Contribution to robust and fiabilist optimization : Application to the design of acoustic radio-frequency filters

François Schott

UBFC
UNIVERSITÉ
BOURGOGNE FRANCHE-COMTÉ

utbm
université de technologie
Belfort-Montbéliard

THÈSE DE DOCTORAT DE L'ÉTABLISSEMENT UNIVERSITÉ BOURGOGNE FRANCHE-COMTÉ

PRÉPARÉE À L'UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

École doctorale n°37

Sciences Pour l'Ingénieur et Microtechniques

Doctorat de Sciences pour l'Ingénieur

par

François Schott

Contribution to robust and fiabilist optimization

Application to the design of acoustic radio-frequency filters

Thèse présentée et soutenue à Besançon, le 20 décembre 2019

Composition du Jury :

ELMAZRIA Omar	Professeur à l'Université de Lorraine - Institut Jean Lamour-CNRS	Président du Jury
BIGEON Jean	Directeur de recherche CNRS, G-SCOP	Rapporteur
IDOUMGHAR Lhassane	Professeur à l'Université de Haute-Alsace - Institut IRIMAS	Rapporteur
BARON thomas	Ingénieur de Recherche, HDR, ENSMM Département Temps-Fréquence, Institut FEMTO ST, (UMR 6174 CNRS/UFC/ENSMM/UTBM)	Directeur de thèse
MEYER Yann	Maître de Conférences HDR, Sorbonne Universités, Université de Technologie de Compiègne, CNRS, UMR 7337 Roberval, centre de recherche Royallieu, CS 60 319, 60203 Compiègne cedex, France	Codirecteur de thèse
CHAMORET Dominique	Maître de Conférences, ICB UMR 6303, CNRS Univ. Bourgogne Franche-Comté, UTBM, Belfort, France	Encadrant
SALMON Sébastien	Ingénieur de recherche, AR-electronique	Invité

N° | 2 | 0 | 1 | 9 | U | B | F | C | A | 0 | 3 | 1 |

SPIM
école doctorale **sciences pour l'ingénieur et microtechniques**

Title: Contribution to robust and fiabilist optimization

Keywords: Optimization, Radio-frequency, Engineering design optimization, Robust optimization, Fiabilist optimization, Surface acoustic wave, SAW filters, MEMS

Abstract:

This thesis aims to develop robust and fiabilist optimiztion means in order to face the future requirements of the radio-frequency (RF) filter market. The goals of this thesis are: to reduce the optimization process timespan, to be able to find a solution that fully satisfies a tough bill of specifications and to reduce failure rate due to manufacturing uncertainties. Several research works has been done to achieve these goals.
During the formulation phase of an engineering design optimization (EDO) process, ambiguities, leading to unsatisfying solutions, could happen. In this case, some phases of the EDO process has to be iterated increasing its timpespan. Therefore, a Framework to properly formulate an optimization problem has been developed. During a run of optimization, for the algorithm to solve the problem according to the designer's preferences and thus avoid un-satisfying solution, two challenges, among others, have to be faced. The variable challenge is about handling mixed variables with different order of magnitudes while the satisfaction challenge is about properly computing satisfaction. The Normalized Evaluations approach has been developed to face these challenges. The resolution method efficiency strongly relies on the choice of its core element: the algorithm. Hence, the high number of optimization algorithms is a challenge for an optimizer willing to choose the correct algorithm. To face this challenge, a Benchmark, being a tool to assess the algorithm performance and to be able to select the correct algorithm for a given problem, has been developed. Algorithm efficiency depends on the values given to its parameters, its setting. A common practice is to tune parameters manually which does not guarantee the best performance. A solution to this problem is to perform meta-optimization (MO) which consists in optimizing an algorithm efficiency by tuning its parameters. A MO approach using a benchmark to evaluate settings has been tested. A fiabilist optimization method, taking the uncertainties into account, has to be developed. However, this method has to do so without degrading resolution time, which is usually the case with fiabilist methods. Therefore, a Sparing Fiabilist Optimization method taking uncertainties into account without increasing too much the numerical resolution timespan has been developed.
These methods have been applied to optimize a RF filter, with a tough bill of specifications, for which no fully satisfying solution where found before the thesis. By using the methods developed during this thesis, a determinist solution, not taking uncertainties into account, which fully satisfies the bill of specifications, has been found. Moreover, a fiabilist solution having a 71% success rate has been found. As a conclusion, it appears that optimization methods developed during these thesis where sufficient to face the future requirements of the radio-frequency filter market.

Titre : Contribution to robust and fiabilist optimization

Mots-clés : Optimisation, Radio-fréquence, Optimisation en conception, Optimisation robuste, Optimisation fiabiliste, Onde de surface acoustique, Filtre SAW, MEMS

Résumé :

Cette thèse vise à développer des moyens d'optimisation robuste et fiabiliste dans le but de faire face aux besoins du marché des filtres radio-fréquence (RF). Les objectifs de cette thèse sont: réduire la durée du processus d'optimisation, être capable de trouver une solution qui satisfasse complètement un cahier des charges difficile et réduire le pourcentage de rebut dû aux incertitudes de fabrication. Plusieurs travaux de recherche ont été menés pour atteindre ces objectifs.
Durant la phase de formulation d'un processus d'optimisation en conception (EDO), des ambiguïtés, conduisant à des solutions insatisfaisantes, peuvent survenir. Dans ce cas, certaines phases du processus d'EDO doivent être itérées augmentant sa durée. Par conséquent, un cadre pour formuler correctement les problèmes d'optimisation a été développé. Durant un run d'optimisation, pour que l'algorithme résolve le problème conformément aux attentes du concepteur et ainsi éviter les solutions insatisfaisantes, deux défis, parmi d'autres, doivent être surmontés. Le défi des variables consiste à gérer des variables mixtes ayant des ordres de grandeur différents tandis que le défi de la satisfaction consiste à correctement calculer celle-ci. L'approche par évaluations normalisées a étée développée pour faire face à ces défis. L'efficacité de la méthode de résolution dépend fortement du choix de son élément central: l'algorithme. De ce fait, le grand nombre d'algorithmes d'optimisation est un défi pour un concepteur souhaitant choisir un algorithme adapté. Pour faire face à ce défi, un benchmark, qui est un outil pour évaluer la performance d'un algorithme et être capable de choisir l'algorithme adapté à un problème, a été développé. L'efficacité d'un algorithme dépend des valeurs données à ses paramètres, son paramétrage. Une pratique commune est de régler les paramètres manuellement, ce qui ne garantit pas les meilleures performances. Une solution à ce problème est de réaliser une méta-optimisation (MO) qui consiste à optimiser l'efficacité d'un algorithme en réglant son paramétrage. Une approche de MO utilisant un benchmark pour évaluer des paramétrages a été testée. Une méthode d'optimisation fiabiliste, prenant en compte les incertitudes, doit être développée. Cependant, cette méthode ne doit pas dégrader le temps de résolution, ce qui est généralement le cas des méthodes fiabilistes. Ainsi, a méthode d'optimisation fiabiliste économe prenant en compte les incertitudes sans trop augmenter le temps de calcul ont été développée.
Ces méthodes ont été appliquées pour optimiser un filtre RF, avec un cahier des charges difficile, pour lequel aucune solution totalement satisfaisante n'avait été trouvée avant la thèse. En utilisant les méthodes développées durant cette thèse, une solution déterministe, ne prenant pas en compte des incertitudes, respectant totalement le cahier des charges, a été trouvée. De plus, une solution fiabiliste ayant un pourcentage de succès de 71% a été trouvée. En conclusion, il apparaît que les méthodes développées durant cette thèse sont suffisantes pour faire face aux besoins futurs du marché des filtres radiofréquences.

UBFC UNIVERSITÉ BOURGOGNE FRANCHE-COMTÉ
Université Bourgogne Franche-Comté
32, avenue de l'Observatoire
25000 Besançon, France

Acknowledgments / Remerciements

Remerciements

Une thèse ce n'est pas seulement un travail mais aussi un thésard. Un thésard qui part à l'aventure pour environ trois ans. Pour moi, cette aventure commence quand ***Sébastien Salmon***, ami et ancien patron, décide de me proposer de faire une thèse au sein de son entreprise, My-OCCS. Aussi, avant de remercier tous ceux qui m'ont aidé dans cette aventure, je commencerai par 'Merci Patron !'.

Pour démarrer une thèse, encore faut-il un financement et un projet. J'aimerais donc remercier le ***FEDER*** pour avoir financé, à travers la bourse FC0001257 - 11002, le projet Smart-Inn dont ma thèse a fait partie.

Pour qu'une thèse se passe bien, et pour faire face aux imprévus de celle-ci, il est fortement conseillé d'avoir de bons encadrants. Des encadrants qui veillent à ce que vous gardiez le cap, à vous former et à s'assurer que tout va bien. Merci à ***Thomas Baron***, ***Yann Meyer***, ***Dominique Chamoret*** et ***Sébastien Salmon*** pour leur bienveillance, leur pédagogie et leur persévérance.

Pour finir une thèse en beauté, le rôle du jury est primordial. Je tiens donc à remercier ***Omar Elmazria***, ***Jean Bigeon*** et ***Lhassane Idoumghar*** pour l'implication et l'ouverture d'esprit avec lesquelles ils ont évalué mon travail, pour m'avoir aidé à prendre du recul par rapport au travail réalisé et pour m'avoir aidé à améliorer le mémoire.

Les travaux menés durant cette thèse ont été possibles grâce à des collaborations avec différents partenaires. J'en profite pour remercier ***Florence Bazzaro***, ***Sophie Collong***, ***Raed Koutta*** ainsi que les employés d'***AR-Electronique*** et de ***Frec|n|sys*** pour leur chaleur humaine, leur motivation et le temps qu'ils ont pris pour m'expliquer leurs domaines respectifs.

Une thèse c'est aussi évoluer au sein d'un labo. Je remercie l'ensemble des membres du ***département temps-fréquences*** pour leur convivialité, leur passion communicative pour la recherche et pour ce qu'ils m'ont appris. Un merci tout particulier pour mes collègues de bureau, ***Alex***, ***Arthur***, ***Damien***, ***Guillaume***, ***Justin***, ***Melvin***, ***Mihaela***, ***Stefania*** et ***Tung***, pour leur bonne humeur, pour l'entraide et pour toutes les discussions (sérieuses ou non) qu'on a pu avoir. Merci également à ***Bernard***, ***Chérita***, ***Nicolas*** et ***Valérie*** pour être passé régulièrement nous voir et prendre le temps de discuter.

J'ai eu la chance d'être bien accompagné tout au long de cette aventure. Un grand merci aux doctorants du TF, ***Alex***, ***Alok***, ***Anthony***, ***Arthur***, ***Étienne***, ***Falzon***, ***Fatima***, ***Gautier***, ***Giacomo***, ***Grégoire***, ***Guillaume***, ***Hala***, ***Isha***, ***Jérémy***, ***Kévin***, ***Melvin***, ***Sabina***, ***Stefania***, ***Tung*** et ***Vincent***, pour tous les bons moments passés ensemble: les midis au RU, pour les soirées et sorties (pool party, fête de la musique, volley

et basket) et pour les conférences, séminaires et autres training schools en tous genres.

C'est d'ailleurs grâce aux doctorants du TF que j'ai rejoint l'A'Doc, L'association des jeunes chercheurs de Franche-Comté. Je remercie les ***membres de l'A'Doc*** et en particulier les membres des bureaux, ***Alex***, ***Alice***, ***Djé***, ***Étienne***, ***Grégoire***, ***Guillaume***, ***Jason***, ***Jérôme***, ***Julien***, ***Marie***, ***Nico*** et ***Romain***, pour les événements, pour m'avoir permis de discuter des hauts et des bas d'une thèse et surtout pour les réunions (pas assez sérieuses pour être efficaces donc toujours suffisamment peu pour passer un bon moment).

