

博士論文

**Adaptively Preserving Solutions
in Feasible and Infeasible Regions
for Solving Severely Constrained
Engineering Design Optimization Problems**

(厳しい制約条件が課された工学設計最適化問題を解くために
実行可能領域と実行不可能領域の解を適応的に保存する方法)

ディアント ヨハネス ビモ

Acknowledgments

I am living my dream. It never occurred to me that I could get an opportunity to conduct my doctoral study in The University of Tokyo and have a lab in Japan Aerospace Exploration Agency. It was not an easy process, but I definitely enjoyed every second of my study during my stay in Japan.

I would like to thank God Almighty, for giving me health, strength, and focus so that I can cope with the pressure and finally finish this dissertation.

I also would like to give my sincere gratitude to Prof. Akira Oyama, for accepting me as his doctoral student and supervising me with your utmost patient. You teach me a lot about how to do a good research. Without your comments, questions, and suggestions, this dissertation would never finish.

My sincere thanks also goes to Prof. Takeshi Tsuchiya, Prof. Koji Shimoyama, Prof. Kenichi Rinoie, and Prof. Taro Imamura for examining my pre-defense and final defense. Your comments and questions are also very critical for the discussion in this dissertation. Special thanks to Prof. Koji Shimoyama for inviting me to visit Tohoku University. You gave me good insights to help me decide my research topic.

I would also like to express my gratitude to Indonesia Endowment Fund for Education (Lembaga Pengelola Dana Pendidikan Republik Indonesia - LPDP RI) for providing me scholarship so that I can conduct my doctoral study without any financial worry.

I do not forget to thank all Oyama Lab members, either the past or the current members for giving me support in the lab. Especially Hiroaki Fukumoto and Shigetaka Kawai, you give me so many help during my stay in Japan on either academic or non-academic related things.

It is also important for me to thank Pramudita Satria Palar, Ph.D, for connecting me to Oyama-sensei, teaching me how to make connection to other fellow researchers, and inviting me for fruitful discussions.

Lastly, I would like to thank my family for their continuous support. Especially my mother for your continuous prayer and my father for your great advise.

Abstract

In the present work, constraint handling techniques (CHTs) suitable for solving severely constrained engineering design optimization problems with evolutionary algorithm (EA) are proposed. The severities of constraint in engineering design optimization problem often include having small feasible region, where the feasible individuals are difficult to find, and many constraints, which make finding constrained optimum even trickier to do. Engineering design problem also often has objective and constraint functions with different orders of magnitudes. If no special treatment is given, the assessment of the functions might be imbalanced because the functions having higher orders of magnitudes will dominate the fitness value. Since engineering design problem often utilizes computationally expensive solver to obtain the objective and constraint function values, the optimizer needs to be able to find the constrained optimum within a small solution evaluation budget. Therefore, CHT which can help the optimizer reaching that goal is needed.

To efficiently find a constrained optimum on a severely constrained problem, studies have been conducted that balancing the search in both feasible and infeasible regions is important. Among other CHTs, multiple constraint ranking (MCR) balances the assessment of constraints by building separate queue for each constraint violation. It also balances the search in both feasible and infeasible regions when the number of constraints is small. However, because it becomes exploring too much of feasible region when the number of constraints is large, it is still not very effective for solving problems with many constraints. Therefore, in the present work, we develop some new CHTs by modifying MCR to enhance its convergence performance mainly for solving severely constrained engineering design optimization problems. Basically, the modifications are conducted in two steps: to adaptively enhance the infeasible region exploration when the number of constraints is large and to adaptively balance the search in both feasible and infeasible regions.

In the first step, we propose some CHTs to enhance the infeasible region exploration by modifying MCR based on three principles: the constraints might have different difficulty levels, omitting the rank of number of constraints violated might give more flexibility in exploring infeasible region, and the portion of constraint violation ranks is too large on problem with large number of constraints. In an investigation on a car structure de-

sign problem, some of our proposed CHTs produce significant convergence improvements compared with MCR. Those convergence improvements might be produced because the CHTs generate more variety of infeasible individuals and many infeasible individuals with lower objective values than the feasible ones. However, the CHTs seem to explore too big of infeasible region. This draws concern that they might face difficulties in generating feasible individuals on other types of problems.

In the second step, we propose some other CHTs by further modifying the obtained CHTs from the previous step to improve the balance of search in both feasible and infeasible region by implementing adaptive weight on the objective value term and constraint violation term. In investigations on a total of 78 benchmark problems comprising various types, some proposed CHTs generate more robust convergence performances compared with the CHTs from the previous step. In a more detailed observation on real-world design problems (RWDPs) (car structure design, MOPTA 2008, and wind turbine), one proposed CHT even tends to generate better convergence improvement than the CHTs from the previous step. Other than having the same observed effects in the previous step, the proposed CHTs which produce more robust performance tend to have small feasibility ratio in the offspring population.

In the last part of our work, we implement some of our proposed CHTs on transonic and multi-point airfoil design optimization problems and compare the performance with MCR and conventional CHT. On transonic airfoil design optimization, the drag coefficient of airfoil in transonic condition is minimized. Our proposed CHTs produce significantly better convergence performance compared with MCR and the conventional CHT. Each optimum shape can produce lower drag coefficient than the baseline design most probably because they generate weaker shockwave on the upper surfaces. On multi-point airfoil design optimization, the combined drag coefficient of airfoil from supersonic, transonic, and subsonic flight conditions is minimized. One of our proposed CHTs produces better convergence than MCR and the conventional CHTs. It seems that the problem definition of multi-point airfoil design optimization in the present work is very difficult because some CHTs face difficulty in obtaining feasible individuals with better objective value than the baseline airfoil. EA seems to favor optimization in supersonic condition because it has larger weight during the optimization.

Contents

| | |
|---|-------------|
| Acknowledgments | i |
| Abstract | iii |
| Contents | v |
| List of Figures | ix |
| List of Tables | xiii |
| 1 Introduction | 1 |
| 1.1 Research Background | 1 |
| 1.2 Literature Study Regarding Constraint Handling Techniques | 3 |
| 1.3 Research Objectives and Scopes | 7 |
| 1.4 Contribution of This Work | 8 |
| 1.5 Thesis Outline | 10 |
| 2 Theory | 11 |
| 2.1 Introduction | 11 |
| 2.2 Evolutionary Algorithm | 11 |
| 2.3 Existing Constraint Handling Technique for EA | 16 |
| 2.3.1 Category A | 17 |
| 2.3.2 Category B | 17 |
| 2.3.3 Category C: Multiple Constraint Ranking | 21 |
| 3 Adaptively Enhancing Infeasible Region Exploration | 23 |
| 3.1 Introduction | 23 |
| 3.2 Generalized Multiple Constraint Ranking | 23 |
| 3.3 Adaptively Enhancing Infeasible Region Exploration | 24 |
| 3.4 Experimental Setup | 26 |
| 3.4.1 Benchmark problem | 26 |

| | | |
|----------|--|-----------|
| 3.4.2 | Optimization setting | 27 |
| 3.5 | Result and Discussion | 27 |
| 3.5.1 | Overall result | 27 |
| 3.5.2 | The effect of the modifications | 29 |
| 3.6 | Summary | 37 |
| 4 | Adaptively Preserving Solutions in Both Feasible and Infeasible Regions | 39 |
| 4.1 | Introduction | 39 |
| 4.2 | Adaptively Preserving Solutions in Both Feasible and Infeasible Regions . . | 39 |
| 4.3 | Experimental Setup | 41 |
| 4.3.1 | Benchmark problems | 41 |
| 4.3.2 | Optimization setting | 43 |
| 4.4 | Result and Discussion | 44 |
| 4.4.1 | Overall result - comparison with MCR | 44 |
| 4.4.2 | Discussion on results related to real-world design problems (RWDPs) | 50 |
| 4.4.3 | Discussion regarding the feasibility ratio in offspring population in RWDPs | 60 |
| 4.5 | Summary | 62 |
| 5 | Airfoil Aerodynamic Design Optimization | 65 |
| 5.1 | Introduction | 65 |
| 5.2 | Airfoil Parameterization: Cubic B-Spline Curve | 66 |
| 5.3 | CFD Solver: LANS3D | 67 |
| 5.4 | Transonic Airfoil Design Optimization Problem | 67 |
| 5.4.1 | Problem Definition and Optimization Setting | 67 |
| 5.4.2 | CFD Validation | 70 |
| 5.4.3 | Optimization Result and Discussion | 73 |
| 5.4.4 | Analysis on the Shape of Optimum Designs and the Flow Fields . . | 80 |
| 5.5 | Multi-point Airfoil Design Optimization | 80 |
| 5.5.1 | Problem Definition and Optimization Setting | 85 |
| 5.5.2 | CFD Validation | 86 |
| 5.5.3 | Optimization Result and Discussion | 88 |
| 5.5.4 | Analysis on the Shape of Optimum Designs and the Flow Fields . . | 94 |
| 5.6 | Summary | 95 |
| 6 | Overall Summary | 99 |
| 6.1 | Conclusion | 99 |
| 6.2 | Obtained Knowledge and Recommended CHT | 100 |
| 6.3 | Future Work | 101 |

List of Figures

| | | |
|------|--|----|
| 3.1 | Box plot of obtained optimum values | 28 |
| 3.2 | N_ν spreading of the CHTs (median of 51 runs). | 30 |
| 3.3 | Objective value spreading of the CHTs (median of 51 runs). | 31 |
| 3.4 | Average α_i of constraint 9 from 51 independent runs | 33 |
| 3.5 | Average α_i of constraint 8 from 51 independent runs | 33 |
| 3.6 | Average α_i of constraint 7 from 51 independent runs | 34 |
| 3.7 | Histories of average diversity in constraint space | 35 |
| 3.8 | Histories of average diversity in design variable space | 36 |
| 4.1 | Visualization of β_1 and β_2 (normalized in such way so that $\beta_1 + \beta_2 = 1$). . | 42 |
| 4.2 | Box plot of G3 problem | 48 |
| 4.3 | Histories of average β_1 of G-MCR-based CHTs on G3 problem | 49 |
| 4.4 | Box plot of car structure design problem. | 52 |
| 4.5 | N_ν and objective value spreadings of selected CHTs (median of 51 runs) on car structure design problem | 53 |
| 4.6 | Average constraint diversity of MCR, MCR_9, G-CMR-L_9, G-CMR-Q_9, and G-CMR-C_9 in car structure design problem. | 54 |
| 4.7 | Box plot of MOPTA 2008 problem. | 54 |
| 4.8 | N_ν and objective value spreadings of selected CHTs (median of 51 runs) on MOPTA 2008 problem | 55 |
| 4.9 | Average constraint diversity of MCR, MCR_9, G-CMR-L_9, G-CMR-Q_9, and G-CMR-C_9 in MOPTA 2008 problem. | 56 |
| 4.10 | Box plot of wind turbine problem. | 57 |
| 4.11 | N_ν and objective value spreadings of selected CHTs (median of 51 runs) on wind turbine problem | 58 |
| 4.12 | Average constraint diversity of MCR, MCR_9, G-CMR-L_9, G-CMR-Q_9, and G-CMR-C_9 in wind turbine problem. | 59 |
| 4.13 | Average feasibility ratio of offspring population over the generation on car structure design problem from 51 independent runs (selected CHTs). . . . | 60 |

| | | |
|------|---|----|
| 4.14 | Average feasibility ratio of offspring population over the generation on MOPTA 2008 problem from 51 independent runs (selected CHTs). | 61 |
| 4.15 | Average feasibility ratio of offspring population over the generation on wind turbine problem from 51 independent runs (selected CHTs). | 62 |
| 5.1 | Illustration of airfoil generation with cubic B-spline curve. | 68 |
| 5.2 | Search space for transonic airfoil design optimization | 69 |
| 5.3 | Area calculation | 69 |
| 5.4 | Grid visualizations at (a) full and (b) near wall views of RAE 2822 airfoil | 71 |
| 5.5 | C_p distribution of RAE 2822 airfoil from LANS3D and NPARC Alliance simulations and experiment by AGARD | 72 |
| 5.6 | Box plot of obtained optimum drag coefficients from 11 independent runs | 73 |
| 5.7 | N_v and objective value spreadings of the CHTs (median of 51 runs) | 74 |
| 5.8 | Average of feasibility ratio from 11 independent runs | 75 |
| 5.9 | Average α_1 of transonic airfoil design from 11 independent runs | 76 |
| 5.10 | Average α_2 of transonic airfoil design from 11 independent runs | 76 |
| 5.11 | Average α_3 of transonic airfoil design from 11 independent runs | 77 |
| 5.12 | Average of diversity in (a) constraint and (b) design variable spaces | 78 |
| 5.13 | (a) Shapes and (b) C_p distributions of baseline and optimum airfoils in median results | 81 |
| 5.14 | C_p flow field of baseline and optimum airfoils (median) | 82 |
| 5.15 | (a) Shapes and (b) C_p distributions of baseline and optimum airfoils in best results | 83 |
| 5.16 | C_p flow field of baseline and optimum airfoils (best) | 84 |
| 5.17 | Search space for multi-point airfoil design optimization | 86 |
| 5.18 | Grid visualizations at (a) full and (b) near wall views of NACA-2S-(40)(015)-(40)(015) airfoil | 87 |
| 5.19 | Box plot of obtained optimum combined drag coefficients from 5 independent runs | 88 |
| 5.20 | N_v and objective value spreadings of the CHTs (median of 51 runs). | 90 |
| 5.21 | Average α_i on multi-point airfoil design from five independent runs | 91 |
| 5.22 | Average of feasibility ratio from 11 independent runs | 92 |
| 5.23 | Average of diversity in (a) constraint and (b) design variable spaces | 93 |
| 5.24 | Shapes of baseline and optimum airfoils in best results | 95 |
| 5.25 | C_p flow field of the baseline (upper) and G-MCR-C_9's airfoil (lower) in supersonic condition. | 96 |
| 5.26 | C_p flow field of the baseline (upper) and G-MCR-C_9's airfoil (lower) in transonic condition. | 97 |

| | |
|--|----|
| 5.27 Cp flow field of the baseline (upper) and G-MCR-C_9's airfoil (lower) in subsonic condition. | 98 |
|--|----|

List of Tables

| | | |
|-----|---|----|
| 3.1 | MCR and its modifications | 25 |
| 3.2 | Statistical significance test (1/2) | 28 |
| 3.3 | Statistical significance test (2/2) | 29 |
| 4.1 | MCR_7, MCR_8, and MCR_9 | 40 |
| 4.2 | G-MCR with new modifications, $\zeta \equiv \frac{N_{feas}}{N_{pop}}$ | 42 |
| 4.3 | CHTs against MCR (all 78 problems) | 45 |
| 4.4 | CHTs against MCR (problems involving only inequality constraints, 38 problems) | 46 |
| 4.5 | CHTs against MCR (problems involving either only equality constraints or both inequality and equality constraints, 40 problems) | 47 |
| 4.6 | Properties of RWDPs | 50 |
| 4.7 | Overall score of significance test (CHTs against other CHTs) on RWDPs | 51 |
| 5.1 | Aerodynamic properties of RAE 2822 airfoil | 72 |
| 5.2 | Aerodynamic properties of RAE 2822 airfoil | 72 |
| 5.3 | Statistical significance test for transonic airfoil design | 75 |
| 5.4 | Flight conditions and aerodynamic properties of baseline airfoil | 85 |
| 5.5 | Aerodynamic properties of NACA-2S-(40)(015)-(40)(015) airfoil | 86 |
| 5.6 | Statistical significance test for multi-point airfoil design | 88 |
| 5.7 | Drag coefficients ($\times 10^4$) of baseline and optimum airfoils (best result) | 95 |

Chapter 1

Introduction

1.1 Research Background

Design optimization is an everlasting process in engineering. As demands in the society changes over time, designers need to keep improving their designs in order to meet those demands. One example is on the development of aircraft. It was started from a dream to fly from more than 2000 years ago, mankind invented things such as kites and gliders. Then, balloons and airships were invented, so that more people can fly. Until in the early 19th century, Wright brothers successfully invented the first heavier-than-air powered-design aircraft. This invention was the pioneer of the current aeronautical science development. Basing on Wright brothers' design, many kinds of aircraft, from subsonic to hypersonic, have been built for many purposes such as commercial and military.

In order to meet their respective purposes, optimization has become an important aspect in making designs efficiently in engineering, that the optimization method itself becomes an important research topic. The oldest type of optimization is by trial and error, where an enormous variety of test subjects are conducted in order to find a single best design. However, because this method is time-consuming and there is no guarantee that the obtained best design is globally optimum, more sophisticated methods such as gradient-based and stochastic optimization methods are more popular in making engineering design nowadays.

Gradient-based method can be utilized when the objective and constraint functions of the optimization problem are differentiable, since it requires gradient information of those objective and constraint functions. One example is in aerodynamic wing/airfoil shape design optimization [1–7], where gradient-based methods such as SNOPT (sparse nonlinear optimizer) [8] and MATLAB's `fmincon` are utilized extensively. Because the calculation of lift, drag, moment coefficients and other properties are modeled mathematically by computational fluid dynamics (CFD) which is continuous and differentiable,

albeit highly nonlinear, it is possible to calculate the gradient. Gradient-based method becomes difficult to utilize, when the objective and/or the constraint functions are non-differentiable. For example, when the objective and constraint functions are evaluated by an experiment, such as what conducted in the flapping wing experiment of micro-aerial vehicle [9, 10], there is no mathematical expression to derive, hence obtaining gradient information is difficult, if possible. And, if (some of) the objective functions, constraint functions, and/or design variables are discrete values such as what occurred in a car structure design problem [11], those functions and/or design variables are non-differentiable. Gradient-based method is also not efficient to utilize when the problem is multi-modal since it is prone to be trapped in the local optima. One can only hope that the initial point is in the correct region thus the global optimum can be obtained.

For such problem which is non-differentiable and/or multi-modal, stochastic method such as evolutionary algorithm (EA) is more popular to utilize than gradient-based method because it has the ability to escape from local optima and find global optimum and it does not require any gradient information. For differentiable problems such as aerodynamic wing/airfoil shape design, EA also offers ease of implementation owing to its simplicity in coupling the optimization algorithm with the function evaluator (CFD). There is no necessity of adding the complexity of the optimization by calculating gradient in every iteration because EA only requires the values of the objective and constraint functions. The main drawback of EA is that it requires a lot of solution evaluations for one optimization. However, with the current advancement of computing technology such as parallelization, such drawback can be alleviated.

Engineering design optimization problem is often formulated with complex expression in such way that the process in obtaining an output from an input is difficult to explain, such as aerodynamic wing/airfoil design problem utilizing CFD and the car structure design problem utilizing finite element method (FEM) which are highly nonlinear. Almost certainly, the calculation of gradient will also be complex. Owing to its ease of implementation, EA has rising popularity in solving engineering design problem despite whether multi-modality and non-differentiability really exist or not. For example in aerospace engineering, EA has also been utilized in solving aerodynamic wing/airfoil design problems [12, 13], as well as many other problems such as space trajectory/exploration design [14–17], Martian rotor design [18, 19], structural design [20], axial compressor [21], and flapping wing of MAV [9]. In mechanical engineering, EA has been utilized to optimize surfboard fin shape [22], high speed train nose [23, 24], and car structure design [25]. EA has also been utilized in other areas such as gaming, robotics, and animation to improve the reinforcement learning or neural network performances [26–30].

Engineering design optimization problem is often severely constrained, i.e. has small feasible region and many constraints. In addition, the objective and constraint functions

can be of different order of magnitudes. When a problem is severely constrained, the feasible solution tends to be difficult to find. Meanwhile, when the objective and constraint functions have different order of magnitudes, imbalanced assessment of functions can occur unless a proper treatment is implemented in the optimizer. Car structure design [11], MOPTA 2008 [31], and wind turbine [32] problems are some suitable engineering design optimization problems. Car structure design optimization problem [11] generates no feasible solution from 1,000,000 random tests. It comprises 54 various kinds of constraints with different orders of magnitudes and/or units such as mass, crashworthiness, body torsional stiffness, and vibration modes. Similarly, MOPTA 2008 [31] and wind turbine [32] problems also generate no feasible solution from 1,000,000 and 100,000 random tests, respectively. MOPTA 2008 problem comprises 63 constraints with different orders of magnitude and/or units such as mass, crash performance, noise vibration, harshness, and durability. Wind turbine problem comprises 22 constraints with different orders of magnitudes and/or units such as deflection, natural frequency, stress, strain, and speed. In these problems, the tools to calculate the gradient is also not provided. Therefore, stochastic method such as EA is the most suitable optimizer to be utilized. EA is devised for solving unconstrained optimization. Therefore, a treatment to handle the constraints, usually called constraint handling technique (CHT), needs to be implemented in order to solve a constrained problem. In the present work, we are interested in devising new CHT which can more efficiently solve an engineering design optimization problem.

1.2 Literature Study Regarding Constraint Handling Techniques

Suppose a minimization problem, we can express the general form of single-objective constrained optimization as follows,

$$\begin{aligned}
 &\text{Minimize} && f(\mathbf{x}) \\
 &\text{subject to} && g_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, n \\
 &&& h_{j-n}(\mathbf{x}) = 0, \quad j = n + 1, \dots, m \\
 &&& \mathbf{x}_L \leq \mathbf{x} \leq \mathbf{x}_U
 \end{aligned} \tag{1.1}$$

where \mathbf{x} denotes the d -dimensional solution vector. Symbols f , g_i , and h_j denote the objective function, the i^{th} inequality constraint, and the $(j - n)^{th}$ equality constraint, respectively. Then, m and n denote the number of all constraints and inequality constraints, respectively. Lastly, both \mathbf{x}_L and \mathbf{x}_U denote the lower and upper bounds of the search space, respectively.

EA is a subset of evolutionary computation. It is a population-based algorithm which conduct an optimization based on the principle of evolution. The basic framework usually

includes parent selection, recombination/ crossover, mutation, and environmental selection. Some types of EA are genetic algorithm [33], genetic programming [34], evolution strategy [35], differential evolution [36], and neuroevolution [37].

In EA, the equality constraints is often modified into inequality constraints, as follows

$$g_j(\mathbf{x}) = |h_{j-n}(\mathbf{x})| - \epsilon \leq 0 \quad j = n + 1, \dots, m \quad (1.2)$$

where ϵ is a small number. Therefore, there are only inequality constraints in the general form. In the present work, if a problem also involves equality constraint(s), we transform them into inequality constraints with Eq. 1.2 with $\epsilon = 1 \times 10^{-4}$.

Before elaborating the literature study in detail, we introduce some common terminologies in EA,

1. individual, is a solution vector \mathbf{x} ,
2. objective value, is a short term for objective function value $f(\mathbf{x})$,
3. constraint value, is a short term constraint function value, either the inequality constraint function $g(\mathbf{x})$ or equality constraint function $h(\mathbf{x})$.
4. feasible individual is an individual which satisfies all constraints, and
5. infeasible individual is an individual which violates at least one constraint.

According to Coello (2019) [38], the present CHTs can be classified into five categories:

1. Penalty functions
2. Special representations and operators
3. Repair algorithms
4. Separation of constraints and objectives
5. Hybrid methods

The penalty functions work in principle by combining the values of objective and constraint functions. This practice makes the EA can be utilized as it is in solving a constrained optimization problem. The practicality of this category makes it the most common CHT type to be utilized in the current research of CHT. In this category, many kinds popular CHTs have been developed, such as superiority of feasible solutions (SoF) [39] and stochastic ranking (SR) [40]. SoF favors feasible individuals over infeasible individuals in such way that if both feasible and infeasible individuals exist in the population, the feasible individual will have better fitness value than the infeasible one. SR assesses the

fitness value of an individual based on either the objective value or the sum of constraint violations, depending on a user-defined probability.

As for the special representations and operators category, the main idea is to transform the original problem into another topologically equivalent function which is easier to optimize. One example is the work by Koziel and Michalewicz [41], where the feasible search space is transformed into n -dimensional cube with a homomorphous mapping. While this CHT was quite competitive (before the publication of stochastic ranking), the implementation of this CHT is quite complex.

In the repair methods category, the main idea is to devise a mechanism to change an infeasible individual into a feasible one. This category of CHT is utilized often in combinatorial optimization such as traveling salesman and knapsack problems. This category of technique is a good choice when the infeasible individual can be easily transformed into the feasible one. However, the technique in this category is mostly problem-dependent and in some cases the repair operator may produce bias in the search.

In the separation of objective and constraint functions category, the objective and constraint functions are assessed separately. The common technique is to expand the problem into multi-objective optimization by considering the constraint functions as an additional objective function. However, the use of multi-objective algorithm adds another challenges such as the usage of Pareto ranking might not be effective in handling constraints. Based on Runnarson and Yao's work [42], it seems that the use of Pareto ranking causes the search to spend most time searching in the infeasible region.

In hybrid methods, coupling of two different techniques or heuristics is considered. One of the techniques or heuristics can also be deterministic (mathematical) programming approach. For example, Wah and Chen [43] couples simulated annealing, another type of evolutionary computation methods, and genetic algorithm for the optimizer. The constraints are assessed with Lagrangian Multipliers.

Due to its ease of implementation, we focus on developing CHTs in penalty functions category. Death penalty [44,45] is the probably the oldest kind of penalty function CHT. It works by giving the infeasible individuals the worst fitness values or simply excluding them during the optimization. This technique is often used in the early development of evolutionary computation. It works quite well when the feasible region of the problem is quite large. However, when the feasible region is strict and the feasible individual is difficult to find, this technique faces difficulty because it depends solely on feasible individuals to search for the global optimum. Some techniques have been developed to improve the performance such as static penalty [46] and dynamic penalty [47]. However, these techniques utilize some user-defined factors which are problem-dependent. It is not a trivial task to set those factors for each problem. To alleviate this issue, Deb [39] proposed superiority of feasible solution (SoF), where no user-defined parameter needed.

The general issue of penalty function category is how to define the penalty factor. If the penalty factor is too high and the optimum is located near the constraint boundary, the optimization algorithm will immediately push the search into feasible region and it will be difficult to move it towards the constraint boundary again. Meanwhile, if the penalty factor is too low, the algorithm might explore mostly in infeasible region. Conventional CHTs such as death penalty and superiority of feasibility (SoF) tend to favor feasible individuals over infeasible individuals. In severely constrained problem, whose feasible individuals are often difficult to produce, this practice might be inefficient since they cannot make use of the information provided by infeasible individuals efficiently to help the search finding constrained optimum. In consequence, there is a high possibility that most individuals will crowd in feasible region and it will be difficult to push them to the constraint boundary.

In recent years, the development of penalty function-based CHTs tend to consider the balanced search between feasible and infeasible regions. By balancing the search from both regions, the attempt to find the constrained optimum can be more efficient. One example is stochastic ranking (SR) [40], a technique which mimics bubble-sort method to rank the individuals in the population based on either the sum of constraint violations or the objective values depends on a user-defined probability, P_f . Then, the same author of SR proposed a new method called global competitive ranking (GCR) [48] which builds the rank of the objective values and rank of the sum of constraint violations directly from the population, then balances the proportion of those ranks in calculating the fitness value based on a user-defined coefficient. The user-defined coefficient P_f is the balancing factor between search in feasible and infeasible regions. In order to make sure that feasible individuals are found without over-penalizing the search, $0.4 \leq P_f < 0.5$ is recommended.

There is also adaptive penalty method (APM) [49] which utilizes average of objective and constraint function values in assessing the infeasible individuals. Because of this practice, the infeasible individuals with small violations might be favored over feasible individuals. Because of the adaptivity, there is no user-defined parameter needed. BIF-EO [50], which modifies SoF, also preserves some good infeasible individuals depending on a user-defined coefficient α . The study obtained that $\alpha = 0.6$ provides the most robust performance for the experimented problems. There is also Ho and Shimizu technique [51] which defines the fitness value based on the separate ranks of objective values, sum of constraint violation values, and number of constraints violated. Then, there is balanced ranking method (BRM) [52] which ranks the queue of feasible individuals and queue of infeasible individuals separately, then merge them. The ranks of feasible queue are unchanged, while the ranks of infeasible queue changes based on the adaptive expression of the penalty function. There is also an extended version of BRM called E-BRM [53].

Even though some of the aforementioned CHTs have considered balanced search in

feasible and infeasible regions, they still do not consider the difference in orders of magnitudes and/or units because they assess the constraints by summing up all of the constraint violations. Due to the difference in orders of magnitudes and/or units in the constraints, summing up the constraint violations can generate imbalanced assessment between the constraints since the constraints with bigger orders of magnitudes will dominate the summed value. Consequently, the search to satisfy the constraints will focus on the constraints with larger orders of magnitudes. One can argue that the constraint violations can be normalized with the currently found extreme values before summing them up to provide more balance assessment between the constraints. However, because the real extreme values are difficult to obtain, this practice might lead to poor representation of the constraints. Also, the investigations of the aforementioned CHTs' performances are mostly on artificial benchmark problems with small number of constraints. There is still scarce, if any, research regarding their performances in solving severely-constrained engineering design optimization problems.

A recent technique called multiple constraint ranking (MCR) balances the assessment of constraints with different order of magnitude because it builds separate queues on the constraint violation values and then sums them up instead of directly sums up the real constraint violation values. It also balances the search from both feasible and infeasible regions when the number of constraints is small, which make it a good candidate for CHT on engineering design optimization. However, it still has one concern. In case of many-constrained problem, the sum of constraint violation ranks will be much larger than the objective value rank, which will cause the search to explore too much of feasible region because it favors feasible individuals over infeasible individuals.

1.3 Research Objectives and Scopes

Based on the literature study, the capability of MCR in alleviating the difference of orders of magnitudes and/or units probably makes it the most suitable CHT for engineering design optimization compared with other aforementioned CHTs. Therefore, we decide to select MCR as the base in devising more suitable CHT for solving engineering design optimization problem with EA. We formulate the objectives of the present work as follows:

1. **To propose modifications of MCR to improve its balance of search in feasible and infeasible regions.**

In principle, we propose to modify MCR based on several principles in order to reduce the proportion of the constraint components in case many constraints are involved. We conduct structured modifications of MCR to obtain the insights of the effect of each modification.

2. To compare the performance of the modifications with MCR and other CHTs.

We compare the performances of the CHTs in various types of problems such as artificial benchmark problems and real-world design problems. However, in analyzing the effect of the modifications, we focus on the optimization results on real-world design problems.

3. To investigate the performance of some best modifications in solving aerodynamic airfoil design optimization problems.

We select some best proposed CHTs and compare the performances with MCR and some other CHTs to obtain better insight of how different CHTs affecting aerodynamic airfoil design optimization problems.

In the present work, we assume a minimization for all experiment. We limit our scope to only solving single-objective optimization problems with moderate solution evaluation budget (up to 30,000 evaluations). This number is based on the recommended budget on car structure design problem [11]. It is considerably smaller than the typical evaluation budget used in the previous works which might be up to 500,000 evaluations or more. Since typical engineering design problems are often computationally expensive, i.e., requiring a lot of time to evaluate one solution, conducting 500,000 evaluations or more might be prohibitive in practice even with supercomputer. Therefore, we decide on 30,000 evaluations as the maximum evaluation budget in the present work. On some problems, we set the evaluation budget even smaller than 30,000 because it is still prohibitive. Because we focus on the performance comparison of CHTs, we utilize standard real genetic algorithm (RGA) with elitist-scheme [39] for all experiments.

1.4 Contribution of This Work

The present work contributes to knowledge development in two kinds of research areas. First is on the general research related to CHT for EA, where the contribution can be summarized as follows:

- Investigation on engineering design problems.

The present work investigates the performances of CHTs not only on artificial benchmark problems, but also on some real-world problems. While widely used in the previous works, the difficulty level of artificial benchmark problems might not reflect the difficulty level of real-world problems. They can be either too easy (the optimum solution is immediately found) or too difficult (no feasible solution is found

during the optimization) to solve. This causes the performance of CHTs obtained on artificial benchmark problems might not be consistent with the performance on real-world problems. Therefore, it is necessary to directly investigate the performance of those CHTs in real-world problems.

- Detailed observation and analysis on the effect of the modifications

Many of the previous works analyze the performances of the CHTs only based on the statistical results (mean, standard deviation/ variance, best, median, and worst values) obtained from multiple independent runs. In the present work, other than comparing the statistical results, we also visualize some history graphs and quantification for the sake of better understanding of the effect of the proposed modifications.

- Proposal of what kind of infeasible individual necessary for more efficient optimization.

In our observation, we obtain some aspects such as what kind of infeasible individual to be preserved, the diversity of infeasible individuals' spreading, and how many feasible and infeasible individuals to be preserved in the population which might help improving the convergence performance by the end of optimization. These aspects have never been observed in the previous works.

In aerospace engineering, especially in the research related to aerodynamic design optimization, EA is usually utilized as a tool to obtain optimized design. To the best of our knowledge, there is no previous work that extensively analyzes the performances of different CHTs in the aerodynamic design optimization with EA, let alone analyzes the effect of balancing the search from both feasible and infeasible regions. Therefore, the contributions of the present work in aerodynamic design optimization are:

- The first to investigate the performances of different CHTs in the optimization.
- The first to analyze the necessity of balancing the search from both feasible and infeasible regions for more efficient optimization.

Of course, the contributions from the general research related to CHTs in EA is also applicable as a part of the contributions in aerodynamic design optimization. The contribution of the present work can be used as a stepping stone in understanding the effect of CHT and the necessity of infeasible region exploration on more complex aerodynamic design optimization such as wing or even aircraft shape design.

1.5 Thesis Outline

The remainder of the present work is as follows,

- Chapter 2

In this chapter, we provide fundamental theories which are necessary for the present work, comprising the theory of EA (especially RGA with elitist scheme) and the types of CHTs.

- Chapter 3

In this chapter, we propose some modifications to improve the convergence performance of MCR by enhancing the infeasible region exploration. We conduct experiment on one real-world benchmark problem (car structure design [11]) to obtain detailed observation of the performance results and the effect of the modifications. In summary, we obtain that some modifications produce significant convergence improvement but they might explore too much of infeasible region.

- Chapter 4

In this chapter, we propose further modifications of MCR to improve the balance between the search in feasible region and search in infeasible region. Now, we conduct experiments on various types of problems as well as compare the performance of the proposed modifications with other types of CHTs such as SoF, SR, GCR, and some more. We obtain that some modifications produce more robust performance compared with the modifications in the previous chapter. In analyzing the effect of the modifications, we focus on the results of optimization on real-world design problems.

- Chapter 5

In this chapter, we investigate the performances of some best CHTs selected from the previous chapter in optimizing two kinds of aerodynamic airfoil design optimizations: transonic airfoil design optimization and multi-point airfoil design optimization.

- Chapter 6

In this chapter, we conclude the present work, emphasize the knowledge obtained by this work, and recommend some future works.

Chapter 2

Theory

2.1 Introduction

In this chapter, we explain the fundamental theories of methods and algorithms required for the experiment and analysis. That includes the theory of EA, especially RGA with elitist scheme, for the optimizer. Then, we explain about types of CHTs that we consider to compare with our proposed CHTs.

2.2 Evolutionary Algorithm

As briefly explained in Chapter 1, EA is an optimization algorithm based on the principle of evolution. There are some types of EA such as genetic algorithm (GA), differential evolution (DE), and evolutionary strategies (ES). Despite the various types of EA, they work with similar operator such as parental selection, crossover/recombination, and mutation. In general, the framework of EA is as follows,

Step 1: Initialize a parent population of N_{pop} individuals randomly.

Step 2: Evaluate the fitness value F of each individual in the parent population.

Step 3: Compute a stopping criterion. If it is met, then stop, else, go to Step 4.

Step 4: Conduct parental selection, crossover, and mutation in parent population and generate offspring population with the size of N_{pop} .

Step 5: Evaluate the fitness value F of each individual in the offspring population.

Step 6: Conduct environmental selection to obtain new parent population for the next generation, then go to step 3.

In Step 1, the individuals in the parent population are generated randomly. In the GA proposed by Goldberg [54], each variable in the individual is represented with "gene" consisting of binary numbers 1 or 0. In order to calculate the objective and constraint values, the individual needs to be decoded into a real number. Suppose an individual $\mathbf{x} = \{x_1, \dots, x_d\}$ with d as the number of variables, and one variable x_j , $j = 1, \dots, d$ is represented by N_{gene} number of genes, $x_j = \{b_1, \dots, b_{N_{gene}}\}$, the individual can be decoded into a real number (normalized between zero and one) with the following formulation [55],

$$x_{j,norm} = 2^{-(t+1)} + \sum_{t=1}^{N_{gene}} b_t 2^{-t} \quad (2.1)$$

Then, in order to calculate the objective and constraint values, $x_{j,norm}$ is first denormalized according to upper and lower bounds of each variable,

$$x_{j,denorm} = x_{j,L} + (x_{j,U} - x_{j,L})x_{j,norm} \quad (2.2)$$

where $x_{j,L}$ and $x_{j,U}$ denote the lower and upper bounds of the j^{th} variable, respectively. The objective and constraint values then can be evaluated as follows,

$$f(\mathbf{x}) = f(\mathbf{x}_{denorm}) \quad (2.3)$$

and

$$g_i(\mathbf{x}) = g_i(\mathbf{x}_{denorm}) \quad (2.4)$$

where $\mathbf{x}_{denorm} = \{x_{j,denorm}, \dots, x_{d,denorm}\}$.

In GA with binary representation (or binary-coded GA), the accuracy of each variable depends on N_{gene} . Basically, the larger N_{gene} , the better the accuracy. However, since it is not trivial to select the most suitable N_{gene} , GA with real number representation, or real-coded GA, is more common to be utilized. In real-coded GA, each variable is represented directly in real number. Therefore, in case of Step 1, if binary-coded GA generates random sequence of binary numbers for one variable, real-coded GA only needs to generate one real number normalized between zero and one for one variable.

In Step 2, fitness value F of an individual is a value assessed in EA in order to do the parental and environmental selections. In case of an unconstrained optimization problem, an individual's fitness value $F(\mathbf{x})$ is equivalent with its objective value $f(\mathbf{x})$,

$$F(\mathbf{x}) = f(\mathbf{x}) \quad (2.5)$$

. In case of a constrained optimization problem, assuming the EA adopts a penalty function CHT, the individual's fitness value is usually a combination of its objective and constraint value representations, which, in principle, is expressed as follows,

$$F(\mathbf{x}) = f_r(\mathbf{x}) + p(\mathbf{x}) \quad (2.6)$$

where $f_r(\mathbf{x})$ is a value which represents the search towards unconstrained optimum and $p(\mathbf{x})$ is the penalty function, which represents the search towards constraint satisfaction. The definitions of $f_r(\mathbf{x})$ and $p(\mathbf{x})$ vary depending on the CHTs. For $f_r(\mathbf{x})$, some CHTs define them with the real objective value $f(\mathbf{x})$, while some with the rank of the objective value. For $p(\mathbf{x})$, it is usually built from the constraint violations of all constraint functions. The i^{th} constraint violation ν_i is expressed as follows,

$$\nu_i(\mathbf{x}) = \max(0, g_i(\mathbf{x})) \quad (2.7)$$

, meaning that an individual \mathbf{x} is defined violating the i^{th} constraint if $g_i(\mathbf{x}) > 0$. There are also various ways in formulating the $p(\mathbf{x})$. Some CHTs define the $p(\mathbf{x})$ based on the real constraint violation values, some based on the rank of the real constraint violation values. Some even include the rank of the number of constraints violated. We will explain in more detail about how $f_r(\mathbf{x})$ and $p(\mathbf{x})$ of some types of CHTs work in solving a constrained optimization problem in the next subsection.

Step 3 is a decision making process to finish the optimization. According to [55], there are several types of stopping criteria in GA:

- Correct answer

The algorithm stops when a correct or an acceptable solution is obtained.

- No improvement

The algorithm stops when there is no convergence improvement of in several generations.

- Statistics

The algorithm stops when the mean or standard deviation of the fitness values in the population reaches a certain level. This means that the values are no longer changing.

- Number of iterations/ generations

If the algorithm do not stop from one of the previous criteria, limit the number of iterations/generations or the algorithm will run forever.

- Local optimizer

If there is no improvement from the result of local optimization, stop the algorithm.

In engineering design optimization, it is common that the optimization is limited by some amount of time because the evaluations of the objective and constraint functions are

quite time-consuming. Therefore, stopping criterion based on the number of iterations/generations is the most common to utilize.

Step 4 is a process of obtaining new individuals (offspring). In general, we expect the offspring individuals to have better fitness values than the parent individuals for the algorithm to converge to the global optimum. There are three attempts in achieving that purpose: parental selection, crossover/ recombination, mutation.

