Contribution to multi-objective optimization under constraints : applications to structural mechanics

Ahmed Tchvagha Zeine

Normandie Université

THESE

Pour obtenir le diplôme de doctorat

Spécialité: **MECANIQUE**

Préparée au sein de « INSA de Rouen Normandie»
En partenariat international avec l'EMI Rabat, Maroc

Contribution à l'optimisation Multi-objectifs sous contraintes: Applications à la mécaniques des structures

Présentée et soutenue par

Ahmed TCHVAGHA ZEINE

Thèse soutenue publiquement le 04 Juillet 2018 devant le jury composé de		
Prof. Adnan Yassine	Université du Havre	Examinateur
Maitre de conférence HDR. Emmanuel Pagnacco	INSA de Rouen Normandie	Examinateur
Prof. Rajae Aboulaich	EMI, Rabat	Rapporteur
Prof. Pierre-Richard DAHOO	Université de Versailles St-Q	Rapporteur
Prof. Bouchaib Radi	Université Hassan 1er, Settat	Rapporteur
Prof. Rachid ELLAIA	EMI, Rabat	Directeur de thèse
Prof. Abdelkhalak EL HAMI	INSA, Rouen	Directeur de thèse

Thèse dirigée par Abdelkhalak EL HAMI, laboratoire de Mécanique de Normandie

INSA ROUEN NORMANDIE

ED PSIME

LMN

إهداء

إلى روح جدتي

التي اختطفها الموت وانا بعيد في غربتي

إلى والداي

النور الذي ينير لي درب النجاح

إلى إخوتي

من تقاسم معي الحياة، و كانو عوني علي الزمن

الي زوجتي

رفيقة الدرب و الغربة

إلى أساتذتي

الشموع التي تحترق لتضئ للآخرين، كل من علمني حرفا

إلى زملائي وزميلاتي

من شجعوا و ساندو و دعو

إليك صغيرتي مريم

أهدي هذا العمل المتواضع، راجيا من الله أن يجد القبول والنجاح

Je dédie ce travail

A Mes très chers parents, frères et sœurs

Aucune dédicace ne saurait exprimer l'amour et l'estime que j'ai pour vous…
Pour leur aide, leur affection et leur amour.
Puisse Dieu, le tout puissant, te préserver et t'accorder santé, longue vie et bonheur.

A Ma femme

A Mes amis(es)

Sans vous aide, vos conseils et vos encouragements ce travail n'aurait vu le jour.

A

***Tous** ceux qui ont participé de près ou de loin à la réalisation de ce travail…*

Ahmed

Remerciements

(Celui qui ne remercie pas les gens, ne remercie pas Allah.) Le prophète Mohammed(saw) (570 - 632).

C'est avec une immense gratitude que je tiens à exprimer ma reconnaissance envers les personnes qui m'ont encouragées et soutenues tout au long de ma thèse.

Je tiens à exprimer mes vifs remerciements au Professeur ELLAIA Rachid, mon directeur de thèse à l'Ecole Mohammadia d'Ingénieurs (EMI) pour son encadrement continu, pour ses remarques constructives qu'il m'a fournies ainsi que pour ses précieux conseils durant toute la période de ma thèse. Je le remercie également pour la confiance et la grande liberté d'idées et de travail qu'il m'a accordées. En dehors de ses apports scientifiques, je n'oublierai pas aussi de le remercier pour ses qualités humaines, son hospitalité et son soutien permanent qui m'ont permis de mener à bien ma thèse doctorat.

Je suis très reconnaissant à M. EL HAMI Abdelkhalak, mon directeur de thèse à l'Institut National des Sciences Appliquées (INSA) de Rouen pour les nombreuses discussions fructueuses, suggestions et remarques constructives qui m'ont permis d'améliorer mes connaissances et la qualité de mon manuscrit. Je le remercie également d'avoir fait preuve d'une patience remarquable pour me motiver et me mettre sur la bonne voie lors de mon séjour en France pour achever ce travail.

Je remercie également M. Adnan Yassine, Professeur à Université du Havre, de m'avoir fait l'honneur de présider le jury de ma soutenance. Ma gratitude et mes sincères remerciements vont à Mme ABOULAICH Rajae, responsable de laboratoire LERMA, pour sa gentillesse, son soutien et ses directives, et d'avoir accepté sans hésitation d'évaluer et de rapporter mes travaux. Je remercie le Professeur Pierre-Richard DAHOO de l'UVSQ de rapporter mon travail. Ses conseils et ses remarques pertinentes m'ont été très utiles. Un grand merci au Professeur Bouchaib Radi de FST de Settat qui m'a fait l'honneur d'accepter d'être rapporteur. Je le remercie vivement pour le temps qu'il a consacré à la

lecture de ce manuscrit.

Je ne manquerai pas l'occasion pour remercier, mon cher Professeur M. PAGNACCO Emmanuel, non seulement pour sa rigueur scientifique, son aide permanente, mais aussi pour la confiance qu'il n'a jamais cessé de témoigner à mon égard, et pour ses encouragements dans des moments où je commence à perdre l'espoir. Merci cher Emmanuel!

Je tiens à remercier ma deuxième famille LERMA où j'ai appris beaucoup de choses qui ont changé ma façon de penser. Mes remerciements s'adressent premièrement aux associés: KAICER Mohamed, OUHIMOU Siham, DAROUICHI Aziz, MOUSSAID Noureddine, et ELALEM Wafae, qui ont été toujours là pour partager leurs expériences et pour remonter le moral par leurs encouragements. Je ne pourrai pas oublier mes soeurs GALLAB Maryam et MEDARHRI Ibtissam, avec qui j'ai partagé des moments précieux que je ne l'oublierai jamais. Elles ont toujours été à mes côtés dans les bons moments comme dans des moments difficiles.

Je remercie également LERMANY Youssef, KHOUAJA Mohamed Ali, TABBAKH Zineb, DAQAQ Fatima-Ezzahra, GANNOUNI Asma, BELMAATI Fatima-Ezzahra, TABBAKH Zineb avec qui j'ai partagé des moments inoubliables.

Mes remerciements s'adressent également aux membres du laboratoire LOFIMS (LMN) pour leur accueil et leur soutien.

Si cette thèse a pu aboutir, le mérite va spécialement à ma très chère famille. Je pense tout d'abord à mon grands-parents, mes parents et mes tantes (Dah, Chcha, Mama, Mohamed, Mohamed Lemin, Selem Bouha, Kena, Fatimetou) qui sans eux je ne serais pas ce que je suis aujourd'hui et j'espère être toujours votre fierté.

J'adresse toute mon affection à ma chérie Khadi, à mes très chères soeurs Mansoura, Najah et Lala et à mes frères Yahya et Yacoub qui ont été toujours à mes côtés pour me soutenir et m'encourager.

Je tiens à remercier également mes amis Khaled, El Hadi, Brahim, Dr. Tar, Dr. Didi, Heyballa, Aziz, Abdellahi NABGHA et Loqman pour leurs accueils, leurs supports et avec lesquels j'ai partagé des moments agréables.

Enfin, je saisie l'occasion pour exprimer ma gratitude envers toutes les personnes ayant contribué de près ou de loin à l'achèvement de ce travail.

A vous tous, je dis merci!

Abstract

The objective of this thesis is the development of multi-objective optimization methods for solving mechanical design problems. Indeed, most of the real problems in the field of mechanical structures have several objectives that are often antagonistic. For example, it is about designing structures by optimizing their weight, their size, and their production costs. The goal of multi-objective optimization methods is the search for compromise solutions between objectives given the impossibility to satisfy all simultaneously.

Metaheuristics are optimization methods capable of solving multi-objective optimization problems in a reasonable calculation time without guaranteeing the optimality of the solution (s). In recent years, these algorithms have been successfully applied to solve the problem of structural mechanics.

In this thesis, two metaheuristics have been developed for the resolution of multi-objective optimization problems in general and of mechanical structures design in particular. The first algorithm called **NNIA+X** is a hybridization of an immune algorithm and three crossover inspired by the original crossover operator of the BSA algorithm. The second one named **MOBSA** used the crossover and mutation operators of the BSA algorithm.

To evaluate the effectiveness and efficiency of these two algorithms, tests on some problems in literature have been made with a comparison with algorithms well known in the field of multi-objective optimization. The comparison results using metrics widely used in the literature have shown that our two algorithms can compete with their predecessors.

Keywords : Metaheuristics, Evolutionary Algorithms, Backtracking Search, Multi-objective optimization, Hybrid algorithm, Structural design, Structural optimization.

Résumé

L'objectif de cette thèse est le développement de méthodes d'optimisation multi-objectif pour la résolution de problèmes de conception des structures mécaniques. En effet, la plupart des problèmes réels dans le domaine de la mécanique des structures ont plusieurs objectifs qui sont souvent antagonistes. Il s'agit, par exemple, de concevoir des structures en optimisant leurs poids, leurs tailles, et leurs coûts de production. Le but des méthodes d'optimisation multi-objectif est la recherche des solutions de compromis entre les objectifs étant donnée l'impossibilité de satisfaire tous simultanément.

Les métaheuristiques sont des méthodes d'optimisation capables de résoudre les problèmes d'optimisation multi-objectif en un temps de calcul raisonnable sans garantie de l'optimalité de(s) solution(s). Au cours des dernières années, ces algorithmes ont été appliqués avec succès pour résoudre le problème des mécaniques des structures.

Dans cette thèse deux métaheuristiques ont été développées pour la résolution des problèmes d'optimisation multi-objectif en général et de conception de structures mécaniques en particulier. Le premier algorithme baptisé **NNIA+X** est une hybridation d'un algorithme immunitaire et de trois croissements inspirés de l'opérateur de croisement original de l'algorithme BSA. Le deuxième algorithme nommé **MOBSA** utilise les opérateurs de croisement et de mutation de l'algorithme BSA. Pour évaluer l'efficacité et l'efficience de ces deux algorithmes, des tests sur quelques problèmes dans littérature ont été réalisés avec une comparaison avec des algorithmes bien connus dans le domaine de l'optimisation multi-objectif. Les résultats de comparaison en utilisant des métriques très utilisées dans la littérature ont démontré que ces deux algorithmes peuvent concurrencer leurs prédécesseurs.

Mots clés: Métaheuristiques, Algorithmes évolutionnaires, Recherche Backtracking , Optimisation multi-objectif, algorithme hybride, Conception structurelle, Optimisation des structures.

Contents

List of Figures

List of Tables

List of Algorithms

Nomenclature

Notation	Explanation
SOP:	Single-objective Problem.
MO:	Muli-objective Optimization.
Ω:	Decision space or feasible solution region.
$\mathbf{F}$:	is the vector of concurrent objective functions to optimize.
m:	is the number of objective functions.
$\mathbf{x}$:	decision variable.
$n_{\mathbf{x}}$:	dimensional decision space where each decision variable.
$x_{li} \leq x_i \leq x_{ui}$:	is bounded by lower and upper limits for $i = 1, \cdots, n_x$.
$g_j(\mathbf{x})$:	inequality constraints.
$h_k(\mathbf{x})$:	equality constraints.
$\mathbf{X}$:	random initial population
$\mathcal{F}^*$:	true Pareto front.
$\widehat{\mathcal{F}}$:	numerical approximation of the true Pareto front.
$\lvert . \rvert$:	defends the cardinality or number of elements in set.
$\widehat{\mathbf{X}}$:	Pareto optimal solutions.
$\widehat{\mathbf{X}}$:	numerical approximation of the Pareto optimal solutions.
$\widehat{\mathbf{x}}_k$:	is a candidate and the set of candidates is the population $\widehat{\mathbf{X}}$.
$\mathscr{R}(0,\ 1)$:	is a uniform distribution returns a real number between 0 and 1.
SBX:	Simulated Binary Crossover.

Chapter I

Introduction

Contents

Nowadays, the industrial world e.g. mechanics civil engineering is evolving from traditional design process to model-based ones. This dynamic is due to two main issues. Firstly, industrial sector aims to become more and more profitable and competitive to ensure its economic viability. Secondly, technological and scientific progress are making available more and more efficient mathematical and computer tools, by means of which the particular specifications of a large spectrum of different systems (and structures) can be simulated [3].

More than any other discipline, the computer sciences has been a revolution for mathematics: by allowing numerical simulation of complex mathematical equations. Thus, the design and analyzis of computer-based computing methods have become a new branch of mathematics. These advances have also enabled mathematics to tackle much more complex real word problems, stemming from immediate industrial or scientific motivations. Simulations allow to provide both qualitative and quantitative answers.

Structural mechanics such as bridges, buildings, machinery or any other load-bearing or load-bearing element is a complex problem in the field of civil engineering, where structural integrity affects safety and functionality. As is usually the case in this and other disciplines, structural design issues require the optimization of several conflicting objectives, such as minimizing the total investment cost while maximizing the security of the final structure. Problems with more than one objective function to be optimized are known as multi-objective optimization (OM) problems, and their main feature is the lack of a single solution that can optimize all objectives at the same time. Instead, solving these problems consists of a set of alternative compromise solutions

In this thesis we focus on the context of structural design, the implication is that there is no single design that minimizes the structure's weight (to reduce the investment cost as much as possible) and, at the same time, maximizes the stiffness (to provide the highest possible degree of safety). Instead, there is a set of designs providing a trade-off between these two conflicting optimization criteria. There is no single method available to effectively solve all optimization problems. Several optimization algorithms have been proposed, examined and analyzed in recent decades. However, optimization in the field of engineering remains an active field of research, since several real optimization problems remain very complex in reality and difficult to solve by the existing algorithms.

Metaheuristics [4] are a class of approximation algorithms which have become popular alternatives for solving both singe to multi-objective optimization problems. According to [5] "In spite of the popularity of multi-objective metaheuristics, a recent survey on multi-objective optimization applied to structural design [6] revealed that the application of metaheuristic techniques in this field has been scarce until very recently "

Multi-Objective Evolutionary Algorithms (MOEAs), which are a specific type of metaheuristic, are considered as, highly suitable for solving Multi-objective optimization Problems due to their ability to compute an approximation to the Pareto front in a single run of the algorithm. In recent years, MOEAs such as the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [7], the Strength Pareto Evolutionary Algorithm 2 (SPEA2) [8] and Multi-Objective Evolutionary Algorithm based on Decomposition MOEA/D [2], have been successfully applied in problems from structural mechanics field. "which are two of the most popular MOEAs in the literature. It is worth noting, however, that NSGA-II and SPEA2 date back to 2000 and there have been significant developments since then. For example, new metaheuristic techniques, evolutionary and non-evolutionary based, have been proposed but rarely have they been applied to structural design problems [5]"

Motivated by the aforementioned gaps, this work raises from the necessity of developing efficient and effective methods in the field of the multi-objective structure design. In this regard, our work explores the multi-objective formulation of the structure and mechanical design problem and the use of MOEAs, for solving for solving such problems.

Gaol

The main goal of this research is to develop effective and efficient MOEAs techniques for structural mechanical and design. These algorithms do well to give the two basic objectives of MOEAs:

- Minimize the distance of the generated solutions to fit the true Pareto-optimal set

• Maximize the diversity of the achieved Pareto-set approximation

• More over, we aim to analyze of the obtained solutions from a structural & mechanical design point of view.

Contributions

This thesis has contributed with the following:

• A new method based-MOEAs using two operators of Backtracking Search Algorithm. This method is called MOBSA.

• We carefully test MOBSA on a set of issues traditionally used by the community, namely `UF`,CF [2]. MOBSA was for two applications in mechanical structures: fourteen bars and two bars random vibration problem.

• A new method, called NNIA+X, based on immune algorithm has bee proposed.NNIA+X is inspired by recombination operator of Backtracking Search algorithm. This algorithm was tested on the `ZDT` problems [9] and DTLZ problems [1], and application in ten bars problems.

As a result of these, the following lists the papers directly derived from this thesis or those where ideas from this work have been used:

• **Journal**

- **A. Tchvagha Zeine**, A. El hami, R.Ellaia and E. Pagnacco, Backtracking Search Algorithm for Multi-objective Design Optimization, *International Journal of Mathematical Modelling and Numerical Optimisation, Vol. 8, No. 2, 2017.*
- **A. Tchvagha Zeine**, N. El Hami, S. Ouhimmou, R. Ellaia and A.El Hami, Multiobjective optimization of trusses using Backtracking Search Algorithm, *Uncertainties and Reliability of Multiphysical Systems., Vol. 1, N°1. (2017) (Published by ISTE Ltd. London, UK).*

- **A. Tchvagha Zeine**, E. Pagnacco, A. El hami and R. Ellaia, An immune multi-objective optimization with Backtracking Search algorithm inspired recombination, *Engineering Applications of Artificial Intelligence, Submitted in 16-Nov-2016.*

• **Congress**

- R. El-Maani, **A. Tchvagha Zeine**, R. Bouchaib, A. El-Hami, and R. Ellaia, Backtracking search optimization algorithm for fluid-structure interaction problems, *4th International Colloquium on Information Science and Technology (IEEE-CiSt'2016), pp.690-695, 24-26, October 2016, Tangiers-Assilah, Morocco.*
- **A. Tchvagha Zeine**, R. Ellaia and A. El-Hami, Hybrid Backtracking Immune Algorithm for Multi-objective Optimization. *TAMTAM15, 04-08 mai 2015, Tanger (Maroc).*
- **A. Tchvagha Zeine**, R. Ellaia and A. El-Hami, Backtracking Immune Algorithm for Continuous Multi-objective Optimization. *IFORS-2017: 21st Conference of the International Federation of Operational Research Societies. 17-21, 2017 July in Quebec City, Canada.*

Structure of the thesis

The structure of this thesis reflects these contributions with four chapters:

• In chapter II, we describe basic concepts related to the topic of single and multi-objective optimization. The general concepts of metaheuristics are described, with emphasis on multi-objective evolutionary algorithms.

• In chapter IV, we present our contribution to multi-objective optimization namely the multi-objective optimization backtracking search algorithm (MOBSA) that we have developed. was, comparisons with some known multi-objective optimization methods are presented.

• Chapter V, presents the application of MOBSA for solving multi-objective design structure problems.

• In chapter VI, we introduce a new hybrid immune algorithm using a crossover operator inspired form backtracking Search algorithm. We also applied this new algorithm structure design of ten bars problem.

We will conclude the manuscript with conclusions and perspectives of the work carried out as well as appendices providing additional information.

Chapter II

The multi-objective Evolutionary Algorithms: State of the art

Contents

Introduction

According to the *Cambridge English dictionary*, the optimization is "*the act of making something as good as possible*". Mathematically speaking, the optimization consists of identifying the solution i.e. single or many decision variable(s), minimizing or maximizing a given function or a set of functions while fulfilling a set of constraints. Accordingly, two kind of optimization problems are distinguished in the literature: single and multi-objective optimization problems.

In the past, the most commonly utilized optimization techniques were gradient-based method that used gradient information to search for solution space near an initial starting point [10]. Compared to stochastic approaches the gradient-based methods converge faster and can obtain solutions with higher for the problem convex. The gradient-based methods can not be easily solved non-convex optimization problems. Other types of optimization methods, known as metaheuristic algorithms, are stochastic approaches. These methods are suitable for global research because of their ability to explore and find promising areas in the search space at an affordable computational time [4, 11]. These algorithms tend to perform well for most of the optimization problems applications in different fields, namely, applied mathematics, engineering, medicine, economics, and other sciences. These methods are extensively utilized in the design of different systems in civil, mechanical, electrical, and industrial engineering [10, 12, 13].

In this chapter, the Section 2, we present some basic definitions related to an optimization problem. In the next Section 3, we introduce the main methods for solving the optimization problems. In Section 4, we devoted to review some of the most Metaheuristics evolutionary algorithms designed for solving Single and multi-objective optimization.

From Single to multi-objective optimization problem

In real-world optimization applications, it is often necessary to optimize single objective or multiple objectives in one problem simultaneously. These problems are usually referred to as Single-objective optimization problem (SOP) or multi-objective optimization (MO) problems. The both types is presented as follows:

Single-objective optimization problem

A general Single-objective optimization problem (SOP) is defined as minimizing (*or maximizing*) an objective function **f**. A SOP can be defined as follows [14, 15]:

$$\begin{cases} \min\limits_{\mathbf{x}\in\Omega} \mathbf{f}(\mathbf{x}) \\ \text{Subject to} \\ \quad g_j(x) \leq 0, \text{ for } j = 1, \cdots, l \\ \quad h_k(x) = 0, \text{ for } k = 1, \cdots, p \\ \quad x_{li} \leq x_i \leq x_{ui}, \text{ for } i = 1, \cdots, n_x, \end{cases} \tag{II.1}$$

where:
n_x-dimensional decision space where each decision variable x_i, x_{li}, x_{ui} bounded by lower and upper limits respectively, $g_j(\mathbf{x})$ are l inequality constraints and $h_k(\mathbf{x})$ are p equality constraints.

Definition 2.1 (Single-Objective Global Minimum Optimization) *:*
The function $\boldsymbol{f}$ *admits a global minimum if and only if:*

$$\forall \boldsymbol{x} \in \Omega : -\infty < \boldsymbol{f}(\boldsymbol{x}^*) \leq \boldsymbol{f}(\boldsymbol{x}) \tag{II.2}$$

$\boldsymbol{x}^*$ *is by definition the global minimum solution,* $\boldsymbol{f}$ *is the objective function, and the set* Ω *is the feasible region of* $\boldsymbol{x}$*. The goal of determining the global minimum solution(s) is called the* global optimization problem *for a single-objective problem.*

The Figure II.1 illustrates the case of a single-objective optimization problem for global optimum and local optimum.

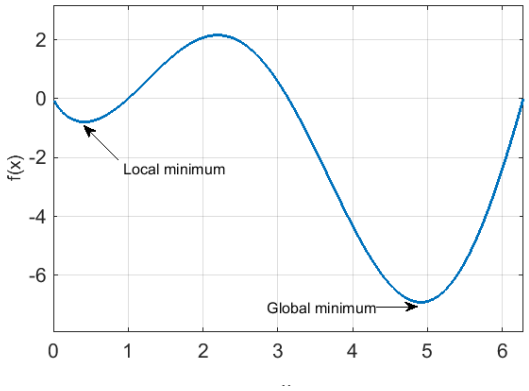

Figure II.1 – Global optimum and local optimum..

Multi-objective optimization problem

Most of real world optimization problems involve two or more, often conflicting, design objectives. Mathematically speaking, a multi-objective optimization problem can be formulated as follows [1, 15–17]:

$$\begin{cases} \min\limits_{\mathbf{x}\in\Omega} \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \cdots, f_m(\mathbf{x}))^{\mathrm{T}} \\ \text{Subject to:} \\ \qquad g_j(x) \leq 0, \text{ for } j = 1, \cdots, l \\ \qquad h_k(x) = 0, \text{ for } k = 1, \cdots, p \end{cases} \tag{II.3}$$

where $\mathbf{F}$ is the vector of concurrent objective functions to optimize, m is the number of objective functions, $\mathbf{x} = (x_1, \cdots, x_n) \in \Omega$ is the n_x-dimensional decision space where each decision variable x_i is bounded by lower and upper limits $x_{li} \leq x_i \leq x_{ui}$ for $i = 1, \cdots, n_x$. $g_j(\mathbf{x})$ are l inequality constraints and $h_k(\mathbf{x})$ are p equality constraints.

The Figure II.2 illustrates the case of a multi-objective optimization problem involving three decision variables and two objective functions.

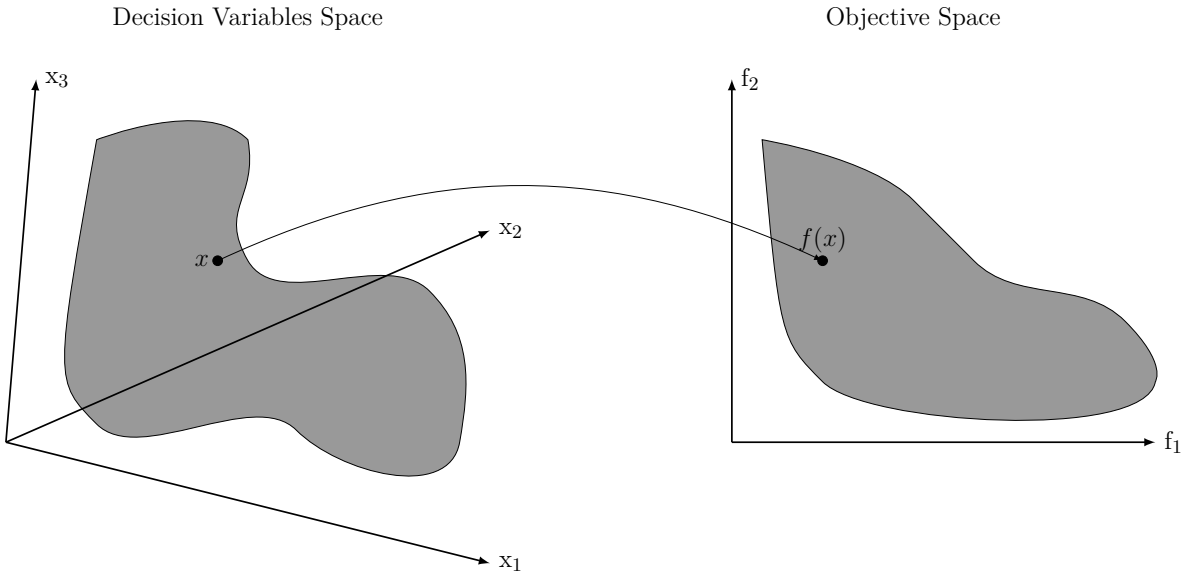

Figure II.2 – A multi-objective optimization problem having three variables and two objectives. The feasible search space is projected to the corresponding objective space.

The solution of such problems is very difficult compared to single-objective optimization. Indeed, for Multi-objective Optimization (MO) Problems, as objectives are usually conflicting, there is no one optimal solution but a set of trade-offs solutions. This set of trade-offs solutions is known as Pareto-optimal set back to the famous Italian economist

Vilfrido Pareto [18].

Pareto dominance and optimality

The main goal of multi-objective optimization is to find an optimal solution for the problem described in II.3. To resolve this problem, most modern multi-objective optimization algorithms use the concept of Pareto dominance. The principle of dominance is to compare two solutions in order to determine wether one of these dominates the other one or not. The Pareto dominance related is defined as follows:

Definition 2.2 (***Pareto dominance***)

Suppose $\boldsymbol{x}_a$ *and* $\boldsymbol{x}_b$ *are two different feasible solutions to the MO problem. Then* $\boldsymbol{x}_a$ *dominates* $\boldsymbol{x}_b$ *(noted as* $\boldsymbol{x}_a \preceq \boldsymbol{x}_b$*) if and only if:*

$$f_i(\boldsymbol{x}_a) \leq f_i(\boldsymbol{x}_b)) \, \forall i \in \{1, \cdots, m\} \tag{II.4}$$

and:

$$\exists k \in \{1, \cdots, m\} \; f_k(\boldsymbol{x}_a) < f_k(\boldsymbol{x}_b) \tag{II.5}$$

Definition 2.3 (***Pareto-optimal solution:***)

A solution $\boldsymbol{x}^*$ *is said to be Pareto-optimal if there is no solution that dominates it:*

$$\nexists \; \boldsymbol{x} \in \Omega : \; \boldsymbol{x} \preceq \boldsymbol{x}^* \tag{II.6}$$

Definition 2.4 (***Pareto-optimal set:***)

For a given MO Problem, the set X^* *of all Pareto-optimal solutions is called the Pareto-optimal set and defined as follows:*

$$\mathrm{X}^* = \{\boldsymbol{x}^* \in \Omega / \nexists \boldsymbol{x} \in \Omega, \boldsymbol{f}(\boldsymbol{x}) \prec \boldsymbol{f}(\boldsymbol{x}^*)\} \tag{II.7}$$

Definition 2.5 (***Pareto-optimal front:***)

The Pareto-optimal front is the set $\mathcal{F}^*$ *of values or outcomes of all the objective functions which corresponds to the solutions:*

$$\mathcal{F}^* = \{\mathbf{f}(\boldsymbol{x}^*) = (f_1(\boldsymbol{x}^*), ..., f_m(\boldsymbol{x}^*))^{\mathrm{T}} \textit{ such that: } \boldsymbol{x}^* \in X^*\} \tag{II.8}$$

Constraints handling

The search space of a constrained MO Problem can be formulated as follows:

$$\mathcal{C} = \begin{cases} g_j(\mathbf{x}) \leq 0, \text{ for } j = 1, \cdots, l \\ h_k(\mathbf{x}) = 0, \text{ for } k = 1, \cdots, p \\ x_{li} \leq x_i \leq x_{ui} \text{ for } i = 1, \cdots, n_x. \end{cases} \tag{II.9}$$

where, $g_j(\mathbf{x})$ are l inequality constraints and $h_k(\mathbf{x})$ are p equality constraints.

Due to the presence of the constraints, the search space is partitioned into feasible and infeasible regions. Many constraints' handling methods have been proposed to solve constrained MO problem [19, 20]. In this thesis, two different constraint handling methods namely *Penalty method* and *constraint dominance method* have been used.

Penalty method:

The idea of penalty methods was first introduced in [21]. This methods consists in replacing the constrained optimization problems by an optimization problems without constraints, by introducing new objective functions to be optimized:

$$\phi_k(\mathbf{x}) = f_k(\mathbf{x}) + r\varphi_q(\mathbf{x}) \tag{II.10}$$

where the penalty function chosen here is :

$$\varphi_q(\mathbf{x}) = \sum_{j=1}^{l} \max\{0, g_j(\mathbf{x})\}^q + \sum_{k=1}^{p} |h_j(\mathbf{x})|^q, \tag{II.11}$$

where q is the penalty exponent and r is a positive penalty parameter. The problem is then solved directly for a value of r large enough, so the constraints are satisfied.

Constraints' dominance method:

This methods proposed in [7], for solving constrained multi-objective optimization is based on the concept of constrained domination, which is also known as superiority of the feasible solution. In the presence of constraints, each solution can be either feasible or infeasible. A solution $\mathbf{x}_a$ is said to constraint-dominates a solution $\mathbf{x}_b$ if any of the following conditions is true:

1. $\mathbf{x}_a$ is feasible and $\mathbf{x}_b$ is infeasible.
2. $\mathbf{x}_a$ and $\mathbf{x}_b$ are infeasible and $\mathbf{x}_a$ has a smaller constraint violation value.

3. $\mathbf{x}_a$ and $\mathbf{x}_b$ are feasible and $\mathbf{x}_a$ dominates $\mathbf{x}_b$ with the usual dominance principle.

Decision Making

In most multi-objective optimization problems, the optimal Pareto set consists of a large or even infinite number of solutions. Nevertheless, in practice, only one solution should be selected from the optimal Pareto set. The person who has the task of choosing the most practical solution from the optimal Pareto set is called the Decision Maker (DM).

The DM preferences are incorporated to induce a total order between the Pareto set elements. There are several approaches in which DM preferences can be incorporated. For example, the DM can classify the set of objectives according to their importance. Another possibility is to obtain a sample of the Pareto front, then to select a solution from this sample. A common classification of MO Problem resolution techniques is based on the time when the DM is required to provide his/her preference information.

For this reason, different decision marking support techniques have been developed for introducing DM preferences in the optimization process [22] This classification is the following:

1. **a priori methods:** It characterizes the so-called 'priori' methods used for their simplicity of implementation and their great generically. Indeed, the objectives of the optimization problem are transformed into a single objective function. Many aggregation's methods including weighted sum, fuzzy integrals, Tchebytchev functions, ... are available in the literature [15]. In the weighted sum case, the decision maker is supposed to quantify a priori the importance of each criterion in order to build a single function. The single-objective optimization process is then launched to determine the *optimal* solution.

2. **a posteriori methods:** In the posterior methods, one seeks to find the optimal Pareto solutions without the DM intervening during the process of resolution. Therefore, the method must provide the DM with the totality of the found Pareto front, then the DM chooses the most suitable solutions for him/her. These methods do not require prior knowledge of the problem and therefore there is no need to model the decision maker's preferences. However, the huge number of solutions obtained can make the optimal Pareto set difficult to analyze for the DM.

3. **interactive methods:** Interactive methods aim to involve the DM throughout the research process. Therefore, throughout the resolution process the DM interacts with the resolution method. The DM can define his/her preferences in a clear and

understandable way. The process is iterated many times until the DM is satisfied. The most efficient methods for obtaining a set of optimal solutions are the posterior methods as the choice of a solution is made after having solved the optimization problem.

Resolution methods

For finding the global optimum, different methods are available in the literature. These methods can be classified into three main categories [23, 24]:

- Enumerative Methods,
- Deterministic Methods,
- Stochastic Methods.

In the following we have presented these three methods:

Enumerative Methods

The enumerative methods evaluate the objective function(s) for each feasible solution. These methods are interesting and efficient in case where the optimization problem has a limited number of solutions. The performances of such methods break down in case of large space problem as it might be impossible to search all the candidate solutions in the space [23].

Deterministic Methods

Deterministic methods do not rely on any random mechanism. They, in contrast, require some mathematical properties of the problem at hand such as drivability and use them to construct a set of potential solutions that converge to a global optimal solution. The deterministic methods produce the same outputs for the same inputs and thus the reproductivity of their results is too easy. These types of methods, as applied to nonlinear minimization problems, generally rely on establishing an iterative procedure, which, after a certain number of iterations, will hopefully converge to the minimum of the objective function(s). The iterative procedure can be written in the following general form [25, 26]

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \gamma \mathbf{d}^t$$

where
γ the search step size
d is the direction of descent
tk is the iteration number.

The most famous deterministic methods available in the literature are described in the following sub-subsections.

Gradient-based methods

The gradient method is probably one of the oldest optimization algorithms for solving non-linear optimization problems [27]. These methods proceed as follows:

we choose a starting point x^0 and we calculate the gradient of the objective function **f** in the point x^0. The gradient indicates the direction of the greatest increase in the objective function **f**, we move a quantity λ^0 in the opposite direction to the gradient and we have the next point x^1:

$$x^1 = x^0 - \lambda^0 \frac{\bigtriangledown \mathbf{f}(x^0)}{\| \bigtriangledown \mathbf{f}(x^0) \|}$$

A series of points $x^0, x^1, \cdots, x^k$ is then obtained, which approach the optimum.

$$x^{k+1} = x^k - \lambda^k \frac{\bigtriangledown \mathbf{f}(x^k)}{\| \bigtriangledown \mathbf{f}(x^k) \|}$$

$\lambda^k > 0$ define the pace of displacement at each iteration. If the displacement step has a given value, the method is called not determined.

The major disadvantage of these methods is the slow convergence for some types of functions, in fact the method may converge towards a local optimum depending on the chosen starting point. In flat or steep regions, convergence will be slowed down considerably

Newton-based methods

Newton-based methods construct an approximation of the Hessian matrix, using only the evolution of the gradients and assuming that the function can be locally approximated by a quadratic limited expansion around the optimum.

The Newton method, is a powerful for solving single-objective Optimization. we assume that the objective function is of class $\mathbf{C}^2$ is positive.

$$\mathbf{x}^{k+1} = x^k - | \bigtriangledown \mathbf{f}(\mathbf{x}^k)|^{-1} + \bigtriangledown^2 \mathbf{f}(\mathbf{x}^k)$$

There is abundant literature on Newton-based methods. We refer the interested reader to the references [28, 29].

Stochastic Methods

As indicated by their name, the stochastic methods use a stochastic process to guide the search to find the minimum of the objective function(s) **f**. The majority of optimization problems are difficult to tackle using the two previous exploration methods as the necessary computation time can be unreasonable or even infinite. This is the case of optimization problems having large and/or complex search spaces. This is also the case of the problems using numerical models without any algebraic information on the link between the decision variables, the objective functions, and constraints. For this kind, Figure II.3 shows an example of black-box optimization problems, the stochastic methods are considered as suitable and efficient. The stochastic methods can be divided into two main groups: *single-solution search methods* and *multiple-solutions search methods also named population-based search methods*. The first group evolve one single solution while the last group evolve a set (population) of solutions in the search space towards the optimal solution without guarantee to attain it. The simulated annealing algorithm proposed in [30] is one of the most known single-solution search methods. This algorithm is inspired from physics laws of thermodynamic evolution to search for the minimum-energy states. The evolutionary algorithms such as genetic algorithms are unboundedly the most known population-based search algorithms. Under the term of metaheuristics algorithms, a large number of both types of stochastic search methods is available in the literature. The next section is dedicated to those algorithms designed for solving difficult optimization problems.

Figure II.3 – Illustration of a complex black-box optimization problem. for optimizing system .

Metaheuristics

The word *metaheuristic* is derived from two Greek words, *meta* which means beyond, in a higher level, and *heuristic* which means the art of inventing, making discoveries. The etymological decomposition of the word thus makes it possible to understand its meaning: a heuristic that makes it possible to find other heuristics, which favors emergence. As a reminder, heuristics are simple empirical rules that are not based on sometimes complex scientific analyzes, but on the experience and relationships accumulated over the results. These rules therefore simply use past results to optimize future research by first examining the more plausible cases [11, 31]. More simply, it will be said that metaheuristics form a family of optimization algorithms designed to solve difficult optimization problems, often coming from the field of operations research, engineering or artificial intelligence.

Metaheuristics appeared in the early 1980s with the aim of tackling difficult optimization problems for which no more efficient classical optimization methods are known.

According to [32] "Metaheuristics are a family of non-exact optimization methods for finding high quality solutions to complex optimization problems which cannot be solved effectively by exact techniques". Metaheuristics cannot, in general, guarantee to find optimal solutions but they tend to produce near-optimal solutions with a reasonably low computational effort. This class of approaches includes, among others, for example: *Evolutionary Algorithms*, *Tabu Search*, *Simulated Annealing* and *Particle Swam Optimization*. *Evolutionary Algorithms* (EA), which are by far the most well-known and widely used metaheuristics [5]. Algorithm 1 describes the Metaheuristics.

Algorithm 1 Pseudo code of Metaheuristics

Function $\widehat{\mathbf{X}} = \text{metaheuristics}\left(n_{\mathbf{x}}, m, \mathbf{f}\left(\mathbf{x}\right), \mathbf{x}_l, \mathbf{x}_u, n_{it}\right)$

1: Generate a uniform random initial population $\widehat{\mathbf{X}}$ of size $n_C \times n_x$ in respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
2: $\widehat{\mathbf{X}} := \left(\widehat{\mathbf{X}} \mid \mathbf{f}\left(\mathbf{x}\right)\right)$, Evaluation;
3: **for** $t = 0 : n_{it}$, **do**
4: $\quad \mathbf{X}_G \leftarrow$ generation $\left(\widehat{\mathbf{X}}\right)$;
5: $\quad \widehat{\mathbf{X}}_t := \left(\widehat{\mathbf{X}}_g \mid \mathbf{f}\left(\mathbf{x}\right)\right)$, Evaluation;
6: $\quad \widehat{\mathbf{X}} \leftarrow \left(\widehat{\mathbf{X}}; \widehat{\mathbf{X}}_t\right)$;
7: **end for**

Simulated annealing

Simulated annealing [30] is inspired by the physical annealing process. The simulated annealing process repeats an iterative procedure that looks for lower cost configurations while controllably accepting configurations that degrade the cost function. We now give the pseudo-code of a typical simulated annealing algorithm for a minimization problem:

Algorithm 2 Pseudo code of Simulated Annealing

Function $\mathbf{x}^* = \mathrm{SA}\left(\mathtt{V}_f, \mathtt{T}_0, \mathtt{S}, \mathbf{f}\left(\mathbf{x}\right), \mathbf{x}_l, \mathbf{x}_u, n_{it}\right)$

Choose, at random, an initial solution $\mathbf{x}_0$ in respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
$\mathbf{x} \leftarrow \mathbf{x}_0$ and $\mathtt{T} \leftarrow \mathtt{T}_0$;
while $\mathrm{T} > \mathrm{S}$, **do**
  $i \leftarrow 0$;
  while $i < n_{it}$; **do**
    $\mathbf{x}^{'} \in \mathrm{V}_f\left(\mathbf{x}\right)$;
    $\Delta\mathbf{f} := \mathbf{f}\left(\mathbf{x}^{'}\right) - \mathbf{f}\left(\mathbf{x}\right)$;
    if $\Delta\mathbf{f} < 0$; **then**
      $\mathbf{x} := \mathbf{x}^{'}$;
    else
      Generate random $r := \mathscr{R}\left(0,\ 1\right)$ uniform distribution in the range;
      if $r < e^{\frac{\Delta\mathbf{f}}{\mathrm{T}}}$; **then**
        $\mathbf{x} := \mathbf{x}^{'}$;
      end if
    end if
  end while
  $\mathrm{T} := \alpha\left(\mathrm{T}\right)$;
end while

where T_0 is the initial temperature, S the minimum threshold that the temperature can reach, α the function decreasing the temperature at certain levels,V_f the neighborhood function.

Other variations of this method have been proposed to deal with continuous search spaces [33]. A detailed presentation of this method is given in [15]

Tabu search

Tabu Search (TS), developed by Glover [34] in 1986, was designed to manage an embedded local search algorithm. The main mechanisms of TS is inspired by the human memory. TS uses a memory mechanism to explore the search space from while escaping from local optima. this Thus, TS uses the memory mechanism to learn from the past explored path. More details from the basic concepts of TS till more recent developments can be found in [34–36]. Tabu search algorithm has been initially developed for combinatorial optimization problems but has been adapted for continuous optimization problems [37]. The Pseudo-code of Tabu Search are presented in Algorithm 3, Ω given a set of feasible solutions and $\mathrm{N}(\mathbf{x})$ defend neighborhood for each feasible solution.

