference)	1.8267E-4

CTP4

$\alpha = 100$	$\alpha = 200$
3.2984E-4	3.2984E-4

CTP5

$\alpha = 100$	$\alpha = 200$	$\alpha = 300$
4.3964E-4	7.6854E-4	2.4613E-4

CTP6

$\alpha = 10$	$\alpha = 50$	$\alpha = 100$	$\alpha = 150$
1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4

CTP7

$\alpha = 10$	$\alpha = 300$
1.8063E-4	1.8165E-4

CTP8

$\alpha = 50$	$\alpha = 100$	$\alpha = 200$	$\alpha = 300$	$\alpha = 400$	$\alpha = 500$
1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4	1.8267E-4

6.5 Constrained Non-uniform circular antenna arrays design

Sections 6.3 and 6.4 described the soft constraint handling technique with AFMDE and it was quantified by performing experiments on benchmark CMOPs. In this section, we will prove its practical usage by applying it to a constrained non-uniform circular antenna array design problem as a case study. Non-uniform circular antenna arrays are popular in mobile and wireless communications such as air and space navigation, radar, sonar and other applications [16] [93]. The antenna array provides the higher directive radiation pattern than a single element antenna. The design problem is formulated as a constrained three-objective optimization problem.

6.5.1 Problem formulation

The non-uniform circular antenna array geometry is shown in Figure 6.3.

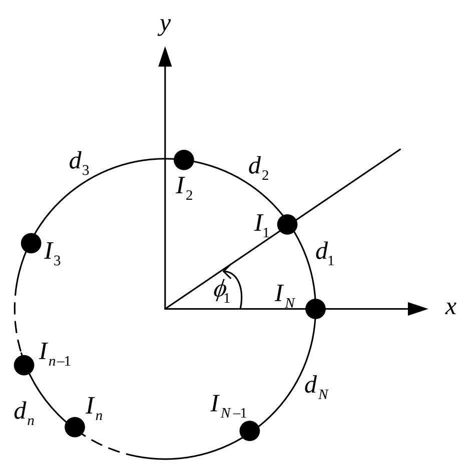


Figure 6.3 A non-uniform circular antenna array on x-y plane

The N isotropic point sources are non-uniformly spaced on a circle (of radius a) lying in the x-y plane. The radiation pattern of this antenna array can be described by its array factor. In the x-y plane, the array factor for the non-uniform circular antenna array can be represented as

$$AF(\phi) = \sum_{n=1}^{N} I_n \exp(j[ka\cos(\phi - \phi_n) + \alpha_n]) \tag{6.9}$$

where I_n = the amplitude excitation of the *n*th element,

ϕ = the azimuth angle measured from the positive x-axis,

ϕ_n = the angular position of the nth element in the x-y plane,

α_n = phase excitation of the *n*th element and can be represented by

$$\alpha_n = -ka\cos(\phi_0 - \phi_n) \tag{6.10}$$

$$ka = \frac{2\pi}{\lambda}a = \sum_{i=1}^{N} d_i \tag{6.11}$$

$$\phi_n = \frac{2\pi}{ka}\sum_{i=1}^{n} d_i \tag{6.12}$$

where *N* is the total number of elements in the circular array, *a* is the radius of the circular. d_i is the distance between two adjacent elements d_n and d_{n-1}. In this design problem, ϕ_0 is chosen to be zero, i.e., the main lobe of the radiation pattern is directed along the x-axis.

The non-uniform circular array synthesis problem is formulated as a CMOP:

f_1: minimizing the maximum side lobe level (*SLLmax*)

f_2: maximizing the directivity (*D*)

f_3: minimizing the first null beamwidth (*FNBW*) amplitudes (*FNBW_dB*)

subject to

$g_1(x) = FNBW \leq 30$

$$g_2(x) = \sum_{i=1}^{N} d_i \le N$$, N is the number of elements

The decision variables are represented as $\mathbf{x} = [d_1, d_2, ..., d_N, I_1, I_2, ..., I_N]$,

where N is the number of elements,

d_i, $i = 1, 2, ..., N$ is the spacing between two adjacent elements d_N and d_{N-1},

I_i, $i = 1, 2, ..., N$ is the amplitude excitation at the *i*th element.

The lower bound of d_i must be 0.5λ to avoid the mutual coupling effect of the array [98].

6.5.2 Experimental setup and results

6.5.2.1 Experimental setup

The proposed soft constraint handling AFMDE is applied to the constrained non-uniform circular antenna array design problem. In this case, the radiation pattern with the main lobe steered to $\phi_0 = 0$. Several experiments are conducted with different number of 12 array elements The population size is 300. The algorithm will stop if it reaches the maximum number of function evaluations at 300,000. The external archive size is 100. The constraint is soft handled with different relaxation degrees. For soft constraint handling experiments, the shape of constraint violation function is controlled by a parameter α. The value of α for both constraints are the same at 100. The tolerance value of overall violation degree is 0.001 in order to guarantee that the obtained solutions are on the boundary of constraints and not far away from the original feasible region. On the other hand, the tolerance value of overall violation degree is zero and α = 10E-10 for the hard constraint handling experiments.

6.5.2.2 Results

The constrained non-uniform circular design is tackled by the proposed soft constraint handling-based AFMDE. The results are shown in Table 6.4. The extreme solutions in the first, second, and the third objective dimension for both hard and soft constraint handling are selected as representatives of the optimal set. The hard constraint handling extreme solutions are shown in the first row which are *a*, *b*, and *c* in the first, second, and third objectives, respectively. The extreme solutions for soft constraint handling are *d*, *e*, and *f* in the first, second, and third objectives, respectively.

Firstly, we compare the extreme solutions of the hard and soft constraints handling in each objective dimensions. In the case of the first objective (minimizing *SLLmax*), we compare solution *a* with solution *d*. When relaxing both constraints, objective value f1 decreases from -7.8267 to -11.4178 which is 45.88% improvement at the cost of 0.0382 violation degree. In the case of second objective (maximizing *D*), we compare the extreme solutions in f2 dimension between solution *b* and *e*. The results show that there is objective improvement in all dimensions at the expense of violation degree at 0.0010, which are 19.08% (from -6.5293 to -7.7748), 3.69% (from 15.4542 to 16.0240), and 10.8% (from -62.8200 to -69.5852) improvement in the first, second, and third objectives, respectively. Solution *c* and *f* are compared in the case of the third objective (minimizing *FNBW* amplitude). In this case, as a result, the f1 and f2 is improved but not f3 values. f1 and f2 are improved by 9.24% (from -5.929 to -9.6939) and 2.41% (from 14.7108 to 15.0657), respectively by the constraint violation degree is 0.188. To sum up, the comparison between the extreme solutions from the optimal solution set of hard and soft constraint handling methods prove that the soft constraints handling technique have gained improvement on one or more objectives. However, only comparisons between extreme solutions are not enough to quantify the soft constraint handling method.

In order to quantify the efficiency of the proposed soft constraint handling approach for the non-uniform circular array design which is a CMOP, hypervolume is computed for hard constraint and soft constraint handling approaches. The last column of Table 6.4 shows the hypervolume of the optimal results for both hard and soft constraint handling cases. It is clearly indicated that the hypervolume increased from the hard constraint handling (2.7349E+6) to the soft constraint handling (2.7791E+6) by 1.612%.

The rectangular plots of normalized radiation pattern in dB for the extreme solutions for hard and soft constraint handling approaches are shown in Figure 6.4. whereas the polar plots for the extreme solutions for hard and soft constraint handling approaches are shown in Figure 6.5. As can be seen from Figure 6.5, the soft constraint results clearly show smaller side lobes level than hard constraint handling results.