Parental selection is a process to select some individuals fitting for mating and producing offspring. Some common parental selection methods are roulette wheel and tournament selection methods. Roulette wheel method selects the parents by generating a probability based on the individual's fitness value in such way that the ones which have better fitness value will have higher probability to be selected as parents. The procedure of roulette wheel is as follows,

1. normalize the fitness values of the individuals in parent population with the maximum and minimum fitness values found in the current generation,
2. sort the parent population based on the fitness values in descending order,
3. compute accumulated fitness value for each individual, where accumulated fitness value is the sum of fitness value of the current individual with the fitness values of all previous individuals,
4. generate a random number between zero and one, and
5. select an individual as a parent if its accumulated fitness function is the first one greater than the random number.

In tournament selection method, the selection process is conducted by competing k selected individuals' fitness values in the current parent population. The fittest individual (the one with the best fitness value) is selected as a parent. The procedure is as follows,

1. select k random individuals from the current parent population, and
2. put the best individual in a matingpool.

We repeat the above steps until there are N_{pop} individuals in the matingpool. The individuals in the matingpool will be the parents for crossover process. In the present work, we utilize binary tournament selection, which is tournament selection with $k = 2$, for all experiments.

After all parents are selected, EA moves to crossover process. This is the most important step in EA to generate new individuals with better fitness values than the old individuals. In canonical EA, a pair of parents are usually selected from the mating pool

to allow them producing a pair of offspring by a crossover method such as simulated binary crossover (SBX) [56] or blend crossover (BLX) [57]. This process is repeated until there is a population of offspring with size of N_{pop} . In the present work, we utilize SBX. Suppose $\mathbf{x}_1 = \{x_{1,1}, \dots, x_{d,1}\}$ and $\mathbf{x}_2 = \{x_{1,2}, \dots, x_{d,2}\}$ are a pair of parents, SBX generates a pair of offspring $\mathbf{c}_1 = \{c_{1,1}, \dots, c_{d,1}\}$ and $\mathbf{c}_2 = \{c_{1,2}, \dots, c_{d,2}\}$ from \mathbf{x}_1 and \mathbf{x}_2 with the following expression,

$$c_{j,1} = 0.5(x_{j,1} + x_{j,2} - \bar{\beta}|x_{j,2} - x_{j,1}|) \quad (2.8)$$

$$c_{j,2} = 0.5(x_{j,1} + x_{j,2} + \bar{\beta}|x_{j,2} - x_{j,1}|) \quad (2.9)$$

where

$$\bar{\beta} = \begin{cases} (\alpha u)^{\frac{1}{\eta_c+1}}, & \text{if } u \leq \frac{1}{\alpha}, \\ (\frac{1}{2-\alpha u})^{\frac{1}{\eta_c+1}}, & \text{otherwise.} \end{cases} \quad (2.10)$$

where $\alpha = 2 - \beta^{-(\eta_c+1)}$, and β is calculated as follows,

$$\beta = 1 + \frac{2}{x_{j,2} - x_{j,1}} \min(x_{j,1} - x_{j,L}, x_{j,U} - x_{j,2}). \quad (2.11)$$

It is assumed that $x_{j,1} < x_{j,2}$ here. The symbols u and η_c denote a random value and distribution index, respectively. In this study, we set $\eta_c = 20$ in all experiment.

During the search, the new individuals generated by crossover might gather in a certain region. This makes the search might get trapped in the local optimum of that certain region. In order to alleviate that phenomenon, mutation is a necessary process to maintain the diversity of population, so that there is a higher chance for the search algorithm to escape from local optima and find the right region to reach the global optimum. Mutation works by occasionally giving slight change in the obtained new individuals from crossover. In the present study, we utilize polynomial mutation [58] as the mutation method. It mutates a new individual $\mathbf{c} = \{c_1, \dots, c_d\}$ into $\mathbf{c}' = \{c'_1, \dots, c'_d\}$ with the following expression,

$$c'_j = c_j + \bar{\delta}(x_{j,U} - x_{j,L}) \quad (2.12)$$

where

$$\bar{\delta} = \begin{cases} (2u + (1 - 2u)(1 - \delta)^{\eta_m+1})^{\frac{1}{\eta_m+1}} - 1, & \text{if } u \leq 0.5, \\ 1 - (2(1 - u) + 2(u - 0.5)(1 - \delta)^{\eta_m+1})^{\frac{1}{\eta_m+1}}, & \text{otherwise.} \end{cases} \quad (2.13)$$

where η_m denotes the mutation distribution index. In this study, we set the distribution index $\eta_m = 20$ for all experiments.

In Step 5, the fitness value each individual in the (mutated) offspring population is calculated with Eq. 2.5 in case of unconstrained problem or Eq. 2.6 in case of a constrained problem. In Step 6, we conduct environmental selection to obtain new parent population for the next generation. In general, there are two types of environmental selection, the

non-elitist and elitist schemes. In non-elitist scheme, the obtained offspring population are to be the new parent population for the next generation. Meanwhile in elitist scheme, the new parent population is built by selecting N_{pop} individuals with the best $F(\mathbf{x})$ s from the combined parent and offspring populations.

2.3 Existing Constraint Handling Technique for EA

In this subsection, we explain in more detail about some existing CHTs which are relevant in this work, especially about how they set their $f_r(\mathbf{x})$ and $p(\mathbf{x})$ in solving a constrained problem. As mentioned in previous subsection, the CHTs define their $f_r(\mathbf{x})$ and $p(\mathbf{x})$ in various ways. As for $f_r(\mathbf{x})$, some define it with the real objective value, others with the rank of objective value queue. As for the penalty function $p(\mathbf{x})$, most CHTs define it based on the sum of constraint violations, expressed as follows,

$$CV(\mathbf{x}) = \sum_{i=1}^m \nu_i(\mathbf{x})^\beta \quad (2.14)$$

where $\nu_i(\mathbf{x})$ is constraint violation that follows Eq. 2.7. The value of β is often set to one or two. By summing the constraint violations up, the CHTs do not consider the difference of magnitude and/or units in the assessment of constraint because the value of CV will be dominated by the constraint violations with higher orders of magnitudes.

In general, we can classify the CHTs into four categories:

1. Category A:

In this category, the CHTs consider feasible individuals better than infeasible individuals.

2. Category B:

In this category, the CHTs consider balanced search from both feasible and infeasible regions. However, they have not considered to alleviate the difference of order of magnitudes which might be possessed by the optimization problems because the assessment of constraints is still based on the sum of constraint violations.

3. Category C:

This category is basically MCR, which considers to alleviate the difference of order of magnitudes which might be possessed by the optimization problems. MCR also considers balanced search from both feasible and infeasible regions in case of small number of constraints.

4. Category D:

This category is for the proposed CHTs which consider not only alleviation of difference of order of magnitudes in the constraints, but also balanced search from both feasible and infeasible regions regardless of the number of constraints. We elaborate the making of this category of CHTs in Chapters 3 and 4.

2.3.1 Category A

We can say that the CHTs in this category is conventional because they tend to favor feasible individuals over infeasible individuals. Some CHTs comprise this category are death penalty, and superiority of feasible solution (SoF). In death penalty [59], the infeasible individuals are given the worst fitness values or simply eliminated from the search process. In the case of severely constrained problems, whose feasible individuals are difficult to obtain, utilization of death penalty is quite challenging because it needs feasible individual to work.

SoF [39] is one of the most popular conventional CHTs through this line of research. SoF provides ease of implementation and it does not need any user-defined parameters. SoF defines the fitness value with the following expression,

$$F(\mathbf{x}) = \begin{cases} f(\mathbf{x}), & \text{if } CV(\mathbf{x}) = 0, \\ f_{worst} + CV(\mathbf{x}), & \text{otherwise.} \end{cases} \quad (2.15)$$

where f_{worst} is the worst objective value of feasible individual in the current population. In case there is no feasible individual, f_{worst} is set to zero. Eq. 2.15 is based on a tournament of two arbitrary individuals in current population, which follows the following considerations

- if two individuals are infeasible, the one with smaller CV is better,
- if one individual is feasible and the other is infeasible, the feasible one is better, and
- if two individuals are feasible, the one with smaller objective value (in case of minimization) is better.

2.3.2 Category B

In this category, the CHTs tend to be recently new. This category comprises stochastic ranking (SR) [40], global competitive ranking (GCR) [48], Ho and Shimizu methods (HnS) [51], adaptive penalty function (APM) [49], balanced infeasible-feasible evolutionary optimization (BIF-EO) [50], and extended balanced constrained ranking (E-BRM) [53].

a. Stochastic Ranking

Stochastic ranking (SR) is also one of the most popular CHTs. SR is probably the first CHT implementing rank system in assessing the objective and constraint functions. In SR, the fitness value is defined as the final rank returned by Algorithm 1.

Algorithm 1: Stochastic bubble sort

```

for  $i = 1$  to  $N_{pop}$  do
    for  $j = 1$  to  $N_{pop} - 1$  do
        sample  $u \in U(0, 1)$  ;
        if  $(CV(\mathbf{x}_j) = CV(\mathbf{x}_{j+1}) = 0)$  or  $(u < P_f)$  then
            if  $(f(\mathbf{x}_j) > f(\mathbf{x}_{j+1}))$  then
                swap( $\mathbf{x}_j, \mathbf{x}_{j+1}$ ) ;
            end
        else
            if  $CV(\mathbf{x}_j) > CV(\mathbf{x}_{j+1})$  then
                swap( $\mathbf{x}_j, \mathbf{x}_{j+1}$ ) ;
            end
        end
    end
end

```

Despite its unique form, it is still valid to put SR in penalty function-based CHT because SR is similar with SoF in some sense. Between two arbitrary individuals, if both individuals are feasible, SR also favors the one with better objective value. If one individual is feasible while the other is infeasible, SR also favors the feasible one. The difference with SoF is that if both individuals are infeasible, SR favors either the one with smaller CV or the one with better objective value, depending on a user-defined probability P_f . The probability P_f is a measure to control the balance of the search in both feasible and infeasible region. In order to increase the chance of obtaining feasible individuals, the value $0 \leq P_f < 0.5$ is recommended. The literature suggests $0.4 \leq P_f < 0.5$ for balanced search in both feasible and infeasible regions.

b. Global Competitive Ranking (GCR)

Global competitive ranking (GCR) is an improvement of SR by a modification of its ranking system. GCR defines the fitness value by building a separate rank for objective values and constraint violations, expressed as follows,

$$F(\mathbf{x}) = P_f \frac{R_f}{N_{pop} - 1} + (1 - P_f) \frac{R_{CV}}{N_{pop} - 1} \quad (2.16)$$

where R denotes the rank for the respective subscripts. R_f stands for the rank built by the queue of the individuals' objective values, while R_{CV} is for the rank built by the queue of individuals' CVs. For each subscript, the rank of an individual is built based on the number of losses it obtain when its subscript's value is compared with other individuals in current population. For example, if there are six individuals with the following objective values $\{f(\mathbf{x}_1), f(\mathbf{x}_2), f(\mathbf{x}_3), f(\mathbf{x}_4), f(\mathbf{x}_5), f(\mathbf{x}_6)\} = \{0.3, 0.6, 3.3, 3.3, 4, 5.6\}$, the R_f s of those individuals are $\{R_{f_1}, R_{f_2}, R_{f_3}, R_{f_4}, R_{f_5}, R_{f_6}\} = \{0, 1, 2, 2, 4, 5\}$. We can notice the same R_f values for the third and fourth individuals because both objective values lose against two other individuals (the first and second individuals). Similar with SR, P_f in GCR is a user-defined parameter to control the balance of the search from both feasible and infeasible regions.

c. Adaptive Penalty Method

In adaptive penalty function (APM), the fitness value is expressed as follows

$$F(\mathbf{x}) = \begin{cases} f(\mathbf{x}), & \text{if } CV(\mathbf{x}) = 0, \\ \bar{f} + \sum_{i=1}^m k_i \nu_i(\mathbf{x}), & \text{otherwise.} \end{cases} \quad (2.17)$$

where

$$\bar{f} = \begin{cases} f(\mathbf{x}), & \text{if } f(\mathbf{x}) > \langle f \rangle, \\ \langle f \rangle, & \text{otherwise.} \end{cases} \quad (2.18)$$

and $\langle f \rangle$ is the average of objective values in the current population. The coefficient k_i is expressed as follows,

$$k_i = \frac{|\sum_{j=1}^{N_{pop}} f(\mathbf{x}_i)|}{\sum_{l=1}^m [\sum_{j=1}^{N_{pop}} \nu_l(\mathbf{x}_j)]^2} \sum_{j=1}^{N_{pop}} \nu_i(\mathbf{x}_j) \quad (2.19)$$

The idea of APM is the values of penalty coefficients should be distributed in a way that those with more difficult-to-solve constraints should have a relatively higher penalty coefficients. APM does not require any user-defined parameters.

d. Ho and Shimizu technique

Ho and Shimizu technique (HnS) also adopts ranking system in defining the fitness value. The fitness value is expressed as follows,

$$F(\mathbf{x}) = \begin{cases} R_f + R_{N_\nu} + R_{CV}, & \text{if } N_{feas} > 0, \\ R_{N_\nu} + R_{CV}, & \text{otherwise.} \end{cases} \quad (2.20)$$

where R_{N_ν} denotes a rank based on the number of constraints violated. The ranking system of each subscript is the same with GCR's. N_{feas} denotes the number of feasible individuals exists in the current population. If feasible individual exists ($N_{feas} > 0$), HnS

formulates the fitness value based on R_f , R_{N_v} , and R_{CV} . Otherwise, HnS focuses only on finding feasible individual by formulating the fitness value only based on R_{N_v} and R_{CV} .

e. **Balanced Infeasibility-Feasibility-based Evolutionary Optimization (BIF-EO)**

BIF-EO is a modification of SoF in the attempt to improve the balancing of search in both feasible and infeasible regions. The idea is by allowing some worse feasible individuals (having large objective values) to be worst than some infeasible individuals having small CV s.

$$F(\mathbf{x}) = \begin{cases} f(\mathbf{x}), & \text{if } CV(\mathbf{x}) = 0, \\ \frac{f_{worst}}{f_{best}} + \overline{CV} + f_m - f_{worst}, & \text{otherwise.} \end{cases} \quad (2.21)$$

where

$$f_m = f_{worst} - \alpha(f_{worst} - f_{best}) \quad (2.22)$$

$$\overline{CV} = \kappa(CV(\mathbf{x}) - CV_{best}) + f_{best} \quad (2.23)$$

$$\kappa = \begin{cases} 1, & \text{if } f_{best} = f_{worst} \text{ or, if } CV_{best} = CV_{worst}, \\ \frac{f_{worst} - f_{best}}{CV_{worst} - CV_{best}}, & \text{otherwise.} \end{cases} \quad (2.24)$$

The parameters f_{worst} and f_{best} are the worst and the best objective values found in current population, respectively. CV_{worst} and CV_{best} denote the worst and the best CV values found in current population. In this technique, α is a user-defined coefficient which controls the balance of the search in both feasible and infeasible regions, ranging between zero and one. In the literature, $\alpha = 0.6$ provides the most robust performance.

f. **Extended Balanced Ranking Method (E-BRM) [53]**

In E-BRM, the fitness value of an individual is given by

$$F(\mathbf{x}) = \begin{cases} rank(f(\mathbf{x}), FCS), & \text{if } \mathbf{x} \text{ is feasible,} \\ rankWeighted(\mathbf{x}, \sigma) + \sqrt{\delta} + \Psi, & \text{otherwise.} \end{cases} \quad (2.25)$$

where $rank(f(\mathbf{x}), FCS)$ is the rank of feasible candidate solution (FCS) in the current population based on the objective values, ordered from the lowest to the highest objective values in case of minimization problem. As for $rankWeighted$, it is defined as

$$rankWeighted(\mathbf{x}, \sigma) = \frac{rank(p(\mathbf{x}), ICS)(1 - f(\sigma)) + rank(f(), ICS)f(\sigma)}{2} \quad (2.26)$$

where $rank(p(\mathbf{x}), ICS)$ and $rank(f(\mathbf{x}), ICS)$ are the ranks of infeasible solution based on penalty violations $p(\mathbf{x})$ and objective value $f(\mathbf{x})$, respectively, among ICS population. $f(\sigma)$ works in weighting the two ICS ranks (violations and objective values), into

evaluation procedure. The penalty violation $p(\mathbf{x})$ is defined as

$$p(\mathbf{x}) = \sum_{i=1}^m (\nu_i(\mathbf{x}))^\beta \quad (2.27)$$

where

$$\beta = 2 + (1 - f(\sigma)) \quad (2.28)$$

where $f(\sigma)$ is a function that balances the penalty according to the amount of constraints not violated in the population, in order to direct the search toward the feasibility space. This function is defined as

$$f(\sigma) = \begin{cases} 0, & \text{if } count(FCS) = 0, \\ \sigma, & \text{otherwise.} \end{cases} \quad (2.29)$$

where σ , ranging from $[0,1]$, is given by

$$\sigma = \frac{\hat{C}}{mN_{pop}} \quad (2.30)$$

\hat{C} is the total amount of constraints not violated in the population at the present generation.

Then, the merging parameters $\sqrt{\delta}$ and Ψ are defined as

$$\sqrt{\delta} = \sqrt{(count(FCS) \frac{count(ICS)}{N_{pop}})} \quad (2.31)$$

$$\Psi = \begin{cases} 0, & \text{if } (count(ICS) + \sqrt{\delta}) > count(FCS), \\ \frac{\psi}{count(ICS)-1} (rank(p(\mathbf{x}), ICS) - 1), & \text{otherwise.} \end{cases} \quad (2.32)$$

where

$$\psi = count(FCS) - (count(ICS) + \sqrt{\delta}). \quad (2.33)$$

2.3.3 Category C: Multiple Constraint Ranking

Multiple constraint ranking (MCR) [60] defines the fitness value $F(\mathbf{x})$ of each individual in the current population based on the ranks built by separate queues of the objective value, number of constraints violated, and constraint violation on each constraint, expressed as follows,

$$F(\mathbf{x}) = \begin{cases} R_f + R_{N_\nu} + \sum_{i=1}^m R_{\nu_i}, & \text{if } N_{feas} > 0, \\ R_{N_\nu} + \sum_{i=1}^m R_{\nu_i}, & \text{otherwise.} \end{cases} \quad (2.34)$$

The definition of rank R for each subscript is the same with GCR's. MCR does not require any user-defined parameters. If feasible individual exists ($N_{feas} > 0$) in the population,

MCR formulates the fitness value based on R_f , R_{N_ν} , and R_{ν_i} . Otherwise, MCR focuses only on finding feasible individual by formulating the fitness value only based on R_{N_ν} and R_{ν_i} . The difference with GCR and HnS is that MCR utilizes $\sum_{i=1}^m R_{\nu_i}$ in assessing the constraint violation instead of R_{CV} . R_{ν_i} denotes the rank of the constraint violation value of the i^{th} constraint function.

By building separate rank for each constraint, MCR alleviates the imbalanced constraint assessment which might appear from difference of order of magnitudes and/or units. When the number of constraints is small, MCR also balances the search in both feasible and infeasible regions because the values of R_f , R_{N_ν} , and the sum of R_{ν_i} will be of about the same order. When the number of constraints is large, however, the sum of R_{ν_i} will have much larger order of magnitude compared with R_f . This condition makes MCR tends to search too much of feasible region because it favors feasible individuals over infeasible individuals during the search, similar with conventional CHTs (Category A). Therefore, by involving the search in infeasible region more when the number of constraints is large, we believe that the convergence performance of MCR can still be improved. This is the reason why we propose a new family of CHTs, namely Category D, which considers not only alleviation of difference of order of magnitudes in the constraints, but also balanced search from both feasible and infeasible regions regardless of the number of constraints. We thoroughly elaborate the making of this category of CHTs in Chapters 3 and 4.

Chapter 3

Adaptively Enhancing Infeasible Region Exploration

3.1 Introduction

In this chapter, we propose some modifications of MCR in order to enhance the infeasible region exploration when the number of constraints is large. The idea of modification is by reducing the portion of the constraint components in the fitness value calculation. We conduct some structural modifications based on three principles: adaptive reduction of R_{ν_i} , omission of R_{N_ν} , and proportionation of rank terms. We conduct our experiment only in one real-world design problem to obtain detailed observation about the effect of the modifications to the convergence performance towards constrained optimum.

3.2 Generalized Multiple Constraint Ranking

First of all, we define a generalized version of MCR (abbreviated as G-MCR) for the sake of easier comparison between MCR and the modifications, expressed as follows,

$$F = \beta_1 R_f + \beta_2 (\eta R_{N_\nu} + \gamma \sum_{i=1}^m \alpha_i R_{\nu_i}). \quad (3.1)$$

The coefficients β_1 and β_2 control the weight in finding unconstrained optimum and satisfying overall constraint components, respectively. Coefficient α_i controls the weight in satisfying the i^{th} constraint, and γ proportionates the summed value of R_{ν_i} with R_f and R_{N_ν} . Lastly, coefficient η controls the weight in minimizing the number of constraints violated. Based on this generalized form, MCR has $\beta_1 = 1$ if $N_{feas} > 0$ and $\beta_1 = 0$ otherwise, $\beta_2 = 1$, $\eta = 1$, $\gamma = 1$, and $\alpha_i = 1$ for $i = 1, \dots, m$.

3.3 Adaptively Enhancing Infeasible Region Exploration

As mentioned in previous chapter, MCR produces imbalanced search because it tends to favor feasible individuals over infeasible individuals in case of many-constrained case. In order to enhance the capability of MCR in exploring infeasible region when the number of constraints is large, we propose some modifications based on the following principles:

- 1st principle

In order to keep the balance of search in both feasible and infeasible regions, the sum of R_{ν_i} needs to be proportionated with R_f and R_{N_ν} . This way, all rank terms (R_f , R_{N_ν} , and the sum of R_{ν_i}) will have the same order of magnitude regardless of the number of constraints.

- 2nd principle

In case of many-constrained problem, there is a possibility that the difficulty level in satisfying each constraint is different. In such case, it is not really necessary to give the same weight on all constraints during the search process. Instead, the easier-to-satisfy constraints can be given smaller weights than the more-difficult-to-satisfy constraints in the formulation of F .

- 3rd principle

The existence of R_{N_ν} might prevent the search from exploring larger infeasible region since it keeps the number of constraints violated small. It might be a good idea to remove it to allow the search assess more infeasible individuals with larger number of constraints violated.

Table 3.1 presents the structured implementation of the proposed modifications to G-MCR in order to gain more understanding of the effect of each modification. In all modifications, we keep the values of β_1 and β_2 the same as MCR's.

In MCR_1, we implement a proportionation based on the number of constraints,

$$\gamma = \frac{1}{m} \quad (3.2)$$

while keeping other coefficients the same with MCR. This modification proportionates the sum of R_{ν_i} with R_f and R_{N_ν} to keep the balance between feasible and infeasible regions regardless of the number of constraints. The consequence of this modification is that the search towards unconstrained optimum and satisfaction of N_ν will be faster when the number of constraints is large because $\frac{1}{m} \sum_{i=1}^m R_{\nu_i}$ is smaller than $\sum_{i=1}^m R_{\nu_i}$. This modification stands as the realization of the 1st principle.

Table 3.1: MCR and its modifications

| CHT | η | γ | α_i |
|-------|--------|-----------------------------------|------------------------------|
| MCR | 1 | 1 | 1 |
| MCR_1 | 1 | $\frac{1}{m}$ | 1 |
| MCR_2 | 1 | 1 | $\frac{N_{viol_i}}{N_{pop}}$ |
| MCR_3 | 1 | $\frac{1}{m}$ | $\frac{N_{viol_i}}{N_{pop}}$ |
| MCR_4 | 1 | $\frac{1}{\sum_{i=1}^m \alpha_i}$ | $\frac{N_{viol_i}}{N_{pop}}$ |
| MCR_5 | 0 | 1 | 1 |
| MCR_6 | 0 | $\frac{1}{m}$ | 1 |
| MCR_7 | 0 | 1 | $\frac{N_{viol_i}}{N_{pop}}$ |
| MCR_8 | 0 | $\frac{1}{m}$ | $\frac{N_{viol_i}}{N_{pop}}$ |
| MCR_9 | 0 | $\frac{1}{\sum_{i=1}^m \alpha_i}$ | $\frac{N_{viol_i}}{N_{pop}}$ |

In MCR_2, we propose a definition of difficulty level in satisfying the constraint,

$$\alpha_i = \frac{N_{viol_i}}{N_{pop}} \quad (3.3)$$

where N_{viol_i} denotes the number of solutions violating the i^{th} constraint ($\nu_i > 0$) in the current population. By this definition, α_i becomes an adaptive value between zero and one. Smaller value of N_{viol_i} indicates that the i^{th} constraint is easier to satisfy compared with other constraints, and vice versa. This proposed modification allows the easier-to-satisfy constraints to obtain smaller weight than the more difficult-to-satisfy constraints during the search. This way, Eq. 3.3 will decrease the proportion of the easier-to-solve R_{ν_i} quite significantly in Eq. (3.1) while only a little for the more difficult-to-solve R_{ν_i} . In consequence, similar with MCR_1, the proportions of R_f and R_{N_ν} become relatively bigger than MCR and thus the search towards unconstrained optimum and satisfaction of N_ν become faster. This modification is a realization of the 2nd principle.

In MCR_3, we implement not only the adaptive α_i from MCR_2 (Eq. 3.3), but also $\gamma = \frac{1}{m}$ (Eq. 3.2) to proportionate the constraint violation rank terms based on the number of constraints. Similarly, in MCR_4, we also implement $\alpha_i = \frac{N_{viol_i}}{N_{pop}}$ and proportionate the constraint violation rank terms. The difference with MCR_3 is that MCR_4 proportionates based on the sum of α_i s with

$$\gamma = \frac{1}{\sum_{i=1}^m \alpha_i} \quad (3.4)$$

instead of Eq. 3.2. MCR_3 and MCR_4 are the realization of both 1st and 2nd principles, only with different ways of proportionation. Proportionating with Eq. 3.4 will make the order of magnitude of $\gamma \sum_{i=1}^m R_{\nu_i}$ always the same with R_f and R_{N_ν} regardless of the

change of adaptive α_i . On the other hand, proportionating based on the number of constraints with Eq. 3.2 might make the portion of $\frac{1}{m} \sum_{i=1}^m R_{\nu_i}$ smaller than other rank terms because $\frac{1}{m}$ tends to be smaller than $\frac{1}{\sum_{i=1}^m \alpha_i}$. Hence, the search towards unconstrained optimum and satisfaction of N_ν on MCR_3 might be faster than MCR_4. Because of their difference, it is worth investigating the effect of both types of γ .

In MCR_5 until MCR_9, we conduct the same structured modifications as conducted in MCR until MCR_4 for γ and α_i . The only difference is that in MCR_5 until MCR_9, we omit R_{N_ν} with $\eta = 0$. MCR_5 is the realization of only the 3rd principle. MCR_6 is the realization of both 1st and 3rd principles. MCR_7 is the realization of both 2nd and 3rd principles. Last but not least, MCR_8 and MCR_9 are the realizations of all three principles, where MCR_8 proportionates the rank terms with Eq. 3.2, while MCR_9 with Eq. 3.4. We propose to omit R_{N_ν} so that the search in infeasible region is not restricted only to infeasible individuals with small N_ν . This way, we allow the search algorithm to explore wider area of infeasible region.

3.4 Experimental Setup

3.4.1 Benchmark problem

In this case, we aim to obtain detailed knowledge of the effect of the proposed modifications. Therefore, we compare the performance of MCR and the modifications only on one benchmark problem, a car structure design optimization [11]. This benchmark problem is a model derived from an actual computationally expensive car structure design optimization [25], which takes around 360 hours of computing time using more than 70,000 cores of supercomputer K [61]. The modeling of the constraints is conducted with radial basis function (RBF). It has 222 design variables, 54 inequality constraints and a very small feasible region (no feasible individual can be found in 1,000,000 random search). The objective and constraint functions are also of different order of magnitude and/or units because they comprise properties such as mass, crashworthiness, body torsional stiffness, and vibration modes. The execution file of this benchmark problem can be downloaded from [62].

Originally, there are two objectives to be optimized: weight minimization and maximization of the number of parts with common thickness. However, we only consider weight minimization as the objective function since we limit our scope to single-objective optimization in the present work. The design variables of the actual optimization are all discrete. However, in the present work, we set them to be continuous for the sake of simplification. Due to the expensive computation in actual optimization, this benchmark problem is to be solved within 30,000 solution evaluations, a considerably small number

compared with typical number of solution evaluations used in CHT investigation, which can reach 500,000 or even more.

3.4.2 Optimization setting

In optimizing the benchmark problem, we utilize real-coded genetic algorithm (RGA) with elitist scheme [39]. To match with the required number of solution evaluations of the benchmark problem, we conduct the optimization with $N_{pop} = 100$ and number of generations of 300. We utilize binary tournament selection as the selection method. We set the probabilities of crossover and mutation to 1 and $\frac{1}{d}$, respectively. As for the crossover and mutation methods, we utilize simulated binary crossover (SBX) [56] and polynomial mutation [58], respectively, with distribution index of 20 for each method. To generate the new parent population, we combine the old parent population and offspring population. The new parent population consists of N_{pop} individuals with the best F s from the combined population.

For each CHT investigation, we conduct 51 independent runs of optimization. Consistent with the objective function of the benchmark problem, we set the optimization into minimization problem. To determine if the convergence performance of MCR and the proposed modifications are significantly different, we conduct Wilcoxon rank-sum test [63] with significance level of 0.05.

3.5 Result and Discussion

3.5.1 Overall result

In this subsection, we discuss about the overall convergence performance of all CHTs. Fig. 3.1 depicts the box plot of the obtained optimum values of all CHTs mentioned in Table 3.1. On the box plot of each CHT, we can observe five statistical values from the optimum value distribution: median (50th percentile), Q1 (25th percentile), Q3 (75th percentile), minimum ($Q1 - 1.5 \times IQR$), and maximum ($Q3 + 1.5 \times IQR$), where IQR denotes the interquartile range ($Q3 - Q1$). Since we set the optimization to minimization, we consider the minimum value as the best result, and the maximum value as the worst. We specify the values below the minimum or above the maximum as outliers. MCR_7, MCR_9, and MCR_8 obtain superior performances with MCR_7 as the best performer in terms of worst, median, and best optimum values compared with MCR and other modifications.

To check the significance of the performances presented in Fig. 3.1, we conduct Wilcoxon rank-sum test with significance level of 0.05. Tables 3.2 and 3.3 present the significance test results of one CHT against other CHTs. Suppose we compare two arbitrary tech-

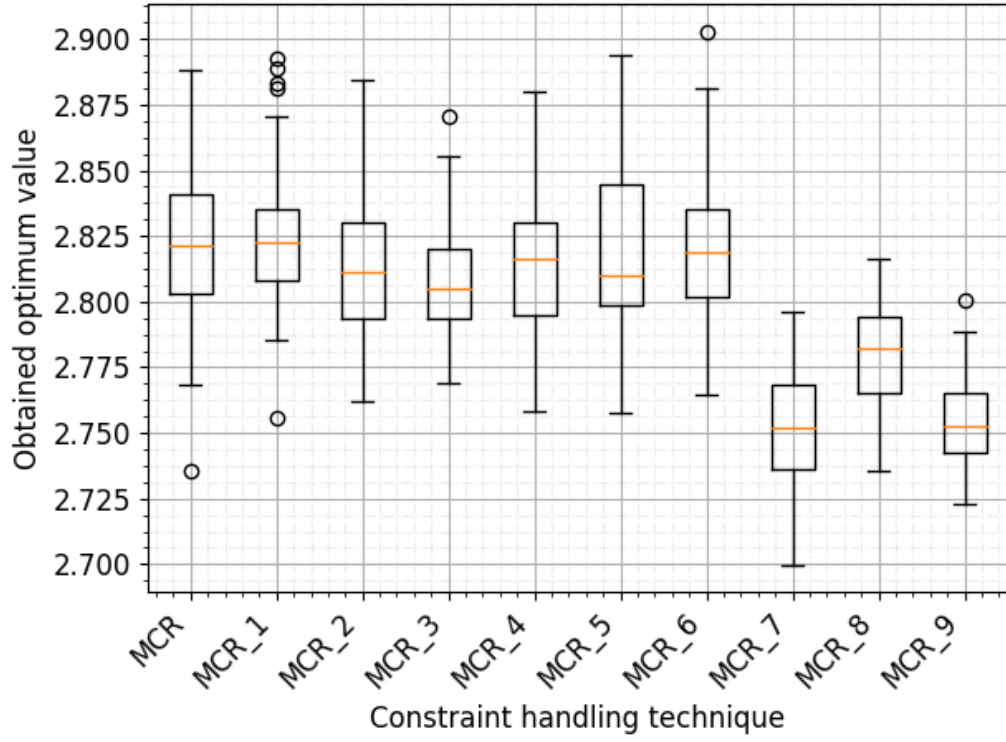


Figure 3.1: Box plot of obtained optimum values

Table 3.2: Statistical significance test (1/2)

| CHT | MCR | MCR_1 | MCR_2 | MCR_3 | MCR_4 |
|-----------|-----|-------|-------|-------|-------|
| vs. MCR | | = | + | + | = |
| vs. MCR_1 | = | | + | + | = |
| vs. MCR_2 | - | - | | = | = |
| vs. MCR_3 | - | - | = | | - |
| vs. MCR_4 | = | = | = | + | |
| vs. MCR_5 | = | = | = | + | = |
| vs. MCR_6 | = | = | = | + | = |
| vs. MCR_7 | - | - | - | - | - |
| vs. MCR_8 | - | - | - | - | - |
| vs. MCR_9 | - | - | - | - | - |

Table 3.3: Statistical significance test (2/2)

| CHT | MCR_5 | MCR_6 | MCR_7 | MCR_8 | MCR_9 |
|-----------|-------|-------|-------|-------|-------|
| vs. MCR | = | = | + | + | + |
| vs. MCR_1 | = | = | + | + | + |
| vs. MCR_2 | = | = | + | + | + |
| vs. MCR_3 | - | - | + | + | + |
| vs. MCR_4 | = | = | + | + | + |
| vs. MCR_5 | | = | + | + | + |
| vs. MCR_6 | = | | + | + | + |
| vs. MCR_7 | - | - | | - | = |
| vs. MCR_8 | - | - | + | | + |
| vs. MCR_9 | - | - | = | - | |

niques, namely CHT_1 vs. CHT_2, the "+", "-", and "=" signs indicate that the performance of CHT_1 is significantly superior, significantly inferior, and competitive, respectively, in comparison with CHT_2.

MCR_7 and MCR_9 are competing as the best performers by obtaining eight "+" signs, followed by MCR_8 by obtaining seven "+" signs. Comparing with MCR, we can also notice that MCR_2 and MCR_3 are significantly superior than MCR. These results show that some modifications provide improvement in handling the constraints and thus leading the search towards better constrained optimum compared with MCR.

3.5.2 The effect of the modifications

Based on the result from the previous subsection, we can notice that the three best CHTs which produce significantly better convergence performance against MCR (MCR_7, MCR_8, MCR_9) implement at least two of the proposed modifications: the adaptive α_i and the omission of R_{N_ν} . As for MCR_8, it also implements $\gamma = \frac{1}{m}$, while MCR_9 also implements $\gamma = \frac{1}{\sum_{i=1}^m \alpha_i}$. In this subsection, we investigate the effect of the modifications to understand how they improve the convergence performance in this car structure design problem.

Histories of N_ν spreading (median result) are shown in Fig. 3.2. MCR_7, MCR_8, and MCR_9 tend to generate infeasible individuals with larger N_ν spreading compared with other CHTs during the optimization. The modifications most likely generate more variety of infeasible individuals compared with other CHTs. This might be one reason why the convergence performance improves. Since there are more variety of infeasible individuals in the population, the quality of interaction between the feasible and infeasible individuals

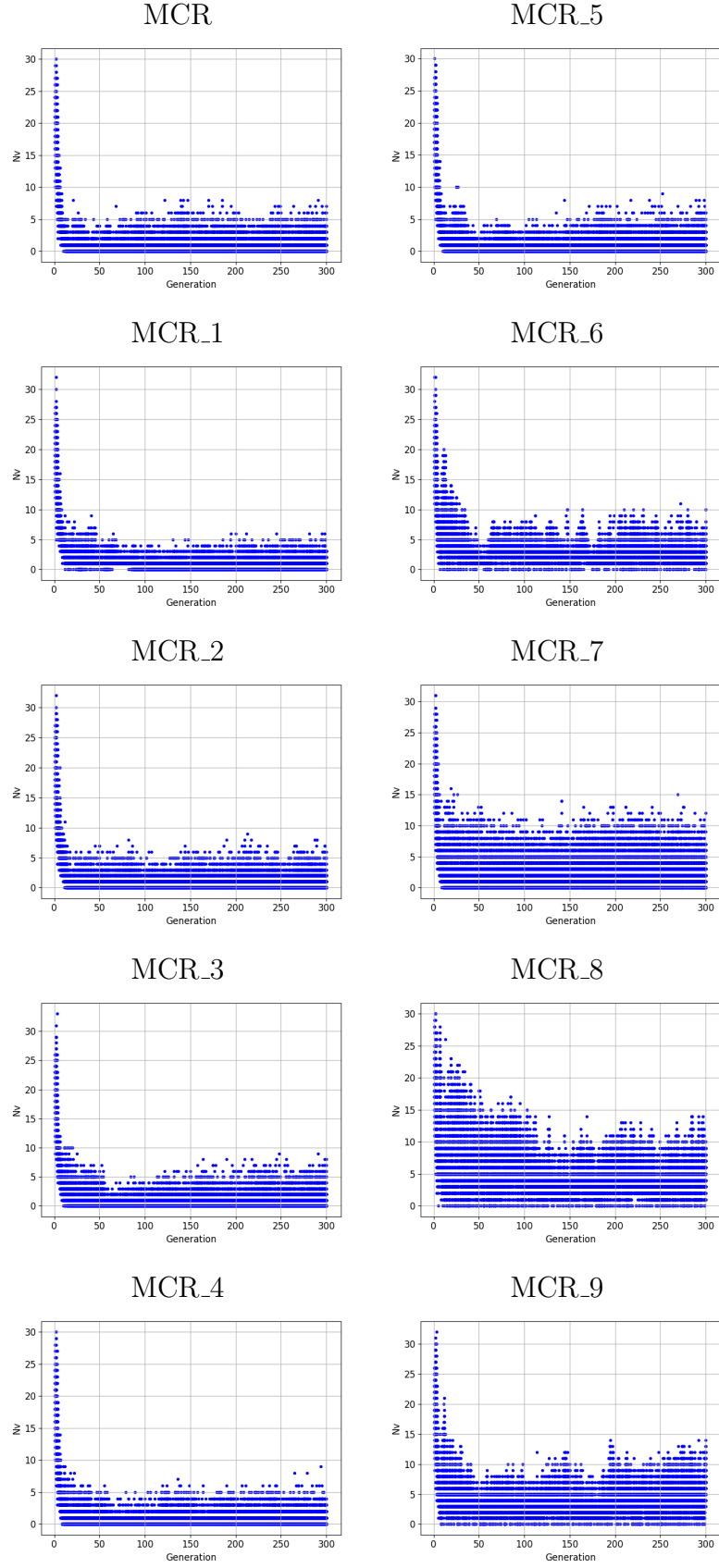


Figure 3.2: N_v spreading of the CHTs (median of 51 runs).