La thèse m'a aussi permis de découvrir le Bozendo, un art martial au bâton long. Merci beaucoup ***aux Bozendokas*** de Besançon et d'ailleurs, et notamment à ***Pierre***, ***Isahia***, ***Sandrine***, ***Titouan***, ***Nico***, ***Elias***, ***Gwen***, ***Marius***, ***Marion***, ***Mathieu***, ***Edward***, ***Alizé***, ***Mikhail***, ***Pierre***, ***Raph***, ***JC***, ***Colombe***, ***Cédric***, ***Jean-Marie***, ***Létitia***, ***Patrice***, ***Céline***, ***Guillaume***, ***Jean-Seb***, ***Bastien***, ***Cassandre***, ***Sophie***, ***Chéima***, ***Gwen*** et ***Mathieu***, pour leur bonne humeur, pour m'avoir aidé à progresser, pour m'avoir fait confiance pour prendre des cours et pour me permettre de pratiquer mon art martial préféré. Merci en particulier à ***Marion*** et à ***Pierre*** pour leur enthousiasme, pour avoir fait vivre le club et pour avoir motivé les troupes sur le tamis comme pour sortir.

Cette thèse, je la dois aussi à l'***UTBM***. Déjà, parce que c'est grâce à l'UTBM que j'ai rencontré Sébastien Salmon qui m'a pris en thèse. Mais aussi, parce que c'est l'UTBM qui m'a préparé à la thèse et qui a servi de cadre à celle-ci. C'est pourquoi je souhaite remercier ***le personnel*** et ***les professeurs*** de l'UTBM ainsi que ***les élèves*** , que ce soit ceux avec qui j'ai étudié ou ceux que j'ai eu la chance d'avoir en cours.

Mais l'UTBM, plus encore qu'une école, ce sont des UTBohémiens, des amis pour la vie. Un très grand merci à ***mes amis UTBohémiens*** pour avoir fait passer les études si vite, pour toutes les soirées, week-ends et vacances où j'ai pu mettre la thèse de côté et m'amuser et pour avoir toujours été là. Parce qu'il n'y aurait pas de Dr Schott sans Mister Keum, merci à ***Saoul***, ***Hans***, ***Camille***, ***Tac***, ***Minal***, ***Magne***, ***Iredla***, ***Glion***, ***Soeur***, ***Dygne***, ***Afumer***, ***Skia***, ***Gobelin***, ***Shka***, ***Iki***, ***Nam***, ***Yther***, ***Piké***, ***Lérone***, ***Bob***, ***Moaljon***, ***6***, ***Mowgly***, ***Zia*** et ***Aoline***. Merci beaucoup à ***Hans*** et ***Camille*** pour avoir relu le mémoire.

J'ai aussi eu la chance de pouvoir compter sur des amis d'un peu partout. Je tiens à remercier ***mes amis de Besançon, de Metz et de Chine***. Merci à ***Paul***, ***Marie***, ***Julien***, ***P.O.***, ***Adri*** et ***Monc*** pour avoir gardé le contact, pour les soirées à Metz et à Nancy et pour les virées dans les Vosges. Merci à ***Théo***, ***Akira***, ***Vincent*** et ***Eduard*** pour les week-ends retrouvailles, les souvenirs de Chine et les soirées parisiennes. Merci à ***Shakit***, ***Andy***, ***Kent*** et toute ***l'équipe de SZAA*** pour tous les bons moments que j'ai passés avec eux durant mon stage en Chine lequel m'a préparé à affronter les difficultés d'une thèse. Merci à ***Groom***, ***Emilia*** et ***Sonia*** pour les bons moments que j'ai passés à Besançon avec eux.

Pour conclure, et il est temps, merci à ma famille, ***Tonio***, ***M'man***, ***P'pa***, ***Jean-Jacques***, ***Amy***, ***Nathalie***, ***Hubert***, ***Jules***, ***Marie***, ***Baptiste***, ***Dora***, ***Audrey***, ***Louis***, ***Léopold***, ***Augustine***, ***Lucie***, ***Clément***, ***Irène***, ***Xavier*** et aussi à ***Caroline*** et ***Pierre-étienne***. À ceux qui m'ont vu grandir, à ceux à qui je dois tant, à ceux qui ont toujours été là, à ceux dont on se sent toujours proche malgré le temps et la distance, merci pour tout et un peu pour le reste aussi.

Acknowledgments

A Ph.D. thesis is not only at work, but also a Ph.D. Student. A student going on a journey for approximately three years. For me, this journey began with ***Sébastien Salmon***, a friend and former boss of mine, proposing me a thesis at his own company, My-OCCS. Also, before thanking all whom helped on this journey, I would begin with 'Thanks Boss!'.

To begin a thesis, one must find a subvention and a project. I would therefore thank the ***ERDF*** for subventioning, through the grant FC0001257 - 11002, the project Smart-Inn which my thesis is part of.

For a theist to go well, and to be able to face the unexpected events related to it, it is strongly advised to have good supervisors. Supervisors whom ensure you stay the course, whom form you and assure everything is going well. Thank you to ***Thomas Baron***, ***Yann Meyer***, ***Dominique Chamoret*** and ***Sébastien Salmon***, for their kindness, their pedagogy and their perseverance.

For a thesis to finish on a high note, the jury is key. I want to thank ***Omar Elmazria***, ***Jean Bigeon*** and ***Lhassane Idoumghar***, for the involvement and open mind they have while evaluating my work, for helping me to step back from my work and for helping me to improve the manuscript.

The research works done during this thesis have been possible thanks to collaborations with different partners. I'm taking this opportunity to thank ***Florence Bazzaro***, ***Sophie Collong***, ***Raed Koutta*** and also the employee of ***AR-Electronique*** and ***Frec\|n\|sys*** for their human warmth, their motivation and the time they spent explaining me their research fields.

A thesis it's also being part of a Lab. I thank all the members of the ***Time and Frequency department*** for their friendliness, their communicative passion for research and for all they taught me. A special thank for my office colleagues, ***Alex***, ***Arthur***, ***Damien***, ***Guillaume***, ***Justin***, ***Melvin***, ***Mihaela***, ***Stefania*** and ***Tung***, for their good mood, for the assistance and for the talkings (serious or not) that we had. Thank you also to ***Bernard***, ***Chérita***, ***Nicolas*** and ***Valérie***, for regularly visiting the office and spending time to talk.

I have been lucky to be well accompanied all along my journey. A big thank to the TF Ph.D. Students, ***Alex***, ***Alok***, ***Anthony***, ***Arthur***, ***Étienne***, ***Falzon***, ***Fatima***, ***Gautier***, ***Giacomo***, ***Grégoire***, ***Guillaume***, ***Hala***, ***Isha***, ***Jérémy***, ***Kévin***, ***Melvin***, ***Sabina***, ***Stefania***, ***Tung*** and ***Vincent***, for all the good times we had together: The lunch at the CROUS, the evenings and the go-outs (pool party, music festival, volley and basket) and for all the conferences, seminars and training schools of any kind.

As a matter of fact, it is thanks to the TF Ph.D. students that I joined the A'Doc, the association of the young researchers of Franche-Comté. I thank the ***A'Doc members*** and in particular the members of the boards, ***Alex***, ***Alice***, ***Djé***, ***Étienne***, ***Grégoire***, ***Guillaume***, ***Jason***, ***Jérôme***, ***Julien***, ***Marie***, ***Nico*** and ***Romain***, for the events, for being here to talk about the ups and downs of a thesis and most of all for the meetings (not serious enough to be efficient and so always not serious enough for having a good time).

The thesis also provides me the opportunity to start practicing the Bozendo, a martial art using a long staff. Thanks a lot to ***the Bozendokas*** of Besançon and elsewhere, notably to ***Pierre***, ***Isahia***, ***Sandrine***, ***Titouan***, ***Nico***, ***Elias***, ***Gwen***, ***Marius***,

Marion, ***Mathieu***, ***Edward***, ***Alizé***, ***Mikhail***, ***Pierre***, ***Raph***, ***JC***, ***Colombe***, ***Cédric***, ***Jean-Marie***, ***Létitia***, ***Patrice***, ***Céline***, ***Guillaume***, ***Jean-Seb***, ***Bastien***, ***Cassandre***, ***Sophie***, ***Chéima***, ***Gwen*** and ***Mathieu***, for their good mood, for helping me to progress, for entrusting me giving class and for giving me the opportunity to practice my favorite martial art. Thank you in particular to ***Marion***, and ***Pierre***, for their enthusiasm, for carrying the club and for motivating people on the tatami and to go out.

This thesis, I also owe it to the ***UTBM***. Firstly, because it is thanks to the UTBM that I meet Sébastien Salmon proposing me this opportunity. But also, because the UTBM prepared me to the thesis and provided the framework for it. That's why I would like to thank ***the staff*** and ***the teachers*** of UTBM and also ***the students***, both the ones I studied with and the ones I had the chance to have in class.

But UTBM, more than a university, it is UTBohemians, friends for life. A very big thank to ***my UTBohemian friends*** for making studies going so fast, for all the evening, weekends and holidays helping me to move the thesis aside to have fun and for always being here. As there shall not be a Dr Schott without a Mister Keum, thank you to ***Saoul***, ***Hans***, ***Camille***, ***Tac***, ***Minal***, ***Magne***, ***Iredla***, ***Glion***, ***Soeur***, ***Dygne***, ***Afumer***, ***Skia***, ***Gobelin***, ***Shka***, ***Iki***, ***Nam***, ***Ythier***, ***Piké***, ***Lérone***, ***Bob***, ***Moaljon***, ***6***, ***Mowgly***, ***Zia*** and ***Aoline***. Thanks a lot to ***Hans*** and ***Camille*** for proofreading this manuscript.

I also have the chance to have friends all around that I can count on. I want to thank ***my friends of Besançon, Metz and China***. Thank you to ***Paul***, ***Marie***, ***Julien***, ***P.O.***, ***Adri*** and ***Monc***, for keeping in touch, for the evenings in Metz and Nancy and for the trips in the Vosges. Thank you to ***Théo***, ***Akira***, ***Vincent*** and ***Eduard***, for the reunion weekends, for the memories from China and of the Paris nights. Thank you to ***Shakit***, ***Andy***, ***Kent*** and all ***the SZAA staff***, for all the good times I had with you during my internship in China, which prepared me for the difficulties I faced during my thesis. Thank you to ***Groom***, ***Emilia*** and ***Sonia***, for the good times I had with them in Besançon.

To conclude, and it is time, Thank you to my family, ***Tonio***, ***Ma'***, ***Pa'***, ***Jean-Jacques***, ***Amy***, ***Nathalie***, ***Hubert***, ***Jules***, ***Marie***, ***Baptiste***, ***Dora***, ***Audrey***, ***Louis***, ***Léopold***, ***Augustine***, ***Lucie***, ***Clément***, ***Irène***, ***Xavier*** and also to ***Caroline*** and ***Pierre-étienne***. To those who see me growing, to those whom I owe so much, to those who always been here, to those whom one always feels closer to no matter the time and the distance, thank you for everything and a little for the rest also.

Contents

Introduction

Social-economic context

In microelectromechanical systems (MEMS) industry, nine macro-economic megatrends, which will affect our present and future, have been reported in [1]. Those megatrends are: smart automotive, mobile phone, fifth generation cellular network technology (5G), hyperscale data centers, augmented reality/virtual reality, artificial intelligence/machine learning, voice processing, health-care and industry 4.0. They are increasing the demand for MEMS. This market should reach $82B by 2023 [1]. When 5G will arrive, there will also be an increasing need for radio-frequency (RF) filters. The radio-frequency micro-electromechanical systems (RF MEMS) will have the highest growth of the overall MEMS market. According to [1], '*Driven by the complexities associated with the move to 5G and the higher number of bands it brings, there is an increasing demand for RF filters in 4G/5G, making RF MEMS the largest-growing MEMS segment. This market will soar from US$2.3B in 2017 to US$15B in 2023*'. To develop telecommunication systems for professional, public and strategic applications, such as radar, frequency sources for very high frequencies (VHF: 30–300 MHz) and ultra high frequencies (UHF: 300–3000 MHz) RF bands are needed. The telecommunication market covers several applications requiring an accurate selection of frequencies. To face the specifications of these applications in terms of compacity and energy consumption, surface acoustic wave (SAW) RF filters are widely used [2]. For instance, 80% of passive filters used in the emission and reception part of mobile phones are SAW filters.