Algorithm 3 Pseudo code of Tabu Search

Choose an initial solution $\mathbf{x} \in \Omega$;
Set $\mathbf{x}^* = \mathbf{x}$;
Generate a subset V^* of solution in $N(\mathbf{x})$;
for $t = 0 : n_{it}$, **do**
  Choose a best $\mathbf{v} \in V^*$ and set $\mathbf{x} = \mathbf{v}$;
  if $\mathbf{f}(\mathbf{x}) < \mathbf{f}(\mathbf{x}^*)$ **then**
    set $\mathbf{x}^* = \mathbf{x}$;
  end if
  return best solution $\mathbf{x}^*$;
end for

Evolutionary algorithms

Evolutionary algorithm (EA) a generic population-based metaheuristic optimization algorithm. An EA uses mechanisms inspired by natural evolution [6, 33]. EAs use a population based approach in which more than one solution participates in an iteration and evolves a new population of solutions in each iteration [16].

1. EAs do not require any derivative information
2. EAs are relatively simple to implement
3. EAs are flexible and have a wide-spread applicability.

A typical EA follows the pseudo code included in Algorithm 4.

Algorithm 4 Pseudo code of EA

Function $\widehat{\mathbf{X}} = \mathrm{EA}(n_{\mathbf{x}}, m, \mathbf{f}(\mathbf{x}), \mathbf{x}_l, \mathbf{x}_u, n_{it})$
Generate a uniform random initial population $\mathbf{X}$ of size $n_{\mathbf{X}} \times n_{\mathbf{x}}$ in respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
$\widehat{\mathbf{X}} := \left(\widehat{\mathbf{X}} \mid \mathbf{f}(\mathbf{x})\right)$, Evaluation;
for $t = 0 : n_{it}$, **do**
  $\mathbf{X}_s \leftarrow$ Selection $\left(\widehat{\mathbf{X}}\right)$;
  $\mathbf{X}_r \leftarrow$ Reproduction $\left(\widehat{\mathbf{X}}_s\right)$;
  $\widehat{\mathbf{X}}_t := \left(\widehat{\mathbf{X}}_r \mid \mathbf{f}(\mathbf{x})\right)$, Evaluation;
  $\widehat{\mathbf{X}} \leftarrow \left(\widehat{\mathbf{X}}; \widehat{\mathbf{X}}_t\right)$;
end for

Evolutionary Programming

Evolutionary Programming (EP) introduced by Fogel [38] in 1960s. Evolutionary Programming was originally developed to simulate evolution as a learning process aiming to generate artificial intelligence. In the classical formulation, the predictors were evolved in

the form of finite state machines. Nowadays, there are several variants of EP for optimizing real-valued parameters, which have become the standard EP [1, 39]. The standard EP algorithm shows in Algorithm 5.

Algorithm 5 Pseudo code of Evolutionary Programming

Function $\widehat{\mathbf{X}} = \mathrm{GA}\left(n_{n_{\mathbf{x}}}, m, \mathbf{f}\left(\mathbf{x}\right), \mathbf{x}_l, \mathbf{x}_u, n_{it}\right)$

1: Generate a uniform random initial population $\mathbf{X}$ of size $n_{\mathbf{X}} \times n_{\mathbf{x}}$ in respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
2: $\widehat{\mathbf{X}} := \left(\mathbf{X} \mid \mathbf{f}\left(\mathbf{x}\right)\right)$, Evaluation;
3: **for** $t = 0 : n_{it}$, **do**
4: Apply a mutation operator over the individuals in $\mathbf{X}$ to create $\mathbf{X}_m$ with $n_{\mathbf{X}}$ children ;
5: $\widehat{\mathbf{X}}_t := \left(\mathbf{X}_m \mid \mathbf{f}\left(\mathbf{x}\right)\right)$, Evaluation;
6: Choose the best $n_{\mathbf{X}}$ individuals from $\mathbf{X} \cup \widehat{\mathbf{X}}_t$to form the population $\widehat{\mathbf{X}}$ for the next generation
7: **end for**

Genetic Algorithm

The genetic algorithm is a technique of optimization and research based on the principles of genetics and natural selection. The **Genetic Algorithms** (GA) method was originally developed by John Holland in [40]. Several changes to this formulation have been developed, including different kinds of representations, *selection*, *crossover* (recombination), *mutation*, and *reproduction*. The pseudo code of GA is presented in the algorithm 7.

Algorithm 6 Pseudo code of Genetic Algorithm

Function $\widehat{\mathbf{X}} = \mathrm{GA}\left(n_{\mathbf{x}}, m, \mathbf{f}\left(\mathbf{x}\right), \mathbf{x}_l, \mathbf{x}_u, n_{it}\right)$

1: Generate a uniform random initial population $\mathbf{X}$ of size $n_{\mathbf{X}} \times n_{\mathbf{x}}$ in respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
2: $\widehat{\mathbf{X}} := \left(\mathbf{X} \mid \mathbf{f}\left(\mathbf{x}\right)\right)$, Evaluation;
3: **for** $t = 0 : n_{it}$, **do**
4: $\mathbf{X}_s :=$ Select $n < n_{\mathbf{X}}$ parents from the $\widehat{\mathbf{X}}$;
5: $\mathbf{X}_c :=$ Create n new offsprings, starting from the parents selected previously, $(\mathbf{X}_s)$;
6: $\mathbf{X}_m :=$ Mute the offsprings produced in the crossing $\mathbf{X}_c$ step with the probability $\mathrm{p_m}$;
7: $\mathbf{X}_r :=$ Replace the n bad chromosomes of the population with offsprings generated previously $\mathbf{X}_m$;
8: $\widehat{\mathbf{X}}_t := \left(\mathbf{X}_r \mid \mathbf{f}\left(\mathbf{x}\right)\right)$, Evaluation;
9: Choose the best n individuals from $\mathbf{X} \cup \widehat{\mathbf{X}}_t$to form the population $\widehat{\mathbf{X}}$ for the next generation
10: **end for**

Differential Evolution

The differential evolution method developed in [41] is an evolutionary method. This optimization method was proposed by Storn and al [42] in 1997, as an alternative to the genetic algorithm methods. DE is an evolutionary algorithm which has been mainly used to solve continuous optimization problems. DE shares similarities with traditional EAs.

Algorithm 7 Pseudo code of Differential Evolution

Function $\widehat{\mathbf{X}} = \mathrm{EA}\left(n_{\mathbf{x}}, \mathrm{CR}, \mathbf{f}\left(\mathbf{x}\right), \mathbf{x}_l, \mathbf{x}_u, n_{it}\right)$

Generate a uniform random initial population $\mathbf{X}$ of size $n_{\mathbf{X}} \times n_{\mathbf{x}}$ in respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
$\widehat{\mathbf{X}} := \left(\mathbf{X} \mid \mathbf{f}\left(\mathbf{x}\right)\right)$, Evaluation;
for $t = 0 : n_{it}$, **do**
  for $t = 0 : n_{\mathbf{X}}$, **do**
    $r_1 := \mathscr{R}\left(0,\ 1\right), r_2 := \mathscr{R}\left(0,\ 1\right), r_3 := \mathscr{R}\left(0,\ 1\right)$;
    Select randomly $r_1 \neq r_2 \neq r_3$;
    $j_{\mathrm{rand}} := \mathscr{R}\left(0,\ n_{\mathbf{x}}\right)$;
    $a := \mathscr{R}\left(0,\ 1\right)$;
    for $j = 1 : n_{\mathbf{x}}$, **do**
      if $a < \mathrm{C}_r$ or $j = j_{\mathrm{rand}}$; **then**
        $\mathbf{u}_{t+1}(i,j) = \mathbf{x}_t(r_3,j) + \mathtt{F}(\mathbf{x}_t(r_1,j) - \mathbf{x}_t(r_2,j))$;
      else
        $\mathbf{u}_{t+1}(i,j) = \mathbf{x}_t(i,j)$;
      end if
    end for
    if $\left(\mathbf{U}_{t+1}(:,j) \mid \mathbf{f}\left(\mathbf{x}\right)\right) < \left(\mathbf{X}_t(:,j) \mid \mathbf{f}\left(\mathbf{x}\right)\right)$; **then**
      $\mathbf{X}_{t+1}(:,j) = \mathbf{U}_{t+1}(:,j)$;
    else
      $\mathbf{X}_{t+1}(:,j) = \mathbf{U}_t(:,j)$;
    end if
  end for
end for

where $\mathscr{R}\left(0,\ 1\right)$ is a function that returns a real number between 0 and 1. C_r and F are user-defined parameters.

Backtracking Search Algorithm

Backtracking Search optimization Algorithm (BSA), is one of the recently proposed evolutionary algorithms (EAs). Motivation of its development was the need for a simpler and effective search algorithm with a single control parameter. According to the BSA's author, it shows a good convergence performance compared to other algorithms, it is able to solve multi-modal problems and it is not over sensitive to the initial value of the control parameter [43].

BSA's performance results from the combination of exploration of the search space and of the use of a search direction matrix. The algorithm can be decomposed into five successive steps: initialization, selection-I, mutation and crossover operators, and selection-II.

The Pseudo code of BSA is presented in algorithm 8:

Algorithm 8 Pseudo code of BSA

```
Initialization population
for t = 0 : n_it, do
    Selection-I
    mutation
    Crossover
    Selection-II
end for
```

Artificial Immune Systems

Artificial Immune Systems (AIS), a variety of bio-inspired metaheuristics have become popular in the last few years [44]. The first immune optimization algorithm proposed in [45] and included an abstraction of clonal selection to solve computational problems.

The AIS algorithms implement a variety of principles observed in the adaptive part of biological immune systems of the vertebrae [26,25] including the negative selection, clonal selection, and immune networks.

Multi-Objective Evolutionary Algorithms

The goal of this thesis is to solve a multi-objective optimization problems (MO) Problems. In MO Problems there is no single solution that minimizes all objectives simultaneously, but a set of solutions that correspond to the best possible trade-offs among the objectives. When solving a MO Problems, one usually not only wants to find this set of trade-off solutions, but also that these solutions are well-distributed along the Pareto front. Evolutionary Algorithms (EA) as part of metaheuristics, have been recognized to be very successful in solving MO problems by finding a representative set of Pareto optimal solutions within a single run [16, 46, 47].

For several years [48, 49], there is a growing interest in the applying of EAs to deal with MO problems, these EAs are called multi-objective evolutionary algorithms (MOEAs). The optimization process is based of an MOEAs in the iteration adaptation of population until a pre-specified optimization goal is illustration in Algorithm 9.

Several MOEAs have been developed and applied in MO Problems, for example: Multi-Objective Genetic Algorithm (MOGA) [50], Niched Pareto Genetic Algorithm [51], Strength Pareto Evolutionary Algorithm (SPEA) [52], and its improved version (SPEA2) [8], Non-dominated Sorting Genetic Algorithm (NSGA) [53], NSGA-II [7] and Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) [54]. The popular MOEAs are presented in detail, in this section.

Algorithm 9 General MOEAs template

Function $\widehat{\mathbf{X}} = \text{MOEA}\left(n_{\mathbf{x}}, n_{\mathbf{X}}, m, \mathbf{f}\left(\mathbf{x}\right)\right)$

Generate a uniform random two populations $\mathbf{X}$ of size $n_{\mathbf{X}} \times n_{\mathbf{x}}$ respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
$\widehat{\mathbf{X}} := \left(\mathbf{X} \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
$t = 0$;
while stopping criterion is not satisfied, **do**
  $\widehat{\mathbf{X}}_t := Evaluate\left(\mathbf{X} \mid \mathbf{f}\left(\widehat{\mathbf{X}}\right)\right)$;
  $\widehat{\mathbf{X}}_t^r := Ranking\left(\widehat{\mathbf{X}} \bigcup \widehat{\mathbf{X}}_t\right)$;
  if $\widehat{\mathbf{X}}_t^r$ not full **then**
    $\widehat{\mathbf{X}}_r := Crowding\_Distance\left(\widehat{\mathbf{X}}_t'' \bigcup \widehat{\mathbf{X}}_t\right)$;
    $\widehat{\mathbf{X}}_t := \left(\widehat{\mathbf{X}}_t^r \bigcup \widehat{\mathbf{X}}_r\right)$;
  end if
end while

Non-dominated sorting genetic algorithm II (NSGA-II)

The Non-dominated sorting genetic algorithm II (NSGA-II), was proposed by Deb et al. [7]. NSGA-II is a revised version of the Non-dominated Sorting Genetic Algorithm (NSGA) proposed by Srinivas and Deb [55].

The pseudo-code of the NSGA-II is shown in Algorithm 10. First, NSGA-II uses the Pareto dominance to classify individuals in the parent's population according to their non-domination level. Then, NSGA-II applies evolutionary operators (selection, crossover, and mutation) to create an offspring population having the same size as the parent population. The combined parents and and offspring population is then partitioned into fronts following the dominance ranks of individuals. Finally, the population for the next generation is selected from the combined population considering the ranked fronts and the crowding distance. Figure II.4 shows a sketch of one iteration from NSGA-II. The crowding distance used in the selection operator of NSGA-II is designed to maintain diversity within the population. NSGA-II procedure has three features:

1. It uses an elitist principle
2. It emphasizes non-dominated solutions.
3. It uses an explicit diversity preserving mechanism

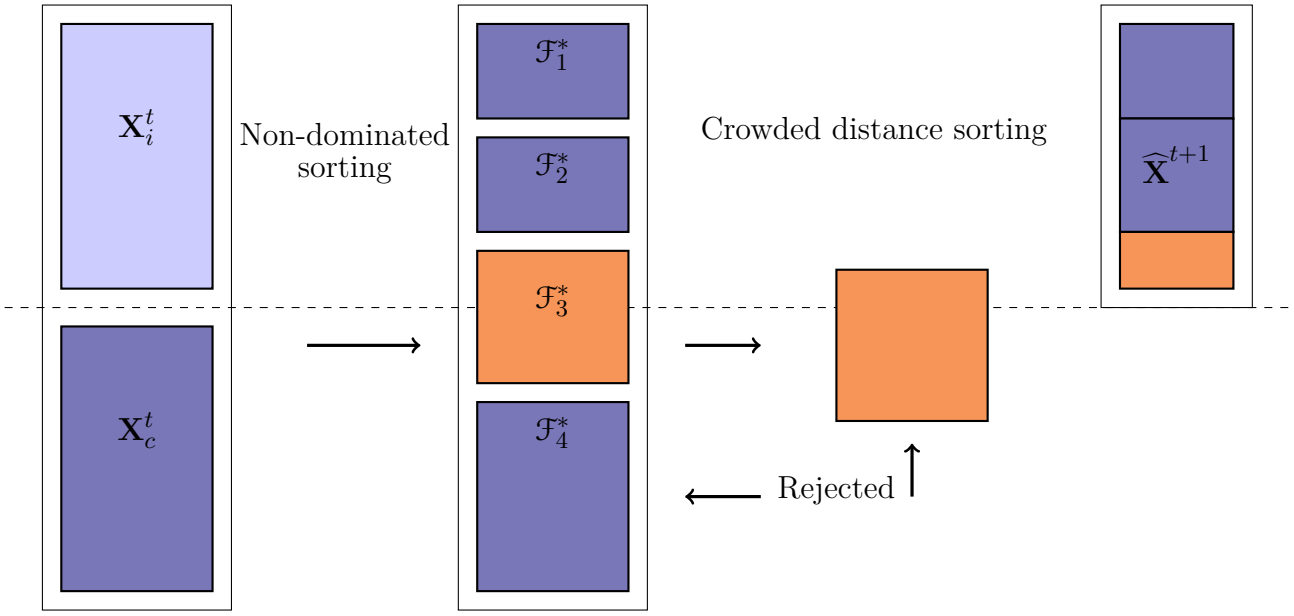

Figure II.4 – Diagram that shows the way in which the NSGA-II works.

Algorithm 10 Pseudo code of the NSGA-II algorithm

Function $\widehat{\mathbf{X}} = \mathrm{NSGA-II}\left(n_{\mathbf{x}}, n_{\mathbf{X}}, m, \mathbf{f}\left(\mathbf{x}\right), n_{it}\right)$

Generate a uniform random two populations $\mathbf{X}$ of size $n_{\mathbf{X}} \times n_{\mathbf{x}}$ respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
$\widehat{\mathbf{X}} := \left(\mathbf{X} \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
$\widehat{\mathbf{X}} := Find\_Non\_Dominated\left(\widehat{\mathbf{X}} \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
for $t = 1 : t_{it}$, **do**
  $\widehat{\mathbf{X}}'_t := Binary\_Tournament\left(\widehat{\mathbf{X}}\right)$;
  $\widehat{\mathbf{X}}''_t := Crossove\&Mutation\left(\widehat{\mathbf{X}}'_t\right)$;
  $\widehat{\mathbf{X}}''_t := \left(\mathbf{X} \mid \mathbf{f}\left(\widehat{\mathbf{X}}''_t\right)\right)$;
  $\widehat{\mathbf{X}}^r_t := Ranking\left(\widehat{\mathbf{X}}''_t \bigcup \widehat{\mathbf{X}}_t\right)$;
  if $\widehat{\mathbf{X}}^r_t$ not full **then**
    $\widehat{\mathbf{X}}_r := Crowding\_Distance\left(\widehat{\mathbf{X}}''_t \bigcup \widehat{\mathbf{X}}_t\right)$;
    $\widehat{\mathbf{X}}_t := \left(\widehat{\mathbf{X}}^r_t \bigcup \widehat{\mathbf{X}}_r\right)$;
  end if
end for

Strength Pareto Evolutionary Algorithm 2 (SPEA2)

"Strength Pareto Evolutionary Algorithm2 (SPEA2) introduced by [8]. a revised version of was proposed by Eckart and al. [52]. This approach has been conceived as a way to integrate different evolutionary algorithms of multi-objective optimization. SPEA uses an external archive $\mathbf{X}_e$. Algorithm 11 shows the pseudo-code for SPEA2.

"SPEA2 has three main differences with respect to its predecessor:

1. it incorporates a fine-grained fitness assignment strategy which takes into account for each individual the number of individuals that dominate it and the number of individuals to which it dominates;

2. it uses a nearest neighbor density estimation technique which guides the search more efficiently,

3. it has an enhanced archive truncation method that guarantees the preservation of boundary solutions.

”

Algorithm 11 Pseudo code of the SPEA2 algorithm

Function $\widehat{\mathbf{X}} = \text{SPEA2}\left(n_{\mathbf{x}}, n_{\mathbf{X}}, m, \mathbf{f}\left(\mathbf{x}\right), n_{it}, n_{\mathbf{X}}, n_{\mathbf{X}_t}\right)$

Generate a uniform random population $\mathbf{X}_t$ of size $n_{\mathbf{X}} \times n_{\mathbf{X}}$ in respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
Create an empty external archive $\mathbf{X}^e \to \emptyset$ of size $n_{\mathbf{X}^e}$;
$t \to 0$
while stopping criterion is not satisfied, **do**
    $\widehat{\mathbf{X}}_t := \left(\mathbf{X}_t \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
    $\widehat{\mathbf{X}}_t^e := \left(\mathbf{X}_t^e \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
    $\mathbf{X}_{t+1} := Find\_Non\_Dominated\left(\widehat{\mathbf{X}} \bigcup \mathbf{X}_t^e\right)$;
    if $n_{\mathbf{X}_{t+1}} > n_{\mathbf{X}_t}$ **then**
        Use the truncation operator to remove elements;
    else
        $n_{\mathbf{X}_{t+1}} < n_{\mathbf{X}_t}$
        Use dominated individuals in $\mathbf{X}_t$ to fill $\mathbf{X}^e$
    end if
end while

Pareto Archived Evolution Strategy (PAES)

The Pareto Archived Evolution Strategy (PAES) a (1+1) evolution strategy initially introduced by Knowles and Corne [56]. PAES uses local search from a population of one and use a reference archive of previously found solutions. Thus archive is used identify the approximate dominance ranking of the current solution vector. A description of PAES is presented in Algorithm 5. An initial individual is first created, who serves as a parent to create new solutions. The next step is to create an offspring from the parent individual, which is achieved by applying a mutation operator to the parent.

The comparisons are made to decide whether the child will be added to the archive or not, and what solution should be the parent for the next generation. PAES stores the non-dominated solutions found so far during the search in an external archive [15, 17]. The pseudo-code of PAES is shows in algorithm 12.

Algorithm 12 Pseudo code of the PAES algorithm

Function $\widehat{\mathbf{X}} = \text{PAES}\left(n, m, \mathbf{f}\left(\mathbf{x}\right), n_{it}, n_{\mathbf{X}}, n_{\mathbf{X}_t}\right)$

Generate a uniform random individual $\mathbf{x}$;
Create an empty external archive $\mathbf{X}^e \longleftarrow \emptyset$ of size $n_{\mathbf{X}^e}$;
Add $\mathbf{x}$ to external archive $\mathbf{X}^e := \mathbf{X}^e \cup \mathbf{x}$;
$t \rightarrow 0$
while stopping criterion is not satisfied, **do**
  Mutate $\mathbf{x}$ to create $\mathbf{x}_m$;
  if $\mathbf{x} \preceq \mathbf{x}_m$; **then**
    Discard $\mathbf{x}_m$;
  else
    if $\mathbf{x}_m \preceq \mathbf{x}$; **then**
      Replace $\mathbf{x}$ with $\mathbf{x}_m$, and add $\mathbf{x}_m$ to archive $\mathbf{X}^e := \mathbf{X}^e \cup \mathbf{x}_m$;
    end if
  else
    if $\exists \mathbf{a} \in \mathbf{X}^e \mid \mathbf{a} \preceq \mathbf{x}_m$; **then**
      Discard $\mathbf{x}_m$;
    end if
  else
    Apply test $(\mathbf{x}, \mathbf{x}_m, \mathbf{X}^e)$ to determine who becomes the new current solution and whether to add $\mathbf{x}_m$ to $\mathbf{X}^e$;
  end if
end while

The multi-objective evolutionary algorithm based on decomposition (MOEA/D)

The multi-objective evolutionary algorithm based on decomposition (MOEA/D) was proposed by Zhong and al. [54]. MOEA decomposes the at hand MO problems into many Single-objective problems (SOPs). This decompositions a linear or nonlinear weighted aggregations of the initial MO problems objectives. Then, MOEA/D defines neighborhood relations among the resulting SOPs using their aggregations weight vectors. To do so, distances between those aggregations vectors are computed. The smallest the distance between the aggregation vectors of $i$and j, the closed are considered by MOEA/D. MOEA/D optimizes each SOPs using, mainly, information from its neighbors.

In the framework of MOEA/D, several aggregation methods, , such as the weighted sum approach, the Tchebytcheff approach, the penalty boundary intersection can be used to decompose The initial MO problems.

Non-dominated neighbor immune algorithm (NNIA)

Non-dominated neighbor immune algorithm (NNIA) has been proposed for solving MO problems [57]. It is a real-coded antibody population-based iterative EA designed to be a global minimizer which uses recombination.

Algorithm 13 Pseudo code of the MOEA/D algorithm

A uniform spread of N weight vectors: $\lambda^1, ..., \lambda^{n_x}$,
Initialize $\mathbf{X}_e \leftarrow \emptyset$
Compute the Euclidean distance between any two weight vectors and then work out the T closest weight vectors to each weight vector. For each $i = 1, \ldots, n_x$, set $\mathrm{B}(i) = \{i_1, \ldots, i_\mathrm{T}\}$, where $\lambda^{i_1}, ..., \lambda^{i_\mathrm{T}}$ are the closest weight vectors to λ^i.
Generate an initial population $\mathbf{X}$
$\widehat{\mathbf{X}} := (\mathbf{X} \mid \mathbf{f}(\mathbf{x}))$;
Initialize $\mathbf{z} = (z_1, \cdots, z_m)^\mathrm{T}$;
for $i = 1, \ldots, n_x$ **do**
    Randomly select two indexes k, l from $\mathrm{B}(i)$, and then generate a new solution $\mathbf{y}$ from $\mathbf{x}^\mathrm{k}$ and $\mathbf{x}^\mathrm{l}$ by using genetic operators
    Apply a repair/improvement heuristic on $\mathbf{y}$ to produce $\mathbf{y}'$
    for $j = 1, \ldots, m$ **do**
        if $z_j > \mathbf{f}(\mathbf{y})$ **then**
            $z_j = \mathbf{f}(\mathbf{y})$
        end if
    end for
    for $j \in \mathrm{B}(i)$ **do**
        if $\mathrm{g}^{te}(\mathbf{y}' \lambda^j, \mathbf{z}) \leq \mathrm{g}^{te}(\mathbf{x}^j \lambda^j, \mathbf{z})$ **then**
            $\mathbf{x}^j = \mathbf{y}'$;
            $\widehat{\mathbf{X}}(:, j) := \left(\mathbf{y}' \mid \mathbf{f}(\mathbf{x})\right)$;
        end if
    end for
    Update of $\mathbf{X}_e := \widehat{\mathbf{X}}$ to $\mathbf{X}_e$ if it is non-dominated with respect to the vectors stored in $\mathbf{X}_e$, and remove from $\mathbf{X}_e$ the vectors dominated by $\mathbf{f}(\mathbf{y}')$
end for
Return $\mathbf{X}_e$;

The heuristic search of NNIA is: proportional cloning, recombination, and hypermutation. The population storing clones is called clone population. The NNIA algorithm is presented in the algorithm 15.

Algorithm 14 Pseudo code of NNIA

Function $\widehat{\mathbf{X}} = \text{NNIA}\left(n_x, m, \mathbf{f}\left(\mathbf{x}\right), \mathbf{x}_l, \mathbf{x}_u, n_{\hat{X}\max}, n_{A\max}, n_C, n_{it}\right)$

1: Generate a uniform random initial population $\widehat{\mathbf{X}}$ of size $n_C \times n_x$ in respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
2: $\widehat{\mathbf{X}} := Find\_Non\_Dominated\left(\widehat{\mathbf{X}} \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
3: **for** $t = 0 : n_{it}$, **do**
4: $\quad \mathbf{X}_A := CD\_Truncation\left(\widehat{\mathbf{X}}, n_{A\max}\right)$;
5: $\quad \mathbf{X}_C := Cloning\left(\mathbf{X}_A, n_C\right)$;
6: $\quad \mathbf{X}_C := Recombination\left(\mathbf{X}_C, \mathbf{X}_A, \mathbf{x}_l, \mathbf{x}_u\right)$;
7: $\quad \mathbf{X}_C := Hypermutation\left(\mathbf{X}_C, \mathbf{x}_l, \mathbf{x}_u\right)$;
8: $\quad \widehat{\mathbf{X}} := Find\_Non\_Dominated\left(\left[\widehat{\mathbf{X}}; \mathbf{X}_C\right] \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
9: $\quad \widehat{\mathbf{X}} := CD\_Truncation\left(\widehat{\mathbf{X}}, n_{\hat{X}\max}\right)$;
10: **end for**

Performances metrics

Multi-objective optimization algorithms, unlike single-optimization algorithms, aim simultaneously to ensure the convergence to, and to maintain the diversity within, the Pareto-optimal set. Two kinds of metrics are necessary to adequately evaluate such performances. In this thesis, we used the following metrics:

Generational distance

The Generational Distance (GD) proposed in [49], is a measure of the distance between approximate solutions of the Pareto-optimal front $\widehat{\mathcal{F}}$ and true Pareto front $\mathcal{F}^*$. The mathematical formulation of GD is the following:

$$\text{GD}(\mathcal{F}^*, \widehat{\mathcal{F}}) = \frac{1}{|\widehat{\mathcal{F}}|} \sqrt{\sum_{p \in \widehat{\mathcal{F}}} d(p, \mathcal{F}^*)^2}, \tag{II.12}$$

where $d(x, \mathcal{F}^*)$ is the minimum Euclidean distance between p and points of $\mathcal{F}^*$.

Inverted generational distance

Inverted generational distance (IGD), According to [58], it is a variant of the Generational Distance. It measures the distances between each solution composing the true optimal

Pareto front $\mathcal{F}^*$ and the computed approximation $\widehat{\mathcal{F}}$. It is computed as:

$$\mathrm{IGD}(\mathcal{F}^*, \widehat{\mathcal{F}}) = \frac{1}{|\widehat{\mathcal{F}}|} \sqrt{\sum_{p^* \in \mathcal{F}^*} d(p^*, \widehat{\mathcal{F}})^2} \tag{II.13}$$

where $d(p^*, \widehat{\mathcal{F}})$ is the minimum Euclidean distance between $\mathcal{F}^*$ and the points in $\widehat{\mathcal{F}}$.

Spacing metric

The Spacing metric (S), proposed by [59] measures the uniformity of the distribution of the obtained solutions in the objective space. The mathematical formulation of S is the following:

$$\mathrm{S}(\widehat{\mathcal{F}}) = \frac{1}{|\widehat{F}|} \sqrt{\sum_{i=1}^{|\widehat{\mathcal{F}}|} (\overline{d} - d_i)^2}, \tag{II.14}$$

where $\overline{d}$ is the average value of all d_i and $d_i = \min_{j=1}^{|\widehat{\mathcal{F}}|} \left[\sum_{k=1}^{m} |f_k(x_i) - f_k(x_j)| \right]$, for $i \neq j$ and $i = 1, ..., |\widehat{\mathcal{F}}|$.

Hypervolume Metric

The Hyper-volume (Hv) performance measure was proposed by [58, 60]. This measure calculates the volume This metric calculates the hypervolume of the multidimensional objective space enclosed by the approximate approximate set $\widehat{\mathbf{X}}$. Mathematically, for each solution $\widehat{\mathbf{x}} \in \widehat{\mathbf{X}}$, a hyper-cube v_i is constructed with a reference point r_p and the solution $\widehat{\mathbf{x}}$ as the diagonal corners of the hypercube. The reference point can simply be found by constructing a vector of worst objective function values. Thereafter, a union of all hypercubes is found and its HV is calculated:

$$\mathrm{Hv} = \text{volume} \left(\bigcup_{i=1}^{n_{\widehat{\mathbf{x}}}} v_i \right), \tag{II.15}$$

Conclusion

In this chapter, we described the basics of both single and multi-objective optimization problems. We thus presented the main differences between those two optimization problems from the point of view of uniqueness or multiplicity of possible solutions. The Pareto dominance concept used for the multi-objective optimization has been presented. The decision-making step or the approaches of integrating the decision-maker preferences

into the resolution process have been discussed. The second main part of this chapter presented the three main resolution methods available to deal with both single and multi-objective optimization problems. We thus distinguished and presented the enumerative methods, determinist methods, and stochastic methods. In the third section, we presented the metaheuristics methods with particular emphasis on the MOEAs similar to those we used in our research work. We reviewed the NSGA-II, SPEA2, PAES and MOEA/D algorithms. We closed this chapter by presenting some performance metrics dedicated to the evaluation and comparison of the MOEAs performances. The generational distance, inverted generational distance, spacing metrics, and the hyper-volume metrics have been presented.

Chapter III

An immune multiobjective optimization with Backtracking Search algorithm inspired recombination

Contents

Based on: **A. Tchvagha Zeine, E. Pagnacco, A. El Hami and R. Ellaia**, *An immune multiobjective optimization with Backtracking Search algorithm inspired recombination*, Engineering Applications of Artificial Intelligence,

Introduction

Many engineering problems have multiple objectives to be optimized. Except in trivial problems, the objective functions are conflicting and there exists a number of Pareto optimal solutions. Over the past decade, Evolutionary Algorithms (EAs) have been recognized to be very successful in solving Multi-objective Optimization (MO) problems by finding a representative set of Pareto optimal solutions or a Pareto front within a single run. State-of-the-art algorithms are described by, for example: [7, 8, 56, 61]. These algorithms, which are population-based, are able to simultaneously explore various regions of the Pareto front.

In the recent years, immune systems have inspired new algorithms to solve MO Problems. Main principles of Artificial Immune Systems (AIS) algorithm are the clonal selection, the mutation [62] and, more recently, the recombination [57, 63–69]. By using a recombination operator, the non-dominated neighbor-based immune optimization algorithm (NNIA) appears to be efficient and effective to deal with MO problems [57].

NNIA has proved that is is beneficial to embed a crossover operator in the algorithm. To achieve this, it uses simulated binary crossover (SBX). But SBX is a recombination operator which performs search near the recombination parent [70].

Hybridization between algorithms is a promising way to improve their performances. For example, immunity-based hybrid evolutionary algorithm known as HAIS for solving both unconstrained and constrained MO problem [71] and the hybrid immune algorithm of [72] is designed by combining the advantages of Gaussian and polynomial mutation. [73] combines immune algorithm and Baldwinian learning. Reference [74], a novel Micro-population Immune Multi-objective Optimization (MIMO) algorithm. Reference [75] introduces a multi-objective immune algorithm with adaptive differential evolution. In reference [76] proposes a hybrid immune multi-objective optimization algorithm with a novel recombination operator which is a combination of a newly designed difference evolution recombination of [77] and the SBX recombination has also been designed.

In the present work, we propose a novel hybrid MO immune algorithm for tackling continuous MO problems. Similarly to the NNIA, it considers features of MO problems: based on the fitness values, best individuals in the trial population are selected and recombined to guide the search toward the Pareto front. But NNIA uses SBX which has mainly a local search ability. In our algorithm, recombination is basically inspired from the one defined for the BSA but adaptations are found to fit in the immune algorithm. Hence, three variants are designed in this work, resulting in new recombination operators for immune algorithm.

This chapter is organized as follows: In Section 3 and 4, NNIA algorithm and BSA

recombination are presented and we propose new algorithms to solve the MO problem. The effectiveness of these algorithms is investigated in Section 5 when confronting to various MO tests problems. A short summary is proposed to conclude this paper.

Multi-objective solution

In this work, strategy for solving the multi-objective optimization problem II.3 consists in computing a representative set of Pareto optimal solutions $\widehat{\mathbf{X}}$ which approximate X^*. Each solution $\widehat{\mathbf{x}}_k$ is a candidate and the set of candidates is the population $\widehat{\mathbf{X}}$. Its maximum size is predefined by the user and it is referred as the size of dominant population.

Moreover, following the reference [57], terms are defined as in immunology:

1. *Antibody*: An antibody refers to the coding of a decision variable $\mathbf{x}$. In this study, a real-valued representation is adopted, being $\mathbf{x}$ itself.

2. *Crowding distance*: The crowding distance (CD) is a measure for diversity maintenance [7]. It reads:
$$\mathrm{CD}(\widehat{\mathbf{X}}) = \sum_{j=1}^{m} \frac{D_j(\widehat{\mathbf{X}})}{f_j^{\max} - f_j^{\min} + \epsilon_D} \tag{III.1}$$
where $f_j^{\max}$ and $f_j^{\min}$ are maximal and minimal values of the j-th objective respectively, ϵ_D is a small number and:
$$D_j(\widehat{\mathbf{X}}) = \begin{cases} \infty & \text{if } \widehat{\mathbf{x}}_k \text{ is a boundary point of } \widehat{\mathbf{X}} \\ \min \left| f_k\left(\widehat{\mathbf{X}}\right) - f_l\left(\widehat{\mathbf{X}}\right) \right| & \text{otherwise} \end{cases} \tag{III.2}$$
for $k, l \in \{1, \cdots, n_x\}$ such that $k \neq l \neq j$.

Immune optimization algorithm and recombination operator

Non-dominated neighbor immune optimization algorithm

Non-dominated neighbor immune algorithm (NNIA) has been proposed for solving MO problems [57]. It is a real-coded antibody population-based iterative EA designed to be a global minimizer which uses recombination.

From a given trial population and knowing its fitness values, the dominant antibodies are first found. Then, they are ranked according to their crowding-distance in order to

select them for elitism (selection-I). Non-dominated individuals found so far are stored in an external population, called dominant population. This population forms the current approximation of the Pareto set. From the dominant population found so far, only partial less-crowded individuals are selected to form the active population for the next iteration (selection-II). The selection is therefore biased towards individuals with a high isolation value. Then, an heuristic search is applied on this active population in order to form the next trial population and all these process steps are repeated until a predefined number of repetitions is reached.

The heuristic search of NNIA is: proportional cloning, recombination, and hypermutation. The population storing clones is called clone population. Cloning an antibody consists in making multiple identical copies of it [78]. The aim is that the greater the crowding-distance value of an individual, the more times the individual will be reproduced. In NNIA, cloning is proportional [79]. Recombination is performed as a crossover between each clone in the clonal population and a randomly selected active antibody. It is realized by replacing at random some gene segments of the clone by the corresponding ones of the selected active antibody, applying the SBX operator of [80] with boundary control. SBX produces a new antibody from a weighted average a the two selected antibodies where weights comes from a parametric probability distribution. Recombination occurs unconditionally in NNIA, i.e. this event has a probability equal to one and it applies to every time steps. Hypermutation operator used in NNIA is the static polynomial hypermutation with boundary control defined in the reference [79]. By using the non-dominated neighbor-based selection and proportional cloning, NNIA pays more attention to the less-crowded regions of the current trade-off front.

Dominant population, active population and clone population at time t are stored by time-dependent variable matrices $\widehat{\mathbf{X}}$, $\mathbf{X}_A$ and $\mathbf{X}_C$, respectively. Their sizes are $n_{\hat{X}}$, n_A and n_C respectively. The size n_C of the clone population is predefined by the user, as well as the maximum size for the dominant population $n_{\hat{X}\max}$ and for the active population $n_{A\max}$. Hence, three cases can be distinguished, depending of the number of non-dominated antibodies:

1. If the number of non-dominated antibodies is less than $n_{\hat{X}\max}$, an update of the dominant population is performed with all of non-dominated antibodies and $n_{\hat{X}} < n_{\hat{X}\max}$.
2. If the number of non-dominated antibodies is equal to $n_{\hat{X}\max}$, they are all selected to form the dominant population and $n_{\hat{X}} = n_{\hat{X}\max}$.
3. If the number of non-dominated antibodies is greater to $n_{\hat{X}\max}$, a truncation selection is achieved by using the non-dominated neighbor-based selection mechanism:

only the non-dominated antibodies part with greater crowding-distance values is preserved for the dominant population and $n_{\hat{X}} = n_{\hat{X}\max}$.

The same rule applies for the active population: the active population comes from the crowding-distance based truncation selection applied to the dominant population. Hence, time-dependent variable matrices $\widehat{\mathbf{X}}$ and $\mathbf{X}_A$ have generally time-dependent sizes, especially at early stages of the optimization process.

The NNIA algorithm is presented in the algorithm 15 where := is the update operator. Inputs are $\mathbf{x}_l$ and $\mathbf{x}_u$ the lower and upper limits vectors (respectively), n_D, $n_{\hat{X}\max}$, $n_{A\max}$, n_C and n_{it}, the number of iterations. The reason for using an initial population with a uniform distribution of solutions over the allowable range of the decision variables is to sample the search space uniformly.

Algorithm 15 Pseudo code of NNIA

Function $\widehat{\mathbf{X}} = NNIA\left(n_x, m, \mathbf{f}\left(\mathbf{x}\right), \mathbf{x}_l, \mathbf{x}_u, n_{\hat{X}\max}, n_{A\max}, n_C, n_{it}\right)$

Generate a uniform random initial population $\widehat{\mathbf{X}}$ of size $n_C \times n_x$ in respect to $\mathbf{x}_l$ and $\mathbf{x}_u$;
$\widehat{\mathbf{X}} := Find\_Non\_Dominated\left(\widehat{\mathbf{X}} \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
for $t = 0 : n_{it}$, **do**
  $\mathbf{X}_A := CD\_Truncation\left(\widehat{\mathbf{X}}, n_{A\max}\right)$;
  $\mathbf{X}_C := Cloning\left(\mathbf{X}_A, n_C\right)$;
  $\mathbf{X}_C := Recombination\left(\mathbf{X}_C, \mathbf{X}_A, \mathbf{x}_l, \mathbf{x}_u\right)$;
  $\mathbf{X}_C := Hypermutation\left(\mathbf{X}_C, \mathbf{x}_l, \mathbf{x}_u\right)$;
  $\widehat{\mathbf{X}} := Find\_Non\_Dominated\left(\left[\widehat{\mathbf{X}}; \mathbf{X}_C\right] \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
  $\widehat{\mathbf{X}} := CD\_Truncation\left(\widehat{\mathbf{X}}, n_{\hat{X}\max}\right)$;
end for

Empirical comparisons with various MO algorithms and several reference problems have been performed in [57]. It has been found that NNIA was able to quite converge to the true Pareto-optimal fronts in solving most of the reference problems.

Recombination and crossovers

NNIA recombination

In NNIA, the SBX is introduced as the recombination operator since a high level of similarity is assumed between biological evolution and the production of antibodies [57]. This operator has been adopted in many MO EAs ([7, 56],...). It simulates the working principle of the single-point crossover on binary-strings.

For a recombination operation, an antibody of the cloning population and an anti-

body of the active population are selected and modified as:

$$\{\mathbf{X}_C\}_{ij} := \begin{cases} \frac{1-\beta}{2}\{\mathbf{X}_C\}_{ij} + \frac{1+\beta}{2}\{\mathbf{X}_A\}_{kj} & \text{if} \quad a=1 \,\&\, b=0 \\ \frac{1+\beta}{2}\{\mathbf{X}_C\}_{ij} + \frac{1-\beta}{2}\{\mathbf{X}_A\}_{kj} & \text{if} \quad a=1 \,\&\, b=1 \\ \{\mathbf{X}_C\}_{ij} & \text{if} \quad a=0 \end{cases} \quad | \, a \sim \mathscr{U}(0,1), \, b \sim \mathscr{U}(0,1)$$

for $i \in \{1,\ldots,n_C\}$, $j \in \{1,\ldots,n_x\}$ and k a random integer uniformly chosen in $\{1,\ldots,n_A\}$. Above, $\mathscr{U}$ is the uniform discrete distribution and β is a sample from a random distribution having the density:

$$p(\beta) = \begin{cases} 0 & \text{if} \quad \beta < 0 \\ \frac{\eta+1}{2}\beta^{\eta} & \text{if} \quad 0 \leq \beta \leq 1 \\ \frac{\eta+1}{2\beta^{\eta+2}} & \text{if} \quad \beta > 1 \end{cases}$$

where η is a real non-negative distribution index, which is chosen by the user. Hence, four independent random variables are involved in this recombination operation: a, b, k and β. In addition, a boundary control is performed by a simple projection:

$$\{\mathbf{X}_C\}_{ij} := \begin{cases} x_{li} & \text{if} \quad \{\mathbf{X}_C\}_{ij} < x_{li} \\ \{\mathbf{X}_C\}_{ij} & \text{if} \quad x_{li} \leq \{\mathbf{X}_C\}_{ij} \leq x_{ui} \\ x_{ui} & \text{if} \quad \{\mathbf{X}_C\}_{ij} > x_{ui} \end{cases}$$

Crossover operator of backtracking search optimization algorithm

BSA has been recently proposed for solving single-objective optimization problems, based on an attempt for a simpler search algorithm [**?**]. It is a real-valued population-based iterative EA designed to be a global minimizer. BSA is based on five successive processes: selection-I, mutation, crossover, boundary control and selection-II. Analysis of BSA shows that mutation is the main search operator of this algorithm, while crossover enables to mix population.