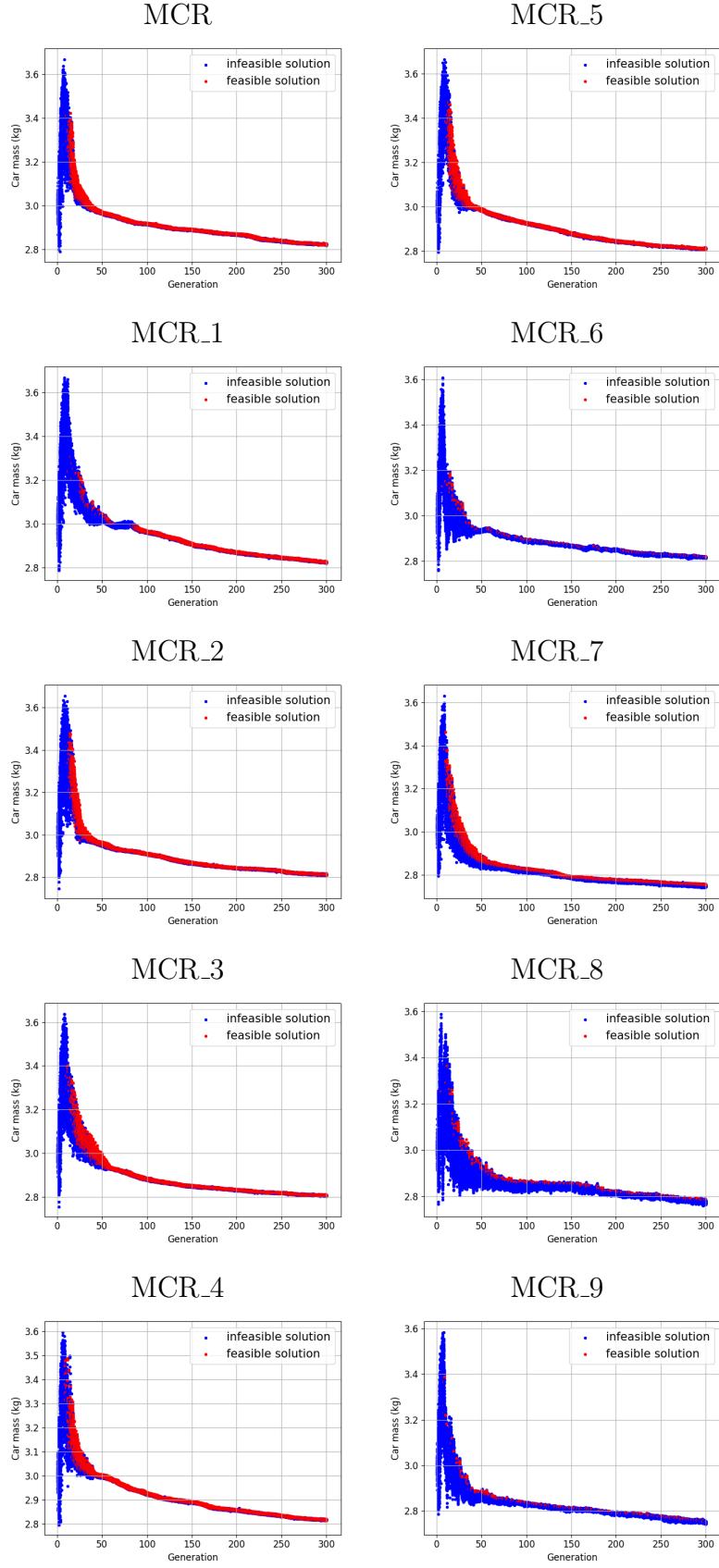


Figure 3.3: Objective value spreading of the CHTs (median of 51 runs).

most likely becomes better, thus leading to convergence improvement.

Another reason which might improve the convergence performance is that MCR_7, MCR_8, and MCR_9 produce many infeasible individuals with better objective values than the feasible ones. Histories of objective value spreading (median result) are depicted in Fig. 3.3. We confirm that there are many infeasible but lower-objective-value solutions for MCR_7, MCR_8, and MCR_9 in almost all generations (except in the very early generation where no feasible individual is found yet).

If we compare the role of each principle of modification, it seems that the adaptive α_i acts as the core of the convergence improvement. As we see in the discussion in the previous subsection, except for MCR_4, other CHTs which produce convergence improvement against MCR (MCR_2, MCR_3, MCR_7, MCR_8, and MCR_9) implement adaptive α_i . Without implementing adaptive α_i , there is no significant difference in convergence improvement compared with MCR.

The other principle, the omission of R_{N_ν} acts as the support of adaptive α_i to increase the amount of convergence improvement against MCR. By omitting the R_{N_ν} , the generated infeasible individuals are not restricted with small N_ν because the constraint assessment is only based on the sum of $\alpha_i R_{\nu_i}$. As depicted in Fig. 3.2, many infeasible individuals generated by MCR_7 have larger N_ν (around 10), while only around six for MCR_2. We can also find similar observation on the comparisons of MCR_3 with MCR_8 and MCR_4 with MCR_9.

Observing more deeply on adaptive α_i and omission of R_{N_ν} , it seems that MCR_7, MCR_8, and MCR_9 can produce infeasible individuals violating more constraints than MCR and other CHTs because each R_{ν_i} has weight that changes in every generation. From 54 constraints possessed by the problem, we observe three kinds of α_i behavior during the optimization:

1. constant $\alpha_i = 0$,
2. α_i that saturates to (almost) zero by the end of optimization, and
3. α_i that stays at or keep increasing to a certain rather big value until the end of optimization.

Starting from the beginning until the end of optimization, there are nine constraints whose α_i values are constant zeroes. Fig. 3.4 depicts the α_i of the 9th constraint, the representative for zero- α_i type. This indicates that these constraints are not active. Next, there are also constraints whose α_i s have rather big values in early generation, then those α_i values saturate to (almost) zero, regardless of the CHTs. Fig. 3.5 depicts the history of α_i of the 8th constraint, representing the 19 constraints whose α_i values saturate to zero. This indicates that those constraints are weakly-active.

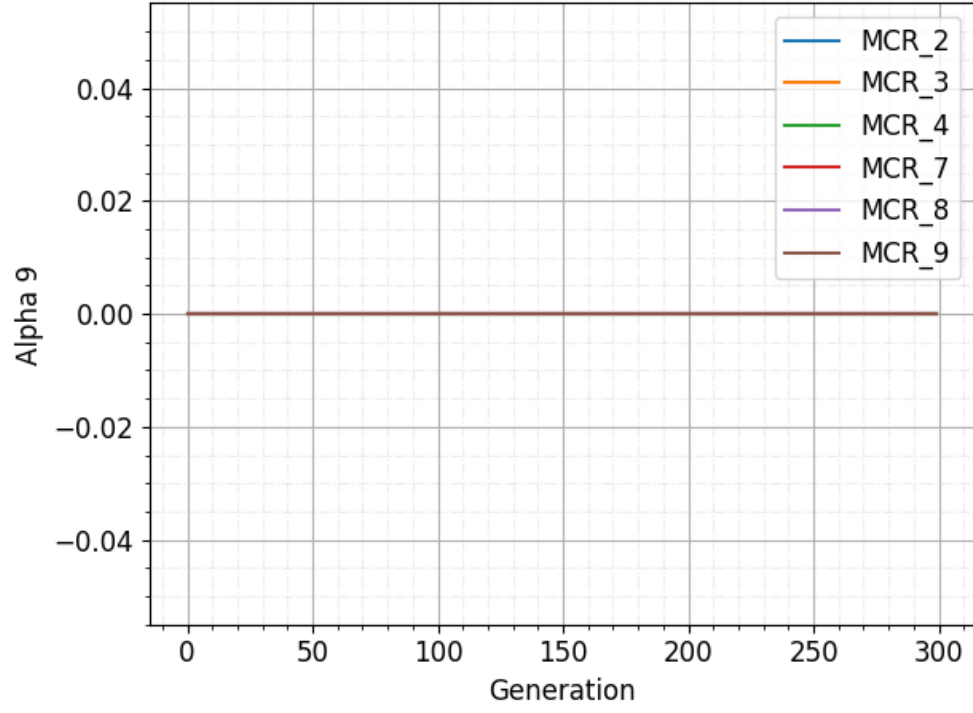


Figure 3.4: Average α_i of constraint 9 from 51 independent runs

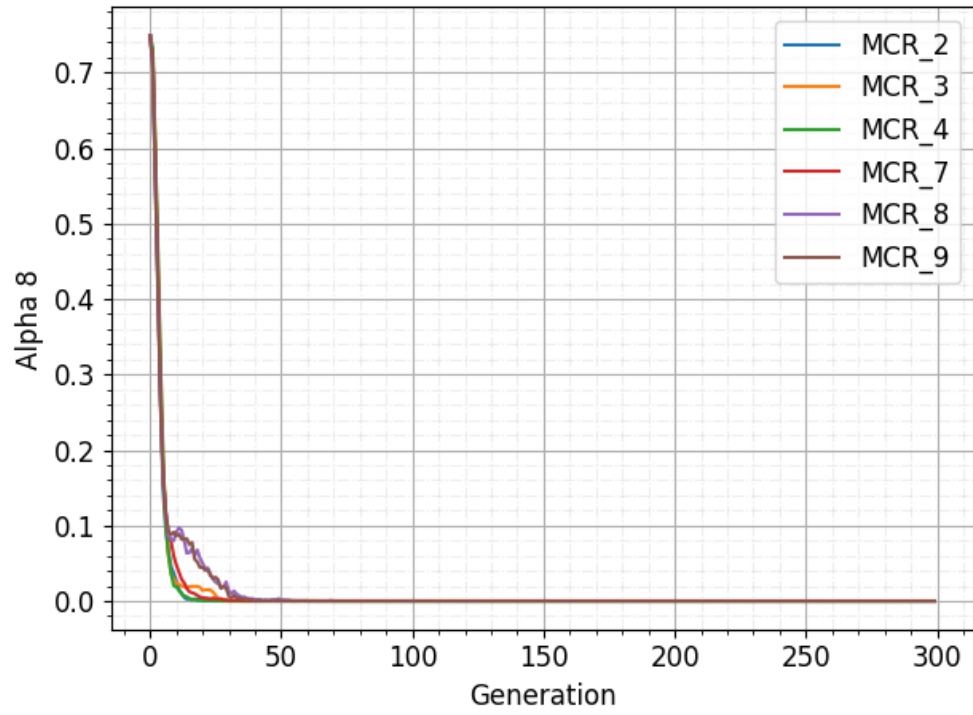


Figure 3.5: Average α_i of constraint 8 from 51 independent runs

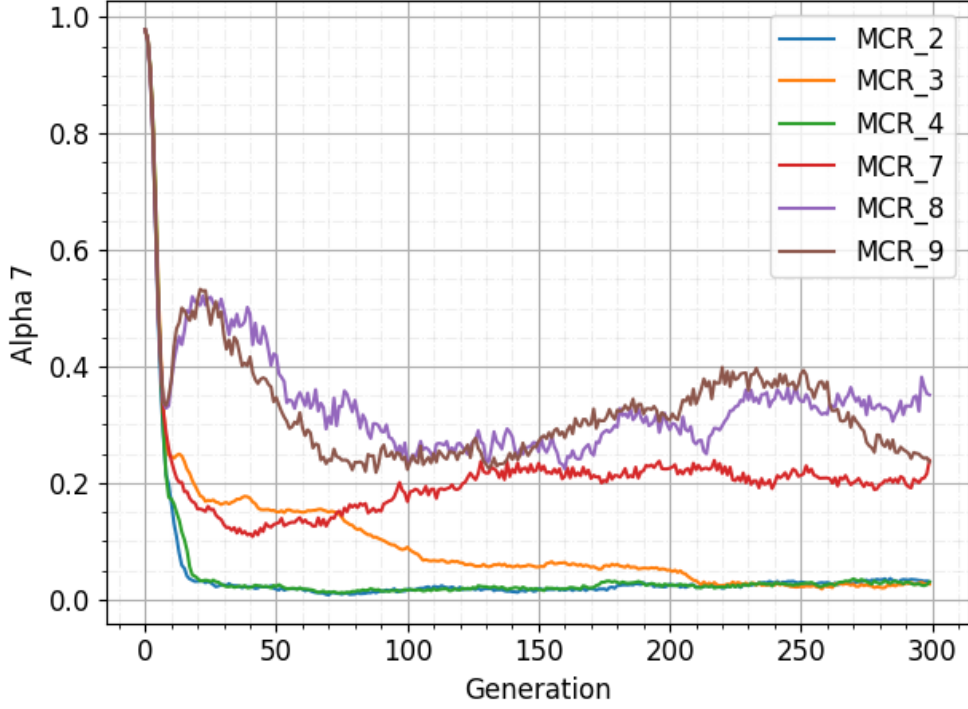


Figure 3.6: Average α_i of constraint 7 from 51 independent runs

Last but not least, there are constraints whose α_i values stay at or keep increasing to a certain rather big value. The 7th constraint represents such constraints. Fig. 3.6 depicts the history of α_i of the 7th constraint. We can observe that all CHTs produce non-zero α_i until the end of optimization. The difference is that MCR_2, MCR_3, and MCR_4 tend to stay at smaller α_i values than MCR_7, MCR_8, and MCR_9, probably due to the existence of R_{N_ν} . There are 26 constraints following these two patterns. This indicates that the constraints are severely-active.

It seems that the increase of variety of infeasible individuals also affects the constraint diversity (diversity of the offspring population in the constraint space). We confirm this by observing Fig. 3.7, which depicts the history of average constraint diversity from 51 independent runs in every generation. In the offspring population in a generation, we measure the constraint diversity based on the average Euclidean distance of individuals' constraint violations to a constraint center in that generation. The constraint center, $\nu_c = \{\nu_{1,c}, \dots, \nu_{m,c}\}^T$ is calculated as follows,

$$\nu_{i,c} = \frac{1}{N_{pop}} \sum_{j=1}^{N_{pop}} \nu_{i,j} \quad (3.5)$$

where $\nu_{i,c}$ denotes the average constraint violation of the i^{th} constraint in the offspring population, $i = 1, \dots, m$. We normalize the constraint violations to be between zero and

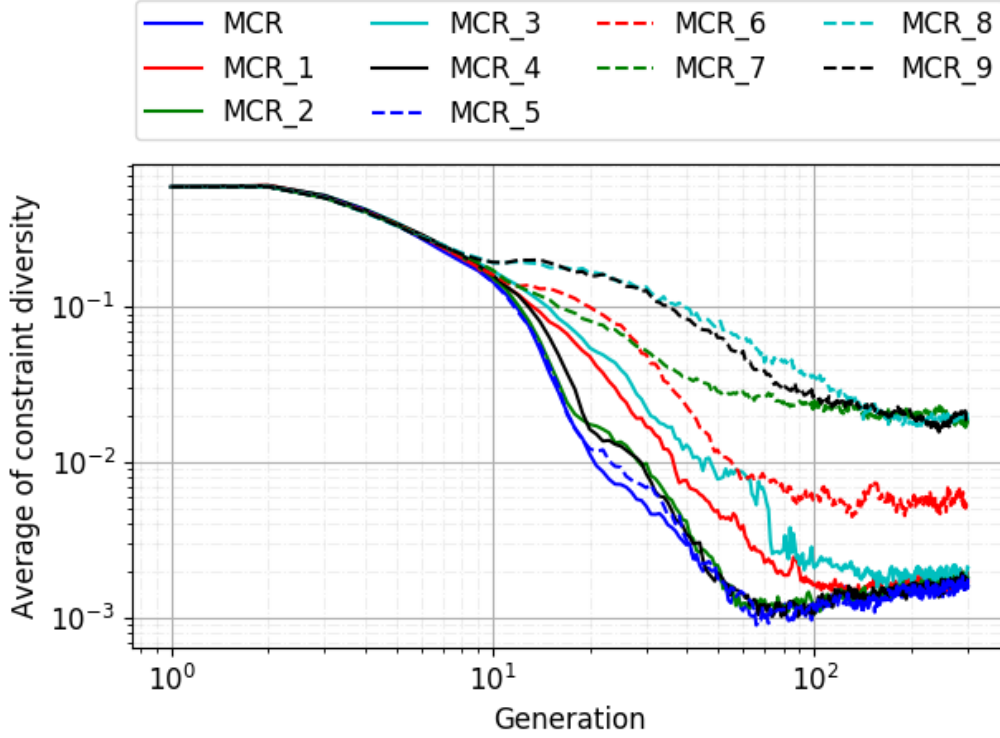


Figure 3.7: Histories of average diversity in constraint space

one with the maximum and minimum values obtained from all CHTs and independent runs in calculating the diversity and center to ensure the balanced calculation in case the constraints have different orders of magnitudes (the minimum value of constraint violation is zero because $\nu_i = \max(0, g_i)$).

The solid and dashed lines are the CHTs implementing and omitting R_{N_ν} , respectively. Comparing the solid and dashed lines with the same color, we can notice that except for MCR_5, CHTs omitting R_{N_ν} produce bigger diversity in the constraint space after the 10th generation. MCR_7, MCR_8 and MCR_9 produce the biggest constraint diversity compared with other CHTs. As for MCR_5, we do not observe the increase of constraint diversity most probably because the benchmark problem has many constraints. Even though the R_{N_ν} is omitted in MCR_5, the portion of $\sum_{i=1}^m R_{\nu_i}$ is still much bigger than R_f , similar with MCR, so that the constraint diversity is also similar.

Even though there is no direct correlation with the design variable space, we notice that other than MCR_5, CHTs implementing omission of R_{N_ν} also produce bigger design variable (DV) diversity (diversity of the offspring population in the design variable space) than CHTs with R_{N_ν} , similar with constraint diversity. As depicted in Fig. 3.8, MCR_6, MCR_7, MCR_8, and MCR_9 produce bigger DV diversity starting at around the 10th generation. We measure the DV diversity with the same formulation with the constraint

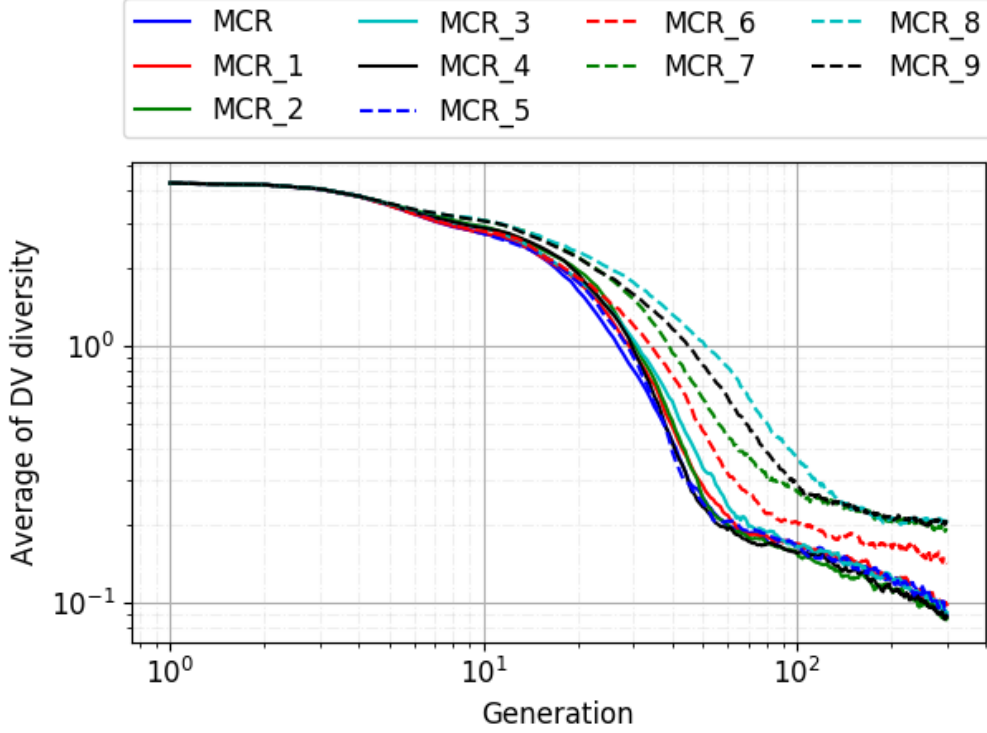


Figure 3.8: Histories of average diversity in design variable space

diversity, only that the data are the design variables of the individuals instead of the constraint violation values.

Proportionating the constraint violation rank terms (the sum of $\alpha_i R_{\nu_i}$) with other rank terms with either $\gamma = \frac{1}{m}$ or $\gamma = \frac{1}{\sum_{i=1}^m \alpha_i}$ tends to enhance the search speed towards unconstrained optimum. If we inspect Fig. 3.3, we can notice that the CHTs with R_{N_ν} omitted, such as MCR_6, MCR_8, and MCR_9, are the ones affected the most with the proportionation of the constraint violation rank terms. From the figure, it seems that even though producing many lower-objective-value-infeasible individuals, MCR_6, MCR_8, and MCR_9 only produce small number of feasible individuals during the optimization, because there are only very few red dots on many blue dots. As for MCR_1, MCR_3, and MCR_4, the ability to generate feasible individuals do not decrease that much compared with MCR because of the existence of R_{N_ν} .

The advantage of proportionating the constraint violation rank terms with either $\frac{1}{m}$ or $\frac{1}{\sum_{i=1}^m \alpha_i}$ is not observed here because the convergence performances of MCR_8 is worse than MCR_7 and MCR_9, and both MCR_7 and MCR_9 are competitive. It is probably because the car structure design problem has many constraints, proportionating the constraint violation rank terms makes the exploration in infeasible region too large, thus causes difficulty in generating other feasible individuals after finding the first feasible individual.

In order to take advantage of the implementation of either $\frac{1}{m}$ or $\frac{1}{\sum_{i=1}^m \alpha_i}$, it is necessary to study the tuning of β_1 and β_2 as the next step.

3.6 Summary

In this chapter, modifications on MCR have been proposed to enhance the infeasible region exploration capability. The effects of the modifications are investigated on a car structure design problem. The modifications are devised based on the consideration that the constraints might have different difficulty levels, omitting R_{N_ν} might give more flexibility in exploring infeasible region, and the portion of constraint violation ranks is too large in problems with large number of constraints.

As a result, we obtain that the CHTs implementing adaptive α_i , the modification proposed to cope with different difficulty levels in satisfying the constraints, significantly improves the convergence performance of MCR in this particular benchmark problem. The CHTs implementing at least adaptive α_i and omission of R_{N_ν} produce the largest convergence improvement against MCR. It is probably because the modifications generate more variety of infeasible individuals and many infeasible individuals with better objective values compared with MCR. More variety of infeasible individuals can be generated most likely because the adaptive α_i changes the weight of each R_{ν_i} in every generation. The increase of variety of infeasible individuals also increases the diversity of offspring population in both constraint and design variable spaces.

Proportionating the constraint violation rank term with either $\frac{1}{m}$ or $\frac{1}{\sum_{i=1}^m \alpha_i}$ keeps the order of magnitude between rank terms similar regardless of the number of constraints. However, the advantage of either one of these implementations is not observed here because the car structure design problem has many constraints which causes difficulty in generating any other feasible individuals after obtaining the first feasible individual. This situation might also occur in general problems. Therefore, it is necessary to study the effect of tuning β_1 and β_2 to take advantage of adopting either $\frac{1}{m}$ or $\frac{1}{\sum_{i=1}^m \alpha_i}$ as the next step.

Chapter 4

Adaptively Preserving Solutions in Both Feasible and Infeasible Regions

4.1 Introduction

In the preceding chapter, we have obtained that some modifications of MCR, namely MCR_7, MCR_8, and MCR_9, generate significant improvement on convergence performance towards constrained optimum on car structure design optimization problem. However, they seem to explore very wide infeasible region since the number of feasible individuals generated in the offspring population is small, especially for MCR_8 and MCR_9. While they work well on that particular car structure design problem, they might face difficulty in generating feasible individuals in other types of problem. In this chapter, we propose to tune β_1 and β_2 in order to improve MCR_7's, MCR_8's, and MCR_9's balance of search in both feasible and infeasible regions.

4.2 Adaptively Preserving Solutions in Both Feasible and Infeasible Regions

Here, we express again the generalized MCR from Eq. (3.1),

$$F = \beta_1 R_f + \beta_2 (\eta R_{N_v} + \gamma \sum_{i=1}^m \alpha_i R_{v_i}). \quad (4.1)$$

We also reinstate the base CHTs (MCR_7, MCR_8, and MCR_9) in Table 4.1. Note that for all three CHTs, $\beta_1 = 1$ if feasible individual exists and $\beta_1 = 0$ otherwise, and $\beta_2 = 1$. To strike better search balance in feasible and infeasible regions, we propose to modify MCR_7, MCR_8, and MCR_9 by tuning β_1 and β_2 adaptively, as follows,

$$\beta_1 = \mu(\zeta) \quad (4.2)$$

Table 4.1: MCR_7, MCR_8, and MCR_9

| CHT | η | γ | α_i |
|-------|--------|-----------------------------------|------------------------------|
| MCR_7 | 0 | 1 | $\frac{N_{viol_i}}{N_{pop}}$ |
| MCR_8 | 0 | $\frac{1}{m}$ | $\frac{N_{viol_i}}{N_{pop}}$ |
| MCR_9 | 0 | $\frac{1}{\sum_{i=1}^m \alpha_i}$ | $\frac{N_{viol_i}}{N_{pop}}$ |

and

$$\beta_2 = 1 - \beta_1 \quad (4.3)$$

where

$$\zeta = \frac{N_{feas}}{N_{pop}} \quad (4.4)$$

where N_{feas} and N_{pop} are the number of feasible individuals in the current population and the population size, respectively. The values of β_1 and β_2 adaptively change depending on the $\mu(\zeta)$, where $\mu(\zeta)$ is an increasing function whose value is between zero and one, depending on ζ . With this principle, the fitness formulation becomes

$$F(\mathbf{x}) \approx R_f \quad (4.5)$$

if N_{feas} is large, and

$$F(\mathbf{x}) \approx \eta R_{N_\nu} + \gamma \sum_{i=1}^m \alpha_i R_{\nu_i} \quad (4.6)$$

if N_{feas} is small. In other words, the algorithm will focus on the search towards finding unconstrained optimum if there are many feasible individuals in the current population, thus encouraging finding more infeasible individuals. Meanwhile, it will focus on finding more feasible individuals if there are a few feasible individuals in the population. This formulation encourages the preservation of both feasible and infeasible individuals in the population, thus maintains the balance of search in both feasible and infeasible regions.

We propose three forms of adaptive β_1 :

1. linear β_1 , $\beta_1 = \zeta$,
2. circular β_1 , $\beta_1 = \sqrt{1 - (\zeta - 1)^2}$, and
3. quadratic β_1 , $\beta_1 = 1 - (\zeta - 1)^2$.

In case of linear β_1 , the weights of R_f and the sum of R_{ν_i} will follow the solid (for β_1) and dashed (for β_2) blue lines depicted in Fig. 4.1. Linear β_1 will generate very small value if the number of feasible individuals in the population is very small. We have conducted

preliminary research about linear β_1 in [64] in the form of MCR-mod, whose fitness $F(\mathbf{x})$ formulation is expressed as follows,

$$F(\mathbf{x}) = \zeta R_f + (1 - \zeta)(R_{N_\nu} + \frac{1}{m} \sum_{i=1}^m R_{\nu_i}). \quad (4.7)$$

We found out that this form might produce very slow convergence rate in case of severely constrained optimization problem where feasible individuals are very difficult to find until the end of optimization. In such case, the value of β_1 becomes small throughout the optimization. In consequence, the search tends to find feasible individuals throughout the optimization instead of finding constrained optimum. This is the reason why we also propose two other forms of adaptive β_1 , in case similar slow convergence phenomenon also happens in the present work. As presented in Table 4.2, the ones implemented with linear β_1 are named G-MCR-L_7, G-MCR-L_8, and G-MCR-L_9.

The circular β_1 generates the weights of R_f and the sum of R_{ν_i} that follow the solid (for β_1) and dashed (for β_2) red lines depicted in Fig. 4.1. It provides bigger weight of R_f compared to linear β_1 when N_{feas} is small, leading to a faster search towards unconstrained optimum. However, in consequence, the ability of circular β_1 to generate feasible individuals might be worse than linear β_1 . As presented in Table 4.2, the ones implemented with circular β_1 are named G-MCR-C_7, G-MCR-C_8, and G-MCR-C_9.

As for the quadratic β_1 , it provides a performance between linear β_1 and circular β_1 in both search weight towards unconstrained optimum and ability to generate feasible individuals. The weights of R_f and the sum of R_{ν_i} generated by quadratic β_1 follow the solid (for β_1) and dashed (for β_2) green lines depicted in Fig. 4.1. As presented in Table 4.2, the ones implemented with green β_1 are named G-MCR-Q_7, G-MCR-Q_8, and G-MCR-Q_9.

4.3 Experimental Setup

4.3.1 Benchmark problems

To investigate the effect of the proposed β_1 s, we conduct numerical experiments in 78 benchmark problems which comprise CEC 2006 (28 problems) [65], CEC 2010 (18 problems, all with $d = 10$) [66], CEC 2017 (28 problems, all with $d = 10$) [67] benchmark problem suites, five analytical engineering design problems (welded beam [68], tension/compression spring [69], symmetric three-bar truss [70], pressure vessel [71], and speed reducer [72] problems), and three real-world design problems (car structure design problem [11], MOPTA 2008 [31], and wind turbine [32] problems). The codes for car structure design, MOPTA 2008, and wind turbine design problems can be downloaded

Table 4.2: G-MCR with new modifications, $\zeta \equiv \frac{N_{feas}}{N_{pop}}$

| CHT | β_1 | β_2 | η | γ | α_i |
|-----------|----------------------------|---------------|--------|-------------------------------|------------------------------|
| G-MCR-L_7 | ζ | $1 - \beta_1$ | 0 | 1 | $\frac{N_{viol_i}}{N_{pop}}$ |
| G-MCR-L_8 | ζ | $1 - \beta_1$ | 0 | $\frac{1}{m}$ | $\frac{N_{viol_i}}{N_{pop}}$ |
| G-MCR-L_9 | ζ | $1 - \beta_1$ | 0 | $\frac{1}{\sum_i^m \alpha_i}$ | $\frac{N_{viol_i}}{N_{pop}}$ |
| G-MCR-C_7 | $\sqrt{1 - (\zeta - 1)^2}$ | $1 - \beta_1$ | 0 | 1 | $\frac{N_{viol_i}}{N_{pop}}$ |
| G-MCR-C_8 | $\sqrt{1 - (\zeta - 1)^2}$ | $1 - \beta_1$ | 0 | $\frac{1}{m}$ | $\frac{N_{viol_i}}{N_{pop}}$ |
| G-MCR-C_9 | $\sqrt{1 - (\zeta - 1)^2}$ | $1 - \beta_1$ | 0 | $\frac{1}{\sum_i^m \alpha_i}$ | $\frac{N_{viol_i}}{N_{pop}}$ |
| G-MCR-Q_7 | $1 - (\zeta - 1)^2$ | $1 - \beta_1$ | 0 | 1 | $\frac{N_{viol_i}}{N_{pop}}$ |
| G-MCR-Q_8 | $1 - (\zeta - 1)^2$ | $1 - \beta_1$ | 0 | $\frac{1}{m}$ | $\frac{N_{viol_i}}{N_{pop}}$ |
| G-MCR-Q_9 | $1 - (\zeta - 1)^2$ | $1 - \beta_1$ | 0 | $\frac{1}{\sum_i^m \alpha_i}$ | $\frac{N_{viol_i}}{N_{pop}}$ |

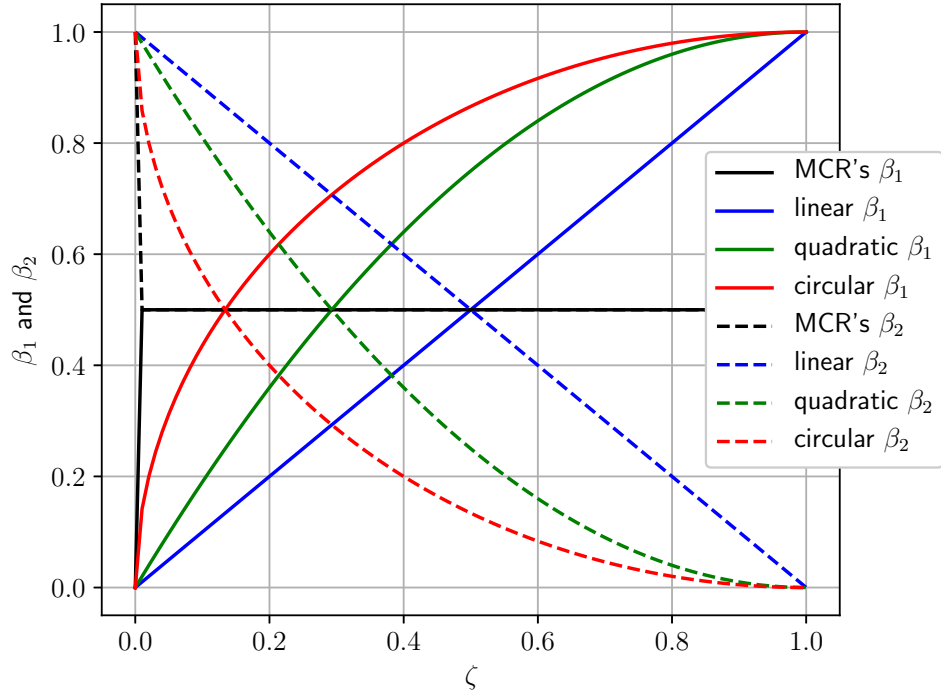


Figure 4.1: Visualization of β_1 and β_2 (normalized in such way so that $\beta_1 + \beta_2 = 1$).

from [62], [73], and [74], respectively. The number of problems involving only inequality constraints is 38 and the number of problems involving only equality constraints or both types of constraints is 40.

4.3.2 Optimization setting

In optimizing the benchmark problems, we utilize real-coded genetic algorithm (RGA) with elitist scheme [39]. All optimizations are set to be a minimization. In the present work, we are interested to investigate the performance of the CHTs within the limit of number of solution evaluations typically used in the real-world design optimization. Therefore, we use the recommended number of solution evaluations in the car structure design problem, 30,000 evaluations, for most benchmark problems. Except for wind turbine problem, we set the number of solution evaluations to 10,000, as suggested in its literature [32].

As for the setting of RGA, we set the population size to 100 and the number of generations to 300 (100 for wind turbine problem). We set the probability of crossover to one and probability of mutation to $\frac{1}{d}$. We utilize binary tournament selection as the selection method, simulated binary crossover [56] as the crossover method, and polynomial mutation [58] as the mutation method. We set the distribution indices for both crossover and mutation to 20. For the elitist scheme, we generate the new parent population based on the N_{pop} individuals with the best $F(\mathbf{x})$ from the combined old parent and offspring populations.

For each benchmark problem, we conduct 51 independent runs of optimization (21 independent runs for wind turbine problem). To investigate whether the results of the CHTs are significantly different, we utilize Wilcoxon rank-sum test [63] with significance level of 0.05 as the significance test method. In comparing the results, other than CHTs from Table 4.2, we also include the performance of other types of CHTs,

- MCR (Eq. (2.34)) and MCR-mod (Eq. (4.7))
- the base CHTs: MCR_7, MCR_8, and MCR_8 (Table 4.1)
- Category A: SoF
- Category B: APM, BIF-EO, SR, GCR, HnS, and E-BRM.

Category A refers to CHTs favoring feasible individuals over infeasible individuals and Category B refers to CHTs implementing balanced search in both feasible and infeasible regions but not implementing balanced assessment on constraints. The detail of each CHT is explained in Chapter 2. The user-defined weights/ probabilities for BIF-EO, SR,

and GCR are set to 0.6, 0.45, and 0.45, respectively, based on the recommendation from each corresponding literature.

4.4 Result and Discussion

First of all, we compare the performance of the CHTs with MCR. Then, we pick some of the best CHTs and discuss why they can become the best. Last but not least, to fulfill our interest in finding the most suitable CHT for real-world design problems, we also conduct a deeper discussion regarding the performance of some best CHTs on the selected real-world design problems.

4.4.1 Overall result - comparison with MCR

In comparing the results with MCR, we conduct the significance test between the last best constrained optimum obtained by each other CHT against the last best constrained optimum obtained by MCR on each problem. If the other CHT is significantly better than MCR, it obtains one win. If it is significantly worse than MCR, the other CHT obtains one loss. If both results are comparable, both CHTs obtains a draw. Then, we sum up all of the obtained wins and losses for all problems to determine which CHT that obtains the biggest score against MCR.

Table 4.3 presents the overall score obtained by other CHTs when compared with MCR. The columns "Win", "Loss", and "Draw" show the number of problems in which the CHT is significantly better than, significantly worse than, and comparable with MCR, respectively. The column "ALL" shows the total score from all 78 problems,

$$ALL = Win - Loss. \quad (4.8)$$

We can observe that many of our proposed modifications tend to obtain higher scores than MCR-mod and most of the CHTs from Category B. Except for GCR, the only CHT from Category B, which also obtains quite high score (22 score). MCR_7 is also quite robust by obtaining 21 score. G-MCR-C_7 and G-MCR-C_9 improve the convergence performance of MCR the most robustly by obtaining 29 score from all 78 problems.

To understand more of the convergence performance of the CHTs, now we breakdown the observation into two categories: on problems involving only inequality constraints (38 problems) and on problems involving either only equality constraints or both types of constraints (40 problems). The significance test scores of both categories are presented in Tables 4.4 and 4.5, respectively. We can observe a quite contrast CHT performances comparing both results. First, we compare the modified CHTs implementing the linear β_1 , quadratic β_1 , and circular β_1 , with the base CHTs (MCR_7, MCR_8, and MCR_9).

Table 4.3: CHTs against MCR (all 78 problems)

| CHT | Win | Loss | Draw | ALL |
|-----------|-----|------|------|-----|
| MCR-mod | 12 | 19 | 47 | -7 |
| MCR_7 | 29 | 8 | 41 | 21 |
| G-MCR-L_7 | 29 | 11 | 38 | 18 |
| G-MCR-Q_7 | 31 | 8 | 39 | 23 |
| G-MCR-C_7 | 36 | 7 | 35 | 29 |
| MCR_8 | 24 | 21 | 33 | 3 |
| G-MCR-L_8 | 30 | 12 | 36 | 18 |
| G-MCR-Q_8 | 31 | 11 | 36 | 20 |
| G-MCR-C_8 | 28 | 13 | 37 | 15 |
| MCR_9 | 23 | 12 | 43 | 11 |
| G-MCR-L_9 | 29 | 14 | 35 | 15 |
| G-MCR-Q_9 | 32 | 8 | 38 | 24 |
| G-MCR-C_9 | 35 | 6 | 37 | 29 |
| SoF | 6 | 22 | 50 | -16 |
| APM | 16 | 11 | 51 | 5 |
| BIF-EO | 7 | 20 | 51 | -13 |
| SR | 18 | 12 | 48 | 6 |
| GCR | 25 | 3 | 50 | 22 |
| HnS | 5 | 7 | 66 | -2 |
| E-BRM | 14 | 28 | 36 | -14 |

On problems involving only inequality constraints (Table 4.4), it seems that our proposed modifications (Tables 4.2 and 4.1) tend to improve the convergence performance of MCR more robustly than MCR-mod, SoF, APM, BIF-EO, SR, GCR, HnS, and E-BRM. Except for MCR_8 and G-MCR-C_8 because they only obtain scores of one and four, respectively. This is an indication that on this type of problems, CHTs with balanced search in both feasible and infeasible regions and balanced assessment on the constraints have better convergence performances than CHTs from either Category A, B, or C. G-MCR-L_7 and G-MCR-Q_9 are the best performers for problems involving only inequality constraints with 25 score for each.

Comparing the CHTs implementing and not implementing adaptive β_1 (still from Table 4.4), we can observe that G-MCR-L_7, G-MCR-Q_7, and G-MCR-C_7 all obtain higher ALL scores (scores of 25, 22, and 19, respectively) than MCR_7 (score of 17). This is an indication that all variants of adaptive β_1 are more robust than MCR_7 when we compare their convergence performances with MCR. Similar observations can also be

Table 4.4: CHTs against MCR (problems involving only inequality constraints, 38 problems)

| CHT | Win | Loss | Draw | ALL |
|-----------|-----|------|------|-----|
| MCR-mod | 9 | 2 | 27 | 7 |
| MCR_7 | 22 | 5 | 11 | 17 |
| G-MCR-L_7 | 25 | 0 | 13 | 25 |
| G-MCR-Q_7 | 24 | 2 | 12 | 22 |
| G-MCR-C_7 | 23 | 4 | 11 | 19 |
| MCR_8 | 15 | 14 | 9 | 1 |
| G-MCR-L_8 | 22 | 1 | 15 | 21 |
| G-MCR-Q_8 | 20 | 6 | 12 | 14 |
| G-MCR-C_8 | 15 | 11 | 12 | 4 |
| MCR_9 | 16 | 6 | 16 | 10 |
| G-MCR-L_9 | 24 | 1 | 13 | 23 |
| G-MCR-Q_9 | 26 | 1 | 11 | 25 |
| G-MCR-C_9 | 22 | 4 | 12 | 18 |
| SoF | 5 | 6 | 27 | -1 |
| APM | 5 | 6 | 27 | -1 |
| BIF-EO | 5 | 6 | 27 | -1 |
| SR | 9 | 7 | 22 | 2 |
| GCR | 8 | 2 | 28 | 6 |
| HnS | 3 | 3 | 32 | 0 |
| E-BRM | 12 | 12 | 14 | 0 |

noticed on comparison of G-MCR-L_8, G-MCR-Q_8, and G-MCR-C_8 with MCR_8, as well as G-MCR-L_9, G-MCR-Q_9, and G-MCR-C_9 with MCR_9. All variants of adaptive β_1 produce more robust convergence performances compared with the base CHTs.