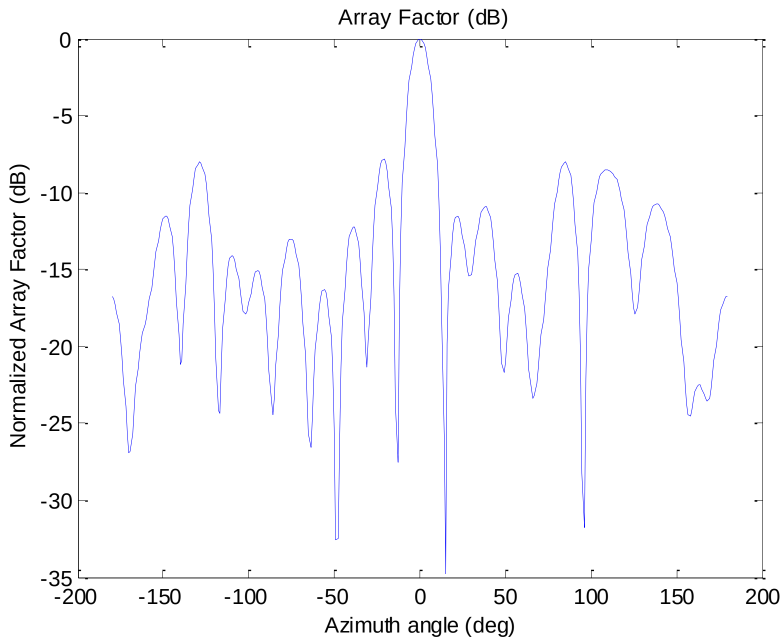
From the results, we can conclude that solutions obtained by soft constraint handling approach improve one or more objectives and also increase the number of optimal solutions found. The soft constraint handling technique provides options for decision makers. They can choose a solution based upon their requirements. If they need higher improvement in some objective dimension, they may choose a solution with acceptable degree of constraint violation. In addition, the proposed soft constraint handling approach provide flexibility in terms of relaxing each constraint differently by setting the threshold values, the shape of relaxation function and tolerance level.

6.6 Remarks

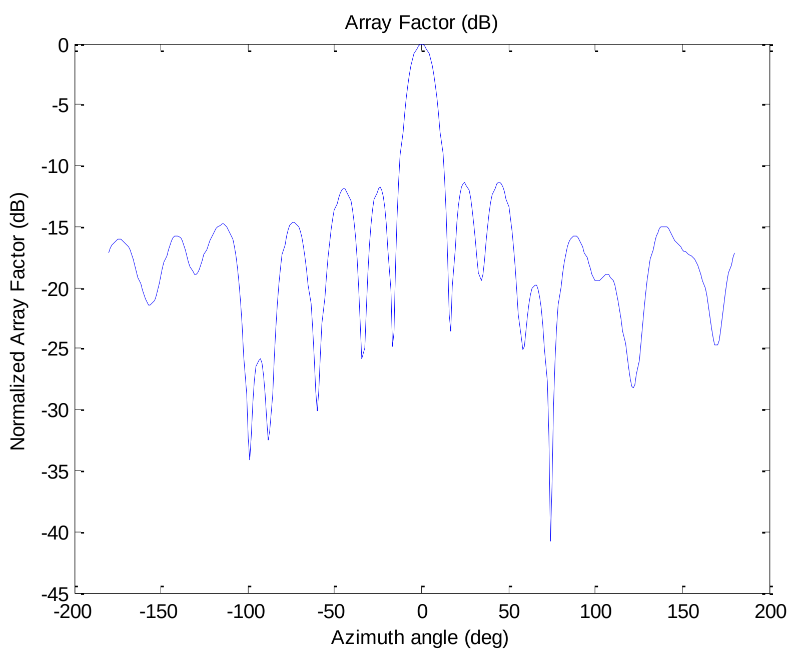
The soft constraint handling approach for CMOPs is presented in this chapter. The constraint violation degree is evaluated by the Gaussian function while the convergence and diversity is preserved by the preference rule strategy. The searching region is extended in order to obtain useful solutions from the infeasible boundary. AFMDE which incorporated with the soft constraint handling technique on CMOP benchmark functions shows the statistic improvement from the hard constraint handling method. In order to prove the practical usage of the soft constraint handling method, AFMDE integrates with the soft constraint handling is applied to the non-uniform circular antenna array design problem. The results demonstrate the relaxations of the constraints with acceptable degrees improve both objective values and the number of optimal solutions found. These provide options for the end users (e.g. decision makers, manufacturers) to make a decision based upon higher-level information and requirements.

Table 6.4 Obtained Pareto solution examples of 12 elements non-uniform circular array design

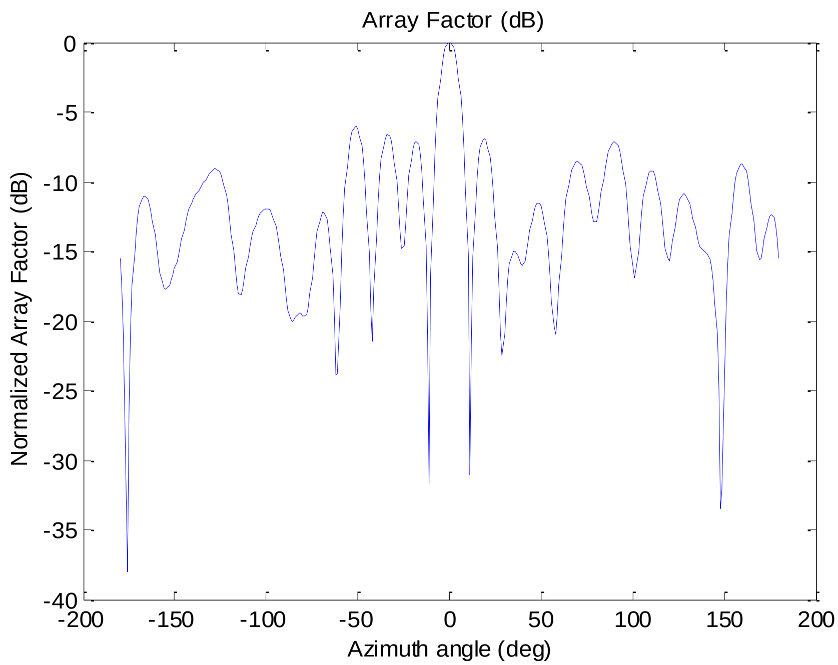
	solutions	f1: *SLLmax*	f2: *D*	f3: *FNBW_dB*	Overall violation	Hypervolume
Hard-constraint Pareto solutions	*a*	-7.8267	15.3618	-63.9410	0	2.7349E+6
	b	-6.5293	15.4542	-62.8022	0	
	c	-5.929	14.7108	-114.4406	0	
Soft-constraint Pareto solutions	*d*	-11.4178 (45.88%)	14.7873 (-)	-59.5044 (-)	0.0382	2.7791E+6 (1.612%)
	e	-7.7748 (19.08%)	16.0240 (3.69%)	-69.5852 (10.8%)	0.0010	
	f	-9.6939 (63.5%)	15.0657 (2.41%)	-105.3999 (-)	0.0188	