Crossover strategy of BSA is very simple. It consists in mixing two inputs populations to form a new output population. Let us denote $\mathbf{X}_P$ and $\mathbf{X}_Q$ the two inputs populations and $\mathbf{X}_R$ the output population, such that all of them are of equal sizes: $n_X \times n_x$. Then, BSA' crossover reads:

$$\{\mathbf{X}_R\}_{ij} := \begin{cases} \{\mathbf{X}_P\}_{ij} & \text{if} \quad \mathbf{T}_{ij} = 0 \\ \{\mathbf{X}_Q\}_{ij} & \text{if} \quad \mathbf{T}_{ij} = 1 \end{cases} \tag{III.3}$$

for $i \in \{1,\ldots,n_X\}$, $j \in \{1,\ldots,n_x\}$ and where $\mathbf{T}$ is a boolean matrix of sizes: $n_X \times n_x$ which

is formed by following the algorithm 16. In this algorithm, η is an integer parameter which control the amount of mixing between $\mathbf{X}_P$ and $\mathbf{X}_Q$. It must be defined by the user such that $0 < \eta \leq n_x$. In addition, a random permuting is first performed on samples (lines) of $\mathbf{X}_P$ before applying relation (III.3).

Algorithm 16 Algorithm for the generation of the T matrix used in the BSA crossover

Initialize $\mathbf{T} := \mathbf{0}$ and $a := \mathscr{U}(0,\ 1)$;
if $a = 0$ **then**
  for $i = 1 : n_X$ **do**
    $u := Permuting(1 : n_x)$;
    $b := \mathscr{U}(0,\ \eta)$;
    for $k = 1 : b$ **do**
      $j = u_k$;
      $T_{ij} = 1$;
    end for
  end for
else
  for $i = 1 : n_X$ **do**
    $j := \mathscr{U}(0,\ n_x)$;
    $T_{ij} = 1$;
  end for
end if

Recombination propositions for an hybrid algorithm

To obtain a more efficient immune algorithm, alternatives for the recombination strategy are of interest. The proposal is an hybridization, consisting in exchanging the crossover operator used for the recombination in NNIA with a novel BSA inspired recombination operator.

In NNIA, recombination is achieved by involving SBX between the clonal population and the active population. But it is important to note that both these populations have different sizes, since it is the main objective of the clonal operator to expand the active population. In contrast, crossover of BSA appears to be very simple and efficient. However it concerns two populations of equal sizes. Hence, due to the unequal size of the clonal population and the active population in NNIA, it is not possible to directly embed the BSA crossover operator for the recombination in NNIA to obtain an hybrid algorithm. To alleviate this incompatibility, two ideas emerge:

1. The first idea consists in expanding the active population in order to obtain an extended active population, having its size equal to the clonal population size. The simplest way to achieve this consists in duplicating the active population;

2. The second idea consists in replacing the active population for the crossover by the clonal population, leading finally to a crossover which uses only the clonal population.

From our experiments on some tests problems, it appears that both these alternative strategies lead to a better performance for the convergency and spacing of the Pareto front. However, results can be worse for the spread of the Pareto front. This observation leads us to a new idea: it is desirable to introduce a degree of mutation in the recombination of the hybrid algorithm. As a consequence, the third proposal consists in mixing duplication of the active population with new, random, individuals. This leads to introduce an additional parameter for the resulting algorithm, being the amount of random individuals used in this third recombination strategy.

Experiments

In this section, we study the efficiency of the hybridization when solving some well-known MO problems including five ZDT problems [9] and five DTLZ problems [1]. These benchmark problems have already been used to test the effectiveness of NNIA in [57] and empirical comparisons have shown that NNIA was able to quite converge to the true Pareto-optimal fronts of these problems, even for the complicated tests problems DTLZ1 and DTLZ3.

For NNIA the number of points along the Pareto front and the number of iterations are fixed by the user. But it is expected that the obtained solutions points are accurate, well-distributed and widely spread. In this work, performance metric indicators are used to compare algorithms. They are the normalized Inverted generational distance (NIDG) metric and the normalized spacing (NSP). NIGD is used to measure the convergency and spreading while NSP is used to measure the distribution diversity.

Performance metrics

Approximate Pareto front solution of MO algorithms must achieve these two goals:

1. Convergence towards the true Pareto front; and

2. Diversity of solutions: the Pareto front must be uniformly distributed and spread over the entire feasible objective space to adequately capture the trade-offs. For benchmarks tests problems, the true Pareto front is known, allowing to exploit performance metrics which used it.

We opted for two performance metrics for assessing algorithms efficiency. To measure the extent of the convergence to the true set of Pareto-optimal solutions and the spread

of the Pareto front set, a normalized version of the Inverted generational distance (IDG) metric proposed by [81] is adopted, while a normalized version of the spacing metric introduced by [**?**] enables to measure the uniformity of the obtained solutions.

Normalized inverted generational distance

The normalized inverted generational distance (NIGD) is based on a proposition of [81]. It is a measure of the distance between the true Pareto front F^*, which is known at n^* discrete values - and stored in the matrix $\mathbf{F}(\mathbf{X}^*)$ - and approximate solutions of the Pareto-optimal front $\mathbf{F}\left(\widehat{\mathbf{X}}\right)$:

$$\mathrm{NIGD}\left(\mathbf{F}(\mathbf{X}^*), \mathbf{F}\left(\widehat{\mathbf{X}}\right)\right) = \frac{1}{n^*}\sqrt{\sum_{j=1}^{n^*}\mathbf{c}_j^2}, \tag{III.4}$$

for:

$$\mathbf{c}_j = \min_{i \in \{1,\ldots,n_X\}} \left(\sqrt{\sum_{k=1}^{m}\left(\overline{\mathbf{F}}_k(\mathbf{X}_i^*) - \overline{\mathbf{F}}_k\left(\widehat{\mathbf{X}}_j\right)\right)^2}\right), \qquad j \in \{1,\ldots,n^*\}$$

where $\overline{\bullet}$ denotes a normalized objective function, ranging from 0 to 1 and defined by:

$$\overline{\mathbf{F}}_k(\mathbf{x}) = \frac{\mathbf{F}_k(\mathbf{x}) - \min\left(\mathbf{F}_k(\mathbf{X}^*)\right)}{\max\left(\mathbf{F}_k(\mathbf{X}^*)\right) - \min\left(\mathbf{F}_k(\mathbf{X}^*)\right)}. \tag{III.5}$$

Hence, the distance is calculated for every true solution with respect to its nearest obtained Pareto optimal solution in the objective space. This implies that this measure is sensitive to the spread of the Pareto front found by the MO procedure. Hence, both diversity and convergence of the approximated set $\mathbf{F}\left(\widehat{\mathbf{X}}\right)$ could be measured using IGD. To obtain smaller values of this measure, the approximated set $\mathbf{F}\left(\widehat{\mathbf{X}}\right)$ must be very close to the Pareto front and cannot miss any part of the whole Pareto front at the same time.

Normalized spacing measure

The spacing metric introduced by [59] aims at measuring how uniformly the obtained solutions are distributed by calculating a relative distance between the nearest solutions to each other. However, in practice, to reduce bias when the orders of magnitudes of the objectives differ considerably, the measure introduced by [59] is modified by taking

normalized objectives functions. This leads to the normalized NSP measure, defined by:

$$\mathrm{NSP}\left(\mathbf{F}\left(\widehat{\mathbf{X}}\right)\right) = \sqrt{\frac{1}{n_X - 1}\sum_{j=1}^{n_X}\left(\mathbf{d}_j - \overline{d}\right)^2} \tag{III.6}$$

for:

$$\mathbf{d}_j = \min_{\substack{i \in \{1,\ldots,n_X\} \\ i \neq j}} \left(\sqrt{\sum_{k=1}^{m}\left(\overline{\mathbf{F}}_k\left(\widehat{\mathbf{X}}_i\right) - \overline{\mathbf{F}}_k\left(\widehat{\mathbf{X}}_j\right)\right)^2}\right), \qquad j \in \{1,\ldots,n_X\}$$

where $\overline{d}$ is the mean of $\mathbf{d}$. When the obtained solutions are more evenly distributed, the NSP measure becomes smaller. Therefore, smaller NSP value indicates a preferable solution. A null value suggests that the Pareto optimal solution set obtained by the algorithm are evenly distributed. But we have to note that NSP does not assess for the spread of the Pareto optimal solution.

Empirical comparison

In this section, performance of five NNIA variant are evaluated. The five variant are:

1. NNIA-X: the NNIA algorithm without crossover;
2. NNIA: the algorithm proposed by [57];
3. NNIA+X1: the hybridization of the NNIA algorithm with the BSA crossover by using the first strategy proposed in the section 4. Inputs of the BSA crossover function are 1) the clonal population and 2) a random permutation of an extended active population obtained by duplicating individuals;
4. NNIA+X2: the hybridization of the NNIA algorithm with the BSA crossover by using the second strategy proposed in the section 4. Inputs of the BSA crossover function are 1) the clonal population and 2) a random permutation of the clonal population;
5. NNIA+X3: the hybridization of the NNIA algorithm with the BSA crossover by using the third strategy proposed in the section 4. Inputs of the BSA crossover function are 1) the clonal population and 2) a random permutation of an extended active population obtained by duplicating individuals with a proportion of random individuals.

For NNIA, parameters proposed in reference [57] are set:

- maximum size of dominant population: $n_{\hat{X}\max} = 100$,
- maximum size of active population: $n_{A\max} = 20$, and
- size of clone population $n_C = 100$,

with the distribution index for SBX that is 15, the distribution index for polynomial mutation that is 20 and the mutation probability of $1/n_x$. These parameters are remains fixed for all experiments, with a number of iterations stopped at 250. In addition, for NNIA+X3, the proportion of random individuals is chosen to be equal to n_A and their distribution is uniform.

Figures III.1, III.2, III.3 and III.4 show the statistic box plots [1] for NIGD and NSP obtained for 1,000 independent runs performed on each test problems ZDT and DTLZ that are chosen by [57][2].

[1]The boxplot shows the median (50th percentile) of the sample data using a horizontal red line. Notched box plots indicate limits of the 95Edges of the rectangle exhibit the 25th and 75th percentile of the observations. When end data are shown using +, the ends of the whiskers displayed by two horizontal black lines are calculated using 1.5 times the interquartile space (the height of the rectangle) and data points beyond the whiskers (the outliers) are displayed using +. However, when there is no data shown using +, horizontal black line are given by the extrema of the data.

[2]From informations given in [1], it is believed that the problem denoted DTLZ6 in [57] is in fact the DTLZ7 problem.

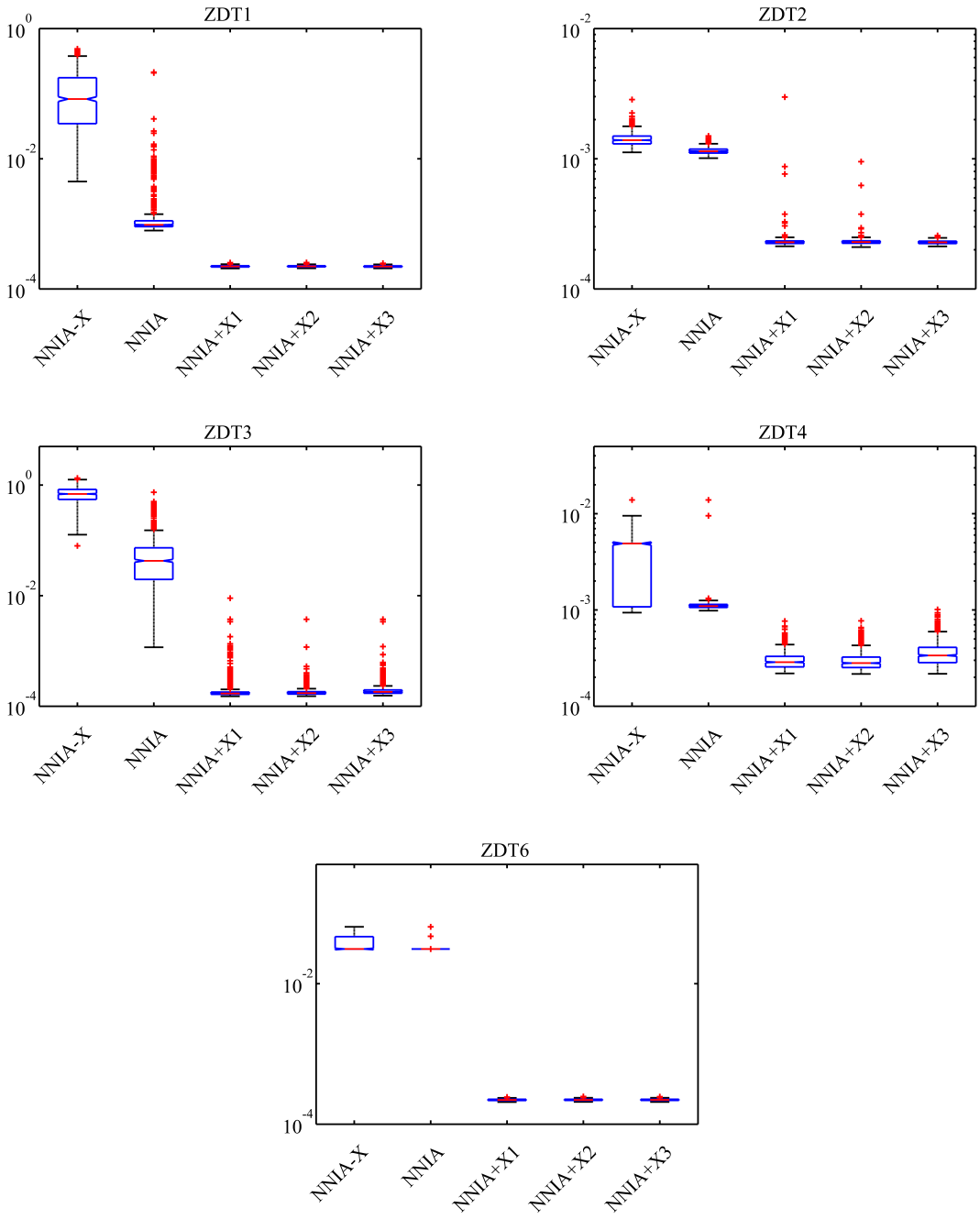

Figure III.1 – Statistics box plots of NIGD obtained from 1000 independent runs of benchmark tests problems ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6.

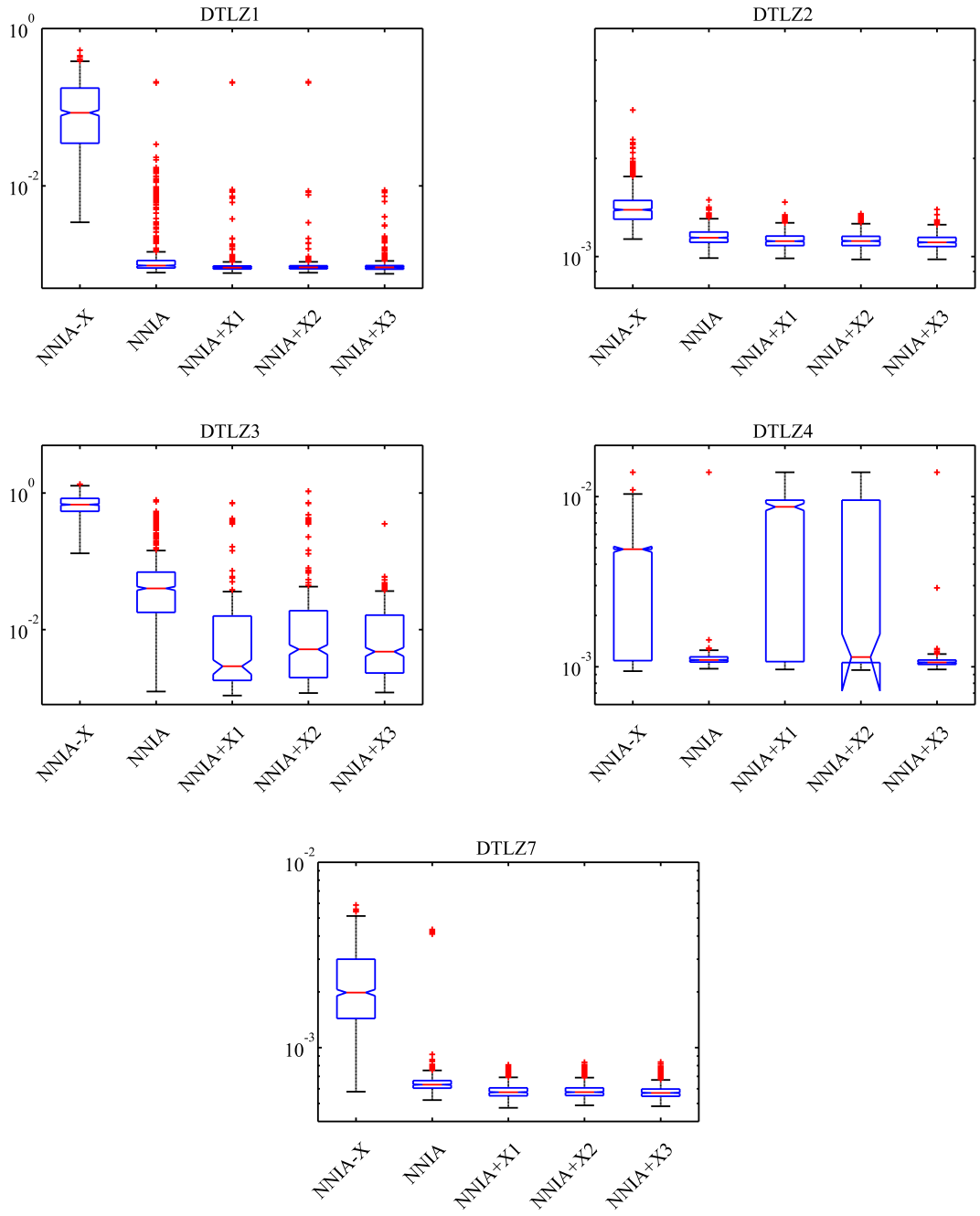

Figure III.2 – Statistics box plots of NIGD obtained from 1000 independent runs of benchmark tests problems DTLZ1, DTLZ2, DTLZ3, DTLZ4 and DTL7.

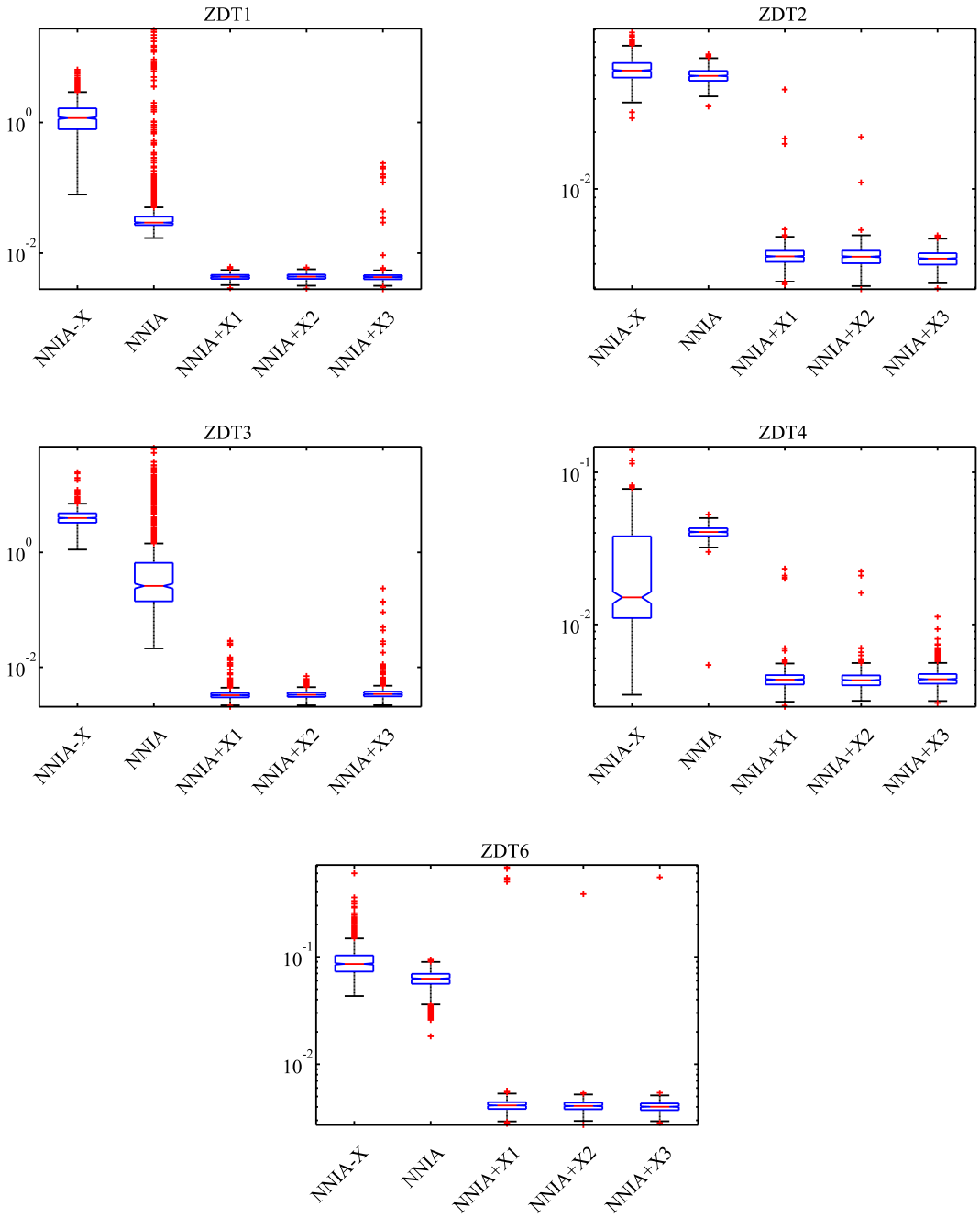

Figure III.3 – Statistics box plots of NSP obtained from 1000 independent runs of benchmark tests problems ZDT1, ZDT2, ZDT3, ZDT4, ZDT6.

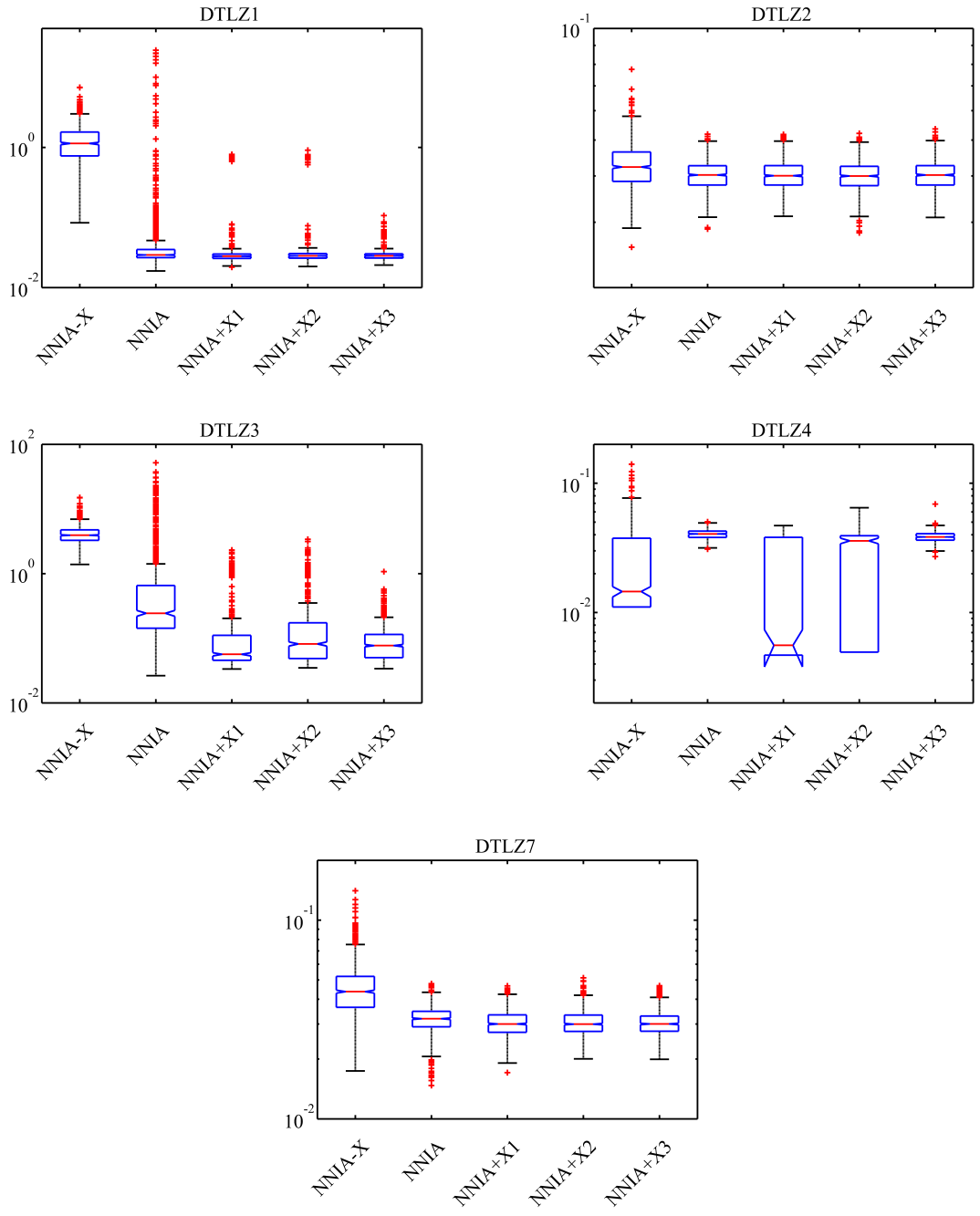

Figure III.4 – Statistics box plots of NSP obtained from 1000 independent runs of benchmark tests problems DTLZ1, DTLZ2, DTLZ3, DTLZ4 and DTL7.

Analysis of NIGD statistic results show the superiority of NNIA over NNIA-X for the treated tests problems. The superiority of NNIA+X1 and NNIA+X2 over NNIA is also observed except for the difficult test problem DTLZ4. For ZDT4, the NNIA+X3 is found to be inferior to NNIA+X1 and NNIA+X2. But NNIA+X3 is found always superior to NNIA for the treated tests problems. Analysis of NSP show the superiority of

NNIA-X	NNIA	NNIA+X1	NNIA+X2	NNIA+X3
2%	11%	13%	11%	0.2%

Tableau III.1 – Percentage of results exhibiting a single point for the Pareto front of the DTLZ4 test problem when 1,000 runs are carrying out.

NNIA over NNIA-X except for ZDT4 et DTLZ4. NNIA+X1, NNIA+X2 and NNIA+X3 appear to be approximately equal or superior to NNIA in all treated cases. But the DTLZ4 problem appear to be the most difficult for all these algorithms, since there exist some runs for which the Pareto front is approximated by only one, single, point. The NSP for this situation is meaningless. Thus, statistics of NSP are not evaluated from the same number of runs between the five algorithms. The Table III.1 show the percentage of results exhibiting a single point for the Pareto front of the DTLZ4 test problem when a sequence of 1,000 runs is carrying out with each algorithm.

Overall, we arrive to the conclusion that NNIA+X3 preserves better population diversity and converges faster than NNIA for these bi-objective and three-objectives tests problems.

Experiments on the ten-bar truss design problem

In this section, we address the multi-objective sizing and topology optimization of truss-like structures which is a continuous subject of researches in structural design [82–84]. In the past, a number of studies have been published where structural optimization problems with multi-objectives are solved using meta-heuristics approach [32].

Problem formulation

Let the design domain comprises a set of nodes with fixed spatial coordinates, a set of supports and a set of loads, as it is sketch in the Figure III.5 for the ten-bar truss test. It is assumed that the structure will be modeled by linear, two nodes, bar elements in linear elasticity, subjected only to axial forces and free from imperfections. In this study, two objective functions have to be minimized: the mass and the displacement; and one objective function has to be maximized: the first flexible natural frequency of the structure.

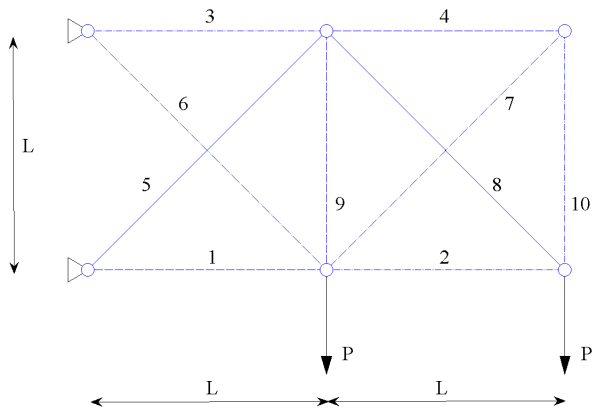

Figure III.5 – Sketch of the ten-bar truss.

Denoting $\mathbf{x} \in \Omega$ the vector of the topological and sizing optimization parameters, such that:

$$0 \leq x_i \leq 1 \quad \text{for} \quad i \in \{1, \ldots, n\}$$

where $n = 10$ is the number of elements, the three individual objectives are:

1. The mass w of the structure:

$$w(\mathbf{x}) = \sum_{i=1}^{n} \rho A l_i x_i,$$

where l_i is the length of the i-th element, $\rho = 2,768$ kg/m^3 is the density of the material and $A = 0.01419352$ m^2 is the element cross-section area.

2. The maximum displacement u of the structure:

$$u(\mathbf{x}) = \max\left(\mathbf{u}^* = \arg\min_{\mathcal{S}}\left(\frac{1}{2}\mathbf{u}^{\mathrm{T}}\mathbf{K}(\mathbf{x})\,\mathbf{u} - \mathbf{u}^{\mathrm{T}}\mathbf{F}\right)\right),$$

where $\mathbf{F}$ is the vector of loads and $\mathbf{K}$ is the stiffness matrix of the finite element (FE) model, having the Young' modulus $E = 68.95$ GPa. The set $\mathcal{S}$ refers to the kinematic admissible space, *i.e.* the one that satisfies the imposed boundary conditions given by the supports while carrying all the prescribed loads, where $P = 448.2$ kN.

3. The opposite of the minimum flexible natural frequency f of the structure, in order to maximize it:

$$-f(\mathbf{x}) = -\min\left(\frac{1}{2\pi}\omega^*\right),$$

$$\text{where}: \left\{\omega^{*2}, \mathbf{u}^*\right\} = \arg\min_{\mathbf{u}\in\mathcal{S}}\left(\omega^2 = \frac{\mathbf{u}^{\mathrm{T}}\mathbf{K}(\mathbf{x})\,\mathbf{u}}{\mathbf{u}^{\mathrm{T}}\mathbf{M}(\mathbf{x})\,\mathbf{u}}\right), \quad \|\mathbf{u}\| \neq 0$$

where $\mathbf{M}$ is the mass matrix of the FE model[3].

Moreover, this MO problem is subjected to constraints for the mechanical stress σ_i for each element i:

$$|\boldsymbol{\sigma}_i(\mathbf{x})| \leq \overline{\sigma} \qquad i \in \{1, \ldots, n\}$$

where $\overline{\sigma} = 172.4$ MPa is the yield strength.

Then, thanks to the introduced formulation and the domain of definition for $\mathbf{x}$, the sizing and the topology of the structure are optimized concurrently. As a consequence, local rigid body modes or kinematic modes may appears in some cases when some design variables are set to zero. For example, when $x_4 = x_7 = 0$, the element 10 is free to rotate. In practice, concerned elements -such as the element 10 in this example- are detected since their stress approach zero. As designs with local rigid body modes or kinematic modes are not of interest, constraints are added to the MO problem formulation:

$$\frac{|\boldsymbol{\sigma}_i(\mathbf{x})|}{\overline{\sigma}} > \epsilon, \qquad i \in \{1, \ldots, n\} \text{ such that } x_i > 0$$

where $\epsilon = 0.001$.

Since the optimal Pareto front is unknown for this problem, unnormalized metric indicators are to assess for the MO algorithm performance. Thus, in practice, we introduce an *a priori* scaling of the three objectives, by defining:

$$f_1(\mathbf{x}) = \frac{w(\mathbf{x})}{7,000}, \qquad f_2(\mathbf{x}) = \frac{u(\mathbf{x}) - 0.016}{20}, \qquad f_3(\mathbf{x}) = \frac{22,500 - (2\pi f(\mathbf{x}))^2}{5,000}$$

Moreover, in order to handle constraints of this MO problem, we use the penalty method. This technique consists in replacing the constrained optimization problems by an optimization problems without constraints, when introducing new objective functions to be optimized:

$$\phi_k(\mathbf{x}) = f_k(\mathbf{x}) + r\varphi(\mathbf{x}) \tag{III.7}$$

where the penalty function chosen here is :

$$\varphi(\mathbf{x}) = \sum_{i=1}^{n} \left(\max\{0, \frac{|\boldsymbol{\sigma}_i(\mathbf{x})|}{\overline{\sigma}} - 1\} \right)^2 + \sum_{i=1}^{n} \left(\max\{0, \epsilon - \frac{|\boldsymbol{\sigma}_i(\mathbf{x})|}{\overline{\sigma}}\} \right)^2 \tag{III.8}$$

and where r is a positive penalty parameter. The problem is then solved directly for a value of r large enough, so the constraints are satisfied. We have chosen here $r = 10^{10}$.

[3]To obtain the best numerical efficiency for the FE analysis, the FE disassembly strategy proposed in reference [85] is involved.

Finally, the MO problem definition for the ten bar truss of this work is:

$$\min_{\mathbf{x}\in\Omega}\left(f_1(\mathbf{x})+r\varphi(\mathbf{x}),\, f_2(\mathbf{x})+r\varphi(\mathbf{x}),\, f_3(\mathbf{x})+r\varphi(\mathbf{x})\right)$$

Numerical simulations for two objectives functions

In this subsection, the ten-bar MO problem of the previous section is subdivided and changed into three more simple MO subproblems having only two objective functions by considering them two-by-two: (w, u), (w, f) and (u, f). The NNIA and NNIA+X3 are used to solve each of these three ten-bar MO problems. Algorithms parameters are kept identical to the ones of the previous subsection 5.2.

Figures III.6 and III.7 show typical Pareto fronts obtained after 250 and 750 iterations (respectively) of one typical run, when both algorithms start from the same initial guess population. In these figures, a better diversity is observed for the NNIA+X3 for each subproblem, and a better convergence is observed for the NNIA+X3 for the (w, f) and (u, f) subproblems. Since each iteration of one of these algorithms requires $n_c = 100$ evaluations of the mechanical problem, 25,000 functions evaluations are achieved when 250 iterations are carried out and 75,000 functions evaluations are achieved when 750 iterations are carried out.

Figure III.8 shows the evolution of two metrics indicators along the number of iterations for one typical run. Metric indicators chosen here are spacing and hyper-volume of Pareto fronts. Spacing evolution is presented in log-log scale in the figure. Each evaluation of the hyper-volume is achieved by using the same anti-utopia point and utopia point for results consistency. Moreover, in order to compare the three MO results on the same graph, a relative hyper-volume is plotted: the graph corresponds to the hyper-volume obtained divided by its maximum value. These graphs show a better diversity and convergence for NNIA+X3 compared to NNIA when early number of iterations are considered.

Tendency observed in the previous figure is confirmed by statistical results of Figure III.9 and Figure III.10. These figures show box plots statistic for spacing and hyper-volume (respectively) when 300 runs stopped at 250 iterations are carried out. For spacing, means and variance are clearly better for NNIA+X3. Hyper-volume statistic results are also better for the NNIA+X3 when considering the (w, u) and (u, f) subproblems, while they are almost identical for the (w, f) subproblem, although the mean and variance are also better for the NNIA+X3. From results for the hyper-volume of the (w, f) subproblem, it is assessed that this subproblem is the most difficult to solve since a wide spread is observed in data for both algorithms.

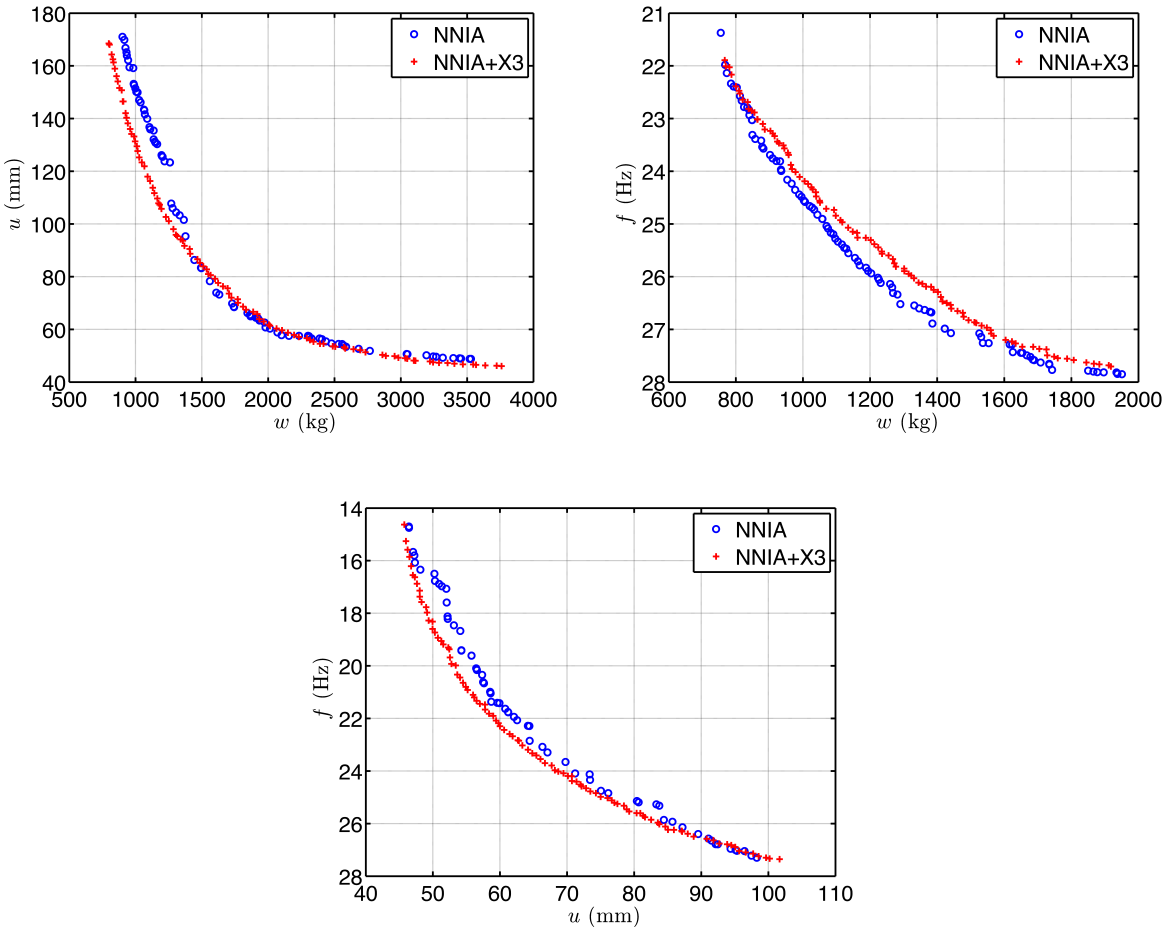

Figure III.6 – Pareto fronts of the ten-bar truss MO problem at iteration number 250 for one typical run when optimizing the three objectives functions two-by-two: (w, u) (up-left), (w, f) (up-right), (u, f) (down).