Comparing the performance of each variant of adaptive β_1 (still from Table 4.4), we can observe that on the modifications of MCR_7, the linear β_1 (G-MCR-L_7) is better than the other two variants 25 score, than followed by quadratic β_1 (G-MCR-Q_7) with 22 score, and lastly circular β_1 (G-MCR-C_7) with 19 score. Similar observation can also be noticed on the modifications of MCR_8. Linear β_1 (G-MCR-L_8) is the best among the β_1 variants with 21 score, followed by quadratic β_1 (G-MCR-Q_8) with 14 score, and circular β_1 (G-MCR-C_8) with 4 score. On the modifications of MCR_9, it is slightly different. Quadratic β_1 (G-MCR-Q_9) is the best among the variants of β_1 with 25 score, followed by linear β_1 (G-MCR-L_9) with 23 score, and then circular β_1 (G-MCR-C_9) with 18 score. On problems involving only inequality constraints, it seems that linear β_1

Table 4.5: CHTs against MCR (problems involving either only equality constraints or both inequality and equality constraints, 40 problems)

| CHT | Win | Loss | Draw | ALL |
|-----------|-----|------|------|-----|
| MCR-mod | 3 | 17 | 20 | -14 |
| MCR_7 | 7 | 3 | 30 | 4 |
| G-MCR-L_7 | 4 | 11 | 25 | -7 |
| G-MCR-Q_7 | 7 | 6 | 27 | 1 |
| G-MCR-C_7 | 13 | 3 | 24 | 10 |
| MCR_8 | 9 | 7 | 24 | 2 |
| G-MCR-L_8 | 8 | 11 | 21 | -3 |
| G-MCR-Q_8 | 11 | 5 | 24 | 6 |
| G-MCR-C_8 | 13 | 2 | 25 | 11 |
| MCR_9 | 7 | 6 | 27 | 1 |
| G-MCR-L_9 | 5 | 13 | 22 | -8 |
| G-MCR-Q_9 | 6 | 7 | 27 | -1 |
| G-MCR-C_9 | 13 | 2 | 25 | 11 |
| SoF | 1 | 16 | 23 | -15 |
| APM | 11 | 5 | 24 | 6 |
| BIF-EO | 2 | 14 | 24 | -12 |
| SR | 9 | 5 | 26 | 4 |
| GCR | 17 | 1 | 22 | 16 |
| HnS | 2 | 4 | 34 | -2 |
| E-BRM | 2 | 16 | 22 | -14 |

produces the most robust performance by becoming the best in improving MCR_7 and MCR_8. Meanwhile, circular β_1 consistently becomes the worst among the β_1 variants because the improvement it gives to MCR_7, MCR_8, and MCR_9 is the least robust. As for quadratic β_1 , its robustness tends to be between linear and circular β_1 s.

On problems involving either only equality constraints or both inequality and equality constraints (Table 4.5), the result observed is very different with the problems involving only inequality constraints. It seems that only some CHTs obtain high scores. From the proposed modifications are G-MCR-C_7, G-MCR-C_8, and G-MCR-C_9 with 10, 11, and 11 scores, respectively. GCR from Category B performs the most robustly by obtaining 16 score. When an equality constraint is involved, the problem most likely has small feasible region. Therefore, the difficulty level in obtaining feasible individuals tends to increase quite significantly. Probably, this is why the performances of some CHTs which are good on problems involving only inequality constraints deteriorate quite significantly on

problems where equality constraints are also involved. The clearest results are the performances of CHTs implementing linear β_1 (G-MCR-L7, G-MCR-L8, and G-MCR-L9). Even though they have good performances on problems involving only inequality constraints, their performances on problems which also involve equality constraints are very poor. G-MCR-L7, G-MCR-L8, and G-MCR-L9 get -7, -3, and -8 scores, respectively. This means that their performances are less robust than even MCR.

One reason that might cause the robustness of G-MCR-L7, G-MCR-L8, and G-MCR-L9 deteriorates is the small β_1 values due to small number of feasible individuals in the population because the feasible individuals are difficult to find. Small number of feasible individuals leads to small β_1 , thus further leading to small weight on R_f . In consequence, the search focuses on finding more feasible individuals than finding the constrained optimum. This is consistent with our preliminary investigation about MCR-mod [64].

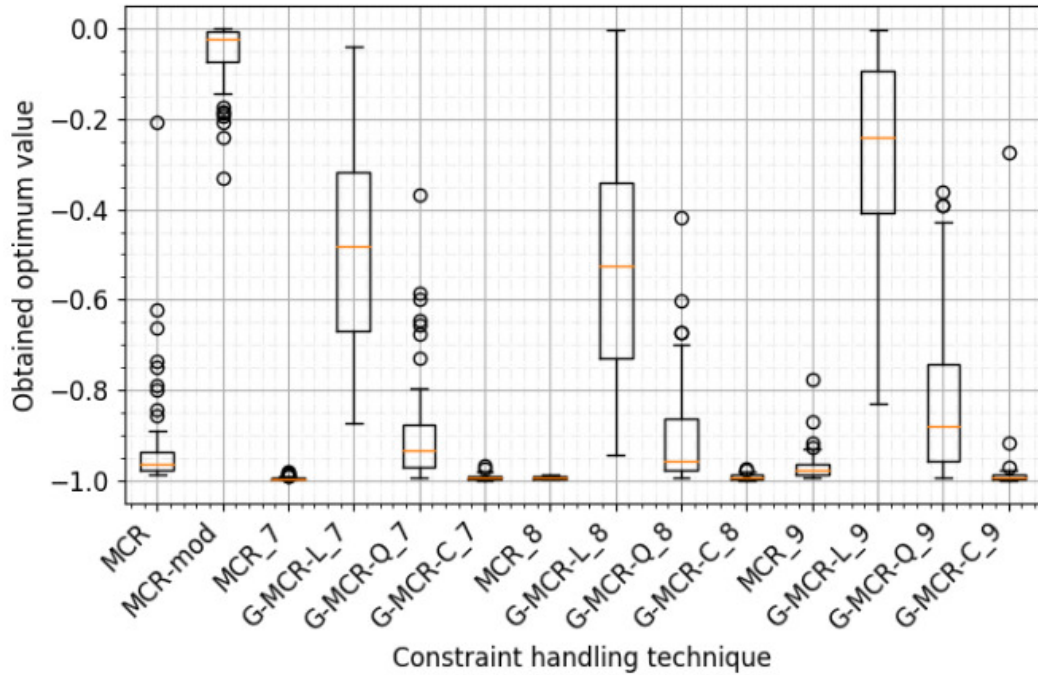


Figure 4.2: Box plot of G3 problem

One example in which the small β_1 value might be the cause of slow convergence on problems involving equality constraints is on G3 problem from CEC 2006 problem suite [65]. It is a problem with one equality constraint and 10 design variables. Since there is only one constraint, G3 is only a problem with small feasible region (no feasible solution is found from 1,000,000 random tests) and there is no effect from many constraints.

Fig. 4.2 depicts the box plots of obtained optimum values by MCR, MCR-mod, and G-MCR-based CHTs on G3 problem. We can observe that the CHTs implementing linear

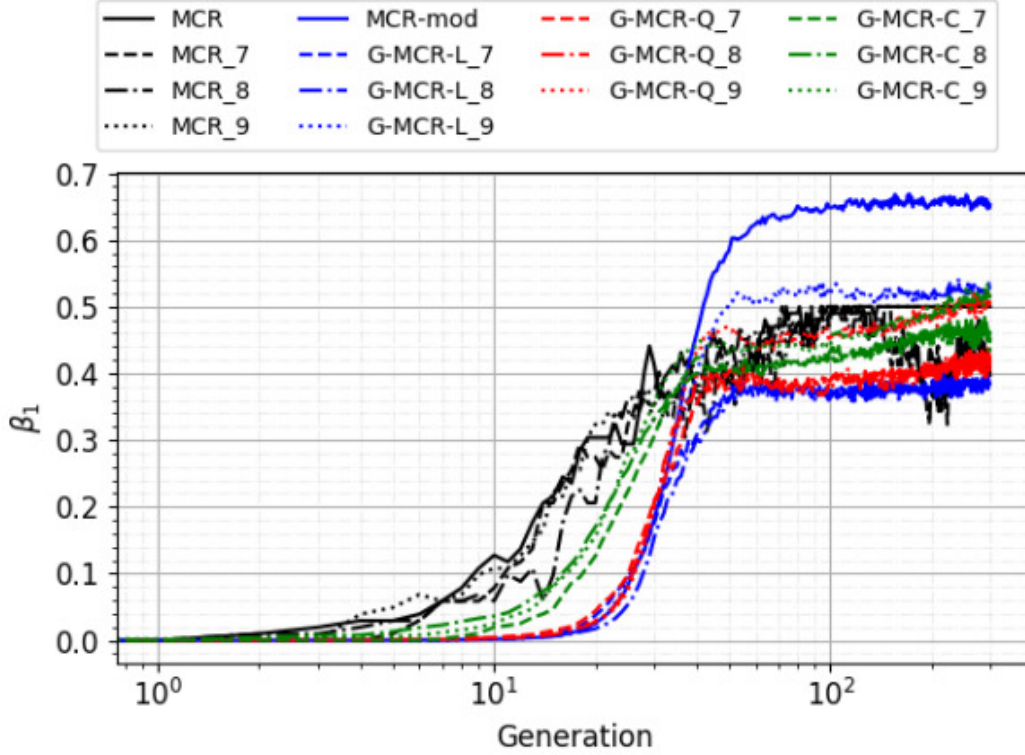


Figure 4.3: Histories of average β_1 of G-MCR-based CHTs on G3 problem

β_1 (MCR-mod, G-MCR-L_7, G-MCR-L_8, and G-MCR-L_9) consistently obtain the worst convergence performance among the CHTs. The best CHTs are the ones not implementing any adaptive β_1 and the ones implementing circular β_1 (MCR, MCR_7, MCR_8, MCR_9, G-MCR-C_7, G-MCR-C_8, G-MCR-C_9).

If we match their convergence performances and their average β_1 histories throughout the generation (depicted in Fig. 4.3), we can find that the CHTs not implementing any adaptive β_1 (black color: MCR, MCR_7, MCR_8, MCR_9) tend to produce the largest average β_1 in the early generation among the CHTs. Their convergence performances are among the best. Meanwhile, the CHTs implementing linear β_1 (blue color: G-MCR-L_7, G-MCR-L_8, G-MCR-L_9) and quadratic β_1 (red color: G-MCR-Q_7, G-MCR-Q_8, G-MCR-Q_9) tend to produce the smallest average β_1 in the early generation. Their convergence performances are among the worst (with CHTs implementing linear β_1 perform the worst). As for CHTs implementing circular β_1 (green color: G-MCR-C_7, G-MCR-C_8, G-MCR-C_9), they tend to produce average β_1 value larger than CHTs implementing linear β_1 and quadratic β_1 but smaller than CHTs not implementing any adaptive β_1 . Their convergence performances are also among the best.

4.4.2 Discussion on results related to real-world design problems (RWDPs)

Table 4.6: Properties of RWDPs

| RWDP | d | m | FR (%) |
|----------------------|-----|-----|--------|
| car structure design | 222 | 54 | 0.00 |
| MOPTA 2008 | 124 | 68 | 0.00 |
| wind turbine design | 32 | 22 | 0.00 |

In the previous subsection, we have observed some CHTs whose performances robustly improve the convergence rate of MCR for general problems. However, we still do not know the exact amount of their convergence improvements. In this subsection, we are interested to know in more detail about the performance of the CHTs such as how big their the convergence improvements are compared with each other and the effect of the modifications. Since we are interested on real-world optimization problems in the present work, we focus our discussion on the results related to the three real-world optimization problems investigated here: car structure design [11], MOPTA 2008 [31], and wind turbine [32] problems. The properties of the RWDPs are shown in Table 4.6, where d , m , and FR denote the number of design variables, number of constraints, and approximated feasible region, respectively. All problems have many constraints (only inequality constraints) and small feasible region (estimated from 1,000,000 random tests for car structure design and MOPTA 2008 problems, 100,000 for wind turbine problem). The constraints include not active, weakly-active, and severely-active constraints, and they have different orders of magnitudes.

a. Car Structure Design

We depict the box plot of the obtained optimum values by each CHTs on Fig. 4.4. We can observe that the G-MCR-based CHTs consistently produce better convergence improvements than MCR, MCR-mod, SoF, and CHTs from Category B. G-MCR-C_9 produces the largest convergence improvement, followed by G-MCR-L_8, G-MCR-Q_8, and MCR_7. We also conduct significant test for one CHT against each other to check if the convergence performances of two CHTs are significantly different. Suppose CHT_1 against CHT_2, CHT_1 will obtain 1, 0, and -1 scores if CHT_1 is significantly superior, significantly inferior, and competitive, respectively, in comparison with CHT_2. Table 4.7 column "car structure design" presents the sum of scores obtained from the significance test. We can notice that G-MCR-C_9 obtains 20 scores, meaning that its convergence performance is significantly superior than all other CHTs. G-MCR-L_8 and G-MCR-Q_8

Table 4.7: Overall score of significance test (CHTs against other CHTs) on RWDPs

| CHT | car structure design | MOPTA 2008 | wind turbine |
|-----------|----------------------|------------|--------------|
| MCR | -14 | -13 | -6 |
| MCR-mod | -13 | -10 | -5 |
| MCR_7 | 14 | 17 | 11 |
| G-MCR-L_7 | 0 | 3 | 11 |
| G-MCR-Q_7 | 1 | 7 | 11 |
| G-MCR-C_7 | 11 | 14 | 11 |
| MCR_8 | 0 | -2 | -13 |
| G-MCR-L_8 | 16 | 12 | 11 |
| G-MCR-Q_8 | 16 | 12 | 11 |
| G-MCR-C_8 | 6 | -2 | -4 |
| MCR_9 | 12 | 9 | -5 |
| G-MCR-L_9 | 5 | 5 | 11 |
| G-MCR-Q_9 | 7 | 11 | 10 |
| G-MCR-C_9 | 20 | 20 | 11 |
| SoF | -14 | -14 | -8 |
| APM | -13 | -11 | -14 |
| BIF-EO | -13 | -15 | -12 |
| SR | -6 | -6 | -18 |
| GCR | -4 | -2 | 11 |
| HnS | -13 | -15 | -7 |
| E-BRM | -18 | -20 | -17 |

follow as the second best with 16 score each, meaning that their convergence performances are better than equivalent to 16 other CHTs. Then, MCR_7 is in the third place with 14 score (equivalent to winning against 14 other CHTs).

It seems that the factors which make MCR_7, G-MCR-L_8, G-MCR-Q_8, and G-MCR-C_9 can obtain the best convergence improvement are consistent with what are mentioned in the preceding chapter. Observing the middle column of Fig. 4.5, we can notice that those CHTs also tend to produce more variety of infeasible individuals compared with MCR. While MCR tends to produce infeasible individuals violating around six constraints, MCR_7, G-MCR-L_8, G-MCR-Q_8, and G-MCR-C_9 tend to produce infeasible individuals violating 10 or more constraints. We can also notice that MCR_7, G-MCR-L_8, G-MCR-Q_8, and G-MCR-C_9 also tend to produce many infeasible individuals with better objective values than the feasible ones by observing that there are many blue dots under the red dots in the right column of Fig. 4.5. Meanwhile, we hardly observe similar

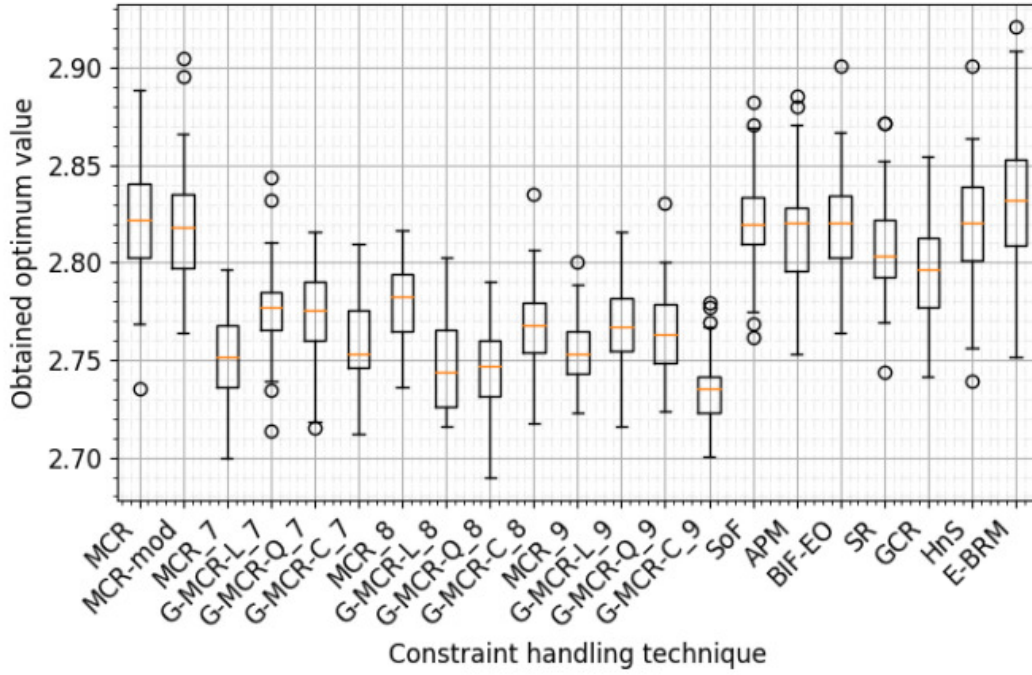


Figure 4.4: Box plot of car structure design problem.

thing in MCR.

Now, we are interested to observe the effect of the implementation of different adaptive β_1 s. Observing the constraint diversity of some CHTs in Fig. 4.6, it seems that all G-MCR-based CHTs have better diversity than MCR. Comparing only the ones implementing adaptive β_1 s, the one implementing circular β_1 seems to have the largest diversity, followed by quadratic β_1 , and then linear β_1 .

b. MOPTA 2008

Fig. 4.7 depicts the box plot of obtained optimum values of each CHT for MOPTA 2008 problem. Similar with car structure design, we can observe that G-MCR-based CHTs tend to produce better convergence performances than MCR, MCR-mod, SoF, and the the CHTs from Category B. Except for MCR_8 and G-MCR-C_8 which seem to be competitive with GCR. G-MCR-C_9 produces the best convergence performance among the CHTs. It seems to be followed by MCR_7 and G-MCR-C_7 in the second and third places, respectively. The significance test result in Table 4.7 column "MOPTA 2008" ensures the observation in the box plot. G-MCR-C_9 becomes the best performer by obtaining 20 scores. Then, it is followed by MCR_7 and G-MCR-C_7 by obtaining 17 and 14 scores, respectively.

Similar with car structure design problem, it seems that MCR_7, G-MCR-C_7, and G-MCR-C_9 can obtain better convergence because they also produce more variety of

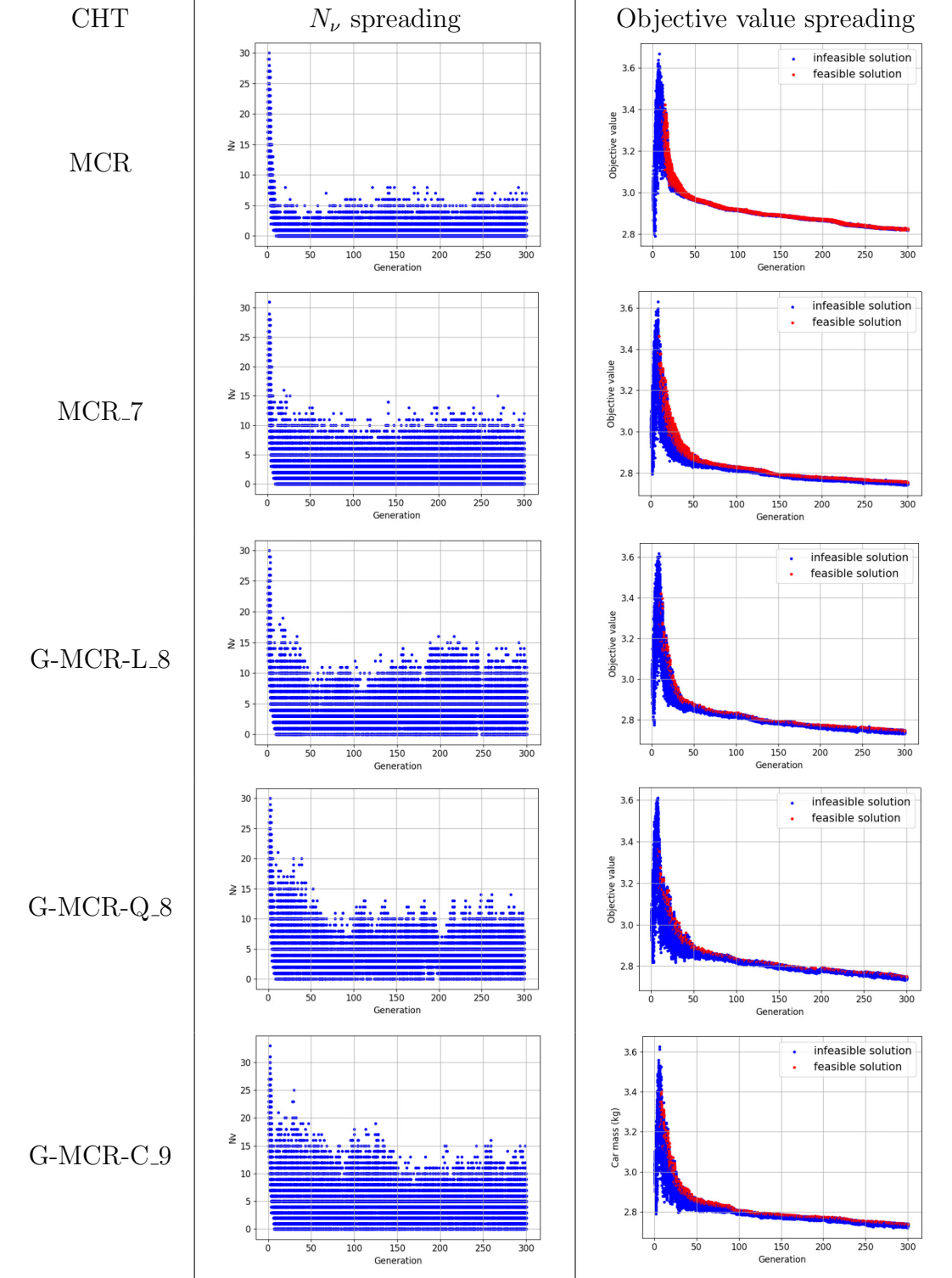


Figure 4.5: N_ν and objective value spreadings of selected CHTs (median of 51 runs) on car structure design problem

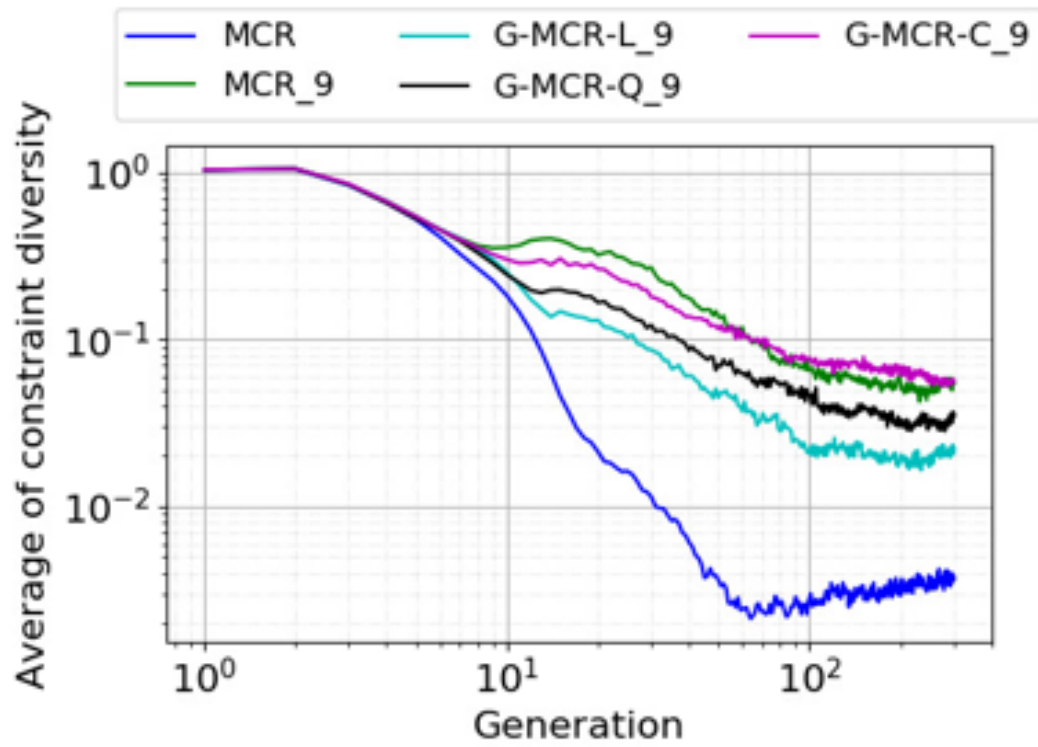


Figure 4.6: Average constraint diversity of MCR, MCR_9, G-CMR-L_9, G-CMR-Q_9, and G-CMR-C_9 in car structure design problem.

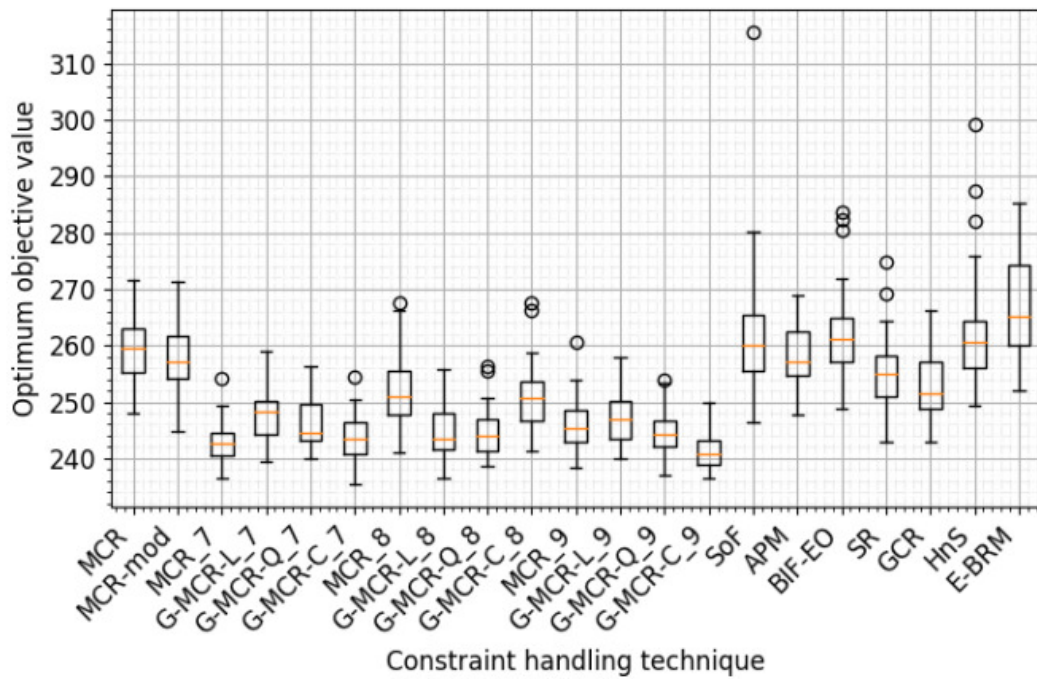


Figure 4.7: Box plot of MOPTA 2008 problem.

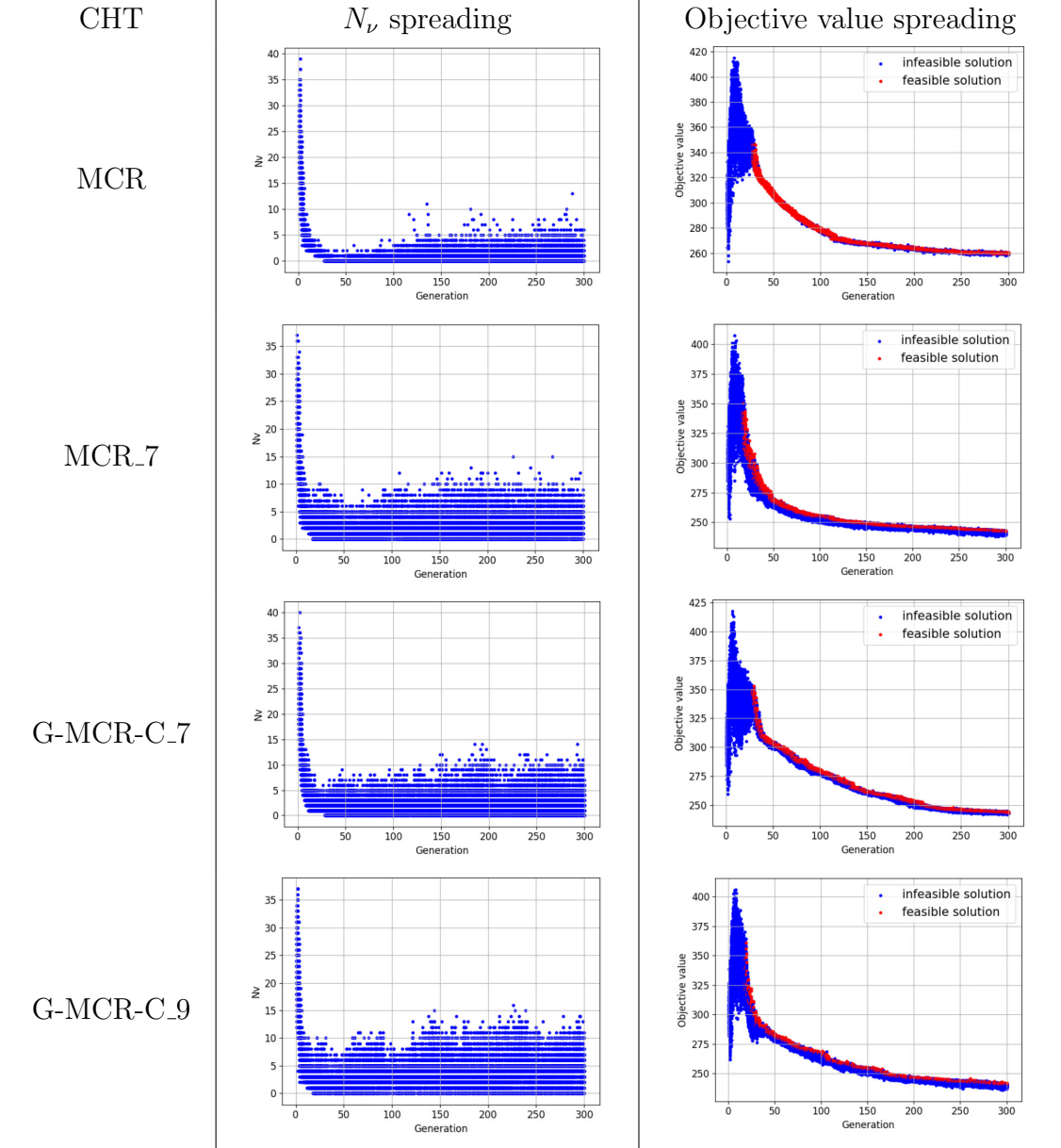


Figure 4.8: N_ν and objective value spreadings of selected CHTs (median of 51 runs) on MOPTA 2008 problem

infeasible individuals as well as many infeasible individuals with lower objective values than the feasible ones. We confirm this by observing Fig. 4.8. In the middle column, we can notice that MCR_7, G-MCR-C_7, and G-MCR-C_9 can produce infeasible individuals violating about 10 or more constraints, while only around five or six for MCR. In the right column, MCR_7, G-MCR-C_7, and G-MCR-C_9 produce many blue dots under the red dots, while there is only a little on MCR. The CHT implementing circular β_1 also tends to have larger diversity than the ones implementing quadratic and linear β_1 s based on Fig. 4.9.

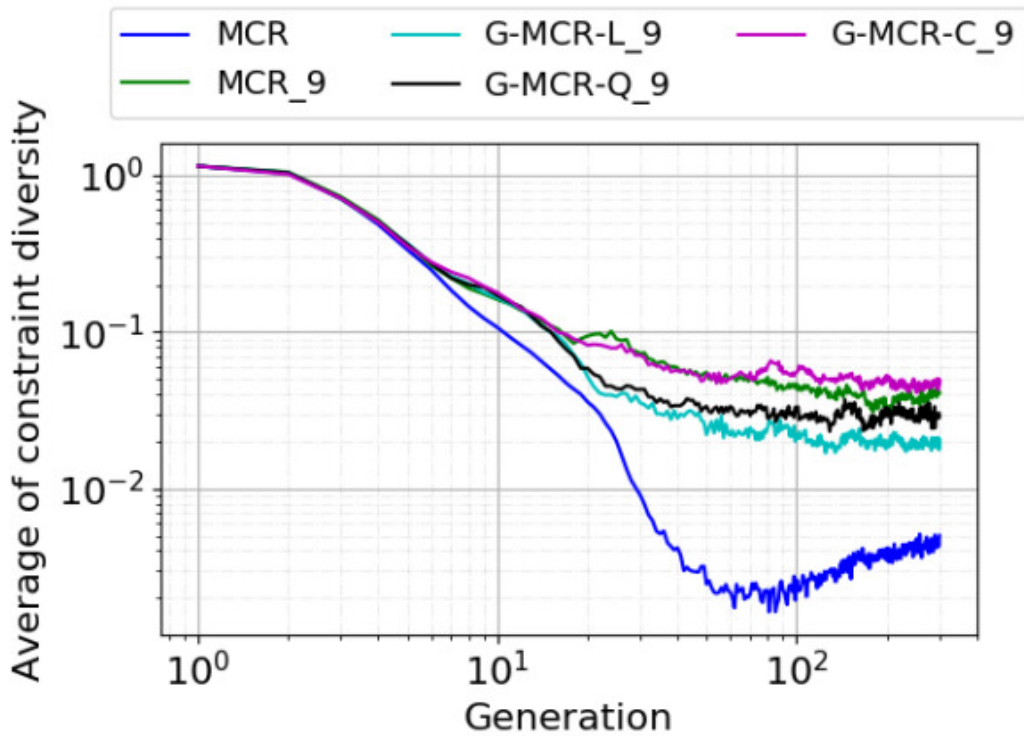


Figure 4.9: Average constraint diversity of MCR, MCR_9, G-CMR-L_9, G-CMR-Q_9, and G-CMR-C_9 in MOPTA 2008 problem.

c. Wind Turbine

Fig. 4.10 depicts the box plot of obtained optimum values of each CHT for wind turbine problem. Now, the trend seems to be different than the previous problems. It seems that many CHTs including MCR almost achieve the global optimum in this problem because the difference of the obtained optimum values tends to be small. Despite that, except for MCR_8, G-MCR-C_8, and MCR_9, most of the G-MCR-based CHTs still perform better than MCR, MCR-mod, SoF, and most of the CHTs in Category B. GCR seems to perform competitive with the best G-MCR-based CHTs in this problem. We ensure

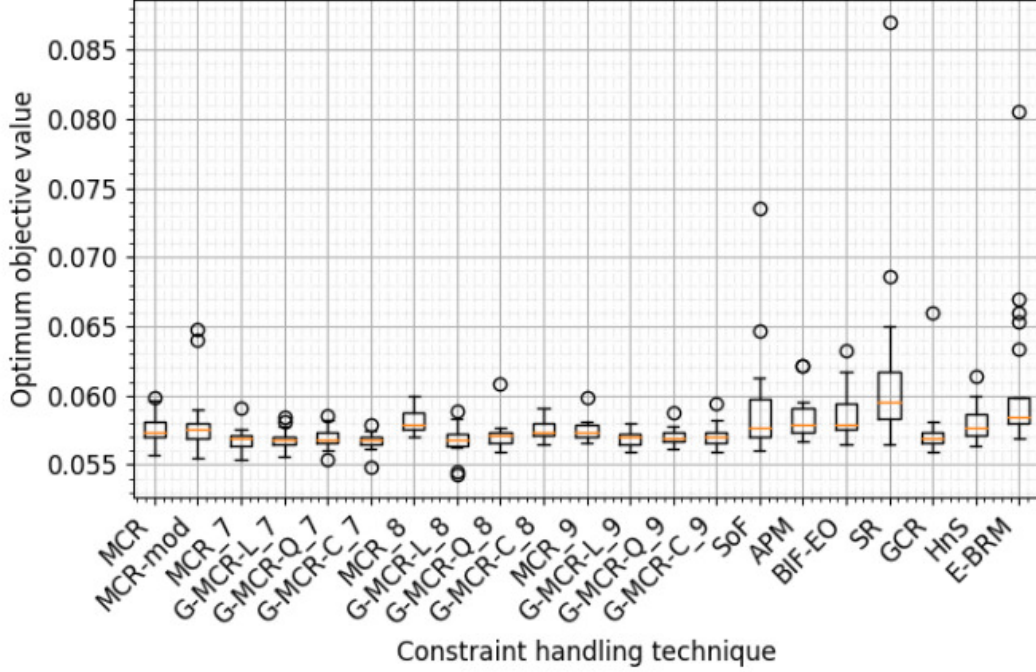


Figure 4.10: Box plot of wind turbine problem.

this observation by significance test presented in Table 4.7 column "wind turbine". There are quite many CHTs perform the best such as MCR_7, G-MCR-L_7, G-MCR-Q_7, G-MCR-C_7, G-MCR-L_8, G-MCR-Q_8, G-MCR-L_9, G-MCR-C_9, and GCR by obtaining 11 scores each.

Since there are many CHTs competitively producing good convergence performances, we select some CHTs to analyze the effects of the modifications. It seems that for G-MCR-based CHTs, the convergence performance can be generated because of similar effect observed in the car structure design and MOPTA 2008. Represented by MCR_7, G-MCR-C_7, and G-MCR-C_9, we can notice in the middle column of Fig. 4.11 that they produce infeasible individuals violating around seven to eight constraints, while only around four or five for MCR. MCR_7, G-MCR-C_7, and G-MCR-C_9 also produce many infeasible individuals with lower objective values than the feasible ones, as shown in the right column of Fig. 4.11. As for GCR, we hardly notice any difference with MCR in the observation on N_v and objective value spreadings. Probably, it is because the user-defined coefficient of 0.45 happens to be the most suitable weight in this problem. The CHT implementing circular β_1 also tends to have larger diversity than the ones implementing quadratic and linear β_1 s based on Fig. 4.12.