Radio-frequency filters: Context of this application domain

The RF devices are used for signal processing in radio-frequency bandwidth, which consists in analyzing, modifying and synthesizing RF signals. They include RF filters, which are used to filter the signal. A signal is a physical phenomenon which can be used to transmit information [3]. By opposition to signal, noise can be defined as a physical phenomenon polluting signal's information [4]. Signal is often described as a phenomenon evolving through time [2] and can be expressed into the frequency domain by using Fourier or Laplace transformation [5]. In a general meaning, signal filtering consists in modifying the signal properties depending on frequencies-based criteria [2]. This thesis focuses on frequency filtering consisting in filtering signal in order to keep frequencies in a given bandwidth.

In order to face the telecommunication and internet of things (IoT) markets needs, improving RF filter specifications is essential. The SAW filters and the bulk acoustic wave (BAW) filters technologies can be used for RF filtering. In agreement with research partners, it has been decided that research in optimization will be focused on SAW filters

cases. According to [6], SAW is better than BAW for applications under 1GHz. Moreover, according to [6]: '*The trend in SAW is to improve performance beyond established limits. Higher process complexity is acceptable if the benefit is large enough, if it allows shrinking size, increase yield or save cost in other ways.*'.

A SAW filter is a MEMS device, with two ports, which could be represented as a four-poles system. This device uses a piezo-electric substrate to propagate an acoustic wave and metallic inter-digited transducers (IDT) to transform an electrical signal into an acoustic wave and back. The signal response of this device could be expressed through different matrices, notably impedance and admittance ones.

In the sixties, White and Voltmer [7] have proven that IDT could be used for exciting and detecting SAW. In the seventies, IDT were mostly used in military applications, such as radars. In the eighties, SAW filters, developed thanks to prior decade innovations, were massively used in color televisions. In the nineties, to face mobile phone market needs, such as energy consumption reduction, researches on the RF components have been prolific. Currently, the SAW filters are part of the RF market which is about US$5B.

The RF filter sizing will impact its specifications. The product sizing is a stage of the product engineering process where the values of the design parameters are defined. Currently, the RF filter sizing is done by RF experts solely using their knowledge. The RF experts could be helped in this task by using engineering design optimization (EDO).

Engineering design optimization: Context of this research domain

Engineering design optimization (EDO) [8] aims to optimize product specifications by tuning the design parameters. It is performed during the sizing phase of the design process. To solve an EDO problem, optimization algorithms are more and more often used. In this case, the designer is likely to be helped by an expert in optimization whom will be refereed as an optimizer.

To solve an EDO problem, the designer, helped by the optimizer, thoroughly formulates the problem. The designer provides information such as the design parameters to tune or how to evaluate a design from its specifications. With these informations, the optimizer implements the problem in an optimization engine [9]. The optimization engine will numerical solve the problem by using a resolution method, whose core is the optimization algorithm. The algorithm solves the problem by testing different values. Its goal is to find a sizing with satisfying specifications.

In order to have a better understanding of the work presented in this manuscript, here is a brief history of EDO:

- 18th century: Newton–Raphson's method to find a local minimum by using derivates.
- 19th century: Euler-Lagrange's works lead to variation computation [10] and to Lagrange's multipliers method able to solve constrained problems.
- 40-50's: Invention of linear programming technics [11] making the transition to modern algorithms. They are mainly used for military and economic applications.
- 70's: First optimization applied to engineering problems. It is the beginning of EDO.

- 80's: Emergence of meta-heuristic methods [12]. A meta-heuristic algorithm is the generalization (meta) of a specialized (heuristic) algorithm. The first generation meta-heuristic algorithms is based on the generalization of the mechanisms of the heuristic algorithms.

- 90's: Emergence of the second generation of meta-heuristic algorithms [12]. This generation of algorithms is inspired by optimization mechanisms which can be found in nature.

- 2000-2010: The number of second generation meta-heuristic algorithms explodes [12, 13]. Benchmarks [14, 15], which are tools to measure the algorithms performance, are published [13].

Thesis context

Passive RF components are currently limited in terms of specifications by several factors. The main ones are: an incomplete knowledge of the properties of some materials, the technological limits and the physical limits of available materials. To face the future requirements of the RF filter markets, solutions to the previously mentioned limits has to be found. In this context, different organizations have decided to make a scientific partnership: AR Electronique, Digital Surf, My-OCCS, Snowray, Freq|n|Sys, and FEMTO-ST. This partnership, created through the FEDER project SMART-INN[1], aims to innovate in the field of RF components through different collaborations.

One of these collaborations aims to develop new passive RF devices with an optimal design. To achieve such a task, robust and fiabilist optimization solutions suited to SMART-INN problematic should be developed. Hence, it has been decided to start this thesis, which goal is to develop optimization methods to meet the partners needs in optimization. These methods are meant to face three challenges:

- **Rapidity**, which is how fast the optimization problem is solved. Currently, sizing a RF filter can last up to two weeks for an experienced designer. One of this thesis goal is to reduce this delay to a few days.

- **Efficiency**, which is how good the specifications of the product are. Currently, when the bill of specifications is tough, even an experienced designer does not always find a sizing respecting all the required specifications. Another goal of this thesis is to find a solution fully satisfying the bill of specifications, even if it is a tough one.

- **Profitability**, which is how affordable the product is. Currently, some designs have a high failure rate due to manufacturing uncertainties. The last goal of this thesis is to reduce this rate to a few percent by taking the manufacturing uncertainties into account during the sizing phase. In order to do so, the optimization process should lead to the selection of a reliable design.

To face these three challenges, five research works have been done:

[1] Systèmes à base de MAtériaux de Rupture, principes et outils Technologiques Innovants pour les Nouveaux composants passifs acousto-électriques pour les télécommunications, les systèmes radiofréquences embarqués et le traitement du signal du futur (SMART-INN)

- The *Framework*: how to properly formulate an optimization problem.
- The *Normalized Evaluations Approach*: how to solve an optimization problem according to the designer's expectations.
- The *Benchmark*: how to assess the performances of algorithms to be able to select the correct one for a given problem.
- The *Meta-Optimization*: how to tune the parameters of an algorithm for it to be as efficient as possible.
- The *Sparing Fiabilist Optimization*: how to take uncertainties into account without increasing too much the numerical resolution timespan.

Organization of the thesis

This document is structured by chapters. The first one is a state-of-the-art of optimization. It is followed by five chapters presenting the research works developed during this thesis. The final chapter, application to radio-frequency, describes the optimization of a SAW filter.

In chapter 1, an optimization state-of-the-art is performed. This chapter introduces information required to appreciate the work developed in other chapters. First, what EDO is and how an EDO process is conducted will be introduced. Then, the main elements of an EDO problem, variables, objectives, constraints and evaluation tools, will be explained. The EDO resolution and the algorithms used for this task will be detailed. This introduction will be concluded by two important notions: robustness and fiability.

In chapter 2, the framework developed to properly formulate an optimization problem is presented. This framework is based on interviews during which the optimizer asks questions to the designer. The optimizer is guided in this task by a questionnaire and a thematic board. The designer's answers are noted to fill in an optimization bill of specifications. Though it has not been tested yet, this framework is proposed to the community.

In chapter 3, a normalized evaluations approach designed to solve an optimization problem according to the designer's expectations, is presented. This approach uses a normalized variables intermediate space to ease the algorithm work. In addition, an advanced solution evaluation method is used by the algorithm to properly assess solutions. The approach has been compared to a classical evaluation method on an industrial problem.

In chapter 4, a benchmark, which is a tool to assess the algorithms performance, is proposed. This benchmark, inspired by the CEC one, aims to improve some aspects of algorithms evaluation. With the proposed benchmark, several stopping criteria are used and several scores are computed thanks to normalization and aggregation means. Several algorithms, commonly used in EDO, have been tested with the proposed benchmark to see if the results obtained are confirmed by literature.

In chapter 5, a meta-optimization approach, which improves the efficiency of an algorithm by tuning its parameters, is presented. This meta-optimization approach uses a design of experiment to the find the values of the parameters optimizing the main score of the benchmark for the algorithm tested. These values have been tested both on the benchmark developed in this thesis and on an industrial problem. The results are compared

to the ones obtained by using the values defined by an expert in optimization and by other meta-optimization approaches.

In chapter 6, a sparing fiabilist method, which aims to take the uncertainties into account without increasing too much the numerical resolution timespan, is presented. This method is based on both statistical fiabilistic evaluations and deterministic evaluations. The solution used for the algorithms' global search mechanisms are evaluated in a fiabilist way whereas other solutions are evaluated in a determinist way. The proposed fiabilist evaluation method has been tested on an uncertain version of the benchmark developed in this thesis. The results obtained by the proposed method have been compared to ones obtained by two other fiabilist approaches.

In chapter 7, how to optimize a RF filter, using methods developed in this thesis, is detailed. After an introduction to the RF domain, how to apply previous chapters works to a practical case is explained. The test problem has been solved with and without considering uncertainties.

Finally, the conclusion: summarizes the main results, discusses how they respond to challenges, what are the current limits and what are the prospects.

Acronyms

Table 1: Acronyms

AI	Artificial Intelligence
BAW	Bulk Acoustic Wave
BBOB	Black-Box Optimization Benchmarking
BHT	Bound Handling Techniques
BIM	Boundary Integral Method
CEC	Congress on Evolutionary Computation
CMA	Covariance Matrix Adaptation
CMAES	Covariance Matrix Adaptation Evolution Strategy
COCO	Comparing Continuous Optimizers
CoSyMA	Composants et Systèmes Micro-Acoustiques
DMS	Double Mode SAW
DoE	Design of Experiment
EDO	Engineering Design Optimization
FEM	Finite Element Method
FEs	Function Evaluations
FO	Fiabilist Optimization
FSR	Feasible Space Ratio
GA	Genetic Algorithm
GECCO	Genetic and Evolutionary Computation Conference
IDT	Inter-Digited Transducers
IEF	Inpedence Elements Filter
IoT	Internet of Things
LCRF	Longitudinally Coupled Resonator Filter
MEMS	Microelectromechanical Systems
MO	Meta-optimization
MOO	Multi-Objective Optimization
NE	Normalized Evaluations
OBS	Optimization Bill of Specifications
OFTS	Objective-Function Test Suite
OIA	Observation-Interpretation-Aggregation
PC	Principal Components
PCA	Principal Component Analysis
PSO	Particle Swarm Optimization
RCTS	Real-Case Test Suite
RBDO	Reliability-Based Design Optimization
RF	Radio-Frequency
SA	Simulated Annealing
SAW	Surface Acoustic Wave
SFO	Sparing Fiabilit Optimization
SMART-INN	Systèmes à base de MAtériaux de Rupture, principes et outils Technologiques Innovants pour les Nouveaux composants passifs acousto-électriques pour les télécommunications, les systèmes radiofréquences embarqués et le traitement du signal du futur
SNR	Signal-to-Noise-Ratio
TCRF	Transversely Coupled Resonator Filter

TS	Test Suite
UHF	Ultra High Frequencies
VHF	Very High Frequencies
WHO	World Health Organization
WOF	Waterfalls Objective-Function

Notations

Many notations are used in the following manuscript and sometimes one is close to another either in writing or in meaning. To ensure a good understanding of the content, all these notations are specified in table 2.