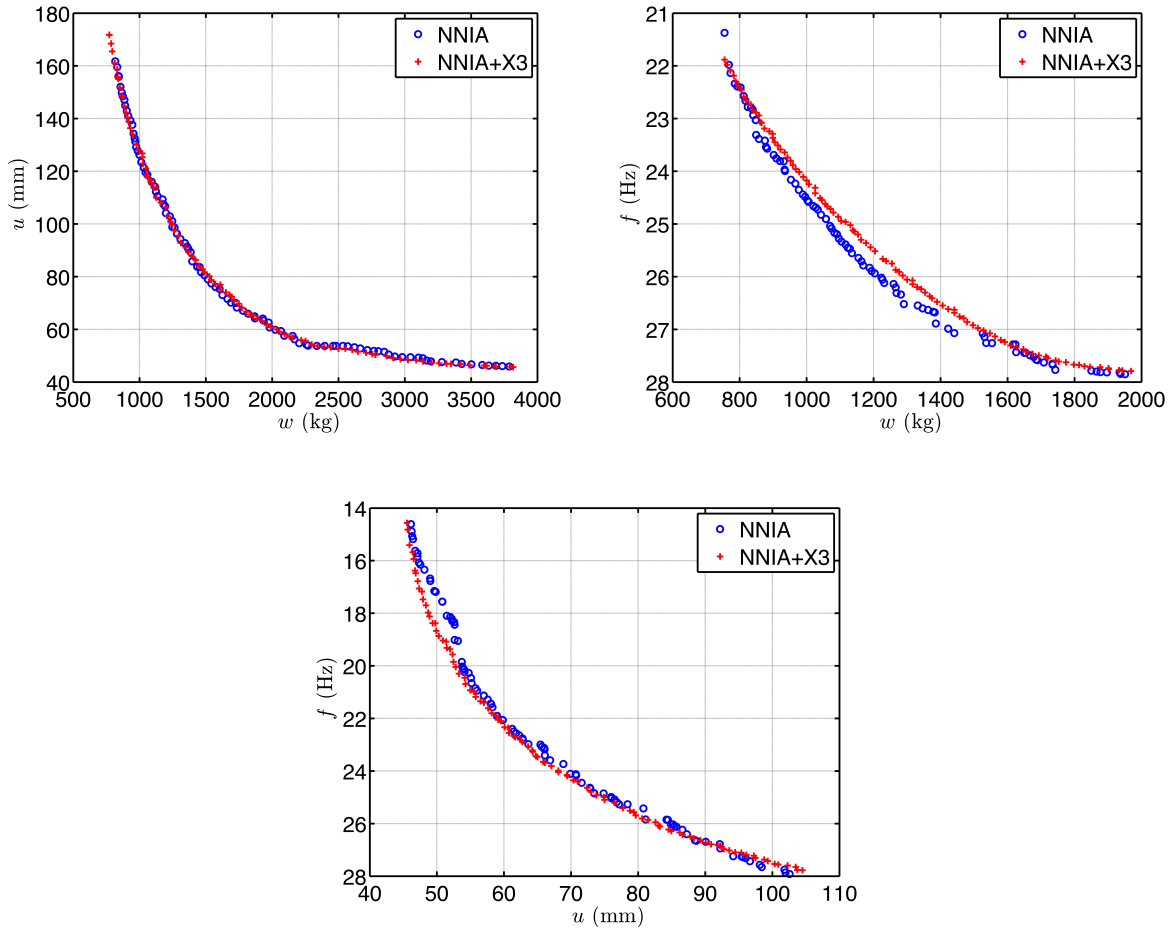

Figure III.7 – Pareto fronts of the ten-bar truss MO problem at iteration number 750 for one typical run when optimizing the three objectives functions two-by-two: (w, u) (up-left), (w, f) (up-right), (u, f) (down).

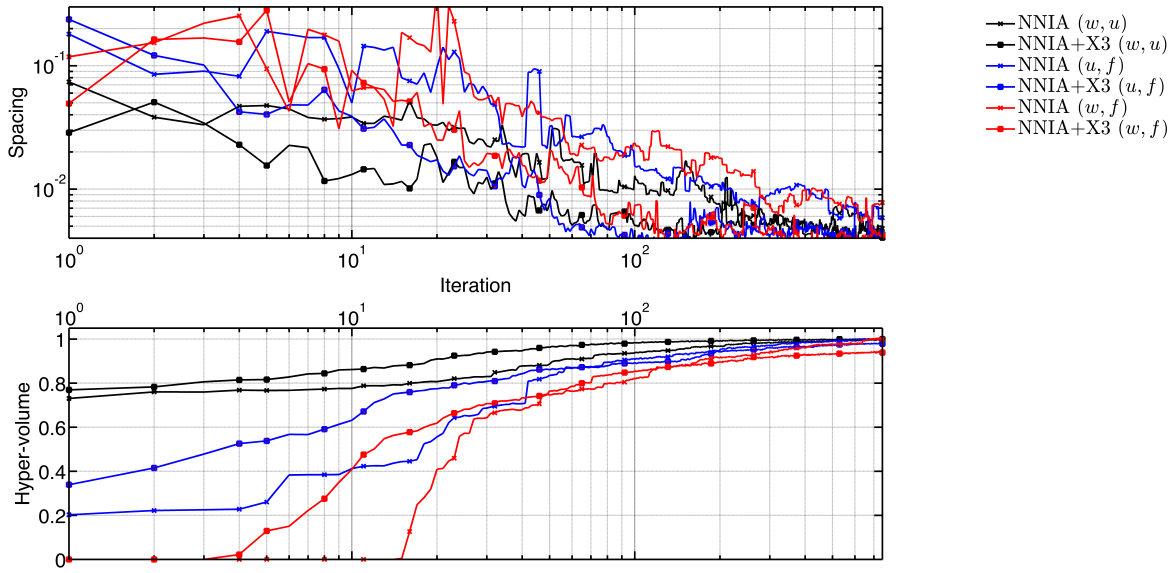

Figure III.8 – Metrics indicators of the ten-bar truss MO problem for one typical run when optimizing the three objectives functions two-by-two: spacing (up) and relative hyper-volume (down).

Numerical simulation for three objectives functions

Figure III.11 shows different views of the Pareto front obtained for one typical run when solving the three objectives ten-bar truss problem, using NNIA+X3 with the following parameter values: size of active population 30, clonal scale 150 and 750 iterations. In this case, the size of the dominant population is not limited to any number and all Pareto points found are kept. Figure III.12 shows the evolution of the number of points in the dominant population for the Pareto front given in Figure III.11. It ends at 2216 Pareto points for this run.

Figure III.13 show box plots statistics when 300 runs are carried out with NNIA and NNIA+X3. It is observed that the number of points for the Pareto front is higher for NNIA+X3, with a better spacing. But the hyper-volume is better for NNIA. Detailed analysis of results has revealed that bad results for hyper-volume are due to a slow convergence to an extreme Pareto front point: the individual optima for the frequency objective. For this problem, the individual minima found for the frequency objective is most of the time better for NNIA than for NNIA+X3. However, it is also found that individual minima of the three objectives are rarely found in the Pareto front by both algorithms.

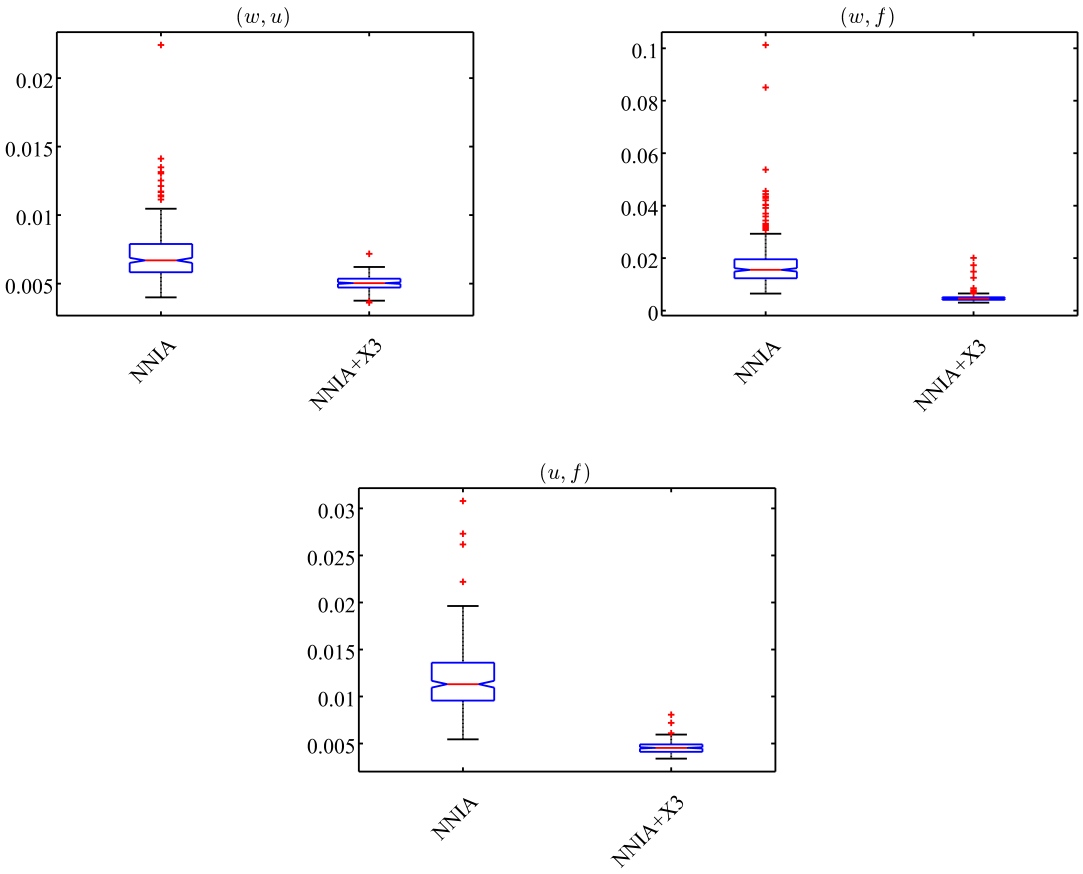

Figure III.9 – Statistics box plots of spacing for 300 runs of the two-by-two MO ten-bar subproblems: (w, u) (left), (w, f) (middle), (u, f) (right).

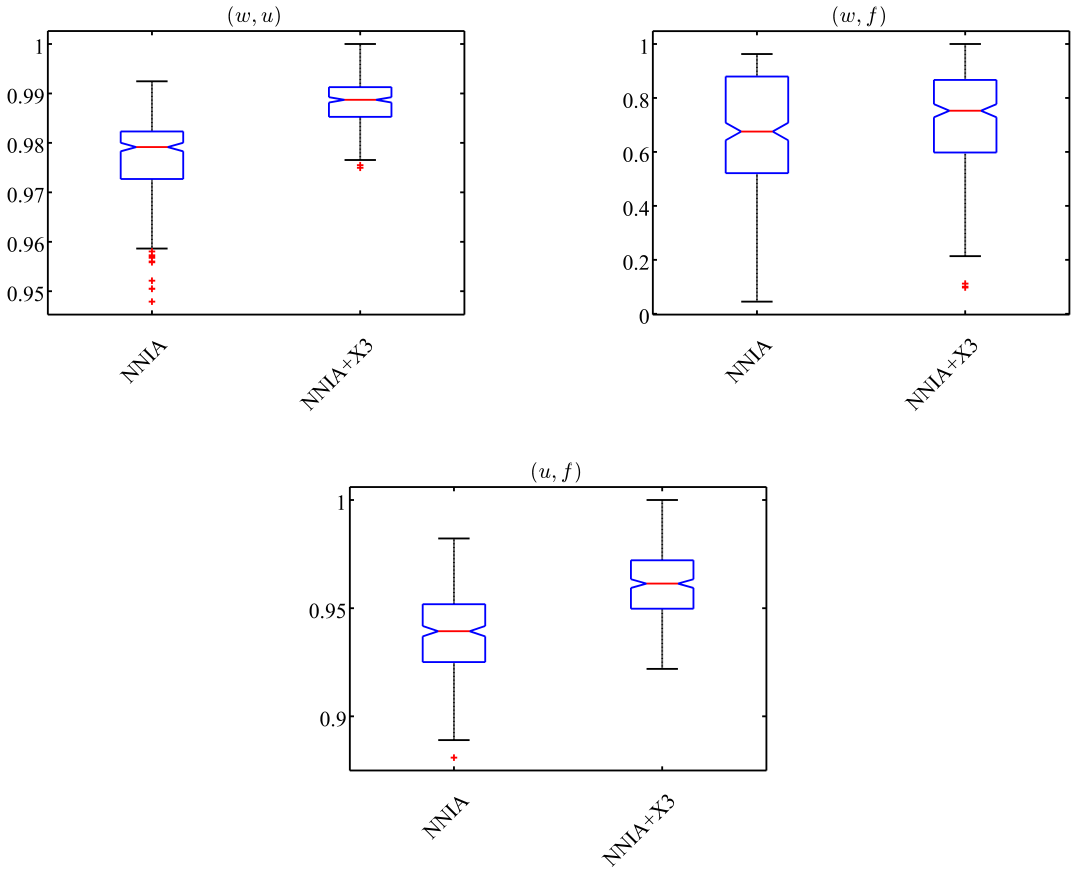

Figure III.10 – Statistics box plots of relative hyper-volume for 300 runs of the two-by-two MO ten-bar subproblems: (w, u) (left), (w, f) (middle), (u, f) (right).

For better results, the idea is to handle the three individual minima found by a mono-objective optimization into the random initial population of both NNIA and NNIA+X3. This simple modification greatly improve performance results. Figure III.14 show statistics box plots when 300 runs of MO problem are carried out when the three individual optima are given in the initial population. In such a situation and for each of the 300 runs done, NNIA+X3 appears to be superior to NNIA for all performance aspects, including the computed hyper-volume.

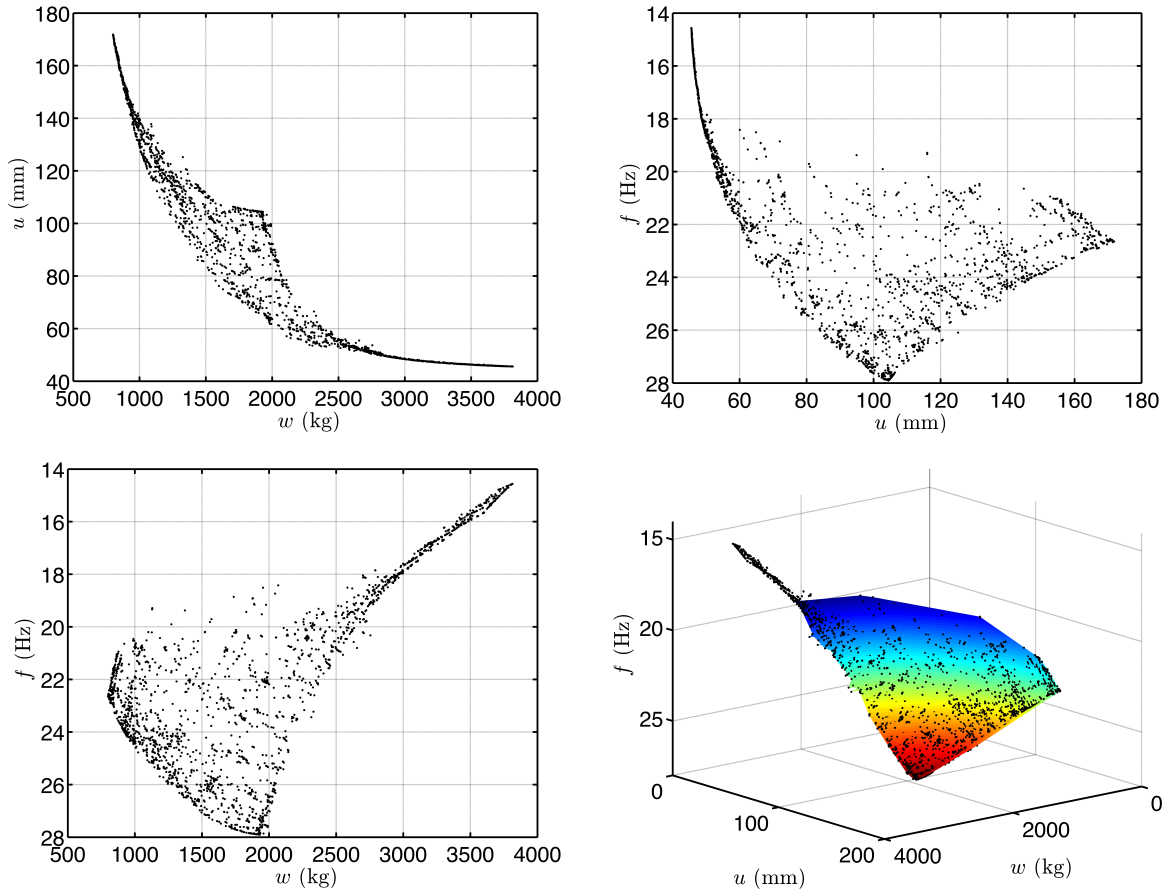

Figure III.11 – Four different views of the Pareto front obtained for one typical run when solving the three objectives functions of the ten-bar truss problem; Colorized surface of the down-right sub-figure is added for a better visualization and the color corresponds to the frequency objective f.

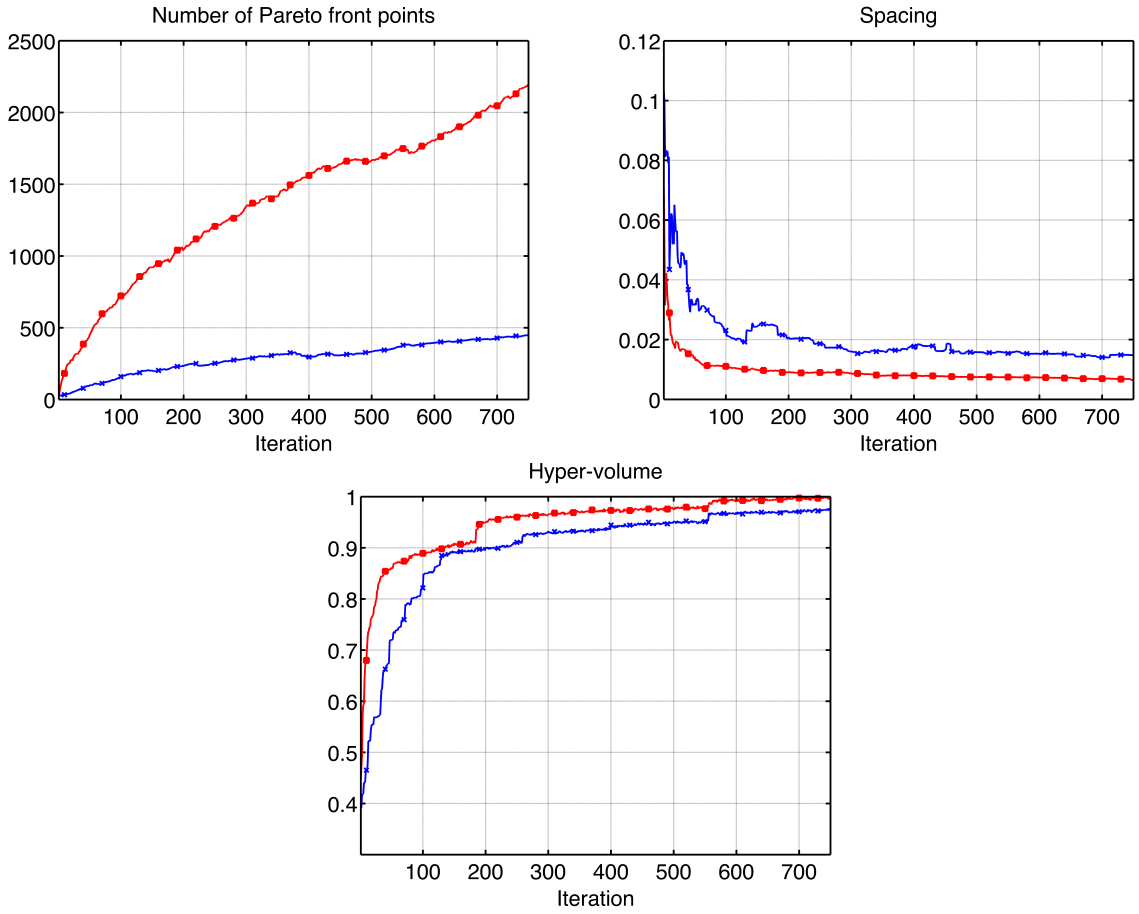

Figure III.12 – Evolution of results for a typical run of the MO ten-bar problem: Number of points for the Pareto front (left), spacing (middle) and relative hyper-volume (right); Blue line with cross markers: NNIA; Red line with squared markers: NNIA+X3.

Conclusion

This work focuses on the recombination step for an Artificial Immune Algorithm. NNIA uses SBX for this operation, by using antibodies of the clonal population and antibodies of the active population. We propose here new recombination strategies, by using the BSA crossover operator when adapting input populations.

For the first recombination strategy, NNIA+X1, the clonal population and an extended active population are concerned, where the extended active population is obtained by duplicating individual antibodies. In the second strategy, NNIA+X2, recombination is achieved by using the clonal population and itself. The third strategy is NNIA+X3, which uses the clonal population and an extended active population obtained by duplicating individual antibodies and a proportion of random individuals. By this way, some

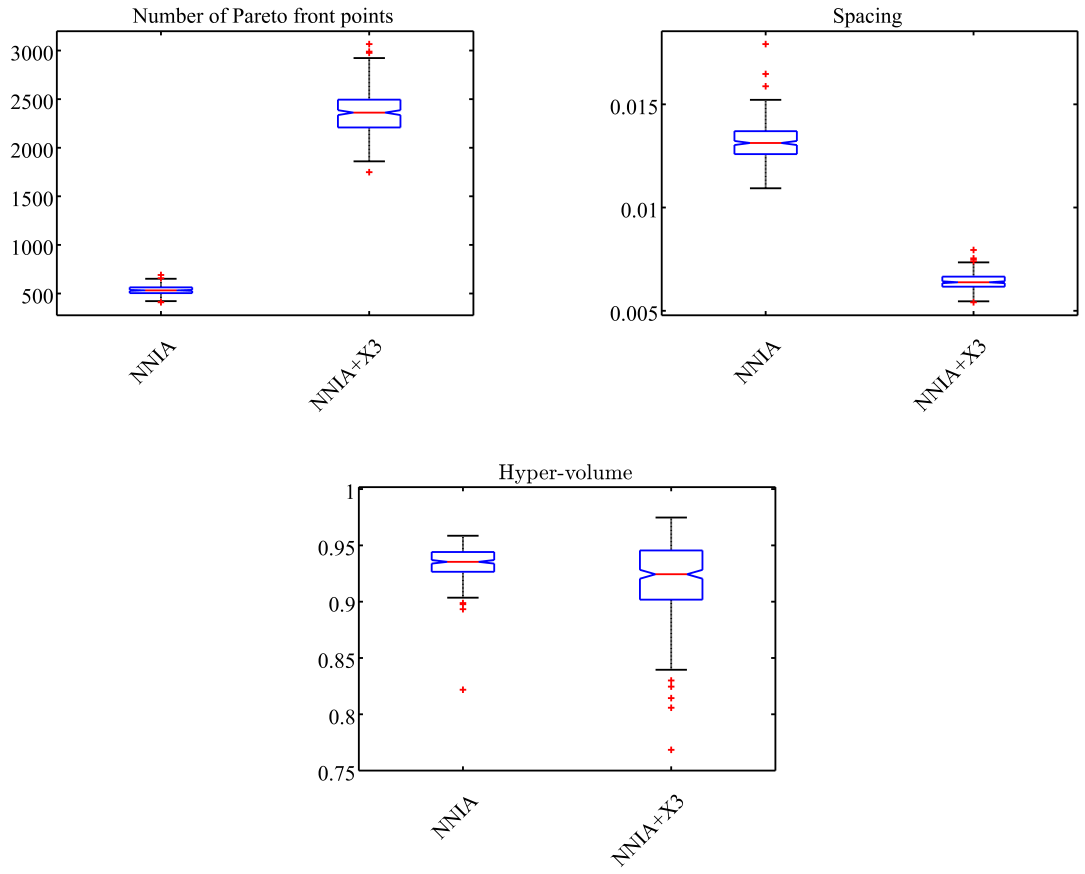

Figure III.13 – Statistics box plots for 300 runs of the three objectives ten-bar problem with NNIA and NNIA+X3 with random initial population: Number of Pareto front points (left), spacing (middle) and relative hyper-volume (right).

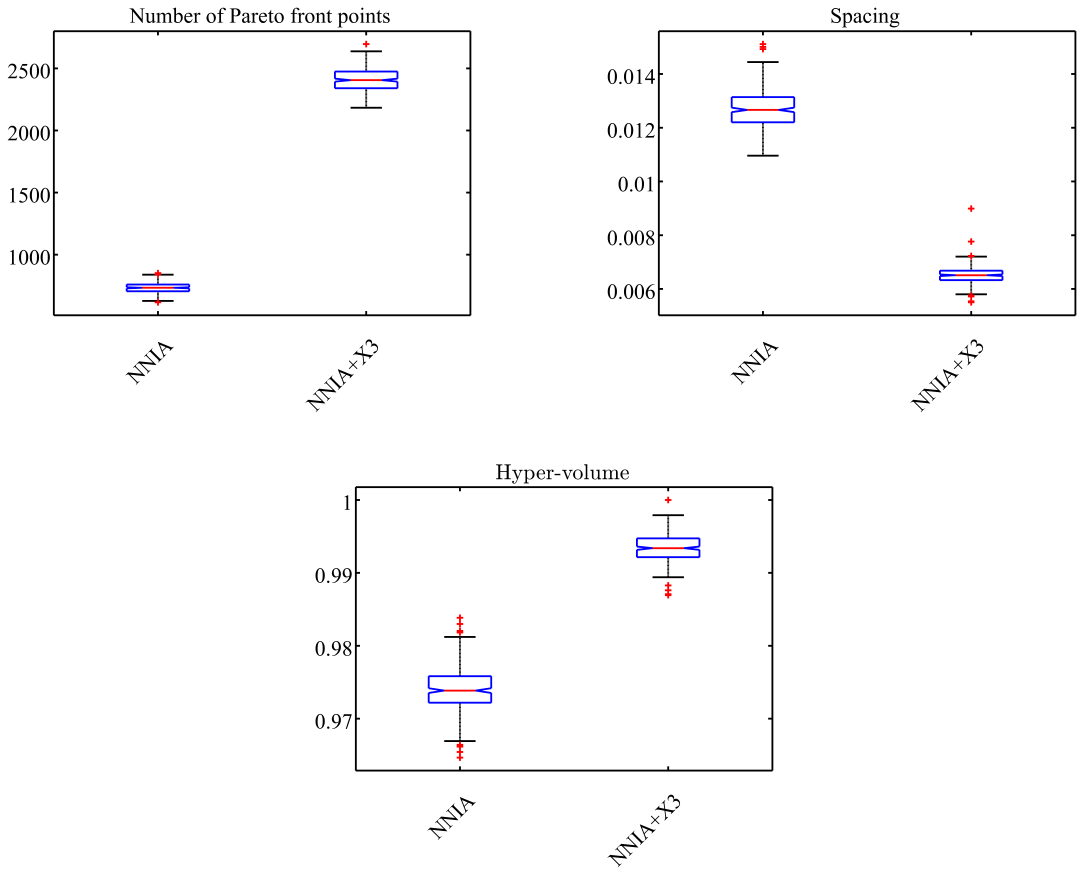

Figure III.14 – Statistics box plots for 300 runs of the three objectives ten-bar problem with NNIA and NNIA+X3 when individual optima are handled in the initial population: Number of Pareto front points (left), spacing (middle) and relative hyper-volume (right).

degree of mutation is introduced in the algorithm.

Results obtained for test problems ZDT and DTLZ of multi-objective optimization concluded that NNIA+X3 can obviously not only accelerate the convergence speed, but also keep the desirable diversity, especially when solving problems with many local Pareto-optimal fronts. Then, we applied this algorithm to solve multi-objective topology optimization of the ten bar truss structure. Experimental results indicate that the proposed NNIA+X3 outperforms NNIA in terms of both convergence rate and solution quality, both for the bi-objective and for three objective problems of the ten bar truss structure.

As NNIA+X does not produce a good approximation of the Pareto Front of the 14-bar problem after 250 iterations.

We are interested in developing a new method for the resolution of MO problem with constraints and the 14-bars problem. This method will be described in the next chapter.

Chapter IV

Multi-objective Optimization Backtracking Search Algorithm

Contents

Based on: **A. Tchvagha Zeine, A. El Hami R. Ellaia and E. Pagnacco**, *Backtracking Search Algorithm for Multi-objective Design Optimization*, International Journal of Mathematical Modelling and Numerical Optimisation, Vol. 8, No. 2, 2017.

Introduction

Most of the real world engineering problems can be formulated as multi-objective optimization (MO) problems. They are characterized by a set of objective functions and by certain constraints to be satisfied. Their solutions are given by a set of non-dominated solutions known as Pareto front. The Pareto optimal set is thus the set of solutions representing the best trade-offs between objectives. A comprehensive survey of the MO can be found in [86].

Over the course of past decade, a significant number of multi-objective algorithms has been developed. Between the population-based algorithms, which is the focus of this work, the most well-regarded ones are: Multi-Objective Genetic Algorithm (MOGA) [50], Non-dominated Sorting Genetic Algorithm (NSGA) [7, 53], Niched Pareto Genetic Algorithm [51], Strength Pareto Evolutionary Algorithm (SPEA) [8, 60], Pareto-Archived Evolution Strategy (PAES) [56] and Multiple Objectives with Particle Swarm Optimization (MOPSO) [87].

In recent years, different types of MO algorithms have been proposed to solve problems with multiple objectives. For example, Generalized Differential Evolution 3 (GDE3) [88], Archive-based Micro Genetic Algorithm (AMGA) [89], Local Search Based Evolutionary Multi-Objective Optimization Algorithm (NSGAIILS) [90], Differential Evolution with Self-adaptation and Local Search Algorithm (DECMOSA-SQP) [91], Multi-Objective Self-adaptive Differential Evolution Algorithm with Objective-wise Learning Strategies (OWMOSaDE) [92], Dynamical Multi-Objective Evolutionary Algorithm (DMOEADD) [89], Multi-Objective Evolutionary Programming (MOEP) [93], Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) [94], Multiple Trajectoire Search (MTS) [95], LiuLi Algorithm [96], Clustering Multi-Objective Evolutionary Algorithm (Clustering MOEA) [97], Enhancing MOEA/D with Guided Mutation and Priority Update (MOEADGM) [98] and an Orthogonal Multi-objective Evolutionary Algorithm with Lower-dimensional Crossover (OMOEAII) [99] are some of the competitive evolutionary MO algorithms that aimed to obtain approximate Pareto front for multi-objective problems. These algorithms are compared in proceeding of "2009 IEEE Congress on Evolutionary Computation" for solving MO problems CEC [54]. From results obtained, MOEA/D appears to be one of the most competitive for solving MO problems. It converts a MO problem into a set of simple single-objective sub-problems. This approach is based on the decomposition methods in mathematics and the optimization paradigm in evolutionary computation [54, 100, 101].

From another hand, Backtracking Search Optimization Algorithm (BSA) is a new nature-inspired algorithm proposed by [43] to solve single optimization problems. BSA's

unique mechanism for generating a trial individual is effective, fast and capable of solving different numerical optimization problems successfully and rapidly, with a simple structure. Since it was introduced, the BSA has attracted many researches and it has been applied to various optimization engineering problems [102, 103]. Previously, two presented multi-objective variants of basic BSA algorithm. For example, reference [104] solve multi-type distributed generators along distribution networks problems using a multi-objective BSA algorithm base on a weighting factor approach. The aim of this paper is also to propose a multi-objective variant of BSA, but with a philosophy closer to the one of BSA , both efficient and simple.

The rest of this paper is organized as follows: In Section 2 we present the BSA. In Section 3, the MOBSA approach to solving MO problems is presented. In Section 4, the proposed approach is validated when comparing with various multi-objective optimization algorithms for several unconstrained and constrained benchmarks problems. Finally, Section 5 provides a short summary of the paper.

Backtracking Search Algorithm

Backtracking Search optimization Algorithm (BSA), is one of the recently proposed evolutionary algorithms. Motivation of its development was the need for a simpler and effective search algorithm with a single control parameter. According to the BSA's author, it shows a good convergence performance compared to other algorithms, it is able to solve multimodal problems and it is not over sensitive to the initial value of the control parameter [43].

BSA's performance results from the combination of exploration of the search space and of the use of a search direction matrix. The algorithm can be decomposed into five successive steps: initialization, selection-I, mutation and crossover operators, and selection-II. We explain these five steps in the following subsections.

Initialization

To generate an initial population, BSA uses a uniform random distribution function. The individuals $\mathbf{X}_{i,j}$ of the initial population are built based on Equation (IV.1):

$$\mathbf{X}_{i,j} \sim a.(x_{lj}, x_{ui}) \quad \text{for} \quad i = 1, \cdots, n_{\mathrm{P}} \quad \text{and} \quad j = 1, \cdots, n_{\mathrm{x}}, \tag{IV.1}$$

where $n_{\mathbf{X}}$ is the population size and $n_{\mathbf{x}}$ is the number of decision variables of the problem, $a := \mathscr{R}(0,\ 1)$ is a random variable which follows a uniform distribution between 0 and 1,

and x_{lj} and x_{uj} are respectively lower and upper bounds of the j-th decision variable. Moreover, BSA requires an archived historical population. Then, it uses the Equation (IV.2) to determine the initial values of the historical population $\mathbf{X}_h$:

$$\mathbf{X}_{hi,j} \sim a.(x_{lj}, x_{ui}) \quad \text{for} \quad i = 1, ..., n_{\mathbf{X}} \quad \text{and} \quad j = 1, ..., n_{\mathbf{x}}. \tag{IV.2}$$

Selection I

The first selection step aims to choose the historical population $\mathbf{X}_h$ in order to determine the search direction. In the same way, as for the initial population, the individuals of the historical population are redefined at the beginning of each iteration and computed using the Equation (IV.3) for each $i = 1, ..., n_{\mathbf{X}}$ and $j = 1, ..., n_{\mathbf{x}}$:

$$\text{Let: } a, b := \mathscr{U}(0,\ 1),\ \text{if } a < b \text{ then } \mathbf{X}_{hi,j} \leftarrow \mathbf{X}_{i,j}. \tag{IV.3}$$

Next, a permuting function is used to modify randomly the sequence of individuals of this historical population, Equation (IV.4):

$$\mathbf{X}_h := \text{Permuting}(\mathbf{X}_h). \tag{IV.4}$$

Mutation operator

The mutation operator generates the initial form of the trial mutated population $\mathbf{X}_m$ by using Equation (IV.5):

$$\mathbf{X}_{mi,j} = \mathbf{X}_{i,j} + F(\mathbf{X}_{hi,j} - \mathbf{X}_{i,j}), \qquad \text{for } i = 1, ..., n_{\mathbf{X}} \quad \text{and} \quad j = 1, ..., n_{\mathbf{x}}, \tag{IV.5}$$

where F is a parameter to control the amplitude of the search direction matrix which is computed as the difference between the historical and the current populations matrices $(\mathbf{X}_h - \mathbf{X})$.

In this paper, we use the value $F = \alpha\mathscr{N}$, where α is a real constant to choose and $\mathscr{N}$ is the standard normal distribution.

Crossover Operator

From the trial population built using the mutation process, the BSA's crossover mechanism creates the final form of the trial population, which is denoted $\mathbf{X}_c$. Based on the fitness values affected to $\mathbf{X}_m$, the best individuals in this population are selected to guide the search through the target population individuals.

Equation (IV.6) shows BSA's crossover strategy:

$$\mathbf{X}_{ci,j} = \begin{cases} \mathbf{X}_{i,j}, & \text{if} \quad \mathbf{map}_{i,j} = 1, \\ \mathbf{X}_{mi,j}, & \text{if} \quad \mathbf{map}_{i,j} = 0, \end{cases} \quad \text{for } i = 1, ..., n_{\mathbf{X}} \quad \text{and} \quad j = 1, ..., n_{\mathbf{x}}, \tag{IV.6}$$

where **map** is a $n_{\mathbf{X}} \times n_{\mathbf{x}}$ binary integer-valued matrix which guide crossover directions. This matrix determines the individuals of the trial population $\mathbf{X}_m$ to be mixed with relevant individuals of $\mathbf{X}$.

In addition, a last operation for the crossover step consists in repairing individuals of $\mathbf{X}_c$ which overflow the allowed search-space. These individuals are regenerated as they are in the initialization step so that they belong to the allowed search-space.

Selection II

BSA performs a second selection step from the trial population generated by the crossover step. Thus, individuals from the trial population which have better fitness than individuals in the historical population $\mathbf{X}_h$ are used to update this last one. Similarly, if the so far obtained global minimum has worst fitness than best individual, BSA uses the best individual as new global minimum in the next iteration.

The algorithm 17 gives the pseudo-code of Selection II.

Algorithm 17 Pseudo code of Selection II

Function $\text{glob}_{min} = \text{Selection\_II}(\mathbf{X}, n_{\mathbf{X}}, \mathbf{f}(\mathbf{x}), \mathbf{X}_c)$

$\mathbf{X}_f := (\mathbf{X} \mid \mathbf{f}(\mathbf{x}))$;
$\mathrm{P}_{cf} := (\mathbf{X} \mid \mathbf{f}(\mathbf{x}))$;
for i=1 to $n_{\mathbf{X}}$ **do**
  if $\mathbf{X}_{cf}(:,i) < \mathbf{X}_f(:,i)$ **then**
    $\mathbf{X} := \mathbf{X}_c$;
    $\mathbf{X}_{cf}(:,i) := \mathbf{X}_f(:,i)$;
  end if
end for
$\mathbf{X}_{f_{best}} := \min(\mathbf{X}_f)$; // best$\in$ {1,...,$n_{\mathbf{X}}$;}
if $\mathbf{X}_{f_{best}} < \text{glob}_{min}$ **then**
  $\text{glob}_{min} := \mathbf{X}_{f_{best}}$;
end if

Proposed multi-objective backtracking search algorithm

In this section, the proposed multi-objective backtracking search algorithm (MOBSA) is presented in detail. MOBSA includes the two main operators mutation and crossover

presented by equation IV.5 and in algorithm IV.6 respectively, with non-dominated sorting and constraint domination principle.

Pseudo code of the proposed MOBSA is presented in algorithm 18.

Algorithm 18 Pseudo code of the proposed MOBSA

Function $\widehat{\mathbf{X}} = \text{MOBSA}\left(n, m, \mathbf{f}\left(\mathbf{x}\right), Max_G\right)$

Generate a uniform random initial population $\widehat{\mathbf{X}}$ respect to $\mathbf{x}_l$ and $\mathbf{x}_u$, size of this population $n_{\widehat{\mathbf{X}}} \times n$;
$\widehat{\mathbf{X}}_1 := Find\_Non\_Dominated\left(\widehat{\mathbf{X}} \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
$\widehat{\mathbf{X}}_1 := Crowding\_Distance\left(\widehat{\mathbf{X}}_1\right)$;
for $t = 1 : Max_G$, **do**
  $\widehat{\mathbf{X}}_c^{i+1} := Crossove + Mutation\left(\widehat{\mathbf{X}}_1, \mathbf{x}_l, \mathbf{x}_u\right)$;
  if $\widehat{\mathbf{x}}_{c,t}^{i+1}$ dominates $\widehat{\mathbf{x}}_t^i$ **then**
    $\widehat{\mathbf{x}}_t^{i+1} = \widehat{\mathbf{x}}_{c,t}^{i+1}$
  end if
  if $\widehat{\mathbf{x}}_t^i$ dominates $\widehat{\mathbf{x}}_{c,t}^{i+1}$ **then**
    $\widehat{\mathbf{x}}_{c,t}^{i+1}$ is rejected
  end if
  if $\widehat{\mathbf{x}}_t^i$ and $\widehat{\mathbf{x}}_{c,t}^{i+1}$, are non-dominated with each other, **then**
    $\widehat{\mathbf{x}}_{c,t}^{i+1}$ add to the population
  end if
  After this step ,$\widehat{\mathbf{X}}_2$ the size of this population between $n_{\widehat{\mathbf{X}}} \times n$ to $2n_{\widehat{\mathbf{X}}} \times n$;
  $\widehat{\mathbf{X}} := Find\_Non\_Dominated\left(\widehat{\mathbf{X}}_2 \mid \mathbf{f}\left(\mathbf{x}\right)\right)$;
  $\widehat{\mathbf{X}} := Crowding\_Distance\left(\widehat{\mathbf{X}}_2\right)$;
end for

Fast non-dominated sorting

The fast non-dominated sorting procedure was developed in the framework of NSGA-II [**?**]. In the elaboration of this procedure, the domination count n_p, the number of solutions which dominate the solution p, and the set of solutions that the solution p dominates S_p are calculated for every solution. The first non-dominated front is thus created and initialized with all solutions having zero as domination count. Then, for each solution p with $n_p = 0$, each member q of its set S_p is visited and its domination count is reduced by one. As a result, if for any member, the domination count is equal to zero, we put it in a separate list Q. The second non-dominated front is then created as the union of all individuals belonging to Q. The procedure is repeated for subsequent fronts (F_3, F_4, etc.) until all individuals are assigned their ranks. The fitness is set to a level number, lower numbers correspond to higher fitness (F_1 is the best).

Crowding distance

The crowding distance defined in [7] is used as an estimate of the diversity measure of individuals surrounding a given individual i in the population. This distance is the average distance between two individuals located on either side of the given particular solution along each objective. The average distance between individuals $i-1$ and $i+1$ boarding the individual i, located on the Pareto front is depicted in Figure IV.1. Such distance is an estimation of the perimeter of cuboid formed by using the nearest neighbors. This metric represents half of the perimeter of the cuboid encompassing the solution i.

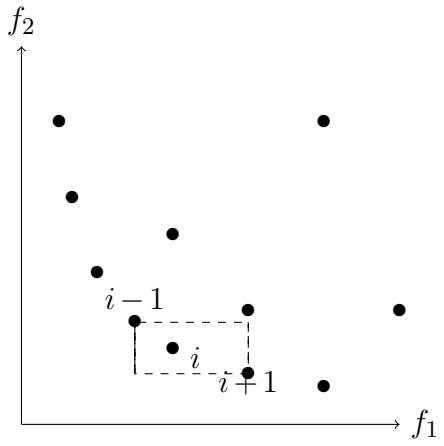

Figure IV.1 – Crowding distance of individual i.

The main consideration from the crowding distance is to find the Euclidean distance between each individual in a front based on their m objectives. The computation of the crowding distance, based on the normalized values of objectives, is given by the algorithm 19, where f_m^{max} and f_m^{min} are the maximum and minimum value of the m^{th} objective function respectively. The sum of individual crowding distance values corresponding to each objective gives the overall crowding distance value.

Algorithm 19 Crowding distance calculation for a set solution $\mathcal{I}$

1: $n = |\mathcal{I}|$ // number of solutions in I;
2: **for** each i, **do**
3: set $\mathcal{I}[i]_{distance} = 0$;
4: **end for**
5: **for** each objective m, $\mathcal{I} = \text{sort}(\mathcal{I}, m)$ **do**
6: $\mathcal{I}[1]_{distance} = \mathcal{I}[n]_{distance} = \infty$;
7: **for** i=2 to (n-1) **do**
8: $\mathcal{I}[i]_{distance} = \mathcal{I}[i]_{distance} + (\mathcal{I}[i+1]_{.m} - \mathcal{I}[i-1]_{.m})/(\mathrm{f}_m^{max} - \mathrm{f}_m^{min})$;
9: **end for**
10: **end for**

In this algorithm, $\mathcal{I}$ is a non-dominated set, n is the number of elements of $\mathcal{I}$, $\mathcal{I}[i]_m$ is the m^{th} objective value of the i^{th} individual in $\mathcal{I}$, and sort$(\mathcal{I}, m)$ is sorting of the individuals of $\mathcal{I}$ according to the m^{th} objective.