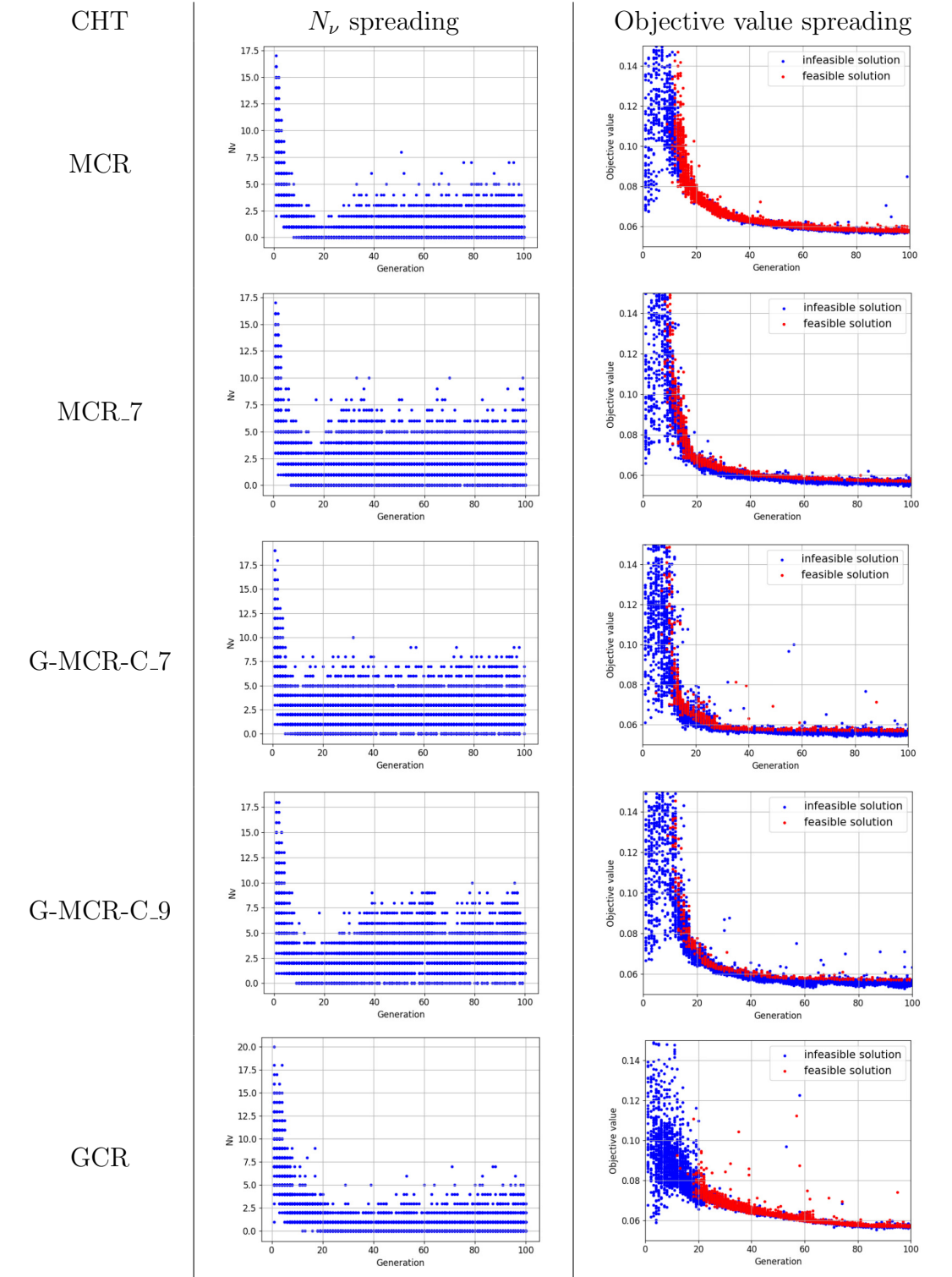


Figure 4.11: N_v and objective value spreadings of selected CHTs (median of 51 runs) on wind turbine problem

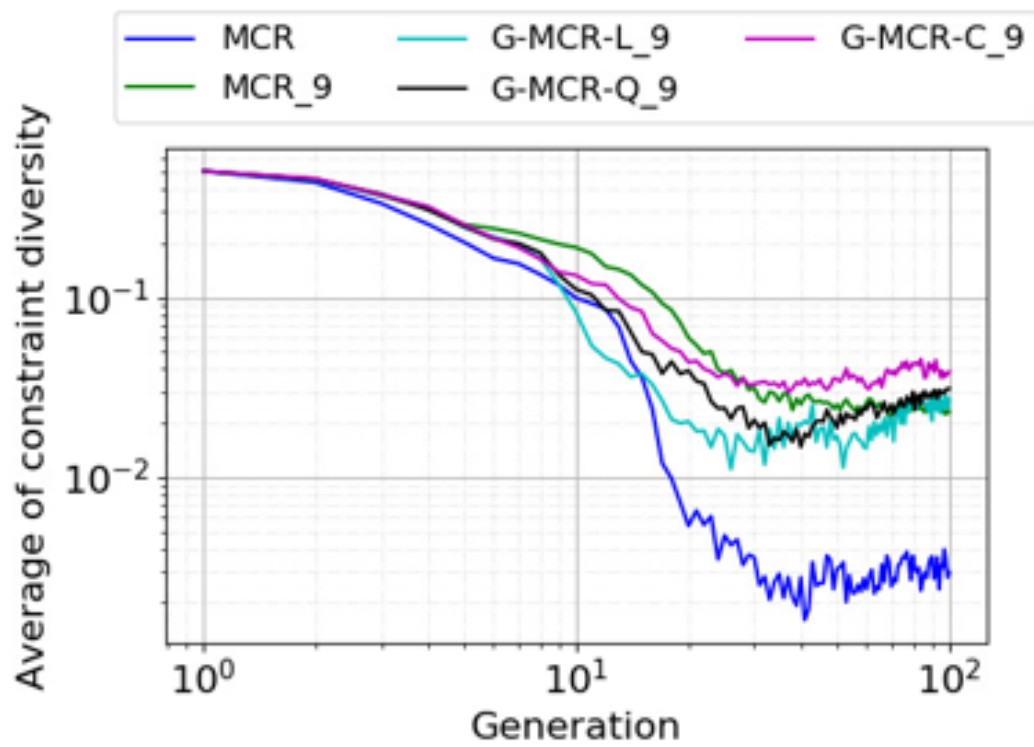


Figure 4.12: Average constraint diversity of MCR, MCR_9, G-CMR-L_9, G-CMR-Q_9, and G-CMR-C_9 in wind turbine problem.

4.4.3 Discussion regarding the feasibility ratio in offspring population in RWDPs

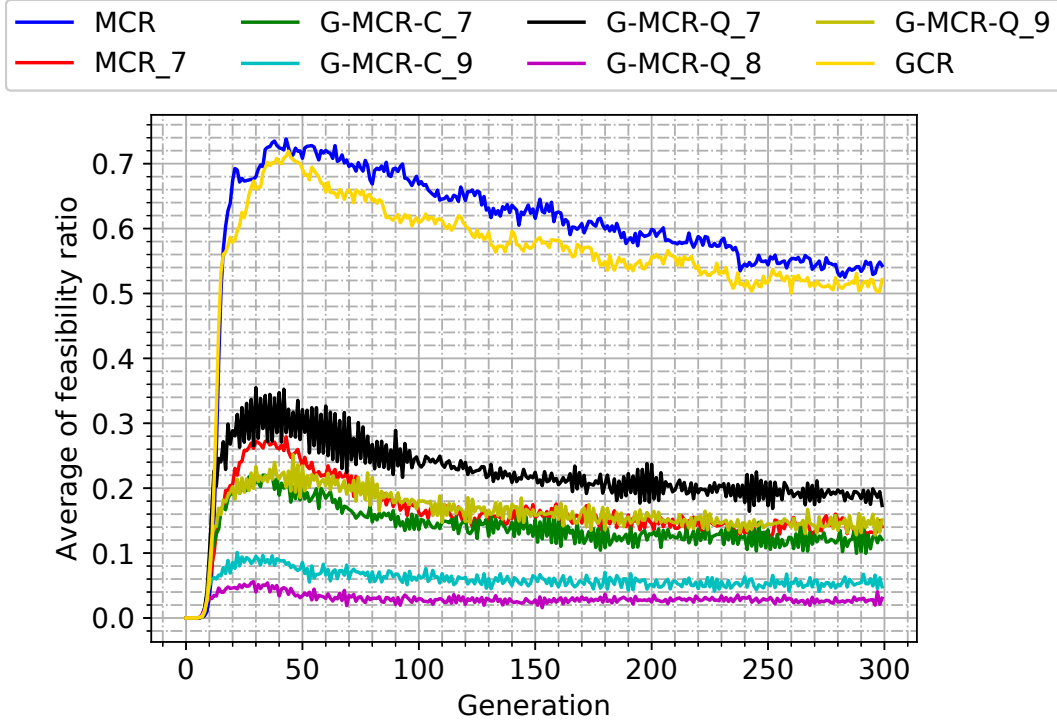


Figure 4.13: Average feasibility ratio of offspring population over the generation on car structure design problem from 51 independent runs (selected CHTs).

Based on the discussion in subsections 4.4.1 and 4.4.2, we obtain that G-MCR-C_9 is one of the most robust CHTs in general problems. It seems that G-MCR-C_9 is also the most robust CHT in real-world design problems because it becomes the best performer in car structure design and MOPTA 2008 problems, and one of the best performers in wind turbine problem. Other than G-MCR-C_9, there are also some CHTs which performs quite robust such as MCR_7, whose total score in the three RWDPs is the second best (from Table 4.7, the total score is $14 + 17 + 11 = 42$), and G-MCR-C_7, which is the other robust CHT on overall problem (from subsection 4.4.1) and performs quite robust in RWDPs.

If we observe the objective value spreading of those CHTs (from Figs. 4.5, 4.8, and 4.11), they tend to obtain rather low, but not very low number of feasible individuals in the offspring population during the search. Therefore, now we observe in more detail the average feasibility ratio generated by the CHTs during the search. The feasibility ratio is

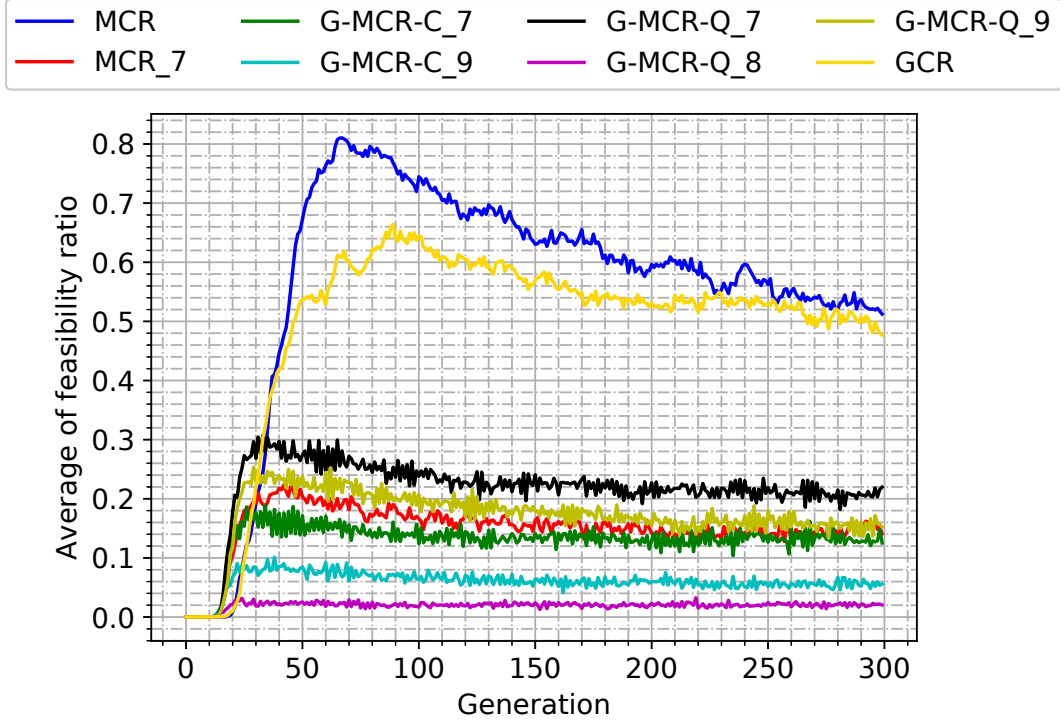


Figure 4.14: Average feasibility ratio of offspring population over the generation on MOPTA 2008 problem from 51 independent runs (selected CHTs).

defined as,

$$\text{feasibility ratio} = \frac{N_{feas,offspring}}{N_{pop,offspring}} \quad (4.9)$$

where $N_{feas,offspring}$ and $N_{pop,offspring}$ stand for the number of feasible individuals in the offspring population and the offspring population size, respectively. For easier comparison, we compare the histories of average feasibility ratio of some selected CHTs from 51 independent runs. We select MCR and CHTs with total score of 20 or more from Table 4.3 (MCR_7, G-MCR-Q_7, G-MCR-C_7, G-MCR-Q_8, G-MCR-Q_9, G-MCR-C_9, and GCR).

Observing the average feasibility ratios of CHTs on car structure design problem depicted in Fig. 4.13, the G-MCR-C_9 obtains the best convergence performance with only around 10% average feasibility ratio in the early generations then decreases to 5% in the late generations, as shown in Fig. 4.13. Almost similar patterns also occur in MCR_7 and G-MCR-C_7. In the early generations, MCR_7 and G-MCR-C_7 have higher average feasibility ratio (around 20%-26%), but both decrease to around 12%-15% at the late generations. We also find similar tendencies in MOPTA 2008 and wind turbine problems, such as shown in Figs. 4.14 and 4.15. The CHTs which generate the best and robust

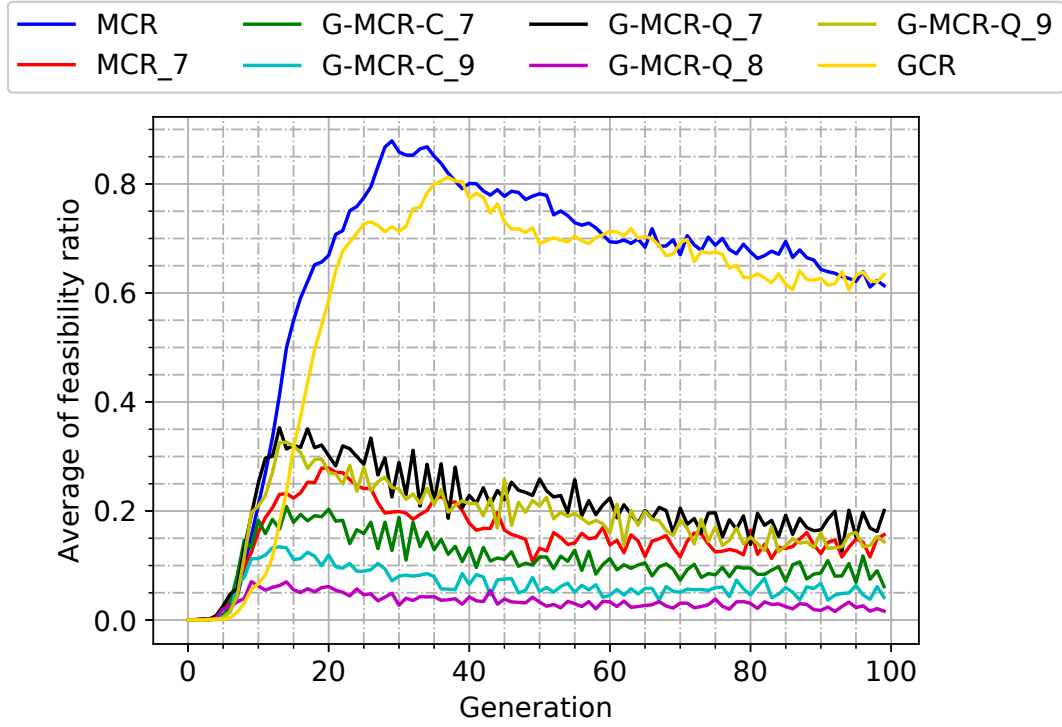


Figure 4.15: Average feasibility ratio of offspring population over the generation on wind turbine problem from 51 independent runs (selected CHTs).

convergence performances tend to have average feasibility ratio of 10%-30% in the early generations and around 5%-15% in the late generations.

In the early generations, the CHTs tend to collect feasible individuals because of small N_{feas} . Meanwhile, from the middle generations to the late, the CHTs tend to focus on finding constrained optimum. This might be the reason why the average feasibility ratios in the early generations tend to be bigger and then decreases in the late generations. That range of average feasibility ratios from middle to late generations (5%-15%) might be the best environment for solving optimization of real-world design problems, at least for three problems investigated in this work.

4.5 Summary

In this chapter, we propose some modifications of G-MCR, the already modified version of MCR in handling the constraints on real-world design optimization problems as well as general problems. The purpose of the modifications is to improve the balance of search in feasible and infeasible regions in every generation so that the convergence performance can be more robust. The modifications are conducted by combining the linear β_1 of MCR-mod

as well as two other proposed β_1 , the circular β_1 and quadratic β_1 , to the G-MCR.

Based on the investigation on 78 benchmark problems with significance test, we obtain some proposed CHTs such as G-MCR-C_9 and G-MCR-C_7 produce the most robust convergence performance in general problems. These two CHTs as well as MCR_7 are also some of the most robust CHTs for solving real-world design problems which have many constraints, small feasible region, and active constraints. In more detailed investigation on the real-world design problems, we also obtain that other than having similar observed effects from the previous chapter (i.e., producing more diverse infeasible individuals and many infeasible individuals with better objective values), the offspring population's feasibility ratio of 5%-15% from the middle to late generations seems to be the best environment for conducting optimization. G-MCR-C_9 is recommended as the CHT for real-world design problems because the performances of G-MCR-C_7 and MCR_7 might be sensitive to number of constraints, since they do not apply any proportionation.

Chapter 5

Airfoil Aerodynamic Design Optimization

5.1 Introduction

Airfoil is a two-dimensional section of an aircraft's wing. In aeronautics, airfoil design is one of the most important optimization problems. It contributes to obtaining the basic aerodynamic performance of an aircraft such as finding the best trade-off between lift and drag coefficients (C_l and C_d , respectively) so that the aerodynamic efficiency ($\frac{C_l}{C_d}$) can be optimum.

While there are many previous works utilized EA (and other global optimization methods) in airfoil design, there is no previous work yet comparing the performance of different CHTs directly on the problem to the best of our knowledge. EA and other methods are often utilized as tools to obtain optimum shape. After an optimum shape is found, the discussion often goes directly to comparing the aerodynamic performance of the baseline and optimum designs without comparing the optimization performance with other CHTs, let alone analyzing the reason why the CHT is capable of producing such optimum solution. One possible reason that hinders the previous works from comparing different CHT performances in airfoil design problem is because the problem is often computationally expensive due to the usage of computational fluid dynamics (CFD) software in obtaining the aerodynamic properties.

The type of CHT has been utilized in the previous work is the conventional one, which favors feasible individuals over the infeasible ones and does not consider the difference of orders of magnitudes on the functions [75–77]. Meanwhile, we have observed in the previous chapters that CHTs with balanced search in both feasible and infeasible regions and balanced assessment of functions with different orders of magnitudes tend to have better convergence performance compared with the conventional ones. Therefore, we

believe that the convergence performance of airfoil design optimization with EA can still be improved with our developed CHTs.

In this chapter, we conduct optimization on two airfoil design problems:

1. transonic airfoil design and
2. multi-point airfoil design.

We compare the performances of five CHTs: MCR, SoF, and the three most robust CHTs we obtained from the discussion preceding chapter: MCR_7, G-MCR-C_7, and G-MCR-C_9. We also investigate SoF as the representative of conventional CHTs (Category A in Chapter 2).

5.2 Airfoil Parameterization: Cubic B-Spline Curve

We parameterize the airfoil with cubic B-spline curve [78]. Suppose there is a set of $h + 1$ control points \mathbf{P}_i , $i = 0, \dots, h$ a B-spline curve \mathbf{C} is defined as follows,

$$\mathbf{C} = \mathbf{C}(u) = \sum_{i=0}^h N_{i,p}(u) \mathbf{P}_i \quad (5.1)$$

where p denotes the degree of the B-spline curve. For cubic B-spline, $p = 3$. $N_{i,p}(u)$ is the basis function, expressed as follows,

$$N_{i,0}(u) = \begin{cases} 1, & \text{if } t_i \leq u < t_{i+1}, \\ 0, & \text{otherwise.} \end{cases} \quad (5.2)$$

$$N_{i,p} = \frac{u - t_i}{t_{i+p} - t_i} N_{i,p-1}(u) + \frac{t_{i+p+1} - u}{t_{i+p+1} - t_{i+1}} N_{i+1,p-1}(u) \quad (5.3)$$

where \mathbf{t} is a set of $m = h + p + 1$ knots, defined as

$$t_0 = t_1 = \dots = t_p = 0, \quad (5.4)$$

$$t_j = \frac{j}{h - p + 1} \text{ if } j = 1, \dots, h - p, \quad (5.5)$$

$$t_{m-p} = t_{m-p+1} = \dots = t_m = 1. \quad (5.6)$$

Fig. 5.1 illustrates a generation of airfoil with cubic B-spline curve with $h + 1 = 9$ control points. The first control point \mathbf{P}_0 is set at the trailing edge. The next control points are placed in clockwise direction sequentially from the lower surface ($\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3$), leading edge (\mathbf{P}_4), upper surface ($\mathbf{P}_5, \mathbf{P}_6, \mathbf{P}_7$), and then go back to the trailing edge (\mathbf{P}_8). The trailing edge control points (\mathbf{P}_0 and \mathbf{P}_8) are fixed at $(\mathbf{x} = \{1, 0\}^T)$, while at leading edge, \mathbf{P}_4 is fixed at $(\mathbf{x} = \{0, 0\}^T)$. The other six control points are set active so

that different setting of them will generate different shape of airfoil. First, the control points will generate the red curve with Eq. (5.1). We can notice that the length L of the red airfoil is less than one and the leading edge location might not be exactly on the origin. Therefore, we then rotate and dilate the red curve to the blue curve to ensure that the leading edge of the airfoil is on the origin and the chord length c is always one. Suppose $\mathbf{C}(u) = \{x(u), y(u)\}$, we conduct the rotation to $\mathbf{C}_r(u) = \{x_r(u), y_r(u)\}$ as follows,

$$x_r(u) = 1 + (x(u) - 1) \cos(\pi - \alpha) - y(u) \sin(\pi - \alpha) \quad (5.7)$$

$$y_r(u) = (x(u) - 1) \sin(\pi - \alpha) + y(u) \cos(\pi - \alpha) \quad (5.8)$$

where α is the angle between the chord line of the red airfoil and line P_0P_4 . As for the dilatation,

$$x_d(u) = \frac{x_r(u) - x_r(u)|_{@LE}}{L} \quad (5.9)$$

$$y_d(u) = \frac{y_r(u) - y_r(u)|_{@LE}}{L} \quad (5.10)$$

where $x_d(u)$ and $y_d(u)$ are the dilated coordinates. $x_r(u)|_{@LE}$ and $y_r(u)|_{@LE}$ are the coordinates of the red airfoil's leading edge and L is its chord length.

5.3 CFD Solver: LANS3D

LANS3D [79–81] is an in-house CFD software developed by ISAS/JAXA. It stands for "LU-ADI [82–85] Navier-Stokes code for three-dimensional flows". The software is capable of simulating either engineering problem [86] or physical problem [87] efficiently and accurately.

5.4 Transonic Airfoil Design Optimization Problem

In this problem, the airfoil shape is optimized under transonic flow regime, where the Mach number (M) is almost one. Under this condition, shockwave might appear because the local flow speed on the airfoil's upper surface reaches $M = 1$. The shockwave contributes to an increase of drag due to the wave drag, whose value is considerable large. In order to reduce this unfavorable effect, shape optimization is often conducted so that the shockwave does not appear.

5.4.1 Problem Definition and Optimization Setting

In the present work, we consider a single-point transonic airfoil shape optimization [76], where the optimization is conducted in one condition of Mach number (M), Reynolds

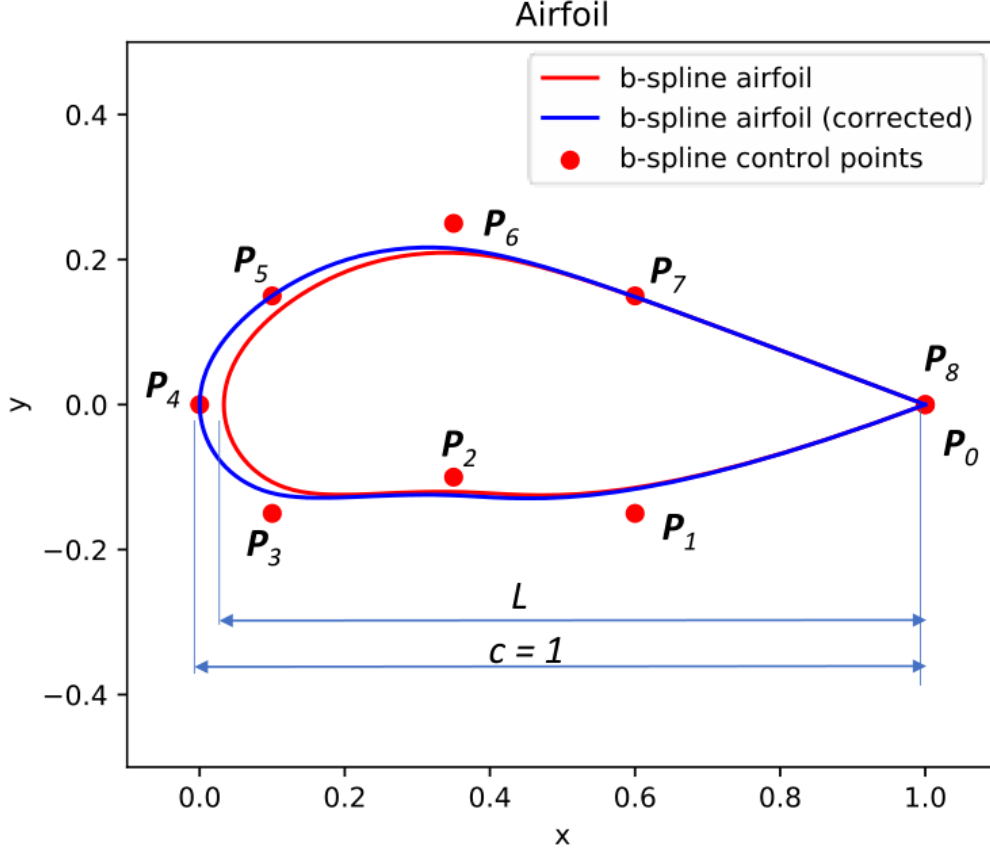


Figure 5.1: Illustration of airfoil generation with cubic B-spline curve.

number (Re), and angle-of-attack (AoA). The problem is defined as follows,

$$\begin{aligned}
 &\text{Minimize} && C_d(\mathbf{x}) \\
 &\text{subject to} && g_1(\mathbf{x}) = C_{l_{ref}} - C_l(\mathbf{x}) \leq 0 \\
 & && g_2(\mathbf{x}) = C_{m_{ref}} - C_m(\mathbf{x}) \leq 0 \\
 & && g_3(\mathbf{x}) = A_{ref} - A(\mathbf{x}) \leq 0 \\
 & && \mathbf{x}_L \leq \mathbf{x} \leq \mathbf{x}_U
 \end{aligned} \tag{5.11}$$

where C_d , C_l , C_m , and A denote the drag coefficient, lift coefficient, pitching moment coefficient at the quarter chord, and area of the airfoil, respectively. Symbol \mathbf{x} denotes a solution vector, which consists of 18 cubic B-spline curve control points (excluding the control points that coincide with the leading and trailing edges) to generate an airfoil.

We set RAE2822 airfoil as the baseline design. We set the lower bound (\mathbf{x}_L) and upper bound (\mathbf{x}_U) of the search space as the ± 0.02 perturbation of baseline's upper and lower surfaces' control points in the Y-axis as illustrated in Fig. 5.2. Except for the four points near trailing edge, to maintain a sharp trailing edge, we set the two points nearest the trailing edge with ± 0.0002 perturbation, and the second nearest with ± 0.002

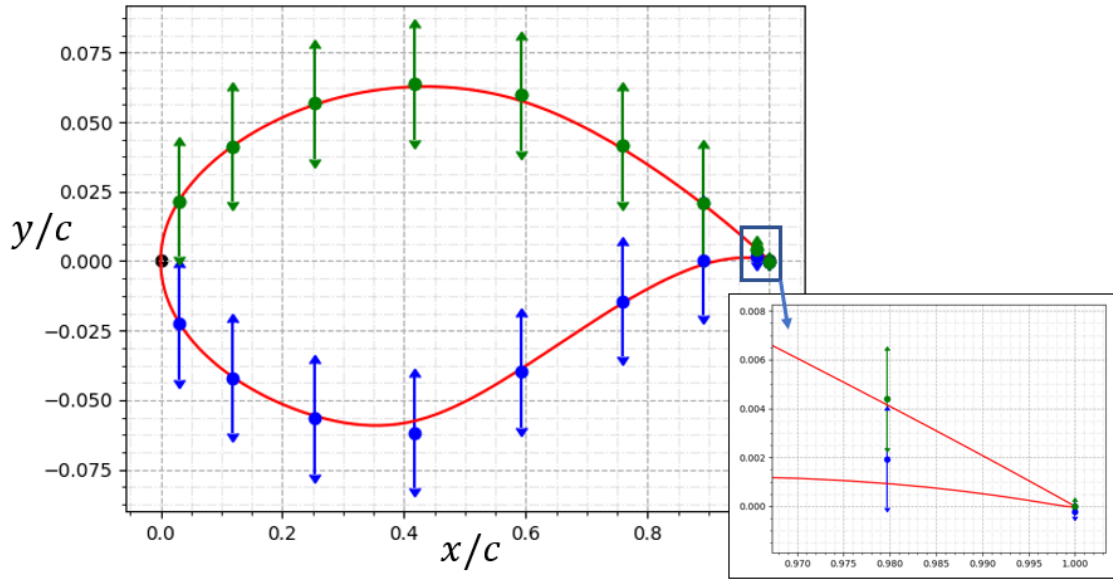


Figure 5.2: Search space for transonic airfoil design optimization

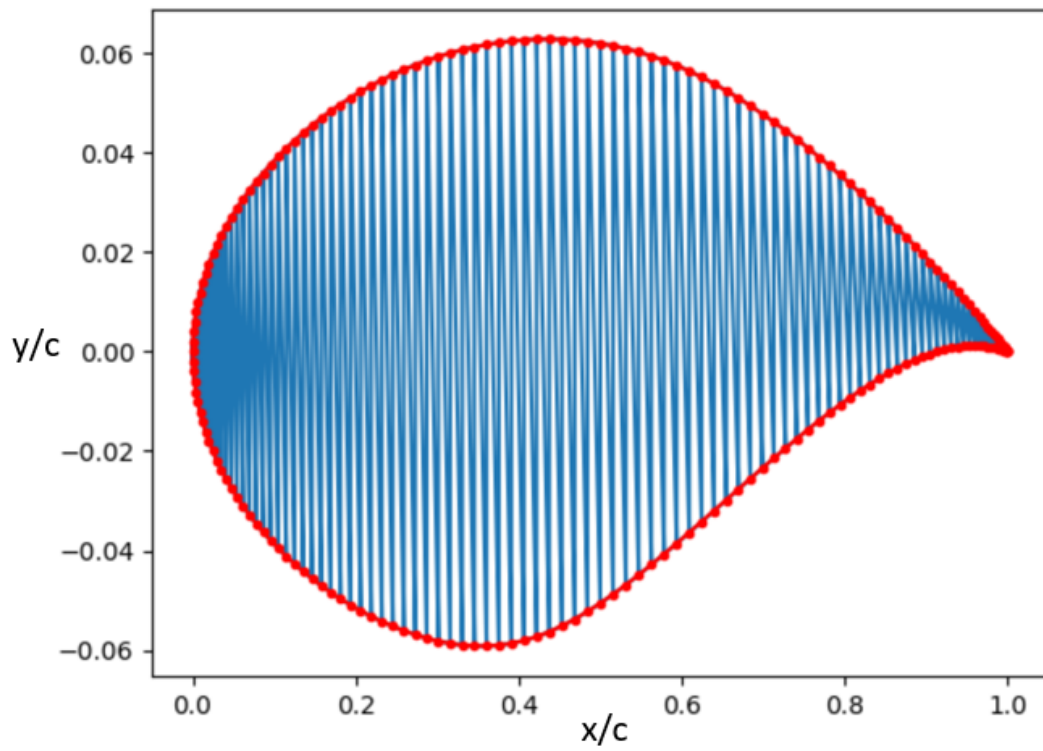


Figure 5.3: Area calculation

perturbation. The control points on leading and trailing edges are fixed at (0,0) for leading edge and at (1,0) for trailing edge.

The aerodynamic properties $C_{l_{ref}}$, $C_{m_{ref}}$, and A_{ref} are the lift coefficient, pitching moment coefficient at the quarter chord, and area of the baseline airfoil, respectively. The flight condition of the optimization is $M = 0.729$, $Re = 6.5 \times 10^6$, and $AoA = 2.31^\circ$. In the said flight condition,

$$C_{l_{ref}} = 0.688729 \quad (5.12)$$

and

$$C_{m_{ref}} = -0.086328. \quad (5.13)$$

The aerodynamic properties are obtained from LANS3D simulation to be explained in the following subsection, together with the LANS3D validation. As for the area of the baseline airfoil, we first discretize the airfoil into triangles, as depicted Fig. 5.3. Then, we approximate the airfoil's area by summing up the area of all triangles. We obtain

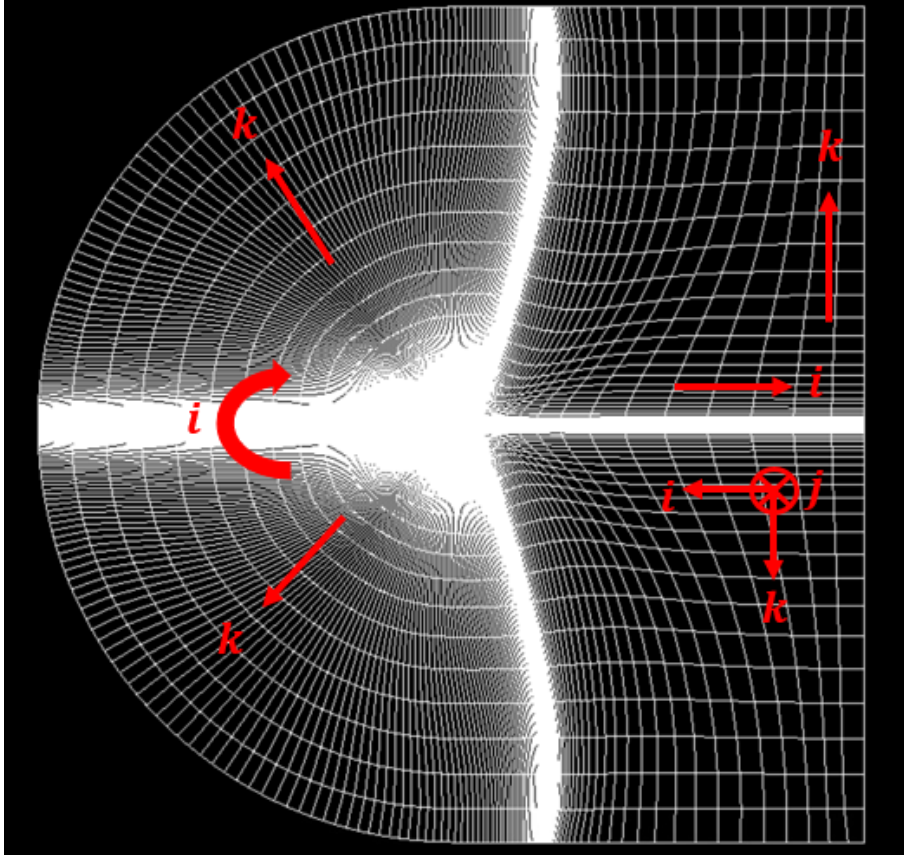
$$A_{ref} = 0.077854675. \quad (5.14)$$

We conduct 11 independent runs for each CHT with 5000 solution evaluations ($N_{pop} = 50$, $N_{gen} = 100$ in the RGA) for each independent run. To enhance the optimization speed, we evaluate the individuals in the population in parallel with multi-processing Pool library provided by Python 3. Since one evaluation with LANS3D requires about eight minutes, one optimization takes about 54 hours to finish with 13 cores.

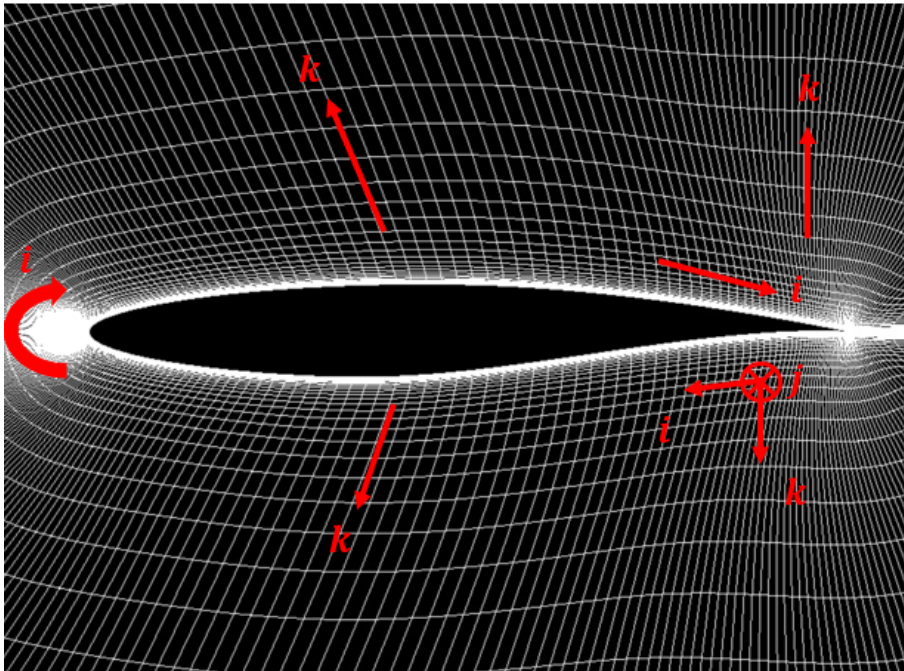
5.4.2 CFD Validation

To validate the accuracy of LANS3D simulation, we compare the aerodynamic properties of the baseline airfoil with the simulation by NPARC (National Program for Applications-Oriented Research in CFD) Alliance [88] and experiment by AGARD (Advisory Group for Aerospace Research and Development) [89]. The flight condition is $M = 0.729$, $Re = 6.5 \times 10^6$, and $AoA = 2.31^\circ$, the same with which set in problem definition. The grids utilized for the simulation is depicted in Figs. 5.4a (farfield grids) and 5.4b. The letters i , j , and k are the coordinate system of the grids. For the i -axis, it follows the opposite direction of streamline for the grids of airfoil's lower surface and the same direction of streamline for the upper surface. As for j -axis, its direction is going outside the picture. As for the k -axis, it is about perpendicular with the i -axis. We set the the number of grid points in i , j , and k as $369 \times 3 \times 65$, respectively, for the simulation. The setting of LANS3D is presented in Table 5.1.

The surface pressure distribution of simulation by LANS3D, simulation by NPARC Alliance, and experiment by AGARD are depicted in Fig. 5.5. As we can observe, LANS3D



(a)



(b)

Figure 5.4: Grid visualizations at (a) full and (b) near wall views of RAE 2822 airfoil

Table 5.1: Aerodynamic properties of RAE 2822 airfoil

| Terms | Method |
|---------------------|--|
| Governing equations | 2D compressible Navier-Stokes |
| Convection term | 3 rd order MUSCL+SHUS |
| Viscous term | 2 nd order central difference |
| Time integration | ADI-SGS |
| Turbulence modeling | Spalart Allmaras |
| Time interval | local time step |

Table 5.2: Aerodynamic properties of RAE 2822 airfoil

| Coefficient | LANS3D | NPARC | AGARD |
|-------------|--------|--------|--------|
| C_l | 0.689 | 0.731 | 0.743 |
| C_d | 0.0131 | 0.0121 | 0.0127 |

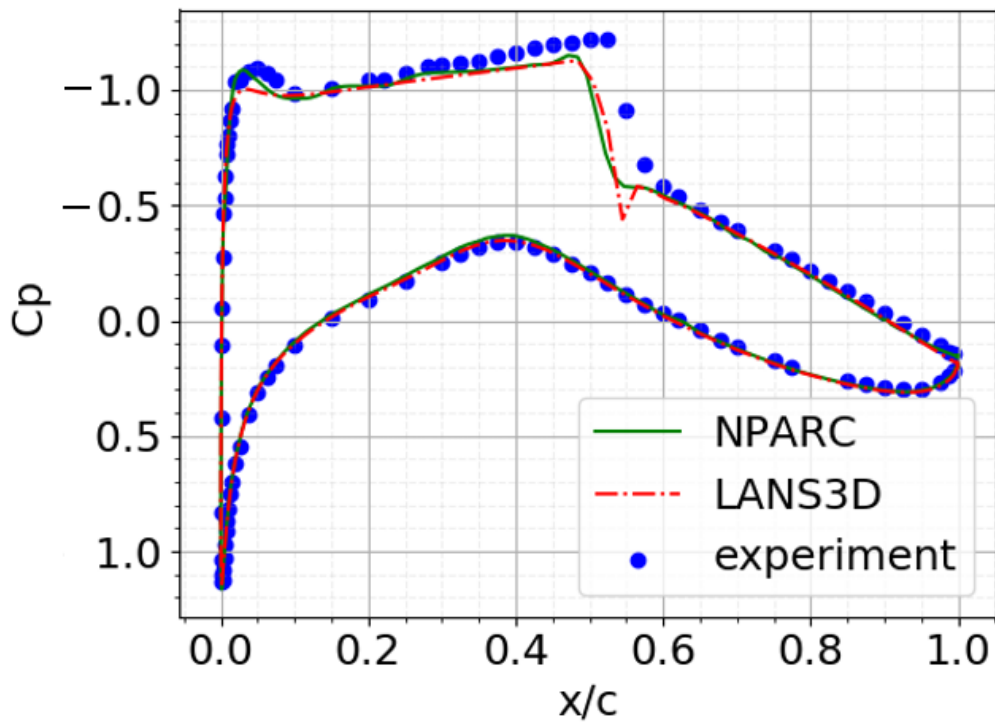


Figure 5.5: C_p distribution of RAE 2822 airfoil from LANS3D and NPARC Alliance simulations and experiment by AGARD

generates slightly smaller C_p near the leading edge and after the shockwave. This might contribute to smaller C_l , as presented in Table 5.2. However, the surface pressure distribution on the lower surface agrees quite nicely with both references on the lower surface. The location of shockwave and the C_d value are also similar. Therefore, we decide that this accuracy is sufficient for optimization.