Table 2: Notations

Optimization	
Objective	
F	Objective-functions vector
f	A single objective-function
N_o	Number of objective
Variable	
X	Variables vector
X^{min}	The vector of the lower bounds of the variables
X^{max}	The vector of the upper bounds of the variables
x	A single variable
x^{min}	The lower bound of a continuous variable
x^{max}	The upper bound of a continuous variable
$\mathfrak{s}$	The scale of a discrete variable
$\mathfrak{l}$	The length of the scale of a discrete variable
x^i	The i-th element of the scale of a discrete variable
D	Dimension/Number of variables of an optimization problem
d	Index for dimension
$\mathcal{S}$	Search space
Constraint	
G	Inequality constraints vector
g	A single inequality constraint
N_g	Number of inequality constraints
H	Equality constraints vector
h	A single equality constraint
N_h	Number of equality constraint
λ	A Lagrange multiplier
Advanced notions	
Δ	Vector of the uncertainties of the variables
δ	A random uncertainties vector
$\mathcal{N}$	The normal distribution law
$\mathcal{X}$	A confidence limit percentage
FSR	Feasible space ratio
Normalized evaluations approach related	

Variable related	
Z	Normalized variables vector
z	A single normalized variable
Y	Bounded normalized variables vector
y	A single bounded normalized variable

Penalties computation	
g^l	The limit value of an inequality constraint
g^r	The reference value of an inequality constraint
h^r	The reference value of an equality constraint
h^m	The error margin of an equality constraint
P	The penalties vector
p	A single penalty
$\mathcal{P}$	The overall penalty
N_j	The number of penalties

Bonuses computation	
B	The bonuses vector
b	A single bonus
$\mathcal{B}$	The overall bonus
N_b	The number of bonuses
C_i	The i-th correspondence point used for a bonus computation
f^r	The objective function value range
f^s	Satisfying objective value
b^s	Satisfying bonus value
c_s	Satisfying correspondence point
f^u	Un-satisfying objective value
b^u	Un-satisfying bonus value
c_u	Un-satisfying correspondence point
$b(f)$	Bonus-function
s	The satisfaction

Benchmark	
General notions	
FEs	The number of allowed Function Evaluations to solve an optimization problem (stopping criteria)
MaxFEs	The 'Maximum Function Evaluations' coefficient (*MaxFEs*) is used to adjust *FEs*.
Number of elements	
N_D	Number of dimensionality used in the benchmark
N_F	Number of test functions used in the benchmark
N_M	Number of *MaxFEs* used in the benchmark
N_C	Number of optimization cases of the benchmark
N_T	Number of runs of an optimization case

N_R	Total number of runs

Functions details	
N_{uni}	Number of uni-modal functions used by the benchmark
N_{multi}	Number of multi-modal functions used by the benchmark
N_{hybrid}	Number of hybrid functions used by the benchmark
$N_{composite}$	Number of composite functions used by the benchmark

Runs	
V_G	Gross value obtain by the algorithm at the end of a run
E_G	Gross number of evaluations made by the algorithm at the end of a run
V_N	Normalized value of the run
E_N	Normalized number of evaluations of the run
R_R	Run Result based on the aggregation of V_N and E_N

Function details	
f_{min}	Minimum of the objective function
f_{max}	Maximum of the objective function
f_{em}	Allowed error margin of the objective function

Case results and sub-results	
R_C	A case result
R_A	A case alpha sub-result
R_O	A case omega sub-result
R_V	A case value sub-result
R_K	A case convergence sub-result

Case's sets of values	
$\Re_R$	Set of run results of an optimization case
$\Re_V$	Set of value sub-results of an optimization case
$\Re_E$	Set of evaluations sub-results of an optimization case

Normalization and aggregation operators	
Υ_E	Normalization operator for the gross number of evaluations
Υ_V	Normalization operator for the gross value
$\mathcal{A}_R$	Aggregation operator used to compute run result
$\mathcal{A}_V$	Aggregation operator used to compute a case value sub-result
$\mathcal{A}_K$	Aggregation operator used to compute a case convergence sub-result
$\mathcal{A}_A$	Aggregation operator used to compute a case alpha sub-result
$\mathcal{A}_O$	Aggregation operator used to compute a case omega sub-result
$\mathcal{A}_C$	Aggregation operator used to compute a case result
$\mathcal{A}_M$	Aggregation operator over *MaxFEs*
$\mathcal{A}_F$	Aggregation operator over functions
$\mathcal{A}_D$	Aggregation operator over dimensions

Normalization and aggregation: support	
V	Values/results vector to aggregate
w	Normalized weights for values/results to aggregate
W	Raw weights for values/results to aggregate
μ_I	Mean of I, a set of values/results
σ_I	Standard deviation of I, a set of values/results
d	Dimensions index
f	Functions index
m	*MaxFEs* index

Global score computation	
R_C^{dfm}	Case result for dimension d, function f and *MaxFEs* m
R_M^{df}	*MaxFEs* intermediate results for dimension d and function f
R_F^{d}	Function intermediate results for dimension d

Meta-optimization	
SNR	Signal-to-noise ratio
y	A DoE test result

PSO	
t	Iteration number
v	The velocity of a particle
p_{best}	The personal best position of a particle
g_{best}	The global best position of all particles
r	A random number
c_1	The cognition parameter
c_2	The social parameter
ω	The inertia of particles
it_{max}	Maximal number of iterations
W_{min}	Minimal inertia
W_{max}	Maximal inertia
v_{min}	The lower bound of random initial velocity given to a particle
v_{max}	The upper bound of random initial velocity given to a particle
V_{factor}	The coefficient to compute initial speed bounds
N	Population size
R	Population radius
R_v	Radius threshold value
R_t	Number of iteration inside the radius to stop

Sparing fiabilist optimization	
Fiability general notions	
A	Uncertain area
s^{δ}	The satisfaction of a single daughter evaluation
s^{Δ}	The set of the satisfactions of the daughters evaluations
μ	Mean

σ	Standard deviation
P	Probability of an event

Sampling size equation notation	
n	Sampling size
t	Margin coefficient deduced from 'c'
c	Confidence limit
p	Supposed proportion of the elements validating the test
e	Error margin on the measure

RF	
RF general notions	
V_ϕ	Phase velocity
λ	Wavelength
k^2	Coupling factor
v_0	Velocity in substrate
v_m	Velocity in electrode
f	Frequency
f_0	The center frequency
DMS design parameters	
h	Electrodes thickness
w	Electrodes aperture
a/p	Ratio of an electrode width over electrodes mechanical period
p	Electrodes Mechanical period
l	Gap length
n	Number of transducers in an element
Filter specifications	
Ψ	Insertion loss
T	Transfer function
R_a	Ripples amplitude
R_n	Number of ripples
Pas	Pass band
Rej	Rejection band
Φ	Group delay
Unclassified	
α, β, γ	Used for intermediate computations inside an equation
i	Index

1

Optimization

Contents

This chapter intends to introduce the branch of optimization considered in this thesis so that the research works developed in the following chapters could be understood and appreciated. This branch of optimization, the Engineering Design Optimization (EDO), is positioned and presented in section 1.1. EDO consist in finding the best solution to a design problem, which as to be formulated as explained in section 1.2. Once the EDO problem formulated, the expert in optimization could set a resolution method to numerically solve the problem. A single numerical resolution of an EDO problem, a run, is described in section 1.3. The core element of this resolution method is the optimization algorithm, which is introduced in section 1.4. To face this thesis challenge, the resolution method robustness and fiability, defined in section 1.5, should be improved.

1.1/ Positioning

Optimization is a word used in different contexts with multiple meanings. From EDO perspective, optimization consists in creating or designing a product for it to satisfy human needs a much as possible [8]. However, to understand how EDO is performed, one must be aware of the fact that EDO is a branch of 'numerical optimization' which is a branch of applied mathematics. Indeed, as numerical optimization consists in finding the best solutions (maximizing or minimizing) for a given problem [9], EDO relies on mathematical tools. However, there is a gap between EDO optimization means, which are applied-oriented, and numerical optimization ones, which are theoretical-oriented [16].

In order to define the context in which EDO is performed, section 1.1.1 will positioned EDO into the design process. The EDO could be done by different means which are defined in section 1.1.2. By using optimization algorithm to solve EDO, the EDO process could be decomposed into three phases detailed in section 1.1.3.

1.1.1/ Engineering Design Opimization in design process

From the different modelings of the engineering process which exists [17], the one from Ulrich & Eppinger [18] will be used to position the EDO process into the design one, which is done in figure 1.1. As depicted, the EDO process is used to performed the detailed design step of the design process, in the case of a sub-product. The EDO process, as it is done in a late design step of a low-level product, is performed through a highly specific design step. In higher product levels, an optimization process will either be a combinatorial optimization problem [19] aiming to define product architecture or a large-scale optimization problem [20] aiming to size several sub-product at a time. Both those scenario, which require specific methods, are out of this thesis spectrum. EDO is often done manually and is time consuming, requiring at least 50% of design life cycle [8]. The timespan of this phase, which depends on the method used to conduct EDO, should be reduced to match the rapidity challenge of this thesis.

1.1.2/ EDO process conduction

Several approaches could be used to conduct the EDO process. They are more or less advised depending on the extensiveness and complexity of the the problem to solve. The extensiveness is the amount of information defining the problem while the complexity is how complex the relation between those elements are. Problem extensiveness and com-

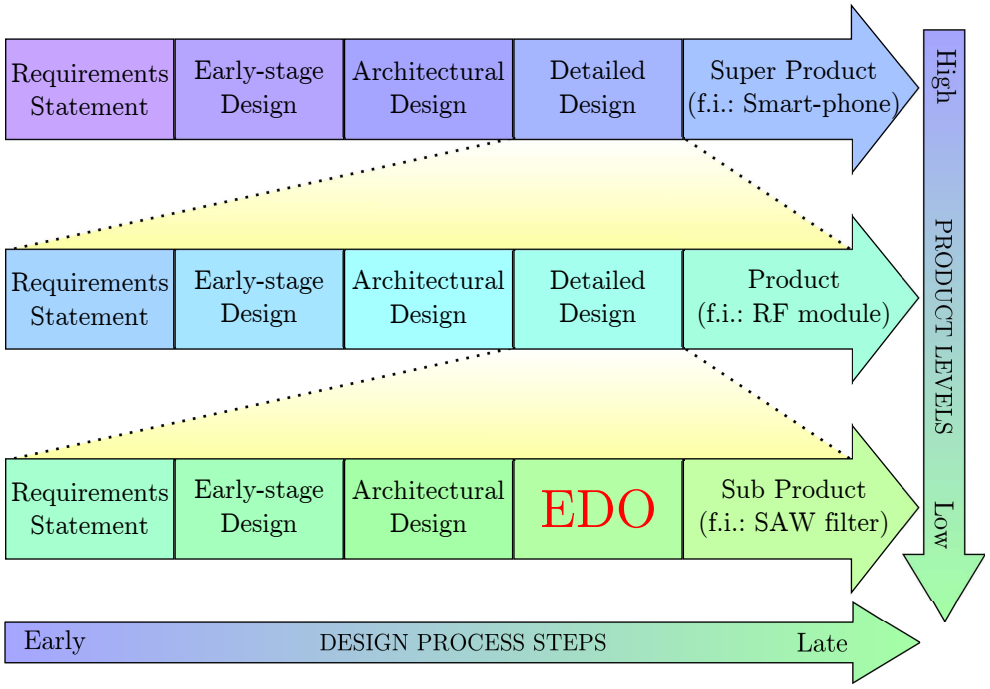


Figure 1.1: EDO process positionning into the design process - Inspired by [21]

plexity can be estimated from table 1.1. This table, which is inspired by the one from [8], presents the EDO problem classifications that makes sense with regards of the problem faced in this thesis. For each classification, the problem could be classified in a group. These groups will influence problem extensiveness and complexity. The different classification presented in this table relies on variables, constraints, objectives and environment properties. Variables, which are detailed in sub-section 1.2.2, are the independent product parameters which can be updated to improve the design of a product. Constraints, which are detailed in sub-section 1.2.4, are the constraints the solution to the optimization problem should fulfill to be accepted. The objectives, which are detailed in sub-section 1.2.3, are the product specifications which need to be minimized or maximized. The environment is the additional properties and information of the problem to solve.

Figure 1.2 positions the different EDO approaches with regards to the problem extensiveness and complexity. For instance, for a problem with a low-extensiveness and a low-complexity an analytic approach is advised.