Advantages of this algorithm are:

- Its simplicity, if we compare to most of the state-of-the-art EAs.
- Its efficiency to solve difficult MO problems, as it is demonstrated in the rest of this paper.

Experimental study of MOBSA

In this section, we compare the performance of MOBSA with other two algorithms, MOEA/D [94] and NSGA-II [7], for solving fifteen well-known complicated multi-objective problems including UF1-UF8 unconstrained problems with variable linkage and constrained CF1-CF7 problems. The set of complicated MO problems studied in Congress on Evolutionary Computation (CEC) [2] is chosen to assess the performance of the proposed MOBSA. Indeed, these MO problems are generally considered in the literature since Pareto fronts obtained have different shapes.

Experimental setting

In our experimental study, the source code of MOEA/D [94] can be found at http://dces.essex.ac.uk/staff/zhang/webofmoead.htm. The code of NSGA-II can be obtained from the original authors [7]. The simulated binary crossover (SBX) operator and polynomial mutation [1] are applied in MOEA/D and NSGA-II for UF1-UF8 unconstrained problems and CF1-CF10 constrained problems. For SBX, the crossover probability pc is set as 0.8 and the distribution index for crossover is set as 15. For polynomial mutation, the distribution index for mutation is set as 20 and the mutation probability pm is set as 0.5. For MOBSA, the value of controls amplitude H used in this paper is $H = 3 \times \mathcal{N}(0, 1)$ ($\mathcal{N}$ is the standard normal distribution). The population size $n_X$of all compared algorithms is set as 300 for UF1-UF8 problems and 300 for CF1-CF7 problems. For MOEA/D, their neighbor size is set as 20. The maximum number of function evaluation is set as 300,000 for 30-dimension UF problems, and 300,000 for 10-dimension CF8-CF10 problems.

Performance metric

In this study, we follow the choice of the reference [2] which considers the Inverted Generational Distance (IGD) metric to assess for quantitative evaluation of the algorithm performance. IGD metric proposed by [58] is as follows. Suppose $\mathbf{X}^*$ is a set of uniformly distributed solutions original Pareto font and let $\widehat{\mathbf{X}}$ be the approximate solutions of the Pareto-optimal. Then:

$$\mathrm{IGD}(\mathbf{X}^*, \widehat{\mathbf{X}}) = \sqrt{\frac{\sum_{\mathbf{x}^* \in \mathbf{X}^*} d(\mathbf{X}^*, \widehat{\mathbf{X}})^2}{|\widehat{\mathbf{X}}|}}, \tag{IV.7}$$

where $d(\mathbf{X}^*, \widehat{\mathbf{X}})$ is the minimum Euclidean distance between $\mathbf{X}^*$ and the points in $\widehat{\mathbf{X}}$.

Experimental study on Unconstrained MO problems

The eight unconstrained benchmark problems UF1-UF8 are addressed in this section. They are defined in the appendix of this paper.

Figure IV.2 provides an example of Pareto fronts obtained by MOBSA for 5 independent runs when using a population size of 300 and a maximum generation number of 1000, for the UF1, UF2 and UF4 problems. As the initial population is random, the Pareto front is different for each run but we can see how the results are obtained each time. However, instead of a visual inspection, MOBSA's performance is best studied from performance indicators such as the IGD metric.

Hence, Figure IV.3 shows the evolution of the IGD along the number of functions evaluations for one typical run. Moreover, in order to compare the performance of MOBSA, we have also plotted the IGD evolution for the MOEA/D and NSGA-II algorithms in graphs of this figure. However, it is obvious that MOBSA's performance can only be studied in a statistical manner since different results are obtained at each run.

Table IV.1, shows the Mean and Standard Deviation (SD) of the IGD metric obtained using three algorithms MOBSA, MOEA/D and NSGA-II. In this table, boldfaced numbers indicate the best of IGD mean across algorithms for each benchmark problem. It can be observed from Table IV.1, that the MOBSA is able to converge better than any other algorithm on five problems (UF1-UF2 and UF6-UF8). For problem ZDT3, the two algorithms MOEA/D and NSGA-II outperforms MOBSA. For problems UF4 and UF3, the MOBSA performs better than MOEA/D, but a little worse than NSGA-II in term of convergence. Concerning results of MOBSA standards deviations, it can be observe that they remain small after 100 runs for all unconstrained problems.

Figure IV.4 illustrates the box plots of comparisons between the proposed MOBSA

and other two compared algorithms MOEA/D and NSGA-II with for unconstraint problems based on the IGD metric. Box plots are used to illustrate the distribution of results over 100 independent runs. Symbol '+' denotes outliers.

As can be seen from Figure IV.4 that MOBSA performances stable and superior to NSGA-II and MOEA/D in solving the more complex problems UF1, UF2, UF6, UF7 and UF8. For tri-objective problem UF8, MOBSA performs as well as MOEA/D and superior to NSGA-II. For two UF4 and UF5 problems, MOBSA is more stable than MOEA/D and but better than NSGA-II. For UF3, MOEA/D performs better than two algorithms MOBSA and NSGA-II.

According to the experimental results in Table IV.1 and Figure IV.4, we can come to the conclusion that the proposed MOBSA converges faster and performs better than the NSGA-II algorithm in solving the bi-objective problems UF1, UF2, UF6, UF7 and tri-objective problem UF8. As for the bi-objective problems UF4 and UF5, NSGA-II converges faster than the MOBSA algorithm. Comparing with MOEA/D, MOBSA converges faster in dealing with most of the bi-objective problems except UF3 problem.

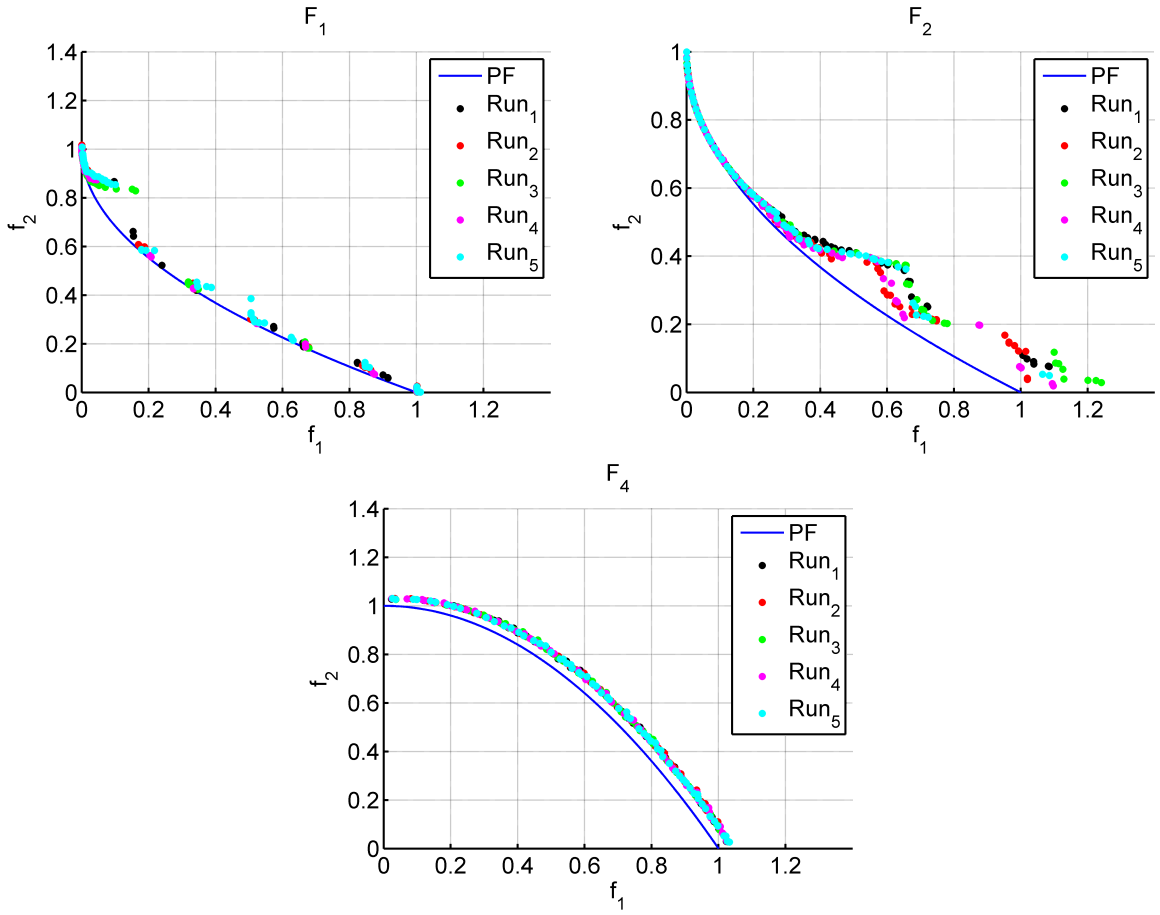

Figure IV.2 – Pareto fronts obtained for 5 independent runs of MOBSA when solving UF1 problem (left) UF2 problem (middle) and UF4 problem (right).

Algorithm	Statistic	UF1	UF2	UF3	UF4	UF5	UF6	UF7	
MOBSA	Mean	**1.61E-03**	**1.46E-03**	9.57E-03	1.17E-03	8.21E-02	**1.04E-02**	**5.36E-04**	**5**
	SD	5.92E-04	4.03E-04	2.29E-03	2.28E-05	5.60E-03	5.22E-04	1.78E-04	
MOEA/D	Mean	4.42E-03	6.22E-03	**4.95E-03**	6.07E-02	3.19E-01	1.76E-01	4.43E-03	
	SD	9,61E-05	1.20E-03	5.67E-03	4.15E-03	1.22E-01	1.12E-01	1.53E-04	
NSGA-II	Mean	2.23E-03	1.90E-03	6.47E-03	**1.10E-03**	**6.36E-02**	1.09E-02	5.17E-03	
	SD	8.57E-04	3.63E-04	1.79E-03	5.84E-05	2.23E-02	5.44E-03	5.24E-04	

Tableau IV.1 – Mean and Standard Deviation of the IGD metric results for the unconstrained benchmark problems

Constrained benchmark problems results

As real engineering problems are inclined to have constrained conditions, we are interested to investigate the capability of the proposed algorithm in handling constrained MO Problems. To do this, we have selected the CF1-CF7 problems in CEC2009 MO Problem contest. They are defined in the appendix of this paper.

Figure IV.5 provides an example of Pareto fronts obtained by MOBSA for 5 independent runs when using a population size of 300 and a maximum generation number of 1000, for the CF1, CF2 and CF4 problems. As it is explained in the previous section, since the initial population is random, the obtained Pareto front is different for each run but we can see how are the results each time. In addition, Figure IV.6 shows the evolution of the IGD along the number of functions evaluations for one typical run, both for the MOBSA and for the MOEA/D algorithm by using the algorithm parameters defined in the previous section[1].

Table IV.2, shows the mean and Standard Deviation (SD) of the IGD metric obtained using three algorithms MOBSA, MOEA/ and NSGA-II. In this table, boldfaced numbers indicate the best of IGD mean across algorithms for each benchmark problem. As can be seen from Table IV.2, MOBSA is capable of finding a better mean of IGD than any other algorithm on all the problems except CF5 and CF6. MOEA/D performs the best on CF5 and CF6 problem in terms of average IGD metric. Moreover, one can note that MOBSA standard deviation remains small for all problems after 100 runs.

Figure IV.7, shows the box plots of the IGD metric obtained by the proposed MOBSA and other two compared algorithms over 100 independent runs in solving the seven bi-objective constraint problems CF. Symbol "+" denotes outliers.

Figure IV.7, shows the box plots of IGD metric obtained by MOBSA , MOEA/D and NSGA-II. It can be seen that MOBSA performs better than two other algorithms in solving five problems CF1-CF4 and CF7, of the seven problems. For UF5 problem, the

[1]However, MOEA/D used a penalty method to handle constraints of these problems, where 10^{15} is chosen for the penalty constant.

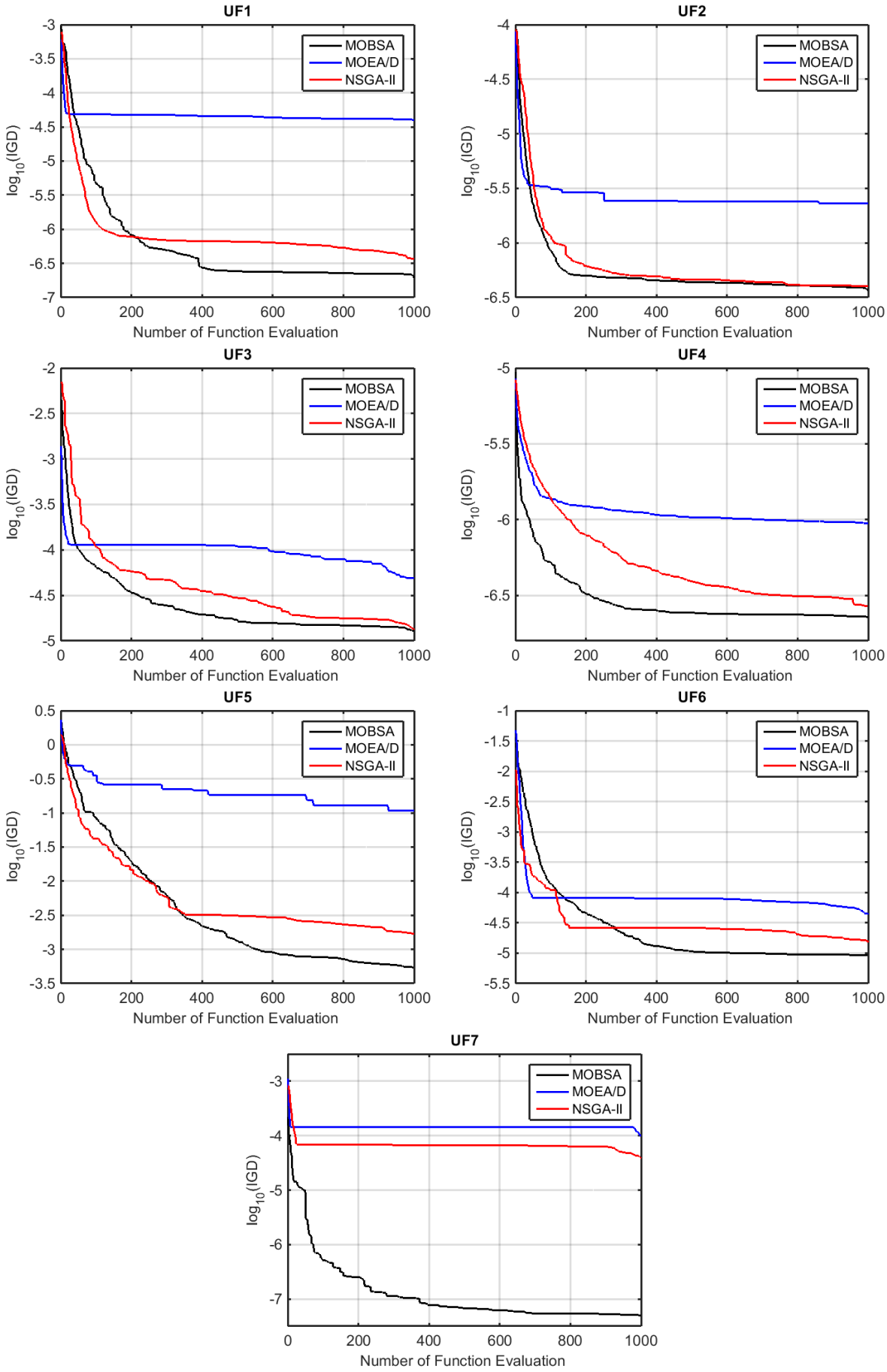

Figure IV.3 – Plot of the IGD evolution versus the number of functions evaluations for UF seri problems by using MOBSA, MOEA/D and NSGA-II algorithms in one typical run.

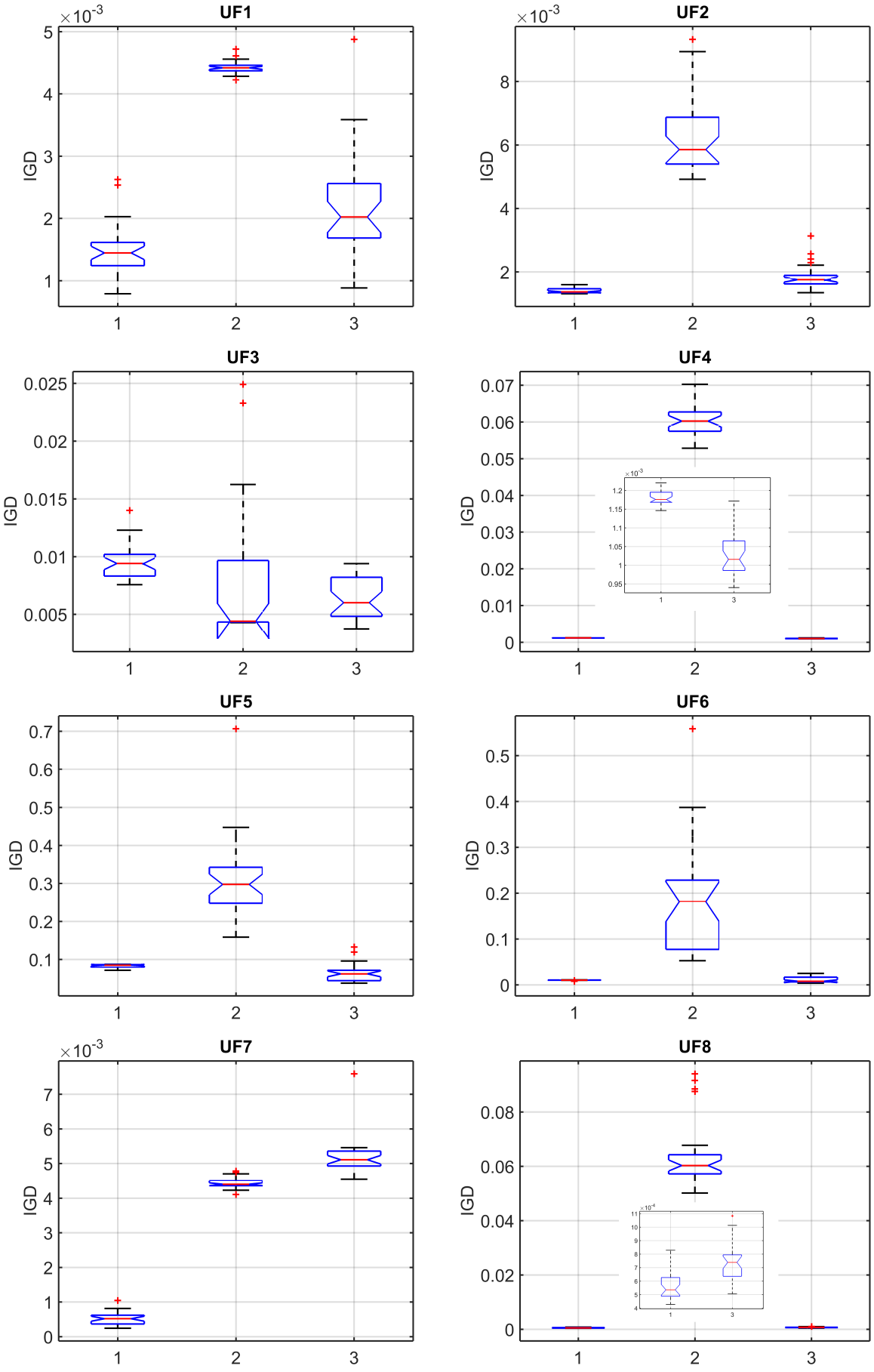

Figure IV.4 – Box plots of IGD metric obtained by three compared algorithms, MOBSA ("1" column), MOEA/D ("2" column) and NSGA-II ("3" column), in solving the eight UF unconstrained problems.

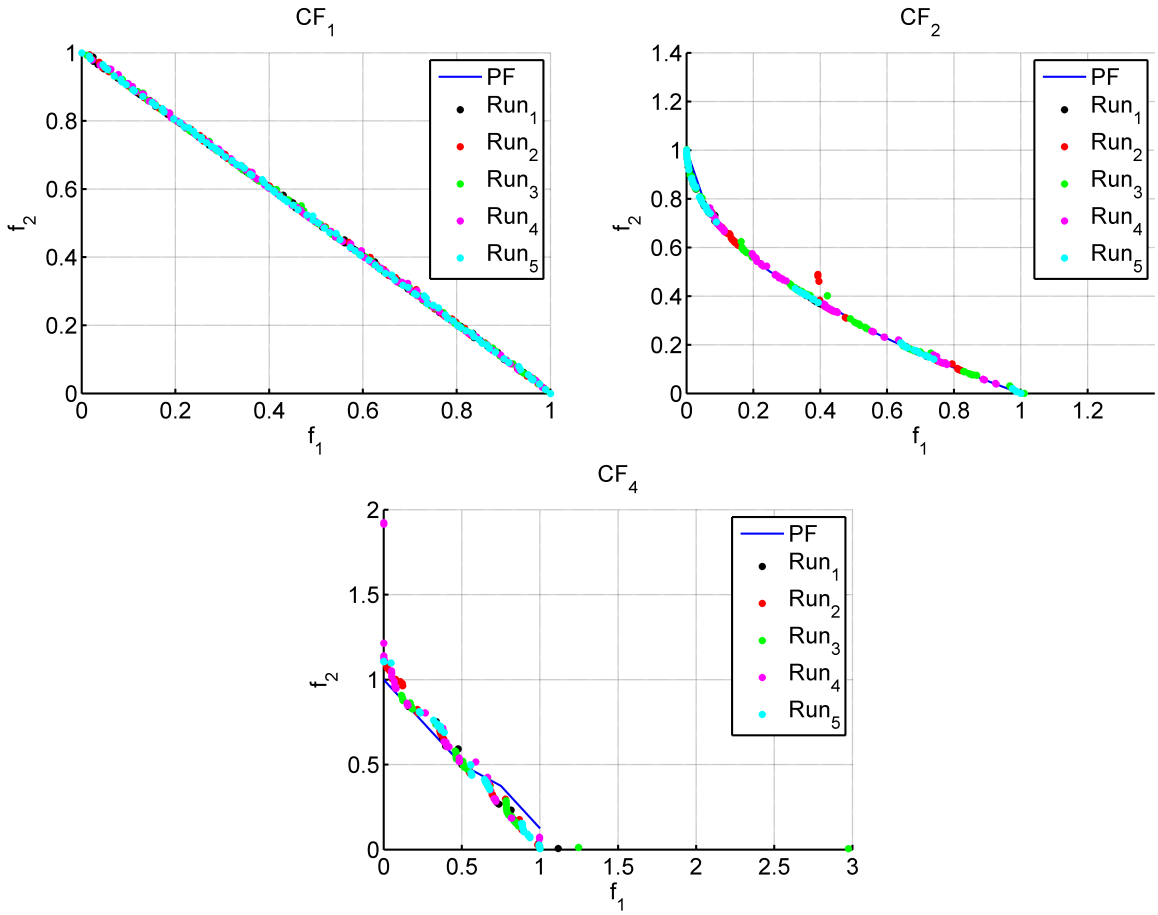

Figure IV.5 – Pareto fronts obtained for 5 independent runs of MOBSA when solving CF1 problem (left), CF2 problem (middle) and CF4 problem (right).

MOBSA algorithm is more stable than MOEA/D, but NSGA-II superior to MOBSA. For UF6, NSGA-II converge faster than two algorithm MOBSA and NSGA-II.

Simulation results show that MOBSA has remarkable performance. It is concluded that MOBSA can obviously converges faster and performs better than the five bi-objective constraint problems CF1-CF4 and CF7.

Algorithm	Statistic	CF1	CF2	CF3	CF4	CF5	CF6	CF7
MOBSA	Mean	**1.31E-05**	**3.39E-03**	**3.01E-03**	**1.67E-03**	4.93e-03	**1.90E-03**	**4.97E-03**
	SD	1.22E-06	1.22e-03	1.47E-04	1.29e-04	1.32e-04	3.65E-05	5.247E-04
MOEA/D	Mean	4.32E-04	1.14E-02	1.76E-02	1.70E-03	1.84E-02	2.41E-03	1.89E-02
	SD	3.88E-05	1 4.61E-03	3.37E-03	1.24E-03	7.65E-03	9.89E-04	9.52E-04
NSGA-II	Mean	5.32E-04	2.30E-03	1.69E-02	4.25E-03	**4.74E-03**	6.11E-05	5.07E-03
	SD	5.14E-05	2.74E-03	3.08E-03	1.29E-03	1.58E-03	3.35E-04	5.24E-04

Tableau IV.2 – Mean and Standard Deviation of the IGD metric results for the constrained benchmark problems

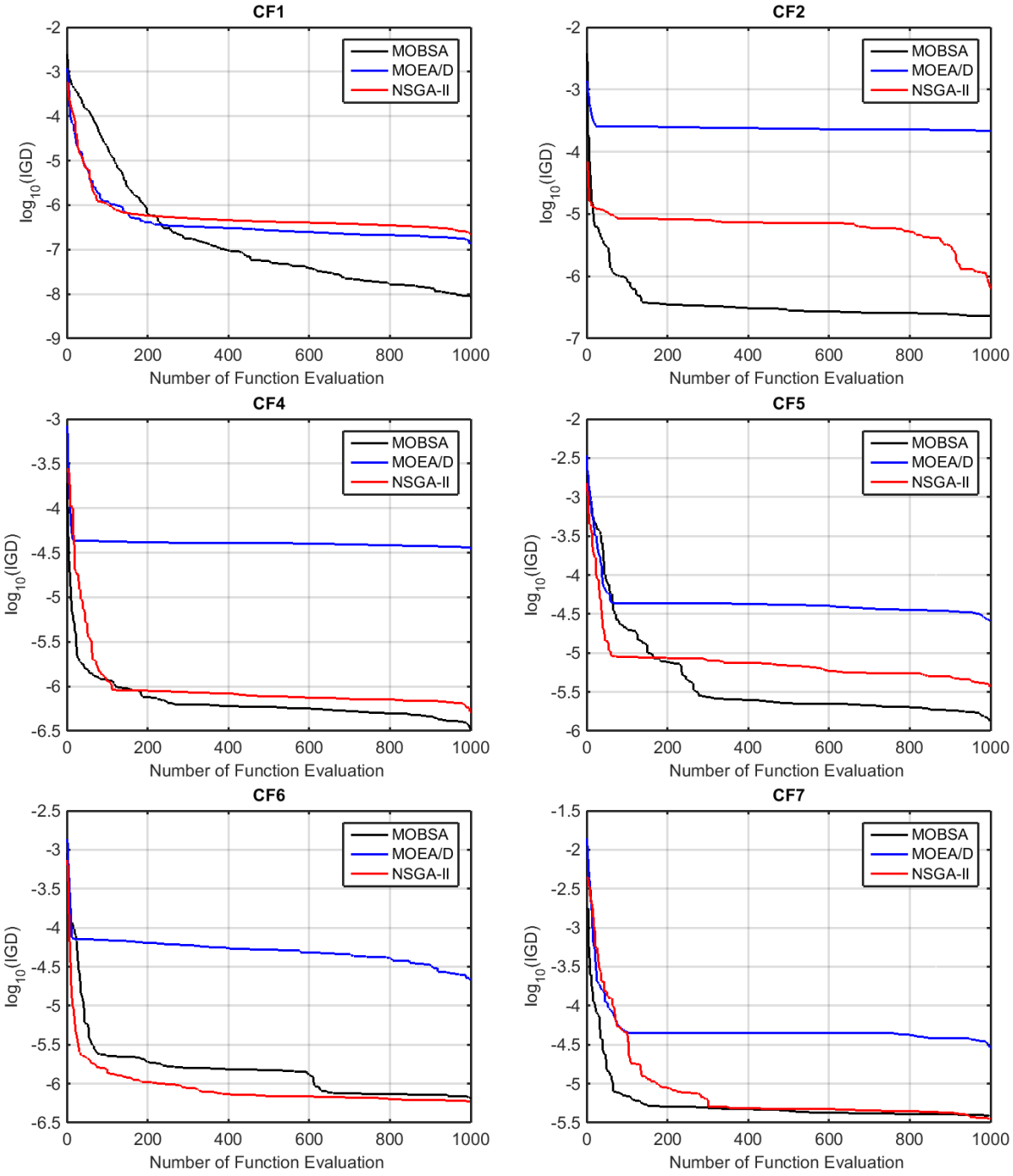

Figure IV.6 – Plot of the IGD evolution versus the number of functions evaluations for CF problems by using MOBSA, MOEA/D and NSGA-II algorithms in one typical run

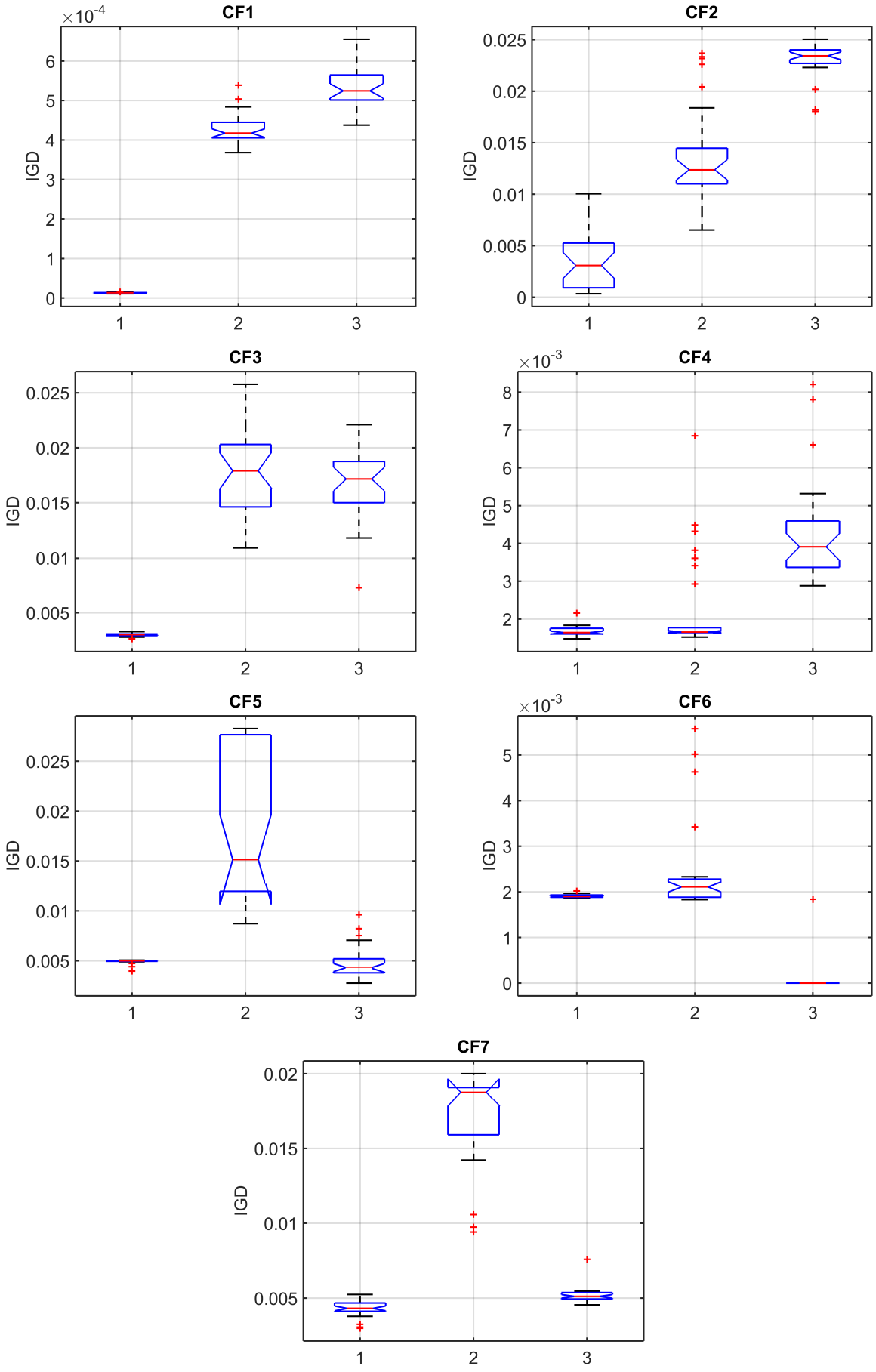

Figure IV.7 – Box plots of IGD metric obtained by three compared algorithms, MOBSA ("1" column), MOEA/D ("2" column) and NSGA-II ("3" column), in solving the seven CF constrained problems.

Conclusion

In this chapter, we propose to extend the Backtracking Search Algorithm to solve multi-objective optimization problems. BSA is a simple and effective global search algorithm with a single control parameter developed for single-objective optimization problems, called Multi-objective Optimization Backtracking Search Algorithm (MOBSA).

MOBSA which adopts the non-dominated sorting concept and the mechanism of crowding distance calculation, to solve unconstrained or constrained multi-objective problems. It is a simple algorithm, which has the advantage of having only one parameter.

The performance of this algorithm is demonstrated experimentally with the help of IGD metric indicator for fiveteen benchmarks problems including eight unconstrained problems and seven constrained problems. Then, it is compared to the most considered algorithms in the field: MOEA/D, NSGA-II and all algorithms in CEC platform. Hence, we can conclude that MOBSA is able to solve complex MO problems Overall, the results showed that MOBSA provides, most of the time, very competitive results compared to others algorithms.

In the next chapter we will be presenting, the applications of this algorithm on the mechanical structures.

Chapter V

Structural design using multi-objective Backtracking Search Algorithm

Contents

Based on: **A. Tchvagha Zeine, N. El Hami, S. Ouhimmou, R. Ellaia and A.El Hami**, *Multiobjective optimization of trusses using Backtracking Search Algorithm*, Uncertainties and Reliability of Multiphysical Systems., Vol. 1, N°1. (2017) (Published by ISTE Ltd. London, UK).

Structural Optimization and Mechanics

Since the last two decades, the optimal design of structures raises more keen interest. The methods of optimization still too little applied to the traditional techniques of mechanical design office in civil engineering, it gradually integrates with it as its reliability increases [3, 105]. The idea of optimal design of structures, even if it is probably very old, is only possible for a few decades thanks to the advent of the computer. Its implementation is described in Figure V.1. Based on an initial design defined by a number of variable value parameters, called design variables, the optimization aims to automatically determine which design is the best in terms of structural performance criteria. The solution found by this iterative process, alternating structural analysis and application of one of the optimization methods, is described as optimal design [106, 107].

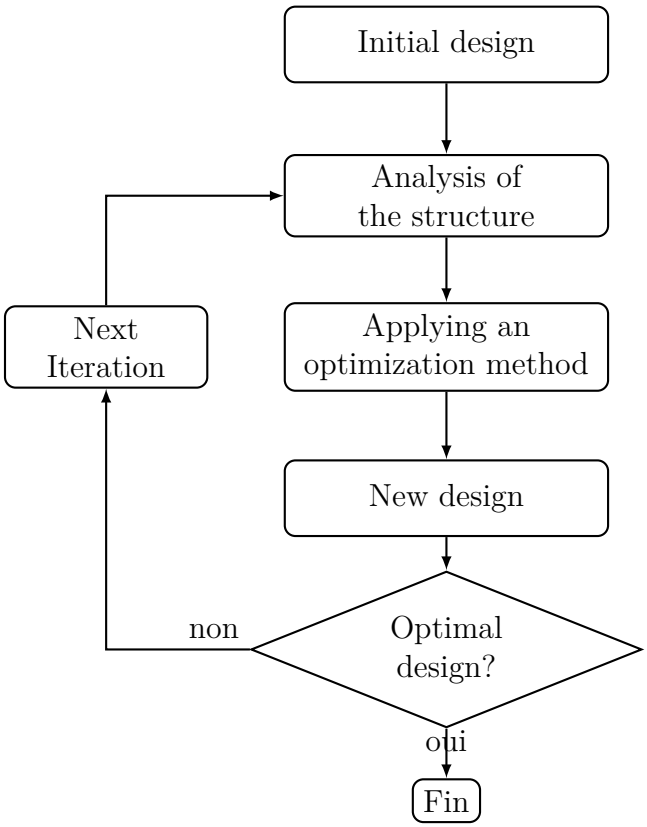

Figure V.1 – Implementation of the optimization of structures.

The finite element method appeared with the need to solve complex computational problems, many industrial software based on the finite element method have been developed for example ANSYS, ABAQUS.... Today, PDE's numerical resolution methods are mature enough to allow them to participate in engineering design assistance processes. In the optimization domain, a number of algorithms are developed for solving these types

of problems [108]. In this thesis we consider stochastic methods as mentioned in the first two chapters.

In the field of calculation of mechanical structures, the engineer therefore has a wide range of methods, supported by computer tools, including the finite element method and optimization methods that are valuable allies for the optimal design of structures in accordance with certain rules or standards.

The development of the art of engineering requires considerable effort to continually improve structural design techniques. Optimization is important in increasing the performance of mass, position or cost. This work addresses the problem of truss structural, which have been dealt with in collaboration with both LERMA laboratory at the Mohammadia School of engineering and the LMN laboratory at the INSA of Rouen.

Introduction

Many structural design problems e.g. truss structural mechanical are naturally multi-objective, i.e., they have several conflicting objectives that have to be optimized simultaneously. Typically, we sometimes aim to minimize the weight of a structure and to maximize the displacement between nodes while enhancing its robustness.

Many structural design problems e.g. truss structural mechanical are naturally multi-objective, i.e., they have several conflicting objectives that have to be optimized simultaneously. Typically, we sometimes aim to minimize the weight of a structure and to maximize the displacement between nodes while enhancing its robustness [109]. However, most of the engineering design problems involve multiple and often conflicting design objectives. The solution of such problems is very difficult compared to single-objective optimization. For Multi-objective Optimization (MO) Problems, as objectives are usually conflicting, there is no one optimal solution but a set of trade-offs solutions. This set of trade-offs solutions is known as Pareto-optimal set.

Over the past decade, metaheuristic algorithms have been found to be more flexible and efficient for solving this kind of optimization problems [16, 110, 111]. Evolutionary algorithms (EAs), as part of metaheuristics, have been applied widely to solve MO Problems this algorithm called Multi-Objective Evolutionary Algorithms (MOEAs). Many studies reporting the resolutions of structural optimization problems using metaheuristics have been published [32, 112–117].

In this chapter, multi-objective Optimization Backtracking search algorithm (MOBSA), presented in the previous chapter IV is used to solve the multi-objective truss structural problems.

The rest of this paper is organized as follows. The next section 3 presents the explicit

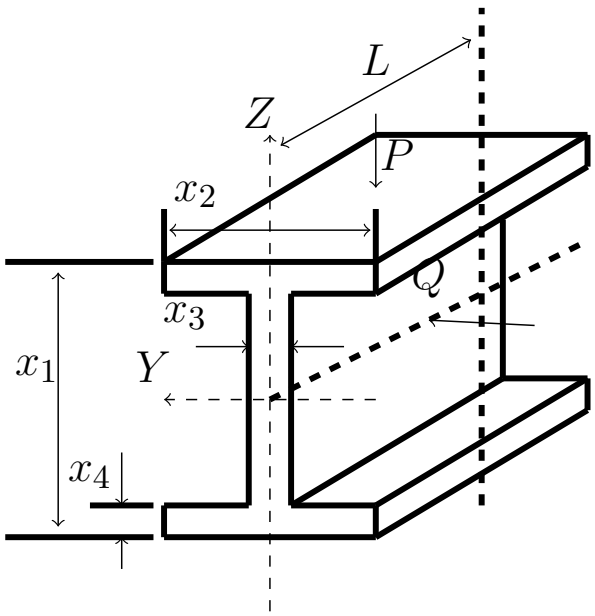

Figure V.2 – I-Beam optimization

case of truss structural design problem. In section 4, we treat the implicit cases of two complex problems.

Truss structural design problem: Explicit case

In this section, we present the explicit case for four classic problems. Those problems have the advantage of being available in analytical form while showing the important characteristics of structural design optimization.

I-Beam optimization

This example has been proposed by [117] as is illustrated on the figure V.2. We want to minimize the weight of the beam and to maximize its displacement. The mathematical definition of the problem is:

$$\begin{cases} \min\limits_{\mathbf{x}} \ f_1(\mathbf{x}) = 2\mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_3(\mathbf{x}_1 - 2\mathbf{x}_4) \\ \max\limits_{\mathbf{x}} \ f_2(\mathbf{x}) = \frac{\mathbf{P}\mathbf{L}^3}{48EI} \end{cases} \tag{V.1}$$

Subject to:

$$\begin{cases} g(\mathbf{x}) = \frac{\mathrm{M}_y}{\mathrm{W}_y} + \frac{\mathrm{M}_z}{\mathrm{W}_z} - \mathrm{k}_g \geq 0, \\ 10 \leq \mathbf{x}_1 \leq 80 \\ 10 \leq \mathbf{x}_2 \leq 50 \\ 0.9 \leq \mathbf{x}_3 \leq 5 \\ 0.9 \leq \mathbf{x}_4 \leq 5 \end{cases} \tag{V.2}$$

where:

$$\mathrm{W}_y = \left[\frac{2\mathbf{x}_2\mathbf{x}_4(4\mathbf{x}_4^2 + 3\mathbf{x}_1(\mathbf{x}_1 - 2\mathbf{x}_4)) + \mathbf{x}_3(\mathbf{x}_1 - \mathbf{x}_4)^3}{6\mathbf{x}_1}\right]$$

$$\mathrm{W}_z = \frac{2\mathbf{x}_4\mathbf{x}_2^3 + \mathbf{x}_3^3(\mathbf{x}_1 - \mathbf{x}_4)}{6\mathbf{x}_2}$$

$$\mathrm{I} = \left[\frac{2\mathbf{x}_2\mathbf{x}_4(4\mathbf{x}_4^2 + 3\mathbf{x}_1(\mathbf{x}_1 - 2\mathbf{x}_4)) + \mathbf{x}_3(\mathbf{x}_1 - 2\mathbf{x}_4)^3}{12}\right]$$

In equation V.1, the function f_1 represents the weight of the beam, and the function f_2 represents the maximum displacement of bream. The values of the deterministic parameters are given in Table V.1. In this table, k_g and E designate the stress yield point and the Young's modulus of the material, respectively, and M_y and M_z designate maximum bending moments of a beam around the x and y axes respectively.