5.4.3 Optimization Result and Discussion

Fig. 5.6 depicts the box plot of the optimum drag coefficient obtained by each CHT after 11 independent runs. The dashed green line depicts the drag coefficient of baseline airfoil, $C_{d_{ref}} = 0.013077$. We can observe that MCR, MCR_7, G-MCR-C_7, and G-MCR-C_9 produce lower C_d than the baseline for all 11 independent runs. As for SoF, some results are worse than the baseline. We can also observe that MCR_7, G-MCR-C_7, and G-MCR-C_9 obtain significantly better convergence performance than MCR and SoF. This is ensured by the significance test result presented in Table 5.3. This indicates that in this problem, balancing the search from feasible and infeasible regions improves the search efficiency, leading to better convergence rate towards constrained optimum. as for MCR_7, G-MCR-C_7, and G-MCR-C_9, their convergence performances are competitive with each other.

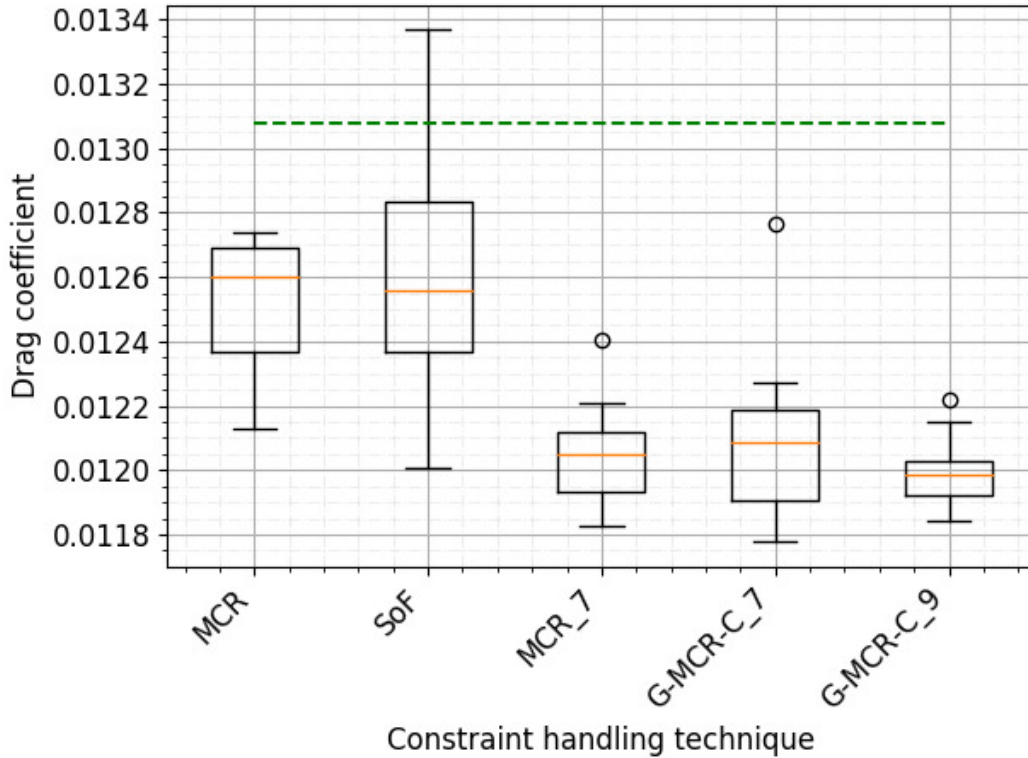


Figure 5.6: Box plot of obtained optimum drag coefficients from 11 independent runs

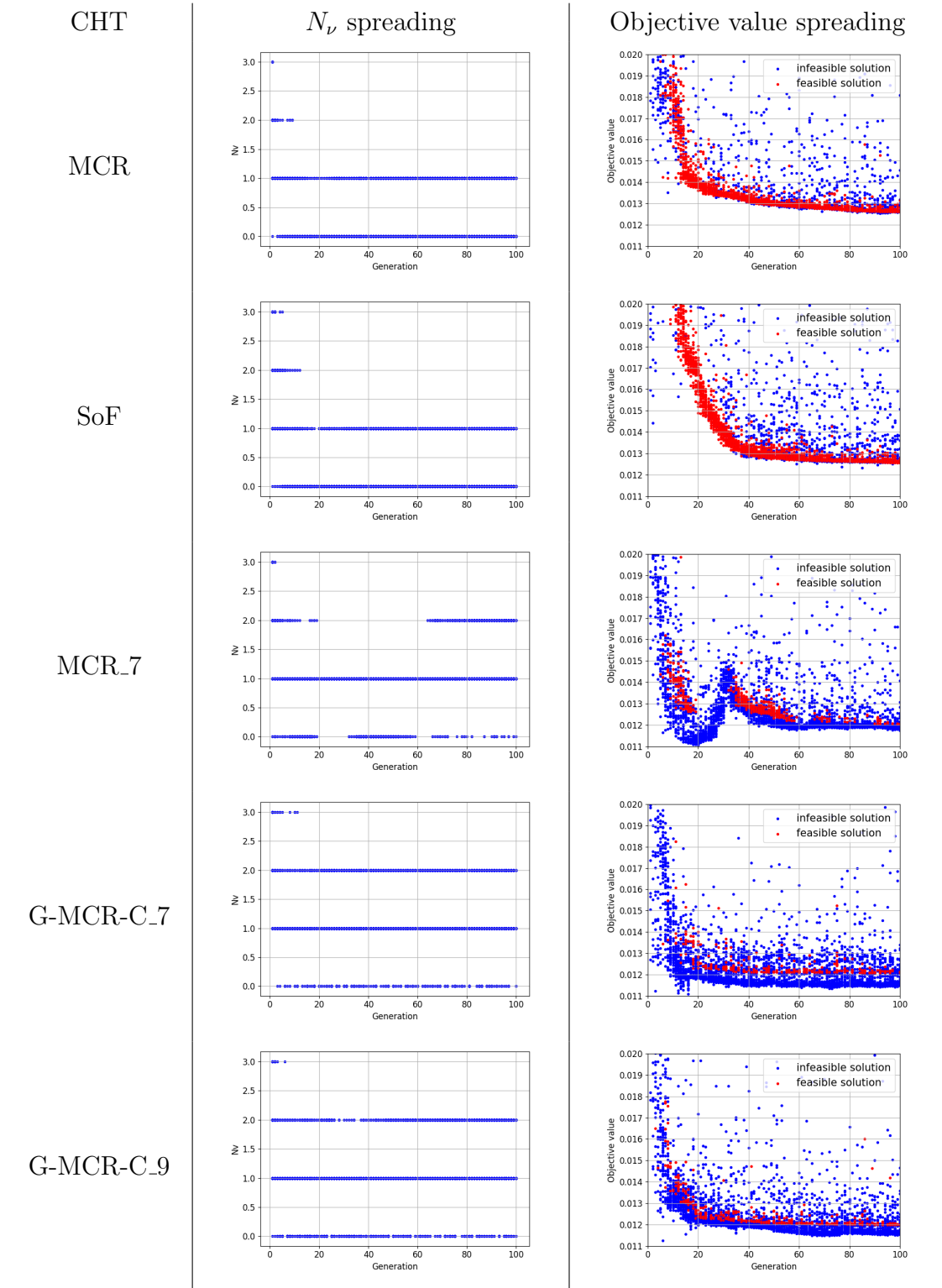


Figure 5.7: N_ν and objective value spreadings of the CHTs (median of 51 runs)

Table 5.3: Statistical significance test for transonic airfoil design

| CHT | MCR | SoF | MCR_7 | G-MCR-C_7 | G-MCR-C_9 |
|---------------|-----|-----|-------|-----------|-----------|
| vs. MCR | | = | + | + | + |
| vs. SoF | = | | + | + | + |
| vs. MCR_7 | - | - | | = | = |
| vs. G-MCR-C_7 | - | - | = | | = |
| vs. G-MCR-C_9 | - | - | = | = | |

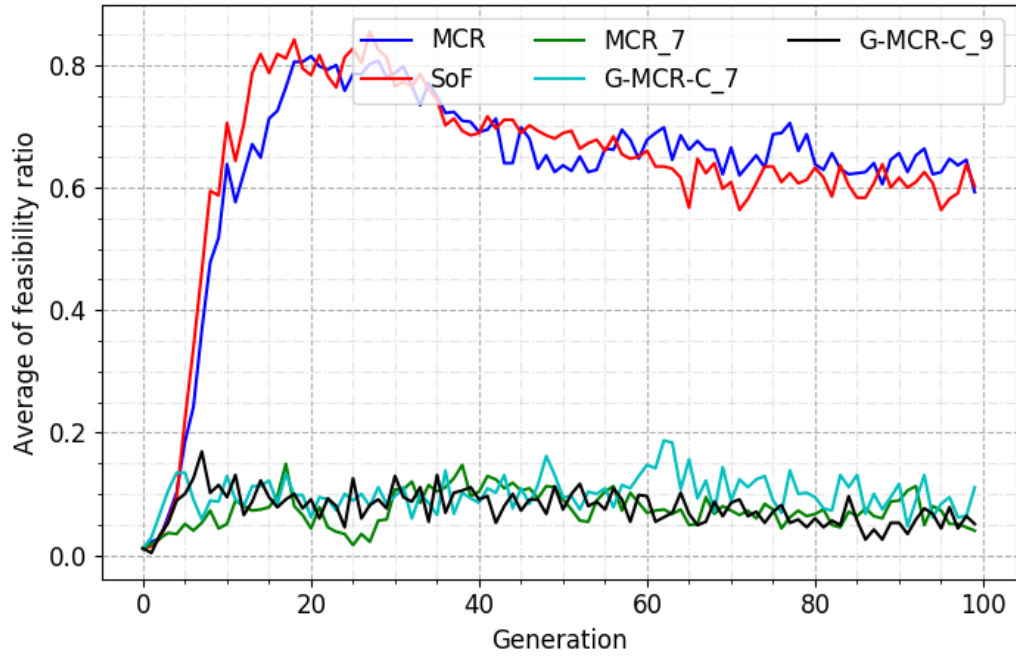


Figure 5.8: Average of feasibility ratio from 11 independent runs

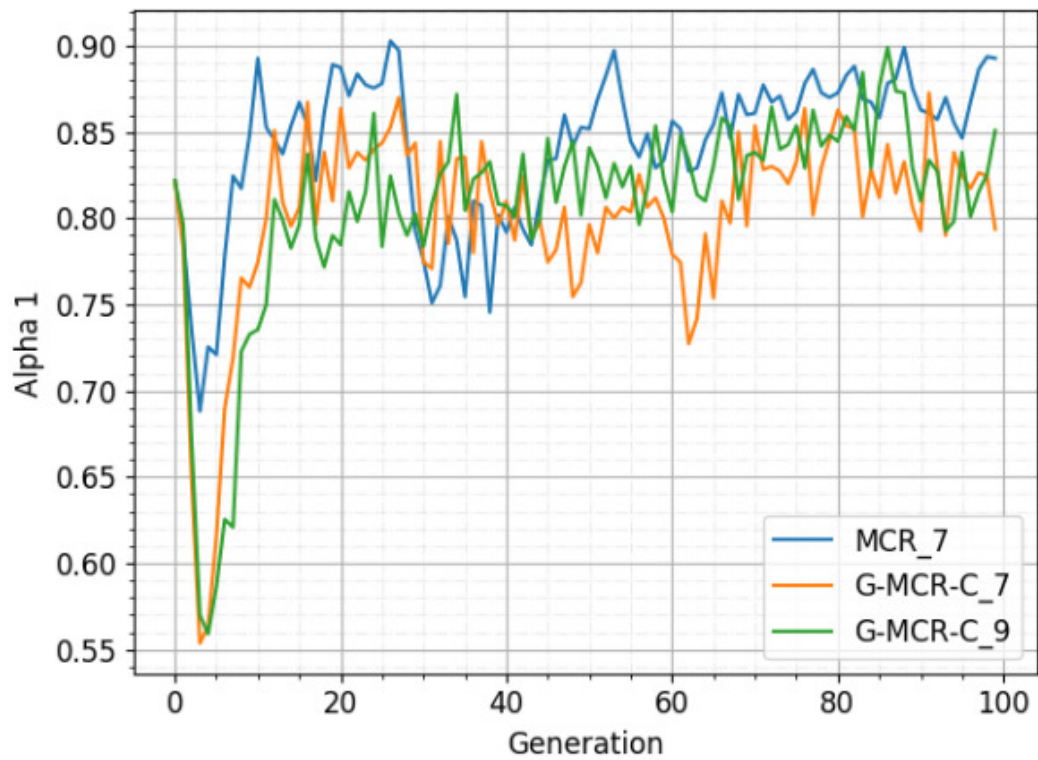


Figure 5.9: Average α_1 of transonic airfoil design from 11 independent runs

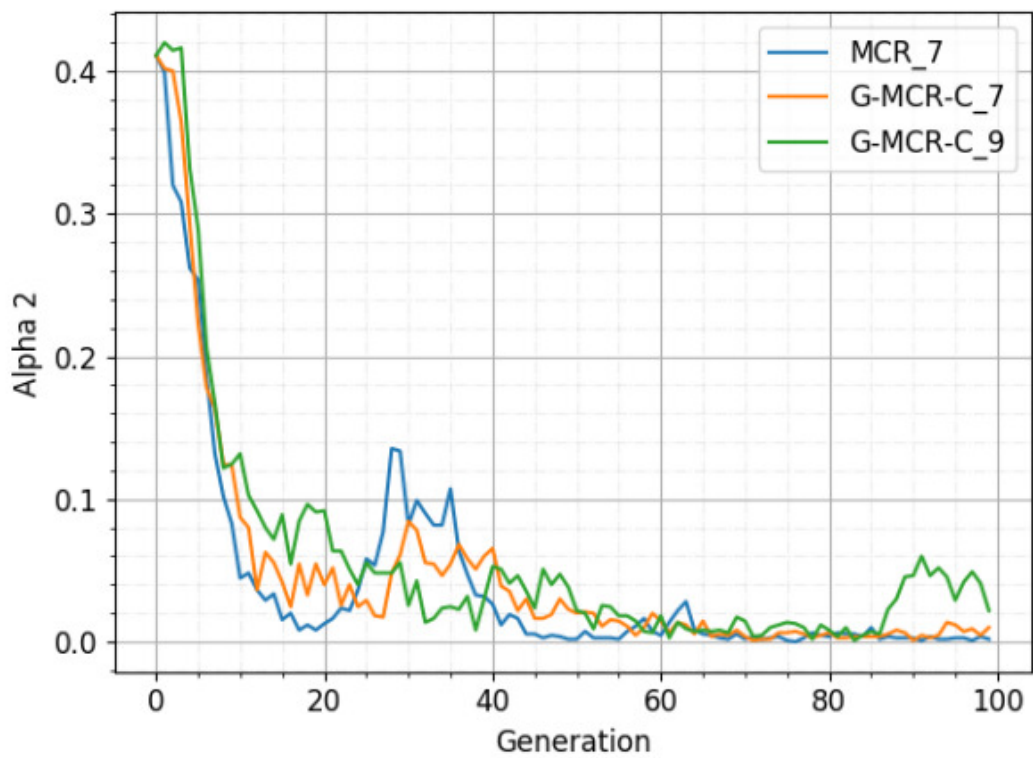


Figure 5.10: Average α_2 of transonic airfoil design from 11 independent runs

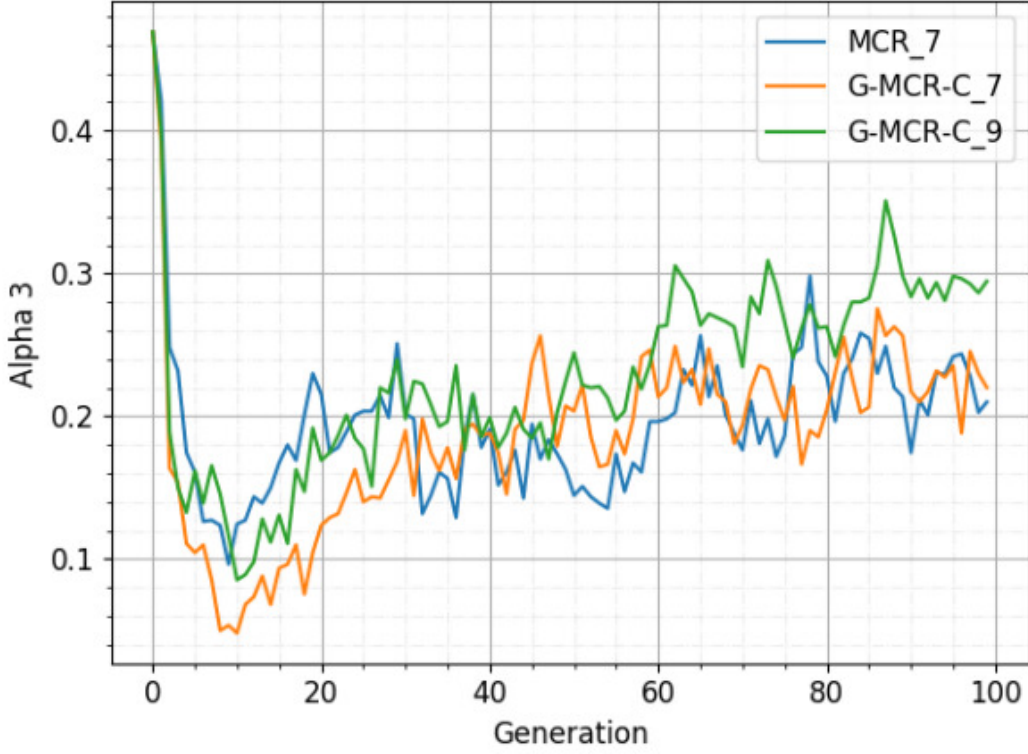
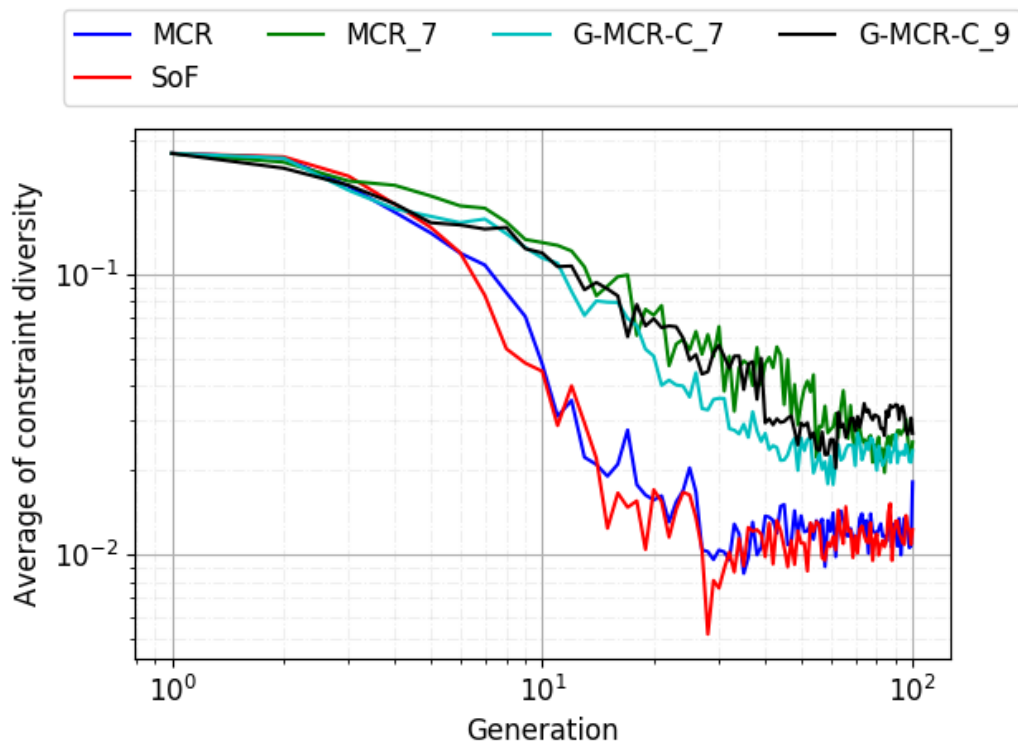


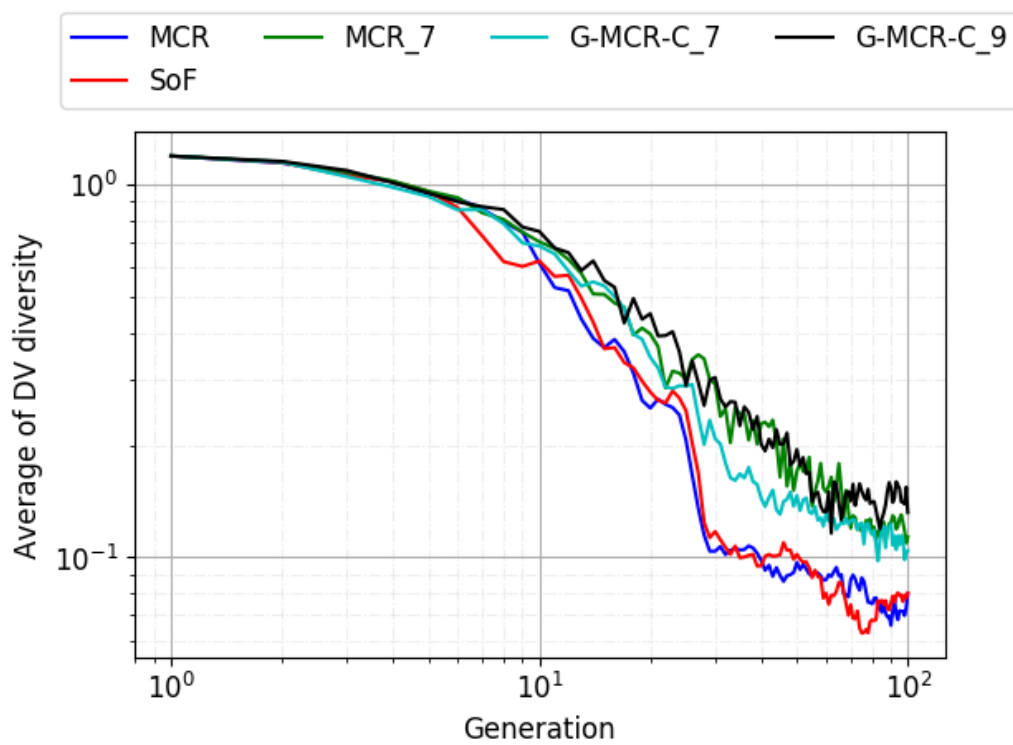
Figure 5.11: Average α_3 of transonic airfoil design from 11 independent runs

From Fig. 5.7, we can also observe that in this problem, MCR_7, G-MCR-C_7, and G-MCR-C_9 consistently produce similar effects as observed in the preceding chapter. Compared with MCR and SoF, which tend to generate infeasible individuals violating only one out of three constraints, MCR_7, G-MCR-C_7, and G-MCR-C_9 produce more variety of infeasible individuals by generating individuals violating two out of three constraints, as shown in the middle column of Fig. 5.7. MCR_7, G-MCR-C_7, and G-MCR-C_9 also produce sufficient infeasible individuals with lower objective values than the feasible ones as shown in the right column of Fig. 5.7. That sufficient number of infeasible individuals corresponds to the average feasibility ratio in the offspring population which is between 5%-15%, as shown in Fig. 5.8.

Observing the histories of average α_i of each constraint in this problem as depicted in Figs. 5.9, 5.10, and 5.11, it seems that the C_l ($g_1(\mathbf{x})$) and area ($g_3(\mathbf{x})$) constraints are severely-active since their corresponding α_i values stay at non-zero values. Especially the C_l constraint, whose average α_1 value is close to one throughout the entire optimization. As for the C_m ($g_2(\mathbf{x})$) constraint, it seems that this constraint tends to be weakly-active because the α_2 value tends to approach zero by the end of optimization. Even though the α_2 value rises a little on G-MCR-C_9 at the late generation, the value is still rather small compared with the other α_i values.



(a)



(b)

Figure 5.12: Average of diversity in (a) constraint and (b) design variable spaces

We also observe that MCR_7, G-MCR-C_7 and G-MCR-C_9 produce relatively more diverse offspring population in the constraint and design variable spaces, as depicted in Figs. 5.12a and 5.12b, respectively. Starting from around the 5th generation, the constraint diversity of MCR and SoF starts to be smaller than MCR_7, G-MCR-C_7, and G-MCR-C_9. It means that MCR and SoF start generating more feasible individuals. It is consistent with the average feasibility ratio depicted in Fig. 5.8. At around the 5th generation, the average feasibility ratio of MCR and SoF start to get bigger than MCR_7, G-MCR-C_7, and G-MCR-C_9, indicating that they start to generate more feasible individuals in the offspring population. It seems that the constraint diversity has an inverse relationship with the average feasibility ratio. When the constraint diversity is small, the average feasibility ratio is big, and vice versa.

On the objective value spreading of MCR_7 (Fig. 5.7, we can notice that the objective values increase at around the 20th-30th generation. After the 30th generation, the feasible individuals found have rather worse objective values than in the 20th generation. In this range of generation, it seems that the search loses all its feasible individuals found in the combined population (combination of parent and offspring populations for selecting new parent population). This makes $\beta_1 = 0$, thus the search fully focuses on finding feasible individuals. In consequence, the objective value history increases until a new feasible individual is found. After it is found again, the objective value history starts to decrease again. This can happen because when all infeasible individuals have better fitness value $F(\mathbf{x})$ than feasible ones and MCR_7 is not equipped with a mechanism to always preserve at least one feasible individual in the population. While we only observe this in MCR_7, it is also possible to happen to other G-MCR-based CHTs because they are also not equipped with a mechanism to preserve at least one feasible individual.

In our opinion, this phenomenon might be both unfavorable or favorable. It might be unfavorable because we lose some precious generations only to find worse feasible individual. It might be favorable if the problem has more than one separated feasible regions. If the search is obliged to keep at least one feasible individual in the population, the infeasible individuals might still be pulled in the region in which the feasible individual exists, thus slows or hinders the exploration to other regions. If the search do not have any obligation to keep at least one feasible individual, it might be easier to explore different regions because there is no interaction with the feasible individual. Therefore, there is also a possibility that the region in which the global optimum exists can be found faster.

5.4.4 Analysis on the Shape of Optimum Designs and the Flow Fields

In this subsection, we compare the shape of baseline and optimum airfoils. We analyze the surface pressure distribution and the flow field to learn why the optimum designs can produce smaller drag coefficients than the baseline. We select the median and best results for the comparison.

a. Median Result

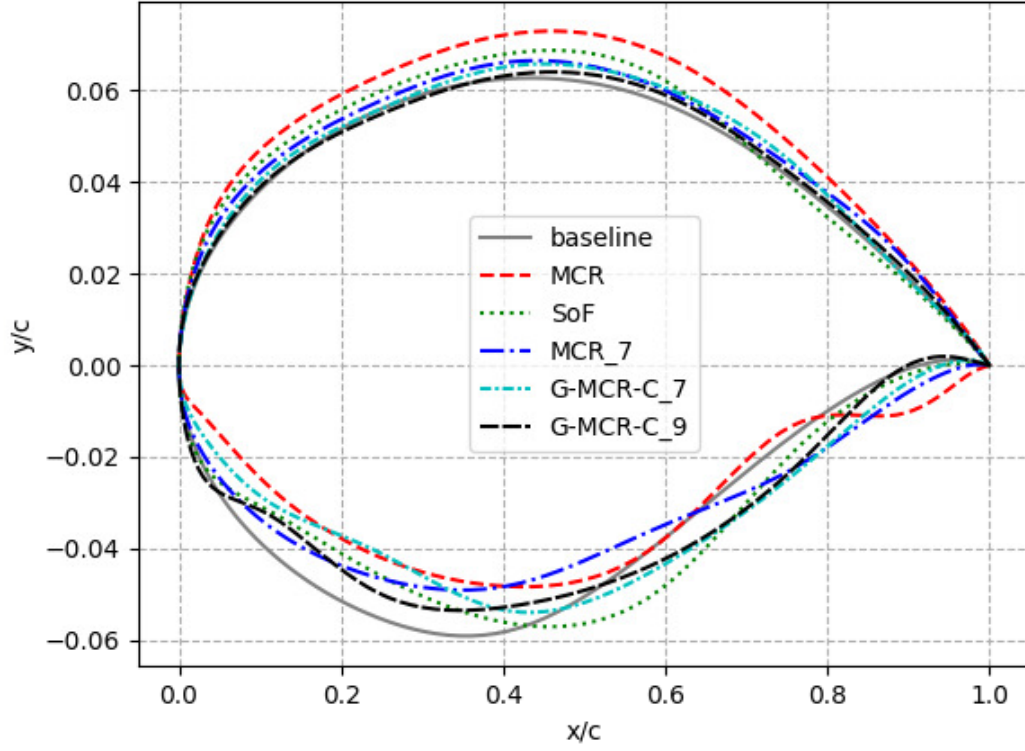
In median result, the airfoil shapes and their C_p distributions are depicted in Figs. 5.13a and 5.13b, respectively. Despite their different shapes, it seems that the airfoils produced by all CHTs generate weaker shockwave on the upper surface, which might be the largest contributor of the drag reduction. The optimum shapes can weaken the shockwave most probably because they have larger curvature radius around the area where the shockwave occurs. This produces the most negative C_p moving closer to the leading edge, indicating that the Mach number reaches sonic condition faster in that area, thus urging the shockwave to occur before the maximum thickness, thus weakening eliminating the shockwave. We can confirm it by observing the C_p flow field depicted by Fig. 5.14. On the upper surface, there is no strong shockwave observed on the optimum airfoils while there is a near-normal shockwave on the baseline. Also, the lowest C_p of optimum airfoils are closer to the leading edge compared with the baseline.

b. Best Result

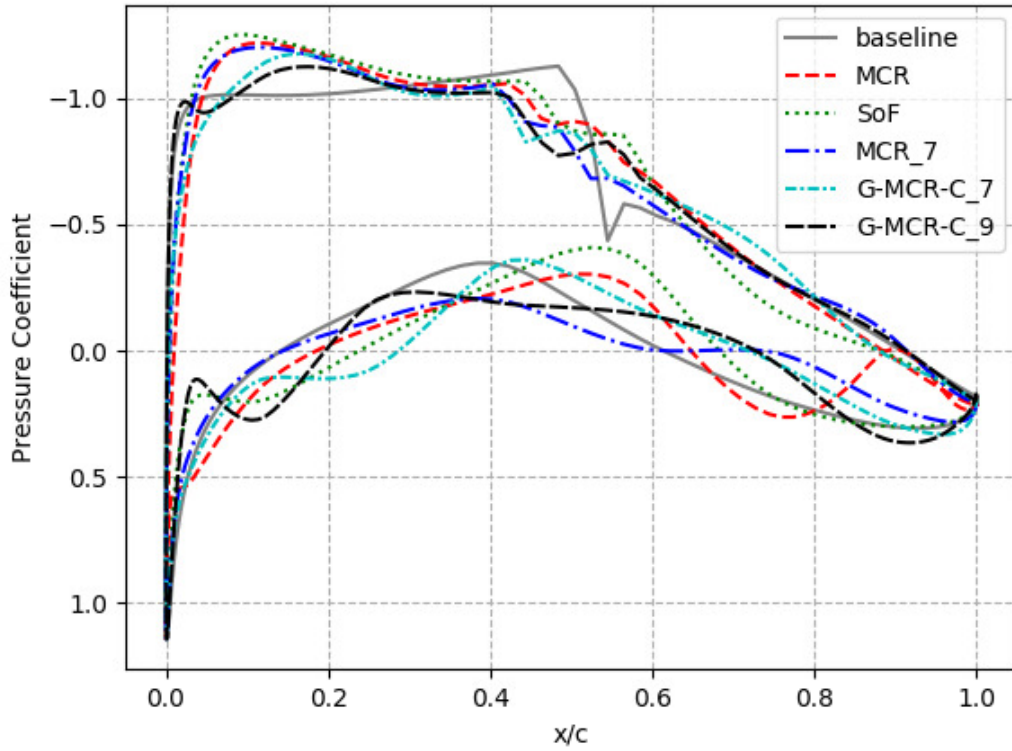
In the best result, the upper surfaces of optimum airfoils seem to produce even weaker shockwave than median results, as shown by the surface pressure distributions in Fig. 5.15b. Probably, the curvature radius around the area of shockwave is even larger. The most negative C_p tends to move even closer to leading edge than the optimum airfoils in median result. We can confirm this as well in the flow field depicted by Fig. 5.16. This might be the reason why the best airfoils produce bigger drag reduction than the median results.

5.5 Multi-point Airfoil Design Optimization

In this problem, we optimize the airfoil shape from three flow regimes simultaneously. This is to model the flight of supersonic business jet where it flies in supersonic speed during transatlantic flight, transonic during overland flight, and subsonic during loitering and descending.



(a)



(b)

Figure 5.13: (a) Shapes and (b) C_p distributions of baseline and optimum airfoils in median results

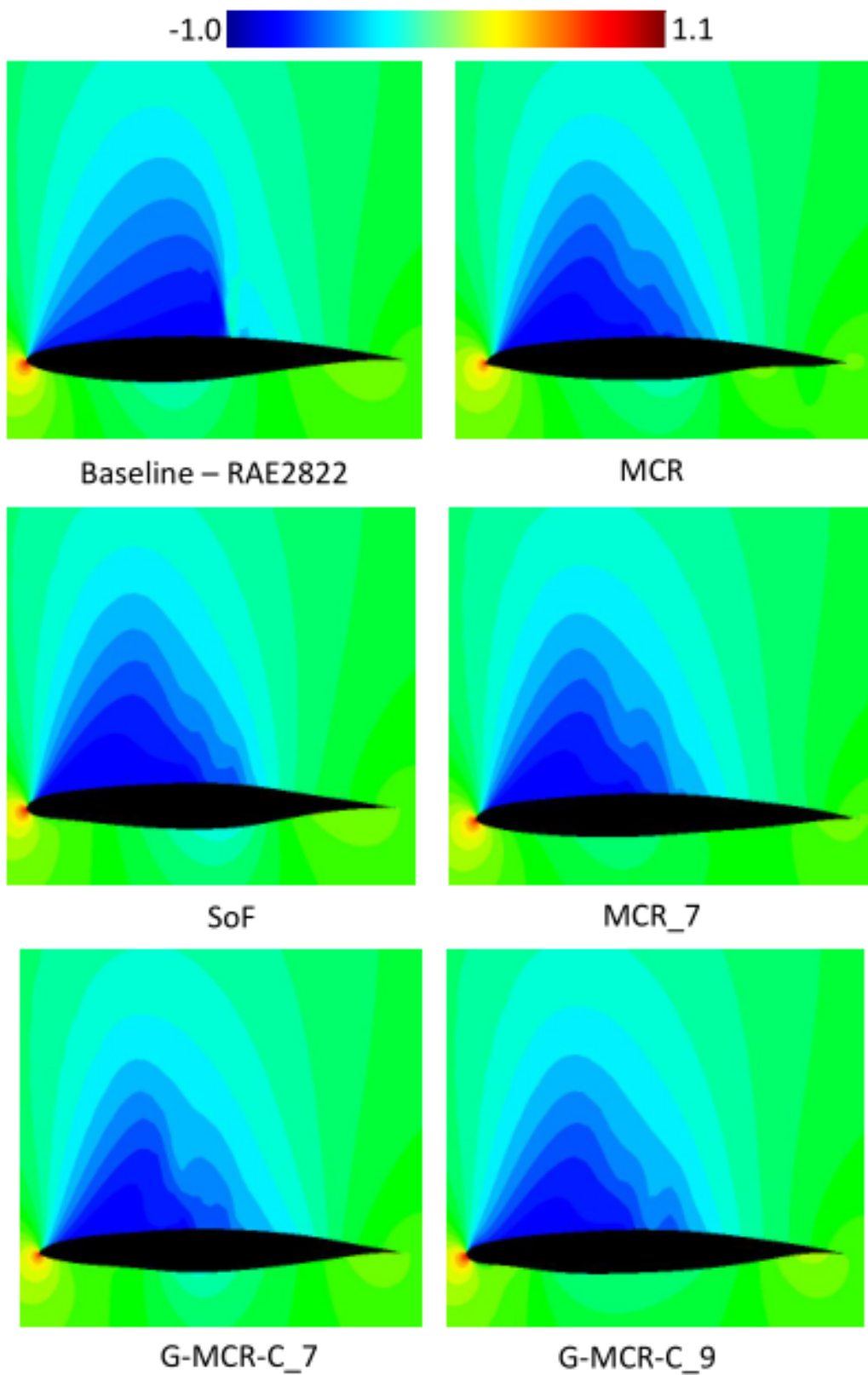
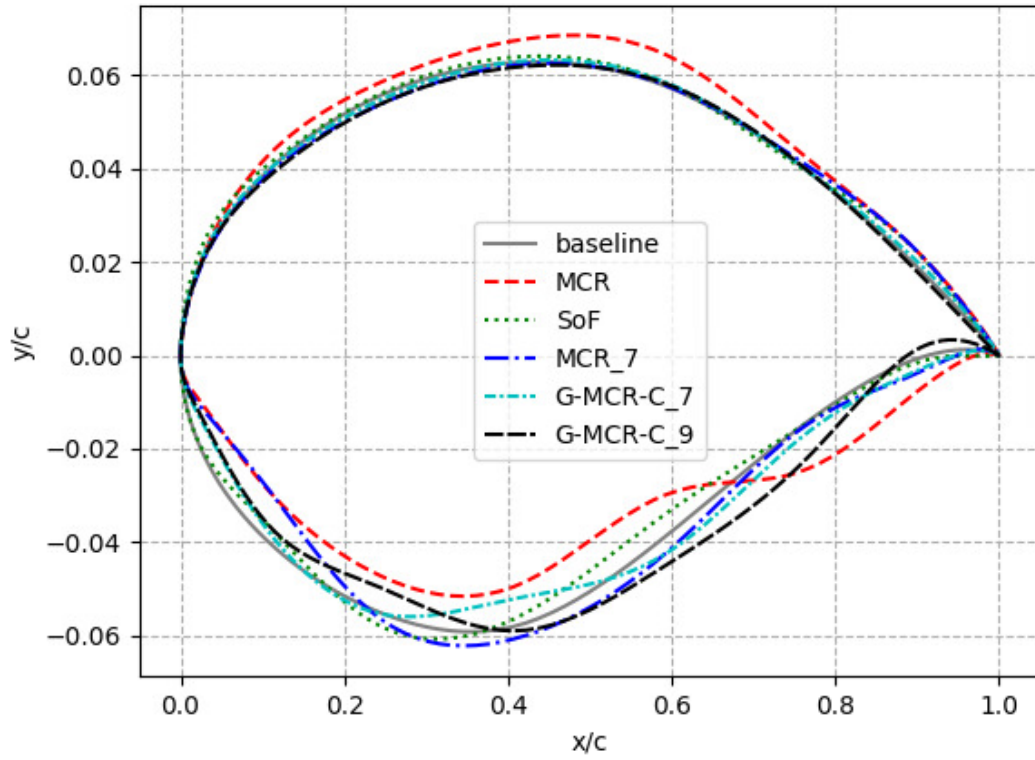
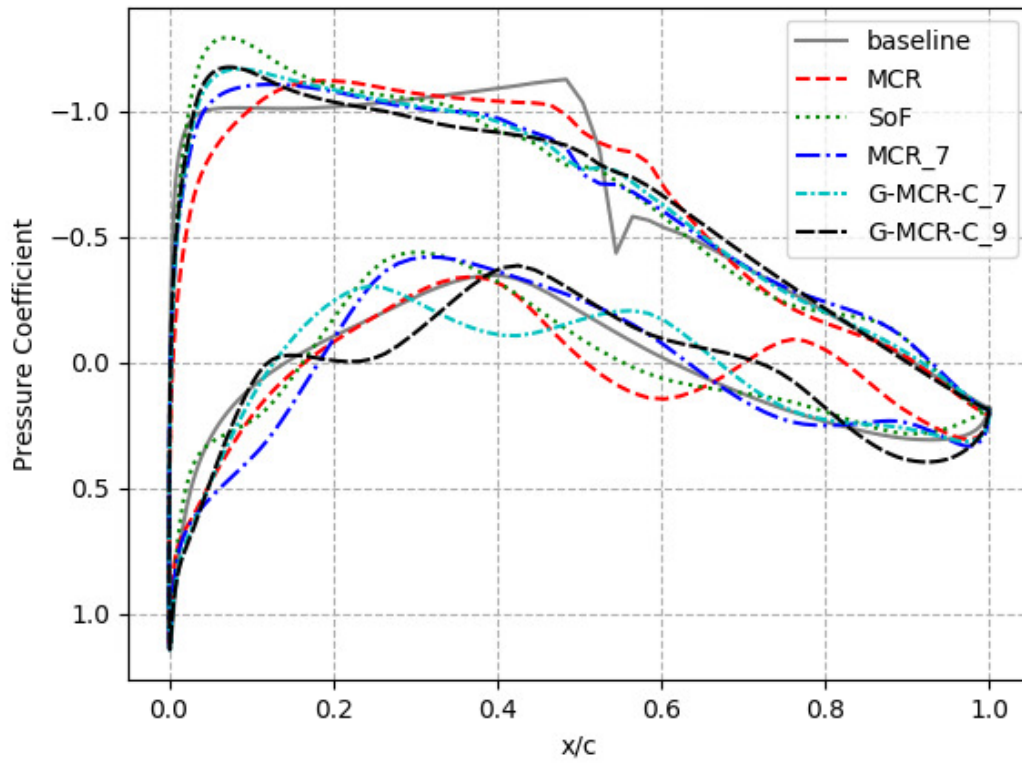