Different approaches are presented by figure 1.2. When a new technical fields arise, no knowledge about it is available. Thus, problem are complex and extensive leading designer to try random design, which is the 'Randomness' approach. With experience some knowledge will be gathered, and designer will be able to test hypotheses such as the influence of structures or design parameters, which corresponds to medium complexity low extensiveness problem. By doing so designer will perfom optimization by a 'Design of Experiment' approach [8]. Thanks to the knowledge acquired from these experiements, designer will develop an expertise. This expertise, will enable designer to solve complex problem. This holds as long as the extensiveness is not too high, in which case too many phenomenum should be considered at a time. By using its expertise to optimize a product, a designer is performing an 'expert-based' approach [8] which could relies on expert rules or Knowledge based systems [8]. For an experience designer to go further,

Table 1.1: EDO problem classifications - inspired by [8]

Classification base		Values	Problem kind
Variables	Number	≤ 10 > 10	Low-extensiveness High-extensiveness
	Nature	Continuous Discrete Mixed	 High-complexity
Constraints	Existence	Un-constrained Constrained	Low-complexity
	Type	In-equality Equality	Low-complexity High-complexity
	Linearity	Linear Non-linear	 High-complexity
	Separability	Separable Non-separable	 High-complexity
Objectives	Number	1 $[2, 10]$ > 10	Low-extensiveness & Low-complexity Low-extensiveness & High-complexity High-extensiveness & High-complexity
	Modality	Uni-modal Multi-modal	Low-complexity High-complexity
	Linearity	Linear Non-linear	Low-complexity High-complexity
	Continuity	Continuous Discontinuous	 High-complexity
Environment	Uncertainties	Determinist Fiabilist	Low-complexity High-complexity
	Search space	Known Unknown	 High-complexity

it should be helped by advanced computation means. First, by using an 'Algorithmic' approach [8] to face high extensiveness problem, in which case an algorithm is used to find an optimal solution. This approach requires the designer to have enough experience of its fields to set the optimization problem to solve. Second, by 'Neuronal Network' approach [22, 23] to face medium complexity and medium extensiveness problem. In this approach, a neuronal network is trained and used to find an optimal solution. The neuronal network efficiency will depends on the quality of designs and indications the designer could provide to train it. Finally, some aspects of the design might be studied enough to model them as low extensiveness and low complexity problems. In this case, these aspects could be optimized by 'Analytic' approach [24, 25], in which case the optimization problem is fully mathematically formulated and solved analytically.

EDO problems are usually solved by an expert-based approach, however the use of algorithms for design optimization is gaining popularity [8]. Indeed, it partially automates the optimization process and allows a better search for the best design. This approach matches with the extensiveness and complexity of the problems to solve in this thesis.

Therefore, this thesis will be restricted to the EDO problem resolution through the algorithmic approach. By using it, the designer will ask an expert in optimization for assistance. This expert, defined as optimizer in this thesis, is in charge of conducting the

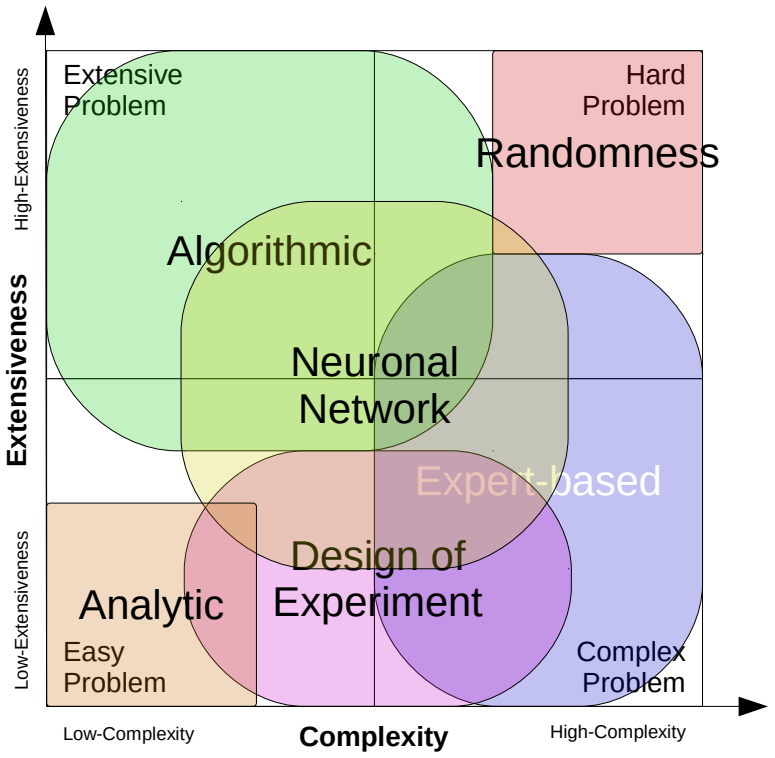


Figure 1.2: The engineering design optimization approaches for the different problems - Based on [8]

optimization process to find a solution satisfying the designer's requirements.

1.1.3/ Concept of an EDO process

The EDO process, being a sub-process of the design process, is introduced in this subsection. An EDO process, represented in figure 1.3, can be decomposed into three main steps as in [9].

- **Formulation:**

The formulation phase, detailed in section 1.2, is the phase during which the optimization problem to solve is defined. During this phase, four elements should be defined: the variables, the objectives, the constraints and the evaluation. This phase is performed by interviews between the designer and the optimizer. During this phase, the problem should be defined leaving the fewest amount of ambiguities [21] as possible.

- **Resolution:**

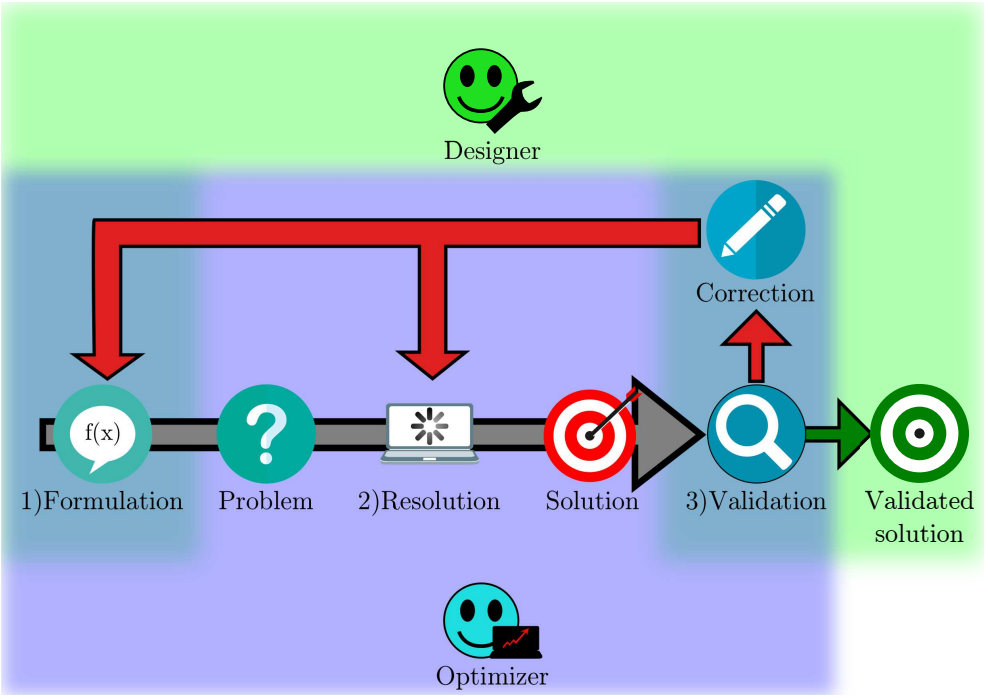


Figure 1.3: The EDO process

The resolution phase is the phase during which an optimization method is set and used to solve the problem. The resolution method is composed of: a bounds handling technique [26], some stopping criteria [9], an optimization algorithm [13] and its setting [27], some eventual simulation tools [9] and an optimization engine [9]. As this thesis is considering optimization from the perspective of engineering sciences and not applied mathematics, information related to computer science such as software, programming languages or library used to develop the optimization engine won't be detailed in this manuscript. For information, the optimization engine used in this thesis has been fully developed by F. Schott and S. Salmon in C# programming language without using already existing optimization library. The core element of the solving method is the optimization algorithm. The solving method and the optimization problem are the two elements forming an optimization case. This case will be run to find a solution to the optimization problem. A run, developed in section 1.3, is a single numerical handling of an optimization case.

- **Validation:**

During the validation phase, the designer decides whether the solution obtained through optimization is validated or not. This phase could be decomposed into several sub-steps. First, the internal verification during which the optimizer validates or not the design. Then, the external verification during which the designer validate or not the design, based on optimization results. Finally, the additional tests during which designer validates or not the solutions based on additional analysis. These analysis are made to test some product specifications that could not be included in the optimization problem, for instance in the case where computing these specifications require to use model too costly in terms of computation time.

1.2/ Problem Formulation

This section will present the first phase of the EDO process, the formulation, which consists in formulating the engineering problem to be solved as an optimization problem following the classical mathematical formulation. In this phase, several hypothesis and choices are made such as: what will be optimized, what design parameters should be chosen as variables or what requirements should be met. As explained by [28], the early stages of a project are the ones influencing the most the final results. Thus, by analogy, the formulation phase, developed in chapter 2, is surely of importance and should not be neglected.

At the end of the formulation phase, the EDO problem is formulated according to a classical mathematical formulation, which is defined in sub-section 1.2.1. This mathematical formulation has a fixed structure that relies on different types of elements detailed in the following sub-sections. The variables, from which objectives and constraints are computed, are presented in sub-section 1.2.2. The objectives, which are to be minimized or maximized, are presented in sub-section 1.2.3. The constraints, which are restricting the problem to satisfying solutions, are presented in sub-section 1.2.4.

1.2.1/ Definition of the optimization problem

From a mathematical point of view, an optimization problem can be formulated as the D-dimensional minimization problem defined in equation 1.1. In this equation, D, the dimension of the problem is the number of design variables X. The vector of objective-functions, F is to be minimized, with respect of the constraints, G and H. The space of possible values for the variables, commonly known as the search space, is noted $\mathcal{S}$.

$$(P)\begin{cases} \min_{X\in\mathcal{S}} \in \ F(X) \\ \quad G(X) \leq 0 \\ \quad H(X) = 0 \\ X = (x_1, \cdots, x_D) \end{cases} \tag{1.1}$$

1.2.2/ Variables

The variables $\mathbf{X} = (x_1, \cdots, x_D) \in \mathcal{S}$ are the independent product parameters which can be updated to improve the design of a product. An optimization problem involves different kinds of variables: continuous or discrete variables. Continuous ones can take any value in a range defined by a lower and an upper bounds (min and max) as follows:

$$X = (x_1, \cdots, x_D) \in \mathcal{S} \Leftrightarrow \begin{cases} x_1 \in [x_1^{min}, x_1^{max}] \\ x_2 \in [x_2^{min}, x_2^{max}] \\ \quad \cdots \\ x_D \in [x_D^{min}, x_D^{max}] \end{cases} \tag{1.2}$$

A discrete variable could take any value of a given set as follows:

$$X = (x_1, \cdots, x_D) \in \mathcal{S} \Leftrightarrow \begin{cases} x_1 \in \mathbb{s}_1 = \left\{x_1^1, x_1^2, \cdots, x_1^{I_1}\right\} \\ x_2 \in \mathbb{s}_2 = \left\{x_2^1, x_2^2, \cdots, x_2^{I_2}\right\} \\ \cdots \\ x_D \in \mathbb{s}_D = \left\{x_D^1, x_D^2, \cdots, x_D^{I_D}\right\} \end{cases} \tag{1.3}$$

Two notions related to the variables are still to be specified: the accuracy and the uncertainty. The accuracy is the expected precision of the solution. The accuracy of the variables can be used as stopping criterion during the run of an optimization problem. The uncertainty corresponds to the manufacturing uncertainty on the variable. The uncertainties are used to perform fiabilist optimization, presented in chapter 6.

1.2.3/ Objective function

In EDO, the objectives, F, are the product specifications which need to be minimized or maximized. By convention an optimization problem consists in minimizing objective-functions, as illustrated in equation 1.1.

- **Single-Objective optimization:** In this case, the vector of objectives $F(X)$ is composed of a single scalar function which will be noted f.
- **Multi-Objective Optimization:** Some real-world problems need to achieve several objectives: Multi-objective Optimization (MOO). In this case, the objective function is a vector defined as follows: $F(X) = (f_1(X), f_2(X), \cdots, f_{N_o}(X))$ where N_o is the number of objectives.

As explained by [29], different techniques could be used to deal with multi-objective problem. A classification of multi-criteria optimization approaches, including multi-objective optimization ones, is presented by [30]. In this classification, fifteen multi-objective methods are listed. These methods are first classified in three categories: Pareto, Outranking and Aggregation. Pareto methods, meant to produce a Pareto front [9], provide multiple Pareto-optimal solutions. On the other hand, outranking methods produce solutions ranked according to their dominance over other solutions, according to objectives values. These two categories are not suitable to find a single solution, which is a compromise between objectives and constraints. Therefore, only aggregation methods, which aims to find a solution making a compromise between objectives and constraints, will be considered. Three methods are mentioned by both [30] and [29]:

- The scalarization method (or weighted-sum method) [9, 29]: This method incorporates multi-objective functions into a scalar fitness function.
- The ε-Constraint method [31]: This approach aims to minimize one objective, say $f_i(X)$, subject to the additional constraint $f_j(x) < \varepsilon_j$.
- The goal programming method [32]: Goal programming is a preference-based classical method for solving multi-objective optimization problems [33].