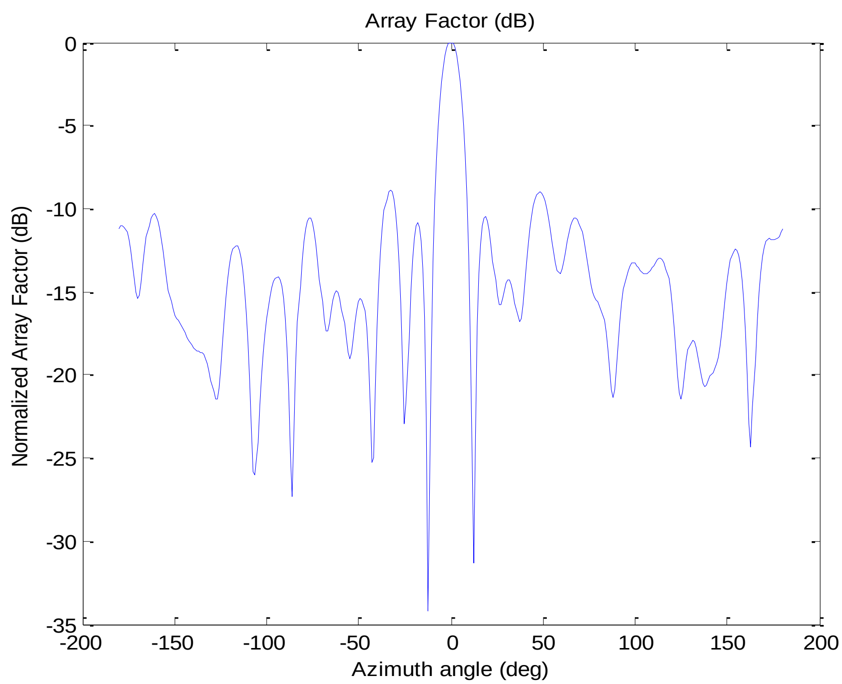
(a) solution *a*



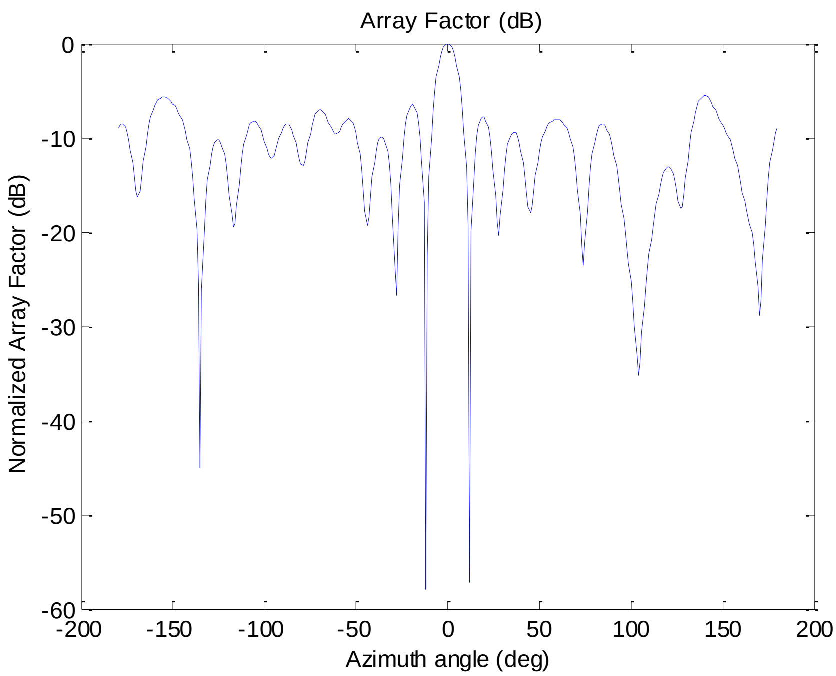
(b) solution *d*



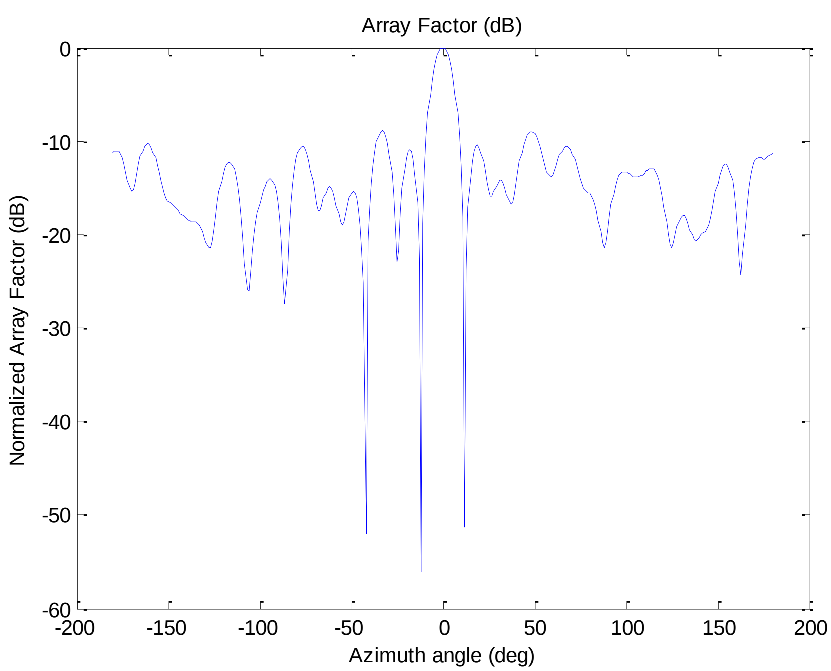
(c) solution *b*



(d) solution *e*

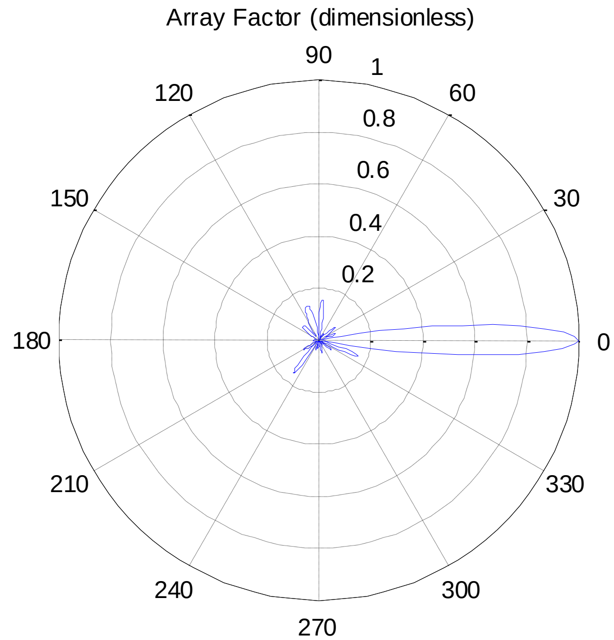


(e) solution *c*

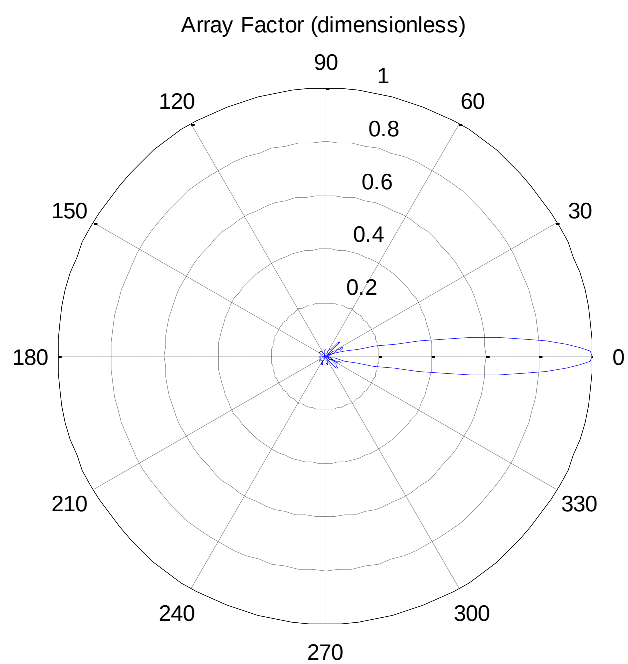


(f) solution *f*

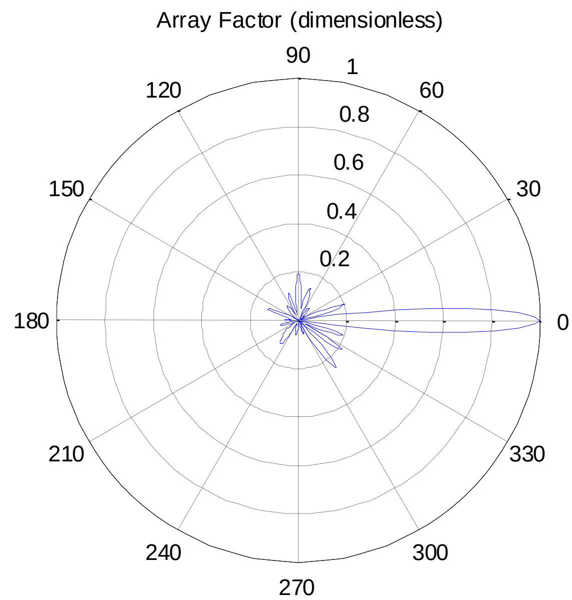
Figure 6.4 The normalized radiation pattern in rectangular plot for the extreme solutions (a) solution *a* (b) solution *d* (c) solution *b*, (d) solution *e* (e) solution *c* (f) solution *f*