Figure 5.14: C_p flow field of baseline and optimum airfoils (median)



(a)



(b)

Figure 5.15: (a) Shapes and (b) C_p distributions of baseline and optimum airfoils in best results

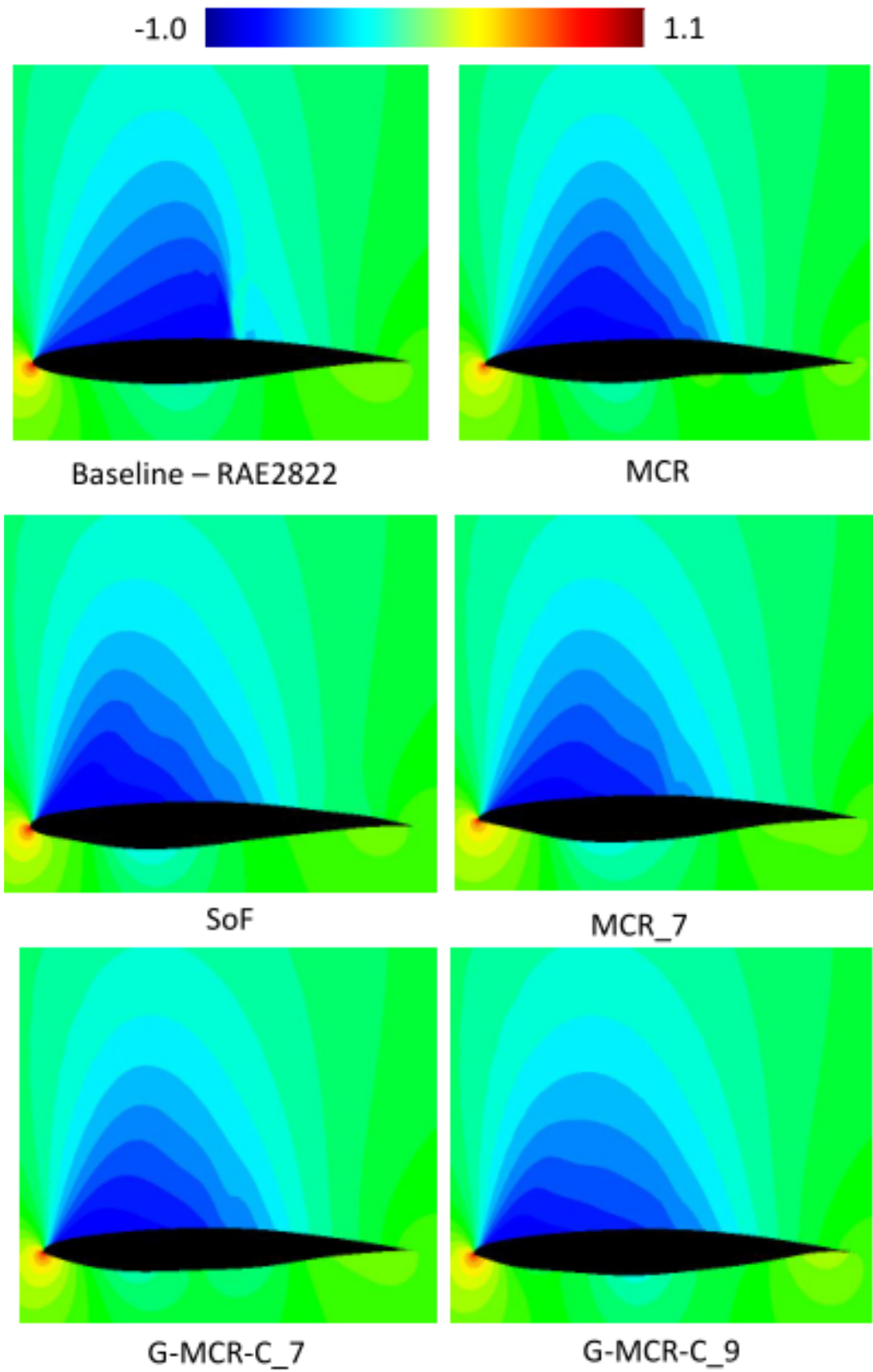


Figure 5.16: C_p flow field of baseline and optimum airfoils (best)

5.5.1 Problem Definition and Optimization Setting

We define the benchmark problem for the multi-point airfoil shape optimization as follows,

$$\begin{aligned}
 &\text{Minimize} && C_d(\mathbf{x}) = 0.85C_{d_{super}}(\mathbf{x}) + 0.1C_{d_{trans}}(\mathbf{x}) + 0.05C_{d_{sub}}(\mathbf{x}) \\
 &\text{subject to} && g_1(\mathbf{x}) = C_{l_{super,ref}} - C_{l_{super}}(\mathbf{x}) \leq 0 \\
 &&& g_2(\mathbf{x}) = C_{m_{super,ref}} - C_{m_{super}}(\mathbf{x}) \leq 0 \\
 &&& g_3(\mathbf{x}) = C_{l_{trans,ref}} - C_{l_{trans}}(\mathbf{x}) \leq 0 \\
 &&& g_4(\mathbf{x}) = C_{m_{trans,ref}} - C_{m_{trans}}(\mathbf{x}) \leq 0 \\
 &&& g_5(\mathbf{x}) = C_{l_{sub,ref}} - C_{l_{sub}}(\mathbf{x}) \leq 0 \\
 &&& g_6(\mathbf{x}) = C_{m_{sub,ref}} - C_{m_{sub}}(\mathbf{x}) \leq 0 \\
 &&& g_7(\mathbf{x}) = A_{ref} - A(\mathbf{x}) \leq 0 \\
 &&& \mathbf{x}_L \leq \mathbf{x} \leq \mathbf{x}_U
 \end{aligned} \tag{5.15}$$

where the subscripts *super*, *trans*, and *sub* denote the supersonic, transonic, and subsonic flight conditions, respectively. The objective function $C_d(\mathbf{x})$ is a composite of drag coefficients in all three flight conditions. The values 0.85, 0.1, and 0.05 are the weights for the C_d s of their corresponding flight conditions based on the ideal flight path from Paris (Charles De Gaulle or CDG) to New York (John F. Kennedy International Airport or JFK) [6]. $C_{l_{super,ref}}$, $C_{l_{trans,ref}}$, and $C_{l_{sub,ref}}$ are the lift coefficient of the baseline airfoil on supersonic, transonic, and subsonic conditions, respectively. We set NACA-2S-(40)(015)-(40)(015) as the baseline airfoil. The flight conditions and aerodynamic properties of the baseline airfoil are presented in Table 5.4. As for the area,

$$A_{ref} = 0.02007138. \tag{5.16}$$

Table 5.4: Flight conditions and aerodynamic properties of baseline airfoil

| Regime | M | Re | AoA (deg.) | $C_{l_{ref}}$ | $C_{m_{ref}}$ | $C_{d_{ref}}$ |
|------------|------|---------|--------------|---------------|---------------|---------------|
| supersonic | 1.45 | 7188300 | 3.53 | 0.239023 | -0.052502 | 0.025485 |
| transonic | 0.95 | 8705900 | 2.39 | 0.403880 | -0.063391 | 0.031100 |
| subsonic | 0.44 | 6852900 | 5.70 | 0.621178 | -0.016698 | 0.058765 |

We set the lower bound (\mathbf{x}_L) and upper bound (\mathbf{x}_U) of the search space as the ± 0.01 perturbation of baseline's upper and lower surfaces' control points in the Y-axis. Except for the four points near trailing edge, to maintain a sharp trailing edge, we set the two points nearest the trailing edge with ± 0.0001 perturbation, and the second nearest with ± 0.001 perturbation. The control points on leading and trailing edges are fixed at (0,0) for leading edge and at (1,0) for trailing edge. The search space is illustrated in Fig. 5.17. There are 18 active control points (in Y-axis only) in total as the design variables, nine for generating lower surface and nine for upper surface.

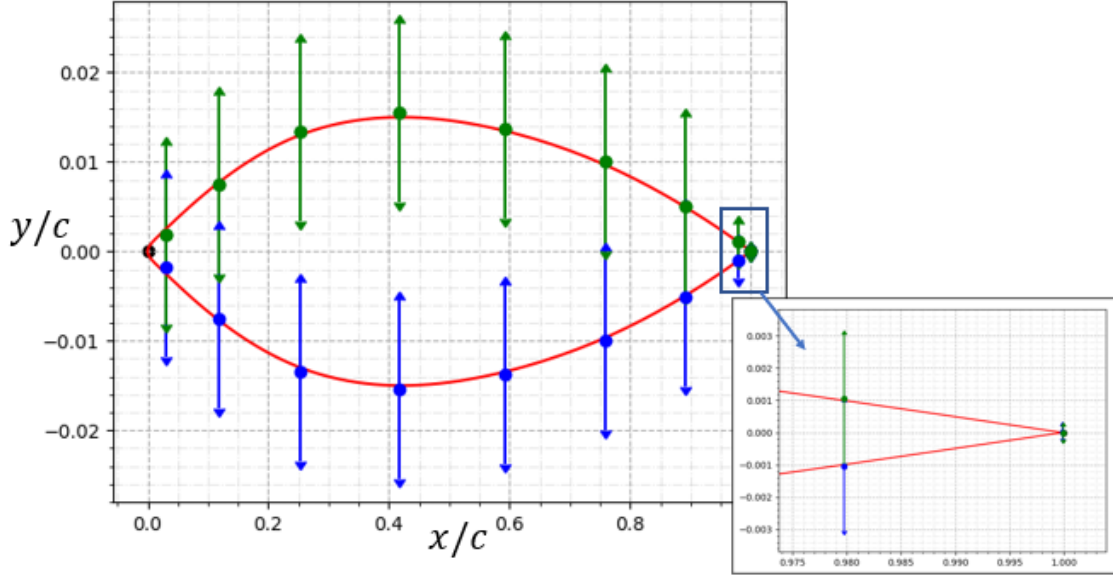


Figure 5.17: Search space for multi-point airfoil design optimization

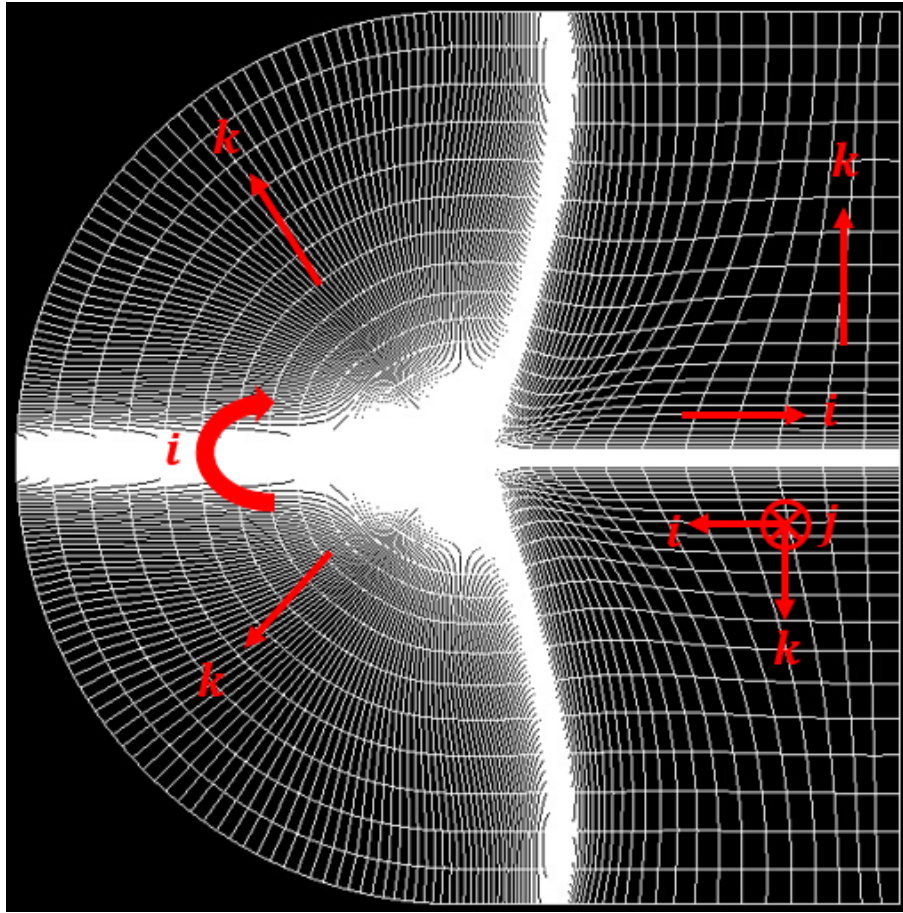
In order to obtain the aerodynamic properties, an airfoil needs to be simulated three times in with LANS3D, each in different flight condition. The supersonic simulation takes about five minutes, the transonic about eight minutes, and the subsonic about 20 minutes. We set a solution evaluation budget of 5000 for one optimization. Therefore, it requires almost 10 days to finish one optimization with 13 cores. We set five independent runs for each CHT. In this problem, we include the baseline airfoil as one of the individuals in the initial population.

Table 5.5: Aerodynamic properties of NACA-2S-(40)(015)-(40)(015) airfoil

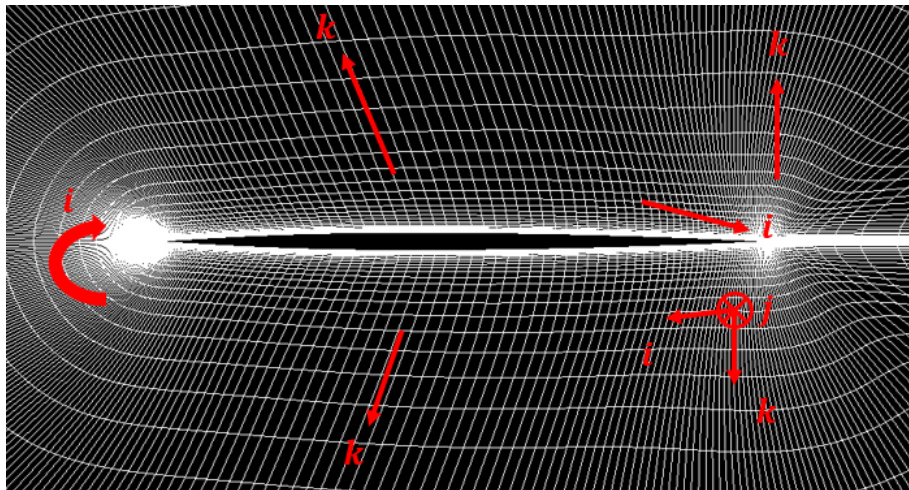
| Coefficient | LANS3D | [90] |
|-------------|---------|---------|
| C_l | 0.1897 | 0.1890 |
| C_d | 0.02197 | 0.02195 |

5.5.2 CFD Validation

To validate the accuracy of LANS3D, we compare the LANS3D simulation of the baseline airfoil with the test problem from [90]. The flight conditions are $M = 1.5$, $Re = 1 \times 10^6$, and $AoA = 3^\circ$. We set the number of grids to $369 \times 3 \times 65$, the same with the setting in transonic airfoil design problem. The farfield and near-wall grids are depicted in Figs. 5.18a and 5.18b, respectively. The setting for LANS3D is the same with what is used in transonic airfoil design, presented in Table 5.1.



(a)



(b)

Figure 5.18: Grid visualizations at (a) full and (b) near wall views of NACA-2S-(40)(015)-(40)(015) airfoil

The comparison of C_l and C_d from LANS3D simulation and [90] is presented in Table 5.5. We can notice that the C_l and C_d values are very close. Therefore, we decide that this setting is sufficient for the optimization.

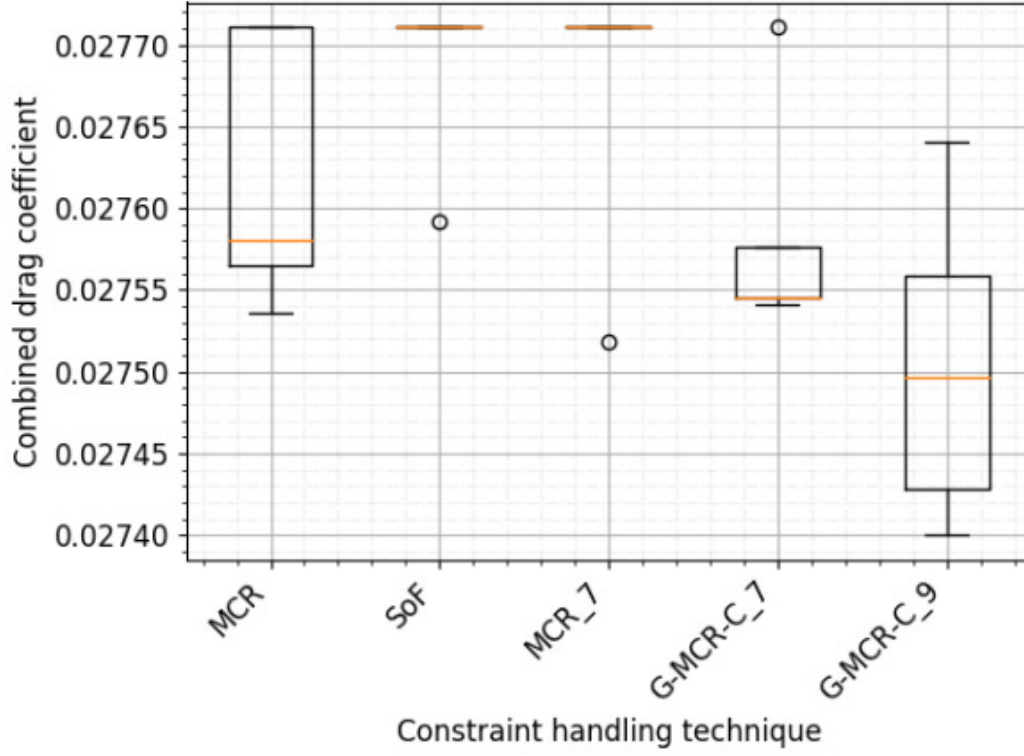


Figure 5.19: Box plot of obtained optimum combined drag coefficients from 5 independent runs

Table 5.6: Statistical significance test for multi-point airfoil design

| CHT | MCR | SoF | MCR_7 | G-MCR-C_7 | G-MCR-C_9 |
|---------------|-----|-----|-------|-----------|-----------|
| vs. MCR | | = | = | = | + |
| vs. SoF | = | | = | + | + |
| vs. MCR_7 | = | = | | = | + |
| vs. G-MCR-C_7 | = | - | = | | = |
| vs. G-MCR-C_9 | - | - | - | = | |

5.5.3 Optimization Result and Discussion

Fig. 5.19 depicts the optimum combined drag coefficients obtained by each CHT after five independent runs. It seems that this problem is very difficult that SoF and MCR_7

can only produce optimum solution with better objective value on one run out of five. It is possible that the baseline airfoil is a local optimum on this problem definition. From all five CHTs, only G-MCR-C_9 can produce objective values better than the baseline airfoil in all five runs. As we can observe in Fig. 5.19, G-MCR-C_9 is the best performer in this problem by obtaining better objective values in worst, median, and best results than other CHTs. We can confirm it in the significance test result presented in Table 5.6 where G-MCR-C_9 obtains three "+" signs. G-MCR-C_7 is the second best by obtaining one "+" sign without any "-" sign. MCR and MCR_7 are the third best by obtaining one "-" sign. It is rather an unusual observation that MCR_7 becomes one of the worst CHTs.

Observing the N_v spreading in offspring population (median result) depicted by the middle column of Fig. 5.20, it seems that almost all constraints imposed in this problem are difficult. All CHTs similarly produce infeasible individuals violating five to six constraints out of seven. Even SoF, the conventional CHT which favor feasible individuals over the infeasible ones, also tends to violate five constraints out of seven. We confirm that all constraints are difficult by observing the α_i values of MCR_7, G-MCR-C_7, and G-MCR-C_9 depicted in Fig. 5.21. The α_i values of all constraints stay at certain non-zero values. The area constraint seems to be the most difficult constraint because it tends to have biggest value compared with other α_i s.

Observing the objective value spreading in offspring population (median result) depicted by the right column of Fig. 5.20, it seems that we can get more understanding why the CHTs produce such results shown in Fig. 5.19. MCR seems to generate rather sufficient infeasible individuals with lower objective values than the feasible ones, which probably contributes to more efficient search. As for SoF, its convergence rate is very slow because it only depends on feasible individuals. That is probably the reason why it cannot obtain better objective value than the baseline until the end of optimization.

We observe rather unusual phenomenon on the objective value spreading of MCR_7. In this problem, it seems that it faces difficulty in generating feasible individuals and explore too big of infeasible region, which might be the reason why MCR_7 performs poorly. Meanwhile, G-MCR-C_7, whose η , γ , and α_i definitions are the same with MCR_7 except it adopts circular adaptive β_1 , generates more robust performance, most likely because its ability to generate feasible individuals is better than MCR_7. As we can observe in the objective value spreading, G-MCR-C_7 can generate feasible individuals from early generation while also generating many infeasible individuals with better objective values than the feasible ones. As for G-MCR-C_9, it performs well most likely because it also produce feasible individuals from the early generation as well as many infeasible individuals with better objective values than the feasible ones.

It seems that the circular adaptive β_1 and $\gamma = \frac{1}{\sum_{i=1}^m \alpha_i}$ play an important role in this

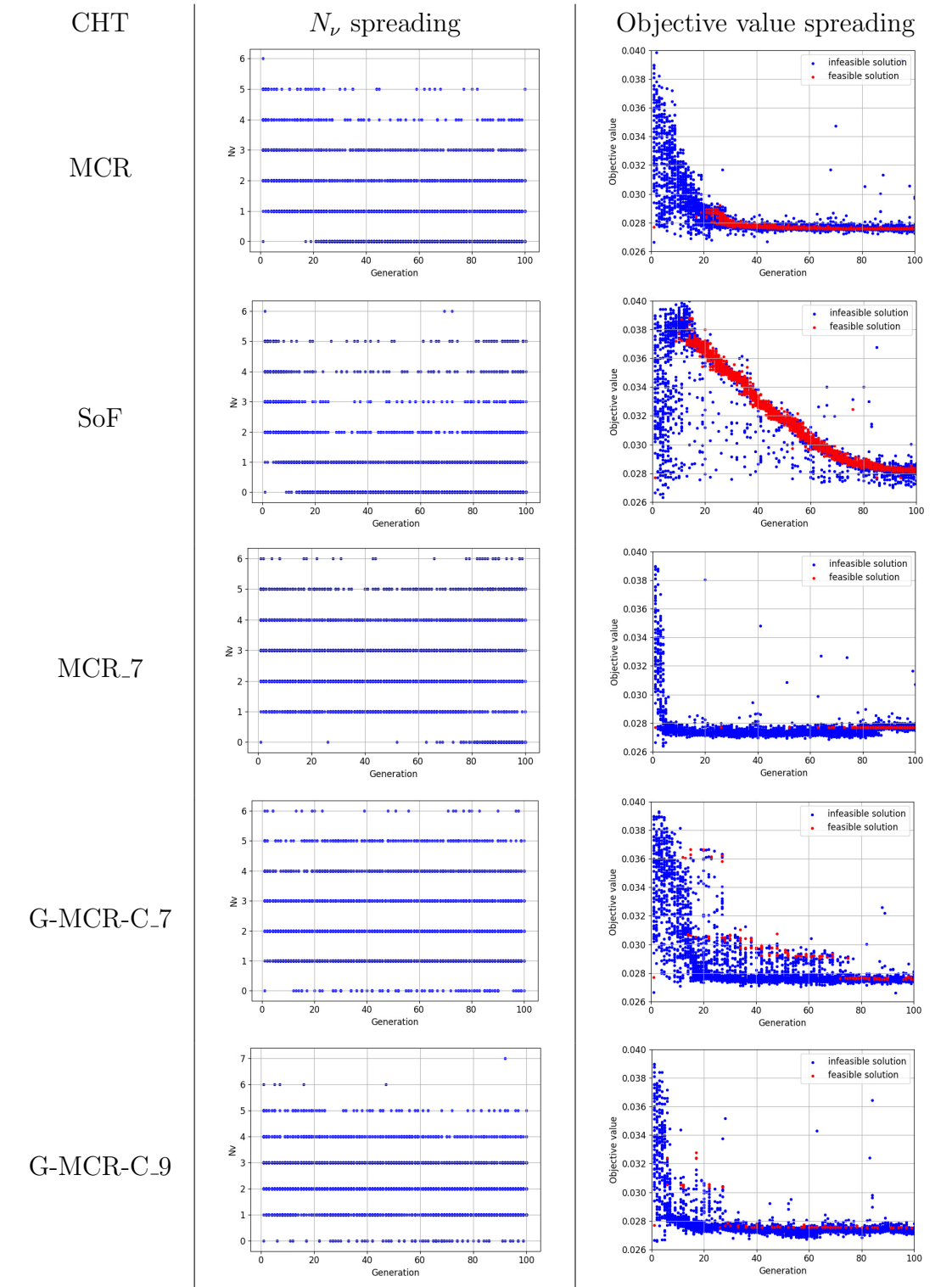


Figure 5.20: N_ν and objective value spreadings of the CHTs (median of 51 runs).

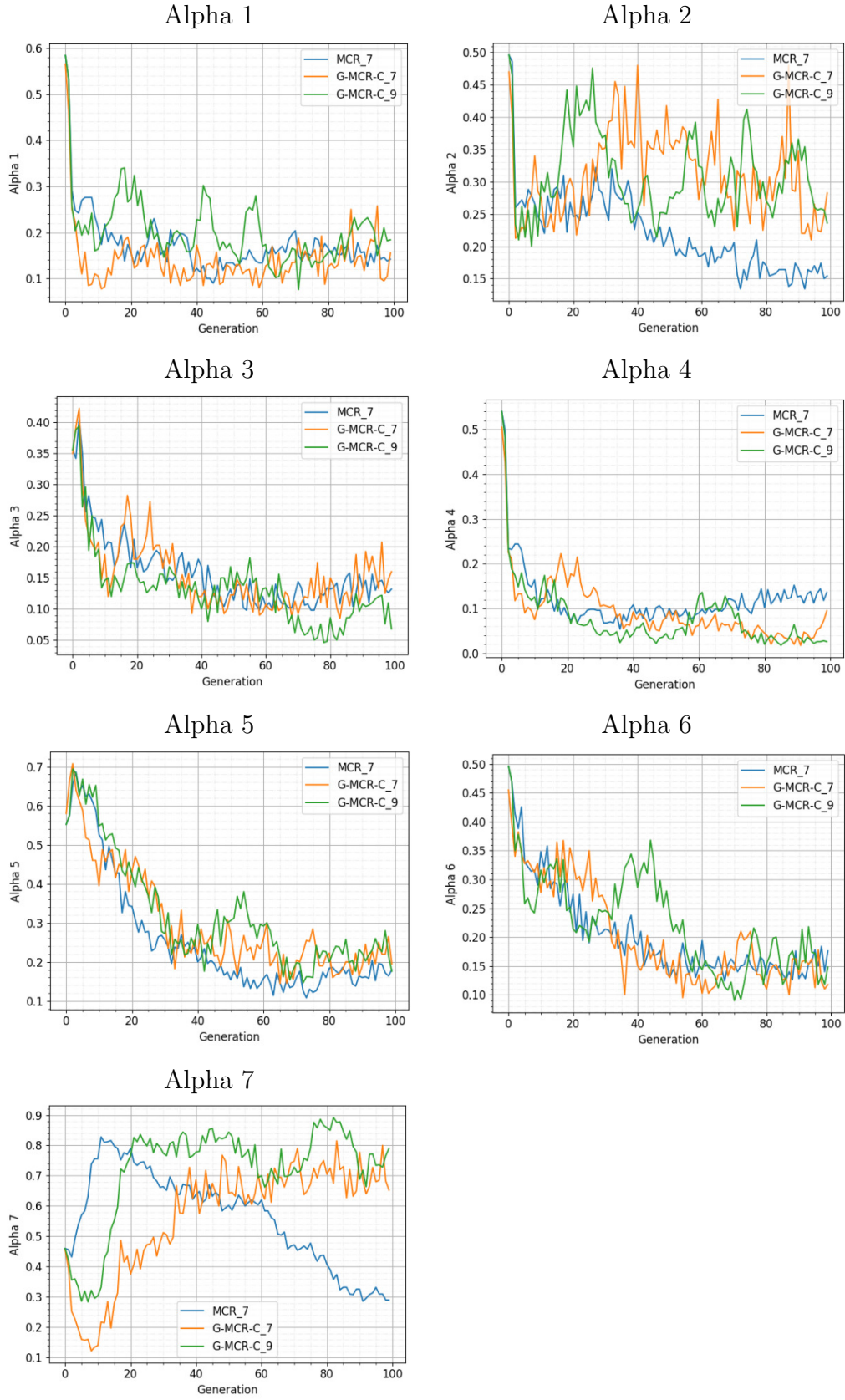


Figure 5.21: Average α_i on multi-point airfoil design from five independent runs

problem. Comparing MCR_7 (without circular adaptive β_1) and G-MCR-C_7 (with circular adaptive β_1), it seems that the circular adaptive β_1 gives G-MCR-C_7 more balanced search in both feasible and infeasible regions, thus G-MCR-C_7 performs better. Then, we compare G-MCR-C_7 ($\gamma = 1$) and G-MCR-C_9 ($\gamma = \frac{1}{\sum_{i=1}^m \alpha_i}$). While both implementing circular adaptive β_1 , it seems that G-MCR-C_9 can produce better performance because the $\gamma = \frac{1}{\sum_{i=1}^m \alpha_i}$ makes term $\gamma \sum_{i=1}^m \alpha_i R_{\nu_i}$ in G-MCR-C_9 relatively smaller than G-MCR-C_7, giving more weight on R_f . Assisted by circular adaptive β_1 to balance the search in both feasible and infeasible regions, G-MCR-C_9 thus produces better result.

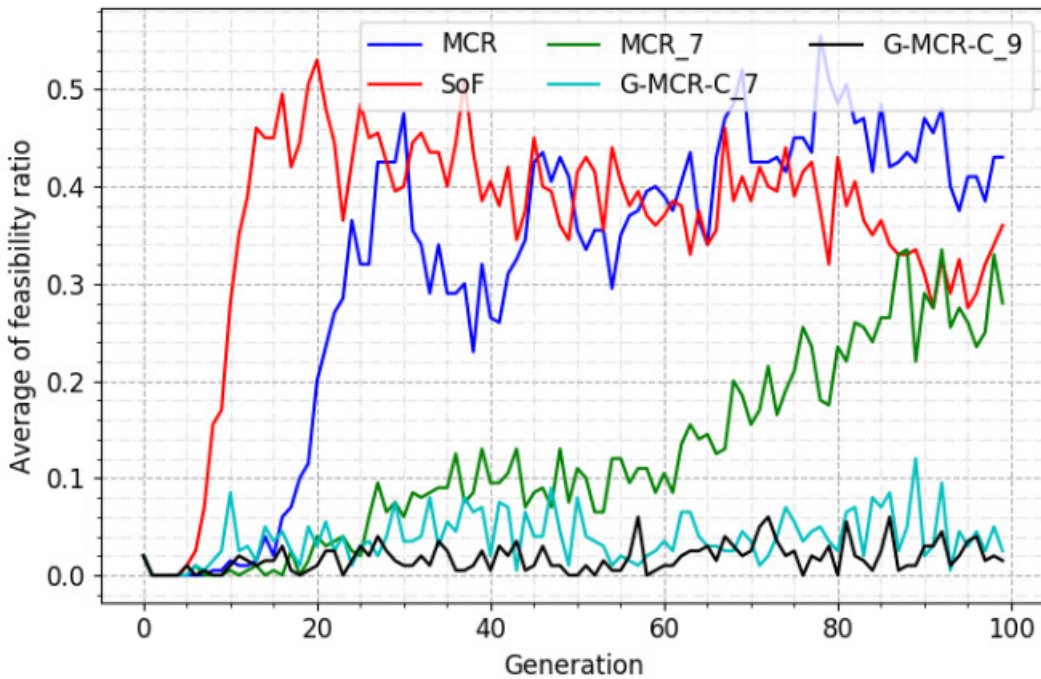
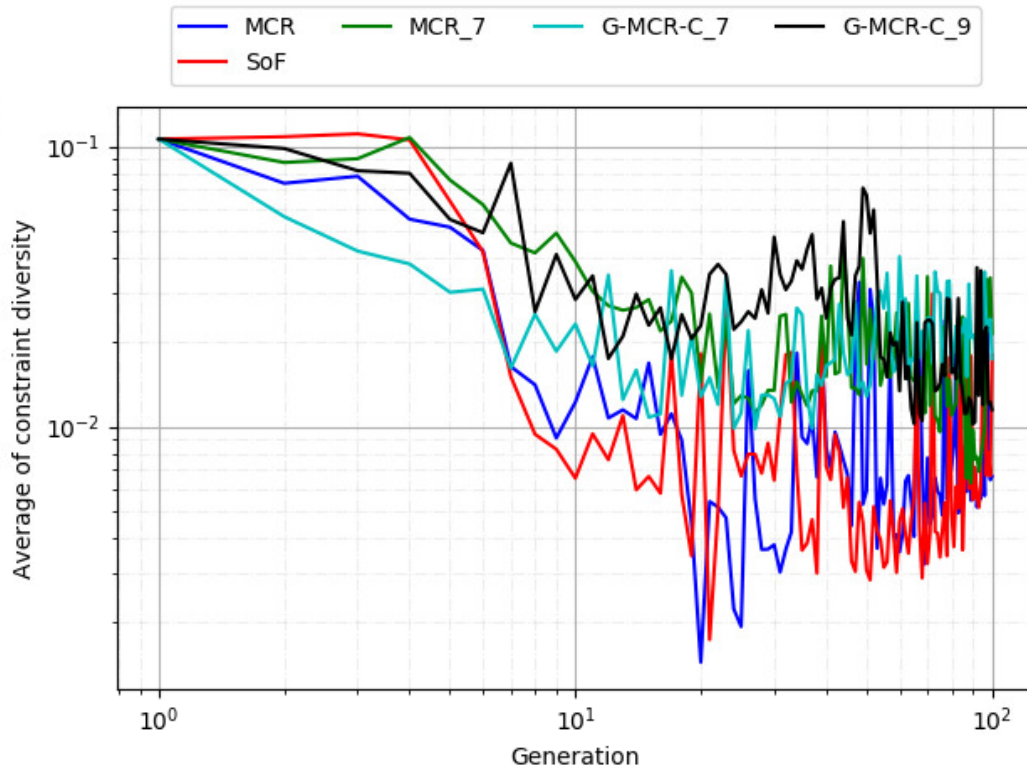
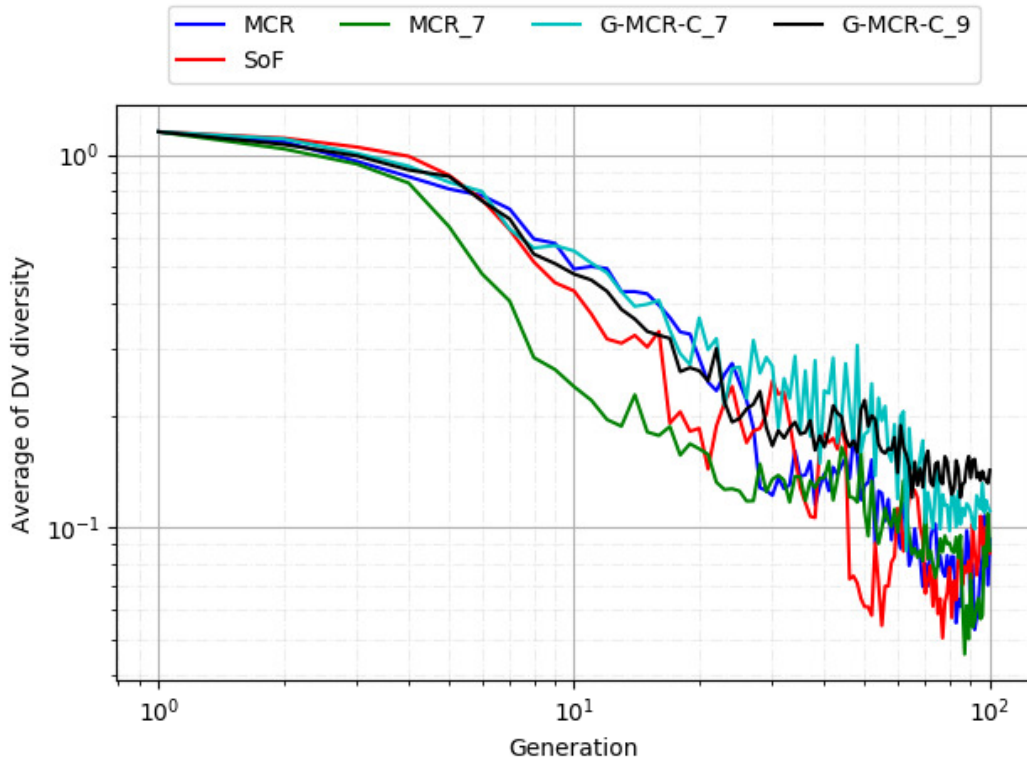


Figure 5.22: Average of feasibility ratio from 11 independent runs

Observing the average feasibility ratio of offspring population depicted by Fig. 5.22, they fluctuate quite significantly due to small number of independent runs. However, we can still notice that G-MCR-C_7 and G-MCR-C_9 keep the feasibility ratio small (0% - 6% for G-MCR-C_9 and 2% - 10%). This makes the value of β_1 rather small, leading to bigger portion of constraint components than the objective value in the fitness formulation. This is probably why MCR also produces rather good performance, because the portion of constraint components is bigger than the objective value. MCR produces rather big feasibility ratio. However, quite many of infeasible individuals have lower objective values than the feasible ones, as shown in Fig. 5.20. Therefore, it can still interact with infeasible individuals to perform quite efficiently. As for SoF, it also produce rather big feasibility ratio. However, because it cannot interact with infeasible individuals, its performance is poor.