This thesis will focus on single-objective problems. Indeed, after discussion with the Smart-Inn partners, for the SAW filter design problems considered in this thesis,

the main difficulty is to respect the constraints linked to the bills of specifications which are considered tough. Therefore, the problems will be formulated so that specifications are used to produce highly constrained single-objective problems. However, the methods developed in this thesis are designed to face both single-objective and multi-objectives problems.

1.2.4/ Constraints

In optimization, constraints are meant to restrict the search for acceptable solutions. A solution could be considered unacceptable for several reasons, such as not respecting some physics rules or being inadequate with the specifications. The part of the search space where solutions are acceptable is usually referred as the feasible search space. In a classical optimization problem given by equation 1.1, several types of constraint can be distinguished:

- G which are the inequality constraints; g_i refereed to the i-th inequality constraint.
- H which are the equality constraints; h_i refereed to the i-th equality constraint.
- $[X^{min}, X^{max}]$ which are the lower and upper bounds, as specified in equation 1.2; $[x_i^{min}, x_i^{max}]$ refereed to the bounds of the i-th variable.

Several constraints handling techniques exist. Some of the most common techniques are based on penalties. They are either direct, if they are added to the objective-function, or indirect, if they are substituted to the objective-function. Penalties can be fixed values or functions. An usual constraint handling technique is the Lagrange multiplier, detailed in sub-section 1.2.1. Other constraints handling techniques such as the constraints relaxation one [34] and the constraints propagation one [35] focus on how to avoid constrained part of the search space. The constraints relaxation technique uses the constraints formulation to reduce the space search. The constraints ordering technique requires that the problem is solved by a decision tree in which constraints are added one by one at each step.

When the feasible space is highly reduced by constraints, the optimization problem is highly constrained. On this kind of problems, previously described techniques are not always efficient. In this case, other techniques should be used, such as the constraints satisfaction problem ones [36] or the water-fall objective function one [37]. These techniques often require the constraints to be ranked in order to be taken into account one after another. Constraints are ranked according to their relative importance, how difficult they are to respect and how flexible they can be.

1.3/ Run of optimization

A run of optimization is a single numerical resolution of an optimization problem. During a run, an optimization algorithm will try to find the optimal solution by searching the value of the variables which maximizes the satisfaction. The algorithm will solve the problem through iterations in which one or several evaluations are made. Historically, the evaluation of a solution simply consists in computing the objective function f from the vector of design variables X. However, in an EDO context, the evaluation process might

be more complex. First, the objective and the constraints might be computed through co-simulation [8]. In this case, the simulations could be black-boxes [38]. Secondly, objectives and constraints might be handled through different techniques [10, 39].

In this case, the algorithm does not directly optimize the result of a mathematical objective-function but determines a 'satisfaction'. A satisfaction is a target-value indicating the satisfaction of a designer towards a solution. This satisfaction need to be maximized by the algorithm to obtain the best solution regarding the criteria fixed by the designer.

The satisfaction combines objectives and constraints thanks to a method chosen by the optimizer and set according to the designer expectations. The concept of satisfaction is a generalization of the performance concept introduced in the observation-interpretation-aggregation (OIA) method [21]. The satisfaction will be detailed in chapter 3.

During the formulation phase, the designer should detail how the solution must be evaluated. Several kinds of element should be carefully described: the evaluation tools, the indicators and the satisfaction-related information. The evaluation tools are elements such as the numerical simulation, the analytic model or the computation techniques used during the evaluation process. For each evaluation tool, the designer should explain to the optimizer how to use it. Inputs, outputs, settings, mandatory data files should be discussed. Indicators are information computed during evaluation used to evaluate the product. Satisfaction information are information used to compute the satisfaction of the designer from objectives and constraints.

A run of optimization is an iterative process that could be explained by a workflow, as in sub-section 1.3.1. This process relies on a core element, the algorithm, which performs one or several evaluations at each iteration. The evaluation and iteration concepts are presented in sub-section 1.3.2. A run of optimization will iterate evaluations until a stopping criterion is reached. Stopping criteria topic is explored in sub-section 1.3.3. Once stopped, how well a run of optimization was could be evaluated. Hence, how to evaluate a run will be presented in sub-section 1.3.4.

1.3.1/ Workflow of an optimization run

In the case of non-determinist approaches, the workflow of an optimization is summarized by figure 1.4. First, the variable are initialized. This can be achieved through different means, for instance randomly [40]. Then, iterations, during which one or several evaluations are made, are performed. An evaluation corresponds to the succession of the following steps: the variables serve as input for evaluation tools; the evaluation tools computes objectives and constraints; objectives and constraints are used to compute satisfaction. The satisfactions obtained by the evaluations will be used by an algorithm to choose new solutions to evaluate. Which solutions an algorithm will choose to evaluate depends on its setting which is the set of values given to its parameters. In a theoretical context, the evaluation tools will be mathematical functions while in a real case context, the evaluation tools often are numerical models. Finally, evaluations are made until the algorithm reaches a convergence criterion. In this case the algorithm is stopped and the best solution found is the run value.

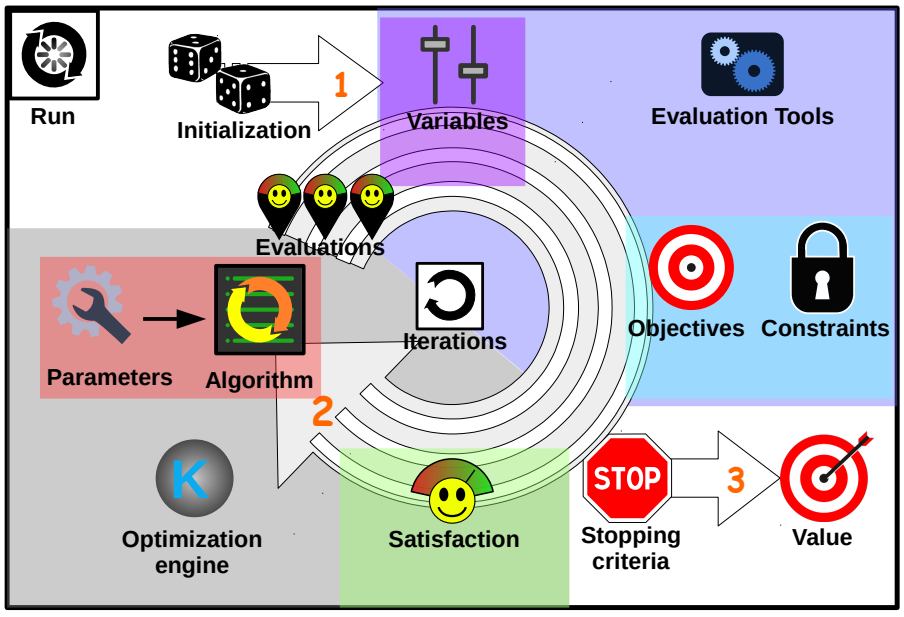


Figure 1.4: A run of optimization working scheme

1.3.2/ Evaluations and iterations

This sub-section presents notions related to evaluations and iterations which will be detailed first. Then, thanks to these notions, the search space landscape notions will be introduced.

As explained in sub-section 1.3.1, an evaluation is a process computing a satisfaction value, from variables, thanks to objectives and constraints values. Evaluations are used by the algorithm to gather information concerning the search space in order to look for the optimum solution. Each evaluation will provide information, such as the satisfaction, at a local point of the search space. These information will be used by the algorithm to choose which points of the search space should be evaluated at the next iteration. During an iteration, evaluations could be done serially, one after another, or in parallel, all at the same time. Performing evaluations in parallel could reduce running time by using more efficiently the available computation resources. However, it is not always technically possible, especially in a co-simulation context.

As explained by [8], an objective-function possesses several mathematical properties, such as multi-modality, separability or linearity. These properties will influence the 'response surface' formed by satisfaction over the search space. This response surface could be considered as a landscape [41], which can be characterize by different metrics, such as the ones presented in section A.10. The algorithm ability to properly explore the search space will depend on the properties of the objective-function and the characteristics of the landscape. They are often used to choose a suitable algorithm and to discuss algorithms results.

1.3.3/ Stopping criteria

When does the algorithm should stop is not a trivial question, especially in EDO as the global optimum value is not necessarily known and nothing can guarantee that the algorithm will be able to find it. Therefore, the stopping criterion should be chosen carefully and correctly set. A stopping criterion is defined by the fact that, once raised, it indicates that the algorithm should stop.

Literature concerning the stopping criteria is scarse [40] and some studies have highlighted the necessity for further researches on stopping criteria in global optimization [42]. The selection of a stopping criterion impacts both the efficiency and the reliability of the algorithm during global optimization [42]. The stopping criterion must be set correctly otherwise the run will fail to converge (criterion considered as too loose) or pointless evaluations will be done (criterion defined as too tight) [9]. It is considered by [40] and [42] that the stopping criterion should be set in a way so that the run must be stopped if one the following situation occurs:

- The algorithm finds the (global) optimum.
- The algorithm is stagnant and it is considered that no significant improvements on the solution can be obtained.

To classify the stopping criterion, [42] proposed a classification based on which element is used as a stopping criterion:

- Error based criteria: The run is stopped when the global optimum has been reached within a given accuracy.
- Exhaustion-based criteria: The run is stopped after a certain amount of time.
- Improvement-based criteria: The run is stopped when the objective-function improvement rate is too low.
- Movement-based criteria: The run is stopped when the the coordinate evolution of the population is too small.
- Distribution-based criteria: The run is stopped when the population concentrates on a region too small.

In [40], the stopping criteria has been classified according to the nature of the metrics used to raised the criteria:

- Threshold criterion: The criterion is raised if a metric is inferior or superior to a given threshold.
- Statistical inference based criterion: The criterion is raised according to statistical distribution of a given metric for the next iteration.
- Fuzzy based criterion: Stopping criterion is raised or not according to fuzzy logic rules over some metrics.

- Other criterion: Other criterion regroup methods that are not classified in any previous groups such as linear regression, Markov chain and automatic termination based ones.

In this thesis, another classification is used to introduce stopping criterion. The main reason of this choice is to be consistent with chapter 4's goal.

- Termination criterion: This criterion indicates that the algorithm should terminate even if it has not converged.
  - Fixed-budget/fixed-cost criteria [43, 44]: The algorithm is stopped after a certain amount of time. As explained by [19], in optimization, time could be measured through different metrics.
  - Fixed-target criteria [43, 44]: The algorithm is stopped when a target value is reached.
- Convergence criterion: This criterion indicates that the algorithm has converged to a search space point supposed to be the optimum one.
  - Algorithm independent: Algorithm independent convergence criteria
  - Algorithm dependent: Convergence criterion that is specific to an algorithm.

Table 1.2 presents a list of usual stopping criteria based on [9]'s one. During this thesis the stopping criteria used are: FEs, Iteration limit, objective-function target and radius.

Table 1.2: Stopping criteria

Classification		Criterion
Termination	Fixed-budget	Function evaluations (FEs) [19]; Iteration limit [9]; (Normalized) Running Time [19]
	Fixed-target	Objective-function target [9]
Convergence	Algorithm independent	Objective function convergence [9]; Population convergence [9]; Radius [45]
	Algorithm dependent	Gene convergence, for GAs [9]; Condition number of a matrix for CMAES; Speed/inertia to low for PSO; Number of iterations without displacement for SA.

1.3.4/ Run review

This section will present how a run is reviewed. This task is important to assess how effective the resolution method is for a particular problem. To evaluate a run, performance measures [46] are used. They are metrics quantifying how well a method performs with regards to an aspect of optimization, for instance convergence speed. First, the choice of using measures of performance over performance profiles will be explained. Then, how a

run could be reviewed will be presented. Finally, performance measures used in this thesis will be introduced.