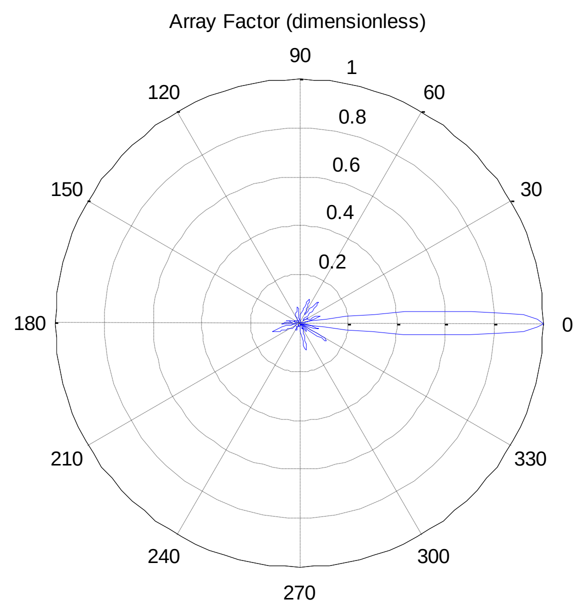
(a) solution a



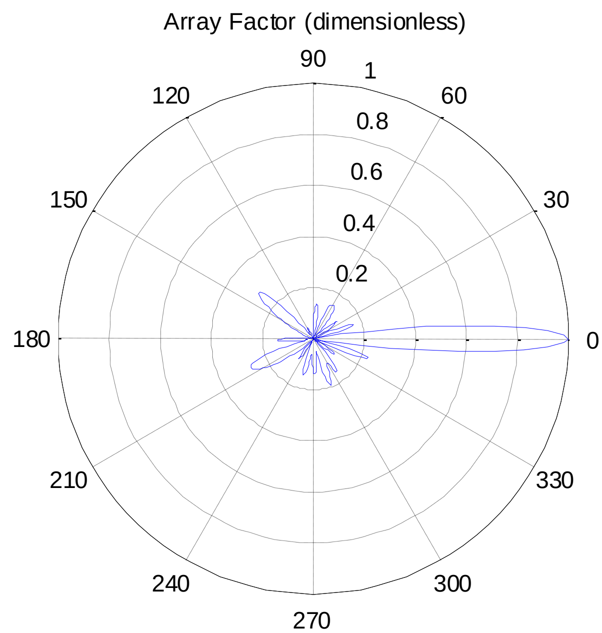
(b) solution *d*



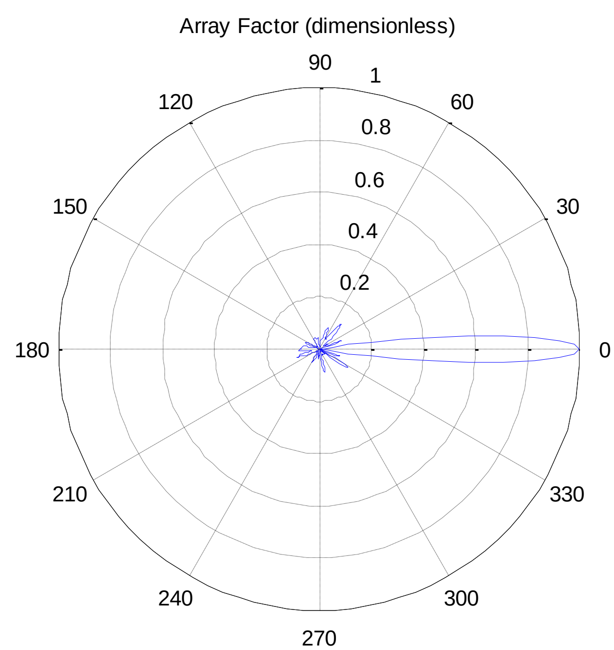
(c) solution *b*



(d) solution *e*



(e) solution *c*



(f) solution *f*

Figure 6.5 The normalized radiation pattern in polar plot for solutions (a) solution *a* (b) solution *d* (c) solution *b*, and soft constrained solutions (d) solution *e* (e) solution *c* (f) solution *f*

CHAPTER VII

CONCLUSION AND FUTURE WORK

Differential evolution (DE) is one of the population-based optimization algorithms. DE has powerful performance and is able to handle multiple types of optimization problems. DE is also easier to be implemented than other evolutionary algorithm representatives and has few control parameters. Real-world optimization problems are mostly multiobjective in nature. Many of them are subjected to a number of constraints. Thus, DE as one of a powerful optimizer and its advantages has motivated this study. In this dissertation, the research goals are the development of a multiobjective differential evolution algorithm to solve the unconstrained and soft-constrained handling multiobjective optimization problems. In addition, the developed algorithm is applied to the practical engineering problems, i.e., antenna design problem.

7.1 Multiobjective Differential Evolution based on Fuzzy Performance Feedback

The delicate balance between exploration and exploitation ability throughout the search process is a key ingredient to any evolutionary algorithms. Differential evolution is one of the evolutionary algorithm representatives. It follows the principle of collective differential intelligence. The mutation strategy and the associated control parameters play important roles to differential evolution performance. How to maintain a good trade-off in between exploration and exploitation and adaptively adjust the control parameters throughout the evolution process have inspired the design of a multiobjective differential evolution in this study.

The design principle of the proposed fuzzy-based multiobjective differential evolution (FMDE) is by adaptively adjusting the associated control parameters of a specific mutation strategy; FMDE can dynamically balance the exploration and exploitation abilities of the population. During the search process, the true Pareto front is unknown, therefore, the performance metrics specifically hypervolume, spacing, and maximum spread are used to monitor the state of evolution process. Hypervolume estimates the convergence status, spacing measures the diversity status, while maximum spread measures the extensiveness of the population. These three performance metrics feed to the fuzzy inference rules derived knowledge from domain experts and published literature. The fuzzy inference rules then dynamically adapt the greedy and distribution parameters of a DE-based mutation strategy over the course of evolution. The effect of dynamically adjust these parameters is balancing the exploitation or exploration ability throughout the search process. A comparative study of FMDE with the chosen state-of-the-art MOEAs on benchmark problems shows that FMDE performs statistically better than those chosen algorithms. The advanced version of FMDE (AFMDE) is proposed to dynamically adjust another control parameter, i.e., crossover rate, which has a direct impact on the performance of MODE. AFMDE performance is competitive with FMDE, yet providing flexibility to better regulate the mutation strategies on some more complicated problems.

Moreover, AFMDE is applied to a 5 by 5 microstrip antenna array for 12.5 GHz broadcasting satellite service synthesis. Three design goals, i.e., maximizing antenna gain, while minimizing side lobe level and reflection coefficient, are optimized simultaneously, Since objective evaluations are computationally expensive, therefore, a radial basis function neural network is developed as a surrogate model for objective evaluation. The results show that AFMDE finds a set of Pareto optimal designs specifically in terms of side lobe level and reflection coefficient. It provides a set of Pareto optimal solutions, not only including those extreme solutions but compromised solutions among the objectives. These solutions offer numerous Pareto optimal choices for the reconfigurable antenna arrays under various real-world complications.

In future work, study on the impact of setting of initial values and ranges of parameters of the proposed algorithms are desired. In addition, research on additional performance measurements to obtain additional evolutionary information from the population in order to continuously improve diversity mechanism is interesting. For the real-world engineering applications, the study on development of a surrogate model, given very limited data samples, as objective evaluation combined with FMDE for optimization is interesting as well.