Tableau V.1 – The parameter values

Var	P(KN)	Q(KN)	E(KN/CM^2)	k_g(KN/CM^2)	L(CM)	M_y(KN.CM)	M_z(KN.CM)
Valeur	600	50	2×10^4	16	200	30000	2500

The results shown in Figure V.3, shows the Pareto-optimal front obtained by the approaches that we compare with MOBSA and NSGA-II algorithms, after 100 iteration, we use a population size of 100 points respectively. The Pareto front obtained by this algorithms is shown in right Figure V.3, the non-dominated solutions obtained by MOBSA with a good distribution original Pareto front (PF) and good spread compared with results obtained by algorithm NSGA-II shown in right) Figure V.3.

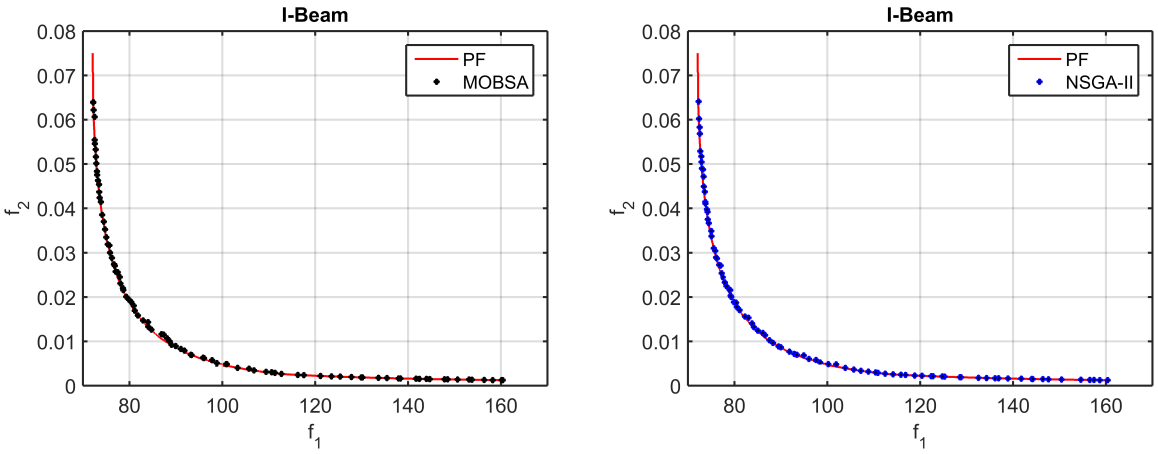

Figure V.3 – Pareto front for MOBSA (left) and NSGA-II (right) the I-Beam

Two bars truss design

This case study, using a two-bar truss design problem, which has been well studied by many researchers [112] and [114]. The problem is shown in Figure V.6.

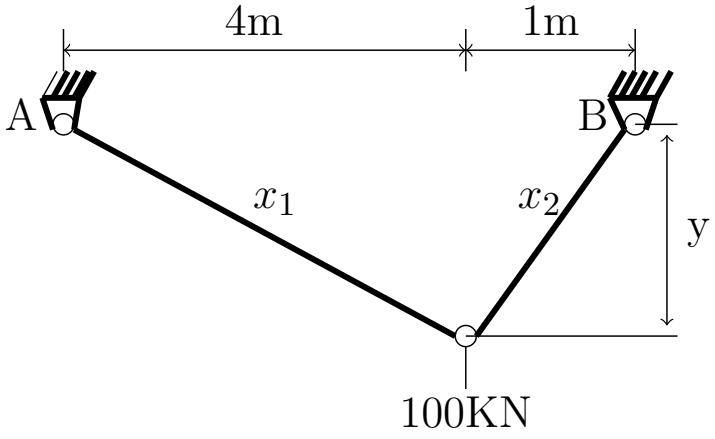

Figure V.4 – The two bars truss design

The mathematical description of the problem is as follows:

$$\begin{cases} \min\limits_{x} \left(f_1(x), f_2(x)\right) \\ \text{such that} \\ f_1(x) = x_1\sqrt{16+x_3^2} + x_2\sqrt{1+x_3^2} \\ f_2(x) = 20\sqrt{16+x_3^2}/(x_1 y) \end{cases} \tag{V.3}$$

Subject to:

$$\begin{cases} f_1 \leq 0.1,\ f_1 \leq 10^5, \\ f_{\text{stress,BC}} = 80\sqrt{1+x_3^2}/(x_2 y) \leq 10^5 \end{cases} \tag{V.4}$$

where

$$0 \leq x_1, x_2, \quad \text{and} \quad 1 \leq x_3 \leq 3$$

The resultant Pareto optimal front curve is shown on Figure V.5 obtained after 100 iterations and the population size is set as 100 points. Figure V.5 shows also the convergence efficiency of MOBSA and the uniformly distributed solutions on the Pareto front after 100 point non-dominated. However, we note that the optimization method that we developed offers better convergence to PF compared to results obtained by NSGA-II.

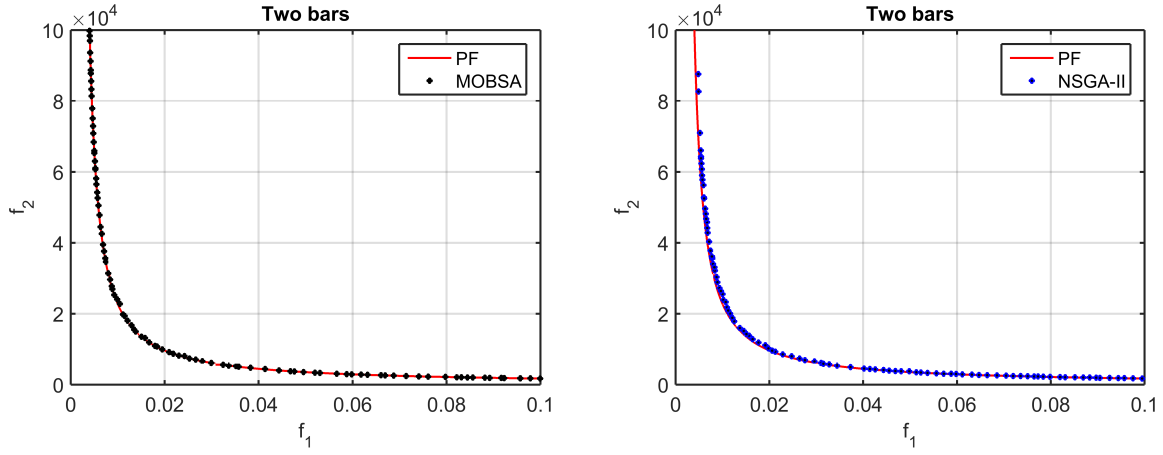

Figure V.5 – Pareto front for MOBSA (left) and NSGA-II (right) the two bars truss design

Four bars

As an example, the multi-objective optimization problem of four bars taken from [32, 112] shown in Figure V.6. The objective is to find an optimal structure of the four bars simultaneously minimizing the total mass and displacement of node C. We consider the following functions :

- f_1: The total volume,
- f_2: The displacement

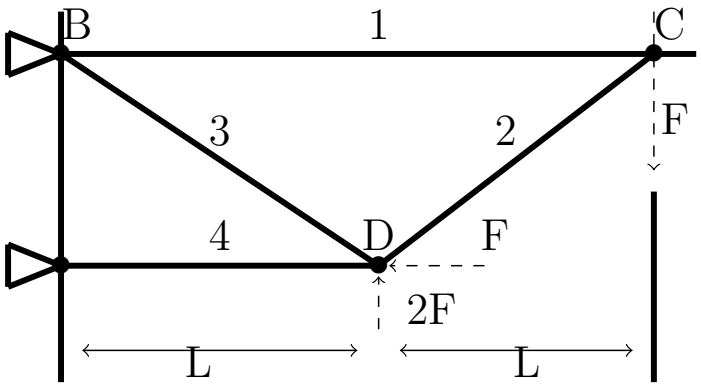

Figure V.6 – Four bars

The mathematic formulations in this functions followings:

$$\min_{x} : \begin{cases} f_1(x) = \mathrm{L}(2x_1 + \sqrt{2}x_2 + \sqrt{x_3} + x_4) \\ f_2(x) = \frac{\mathrm{FL}}{\mathrm{E}}(\frac{2}{x_1} - \frac{2\sqrt{2}}{x_2} + \frac{2}{x_4}) \end{cases} \tag{V.5}$$

Subject to:

$$\begin{cases} \frac{\mathrm{F}}{\sigma} \leq x_1, x_4 \leq 3\frac{F}{\sigma} \\ \sqrt{2}\frac{\mathrm{F}}{\sigma} \leq x_2, x_3 \leq 3\frac{F}{\sigma} \end{cases} \tag{V.6}$$

The resultant Pareto optimal front curve in figure V.7 obtained after 250 iterations and the population size is set as 50 points. In figure V.7, demonstrate the convergence efficiency of BSAMO, we note that the uniformly distributed solutions on the Pareto front after 50 and 100 point non-dominated. However, we note that the optimization method that we developed offers better convergence to PF.

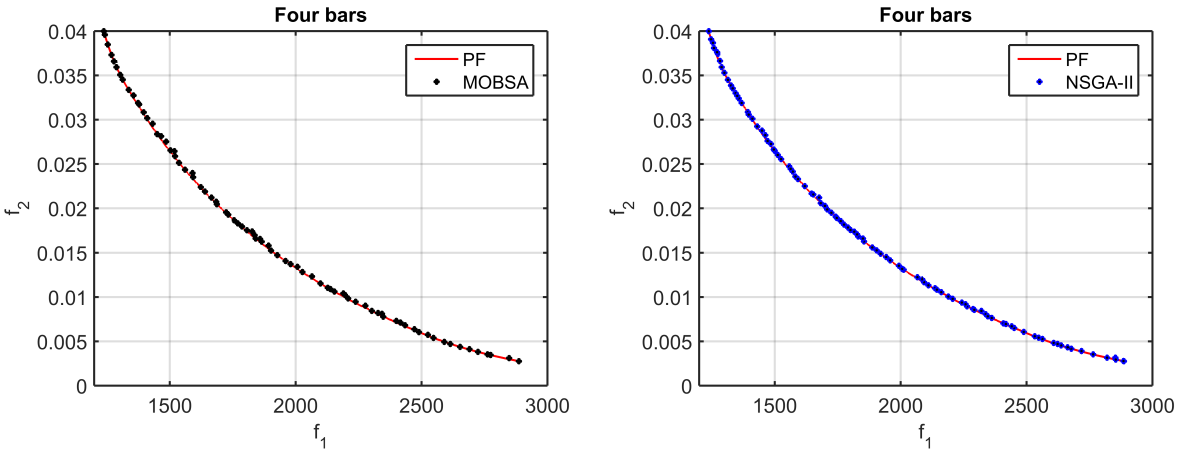

Figure V.7 – Pareto front of the Four Bars

Design of a disc brake system

In this application, we deal with the disc brake design optimization studied by [118] and [119]. The objectives are to minimize the mass of the brake system and to minimize the stopping time. We consider four design variables $\mathbf{x} = (x_1, x_2, x_3, x_4)$: the inner radius of the disc x_1, the outer radius of the disc x_2, the engaging force x_3 and the number of friction surfaces x_4. The mathematical formulation of the problem is as follows:

$$\min_{\mathbf{x}} : \begin{cases} f_1(\mathbf{x}) = 4.9 \times 10^{-5}(x_2^2 - x_1^2)(x_4 - 1) \\ f_2(\mathbf{x}) = \dfrac{9.82 \times 10^6 (x_2^2 - x_1^2)}{x_3 x_4 (x_2^3 - x_1^3)} \end{cases} \tag{V.7}$$

subject to:

$$\begin{cases} \mathrm{c}_1(\mathbf{x}) = 20 - (x_2 - x_1) \leq 0, \\ \mathrm{c}_2(\mathbf{x}) = 2.5(x_4 + 1) - 30 \leq 0 \\ \mathrm{c}_3(\mathbf{x}) = \dfrac{x_3}{3.14(x_2^2 - x_1^2)} - 0.4 \leq 0, \\ \mathrm{c}_4(\mathbf{x}) = \dfrac{2.22 \times 10^{-3} x_3 (x_2^3 - x_1^3)}{(x_2^2 - x_1^2)^2} - 1 \leq 0, \\ \mathrm{c}_5(\mathbf{x}) = 900 - \dfrac{2.66 \times 10^{-2} x_3 x_4 (x_2^3 - x_1^3)}{(x_2^2 - x_1^2)} \leq 0, \end{cases} \tag{V.8}$$

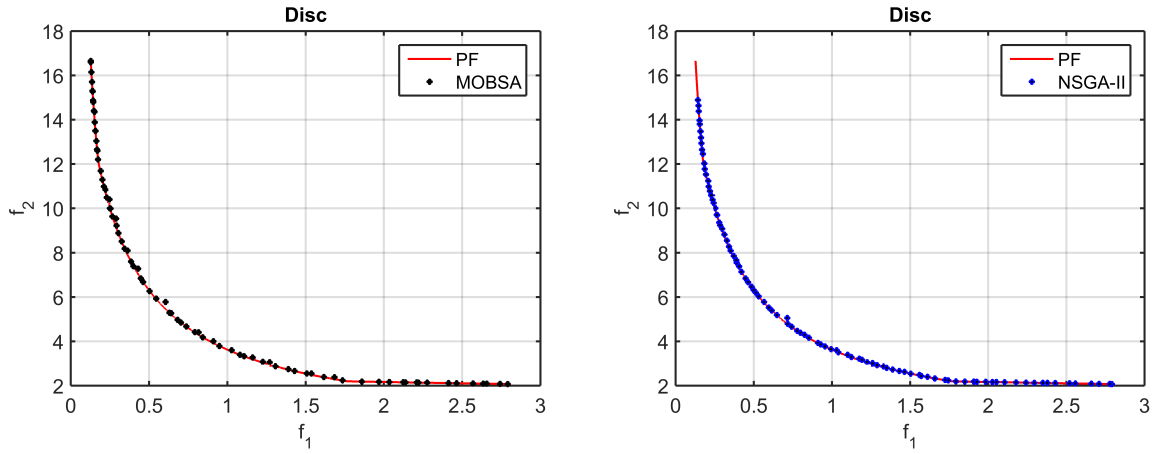

Figure V.8 – Pareto front for MOBSA (left) and NSGA-II (right) for the disc brake design application

Figure V.8 shows the approximated Pareto fronts obtained using MOBSA and NSGA-II for this design application. MOBSA generally yields better approximations of the Pareto front compared to NSGA-II.

Design of a welded beam

This second application concerns the multi-objective design of a welded beam as formulated in [118, 119]. In this problem, we optimize two objective functions namely: the fabrication cost f_1 and the end deflection $f_2 = \delta$. This problem has four design variables $\mathbf{x} = (w, L, d, h)$: the width w and length L of the welded area, the depth d and thickness h of the main beam. The mathematical formulation of this application is as follows:

$$\min_{\mathbf{x}} : \begin{cases} f_1(\mathbf{x}) = 1.10471 w^2 L + 0.04811 dh(14 + L) \\ f_2(\mathbf{x}) = \delta \end{cases} \tag{V.9}$$

subject to:

$$\begin{cases} \mathrm{c}_1(\mathbf{x}) = w - h \leq 0, \ \mathrm{c}_2(\mathbf{x}) = \delta - 0.25 \leq 0 \quad \mathrm{c}_3(\mathbf{x}) = \tau - 13600 \leq 0, \\ \mathrm{c}_4(\mathbf{x}) = \sigma - 30000 \leq 0, \quad \mathrm{c}_5(\mathbf{x}) = 0.10471 w^2 + 0.04811 hd(14 + \mathrm{L}) - 5 : 0 \leq 0, \\ \mathrm{c}_6(\mathbf{x}) = 0.125 - \mathrm{W} \leq 0, \quad \mathrm{c}_7(\mathbf{x}) = 6000 - \mathrm{P} \leq 0, \end{cases} \tag{V.10}$$

for $0.1 \leq l, d \leq 10, \quad 1.125 \leq w, h \leq 2$ and where:

$$\mathrm{Q} = 6000(14 + \frac{\mathrm{L}}{2}), \quad \mathrm{D} = \frac{1}{2}\sqrt{\mathrm{L}^2 + (w+d)^2}, \quad \delta = \frac{65856}{30000 hd^3},$$

$$\mathrm{J} = \sqrt{2}\left[\frac{\mathrm{L}^2}{6} + \frac{(w+d)^2}{2}\right], \quad \beta = \frac{\mathrm{QD}}{\mathrm{J}}, \quad \sigma = \frac{504000}{hd^2},$$

$$\alpha = \frac{6000}{\sqrt{2} w \mathrm{L}}, \quad \tau = \sqrt{\alpha^2 + \frac{\alpha\beta \mathrm{L}}{\mathrm{D}} + \beta^2}, \qquad \mathrm{P} = \frac{4.013 \times 30 \times 10^6}{196} \frac{dh^3}{6} (1 - d\frac{\sqrt{\frac{30}{48}}}{28}).$$

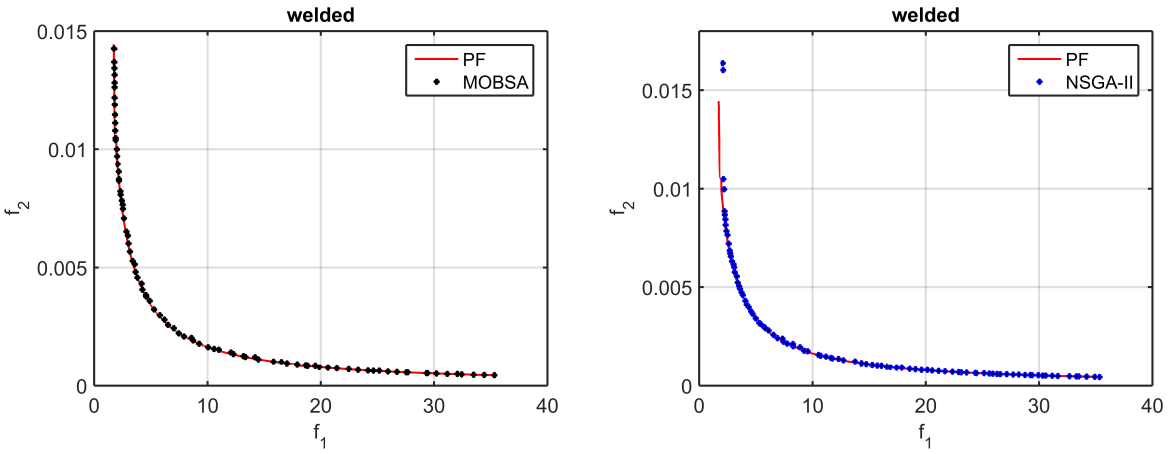

Figure V.9 – Pareto front for MOBSA (left) and NSGA-II (right) for the welded beam design application

The obtained Pareto fronts are shown in Figure V.9, with the fabrication cost f_1 and the end deflection f_2 on the horizontal and vertical axes, respectively. As it can be seen, a better quality distribution and approximated Pareto front is obtained for MOBSA compared to NSGA-II.

Truss structural design problem: Implied case

In this section, we address the multi-objective sizing and topology optimization of truss-like structures which is a continuous subject of researches in structural design [32, 82–84].

Fourteen-bar truss design problem

In a first step, the fourteen-bar truss design problem formulation is introduced. Then, a first application case is presented where we consider three bi-objective and constrained optimization problems. Next, a second application case concerned one three objective constrained optimization problem.

Problem formulation

Let the design domain comprises a set of nodes with fixed spatial coordinates, a set of supports and a set of loads, as it is sketch in the Figure V.10 for the fourteen-bar truss test. It is assumed that the structure will be modeled by linear, two nodes, bar elements in linear elasticity, subjected only to axial forces and free from imperfections.

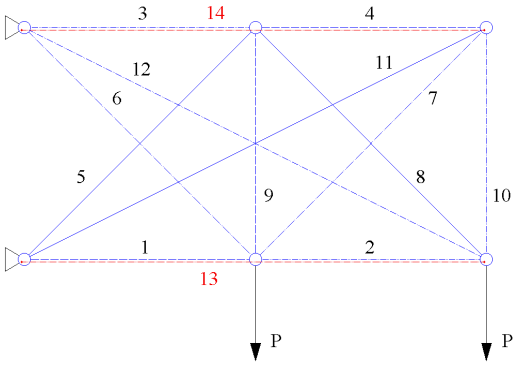

Figure V.10 – Sketch of the fourteeen-bar truss.

From a mechanical point of view, several choices can be of interest for objectives functions. For example, two objectives functions can be minimized: the mass and the displacement while one objective function can be maximized: the first flexible natural

frequency of the structure. Then, denoting $\mathbf{x} \in \Omega$ the vector of the topological and sizing optimization parameters, such that:

$$0 \leq x_i \leq 1 \quad \text{for} \quad i \in \{1, \ldots, n\}$$

where $n = 14$ is the number of elements, the three individual objectives that can be minimized are:

1. The mass w of the structure:

$$w(\mathbf{x}) = \sum_{i=1}^{n} \rho A l_i x_i,$$

where l_i is the length of the i-th element, $\rho = 2{,}768$ kg/m^3 is the density of the material and $A = 0.01419352$ m^2 is the element cross-section area.

2. The maximum displacement u of the structure:

$$u(\mathbf{x}) = \max\left(\mathbf{u}^* = \arg\min_{\mathcal{S}}\left(\frac{1}{2}\mathbf{u}^{\mathrm{T}}\mathbf{K}(\mathbf{x})\,\mathbf{u} - \mathbf{u}^{\mathrm{T}}\mathbf{F}\right)\right),$$

where $\mathbf{F}$ is the vector of loads and $\mathbf{K}$ is the stiffness matrix of the finite element (FE) model, having the Young' modulus $E = 68.95$ GPa. The set $\mathcal{S}$ refers to the kinematic admissible space, *i.e.* the one that satisfies the imposed boundary conditions given by the supports while carrying all the prescribed loads, where $P = 448.2$ kN.

3. The opposite of the minimum flexible natural frequency f of the structure, in order to maximize it:

$$-f(\mathbf{x}) = -\min\left(\frac{1}{2\pi}\omega^*\right),$$

$$\text{where}: \left\{\omega^{*2}, \mathbf{u}^*\right\} = \arg\min_{\mathbf{u}\in\mathcal{S}}\left(\omega^2 = \frac{\mathbf{u}^{\mathrm{T}}\mathbf{K}(\mathbf{x})\,\mathbf{u}}{\mathbf{u}^{\mathrm{T}}\mathbf{M}(\mathbf{x})\,\mathbf{u}}\right), \quad \|\mathbf{u}\| \neq 0$$

where $\mathbf{M}$ is the mass matrix of the FE model[1].

Moreover, all possible optimization problems must be subjected to constraints for the mechanical stress σ_i for each element i:

$$|\boldsymbol{\sigma}_i(\mathbf{x})| \leq \overline{\sigma} \qquad i \in \{1, \ldots, n\}$$

where $\overline{\sigma} = 172.4$ MPa is the yield strength.

[1] To obtain the best numerical efficiency for the FE analysis, the FE disassembly strategy proposed in reference [85] is involved.

Then, thanks to the introduced formulation and the domain of definition for $\mathbf{x}$, the sizing and the topology of the structure are optimized concurrently. As a consequence, local rigid body modes or kinematic modes may appears in some cases when some design variables are set to zero. For example, when $x_4 = x_7 = 0$, the element 10 is free to rotate. In practice, this problem is detected since minimal natural frequency of the bar truss is null. As designs with local rigid body modes or kinematic modes are not of interest, one supplementary constraint is added to the MO problem formulation:

$$f(\mathbf{x}) > \epsilon$$

where $\epsilon = 10^{-5}$.

Numerical simulations for the three bi-objectives functions problems

In this subsection, the performance of MOBSA is compared with NSGA-II in solving the three objective functions by considering them two-by-two: (w, u), (w, f) and (u, f).

Besides that, the parameters used in NSGA-II are: population size: 50; mutation probability: pm = 0,5; crossover probability: c = 0,8; and distribution index of SBX: 15. The parameter used in MOBSA is 100 for the population size.

Figures V.11 shows typical Pareto fronts obtained after 250 iterations of one typical run, when both algorithms start from the same initial guess population. From this figure, a better diversity and a better convergence are observed for MOBSA compared to NSGA-II. This is confirmed by the range of objective values obtained that are compared in Table V.2.

Problem	Range of Pareto fronts values	MOBSA	NSGA-II
(w, u)	w (kg) u (mm)	(745, 6012) (-24.4, 177.2)	(749, 6345) (-24.4, 176.9)
(w, f)	w (kg) f (Hz)	(727, 1973) (-28.1, -21.2)	(727, 1974) (-28.2, -18.8)
(u, f)	u (mm) f (Hz)	(24.4, 104.9) (-28.1, -14.1)	(24.4, 102.1) (-27.9, -13.6)

Tableau V.2 – Comparison of the results for the three bi-objectives fourteen-bar truss design problems.

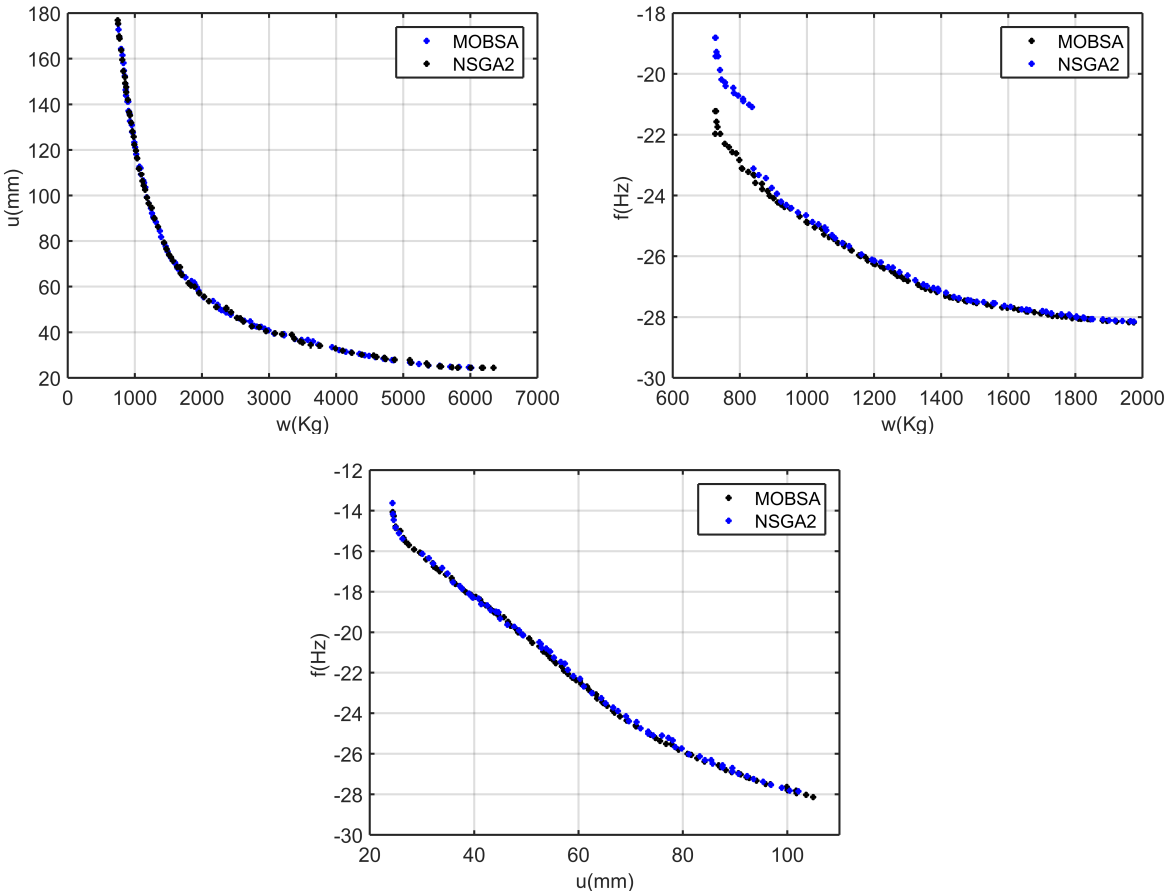

Figure V.11 – Pareto fronts of the fourteen-bar truss MO problem.

Numerical simulation for three objectives functions problem

The three objectives functions problem is considered here. Figure V.12 shows views of the Pareto front obtained for one typical run of 250 iterations using MOBSA with a population size of 5000. This result is satisfactory, showing that MOBSA is efficient for solving complex problems design of truss structures when three objectives and multiple constraints exist.

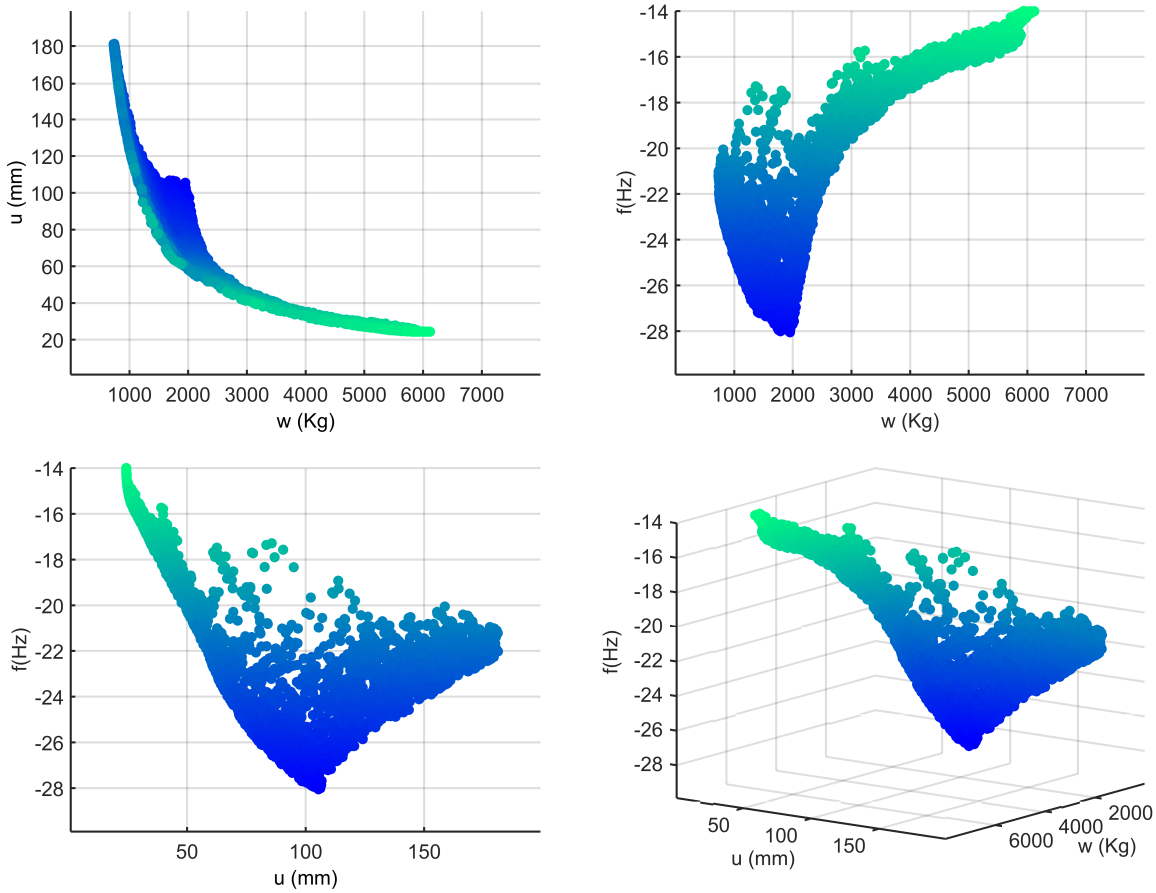

Figure V.12 – Four different views of the Pareto front obtained for one typical run when solving the three objectives functions of the fourteen-bar truss problem, Colorized points of the sub-figures is added for a better visualization and the color corresponds to the frequency objective f .

Truss structure design subjected to random loads

In this section, the truss structure subjected to random Gaussian loading presented in reference [120] is investigated (see Figure V.13).

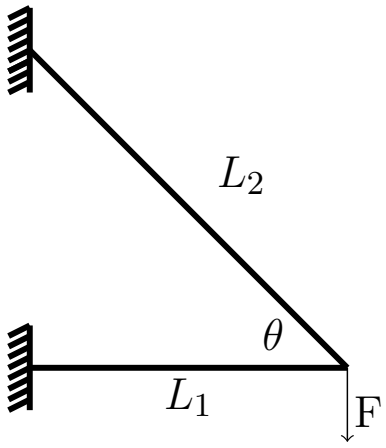

Figure V.13 – Sketch of the two bars truss

Problem formulation

The areas of the two bars are decision variables of the optimization. The left end of the bars is fixed while the other end is subjected to a mean plus a fluctuating load. The deterministic material properties of the steel are the Young's modulus $E = 2.1 \times 10^{11}$ Pa and the density $\rho = 7800$ kg m^{-3}. The horizontal bar has an area S_1 and length $L_1 = 1$ m while the other bar has an area S_2 and a length L_2 such that $L_2 = \frac{L_1}{cos(\theta)}$. $\theta = \frac{\pi}{4}$ rad is the angle between the two bars. Bounds for the areas are: $2.2 \leq S_1 \leq 20$ and $5 \leq S_2 \leq 10$ if they are expressed in mm^2. PSDbars

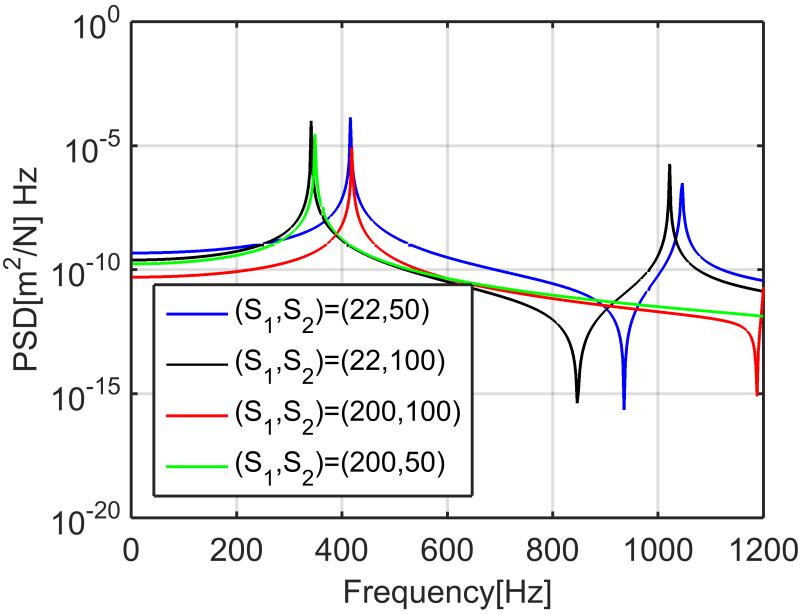

Figure V.14 – PSD of the two-bar truss for four extremum designs.

In this problem, it is of interest to design the truss in respect to a vertical, Gaussian, stationary load process $F(t)$, where t denotes the time. It is described by its mean $\overline{F} = 10{,}000$ N and its power spectrum density (PSD), which is modeled by a constant function over a limited range of frequencies (f_1, f_2):

$$\begin{cases} 2000 & \text{if} \quad f_1 \leq f \leq f_2 \\ 0 & \text{if} \quad \text{not} \end{cases} \tag{V.11}$$

From a linear hypothesis for the structural behavior, the random vertical displacement $v(t)$ is obviously Gaussian too, characterized by its mean μ_V and its standard deviation σ_V. From reference [120], it has been established that $\mu_V = h(\omega = 0) \times \overline{F}$ or

$$\mu_V = \left(\frac{1}{k_1 \tan(\theta)^2} + \frac{1}{k_1 \sin(\theta)^2} \right) \times \overline{F}$$

and

$$\sigma_V^2 = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \Phi_{VV}(\omega) d\omega$$

with

$$\Phi_{VV} = h(\omega) \Phi_{FF}(\omega) h^H(\omega)$$

where $h(\omega)$ is the frequency response function of the truss:

$$h(\omega) = \left[\frac{k_2 (k_1 - \omega^2 m) \sin(\theta)^2}{k_1 - \omega^2 m + k_2 \cos(\theta)^2} - \omega^2 m \right]^{-1}$$

for $k_1 = E S_1 L_1$, $k_2 = E S_2 L_2$, $m = \frac{\rho}{3}(S_1 L_1 + S_2 L_2)$ and where $\omega = 2\pi f$ is the angular frequency.

Then, two objectives are considered for the optimization problem: the mean and the standard deviation of the vertical displacement. It is thus formulated as follows:

$$\min_{(S_1, S_2)} \begin{cases} \mu_V(S_1, S_2), \\ \sigma_V(S_1, S_2) \end{cases} \tag{V.12}$$

subject to:

$$\begin{cases} 22 \leq S_1 \leq 200, \\ 50 \leq S_2 \leq 100 \end{cases} \tag{V.13}$$

for two frequency bands: (f_1, f_2)=(100, 300) and (f_1, f_2)=(600, 800).

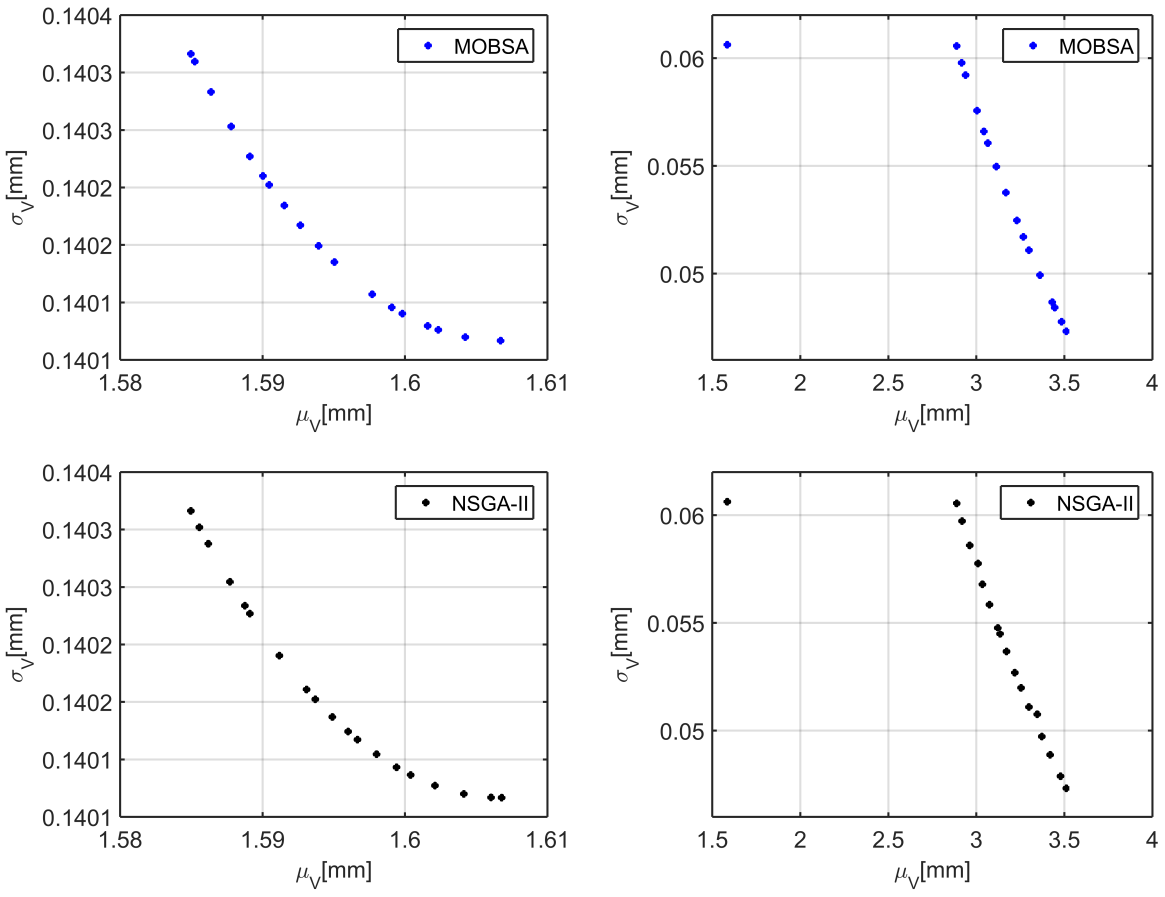

Figure V.15 – MOBSA (up) and NSGA-II (down) Pareto fronts obtained for the frequency band (100, 300) (left) and (600, 800) Hz (right) of the two-bar truss optimization problem.

The problem under consideration is a MO optimization problem having two objectives – μ_V and σ_V in Equation V.12 – conflicting. This is shown in Figure V.15. Reason is the following. An excited structure responds according to its natural-dynamics modes, or, equivalently, to its frequency responses functions or impulse responses functions. Hence, the dynamic response is affected by the shape of the excitation profile and the structural responses functions, which in turn leads to the resulting standard deviation in the structural response. On another hand, the mean structural response is given by the structural null frequency response or, equivalently, its static response, independently of its dynamic response. As a consequence, these two quantities, mean and standard deviation can be conflicting, depending on the shape of the excitation profile and on the shape of the structural responses functions. Figure V.14 is also provided to better understand this point.