The performance profile approach is a usual approach advised by [19]. However, when results from different optimization cases are aggregated, which is done in chapter 4, the benefit of using a performance profile compared to measures of performance is limited. Indeed, a performance profile is the evolution of a measure of performance with respect to a criterion [47] and this evolution may have a different shape from one case to another. Therefore, a performance profile obtained by the aggregation of performance profiles from different cases may have a shape that is not representative of the cases considered. As the measures of performance, being scalars, are easier to manipulate and analyse than performance profile, this approach has been chosen for this thesis.

Several performance measure categories have been defined [46]: efficiency, quality and reliability. Quality measures either how good is the satisfaction value found at the end of the run or how fast the algorithm has converged to a solution. Efficiency is a combination of value and convergence qualities. Reliability measures the algorithm statistical quality. From these categories different performance measures could be defined. Some usual performance measures are presented in [48]. For instance, the mean of several runs' final satisfaction could be used to judge the value reliability.

Three performances will be used to review the resolution method performance on a single run: the value quality, the convergence quality and the efficiency. The value quality represents the satisfaction of the designer with regards to the satisfaction value. The convergence quality reviews the convergence speed representing how fast the algorithm converges to a solution. Finally, the efficiency combines the two previous performance measures to review the quality of solution (value) while considering the computational cost (convergence). In addition, two reliability performance measures will be used to review the resolution method over several runs. These performances are: the alpha-reliability and the omega-reliability. The alpha-reliability represents the best efficiency obtained over a large number of runs. The omega-reliability represents the worst efficiency the resolution method is likely to obtain if a limited number of runs are performed. How these performance measures are computed is detailed in chapter 4.

1.4/ Optimization algorithms

This section presents the optimization algorithms which are the core of the resolution method. As explained in sub-section 1.4.1, a large amount of optimization algorithms, which could be classified according to several criteria, exists. In order to develop and test methods in this thesis, some algorithms have been chosen. The chosen algorithms and the reasons of this choice are given in sub-section 1.4.2. In particular, the PSO have been chosen as reference and so a focus will be done on this algorithm in sub-section 1.4.3.

1.4.1/ Optimization algorithms overview

This sub-section introduces the topic of optimization algorithms. First, a few words will be said about the number of optimization algorithms. Then, the way how they can be classified will be discussed. Finally, the main families and how they are composed will be presented.

The number of publications related to meta-heuristic algorithms is given in figure 1.5. It can be observed that this number is exploding. Several explanations to this burst are given by [12] and [13]. As advised by [12], it has been chosen not to focus on developing algorithms but optimization means. The high number of optimization algorithms is a challenge for an optimizer willing to choose the correct algorithm. It also complicates the task of classifying and studying algorithms.

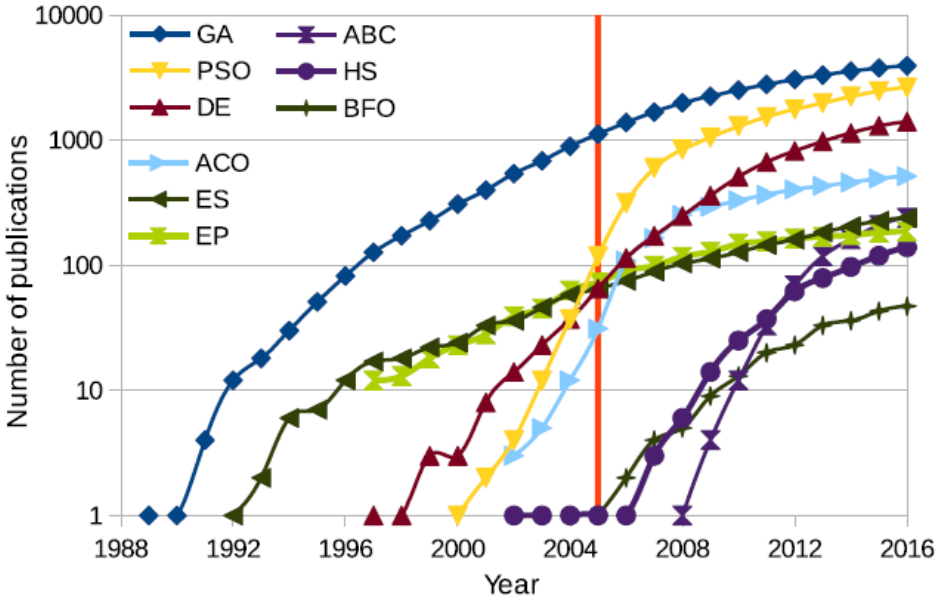


Figure 1.5: Evolution of the number of publications related to meta-heuristic algorithms [13]

Optimization algorithms could be classified in different ways [9]. Some of them are based on the properties of the optimization problem they are meant to solve. Some of them are based on the mechanisms of the algorithms. Here under is a list of common classifications:

- Problem properties based classifications:
  - Heuristic or Meta-heuristic [9]
  - Continous or Discrete
  - Global or Local search [9]
  - Uni-modal or Multi-modal
  - Linear or Non-linear
  - Single-objective or Multi-objective [9]
- Algorithm mechanism [12] based classifications:
  - Stochastic or Deterministic [9]
  - Population-based or Trajectory-based [9]
  - Bio-inspired or Not [9]
  - Descent-direction or Not

- Swarm-based or Not
- Derivate-based or Derivate-free [9]
- Simplex-based or Not

The large number of optimization algorithms is also due to the phenomenon of 'evolution', which consists in modifying an already existing algorithm to improve its performances. This phenomenon could be repeated thus forming a long chain of evolution like in nature. For instance, the RB-IPOP-CMA-ES [49] is the evolution of the IPOP-CMA-ES [50] which is the evolution of the u-CMA-ES [51] being the evolution of the initial CMA-ES [52]. Also, algorithms could be conceived by hybridizing existing algorithms, such as [53] which is an hybrid evolutionary-based method combining the particle swarm algorithm and the chaotic search. The algorithms classification and the evolution phenomenon will lead to the creation of families of algorithms. Figure 1.5 presents the evolution of publications for several families of meta-heuristic algorithms. From this figure it could be seen that some families are extremely prolific such as GA and PSO ones while others are smaller such as the BFO one.

1.4.2/ The algorithms used in this thesis

This sub-section will present the algorithms used in this thesis. First, how those algorithms have been chosen will be explained. Then, they will be briefly described. Finally, their main mechanisms will be explained.

For this thesis, it has been chosen to focus on meta-heuristic, continuous, global search, multi-modal, non-linear, single-objective, stochastic, population-based, derivate-free algorithms. The reasons behind this choice are the following: meta-heuristic to be able to face different problems without having to redevelop an algorithm every time. Continuous as EDO problems are mostly either continuous ones or mixed-variables ones with a majority of continuous variables. How mixed-variables are handled will be detailed in chapter 3. During this thesis we aim to automate the sizing phase so that the designer does not have to provide an initial design to refine. Therefore, the EDO problems to solve are global search ones requiring global search algorithms. They should be multi-modal and non-linear as nothing guarantees the problem to be either uni-modal or linear. This thesis aims to solve optimization problems in order to find the solution that satisfies the designer the most. Therefore, a method finding a single solution instead of several ones must be elaborated. That is why single objective algorithms are looked for. In the case of a multi-objective problem, as explained in chapter 3, an evaluation method making compromises between objectives will be used. Stochastic algorithms will be preferred to deterministic ones as they improve the search space exploration in a complex problem, which is the case here. Parallel evaluations to improve running time, which is important in the co-simulation context of this thesis. For parallel evaluations to be cost-effective, the number of evaluations per iterations should be high enough. As this is guaranteed by population-based algorithms, such ones will be chosen. Finally, as nothing guarantees the objective function to have a continuous first order derivative, especially as the problem are highly constrained, derivative-free algorithms will be preferred.

The algorithms used in this thesis are the particle swarm optimization (PSO) [54], the covariance matrix adaptation evolution strategy (CMAES) [52], the genetic algorithm (GA) [55], the Cuttlefish [56] and the simulated annealing (SA) [57]. More detailed information about these algorithms and their settings are given in section A.7 and in sub-section

1.4.3 for the PSO. They are reference algorithms of different families, meaning that many algorithms of their families are their direct or indirect evolutions. It has been chosen to use reference algorithms to avoid the bias of having more or less advanced representatives for the different families which would be an issue during comparison. However, using advanced algorithms, such as the L-SHADE one [58] is a prospect. A brief description of each of these algorithms is given here:

- The PSO algorithm, detailed in sub-section 1.4.3, belongs to the category of swarm intelligence techniques. In PSO, each solution of the optimization problem is considered as a particle in the search space, adjusting its position according to its own flying experience and the others particles' ones.
- The CMAES, detailed in sub-section A.7.1, derived from the concept of self-adaptation in evolution strategies. The CMA (Covariance Matrix Adaptation) adapts the co-variance matrix of a multi-variate normal search distribution according to two main principles. The maximum-likelihood principle increases the probability of successful candidate solutions at each steps. The evolution path principle contains significant information about the correlation between consecutive step.
- The genetic algorithm, detailed in sub-section A.7.2, is so that a population of candidate solutions, called individuals, is evolved towards better solutions. Each candidate solution has a chromosome, which can be mutated and altered. A chromosome is a set of D genes.
- Simulated annealing, detailed in sub-section A.7.3, is inspired by the process of annealing in metallurgy. Annealing involves heating and cooling a material to alter its physical properties due to the changes in its internal structure. As the metal cools its new structure becomes stable, consequently causing the metal to retain its newly obtained properties.
- The Cuttlefish algorithm, detailed in sub-section A.7.4, mimics the mechanism of the color changing behavior used by the cuttlefish to solve numerical global optimization problems. The Cuttlefish algorithm considers two main processes: rejection and visibility. The population (cells) is divided into four groups. Two of them are used for global search, while the others are used for local search.

A few mechanisms and properties of the algorithms will be referred to along the manuscript. To ease comprehension, they are briefly explained here. First, the population, which is considering the evaluations of an iteration as a population of points in the search space. This population will either be displaced, as in PSO, Cuttlefish or SA, or evolved, as in CMAES or GA. When the population is displaced, it might happen that individuals 'socialize'. Socialization simply means that information are exchanged between individuals. Two important notions, which should be balanced, are exploration and exploitation [59]. Exploration is the process of visiting entirely new regions of a search space, whilst exploitation is the process of visiting the regions of the search space within the neighborhood of previously visited points. Exploration and exploitation are often called 'global search' and 'local search'. Used algorithms include mechanisms balancing exploration and exploitation, For instance PSO displaces individuals both in the direction of the best solution found by the population and in the direction of the best solution found by the individuals.

1.4.3/ Focus on: PSO

This sub-section will focus on the PSO algorithm, as it is the one used for development. Firstly, an introduction of the algorithm is done. Secondly, its particular stopping criteria are given. Thirdly, its pseudo-code is written. Finally, its setting is explained.

1.4.3.1/ Introduction

PSO is a global optimization algorithm described as sociologically inspired. The Particle Swarm Optimization (PSO) algorithm belongs to the category of swarm intelligence techniques. In PSO, each solution of the optimization problem is considered as a particle in the search space, adjusting its position according to its own flying experience and the others particles' ones [54]. The PSO algorithm has only a small number of parameters which need to be adjusted and is easy to implement.

In a basic PSO algorithm, members of a swarm fly in the search field (of D dimensions) and are attracted by their personal best solution and by the best solution of their neighbour [60]. Each particle has a memory storing all data related to its flight (location, speed and its personal best solution). It can also inform its neighbors, i.e. communicate its speed and position. This ability is known as socialization. For each iteration, the objective-function is evaluated for each member of the swarm. Then the leader of the whole swarm can be determined: it is the particle with the best personal solution. The process leads at the end to the best global solution. At each iteration t, the location and speed of one particle are updated as in equation 1.4.

$$\begin{cases} v_{t+1} = \omega_t v_t + c_1 r_1 (p_{best} - X_t) + c_2 r_2 (g_{best} - X_t) \\ X_{t+1} = v_{t+1} + X_t \end{cases} \tag{1.4}$$

In equation 1.4, p_{best} is the personal best previous position of the particle and g_{best} is the best global position among all the particles in the swarm (figure 1.6).