7.2 Soft Constraint Handling

The main challenges for constrained optimization are optimizing the objective functions and simultaneously handling constraints. Some constraints can be violated within an acceptable degree without compromising the objectives. In real-world applications, some feasible solutions from the hard constraint handling may be impossible or difficult to be implemented. Therefore, if the constraints are soft handled, it may provide a set of optimal solutions within an acceptable range of constraint violation degree for the decision makers to choose from. This practical needs to soft constraint handling principle has motivated the design of soft constraint approach in this thesis. The proposed soft constraint handling is based upon the concept that the constraints can be violated within acceptable degrees to extend the searching region in order to obtain useful solutions from the infeasible side of feasibility boundary. As a result, the higher quality solutions, i.e., objective improvement, can be found. The violation degree can be quantified by the Gaussian function or others. Elites and diversity are preserved by the preference rule strategy and the crowding distance, respectively. The preference rule strategy, based on relaxation degrees and tolerance level of the constraints, is actually non-domination sorting the feasible and infeasible individuals simultaneously. In addition, the search region is extended by this preference rule strategy because infeasible solutions can be ranked higher than feasible solutions if their objective values are better within acceptable violation degrees. AFMDE is integrated with the proposed soft constraint handling approach to solve the constrained multiobjective benchmark problems. The same soft constraint handling approach can also be integrated within other MOEAs. The results show the soft constraint handling can achieve significant objective improvement at the cost of acceptable degree of constraint violations compared to the hard constraint approach.

AFMDE incorporated with the soft constraint handling is applied to the non-uniform antenna array design problem in order to prove the concept that if the constraints are relaxed at some extent, it can improve the quality of the solutions found. The non-uniform circular array design is formulated as three objectives optimization problem subjected to two constraints. The objectives are minimizing the side lobe level and the first null beamwitdth amplitude while maximizing the directivity of the array subject to the specific first null beamwidth and size of the array. Both constraints are treated as soft ones. The results show that if the relaxation degree is higher, the objective improvements on infeasible solutions found at an acceptable degree of constraint violation can be much better.

In the future work, the parameter sensitivity of the soft constraint handling is desired. The soft constraint handling technique may be extended to solve more complex real-world optimization problems, i.e., the dynamic multiobjective optimization problems (DMOPs) such as scheduling in manufacturing plant, dynamic resource management, etc. In DMOPs, the objective functions change over time. As a consequence, the optimal Pareto front changes over time as well. Therefore, the challenge is that optimization algorithms need to track the environmental change and be able find the moving Pareto optimal front and Pareto optimal set. The mathematical model for DMOP can be defined as

$$\min_{\mathbf{x}\in\Re^n} \mathbf{f}(\mathbf{x},t) = \{ f_1(\mathbf{x},t), f_2(\mathbf{x},t), \ldots, f_k(\mathbf{x},t)\} \tag{7.1}$$

subject to

$$g_j(\mathbf{x},t) \le 0; \qquad j = 1,2,3,...,l \tag{7.2}$$

$$h_j(\mathbf{x},t) = 0; \qquad j = l+1, l+2,...,m \tag{7.3}$$

$$x_i^L \le x_i \le x_i^U; \qquad i = 1,2,3,...,n \tag{7.4}$$

where $$\mathbf{x} = [x_1, x_2, x_3,..., x_n]^T \in \Re^n \tag{7.5}$$

Function $\mathbf{f}(\mathbf{x},t)$ denotes a set of objectives to be minimized with respect to time *t*, **x** is a decision vector of *n* decision variables, where each decision variable x_i is bounded by a lower bound x_i^L and upper bound x_i^U. It is not only the objectives that change over time, but also the inequality (7.2) and equality (7.3) constraints change over time as well.

REFERENCES

[1] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*: Wiley, 2009.

[2] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen, *Evolutionary Algorithms for Solving Multi-objective Problems*, 2 ed.: Springer US, 2007.

[3] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution A Practical Approach to Global Optimization*: Springer, 2005.

[4] A. Abraham, *Evolutionary Computation*. London: John Wiley and Sons Ltd., 2005.

[5] H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor: The University of Michigan Press, 1975.

[6] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence through Simulated Evoluiton*. New York: John Wiley, 1966.

[7] I. Rechenberg, "Evoluitonsstrategie: optimierung technischer systeme nach prinzipien der biologischen evoluiton," Fromman-Holzboog, Stuttgart, 1973.

[8] H. P. Schwefel, "Numerische optimierung von computer-modellen mittels der evolutionsstrategie," Birkhaeuser, Basel, 1977.

[9] D. Dasgupta, "An overview of artificial immune systems and their applications," *Artificial Immune Systems and Their Applications*, D. Dasgupta, Ed., ed: Springer Berlin Heidelberg, 1999, pp. 3-21.

[10] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, pp. 3902-3933, 2005.

[11] Z. Kobti, R. G. Reynodls, and T. Kohler, "A multi-agent simulation using cultural algorithms: the effect of culture on the resilience of social systems," in *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on*, 2003, pp. 1988-1995 Vol.3.

[12] R. Storn and K. Price, "Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces", March, 1995 [online]. Available: http://www1.icsi.berkeley.edu/~storn/TR-95-012.pdf.

[13] R. Storn, "On the usage of differential evolution for function optimization," *Fuzzy Information Processing Society NAFIPS Biennial Conference of the North American*, 1996, pp. 519-523.

[14] R. Storn and K. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization,* vol. 11, pp. 341-359, 1997.

[15] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary Computation,* vol. 8, pp. 173-195, 2000.

[16] C. A. Balanis, *Antenna Theory: Analysis and Design*: Wiley-Interscience, 2005.

[17] R. Garg, *Microstrip Antenna Design Handbook*: Artech House, 2001.

[18] R. Waterhouse, *Microstrip Patch Antennas: A Designer's Guide,* Springer, 2003.

[19] V. Feoktistiov. (2006). *Differential Evolution: In Search of Solutions*.

[20] K. P. Wong and Z. Y. Dong, "Differential evolution, an alternative approach to evolutionary algorithm," in *Proc. of Intelligent Systems Application to Power Systems (ISAP),* 2005, pp. 73-83.

[21] S. Rahnamayan, "Opposition-based differential evolution," Doctor of Philosophy, Systems Design Engineering, University of Waterloo, Waterloo, Ontario, Canada, 2007.

[22] A. Nobakhti and H. Wang, "A simple self-adaptive differential evolution algorithm with application on the ALSTOM gasifier," *Apply Soft Computing,* vol. 8, no. 1, pp. 350-370, 2008.

[23] A. Biwas, S. Das, A. Abraham, and S. Dasgupta, "Design of fractional-order PIλDμ controllers with an improved differential evolution," *Engineering Applications of Artificial Intelligence,* vol. 22, pp. 343-350, 2009.

[24] S. Aydin and H. Temeltas, "Fuzzy-differential evolution algorithm for planning time-optimal trajectories of a unicycle mobile robot on a predefined path," *Advanced Robotics,* vol. 18, pp. 725-748, 2004.

[25] J. Chakraborty, A. Konar, L. C. Jain, and U. K. Chakraborty, "Cooperative multi-robot path planning using differential evolution," *Journal of Intelligent and Fuzzy Systems,* vol. 20, pp. 13-27, 2009.

[26] C. Cheng-Hung, L. Cheng-Jian, and L. Chin-Teng, "Nonlinear system control using adaptive neural fuzzy networks based on a modified differential evolution," *IEEE Trans. on Systems, Man, and Cybernetics, Part C: Applications and Reviews,* , vol. 39, pp. 459-473, 2009.

[27] L. Yung-Chien, H. Kao-Shing, and F. Wang, “Plant scheduling and planning using mixed-integer hybrid differential evolution with multiplier updating,” in *Proc. of IEEE Congress on Evolutionary Computation,* vol.1, 2000, pp. 593-600

[28] M. F. Tasgetiren, P. N. Suganthan, and Q.-K. Pan, “An ensemble of discrete differential evolution algorithms for solving the generalized traveling salesman problem,” *Applied Mathematics and Computation 215,* pp. 3356-3368, 2010.

[29] R. Angira and B. V. Babu, “Optimization of process synthesis and design problems: A modified differential evolution approach,” *Chemical Engineering Science,* vol. 61, pp. 4707-4721, 2006.