Numerical simulations

From our experience, MOEA/D does not succeed in solving this problem while NSGA-II [7] does it. Figure V.15, illustrates results obtained by MOBSA and NSGA-II for the two considered frequencies band, when starting from the same initial population and for 100 iterations. Parameters used in NSGA-II are: population size: 20; mutation probability: $p_m = 0.5$; crossover probability: $c = 0.8$; and distribution index of SBX: 15.

To compare algorithms, 30 independent runs are achieved. Figure V.16 shows the box-plots of IGD obtained by MOBSA and NSGA-II for the two frequency bands. It can be seen that, statistically, MOBSA performs better than NSGA-II.

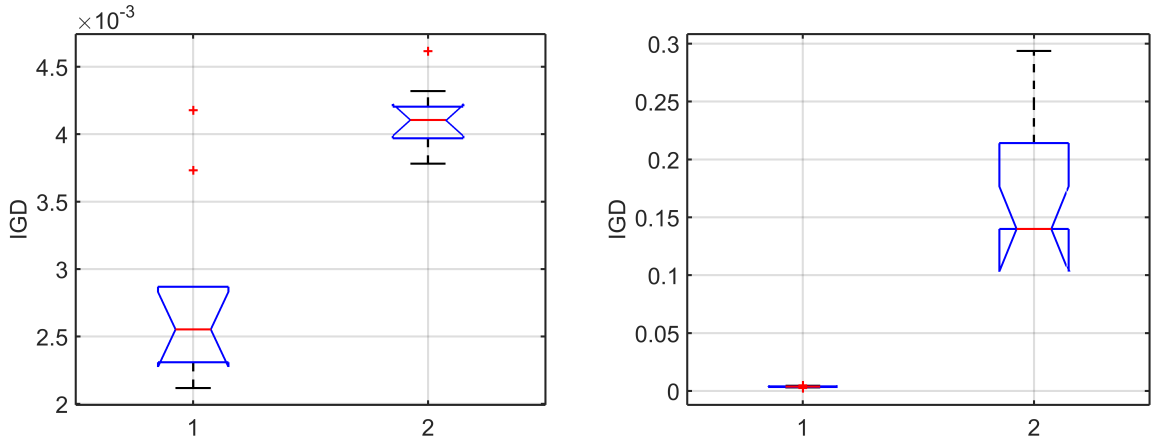

Figure V.16 – Box-plots of IGD for the frequency band (100, 300) (left) and (600, 800) Hz (right) for the two-bar truss obtained by MOBSA ("1" column) and NSGA-II ("2" column)

Conclusion

The results presented in this chapter illustrate the application of the proposed method MOBSA presented in chapter IV in the field of structural design. We presented interesting results in the optimal design of static structures, in the explicit case. We also dealt with two implicit cases for which the results were satisfactory. The extension to implicit cases has pushed us to confront the problems we have encountered and to highlight them in order to eventually lead to the development of a methodology for the resolution of complex structure design problems.

The proposed methodology was implemented and tested by numerical simulation on a basic case namely the beam, then a number of applications were processed. For each of the problems the results given by MOBSA, with the same convergence parameters which

were fixed a priori following the recommendations of the previous chapter, were compared with those given by other methods chosen in the literature and allowed to validate the consistency and effectiveness of our proposed algorithm MOBSA.

Extending our study to other structural design problems is an interesting research path that we intend to explore in the near future. Another interesting research line has to do with the fact that the variation operators used in the metaheuristics operate without any knowledge of the problem, with suggests that BSA operators could be designed to provide more effective and efficient search capabilities, which could speed up the optimization process.

Chapter VI

MOBSA application in fluid-structure interaction problems

Contents

Based on: **R. ElMaani, A. Tchvagha Zeine, R. Bouchaib, A. El Hami, R. Ellaia,** *Backtracking search optimization algorithm for fluid-structure interaction problems*, 4th International Colloquium on Information Science and Technology (IEEE-CiSt'2016), pp.690-695, 24-26, October 2016, Tangier-Assilah, Morocco.

Fluid-structure interaction problem

A general fluid structure interaction problem consists of the description of the fluid and solid fields, appropriate fluid structure interface conditions at the conjoined interface and conditions for the remaining boundaries, respectively.

In the following, the fields and interface conditions are introduced, furthermore, a brief sketch of the solution procedure for each of the fields is presented.

About the fluid

All kinds of fluid flow and transport phenomena are governed by basic conservation principles such as conservation of mass, momentum and energy. All these conservation principles are solved according to the fluid model which gives set of partial differential equations, called the governing equations of the fluid. The following part elaborates on the theoretical background of CFD and the way it is employed for this particular case [121].

The continuity equation for a compressible fluid can be written as follows

$$\frac{\partial \rho}{\partial t} + div\,(\rho u) = 0 \tag{VI.1}$$

where ρ represents the density and u represents velocity of the fluid. The first term of the equation is the rate of change of density with respect to time and the next term is net flow of mass out of the element boundaries.

Newton's second law states that the rate of change of momentum of a fluid particle equals to the sum of the forces acting on a particle. The forces acting on a body are a combination of both surface and body forces. When this law is applied for Newtonian fluid (viscous stress is proportional to the rates of deformation) resulting equations are called as Navier-Stokes equations. The equations written below explain the momentum conservation principle [121]:

$$\frac{\partial(\rho u_i)}{\partial t} + div(\rho u_i u) = -\frac{\partial p}{\partial x} + div(\mu \nabla u_i) + S_{Mx} \tag{VI.2}$$

$$\frac{\partial(\rho u_j)}{\partial t} + div(\rho u_j u) = -\frac{\partial p}{\partial y} + div(\mu \nabla u_j) + S_{My} \tag{VI.3}$$

$$\frac{\partial(\rho u_k)}{\partial t} + div(\rho u_k u) = -\frac{\partial p}{\partial z} + div(\mu \nabla u_k) + S_{Mz} \tag{VI.4}$$

where ρ represents the density; u represents velocity vector; u_i; u_j; u_k are the velocity components in Cartesian coordinate system, μ is the dynamic viscosity and S_M represents

the momentum source term. Since the problem at hand does not involve the heat transfer, energy equation is not considered.

About the structure

In structural mechanics problems, in general, the task is to determine deformations of solid bodies, which arise because of the action of various kinds of forces. From this, for instance, stresses in the body can be determined, which are of great importance for many applications. For the different material properties there exist a large number of material laws, which together with the balance equations lead to diversified complex equation systems for the determination of deformations (or displacements).

The basic governing equation of motion is given as [122]:

$$m\ddot{u} + c\dot{u} + ku = f(t) \tag{VI.5}$$

where m is a structural mass matrix, $\ddot{u}$ is an acceleration vector, c is a structural damping matrix, $\dot{u}$ is a velocity vector, k is a structural stiffness matrix, u is a displacement vector, f is a force vector which is a function of time, the structural damping is not involved in the finite element model so the above governing equation is modified into following form:

$$m\ddot{u} + ku = f(t) \tag{VI.6}$$

It is normal practice to use a numerical technique called finite element method (FEM) to find the solution for the equation (VI.6). The basic principle behind this method of finding an approximate solution to the differential equations is to divide the volume of a structure or system in to smaller (finite) elements such that infinite number of DOFs is converted to a finite value [123, 124].

Interface Conditions

The main conditions at the interface are the dynamic and kinematic coupling conditions. The force equilibrium requires the stress vectors to be equal as:

$$\sigma^f.n = \sigma^s.n \quad \forall x \in \Gamma^{fsi} \tag{VI.7}$$

also the normal velocities at interface the interface have to match as

$$u.n = \frac{\partial d}{\partial t}.n \quad \forall x \in \Gamma^{fsi} \tag{VI.8}$$

Numerical discretization

The numerical computation is developed in two steps. In the first one, the conservation equations are formulated and an approach is adopted to evaluate all the terms. In the second one, a segregated, sequential solution algorithm is used to form the element matrices, to assemble them and to solve the resulting system for each variable separately ϕ [125]. In order to solve the governing equations of the fluid motion (VI.2) (VI.3) (VI.4), their discretized form must first be generated. Thus, the first step is the generation of a grid, which consists of dividing the solution domain into a finite number of computational elements [126]. In the second step, each term of the partial differential equation describing the flow is written in such a manner that the computer can be programmed to calculate it [127].

Finite element discretization

The FEM divides the continuum region of interest into a number of simply shaped regions called elements. In this discretization method, the variables within each element, are interpolated using a local polynomial $N_j(x_i)$ (shape or interpolation function) in terms of the values j at a set of nodal points j in a way that guarantees continuity of the solution across element sides [128**?**]:

$$\phi = \sum_{j=1}^{n} N_j \phi_j \tag{VI.9}$$

where N_j is a polynomial shape function at nodes j and n is the number of nodes on the element. The discretization process, therefore, consists of deriving the element matrices to put together the matrix equation [125]:

$$\left(\left[A_e^{transient}\right] + \left[A_e^{advection}\right] + \left[A_e^{diffusion}\right]\right)\{\phi_e\} = \left\{S_e^{\phi}\right\} \tag{VI.10}$$

Galerkin's method of weighted residuals is used to form the element integrals [125]. Each degree of freedom is solved in sequential fashion. The equations are coupled, so that each equation is solved with intermediate values of the other degrees of freedom. The process of solving all the equations in turn and then updating the properties is called a global iteration. Before showing the entire global iteration structure, it is necessary to see how each equation is formed [125].

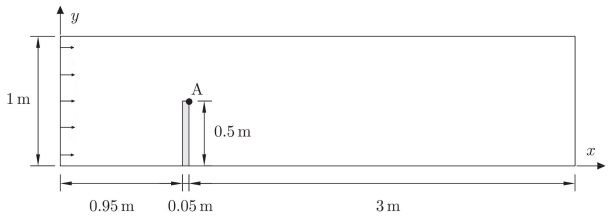

Figure VI.1 – Geometry and boundary conditions of the problem

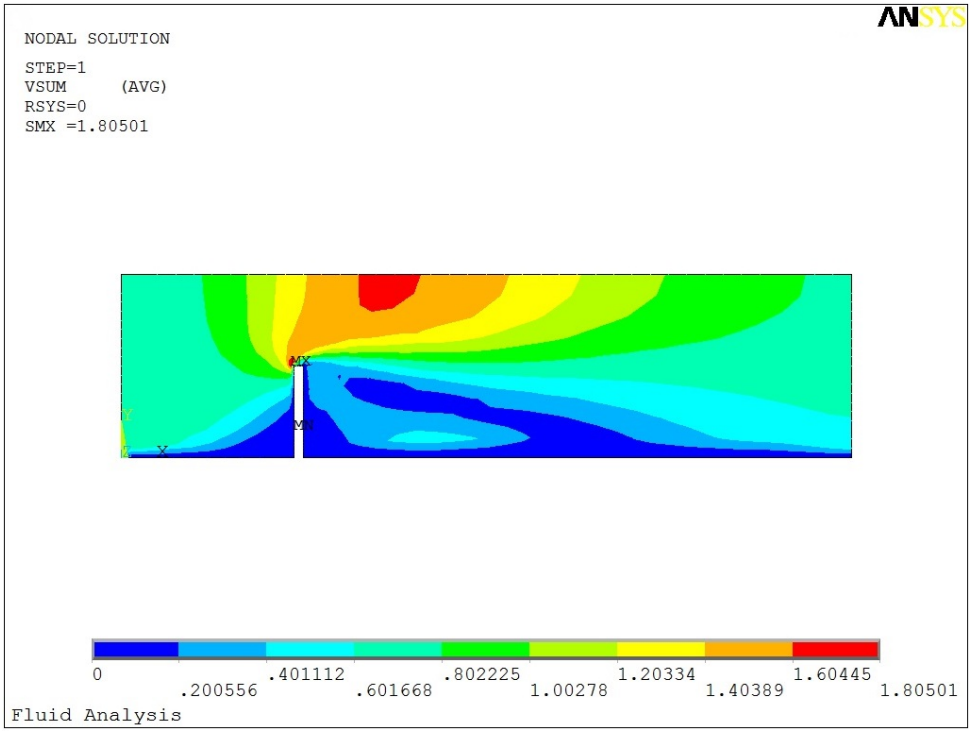

Figure VI.2 – Fluid velocity

FSI optimization

This section illustrates a steady-state fluid-structure interaction problem. This problem demonstrates the use of nonlinear large-deflection structural coupling for a fluid domain.

We consider a 2D elastic plate subjected to forces of a fluid flow in a channel with inlet velocity 0.2 m/s. The geometry of the problem is shown in figure VI.1.

At the outlet, a zero gradient condition is applied. At the walls of the channel and the beam, a no-slip condition is employed. The relevant material parameters for the flow and the structure are: $\rho_f = 1000$ kg/m^3, $\mu_f = 1$ kg/m.s, $E_s = 5000$ N/m^2 et $\nu_s = 0.4$.

The flow domain is discretized with 844 quadrilateral elements and for the plate 114 elements are used.

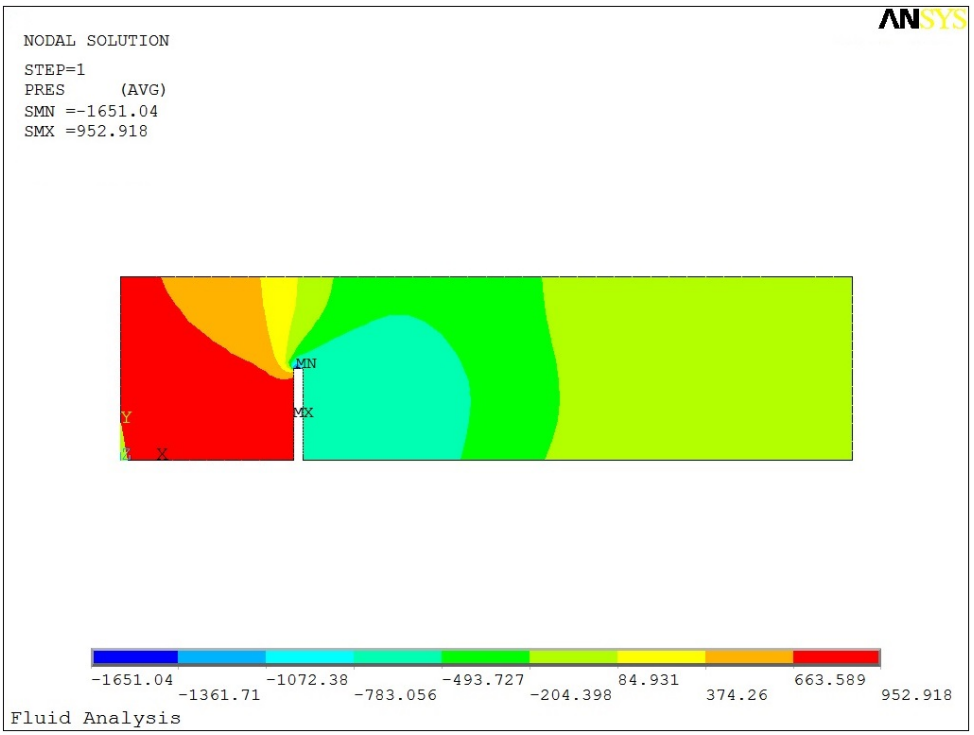

Figure VI.3 – Fluid flow pressure

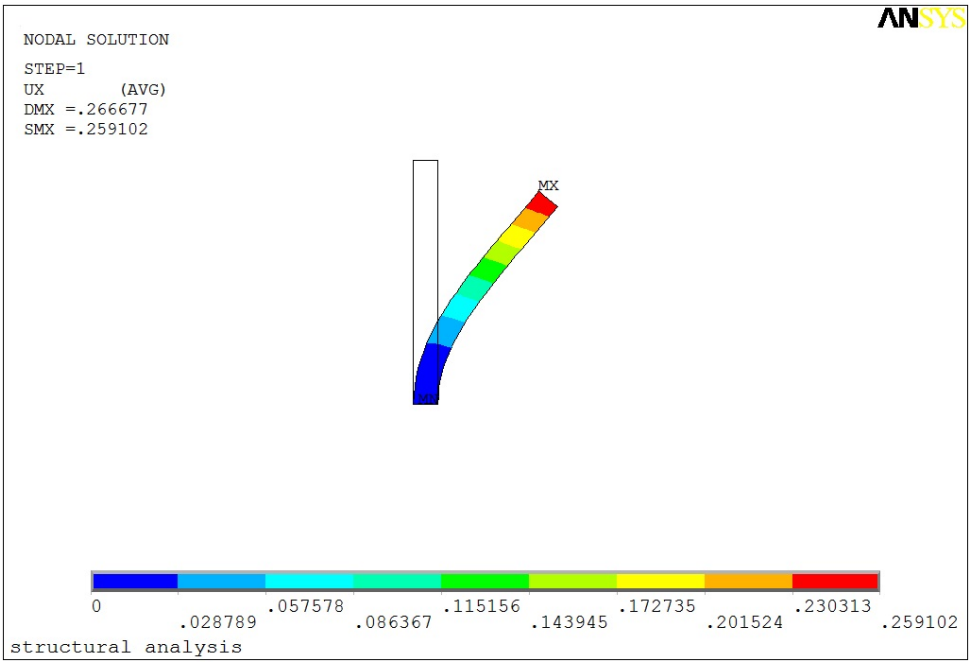

Figure VI.4 – Initial deflection of the plate

The multiple objective for this optimization is to obtain a minimal deflection of the

plate with minimal volume:

$$\min_{d \in D} f(d): \quad \begin{pmatrix} u_x^A \\ V \end{pmatrix} \tag{VI.11}$$

The shape of the plate is defined by two splines, which are connected at the top and bottom sides. The parameterization is done with 5 design variables according to figure VI.5. This allows a horizontal displacement of the points of the plate within the design space defined by:

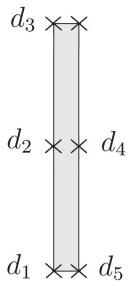

Figure VI.5 – Parametrization of the plate with 5 design variables

$$\begin{cases} 0.85 \leq d_1 \leq 0.97 \\ 0.90 \leq d_2 \leq 0.97 \\ 0.94 \leq d_3 \leq 0.99 \\ 0.98 \leq d_4 \leq 1.05 \\ 0.98 \leq d_5 \leq 1.1 \end{cases} \tag{VI.12}$$

The FSI problem is solved by the non-dominated BSA approach in this part. The Pareto front solutions are generated and updated as the optimization progresses.

Results

The NNIA was used to compare with non-dominated BSA for the FSI problem. The parameter for NNIA included: Population size and Clone population size 20, Antibodies number 20 and Mutation probability $pm = 0.5$, maximum iteration of 5.

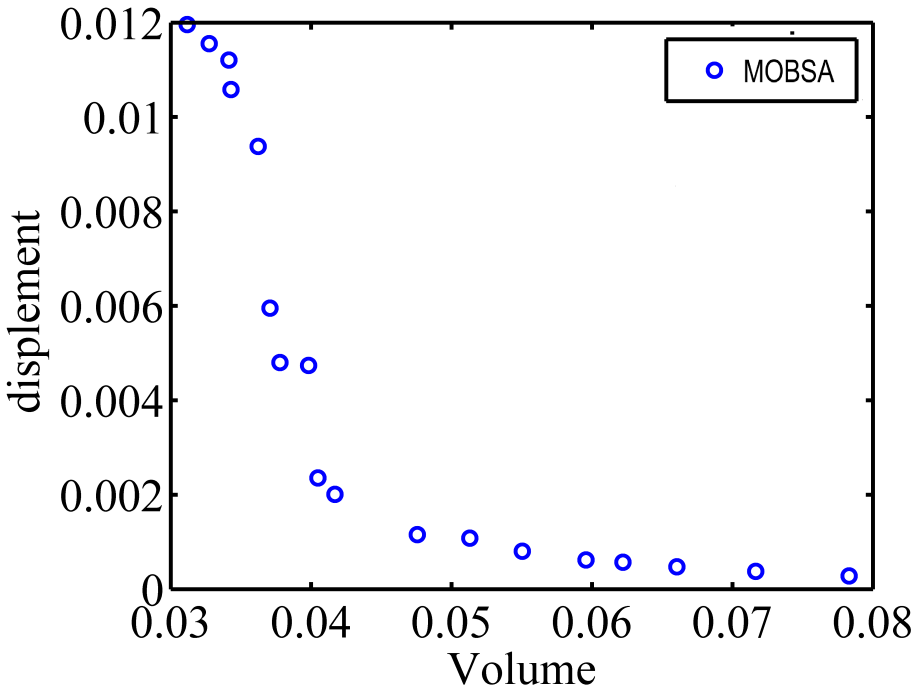

Figure VI.6 – Pareto fronts of FSI optimization

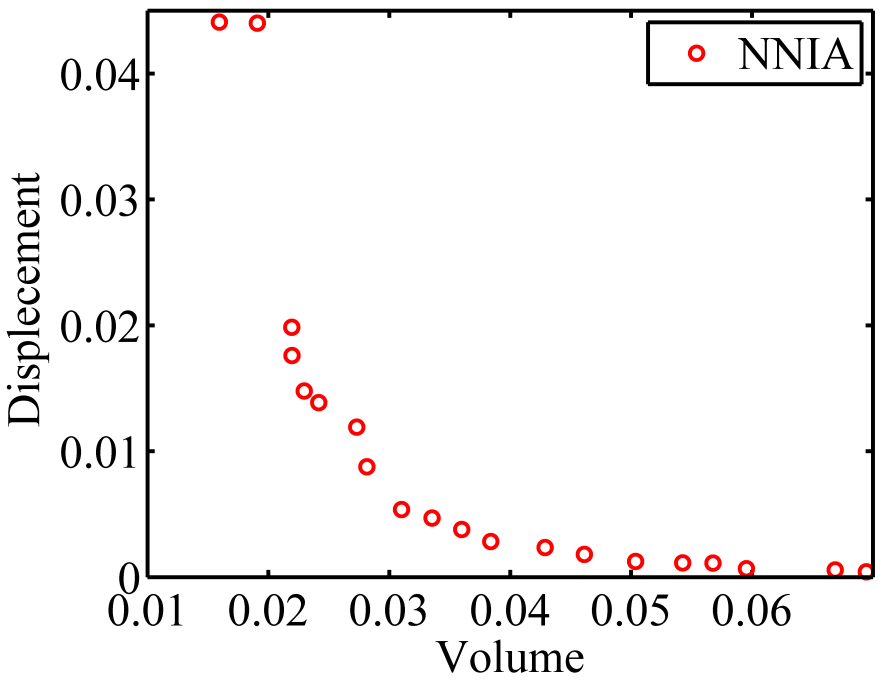

Figure VI.7 – Pareto fronts of FSI optimization using NNIA

Figure VI.6 and VI.7 shows the optimized front obtained using non-dominated BSA algorithm and NNIA respectively. According to Figure VI.6 the non-dominated solutions

obtained by non-dominated BSA with a good distribution and good spread compared with results obtained by algorithm NNIA shown in Figure VI.7.

Conclusion

This paper has introduced BSA, a new evolutionary-computing-based global search algorithm for solve the problem of fluid-structure interaction. BSA's algorithmic structure enables it to benefit from previous generation populations by using solutions it has found in the past for a given problem as it searches for solutions with better fitness values. In comparison the result of algorithm non-dominated BSA's with the NNIA algorithm, the solution sets of Pareto shows our algorithm better efficiency and good distributions. The numerical study is performed using a developed code which couples MATLAB (for the optimization problem) and ANSYS (for the finite element model) to evaluate the calculated objective functions.

Conclusion and perspectives

Conclusion

In this dissertation, we present the development of novel optimization methods for multi-objective optimization and for multi-objective design structure problems. Next to our research motivations, objectives and terminologies introduction (chapter I), we have explained mono and multi-objective optimization methods in chapter II. In addition, in this chapter we have presaged the metaheuristics especially evolutionary algorithms, this type of methods use different types of evaluations and selections, each with its own advantages and disadvantages.

The main results of this thesis are the two approaches that we propose to solve multi-objective optimization problems. The first one converts the Backtracking Search Algorithm (BSA) designed for single-objective problem to an algorithm named **MOBSA** for solving problem of multi-objective optimization. The second one proposes a recombination basically inspired from the one defined for the Backtracking Search Algorithm (BSA) but adaptations are found to fit in the immune algorithm.

In Chapter IV, the first contribution is presented for **MOBSA** based on BSA's operators. The effectiveness of this method is validated by tests functions of various difficulties: unconstraint, constraint, continues and discontinues problems. A comparative study of the obtained results by this approach and other optimization algorithms drawn from the literature shows the effectiveness of this method thanks to the good compromise between the exploitation and the exploration of the field of research and which gives better results in the field. The results obtained also show the robustness of the **MOBSA** and the possibility of finding the Pareto front with a good precision compared to conventional approaches. The **MOBSA** is successfully applied to mechanical structure problems in chapter V for two and three objective functions, where objective functions are contradictory. In the implicit cases we use the finite element analysis to evaluate objective function values. The application of 14 bars was not found in literature for multi-objectives problems. Therefore, we conclude that **MOBSA** offers a potential alternative solution for

solving different multi-objective optimization problems, such as BSA for single-objective optimization problems.

In chapter VI, **MOBSA** is successfully applied to multi-objective optimization of fluid-structure interaction (FSI) problems and compered between NNIA algorithm.

Chapter III, we present our second contribution, which is a hybrid BSA with an Immune algorithm. The proposed algorithm aims to update the solutions found in the optimization phase without restarting optimization. We have tested the proposed NNIA+X algorithm on ten benchmarks problems, **ZDT** and **DZDT**. For the comparison we propose two metrics of qualities based on two other metrics presented in chapter II. These results show that our proposal gives better results for all tested benchmarks. It does not apply either to all the problems of fourteen bars that we have in Chapter V.

perspective

The future research topics may focus on the following:

• Extension of the proposed MOBSA for solving complex problems in the field of optimum structural design for examples 72, 120 bars.

• Robust Multi-objective Backtracking Search Algorithm (MOBSA) will propose for solving multi-objective problems under incertitude.

• Develop a simple method with lower computing complexity to describe the distribution of current PF, called called Multiobjective Optimization using Flower Pollination Algorithm (MOPFA).

• Based for three crossover proposed in Chapter III, we use of information about the evolving environment for finding better searching directions.

"Choose a job you like and you will not have to work a single day of your life."
Confucius

Annexe: The problems of ZDT and CEC2009

Functions test ZDT in reference [1]:

ZDT1:

For ZDT1 with convex Pareto front, it can be seen from in Equation 1:

$$\min_x : \begin{cases} f_1(\mathbf{x}) = \mathbf{x}_1 \\ f_2(\mathbf{x}) = g(\mathbf{x}) \left(1 - \sqrt{\frac{f_1(\mathbf{x})}{g(\mathbf{x})}}\right) \end{cases} \tag{1}$$

where:

$$g(x) = 1 + 9\frac{\sum_{i=2}^{\mathrm{n_x}} \frac{\mathbf{x}_i}{\mathrm{n_x}-1}}{\mathrm{n_x}-1}; \qquad \mathbf{x}_i \in [0,1]; \qquad n_x = 30;$$

ZDT2

For ZDT2 with nonconvex Pareto front, it can be seen from in Equation 2:

$$\min_x : \begin{cases} f_1(\mathbf{x}) = \mathbf{x}_1 \\ f_2(\mathbf{x}) = g(\mathbf{x}) \left(1 - \frac{f_1(\mathbf{x})}{g(\mathbf{x})}\right)^2 \end{cases} \tag{2}$$

where:

$$g(x) = 1 + 9\frac{\sum_{i=2}^{\mathrm{n_x}} \frac{\mathbf{x}_i}{\mathrm{n_x}-1}}{\mathrm{n_x}-1}; \qquad \mathbf{x}_i \in [0,1]; \qquad n_x = 30;$$

ZDT3

For ZDT3 with disconnected Pareto front, it can be seen from in Equation 3:

$$\min_x : \begin{cases} f_1(\mathbf{x}) = \mathbf{x}_1 \\ f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - \sqrt{\frac{f_1(\mathbf{x})}{g(\mathbf{x})}} - \frac{f_1(\mathbf{x})}{g(\mathbf{x})} \sin(10\pi f_1(\mathbf{x}))\right] \end{cases} \tag{3}$$

where:

$$g(x) = 1 + 9\frac{\sum_{i=2}^{\mathrm{n_x}} \frac{\mathbf{x}_i}{\mathrm{n_x}-1}}{\mathrm{n_x}-1}; \qquad \mathbf{x}_i \in [0,1]; \qquad n_x = 30;$$

ZDT6

For ZDT6 with nonconvex Pareto front, it can be seen from in Equation 4:

$$\min_x : \begin{cases} f_1(\mathbf{x}) = 1 - \mathbf{exp}(4\mathbf{x}_1 \sin(6\mathbf{x}_1\pi)) \\ f_2(\mathbf{x}) = g(\mathbf{x})\left(1 - \left(\frac{f_1(\mathbf{x})}{g(\mathbf{x})}\right)^2\right) \end{cases} \tag{4}$$

where:

$$g(x) = 1 + 9\frac{\sum_{i=2}^{\mathrm{n_x}} \frac{\mathbf{x}_i}{\mathrm{n_x}-1}}{\mathrm{n_x}-1}; \qquad \mathbf{x}_i \in [0,1]; \qquad n_x = 10;$$

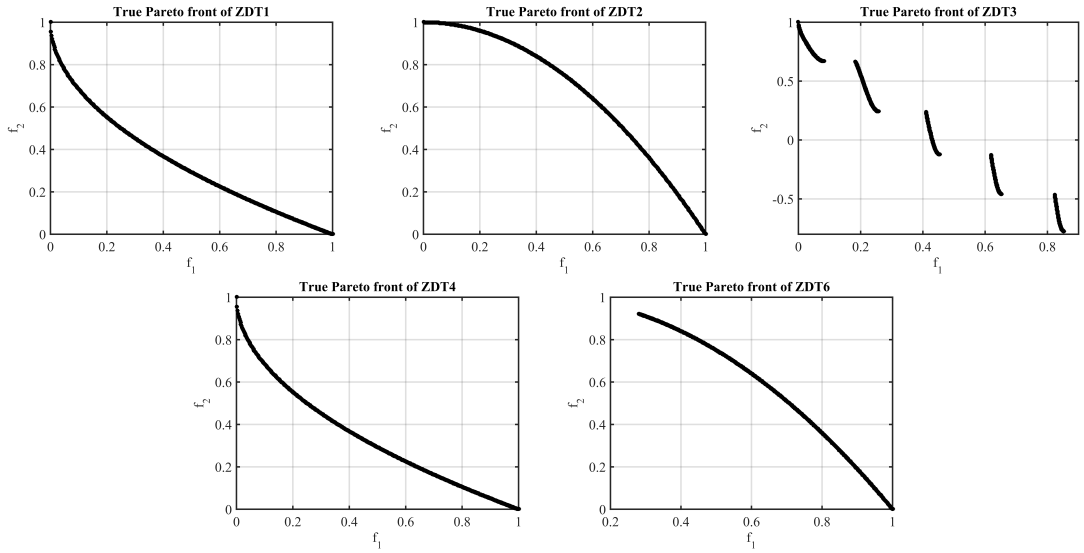

Figure A1.1 – Pareto-front for ZDT problems.

Function test CEC 2009 in reference [2]:

Unconstrained multi-objective test problems

- **Unconstrained Problem UF1:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \frac{2}{J_1} \sum\limits_{j \in J_1} [x_j - \sin(6\pi x_1 + \frac{j\pi}{n})]^2 \\ f_2(\mathbf{x}) = 1 - \sqrt{x_1} + \frac{2}{J_2} \sum\limits_{j \in J_2} [x_j - \sin(6\pi x_1 + \frac{j\pi}{n})]^2 \end{cases} \tag{5}$$

where:

$$J_1 = \{j | \text{j is odd and} \quad 2 \leq j \leq n\} \qquad \text{and} \qquad J_2 = \{j | \text{j is even and} \quad 2 \leq j \leq n\};$$

The search space is $[0,1] \times [-1,1]^{n-1}$.

- **Unconstrained Problem UF2:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \frac{2}{J_1} \sum\limits_{j \in J_1} y_j^2 \\ f_2(\mathbf{x}) = 1 - \sqrt{x_1} + \frac{2}{J_2} \sum\limits_{j \in J_2} y_j^2 \end{cases} \tag{6}$$

where:

$$J_1 = \{j | \text{j is odd and} \quad 2 \leq j \leq n\} \qquad \text{and} \qquad J_2 = \{j | \text{j is even and} \quad 2 \leq j \leq n\};$$

The search space is $[0,1] \times [-1,1]^{n-1}$.

$$y_j : \begin{cases} x_j - [0.3x_1^2 \cos(24\pi x_1 + \frac{4j\pi}{n}) 0.6x_1] \cos(6\pi x_1 + \frac{j\pi}{n}) & j \in J_1 \\ x_j - [0.3x_1^2 \cos(24\pi x_1 + \frac{4j\pi}{n}) 0.6x_1] \sin(6\pi x_1 + \frac{j\pi}{n}) & j \in J_2 \end{cases} \tag{7}$$

- **Unconstrained Problem UF3:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}(4 \sum\limits_{j \in J_1} y_j^2 - 2 \prod\limits_{j \in J_1} \cos(\frac{20 y_j \pi}{\sqrt{j}}) + 2) \\ f_2(\mathbf{x}) = 1 - \sqrt{x_1} + \frac{2}{|J_2|}(4 \sum\limits_{j \in J_2} y_j^2 - 2 \prod\limits_{j \in J_2} \cos(\frac{20 y_j \pi}{\sqrt{j}}) + 2) \end{cases} \tag{8}$$

where:

$$J_1 = \{j | \text{j is odd and} \quad 2 \leq j \leq n\} \qquad \text{and} \qquad J_2 = \{j | \text{j is even and} \quad 2 \leq j \leq n\};$$

$$y_j = x_j - x_1^{0.5(1+3\frac{j-2}{n-2})}, \quad j = 2, ..., n,$$

The search space is $[0,1] \times [-1,1]^{n-1}$.

- **Unconstrained Problem UF4:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}(4 \sum\limits_{j \in J_1} h(y_j) \\ f_2(\mathbf{x}) = 1 - x_1^2 + \frac{2}{|J_2|}(4 \sum\limits_{j \in J_2} h(y_j) \end{cases} \tag{9}$$

where:

$$J_1 = \{j | \text{j is odd and} \quad 2 \le j \le n\} \qquad \text{and} \qquad J_2 = \{j | \text{j is even and} \quad 2 \le j \le n\};$$

$$y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}), \quad j = 2, ..., n, \quad h(t) = \frac{|t|}{1 + \exp^{2|t|}}$$

The search space is $[0,1] \times [-2,2]^{n-1}$.

- **Unconstrained Problem UF5:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + (\frac{1}{2N} + \epsilon)|\sin(2N\pi x_1)| + \frac{2}{|J_1|} \sum\limits_{j \in J_1} h(y_j) \\ f_2(\mathbf{x}) = 1 - x_1 + (\frac{1}{2N} + \epsilon)|\sin(2N\pi x_1)| + \frac{2}{|J_2|} \sum\limits_{j \in J_2} h(y_j) \end{cases} \tag{10}$$

where:

$$J_1 = \{j | \text{j is odd and} \quad 2 \le j \le n\} \qquad \text{and} \qquad J_2 = \{j | \text{j is even and} \quad 2 \le j \le n\}; \quad N \text{is an integer}, \epsilon$$

$$y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}), \quad j = 2, ..., n, \quad h(t) = 2t^2 - \cos(4\pi t) + 1$$

The search space is $[0,1] \times [-1,1]^{n-1}$.

- **Unconstrained Problem UF6:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 \quad + \max\{0, 2(\frac{1}{2N} + \epsilon)|\sin(2N\pi x_1)|\} + \frac{2}{|J_1|}(4 \sum\limits_{j \in J_1} (y_j)^2 - 2 \prod\limits_{j \in J_1} \cos(\frac{20 y_j \pi}{\sqrt{j}}) + 2) \\ f_2(\mathbf{x}) = 1 - x_1 + \max\{0, 2(\frac{1}{2N} + \epsilon)|\sin(2N\pi x_1)|\} + \frac{2}{|J_2|}(4 \sum\limits_{j \in J_2} (y_j)^2 - 2 \prod\limits_{j \in J_1} \cos(\frac{20 y_j \pi}{\sqrt{j}}) + 2) \end{cases} \tag{11}$$

where:

$$J_1 = \{j | \text{j is odd and} \quad 2 \le j \le n\} \qquad \text{and} \qquad J_2 = \{j | \text{j is even and} \quad 2 \le j \le n\}; \quad N \text{is an integer}, \epsilon$$

$$y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}), \quad j = 2, ..., n, \quad h(t) = 2t^2 - \cos(4\pi t) + 1$$

The search space is $[0,1] \times [-1,1]^{n-1}$.

- **Unconstrained Problem UF7:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = \sqrt[5]{x_1} + \sum\limits_{j \in J_1} y_j^2 \\ f_2(\mathbf{x}) = 1 - \sqrt[5]{x_1} + \sum\limits_{j \in J_2} y_j^2 \end{cases} \tag{12}$$

where:

$$J_1 = \{j | \text{j is odd and} \quad 2 \le j \le n\} \qquad \text{and} \qquad J_2 = \{j | \text{j is even and} \quad 2 \le j \le n\}$$

$$y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}), \quad j = 2, ..., n,$$

The search space is $[0,1] \times [-1,1]^{n-1}$.

- **Unconstrained Problem UF8:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{|J_1|} \sum\limits_{j \in J_1} y_j^2 \\ f_2(\mathbf{x}) = \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{|J_2|} \sum\limits_{j \in J_2} y_j^2 \\ f_3(\mathbf{x}) = \sin(0.5x_2\pi) + \frac{2}{|J_3|} \sum\limits_{j \in J_2} y_j^2 \end{cases} \tag{13}$$

where:

$\mathtt{J}_1 = \{j | 3 \le j \le n, \text{and} j - 1 \quad \text{is a multiplication of 3}\}$,
$\mathtt{J}_2 = \{j | 3 \le j \le n, \text{and} j - 1 \quad \text{is a multiplication of 3}\}$,
$\mathtt{J}_1 = \{j | 3 \le j \le n, \text{and} j - 1 \quad \text{is a multiplication of 3}\}$,

$$y_j = x_j - x_2 \sin(6\pi x_1 + \frac{j\pi}{n});$$

The search space is $[0,1]^2 \times [-2,2]^{n-2}$.

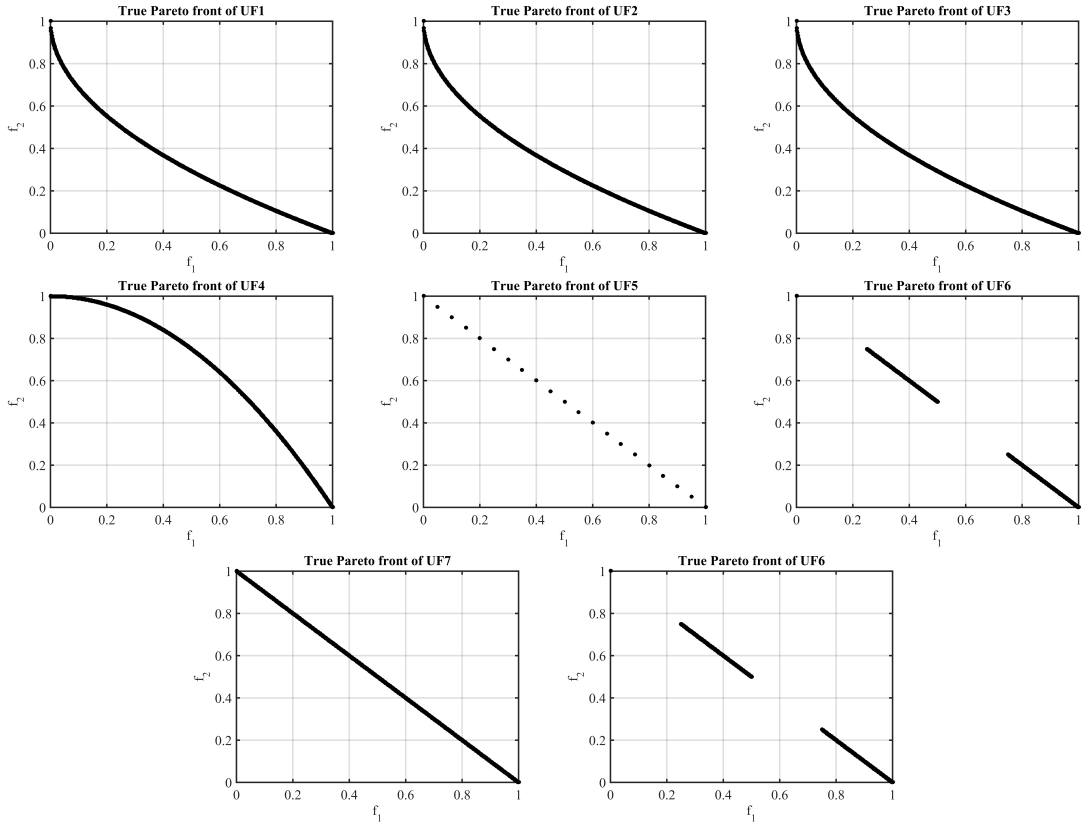

Figure A1.2 – Pareto-front for CF series problems.