(a)



(b)

Figure 5.23: Average of diversity in (a) constraint and (b) design variable spaces

As for MCR_7, we can see an increasing feasibility ratio despite our previous discussion that it explores too big of infeasible region. It is because at some runs, all individuals in the new parent population are the baseline airfoil starting from around the middle of optimization. This case of individual duplication is possible to happen because of the matingpool scheme in the binary tournament selection. This shows an inefficiency of the RGA. However, we can also observe that the baseline seems to be a local optimum surrounded by constraint boundary. Since all individuals in the parent population are the same baseline airfoil, the mechanism to produce new different individuals is only from mutation. Even after the individuals are slightly mutated, different feasible individuals still cannot be found, indicating that the baseline airfoil might be surrounded by the constraint boundary.

Observing the diversity in constraint and design variable spaces as depicted in Figs. 5.23a and 5.23b, respectively, the diversity histories are more fluctuating than the previous problems we have investigated. Probably, it is due to the small number of independent runs. Even though it is not very clear, it seems that G-MCR-C_7 and G-MCR-C_9 tend to generate the most diverse offspring population in both constraint and design variable spaces. MCR_7 is also one of the CHTs which produces the biggest diversity in the constraint space, probably because it explores very big infeasible region. However, together with SoF and MCR, it produces one of the smallest diversity in the design variable space, probably because of the duplication of baseline airfoil in the new parent population.

5.5.4 Analysis on the Shape of Optimum Designs and the Flow Fields

In multi-point airfoil design, we only observe the optimum shapes in the best results because the drag reduction is small. Fig. 5.24 depicts the shapes of the baseline and optimum airfoils in best results. In supersonic condition, the optimum airfoils have smaller drag coefficients than the baseline airfoil, with G-MCR-C_9 producing the biggest reduction. In transonic condition, the drag coefficients of the optimum airfoils are bigger than the baseline airfoil, with MCR producing the biggest increase. In subsonic condition, the results vary. MCR, SoF, and MCR_7 produce drag reduction baseline while G-MCR-C_7 and G-MCR-C_9 produce drag increase. It seems that the optimizer favors optimization in supersonic condition. It is reflected from the optimum shapes, which are thinner than the baseline near the leading edge. This kind of shape is useful to reduce the shockwave intensity at the leading edge, which is good for supersonic flight. It is probably because the supersonic condition has much larger weight during the optimization (0.85, while only 0.1 and 0.05 for transonic and subsonic conditions, respectively). For further observation, we focus on comparing the C_p flow field of the baseline and G-MCR-C_9's airfoils.

Observing the shockwave occurred at the leading edge of the baseline and G-MCR-C_9's airfoil (depicted in Fig. 5.25), the G-MCR-C_9's airfoil seems to produce weaker shockwave than the baseline due to thinner shape near leading edge, leading to drag reduction. In transonic condition, it seems that G-MCR-C_9's airfoil produces stronger shockwave on the upper surface, leading to drag increase, as depicted in Fig. 5.26. In subsonic condition, there seems to be not much difference in the C_p flow field between the baseline and G-MCR-C_9's airfoil. Observing Fig. 5.27, the factor yielding to small drag increase in G-MCR-C_9 is probably the lower C_p produced in the pointed area.

Table 5.7: Drag coefficients ($\times 10^4$) of baseline and optimum airfoils (best result)

| Regime | Baseline | MCR | SoF | MCR_7 | G-MCR-C_7 | G-MCR-C_9 |
|------------|----------|--------|--------|--------|-----------|-----------|
| supersonic | 254.85 | 251.29 | 252.26 | 252.16 | 251.92 | 250.81 |
| transonic | 311.00 | 324.51 | 322.41 | 315.50 | 318.81 | 316.58 |
| subsonic | 587.65 | 586.16 | 585.13 | 585.90 | 587.88 | 588.93 |

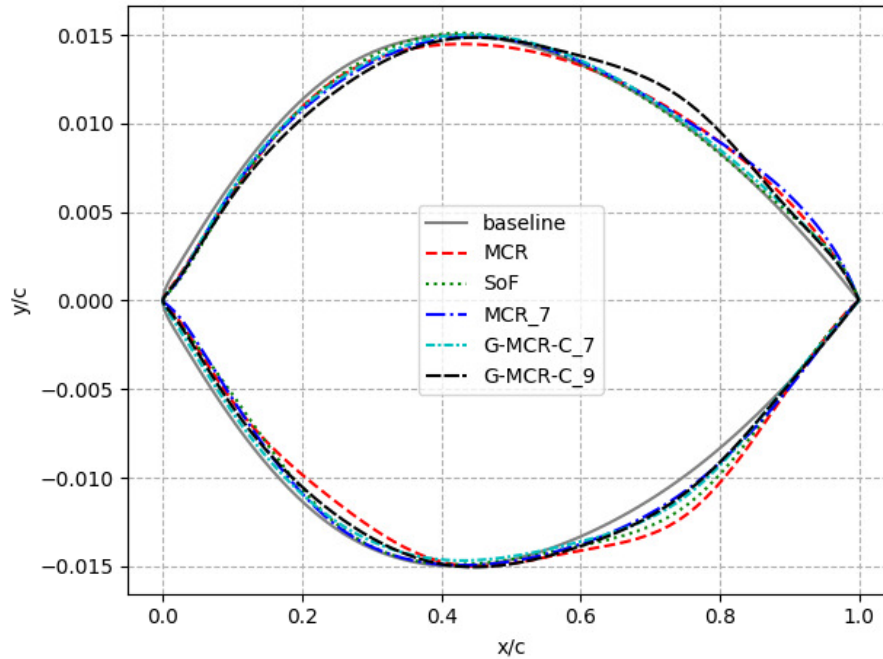


Figure 5.24: Shapes of baseline and optimum airfoils in best results

5.6 Summary

Implementations of various types of CHTs are investigated on two kinds of airfoil aerodynamic design optimization problems. The first problem is a transonic airfoil design where

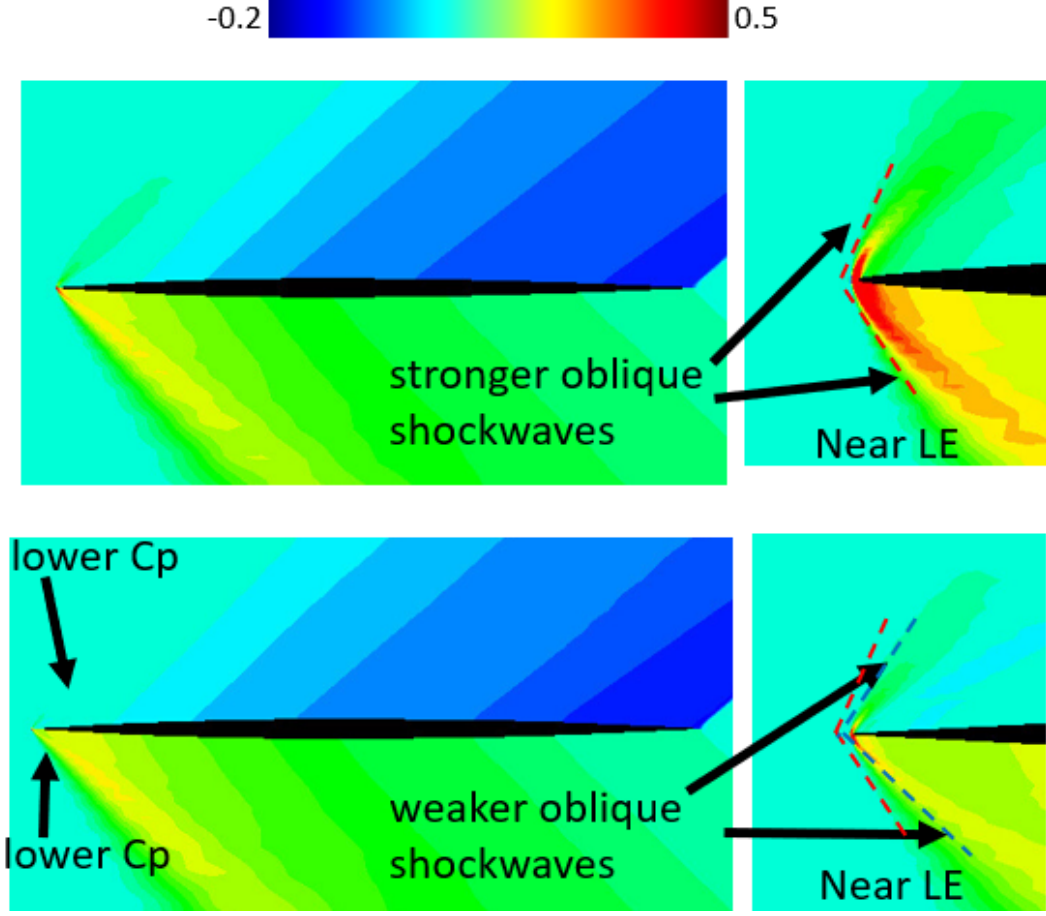


Figure 5.25: C_p flow field of the baseline (upper) and G-MCR-C₉'s airfoil (lower) in supersonic condition.

the airfoil shape is optimized under a single, transonic condition. The second problem is a multi-point airfoil design where the airfoil is optimized simultaneously on three speed conditions: supersonic, transonic, and subsonic conditions. The CHTs comprise SoF, MCR, and the most robust CHTs obtained from the preceding chapter: MCR₇, G-MCR-C₇, and G-MCR-C₉.

In transonic airfoil design, MCR₇, G-MCR-C₇ and G-MCR-C₉ produce significantly better convergence performance than MCR and SoF. This indicates that CHTs with balanced search in both feasible and infeasible regions can find the constrained optimum more efficiently than conventional CHTs in this problem. MCR₇, G-MCR-C₇ and G-MCR-C₉ also produce more variety of infeasible individuals and sufficient infeasible individuals with lower objective values (between 5%-15%). The increase of variety is also confirmed by the increase of offspring population diversity in constraint and design variable spaces. We also observe that the combined population of RGA might lose all feasible individuals with G-MCR-based CHTs. Therefore, a mechanism to always preserve at least one

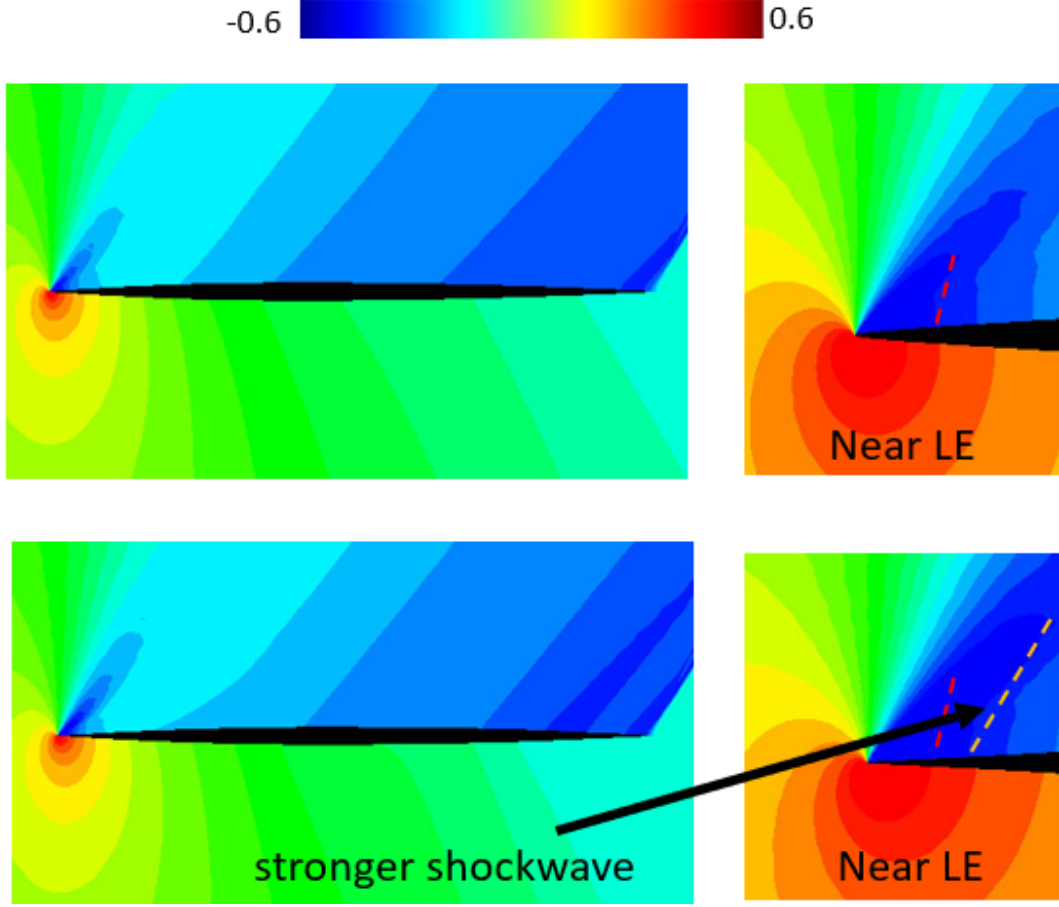


Figure 5.26: C_p flow field of the baseline (upper) and G-MCR-C_9's airfoil (lower) in transonic condition.

feasible individual might be necessary to develop in the future. In the analysis of shapes and aerodynamic performances, we obtain that the optimum airfoils have different shapes even though some of them have similar drag coefficients. Despite their different shapes, the optimum airfoils tend to have larger curvature radius around the area of shockwave, moving the most negative C_p on upper surface closer to the leading edge, weakening or omitting the shockwave, thus leading to smaller drag coefficients.

In multi-point airfoil design, only G-MCR-C_9 performs statistically better than MCR. SoF cannot find any feasible individuals better than the baseline from five independent runs because in search the feasible region too long. Meanwhile, MCR_7 also performs rather poorly by obtaining only one optimum solution out of five runs with better objective value than the baseline, but because it tends to explore the infeasible region too long. The circular adaptive β_1 and proportionation with $\gamma = \frac{1}{\sum_{i=1}^m \alpha_i}$ seems to play an important role in balancing the search in both feasible and infeasible regions. That is why G-MCR-C_9 can produce better convergence performance than other CHTs. It seems that the baseline

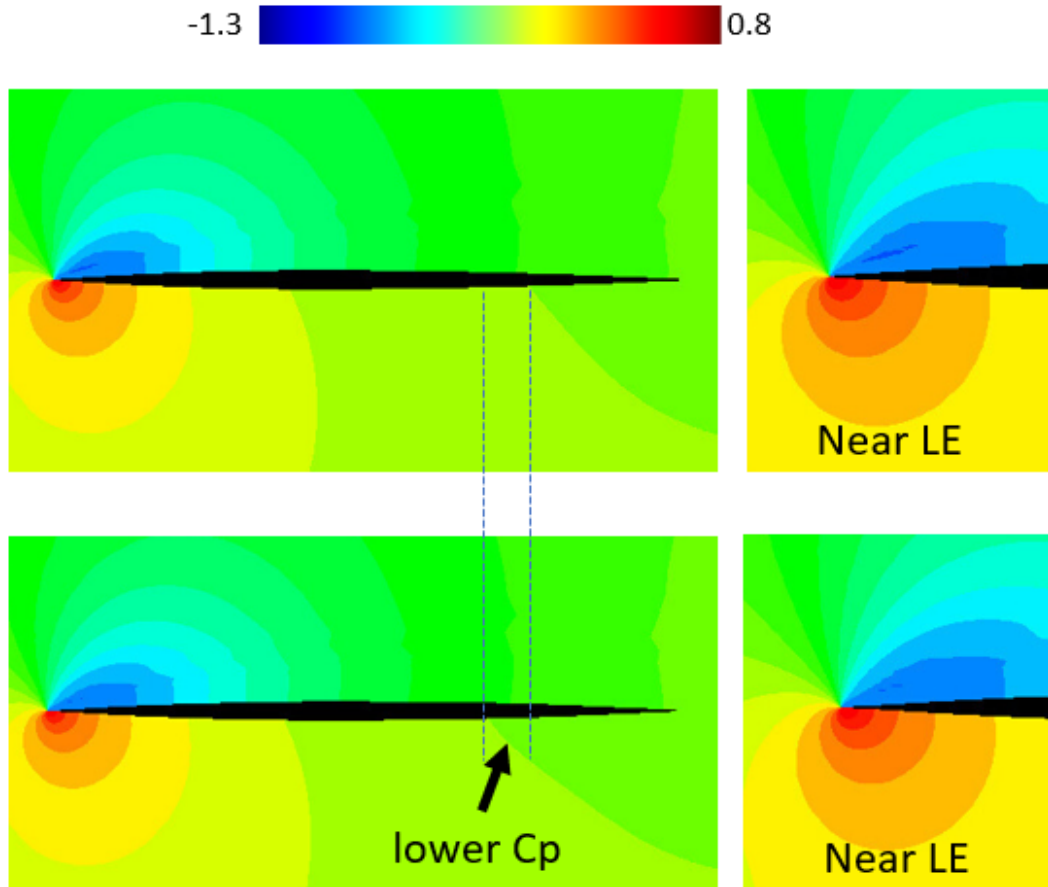


Figure 5.27: C_p flow field of the baseline (upper) and G-MCR-C₉'s airfoil (lower) in subsonic condition.

is one of the local optima and is surrounded by the constraint boundary. Analyzing the shapes and aerodynamic performances of the optimum airfoils, the optimizer seems to favor optimization in supersonic condition because of larger weight. It is reflected from the optimum shapes which have thinner shapes near the leading edge, which is useful to reduce shockwave intensity in supersonic condition. The drag coefficients of optimum airfoils in supersonic condition are smaller than the baseline, while they are bigger in transonic condition. In subsonic condition, the drag coefficients of both baseline and optimum airfoils only have very slight difference.

Chapter 6

Overall Summary

6.1 Conclusion

In the present work, we have proposed a new category of CHT mainly for solving severely constrained engineering design optimization problem which not only balances the assessment of constraints with different order of magnitude, but also balances the search in both feasible and infeasible regions. The modifications are based on some modifications of MCR, which already balances the assessment of constraints with different orders of magnitudes. It also balances the search in both feasible and infeasible regions when the number of constraints is small. However, MCR becomes favoring feasible individuals when the number of constraints is large, making it not very efficient for engineering design optimization.

In our first attempt, we propose to modify MCR by enhancing its infeasible region exploration capability. We devise the modifications by following three principles: the constraints might have different difficulty levels, omitting R_{N_v} might give more flexibility in exploring infeasible region, and the portion of constraint violation ranks is too large in problems with large number of constraints. We investigate the effects of the modifications on a car structure design problem and find that CHTs namely MCR_7, MCR_8, and MCR_9 significantly improve the convergence performance. It is probably because the modifications generate more variety of infeasible individuals and many infeasible individuals with lower objective values compared with MCR. The increase of variety of infeasible individuals also increases the diversity of offspring population in both constraint and design variable spaces. However, MCR_7, MCR_8, and MCR_9 seem to too big of infeasible region, especially MCR_8 and MCR_9. They might face difficulty in generating any other feasible individuals after obtaining the first feasible individual.

In our second attempt, we propose other modifications by tuning β_1 and β_2 to improve the balance of search in feasible and infeasible regions so that the convergence

performance can be more robust. We investigate three forms of β_1 : linear β_1 , circular β_1 , and quadratic β_1 , to MCR_7, MCR_8, and MCR_9. Based on the investigation on 78 benchmark problems with significance test, we obtain that a CHT named G-MCR-C_9 produces the best and the most robust convergence performance in general problems as well as real-world design problems (RWDPs) which have many constraints, small feasible region, and active constraints. We also obtain that for three RWDPs investigated here, the offspring population's feasibility ratio of 5%-15% from the middle to late generations seems to be the best environment for conducting optimization.

Implementations of various types of CHTs are investigated on two kinds of airfoil aerodynamic design optimization problems: transonic airfoil design and multi-point airfoil design. In transonic airfoil design, we obtain that CHTs with balanced search in both feasible and infeasible regions can find the constrained optimum more efficiently than conventional CHTs in this problem. In the analysis of shapes and aerodynamic performances, the optimum airfoils have different shapes even though some of them have similar drag coefficients. Despite their different shapes, the optimum airfoils tend to have larger curvature radius around the area of shockwave, moving the most negative C_p on upper surface closer to the leading edge, weakening or omitting the shockwave, thus leading to smaller drag coefficients.

In multi-point airfoil design, it seems that the baseline airfoil is a local optimum because many CHTs cannot find feasible individuals with better objective value. Only G-MCR-C_9 can find better optimum solutions than the baseline in all five runs and it performs statistically better than MCR. It seems that the baseline is one of the local optima and is surrounded by the constraint boundary. Analyzing the shapes and aerodynamic performances of the optimum airfoils, the optimizer seems to favor optimization in supersonic condition because of the thinner shapes near the leading edge, which are useful to reduce the shockwave intensity in supersonic condition. The drag coefficients of optimum airfoils in supersonic condition are smaller than the baseline, while they are bigger in transonic condition. In subsonic condition, the drag coefficients of both baseline and optimum airfoils only have very slight difference.

6.2 Obtained Knowledge and Recommended CHT

Based on the observation on the effects of the proposed CHTs producing good convergence, we summarize three factors that might be important to consider when devising a CHT:

1. Having many variety of infeasible individuals in the offspring population (diverse population) might assist in generating new individuals with better quality, thus the search towards constrained optimum becomes more efficient.

2. Having many infeasible individuals with lower objective values than the feasible ones might help pulling the search towards constrained boundary, thus the constrained optimum might be able to be found faster.
3. During the optimization, it seems that there is no need to generate many feasible individuals in the population. In car structure design, MOPTA 2008, wind turbine, and transonic airfoil design, the number of feasible individuals between 5%-15% of the population size seems to be the most appropriate environment for optimization. In multi-point airfoil design, it is between 2%-6%. Based on these results, while the exact number of feasible individuals might be problem-dependent, it is a small number in general.

After investigating many kinds of CHTs and modifications of MCR, we obtain that G-MCR-C_9 has the most robust performance. It consistently becomes one of the best, if not the best, in terms of convergence performance on the RWDPs investigated in the present work, including the airfoil design problems, probably because it considers all three aforementioned factors. Therefore, we recommend to utilize G-MCR-C_9 as the CHT for solving severely constrained engineering design optimization problem with EA.

6.3 Future Work

In the investigation, we observe one possible shortcoming of the proposed CHTs despite their convergence improvement against MCR. The G-MCR-based CHT might lose feasible individual in the middle of optimization because there is no mechanism to intentionally preserve at least one feasible individual in the population. If, after further thorough investigation, this condition is really a shortcoming, such mechanism to intentionally preserve feasible individuals in the population might be necessary to develop.

It is also interesting to implement G-MCR-based CHTs in multi-/many-objective optimization problems. Since the best CHTs such as G-MCR-C_9 can produce more diversity in the constraint and design variable spaces, probably it will help multi-/many-objective optimization algorithms maintain the diversity and uniformity in the objective space. Of course, there will be more challenges to address in multi-/many-objective optimization such as maintaining both diversity and convergence towards constrained Pareto-optima, while the diversity in both feasible and infeasible regions might need to be maintained as well.

On the two airfoil design problems presented here, we have observed that the CHTs with balanced search in both feasible and infeasible regions have better convergence performance than conventional CHTs. It would be interesting to expand the implementation of the CHTs to a more complex aerodynamic design problems such as wing design.

Bibliography

- [1] George R Anderson, Marian Nemec, and Michael J Aftosmis. Aerodynamic shape optimization benchmarks with error control and automatic parameterization. In *53rd AIAA Aerospace Sciences Meeting*, page 1719, 2015.
- [2] Zhoujie Lyu, Gaetan KW Kenway, and Joaquim RRA Martins. Aerodynamic shape optimization investigations of the common research model wing benchmark. *AIAA Journal*, 53(4):968–985, 2015.
- [3] Anand Amrit, Xiaosong Du, Andrew S Thelen, Leifur T Leifsson, and Slawomir Koziel. Aerodynamic design of the rae 2822 in transonic viscous flow: Single-and multi-point optimization studies. In *35th AIAA Applied Aerodynamics Conference*, page 3751, 2017.
- [4] Gaetan KW Kenway and Joaquim RRA Martins. Multipoint aerodynamic shape optimization investigations of the common research model wing. *AIAA Journal*, 54(1):113–128, 2016.
- [5] David A Burdette, Gaetan K Kenway, and Joaquim Martins. Performance evaluation of a morphing trailing edge using multipoint aerostructural design optimization. In *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 0159, 2016.
- [6] Marco Mangano and Joaquim Martins. Multipoint aerodynamic shape optimization for subsonic and supersonic regimes. In *AIAA Scitech 2019 Forum*, page 0696, 2019.
- [7] David Koo and David W Zingg. Investigation into aerodynamic shape optimization of planar and nonplanar wings. *AIAA Journal*, pages 250–263, 2018.
- [8] Philip E Gill, Walter Murray, and Michael A Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM review*, 47(1):99–131, 2005.
- [9] Anirban Chaudhuri, Raphael T Haftka, Peter Ifju, Kelvin Chang, Christopher Tyler, and Tony Schmitz. Experimental flapping wing optimization and uncertainty quan-

- tification using limited samples. *Structural and Multidisciplinary Optimization*, 51(4):957–970, 2015.
- [10] Shuanghou Deng, Mustafa Percin, and Bas van Oudheusden. Experimental investigation of aerodynamics of flapping-wing micro-air-vehicle by force and flow-field measurements. *AIAA Journal*, 54(2):588–602, 2016.
- [11] Takehisa Kohira, Hiromasa Kemmotsu, Oyama Akira, and Tomoaki Tatsukawa. Proposal of benchmark problem based on real-world car structure design optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 183–184, 2018.
- [12] Nathan Ricks, Panagiotis Tsirikoglou, Francesco Contino, and Ghader Ghorbaniasl. A cfd-based methodology for aerodynamic-aeroacoustic shape optimization of airfoils. In *AIAA Scitech 2020 Forum*, page 1729, 2020.
- [13] Raed I Bourisli and Firas Z Hamadeh. Optimizing naca airfoil thickness function parameters for maximum lift-to-drag ratio. In *AIAA Scitech 2020 Forum*, page 1297, 2020.
- [14] Ravi Teja Nallapu and Jekan Thangavelautham. Design of spacecraft swarm flybys for planetary moon exploration. In *AIAA Scitech 2020 Forum*, page 0954, 2020.
- [15] Garima Aggarwal and Ramanan RV. A unified approach for the optimal constellation design of satellites in low-earth circular/elliptical orbits for continuous coverage. In *AIAA Scitech 2020 Forum*, page 1915, 2020.
- [16] Diogene Alessandro Dei Tos and Nicola Baresi. Genetic optimization for the orbit maintenance of libration point orbits with applications to equuleus and lumio. In *AIAA Scitech 2020 Forum*, page 0466, 2020.
- [17] Rahul Rughani and David Barnhart. Using genetic algorithms for safe swarm trajectory optimization. In *AIAA Scitech 2020 Forum*, page 1916, 2020.
- [18] Kazufumi Uwatoko, Masahiro Kanazaki, Hiroki Nagai, Koji Fujita, and Akira Oyama. Blade element theory coupled with cfd applied to optimal design of rotor for mars exploration helicopter. In *AIAA Scitech 2020 Forum*, page 1284, 2020.
- [19] Witold J Koning, Ethan A Romander, and Wayne Johnson. Optimization of low reynolds number airfoils for martian rotor applications using an evolutionary algorithm. In *AIAA Scitech 2020 Forum*, page 0084, 2020.

- [20] Travis D Skinner, Siddhant Datta, Aditi Chattopadhyay, and Asha Hall. Biaxial fatigue damage in quasi isotropic laminates. In *AIAA Scitech 2020 Forum*, page 0475, 2020.
- [21] Cong-Truong Dinh, Sang-Bum Ma, and Kwang-Yong Kim. Aerodynamic optimization of a single-stage axial compressor with stator shroud air injection. *AIAA Journal*, pages 2739–2754, 2017.
- [22] Konstantinos Sakellariou, Zeeshan A Rana, and Karl W Jenkins. Optimisation of the surfboard fin shape using computational fluid dynamics and genetic algorithms. *Proceedings of the Institution of Mechanical Engineers, Part P: Journal of Sports Engineering and Technology*, 231(4):344–354, 2017.
- [23] Masahiro Suzuki and Koji Nakade. Multi-objective design optimization of high-speed train nose. *Journal of Mechanical Systems for Transportation and Logistics*, 6(1):54–64, 2013.
- [24] Gang Xu, Xifeng Liang, Shuanbao Yao, Dawei Chen, and Zhiwei Li. Multi-objective aerodynamic optimization of the streamlined shape of high-speed trains based on the kriging model. *PloS one*, 12(1), 2017.
- [25] Akira Oyama, Takehisa Kohira, Hiromasa Kemmotsu, Tomoaki Tatsukawa, and Takeshi Watanabe. Simultaneous structure design optimization of multiple car models using the k computer. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–4. IEEE, 2017.
- [26] Felipe Petroski Such, Vashisht Madhavan, Edoardo Conti, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. *arXiv preprint arXiv:1712.06567*, 2017.
- [27] Xingwen Zhang, Jeff Clune, and Kenneth O Stanley. On the relationship between the openai evolution strategy and stochastic gradient descent. *arXiv preprint arXiv:1712.06564*, 2017.
- [28] Edoardo Conti, Vashisht Madhavan, Felipe Petroski Such, Joel Lehman, Kenneth Stanley, and Jeff Clune. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. In *Advances in neural information processing systems*, pages 5027–5038, 2018.
- [29] Joel Lehman, Jay Chen, Jeff Clune, and Kenneth O Stanley. Safe mutations for deep and recurrent neural networks through output gradients. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 117–124, 2018.

- [30] Joel Lehman, Jay Chen, Jeff Clune, and Kenneth O Stanley. Es is more than just a traditional finite-difference approximator. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 450–457, 2018.
- [31] Donald R Jones. Large-scale multi-disciplinary mass optimization in the auto industry. In *MOPTA 2008 Conference (20 August 2008)*, 2008.
- [32] Akira Oyama, Hiroaki Fukumoto, Tomoaki Tatsukawa, and Nobuo Namura. Benchmark problem based on design optimization problem of wind turbine for power generation. In *Proceedings of the Japanese Evolutionary Computation Symposium*, pages 53–58, 2019.
- [33] John Henry Holland et al. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.
- [34] John R Koza and John R Koza. *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press, 1992.
- [35] Hans-Georg Beyer and Hans-Paul Schwefel. Evolution strategies—a comprehensive introduction. *Natural computing*, 1(1):3–52, 2002.
- [36] Rainer Storn and Kenneth Price. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.
- [37] Kenneth O Stanley and Risto Miikkulainen. A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130, 2003.
- [38] Carlos A Coello Coello. Constraint-handling techniques used with evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 485–506, 2019.
- [39] Kalyanmoy Deb. An efficient constraint handling method for genetic algorithms. *Computer methods in applied mechanics and engineering*, 186(2-4):311–338, 2000.
- [40] Thomas P. Runarsson and Xin Yao. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on evolutionary computation*, 4(3):284–294, 2000.
- [41] S. Koziel and Z. Michalewicz. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary Computation*, 7(1):19–44, 1999.

- [42] Thomas Philip Runarsson and Xin Yao. Search biases in constrained evolutionary optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35(2):233–243, 2005.
- [43] Benjamin W Wah and Yixin Chen. Hybrid constrained simulated annealing and genetic algorithms for nonlinear constrained optimization. In *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546)*, volume 2, pages 925–932. IEEE, 2001.
- [44] Oliver Kramer. A review of constraint-handling techniques for evolution strategies. *Applied Computational Intelligence and Soft Computing*, 2010, 2010.
- [45] Hans-Paul Paul Schwefel. *Evolution and optimum seeking: the sixth generation*. John Wiley & Sons, Inc., 1993.
- [46] Abdollah Homaifar, Charlene X Qi, and Steven H Lai. Constrained optimization via genetic algorithms. *Simulation*, 62(4):242–253, 1994.
- [47] Jeffrey A Joines and Christopher R Houck. On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with ga’s. In *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, pages 579–584. IEEE, 1994.
- [48] Thomas Runarsson and Xin Yao. Constrained evolutionary optimization. In *Evolutionary optimization*, pages 87–113. Springer, 2003.
- [49] Helio JC Barbosa and Afonso CC Lemonge. *An adaptive penalty method for genetic algorithms in constrained optimization problems*. INTECH Open Access Publisher, 2008.
- [50] Zhichao Lu, Kalyanmoy Deb, and Hemant Singh. Balancing survival of feasible and infeasible solutions in constraint evolutionary optimization algorithms. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2018.
- [51] Pei Yee Ho and Kazuyuki Shimizu. Evolutionary constrained optimization using an addition of ranking method and a percentage-based tolerance value adjustment scheme. *Information Sciences*, 177(14):2985–3004, 2007.
- [52] Max de Castro Rodrigues, Beatriz Souza Leite Pires de Lima, and Solange Guimarães. Balanced ranking method for constrained optimization problems using evolutionary algorithms. *Information Sciences*, 327:71–90, 2016.

- [53] Max de Castro Rodrigues, Solange Guimarães, and Beatriz Souza Leite Pires de Lima. E-brm: A constraint handling technique to solve optimization problems with evolutionary algorithms. *Applied Soft Computing*, 72:14–29, 2018.
- [54] DE Goldberg. Genetic algorithms in search, optimization, and machine learning, addison-wesley, reading, ma, 1989. *NN Schraudolph and J*, 3(1), 1989.
- [55] Randy L Haupt and Sue Ellen Haupt. Practical genetic algorithms. 2004.
- [56] Kalyanmoy Deb, Ram Bhushan Agrawal, et al. Simulated binary crossover for continuous search space. *Complex systems*, 9(2):115–148, 1995.
- [57] Francisco Herrera, Manuel Lozano, and Ana M Sánchez. A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study. *International Journal of Intelligent Systems*, 18(3):309–338, 2003.
- [58] Kalyanmoy Deb and Debayan Deb. Analysing mutation schemes for real-parameter genetic algorithms. *IJAISC*, 4(1):1–28, 2014.
- [59] Efrén Mezura-Montes and Carlos A Coello Coello. Constraint-handling in nature-inspired numerical optimization: past, present and future. *Swarm and Evolutionary Computation*, 1(4):173–194, 2011.
- [60] Rafael de Paula Garcia, Beatriz Souza Leite Pires de Lima, Afonso Celso de Castro Lemonge, and Breno Pinheiro Jacob. A rank-based constraint handling technique for engineering design optimization problems solved by genetic algorithms. *Computers & Structures*, 187:77–87, 2017.
- [61] What is k? <https://www.r-ccs.riken.jp/en/k-computer/about/>. Accessed: 2020-06-03.
- [62] Benchmark problem based on real-world car structure design optimization: Mazda bechmark problem. <http://ladse.eng.isas.jaxa.jp/benchmark/>. Accessed: 2020-08-07.
- [63] Jean Dickinson Gibbons and Subhabrata Chakraborti. *Nonparametric Statistical Inference: Revised and Expanded*. CRC press, 2014.
- [64] Yohanes Bimo Dwianto, Hiroaki Fukumoto, and Akira Oyama. On improving the constraint-handling performance with modified multiple constraint ranking (mcr-mod) for engineering design optimization problems solved by evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 762–770, 2019.

- [65] JJ Liang, Thomas Philip Runarsson, Efren Mezura-Montes, Maurice Clerc, Ponnuthurai Nagaratnam Suganthan, CA Coello Coello, and Kalyanmoy Deb. Problem definitions and evaluation criteria for the cec 2006 special session on constrained real-parameter optimization. *Journal of Applied Mechanics*, 41(8):8–31, 2006.
- [66] Rammohan Mallipeddi and Ponnuthurai Nagaratnam Suganthan. Problem definitions and evaluation criteria for the cec 2010 competition on constrained real-parameter optimization. *Nanyang Technological University, Singapore*, 24, 2010.
- [67] Guohua Wu, R Mallipeddi, and PN Suganthan. Problem definitions and evaluation criteria for the cec 2017 competition on constrained real-parameter optimization. *National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, Technical Report*, 2017.
- [68] Ali Sadollah, Ardeshir Bahreininejad, Hadi Eskandar, and Mohd Hamdi. Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. *Applied Soft Computing*, 13(5):2592–2612, 2013.
- [69] Carlos A Coello Coello. Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry*, 41(2):113–127, 2000.
- [70] Tapabrata Ray and Kim-Meow Liew. Society and civilization: An optimization algorithm based on the simulation of social behavior. *IEEE Transactions on Evolutionary Computation*, 7(4):386–396, 2003.
- [71] BK Kannan and Steven N Kramer. An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. 1994.
- [72] Najeh Ben Guedria. Improved accelerated pso algorithm for mechanical engineering optimization problems. *Applied Soft Computing*, 40:455–467, 2016.
- [73] Mopta 2008 benchmark. <https://www.miguelanjos.com/jones-benchmark>. Accessed: 2020-08-09.
- [74] The 3rd evolutionary computation competition. <http://www.jpnnsec.org/files/competition2019/EC-Symposium-2019-Competition-English.html>. Accessed: 2020-08-09.
- [75] Daniel J Poole, Christian B Allen, and Thomas CS Rendall. A generic framework for handling constraints with agent-based optimization algorithms and application to aerodynamic design. *Optimization and Engineering*, 18(3):659–691, 2017.

- [76] Oleg Chernukhin and David W Zingg. Multimodality and global optimization in aerodynamic design. *AIAA journal*, 51(6):1342–1354, 2013.
- [77] DJ Poole, CB Allen, and TCS Rendall. Global optimization of wing aerodynamic optimization case exhibiting multimodality. *Journal of Aircraft*, 55(4):1576–1591, 2018.
- [78] Richard H Bartels, John C Beatty, and Brian A Barsky. *An introduction to splines for use in computer graphics and geometric modeling*. Morgan Kaufmann, 1995.
- [79] Kozo Fujii and Yoshiaki Tamura. Capability of current supercomputers for the computational fluid dynamics. In *Proceedings of the 1989 ACM/IEEE conference on Supercomputing*, pages 71–80, 1989.
- [80] K Fujii. Developing an accurate and efficient method for compressible flow simulations-an example of cfd in aeronautics. 1989.
- [81] Kozo Fujii and N Satofuka. Flashback: 30 years numerical fluid mechanics and aerodynamics in japan, other asian countries and the western pacific rim. In *100 Volumes of 賽朗 otes on Numerical Fluid Mechanics 賽風*, pages=109–115, year=2009, publisher=Springer.
- [82] K Fujii and S Obayashi. Practical applications of new lu-adi scheme for the three-dimensional navier-stokes computation of transonic viscous flows. In *24th Aerospace Sciences Meeting*, page 513, 1986.
- [83] S Obayashi, K Kuwahara, Kozo Fujii, and K Matsushima. Improvements in efficiency and reliability for navier-stokes computations using the lu-adi factorization algorithm. In *24th Aerospace Sciences Meeting*, page 338, 1986.
- [84] Kozo Fujii and Shigeru Obayashi. Navier-stokes simulations of transonic flows over a wing-fuselage combination. *AIAA journal*, 25(12):1587–1596, 1987.
- [85] Shigeru Obayashi, Kozo Fujii, and Sharad Gavali. Navier-stokes simulation of wind-tunnel flow using lu-adi factorization algorithm. 1988.
- [86] Kozo Fujii. Progress and future prospects of cfd in aerospace 賽埜 ind tunnel and beyond. *Progress in Aerospace Sciences*, 41(6):455–470, 2005.
- [87] Kozo Fujii. Cfd contributions to high-speed shock-related problems. *Shock Waves*, 18(2):145, 2008.
- [88] Rae 2822 transonic airfoil: Study 4? <https://www.grc.nasa.gov/WWW/wind/valid/raetaf/raetaf04/raetaf04.html>. Accessed: 2020-06-19.

- [89] JJ Thibert, M Grandjacques, and LH Ohman. 'experimental data base for computer program assessment,' agard ar-138, 1979.
- [90] David Rodriguez, Peter Sturdza, Yoshifumi Suzuki, Hervé Martins-Rivas, and Andy Peronto. A rapid, robust, and accurate coupled boundary-layer method for cart3d. In *50th AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition*, page 302, 2012.