[30] B. V. Babu and K. K. N. Sastry, “Estimation of heat transfer parameters in a trickle-bed reactor using differential evolution and orthogonal collocation,” *Computers & Chemical Engineering,* vol. 23, pp. 327-339, 1999.

[31] N. Noman and H. Iba, “Inferring gene regulatory networks using differential evolution with local search heuristics,” *IEEE/ACM Trans. on Computational Biology and Bioinformatics,* vol. 4, no. 4, pp. 634-647, 2007.

[32] R. Xu, G. K. Venayagamoorthy, and D. C. Wunsch Ii, “Modeling of gene regulatory networks with hybrid differential evolution and particle swarm optimization,” *Neural Networks,* vol. 20, pp. 917-927, 2007.

[33] R. Bitello and H. S. Lopes, “A differential evolution approach for protein folding,” *IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology, 2006.* 2006, pp. 1-5.

[34] N. Chauhan, V. Ravi, and D. Karthik Chandra, “Differential evolution trained wavelet neural networks: Application to bankruptcy prediction in banks,” *Expert Systems with Applications,* vol. 36, pp. 7659-7665, 2009.

[35] G. D. Magoulas, V. P. Plagianakos, and M. N. Vrahatis, “Neural network-based colonoscopic diagnosis using on-line learning and differential evolution,” *Applied Soft Computing,* vol. 4, pp. 369-379, 2004.

[36] A. Qing, “Dynamic differential evolution strategy and applications in electromagnetic inverse scattering problems,” *IEEE Trans. on Geoscience and Remote Sensing,* vol. 44, pp. 116-125, January 2006.

[37] C. Yikai, Y. Shiwen, and N. Zaiping, “The application of a modified differential evolution strategy to some array pattern synthesis problems,” *IEEE Trans on Antennas and Propagation,* vol. 56, pp. 1919-1927, 2008.

[38] S. K. Goudos and J. N. Sahalos, “Pareto optimal microwave filter design using multiobjective differential evolution,” *IEEE Trans. on Antennas and Propagation,* vol. 58, pp. 132-144, 2010.

[39] S. Das, A. Abraham, and A. Konar, “Automatic clustering using an improved differential evolution algorithm,” *IEEE Trans. on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 38, pp. 218-237, 2008.

[40] V. Aslantas, “An optimal robust digital image watermarking based on SVD using differential evolution algorithm,” *Optics Communications,* vol. 282, pp. 769-777, 2009.

[41] U. K. Chakraborty, *Advances in DE*: Springer, 2008.

[42] R. Mallipeddi and P. N. Suganthan, “Differential evolution algorithm with ensemble of populations for global numerical optimization,” *OPSEARCH,* vol. 46, pp. 184-213, 2009.

[43] H.-Y. Fan and J. Lampinen, “A trigonometric mutation operation to differential evolution,” *Journal of Global Optimization,* vol. 27, pp. 105-129, 2003.

[44] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, “Differential evolution using a neighborhood-based mutation operator,” *IEEE Trans. on Evolutionary Computation,* vol. 13, pp. 526-553, 2009.

[45] G. C. Onwubolu and B. V. Babu, *New Optimization Techniques in Engineering,* 1 Ed. vol. 141: Springer-Verlag Berlin Heidelberg, 2004.

[46] W. F. Leong, “Multiobjective Particle Swarm Optimization: Integration of Dynamic Population and Multiple-Swarm Concepts and Constraint Handling,” Doctor of Philosophy, School of Electrical and Computer Engineering, Oklahoma State University, 2008.

[47] H. A. Abbass, R. Sarker, and C. Newton, “PDE: a Pareto-frontier differential evolution approach for multi-objective optimization problems,” in *Proc. of IEEE Congress on Evolutionary Computation,* 2001, pp. 971-978 vol. 2.

[48] H. A. Abbass, “The self-adaptive Pareto differential evolution algorithm,” in *Proc. of IEEE Congress on Evolutionary Computation,* 2002, pp. 831-836.

[49] F. Xue, A. C. Sanderson, and R. J. Graves, “Pareto-based multi-objective differential evolution,” in *Proc. of IEEE Congress on Evolutionary Computation,*, 2003, pp. 862-869, vol. 2.

[50] F. Xue, A. C. Sanderson, and R. J. Graves, “Multi-objective differential evolution and its application to enterprise planning,” in *Procs of IEEE International Conference on Robotics and Automation,* 2003, pp. 3535-3541 vol.3.

[51] S. Kukkonen and J. Lampinen, “GDE3: The third evolution step of generalized differential evolution,” in *Proc. of IEEE Congress on Evolutionary Computation*, 2005.

[52] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. on Evolutionary Computation*, vol. 6, pp. 182-197, 2002.

[53] S. Kukkonen and J. Lampinen, “An empirical study of control parameters for the third version of generalized differential evolution (GDE3),” *IEEE Congress on Evolutionary Computation*, 2006, pp. 2002-2009.

[54] S. Kukkonen and J. Lampinen, “Performance assessment of generalized differential evolution 3 (GDE3) with a given set of problems,” *IEEE Congress on Evolutionary Computation*, 2007, pp. 3596-3600.

[55] S. Kukkonen, S. R. Jangam, and N. Chakraborti, “Solving the molecular sequence alignment problem with generalized differential evolution 3 (GDE3),” in *Procs. of IEEE Symposium on Computational Intelligence in Multicriteria Decision Making (CMDM)*, 2007, pp. 302-309 .

[56] T. Robic and B. Fillipic, “DEMO: Differential evolution for multiobjective optimization,” in *Procs of the Third International Conference on Evolutionary Multi-Criterion Optimization-EMO*, 2005, pp. 520-533.

[57] A. W. Iorio and X. Li, “Solving rotated multi-objective optimization problems using differential evolution,” in *AI 2004: Advances in Artificial Intelligence:* vol. 3339, G. I. Webb and X. Yu, Eds., ed Heidelberg: Springer, 2004, pp. 861-872.

[58] Z. Qingfu and L. Hui, “MOEA/D: A Multiobjective evolutionary algorithm based on decomposition,” *Evolutionary Computation, IEEE Transactions on*, vol. 11, pp. 712-731, 2007.

[59] V. L. Huang, A. K. Qin, P. N. Suganthan, and M. F. Tasgetiren, “Multi-objective optimization based on self-adaptive differential evolution algorithm,” in *Procs. of IEEE Congress on Evolutionary Computation*, 2007, pp. 3601-3608.

[60] A. K. Qin, V. L. Huang, and P. N. Suganthan, “Differential evolution algorithm with strategy adaptation for global numerical optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 13, pp. 398-417, April 2009.

[61] V. L. Huang, S. Z. Zhao, R. Mallipeddi, and P. N. Suganthan, “Multi-objective optimization using self-adaptive differential evolution algorithm,” in *Proceedings of IEEE Congress on Evolutionary Computation* 2009, pp. 190-194.

[62] A. Zamuda, J. Brest, B. Boskovic, and V. Zumer, “Differential evolution for multiobjective optimization with self-adaptation,” in *Proceedings of IEEE Congress on Evolutionary Computation*, 2007, pp. 3617-3624.

[63] J. Zhang and A. C. Sanderson, “Self-adaptive multi-objective differential evolution with direction information provided by archived inferior solutions,” in *Proceedings of IEEE Congress on Evolutionary Computation*, 2008, pp. 2801-2810.

[64] D. Ning and W. Yuping, "Multiobjective differential evolution based on opposite operation," in *International Conference on Computational Intelligence and Security*, 2009, pp. 123-127.

[65] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Trans. on Evolutionary Computation,* vol. 12, pp. 64-79, Febuary 2008.

[66] J. Bader and E. Zitzler, "Hype: An algorithm for fast hypervolume-based many-objective optimization," *Evol. Comput.,* vol. 19, pp. 45-76, 2011.

[67] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Parallel Problem Solving from Nature - PPSN VIII*. vol. 3242, X. Yao, E. Burke, J. Lozano, J. Smith, J. Merelo-Guervós, J. Bullinaria, J. Rowe, P. Tiňo, A. Kabán, and H.-P. Schwefel, Eds., ed: Springer Berlin Heidelberg, 2004, pp. 832-842.