Algorithm	Statistic	UF1	UF2	UF3	UF4	UF5	UF6	UF7	UF8
MOBSA	Mean	**1.61E-03**	**1.46E-03**	9.57E-03	1.17E-03	8.21E-02	**1.04E-02**	**5.36E-04**	**5.54E-04**
	SD	5.92E-04	4.03E-04	2.29E-03	2.28E-05	5.60E-03	5.22E-04	1.78E-04	1.24E-04
MTS	Mean	6.46E-03	6.15E-03	5.31E-02	2.36E-02	**1.49E-02**	5.92E-02	4.08E-02	1.13E-01
	SD	3.49E-04	5.08E-04	1.17E-02	6.64E-04	3.28E-03	1.06E-02	1.44E-02	1.29E-02
DMOEADD	Mean	1.04E-02	6.79E-03	3.34E-02	4.27E-02	3.15E-01	6.67E-02	1.03E-02	6.84E-02
	SD	2.37E-03	2.02E-03	5.68E-03	1.39E-03	4.66E-02	1.03E-02	9.46E-03	9.12E-03
MOABC	Mean	6.18E-03	4.84E-03	5.12E-02	5.80E-02	7.78E-02	6.54E-02	5.57E-02	6.73E-02
	SD	-	-	-	-	-	-	-	-
LiuLi	Mean	7.85E-03	1.23E-02	1.50E-02	4.35E-02	1.62E-01	1.76E-01	7.30E-03	8.24E-02
	SD	2.09E-03	3.32E-03	2.40E-02	6.50E-04	2.82E-02	8.29E-02	8.90E-04	7.33E-03
MOEADGM	Mean	6.20E-03	6.40E-03	4.29E-02	4.76E-02	1.79E+00	5.56E-01	7.60E-03	2.45E-01
	SD	1.13E-03	4.30E-04	3.41E-02	2.22E-03	5.12E-01	1.47E-01	9.40E-04	8.54E-02
GDE3	Mean	5.34E-03	1.20E-02	1.06E-01	2.65E-02	3.93E-02	2.51E-01	2.52E-02	2.49E-01
	SD	3.42E-04	1.54E-03	1.29E-02	3.72E-04	3.95E-03	1.96E-02	8.89E-03	3.55E-02
DECMOSA-SQP	Mean	7.70E-02	2.83E-02	9.35E-02	3.39E-02	1.67E-01	1.26E-01	2.42E-02	2.16E-01
	SD	3.94E-02	3.13E-02	1.98E-01	5.37E-03	8.95E-02	5.62E-01	2.23E-02	1.21E-01
Clustering MOEA	Mean	2.99E-02	2.28E-02	5.49E-02	5.85E-02	2.47E-01	8.71E-02	2.23E-02	2.38E-01
	SD	3.30E-03	2.30E-03	1.47E-02	2.70E-03	3.84E-02	5.70E-03	2.00E-03	2.30E-02
AMGA	Mean	3.59E-02	1.62E-02	7.00E-02	4.06E-02	9.41E-02	1.29E-01	5.71E-02	1.71E-01
	SD	1.03E-02	3.17E-03	1.40E-02	1.75E-03	1.21E-02	5.66E-02	6.53E-02	1.72E-02
OMOEAII	Mean	8.56E-02	3.06E-02	2.71E-01	4.62E-02	1.69E-01	7.34E-02	3.35E-02	1.92E-01
	SD	4.07E-03	1.61E-03	3.76E-02	9.67E-04	3.90E-03	2.45E-03	1.74E-03	1.23E-02
OW MOSaDE	Mean	1.22E-02	8.10E-03	1.03E-01	5.13E-02	4.30E-01	1.92E-01	5.85E-02	9.45E-02
	SD	1.20E-03	2.30E-03	1.90E-02	1.90E-03	1.74E-02	2.90E-02	2.91E-02	1.19E-02
MOEP	Mean	5.96E-02	1.89E-02	9.90E-02	4.27E-02	2.25E-01	1.03E-01	1.97E-02	4.23E-01
	SD	1.28E-02	3.80E-03	1.32E-02	8.35E-04	3.44E-02	3.45E-02	7.51E-04	5.65E-02
NSGAIILS	Mean	1.15E-02	1.24E-02	1.06E-01	5.84E-02	5.66E-01	3.10E-01	2.13E-02	8.63E-02
	SD	7.30E-03	9.11E-03	6.86E-02	5.12E-03	1.83E-01	1.91E-01	1.95E-02	1.24E-02

Tableau A1.1 – Mean and Standard Deviation of the IGD metric results for the unconstrained benchmark problems

Constrained multi-objective test problems

- **Constrained Problem CF1:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \sum\limits_{j\in J_1}(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})) \\ f_2(\mathbf{x}) = 1 - x_1 + \sum\limits_{j\in J_2}(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})) \\ \text{Subject to:} \\ \frac{t}{1+e^{4\lvert t\rvert}} \geq 0; t = f_1 + f_2 - a\lvert\sin[N\pi(f_1 - f_2 + 1)]\rvert - 1. \end{cases} \tag{14}$$

where:

$$J_1 = \{j|\text{j is odd and} \quad 2 \leq j \leq n\} \qquad \text{and} \qquad J_2 = \{j|\text{j is even and} \quad 2 \leq j \leq n\}$$

where N is an integer and $a \geq \frac{1}{2N}$
The search space is $[0,1]^n$.

- **Constrained Problem CF2:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \sum\limits_{j\in J_1}(x_j - x_1^{2(1+\frac{3(j-2)}{n-2})})^2 \\ f_2(\mathbf{x}) = 1 - x_1 + \sum\limits_{j\in J_2}(x_j - x_1^{2(1+\frac{3(j-2)}{n-2})})^2 \\ \text{Subject to:} \\ f_1 + f_2 - a\lvert\sin[N\pi(f_1 - f_2 + 1)]\rvert - 1 \geq 0 \end{cases} \tag{15}$$

where:

$$J_1 = \{j|\text{j is odd and} \quad 2 \leq j \leq n\} \qquad \text{and} \qquad J_2 = \{j|\text{j is even and} \quad 2 \leq j \leq n\}$$

where N is an integer and $a \geq \frac{1}{2N}$
The search space is $[0,1]^n$.

- **Constrained Problem CF3:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \frac{2}{\lvert J_1\rvert}(4\sum\limits_{j\in J_1} y_j^2 - 2\prod\limits_{j\in J_1}\cos(\frac{20y_j\pi}{\sqrt{j}}) + 2) \\ f_2(\mathbf{x}) = 1 - \sqrt{x_1} + \frac{2}{\lvert J_2\rvert}(4\sum\limits_{j\in J_2} y_j^2 - 2\prod\limits_{j\in J_2}\cos(\frac{20y_j\pi}{\sqrt{j}}) + 2) \\ \text{Subject to:} \\ f_1^2 + f_2 - a\lvert\sin[N\pi(f_1^2 - f_2 + 1)]\rvert - 1 \geq 0. \end{cases} \tag{16}$$

where:

$$J_1 = \{j|\text{j is odd and} \quad 2 \le j \le n\} \qquad \text{and} \qquad J_2 = \{j|\text{j is even and} \quad 2 \le j \le n\};$$

$$y_j = x_j - x_1^{0.5(1+3\frac{j-2}{n-2})}, \quad j = 2, ..., n,$$

The search space is $[0,1] \times [-2,2]^{n-1}$.

- **Constrained Problem CF4:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|} \sum\limits_{j \in J_1} h(y_j) \\ f_2(\mathbf{x}) = 1 - x_1^2 + \frac{2}{|J_2|} 4 \sum\limits_{j \in J_2} h(y_j) \\ \text{Subject to:} \\ \frac{t}{1+e^{4|t|}} \ge 0; \end{cases} \tag{17}$$

where:

$$J_1 = \{j|\text{j is odd and} \quad 2 \le j \le n\} \qquad \text{and} \qquad J_2 = \{j|\text{j is even and} \quad 2 \le j \le n\};$$

$$y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}), \quad j = 2, ..., n, \quad t = x_2 - \sin(6\pi x_1 + \frac{j\pi}{n}) - 0.5x_1 + 0.25,$$

The search space is $[0,1] \times [-2,2]^{n-1}$.

$$h_2(t) = \begin{cases} |t| & \text{if} \quad t < \frac{3}{2}(1 - \frac{\sqrt{2}}{2}) \\ 0.125 + (t-1)^2 & \text{otherwise} \end{cases}$$

and

$$h_j(t) = t^2, \quad \text{for} \quad j = 2, ..., n,$$

- **Constrained Problem CF5:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|} \sum\limits_{j \in J_1} h(y_j) \\ f_2(\mathbf{x}) = 1 - x_1^2 + \frac{2}{|J_2|} \sum\limits_{j \in J_2} h(y_j) \\ \text{Subject to:} \\ x_2 - \sin(6\pi x_1 + \frac{j\pi}{n}) - 0.5x_1 + 0.25 \ge 0; \end{cases} \tag{18}$$

where:

$$J_1 = \{j|\text{j is odd and} \quad 2 \le j \le n\} \qquad \text{and} \qquad J_2 = \{j|\text{j is even and} \quad 2 \le j \le n\};$$

$$y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}), \quad j = 2, ..., n, \quad t = x_2 - \sin(6\pi x_1 + \frac{j\pi}{n}) - 0.5x_1 + 0.25,$$

The search space is $[0,1] \times [-2,2]^{n-1}$.

$$y_j = \begin{cases} x_j - 0.8x_1 \cos(6\pi x_1 + \frac{j\pi}{n}) & if \quad j \in J_1 \\ x_j - 0.8x_1 \sin(6\pi x_1 + \frac{j\pi}{n}) & if \quad j \in J_2 \end{cases}$$

$$h_2(t) = \begin{cases} |t| & \text{if} \quad t < \frac{3}{2}(1 - \frac{\sqrt{2}}{2}) \\ 0.125 + (t-1)^2 & \text{otherwise} \end{cases}$$

and

$$h_j(t) = t^2 - \cos(4\pi t) + 1, \quad \text{for} \quad j = 2, ..., n,$$

- **Constrained Problem CF6:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|} \sum\limits_{j \in J_1} (y_j)^2 \\ f_2(\mathbf{x}) = (1 - x_1)^2 + \frac{2}{|J_2|} \sum\limits_{j \in J_2} (y_j)^2 \\ \text{Subject to:} \\ x_2 - \sin(6\pi x_1 + \frac{2\pi}{n}) - sing(0.5(1-x_1) - (1-x_1)^2)\sqrt{|0.5(1-x_1) - (1-x_1)^2|} \geq 0; \\ x_4 - \sin(6\pi x_1 + \frac{4\pi}{n}) - sing(0.5\sqrt{1-x_1} - (1-x_1))\sqrt{|0.5\sqrt{1-x_1} - (1-x_1)|} \geq 0; \end{cases} \tag{19}$$

where:

$$J_1 = \{j | \text{j is odd and} \quad 2 \leq j \leq n\} \qquad \text{and} \qquad J_2 = \{j | \text{j is even and} \quad 2 \leq j \leq n\};$$

The search space is $[0,1] \times [-2,2]^{n-1}$.

$$y_j = \begin{cases} x_j - 0.8x_1 \cos(6\pi x_1 + \frac{j\pi}{n}) & \text{if} \quad j \in J_1 \\ x_j - 0.8x_1 \sin(6\pi x_1 + \frac{j\pi}{n}) & \text{if} \quad j \in J_2 \end{cases}$$

- **Constrained Problem CF7:**

$$\min_x : \begin{cases} f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1} h(y_j) \\ f_2(\mathbf{x}) = (1-x_1)^2 + \frac{2}{|J_2|}\sum_{j\in J_2} h(y_j) \\ \text{Subject to:} \\ x_2 - \sin(6\pi x_1 + \frac{j\pi}{n}) - sing(0.5(1-x_1) - (1-x_1)^2)\sqrt{|0.5(1-x_1) - (1-x_1)^2|} \geq 0; \\ x_2 - \sin(6\pi x_1 + \frac{j\pi}{n}) - sing(0.5\sqrt{1-x_1} - (1-x_1))\sqrt{|0.5\sqrt{1-x_1} - (1-x_1)|} \geq 0; \end{cases} \tag{20}$$

where:

$$J_1 = \{j|\text{j is odd and} \quad 2 \leq j \leq n\} \qquad \text{and} \qquad J_2 = \{j|\text{j is even and} \quad 2 \leq j \leq n\};$$

$$y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}), \quad j = 2, ..., n, \quad t = x_2 - \sin(6\pi x_1 + \frac{j\pi}{n}) - 0.5x_1 + 0.25,$$

The search space is $[0,1] \times [-2,2]^{n-1}$.

$$y_j = \begin{cases} x_j - 0.8x_1 \cos(6\pi x_1 + \frac{j\pi}{n}) & \text{if} \quad j \in J_1 \\ x_j - 0.8x_1 \sin(6\pi x_1 + \frac{j\pi}{n}) & \text{if} \quad j \in J_2 \end{cases}$$

$$h_2(t) = h_4(t) = t^2$$

and

$$h_j(t) = t^2 - \cos(4\pi t) + 1, \quad \text{for} \quad j = 3, 5, ..., n,$$

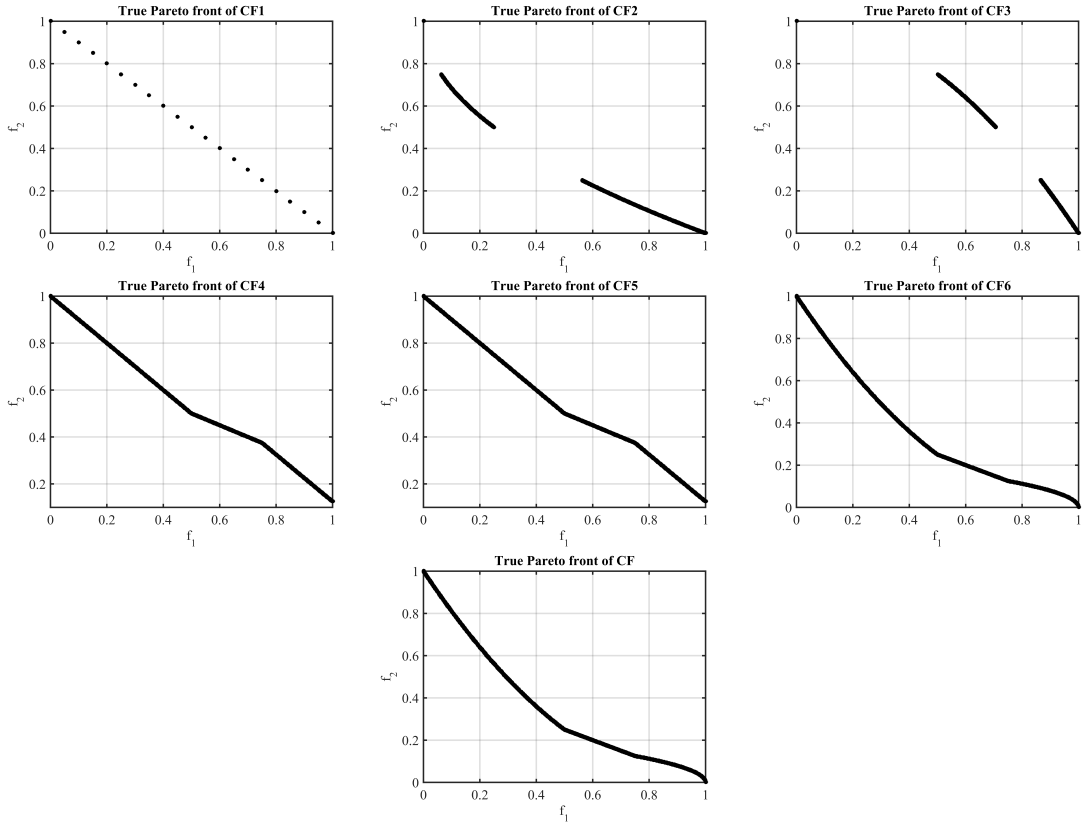

Figure A1.3 – Pareto-front for CF series problems.

Algorithm	Statistic	CF1	CF2	CF3	CF4	CF5	CF6	CF7
MOBSA	Mean	**1.31E-05**	3.39E-03	**3.01E-03**	**1.67E-03**	4.93e-03	**1.90E-03**	**4.97E-03**
	SD	1.22E-06	1.22e-03	1.47E-04	1.29e-04	1.32e-04	3.65E-05	5.247E-04
DMOEADD	Mean	1.13E-02	**2.10E-03**	5.63E-02	6.99E-03	1.58E-02	1.50E-02	1.91E-02
	SD	2.76E-03	4.53E-04	7.57E-03	1.46E-03	6.66E-03	6.46E-03	6.12E-03
MOABC	Mean	9.92E-03	1.03E-02	8.62E-02	4.52E-03	6.78E-02	4.83E-03	1.69E-02
	SD	-	-	-	-	-	-	-
LiuLi Algorithm	Mean	8.50E-04	4.20E-03	1.83E-01	1.42E-02	1.10E-01	1.39E-02	1.04E-01
	SD	1.10E-04	2.64E-03	4.21E-02	3.29E-03	3.07E-02	2.59E-03	3.51E-02
MTS	Mean	1.92E-02	2.68E-02	1.04E-01	1.11E-02	2.08E-02	1.62E-02	2.47E-02
	SD	2.57E-03	1.47E-02	1.56E-02	1.37E-03			
MOEADGM	Mean	1.08E-02	8.00E-03	5.13E-01	7.07E-02	5.45E-01	2.07E-01	5.36E-01
	SD	2.50E-03	9.99E-03	7.14E-02	1.01E-01	1.72E-01	1.00E-04	1.00E-01
DECMOSA-SQP	Mean	1.08E-01	9.46E-02	1.00E+06	1.53E-01	4.13E-01	1.48E-01	2.60E-01
	SD	1.96E-01	2.94E-01	0.00E+00	4.67E-01	5.91E-01	1.25E-01	2.60E-01

Tableau A1.2 – Mean and Standard Deviation of the IGD metric results for the constrained benchmark problems

References bibliographiques

[1] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems; technical report 112. Technical report Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland. (2001). ix, 3, 10, 20, 39, 42, 68, 109

[2] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari. *Multiobjective optimization test instances for the CEC 2009 special session and competition.* Working Report, CES-887, School of Computer Science and Electrical Engineering, University of Essex (2008). ix, 2, 3, 68, 69, 111, 113, 115, 117, 119, 121

[3] Wafae EL ALEM. *Contribution à l'optimisation globale robuste & multi-objectif associée à la modélisation numérique en mécanique des structures.* Thèse de Doctorat, Thèse de doctorat en cotutelle : l'Ecole Mohammadia d'Ingénieurs & l'Institut National des Sciences Appliquées de Rouen (2012). 1, 79

[4] F. Glover and G. A. Kochenberger. *Handbook of metaheuristics.* Springer (2003). 2, 8

[5] G. R. Zavala, A. J. Nebro, F. Luna, and C. A. Coello C. *Structural design using multi-objective metaheuristics. comparative study and application to a real-world problem.* Structural and Multidisciplinary Optimization **53**(3), 545–566 March (2016). 2, 17

[6] Gustavo R. Zavala, Antonio J. Nebro, Juan J. Durillo, and Francisco Luna. *Integrating a multi-objective optimization framework into a structural design software.* Advances in Engineering Software **76**, 161–170 (2014). 2, 19

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. *Nsga-ii: A fast and elitist multi-objective genetic algorithm.* IEEE Transactions on Evolutionary Computation **6(2)**, 182–197 (2002). 2, 12, 22, 23, 33, 34, 36, 62, 67, 68, 96

[8] E. Zitzler, M. Laumanns, and L. Thiele. *Spea2: improving the strength pareto evolutionary algorithm, a in evolutionary methods for design.* Optimization

and Control with Applications to Industrial Problems, Athens, Greece pages 95–100 (2002). 2, 22, 24, 33, 62

[9] E. Zitzler, K. Deb, and L. Thiele. *Comparison of multiobjective evolutionary algorithms: Empirical results.* IEEE Transactions on Evolutionary Computation pages 173–195 (2000). 3, 39

[10] A. Kaveh. *Advances in Metaheuristic Algorithms for Optimal Design of Structures.* Second Edition, SpringerLink : Bücher. Springer International Publishing (2014). 8

[11] I. H. Osman and G. Laporte. *Metaheuristics: A bibliography.* Annals of Operations Research **63**, 513–623 October (1996). 8, 17

[12] X. S. Yang. *Nature-inspired Metaheuristic Algorithms.* Luniver Press (2010). 8

[13] A. E Hami and B. Radi. *Incertitudes, optimisation et fiabilité des structures.* Hermes Science Publications (2013). 8

[14] T. Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms.* Oxford University Press, Oxford, UK (1996). 9

[15] C. A. C. Coello and G. B. Lamont D. A. V. Veldhuizen. *Evolutionary algorithms for solving multi-objective problems, 2nd edition.* Genetic and Evolutionary Computation. Springer US. (2007). 9, 10, 13, 18, 25

[16] Kalyanmoy Deb. *Introduction to Evolutionary Multiobjective Optimization* pages 59–96. Springer Berlin Heidelberg, Berlin, Heidelberg (2008). 10, 19, 22, 80

[17] A. R. Pérez. *Surrogate-assisted evolutionary multi-objective full model selection.* Thèse de Doctorat, Instituto Nacional de Astrofísica, Óptica y Electrónica (2016). 10, 25

[18] V. Pareto. *Cours d'economie politique.* Librairie Droz (1964). 11

[19] Z. Michalewicz and M. Schoenauer. *Evolutionary algorithms for constrained parameter optimization problems.* Evolutionary Computation **4**(1), 1–32 (1996). 12

[20] C. A. C. Coello. *Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art.* Computer Methods in Applied Mechanics and Engineering **191**(11), 1245–1287 (2002). 12

[21] R. Courant. *Variational methods for the solution of problems of equilibrium and vibrations.* Bulletin of the American Mathematical Sociaty **49**(1), 1–23 (1946). 12

[22] C-L. Hwang and A. S. M. Masud. , **62**. Lecture Notes in Economics and Mathematical Systems, Springer Science & Business Media (2012). 13

[23] S. Bandyopadhyay and S. Saha. *Some Single- and Multiobjective Optimization Techniques* pages 17–58. Springer Berlin Heidelberg, Berlin, Heidelberg (2013). 14

[24] S.K. Pal S. Bandyopadhyay. *Classification and Learning Using Genetic Algorithms: Applications in Bioinformatics and Web Intelligence*. Natural Computing Series. Springer Berlin Heidelberg (2007). 14

[25] Y.B. Bard. *Nonlinear Parameter Estimation*. Academic Press, New York. (1974). 14

[26] R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, New York (2000). 14

[27] Michel Minoux. *Programmation mathématique: théorie et algorithmes*. Dunod, Paris (1983). 15

[28] C.T. Kelley. *Iterative Methods for Optimization*. Society for Industrial and Applied Mathematics (1999). 16

[29] C.T. Kelley. *Solving Nonlinear Equations with Newton's Method*. Society for Industrial and Applied Mathematics (2003). 16

[30] S. Kirkpatrick, C. D. Gelat, and M. P. Vecchi. *Optimization by simulated annealing*. Science **220**(4598), 671–680 (1983). 16, 17

[31] M. Gendreau and J-Y. Potvin. *Handbook of metaheuristics*. Springer,Second Edition (2010). 17

[32] G. R. Zavala, A. J. Nebro, F. Luna, and C. A. C. Coello. *A survey of multi-objective metaheuristics applied to structural optimization*. Structural and Multidisciplinary Optimization **49(4)**, 537–558 (2014). 17, 47, 80, 84, 88

[33] Mario. Villalobos-Arias, C. A. C. Coello, and O. Hernández-Lerma. Asymptotic convergence of some metaheuristics used for multiobjective optimization. In A. H. Wright, M. D. Vose, K. A. De Jong, and L. M. Schmitt, editors, *Foundations of Genetic Algorithms*, pages 95–111, Berlin, Heidelberg (2005). Springer Berlin Heidelberg. 18, 19

[34] Fred Glover. *Future paths for integer programming and links to artificial intelligence*. Computers & Operations Research **13**(5), 533 – 549 (1986). Applications of Integer Programming. 18

[35] Michel Gendreau. *An Introduction to Tabu Search* pages 37–54. Springer US, Boston, MA (2003). 18

[36] M. Gendreau and J.Y. Potvin. *Chapter 6: Tabu search* pages 165–186. Springer (2006). 18

[37] Djurdje Cvijović and Jacek Klinowski. *Taboo Search: An Approach to the Multiple-Minima Problem for Continuous Functions* pages 387–406. Springer US, Boston, MA (2002). 18

[38] Lawrence J. Fogel. *Intelligence Through Simulated Evolution: Forty Years of Evolutionary Programming*. John Wiley & Sons, Inc., New York, NY, USA (1999). 19

[39] A. E.Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer Verlag (2003). 20

[40] J-H Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA (1992). 20

[41] R. Storn and K. Price. Minimizing the real functions of the icec'96 contest by differential evolution. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 842–844 5 (1996). 20

[42] R. Storn and K. Price. *Differential evolution, a simple and efficient heuristic for global optimization over continuous spaces*. Journal of Global Optimization **11**(4), 341–359 Dec (1997). 20

[43] P. Civicioglu. *Backtracking Search Optimization Algorithm for numerical optimization problems*. Applied Mathematics and Computation **219**(15), 8121–8144 (2013). 21, 62, 63

[44] D. Corne, M. Dorigo, F. Glover, D. Dasgupta, P. Moscato, R. Poli, and K. V. Price, editors. *New Ideas in Optimization*. McGraw-Hill Ltd, UK, Maidenhead, UK, England (1999). 22

[45] T. Fukuda, K. Mori, and M. Tsuklyama. *Immune networks using genetic algorithm for adaptive production scheduling*. IFAC Proceedings Volumes **26**(2, Part 4), 353 – 356 (1993). 22

[46] G. B. Lamont C. C. Coello and D. A. van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, Paris (2007). 22

[47] A. Zhou, B. Qu, H.Li, S-Z. Zhao, P-N Suganthan, and Q. Zhang. *Multiobjective evolutionary algorithms: A survey of the state of the art*. Swarm and Evolutionary Computation **1**(1), 32–49 (2011). 22

[48] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pages 93–100. L. Erlbaum Associates Inc. (1985). 22

[49] David A. Van Veldhuizen and Gary B. Lamont. Multiobjective evolutionary algorithm research: A history and analysis, (1998). 22, 28

[50] C. M. Fonseca and P. J. Flemming. *Multiobjective optimization and multiple constraint handling with evolutionary algorithms, part ii: application example*. IEEE Transactions on Systems, Man and Cybernetics **28**, 38–47 (1998). 22, 62

[51] H. Jeffrey, H. Jeffrey, N. Nicholas, and G. E. David. *A Niched Pareto Genetic Algorithm for Multiobjective Optimization.* in proceedings of the first ieee conference on evolutionary computation, ieee world congress on computational intelligence **1**, 82–87 (1994). 22, 62

[52] E. Zitzler and L. Thiele. *Multiobjective optimization using evolutionary algorithms a comparative case study.* In: Eiben AE, Bäck T, Schoenauer M, Schwefel H-P (eds) In Parallel problems solving from nature. Springer, Berlin, Germany **1498**, 292–301 (1998). 22, 24

[53] N. Srinivas and K. Deb. *Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms.* Evolutionary Computation **2**(3), 221–248 (1994). 22, 62

[54] Q. Zhang, W. Liu, and H. Li. The performance of a new version of moea/d on cec09 unconstrained mop test instances. In *2009 IEEE Congress on Evolutionary Computation*, pages 203–208 (2009). 22, 26, 62

[55] N. Srinivas and K. Deb. *Multiobjective optimization using nondominated sorting in genetic algorithms.* Evolutionary Computation **2(3)**, 221–248 (1994). 23

[56] D. W. Corne, , and J. D. Knowles. *Paes: Approximating the nondominated front using the pareto archived evolution strategy.* Evolutionary Computation **8(2)**, 149–172 (2000). 25, 33, 36, 62

[57] M. Gong, L. Jiao, H. Du, and L. Bo. *Multiobjective immune algorithm with nondominated neighbor-based selection.* Evolutionary Computation **16(2)**, 225–255 (2008). 26, 33, 34, 36, 39, 41, 42

[58] J. J. Durillo and A. J. Nebro. *jMetal: A Java framework for multi-objective optimization.* Advances in Engineering Software **42**(10), 760–771 (2011). 28, 29, 69

[59] J. R. Schott. *Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization.* Massachusetts Institute of Technology, Department of Aeronautics and Astronautics (1995). 29, 40

[60] E. Zitzler. *Evolutionary algorithms for multiobjective optimization: methods and applications.* Doctoral dissertation ETH 13398, Swiss federal institue of technology (ETH). Zurich, Switzerland **1498** (1998). 29, 62

[61] E. Zitzler and L. Thiele. *Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach.* IEEE Transactions on Evolutionary Computation **3(4)**, 257–271 (1999). 33

[62] S. N. Omkar, R. Khandelwal, S. Yathindra, G. N. Naik, and S. Gopalakrishn. *Artificial immune system for multi-objective design optimization of composite structures.* Engineering Applications of Artificial Intelligence **21**, 1416–1429 (2008). 33

[63] G. C. Luh, C. H. Chueh, and W. W. Liu. *Moia: multi-objective immune algorithm.* Engineering Optimization **35**, 143–64 (2003). 33

[64] C. Coello and N. Cortes. *Solving multiobjective optimization problems using an artificial immune system.* Genetic Programming and Evolvable Machines **6(2)**, 163–190 (June 2005). 33

[65] J. Gao and J. Wang. *Wbmoais: a novel artificial immune system for multi-objective optimization.* Computers and Operations Research **37(1)**, 50–61 (2010). 33

[66] J. Shi, M. Gong, W. Ma, and Licheng Jiao. *A multiobjective immune algorithm based on a multiple-affinity model.* European Journal of Operational Research **202(1)**, 60–72 (2010). 33

[67] A. Zinflou, C. Gagn, and M. Gravelc. *Gismoo: A new hybrid genetic/immune strategy for multiple-objective optimization.* Computers and Operations Research **39(9)**, 1951–1968 (2012). 33

[68] R. Shang, L. Jiao, and F. Liu M. Ma. *A novel immune clonal algorithm for mo problems.* IEEE Transactions on Evolutionary Computation **16(1)**, 35–50 (2012). 33

[69] L. Jiao, F. Liu, and W. Ma. *A novel immune clonal algorithm for mo problems.* IEEE Transactions on Evolutionary Computation **16(1)**, 35–50 (2012). 33

[70] K. Deb and H.G. Beyer. *Self-adaptive genetic algorithms with simulated binary crossover.* Evolutionary Computation **9(2)**, 197–221 (2001). 33

[71] E. Y. C. Wong, H. S. C. Yeung, and H. Y.K. Lau. *Immunity based hybrid evolutionary algorithm for multiobjective optimization in global container repositioning.* Engineering Applications of Artificial Intelligence **22**, 842–854 (2009). 33

[72] Q. Lin, J. Chen, and Z. Ji. *A hybrid immune multiobjective optimization algorithm.* European Journal of Operational Research **204(2)**, 294–302 (2010). 33

[73] Y. Qi, F. Liu, M. Liu, M. Gong, and L. Jiao. *Multi-objective immune algorithm with baldwinian learning.* Applied Soft Computing **12(8)**, 2654–2674 (2012). 33

[74] Q. Lin and J. Chen. *A novel micro-population immune multi-objective optimization algorithm.* Computers and Operations Research **40(6)**, 1590–1601 (2013). 33

[75] Q. Lin, Q. Zhu, P. Huang, J. Chen, Z. Ming, and J. Yu. *A novel hybrid multi-objective immune algorithm with adaptive differential evolution.* Computers and Operations Research **62**, 95–111 (2015). 33

[76] Y.T. Qi, Z.T. Hou, M.L. Yin, H.L. Sun, and J.B. Huang. *An immune multi-objective optimization algorithm with differential evolution inspired recombination.* Applied Soft Computing **547(29)**, 395–410 (2015). 33

[77] S. Dasa and P.N. Suganthan. *Differential evolution: a survey of the state-of-the-art*. IEEE Transactions on Evolutionary Computation **15(1)**, 4–31 (2011). 33

[78] L. N. De Castro and F. J. Von Zuben. *Learning and optimization using the clonal selection principle*. IEEE Transactions on Evolutionary Computation **6(3)**, 239–251 (2002). 35

[79] V. Cutello, G. Nicosia, and M. Pavone. *Exploring the capability of immune algorithms: A characterization of hypemutation operators*. In Proceedings of Third International Conference on Artificial Immune Systems, ICARIS 2004,Catania, Italy **3239**, 263–276 (2004). 35

[80] Deb Kalyanmoy and R.B. Agarwal. *Simulated binary crossover for continuous search space*. Complex Systems **9**, 115–148 (1995). 35

[81] M.R. Sierra and C. A. C. Coello. *Improving pso-based multi-objective optimization using crowding, mutation and ϵ-dominance*. In Proceedings of the Evolutionary Multi-Criterion Optimization. **3239**, 263–276 (2005). 40

[82] M. Arkadiusz. *Geometrical aspects of optimum truss like structures for three-force problem*. Structural and Multidisciplinary Optimization **45(1)**, 21–32 (2012). 47, 88

[83] J. N. Richardson, S. Adriaenssens, P. Bouillard, and R. F. Coelho. *Multiobjective topology optimization of truss structures with kinematic stability repair*. Structural and Multidisciplinary Optimization **46**, 513–532 (2012). 47, 88

[84] 47, 88

[85] F.M. Hemez and E. Pagnacco. *Statics and inverse dynamics solvers based on strain-mode disassembly*. Revue Européenne des Eléments Finis **9(5)**, 511–560 (2000). 49, 89

[86] A. Zhoua, H.Li B. Qub, S. Zhao, P. Suganthan, and Q. Zhangd. *Multiobjective evolutionary algorithms: A survey of the state of the art*. Swarm and Evolutionary Computation **1**, 32–49 (2011). 62

[87] C.A.C. Coello, G.T. Pulido, and M.S. Lechuga. *Handling multiple objectives with particle swarm optimization*. IEEE Transactions on Evolutionary Computation **8**(3), 256–279 (2004). 62

[88] S. Kukkonen and J. Lampinen. Performance assessment of generalized differential evolution 3 with a given set of constrained multi-objective test problems. In *2009 IEEE Congress on Evolutionary Computation*, pages 1943–1950 (2009). 62

[89] S. Tiwari, G. Fadel, P. Koch, and K. Deb. Performance assessment of the hybrid archive-based micro genetic algorithm (amga) on the cec09 test problems. In *2009 IEEE Congress on Evolutionary Computation*, pages 1935–1942 (2009). 62

[90] K. Sindhya, A. Sinha, K. Deb, and K. Miettinen. Local search based evolutionary multi-objective optimization algorithm for constrained and unconstrained problems. In *2009 IEEE Congress on Evolutionary Computation*, pages 2919–2926 (2009). 62

[91] A. Zamuda, J. Brest, B. Boskovic, and V. Zumer. Differential evolution with self-adaptation and local search for constrained multiobjective optimization. In *2009 IEEE Congress on Evolutionary Computation*, pages 195–202 (2009). 62

[92] V. L. Huang, S. Z. Zhao, R. Mallipeddi, and P. N. Suganthan. Multi-objective optimization using self-adaptive differential evolution algorithm. In *2009 IEEE Congress on Evolutionary Computation*, pages 190–194 (2009). 62

[93] B. Y. Qu and P. N. Suganthan. Multi-objective evolutionary programming without non-domination sorting is up to twenty times faster. In *2009 IEEE Congress on Evolutionary Computation*, pages 2934–2939 (2009). 62

[94] Q.Zhang and H. Li. *MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition.* IEEE Transactions on Evolutionary Computation **11**(6), 712–731 (2007). 62, 68

[95] L. Y. Tseng and C. Chen. Multiple trajectory search for unconstrained/constrained multi-objective optimization. In *2009 IEEE Congress on Evolutionary Computation*, pages 1951–1958 (2009). 62

[96] M. Liu, X. Zou, Y. Chen, and Z. Wu. Performance assessment of dmoea-dd with cec 2009 moea competition test instances. In *2009 IEEE Congress on Evolutionary Computation*, pages 2913–2918 (2009). 62

[97] Y. Wang, C. Dang, H. Li, L. Han, and J. Wei. A clustering multi-objective evolutionary algorithm based on orthogonal and uniform design. In *2009 IEEE Congress on Evolutionary Computation*, pages 2927–2933 (2009). 62

[98] C. Chen, Y. Chen, and Q. Zhang. Enhancing moea/d with guided mutation and priority update for multi-objective optimization. In *2009 IEEE Congress on Evolutionary Computation*, pages 209–216 (2009). 62

[99] S. Gao, S., B. Xiao, L. Zhang, Y. Shi, X. Tian, Y. Yang, H. Long, X. Yang, D. Yu, and Z. Yan. An orthogonal multi-objective evolutionary algorithm with lower-dimensional crossover. In *2009 IEEE Congress on Evolutionary Computation*, pages 1959–1964 (2009). 62

[100] S. X. Zhang, L. M. Zheng, L. Liu, S. Y. Zheng, and Y. M. Pan. *Decomposition-based multi-objective evolutionary algorithm with mating neighborhood sizes and reproduction operators adaptation.* Soft Computing pages 1–12 (2016). 62

[101] J. Yazdi. *Decomposition based multi objective evolutionary algorithms for design of large-scale water distribution networks.* Water Resources Management **30**(8), 2749–2766 (2016). 62

[102] K. Guney, A. Durmus, and S. Basbug. *Backtracking search optimization algorithm for synthesis of concentric circular antenna arrays*. International Journal of Antennas and Propagation (2014). 63

[103] Ahmed M. Othman and Almoataz Y. Abdelaziz. *Enhanced backtracking search algorithm for optimal coordination of directional over-current relays including distributed generation*. Electric Power Components and Systems **44**(3), 278–290 (2016). 63

[104] A. El-Fergany. *Multi-objective Allocation of Multi-type Distributed Generators along Distribution Networks Using Backtracking Search Algorithm and Fuzzy Expert Rules*. Electric Power Components and Systems **44**(3), 252–267 (2016). 63

[105] H. Hachimi. *Hybridization of metaheuristic algorithms in global optimization and their applications*. Theses INSA de Rouen (2013). 79

[106] P. Gourmelen, M. Bruyneel, and J-C. Craveur. *Optimisation des structures mécaniques, Méthodes numériques et éléments finis*. Technique et Ingénierie, Dunod (2016). 79

[107] A. EL Hami and B. Radi. *Incertitudes optimisation et fiabilité des structures*. Lavoisier (2013). 79

[108] A. H. Gandomi, X. S. Yang, S. Talatahari, and A. H. Alavi. *Metaheuristic Applications in Structures and Infrastructures*. Elsevier, Oxford (2013). 80

[109] R. H. Lopeza, D. Lemossea, J. E. Souza de Cursia, J. Rojasb, and A. El-Hami. *An approach for the reliability based design optimization of laminated composites*. Engineering Optimization **43**, 1079–1094 (2011). 80

[110] E. Talbi. *Metaheuristics: From Design to Implementation*. Wiley Publishing (2009). 80

[111] Xin-She Yang. *Nature-Inspired Metaheuristic Algorithms*. Luniver Press (2008). 80

[112] C.Coello and G. Pulido. *Multiobjective structural optimization using a microgenetic algorithm*. Structural and Multidisciplinary Optimization **30**, 388–403 (2005). 80, 83, 84

[113] L. Li and F. Liu. *Group search optimization for applications in structural design*. Adaptation, Learning, and Optimization **9** (2011). 80

[114] W. Gong, Z. Cai, and L. Zhu. *An effective multiobjective differential evolution algorithm for engineering design*. Structural and Multidisciplinary Optimization **38**, 137–157 (2009). 80, 83

[115] H. Zidani, E. Pagnacco, R. Sampaio, R. Ellaia, and J. E. Souza de Cursi. *Multi-objective optimization by a new hybridized method: applications to random mechanical systems*. Engineering Optimization **45(8)**, 917–939 (2012). 80

[116] X. Yang and S. Deb. *Multiobjective cuckoo search for design optimization.* Computers and Operations Research **40**, 1616–1624 (2013). 80

[117] T. Erfani, S. V. Utyuzhnikov, and B. Kolo. *A modified directed search domain algorithm for multiobjective engineering and design optimization.* Structural and Multidisciplinary Optimization **48(6)**, 1129–1141 (2013). 80, 81

[118] W. Gong, Z. Cai, and L. Zhu. *An effective multiobjective differential evolution algorithm for engineering design.* Structural and Multidisciplinary Optimization **38**, 137–157 (2009). 86, 87

[119] X. Yang and S. Deb. *Multiobjective cuckoo search for design optimization.* Computers and Operations Research **40**, 1616–1624 (2013). 86, 87

[120] H. Zidani, E. Pagnacco, R. Sampaio, R. Ellaia, and J. E. Souza de Cursi. *Multi-objective optimization by a new hybridized method: applications to random mechanical systems.* Engineering Optimization **45**(8), 917–939 (2013). 92, 94

[121] H. Versteeg and W. Malalasekera. *An introduction to computational fluid dynamics: The finite volume method approach.* 2nd Edition (2007). 99

[122] A. K. Chopra. *Dynamics of structures theory and applications to earthquake engineering, 2nd edition.* New Jersey Pearson (2007). 100

[123] B. Radi R. El Maani and A. El-Hami. *Reliability study of a coupled three dimensional system with uncertain parameters.* Journal of Advanced Materials Research **1099**, 87–93. (2015). 100

[124] R. El Maani, B. Radi, and A. El-Hami. *Some decomposition methods in the analysis of repetitive structures.* Computers and Structures **58(5)**, 973–980. (1996). 100

[125] *Ansys workbench 14.0- user guide.* 101

[126] V. V Ranade. *Computational flow modelling for chemical reactor engineering.* Process Systems Engineering. Academic Press, New York, USA **58(5)**, 973–980. (1996). 101

[127] C. T. Shaw. *Using a segregated finite element scheme to equations solve the incompressible navierstokes.* International Journal for Numerical Methods in Fluids **12(1)**, 81–92 (1991). 101

[128] O. C. Zienkiewicz and K. Morgan. *Finite element and approximation.* International Journal for Numerical Methods in Fluids **12(1)**, 81–92 (1991). 101