[68] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research,* vol. 181, pp. 1653-1669, 2007.

[69] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: improving the strength Pareto evolutionary algorithm," Swiss Federal Institute of Technology, Zurich, Germany2001.

[70] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II," *IEEE Trans. on Evolutionary Computation,* vol. 13, pp. 284-302, April 2009.

[71] R. Joshi and A. C. Sanderson, "Minimal representation multisensor fusion using differential evolution," *IEEE Trans. on Systems, Man and Cybernetics, Part A: Systems and Humans,* vol. 29, pp. 63-76, 1999.

[72] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Trans. on Evolutionary Computation,* vol. 15, pp. 55-66, February 2011.

[73] D. Zaharie, "Influence of crossover on the behavior of differential evolution algorithms," *Appl. Soft Comput.,* vol. 9, pp. 1126-1138, 2009.

[74] K. Deb and S. Jain, "Running performance metrics for evolutionary multi-objective optimization," *KanGal Technical Report No. 2002004*, 2002.

[75] D. V. Veldhuizen, "Multiobjective evolutionary algorithms: classifications, analyses, and new innovations," PhD Thesis, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 1999.

[76] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler, "Theory of the hypervolume indicator: optimal μ-distributions and the choice of the reference point," in *Proc. of ACM*

SIGEVO Workshop on Foundations of Genetic Algorithms, Orlando, FL, 2009, pp. 87-102.

[77] W. Rui, R. C. Purshouse, and P. J. Fleming, "Preference-inspired coevolutionary algorithms for many-objective optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 17, pp. 474-494, 2013.

[78] J. R. Schott, "Fault tolerant design using single and multicriteria genetic algorithm optimization," MS Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge Massachusetts, 1995.

[79] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," *Proceedings of the 2002 Congress on Evolutionary Computation*, 2002, pp. 825-830.

[80] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Trans on Evolutionary Computation*, vol. 10, pp. 477-506, 2006.

[81] C. A. C. Coello and N. C. Cort´es, "Solving multiobjective optimization problems using an artificial immune system," *Genetic Programming and Evolvable Machines*, vol. 6, pp. 163-190, 2005.

[82] J. J. Durillo and A. J. Nebro, "jMetal: A Java framework for multi-objective optimization," *Advances in Engineering Software*, vol. 42, pp. 760-771, 2011.

[83] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adaping control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Trans. on Evolutionary Computation*, vol. 10, pp. 646-657, 2006.

[84] R. Mallipeddi and P. N. Suganthan, "Ensemble differential evolution algorithm for CEC2011 problems," in *IEEE Congress on Evolutionary Computation*, 2011, pp. 1557-1564.

[85] Z. Jingqiao and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Trans on Evolutionary Computation*, vol. 13, pp. 945-958, 2009.

[86] Z. Bayraktar, M. Komurcu, J. A. Bossard, and D. H. Werner, "The wind driven optimization technique and its application in electromagnetics," , *IEEE Transactions on, Antennas and Propagation*vol. 61, pp. 2745-2757, 2013.

[87] J. Guo and J. Li, "Pattern synthesis for probe-fed microstrip conformal antenna array," in *Microwave Conference, 2009. APMC 2009. Asia Pacific*, 2009, pp. 2002-2005.

[88] X. Zheng, C. Liao, L. Chang, J. Fang, and H.-j. Fu, "Automated design of an ultra-wideband microstrip patch antenna," in *International Conference on Microwave Technology and Computational Electromagnetics*, 2009, pp. 80-83.

[89] M. Spasos, R. Nilavalan, K. Tsiakmakis, N. Charalampidis, and S. W. Cheung, “Optimization of a 12.5GHz microstrip antenna array using Taguchi's method,” *International Journal of Antennas and Propagation,* vol. 2011, 2011.

[90] D. Simon, “Training radial basis neural networks with the extended kalman filter,” *Neurocomputing,* vol. 48, pp. 455-475, 2002.

[91] J. Liu, *Radial Basis Function (RBF) Neural Network Control for Mechanical Systems*. DE: Springer Verlag, 2013.

[92] Y. Wu, H. Wang, B. Zhang, and K.-L. Du, “Using radial basis function networks for function approximation and classification,” *ISRN Applied Mathematics,* vol. 2012, p. 34, 2012.

[93] M. A. Panduro, A. L. Mendez, R. Dominguez, and G. Romero, “Design of non-uniform circular antenna arrays for side lobe reduction using the method of genetic algorithms,” *AEU International Journal of Electronics and Communications,* vol. 60, pp. 713-717, 2006.

[94] S. Koziel, D. Ciaurri, and L. Leifsson, “Surrogate-Based Methods,” in Computational Optimization, Methods and Algorithms. vol. 356, S. Koziel and X.-S. Yang, Eds., ed: Springer Berlin Heidelberg, 2011, pp. 33-59.

[95] C. Jariyatantiwait, and G.G. Yen, “Multiobjective differential evolution based on fuzzy performance feedback,” *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 5, pp. 45-64, 2015.

[96] L. Tang, G. Wang, and Z. Chen, “Integrated charge batching and casting width selection at Baosteel,” *Operation Research*, vol. 62, no. 4, pp. 772-787, 2014.

[97] L. Tang, Y. Zhao, and J. Liu, “An improved differential evolution algorithm for practical dynamic scheduling in steelmaking-continuous casting production,” *IEEE trans. on Evolutionary Computation,* vol. 18, no. 2, pp. 209-225, 2014.

[98] M. A. Panduroa, D. H. Covarrubiasa, C. A. Brizuelaa, and F. R. Maranteb, “A multi-objective approach in the linear antenna array design,” *Int. J. Electron. Commun.*, vol. (AEÜ) 59, pp. 205-212, 2005.

[99] D. S. Brooomhead and D. Lowe, “Multivariable functional interpolation and adaptive networks,” *Complex Systems,* vol. 2, no. 3, pp. 321-355, 1988.

[100] Z. Michalewicz and M. Schoenauer, “Evolutionary algorithms for constrained parameter optimization problems,” Evolut. Comput., vol. 4, no. 1, pp. 1-32, 1996.

[101] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Trans. on Evolutionary. Computation,* vol. 6, no. 2, pp. 182-197, 2002.

[102] N. Srinivas and K. Deb, "Multiobjective function optimization using nondominated sorting genetic algorithms," *Evolutionary Computation*, vol. 2, no.3, pp. 221-248, 1995.

[103] M. Tanaka, "GA-based decision support system for multicriteria optimization," in *Proc. of IEEE International Conference on Syst. Man. Cybernetics.*, 1995, pp. 1556-1561.

[104] A. Osyczka and S. Kundu, "A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm", *Struct. Optimz.*, vol. 10, no. 2, pp. 94-99, 1995.

[105] A. Homaifar, C. X. Qi, and S. H. Lai, "Constrained optimization via genetic algorithms," *Simulat.*, vol. 62, no. 4, pp. 242-253, 1994.

[106] J. A. Joines and C. R. Houck, "On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GA's," in *Proc. 1st IEEE Conf. Evolut. Comput*, 1994, pp. 579-584.

[107] S. Mu, H. Su, W. Mao, Z. Chen, and J. Chu, "A new genetic algorithm to handle the constrained optimization problem," in *Proc. 41st IEEE Conf. Decis. Control*, Las Vegas, Nevada, USA, 2002, pp. 739-740.

[108] J. C. Bean, and A. Hadj-Alouane, "A dual genetic algorithm for bounded integer programs," *Tech. Rep*. 92-53, Univ. Michigan, Oct., 1993.

[109] B. Tessema, and G. G. Yen, "An adaptive penalty formulation for constrained evolutionary optimization," *IEEE Trans. on System Man and Cybernatics:Part A*, vol. 39, no. 3, pp. 565-578, 2009.

[110] Y. G. Woldesenbet, G. G. Yen, and B. G. Tessema, "Constraint handling in multiobjective evolutionary optimization," *IEEE Trans. on Evolutionary Computation*, vol. 13, no. 3, pp. 514-525, 2009.

[111] A. B. Hadj-Alouane and J. C. Bean, "A genetic algorithm for the multiple-choice integer program," *Operation Research*, vol. 45, no.1, pp. 92-101, 1997.

[112] B. Y. Qu and P. N. Suganthan, "Constrained multi-objective optimization algorithm with diversity enhanced differential evolution," in *Proc. of IEEE Congress on Evolutionary Computation*, 2010, pp. 1-5.

[113] D. E. C. Vargas, A. C.C. Lemonge, H. J.C. Barbosa, and H. S. Bernardino. "Constraint handling in multiobjective evolutionary optimization," in *Proc. of IEEE Congress on Evolutionary Computation*, 2013, pp. 1342-1349.

[114] S. Ben Hamida and M. Schoenauer, "ASCHEA: new results using adaptive segregational constraint handling." in *Proc. of IEEE Congress on Evolutionary Computation*, 2002, pp. 884-889.

[115] C. A. Coello Coello, “Constraint-handling using an evolutionary multiobjective optimization technique,” *Civil. Eng. Env. Syst.*, vol. 17, no. 4, pp. 319-346, 2000.

[116] T. Ray, K. Tai, and C. Seow, “An evolutionary algorithm for multiobjective optimization,” *Engineering Optimizatio*, vol. 33, no. 3, pp. 399-424, 2001.

[117] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, 2002.

[118] P. D. Surry and N. J. Radcliffe, “The COMOGA method: constrained optimization by multi-objective genetic algorithms,” *Control Cybern.*, vol. 26, pp. 391-412, 1997.

[119] S. Venkatraman and G. G. Yen, “A generic framework for constrained optimization using genetic algorithms,” *IEEE Trans. on Evolutionary Computation*, vol. 9, no. 4, pp. 424-435, 2005.

[120] G. G. Yen and W. F. Leong, “A multiobjective particle swarm optimizer for constrained optimization,” *International Journal of Swarm Intelligence Research*, vol. 2, no. 1, pp. 1-23, 2011.

[121] L. Tang and X. Wang, “A hybrid multiobjective evolutionary algorithm for multiobjective optimization problems,” *IEEE Trans. on Evolutionary Computation*, vol. 17, no. 1, pp. 20-45, 2013.

[122] I. Berrada, J. A. Ferland, and P. Michelon, “A multi-objective approach to nurse scheduling with both hard and soft constraints,” *Socio-Economic Planning Science*, vol. 30, no. 3, pp. 183-193, 1996.

[123] D. Wang and S.C. Fang, “A genetics-based approach for aggregated production planning in a fuzzy environment,” *IEEE Trans. on System Man and Cybernatics:Part A*, vol. 27, no. 5, pp. 636- 645, 1997.

[124] H. Fargier and J. Lamothe, “Handling soft constraints in hoist scheduling problems: the fuzzy approach,” *Soft Computing for Planning and Scheduling* , vol. 14, no. 3, pp. 387-399, 2001.

[125] A. Isaacs, T. Ray, W. Smith. “Blessings of maintaining infeasible solutions for constrained multi-objective optimization problems,” in *Procs. of IEEE Congress on Evolutionary Computation*, 2008, pp. 2785-2792.

[126] H. K. Singh, A. Isaacs, T. Ray, and W. Smith, “Infeasibility driven evolutionary algorithm (IDEA) for engineering design optimization,” in *AI 2008: Advances in Artificial Intelligence*, Springer, Berlin: Heidelberg, 2008, pp. 104-115.

[127] T. Ray, H. K. Singh, A. Isaacs, and W. Smith, “Infeasibility driven evolutionary algorithm for constrained optimization,” in Constraint-handling in evolutionary

optimization, ser. Studies in Computational Intelligence, E. Mezura-Montes, Springer, Berlin Heidelberg, 2009, vol. 198, pp. 145-165.

[128] H. K. Singh, M. Asafuddoula, and T. Ray, “Solving problems with a mix of hard and soft constraints using modified infeasibility driven evolutionary algorithm (IDEA-M),” in *Procs. of IEEE Congress on Evolutionary Computation*, Beijing, China, Jul, 2014, pp. 983-990.

[129] L. V. Santana-Quintero , A. G. Hernández-Díaz , J. Molina , C. A. C. Coello and R. Caballero "DEMORS: A hybrid multiobjective optimization algorithm using differential evolution and rough sets for constrained problems," *Computer and Operation Research*, vol. 37, no. 3, pp. 470-480, 2010.

[130] B. Y. Qu and P. N. Suganthan, “Constrained multi-objective optimization algorithm with diversity enhanced differential evolution,” *Procs. of IEEE Congress on Evolutionary Computation*, 2010, pp. 1-5.

[131] B. Y. Qu and P. N. Suganthan, “Constrained multi-objective optimization algorithm with an ensemble of constraint handling methods,” *Engineering Optimization*, vol. 43, no. 4, 2011, pp. 403-416.

[132] K. Zielinski, P. Dagmar, and R. Laur. "Constrained Multi-Objective Optimization Using Differential Evolution." *Third International Conference on Computational Intelligence, Robotics and Autonomous Systems*, 2005.

[133] M. Zhang, H. Geng, W. Luo, L. Huang, and X. Wang, “A Hybrid of Differential Evolution and Genetic Algorithm for Constrained Multiobjective Optimization Problems,” in Simulated Evolution and Learning. vol. 4247, T.-D. Wang, X. Li, S.-H. Chen, X. Wang, H. Abbass, H. Iba, G.-L. Chen, and X. Yao, Eds., ed: Springer Berlin Heidelberg, 2006, pp. 318-327.

[134] E. Mezura-Montes and C. A. Coello Coello, “Constraint-handling in nature-inspired numerical optimization: past, present and future,” *Swarm and Evolutionary Computation*, vol. 1, no. 4, pp. 173-194, 2011.

[135] C. A. Coello Coello, “Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art,” *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 11-12, pp. 1245-1287, 2002.

[136] Z. Michalewicz, D. Dasgupta, R. G. Le Riche and M. Schoenauer. “Evolutionary algorithms for constrained engineering problems”, *Computer and Industrial Engineering*, vol. 30, no. 4, pp. 851-870, 1996.

[137] A. Zamuda, J. Brest, B. Bokovic, and V. Zumer, “Differential Evolution with Self-adaptation and Local Search for Constrained Multiobjective Optimization,” *IEEE Congress on Evolutionary Computation*, 2009, pp. 195-202.

[138] H. Min-Nan, C. Tsung-Che, and F. Li-Chen, “A hybrid constraint handling mechanism with differential evolution for constrained multiobjective optimization,” *IEEE Congress on Evolutionary Computation*, 2011, pp. 1785-1792.

[139] J. J. Liang, B. Zheng, F. Y. Xu, B. Y. Qu, and H. Song, “Multi-objective differential evolution algorithm based on fast sorting and a novel constraints handling technique,” *IEEE Congress on Evolutionary Computation*, 2014, pp. 445-450.

[140] J. Heaton, Introduction to neural networks for Java, 2 ed.: Heaton Research, Inc., 2008

VITA

Chatkaew Jariyatantiwait

Candidate for the Degree of

Doctor of Philosophy

Thesis: MULTIOBJECTIVE DIFFERNTIAL EVOLUTION BASED ON FUZZY PERFORMANCE FEEDBACK: SOFT CONSTRAINT HANDLING AND ITS APPLICATION IN ANTENNA DESIGNS

Major Field: Electrical Engineering

Biographical:

Education:

Completed the requirements for the Doctor of Philosophy in Electrical Engineering at Oklahoma State University, Stillwater, Oklahoma in December, 2015.

Obtained the Master of Engineering in Electrical Engineering at Chiangmai University, Chiang Mai, Thailand in 2002.

Obtained the Bachelor of Engineering in Electrical Engineering at Chiangmai University, Chiang Mai, Thailand in 1997.

Experience:

Lecturer at the Faculty of Engineering, Rajamangala University of Technology Phra Nakhon, Bangkok, Thailand, 2002 – present.