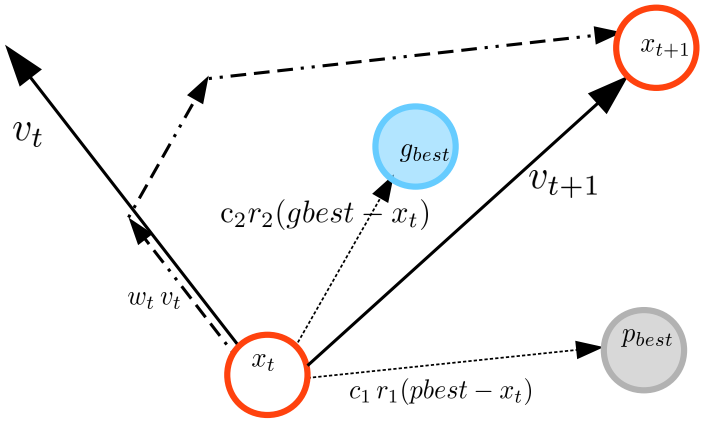


Figure 1.6: PSO particles displacement

The parameters r_1 and r_2 are two random numbers between 0 and 1. The constant c_1 and c_2 represent trust parameters indicating how much confidence the current particle has in itself and how much confidence it has in the swarm. These acceleration constants c_1 and

c_2 indicate the stochastic acceleration terms which pull each particle towards its own best position and the swarm best one. The role of the inertia weight ω is considered important for the convergence behavior of the PSO algorithm. The inertia weight is employed to control the impact of the previous history of velocities on the current velocity. Thus, the parameter ω regulates the trade-off between the global (wide ranging) and the local (nearby) exploration abilities of the swarm. A proper value for the inertia weight provides balance between the global and the local exploration ability of the swarm and thus results in better solutions. Numerical tests imply it is preferable to initially set the inertia to a large value, to promote global exploration of the search space and to gradually decrease it to obtain refined solutions. The weight of particles is decreasing through time in order to fit the search area shrinking. The weight is decreasing according to the equation 1.5 law:

$$\begin{cases} \omega_t = \omega_{max} - \Delta \cdot (t/t_{max}) \\ \Delta = \omega_{max} - \omega_{min} \end{cases} \tag{1.5}$$

1.4.3.2/ Stopping criteria

This original version has been modified to include a convergence criterion, the radius improvement as described in [45]. The PSO stops if, for r_t iterations, evaluated points are contained inside a radius, meaning that, for each dimension, equation 1.6 is verified. The radius notion is displayed in figure 1.7.

$$\forall < i, j, k >; point_i[x_k] - point_j[x_k] \leq 0.01 \cdot (x_k^{max} - x_k^{min}) \tag{1.6}$$

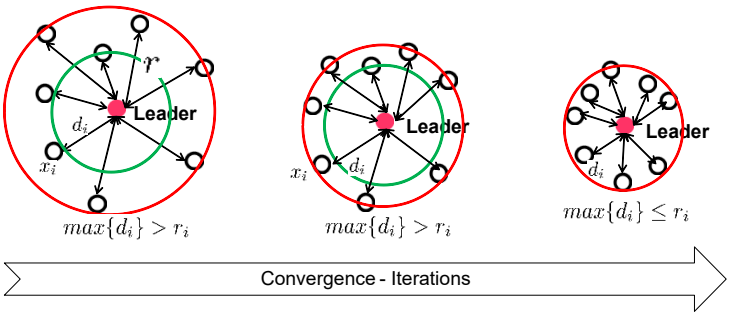


Figure 1.7: PSO radius stopping criterion

1.4.3.3/ Pseudocode

PSO pseudo code is given by algorithm 1.

1.4.3.4/ Setting

Table 1.3 gives PSO default setting used in this thesis. This setting has been defined by an expert in optimization and is also based on articles [45, 61, 62, 63]. Chapter 5 aims to improve PSO efficiency by optimizing its setting. Therefore, other settings than the one presented here could be used in this thesis.

Algorithm 1 PSO pseudo code

Require: Initialization

1: Initialize N particles: random position (X), weight (w) and random velocity (v)
2: Evaluate particles (f)
3: Find particles'personal best (p_{best}) and global best (g_{best})
4: Compute swarm radius (R)
5: **while** $t \leq t_{max}$ and $R < R_v$, the radius threshold **do**
6: Evaluate particles (f)
7: Update personal best (p_{best}), global best (g_{best}), velocity (v), weights (w) and position (X)
8: Compute swarm radius (R)
9: **end while**
10: **return** g_{best}

Table 1.3: PSO default setting

Parameter	Value	Remarks
t_{max}	$t_{max} = FEs/N$	Iteration limit
N	20	Number of particles of the swarm
c_1	1	Personal best factor
c_2	1	Global best factor
w_{min}	0.4	Minimal inertia factor
w_{max}	0.9	Maximal inertia factor
R_t	10	Number of iteration inside radius to stop
R_v	1e−3	Radius threshold value

1.5/ Robustness and Fiability

This section will develop two important notions required to understand this manuscript: the robustness and the fiability. The robustness, which is defined in sub-section 1.5.1, represent how well an algorithm performs on a problem when no uncertainties are taken into account. The uncertainties, which are presented in sub-section 1.5.2, will induce a difference between the theoretical solution and the practical one. When uncertainties are taken into account, how well an algorithm performs on a problem is linked to its fiability, which is defined in sub-section 1.5.3.

1.5.1/ Robustness

The robustness is the ability of an algorithm to converge to the global optimum. This idea is summarized by figure 1.8. The robustness of an algorithm will impact value quality performance measure of a run (see sub-section 1.3.4). The robustness is one of the major criteria for an algorithm selection. However, robustness is not taking uncertainties into account.

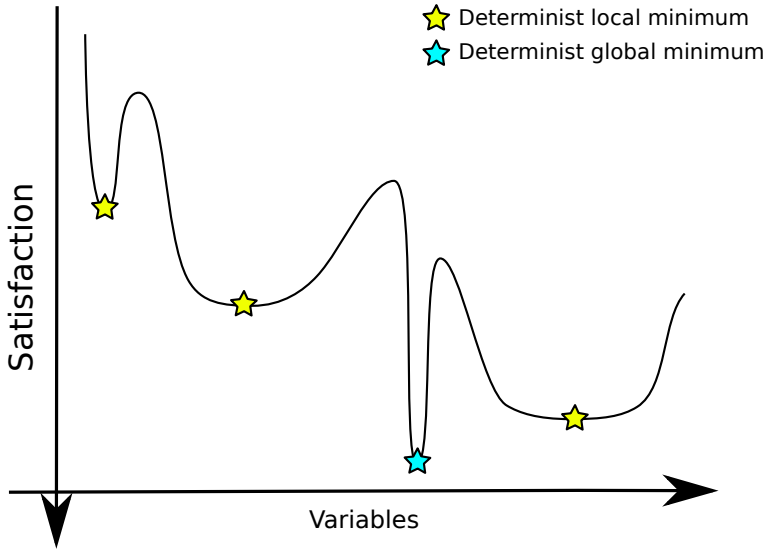


Figure 1.8: Robust optimization minimums

1.5.2/ Uncertainties

The uncertainties [64] are elements making a difference between the theoretical solution and the practical one. Uncertainties could have several sources [65]: estimated problem information, measure problem information, solution implementation uncertainties. Uncertainties could be classified into different types [64]:

- uncertainties on parameters: some parameters values are stochastic or different from the real ones.
- uncertainties on variables: difference between the exact optimal variable value and the one on the manufactured product.
- uncertainties on model: the numerical model used to optimize the product does not perfectly represent the reality
- uncertainties on limits / constraints: the limits / constraints values could be dependent to uncertain factors

Another uncertainties classification system, based on product life cycle, has been proposed by [66]:

- design phase uncertainties: Model or data related uncertainties
- Manufacturing: Environment related uncertainties
- Product aging: Product properties, such as material related ones, may change in function of time

In this thesis uncertainties are taken into account to reduce the manufacturing failure ratio. Therefore, only uncertainties related to manufacturing phases will be considered. In

addition, due to co-simulation restrictions, only uncertainties on variables will be considered. In order to take these uncertainties into account, the classical formulation (equation 1.1) has to be updated, leading to equation 1.7. In this equation, an uncertainties vector δ is randomly drawn, thanks to a normal distribution law U, into an interval defined by the variables uncertainties vector Δ.

$$\begin{cases} Min : F(X, \Delta) \\ P[G(X, \Delta) \leq 0] \geq \mathcal{X}\% \\ P[H(X, \Delta) = 0] \geq \mathcal{X}\% \\ x = (x_1, \cdots, x_D) \in \mathcal{S} \\ \forall i, A_i = [x_i - \Delta_i, x_i + \Delta_i] \end{cases} \tag{1.7}$$

By taking uncertainties into account, fiabilist optimization, by opposition of deterministic one, is performed. In the case where only uncertainties on variables are considered, Fiabilist optimization could be summarized by figure 1.9. To perform fiabilist optimization, fiabilist resolution method should be used and an algorithm should be chosen for its fiability.

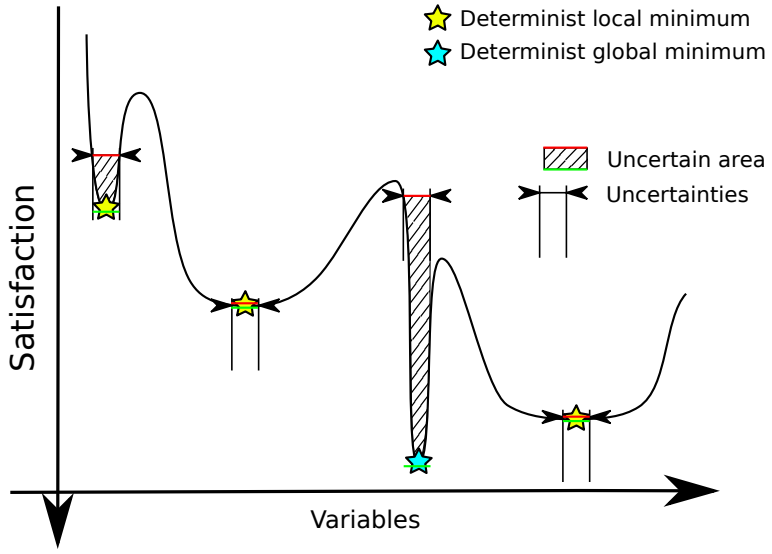


Figure 1.9: Optimization under uncertainties: the fiabilist optimization

1.5.3/ Fiability

The fiability is the ability of an algorithm to converge to the global fiabilist optimum. This idea is summarized by figure 1.10. To find the global fiabilist optimum, fiabilist evaluations should be made, as it will be discussed in chapter 6. The fiability of an algorithm will impact the value reliability performance measure of a run (see sub-section 1.3.4).

1.6/ Conclusion

This chapter intends to introduce optimization so that research work presented in the following chapters could be understood and appreciated.

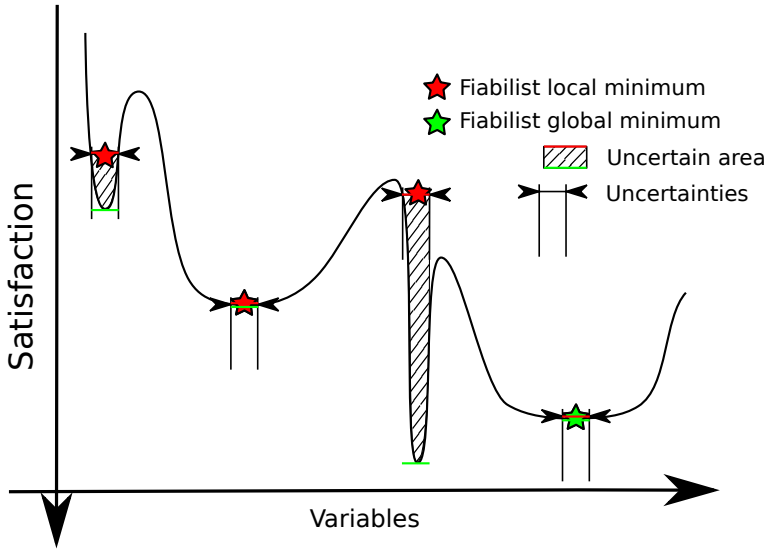


Figure 1.10: Fiabilist optimization minimums

In this thesis, the branch of optimization considered is the Engineering Design Optimization (EDO) which consist in optimizing an engineering product sizing. The EDO process takes place at the sizing phase of a design process. In this thesis case, the algorithmic approach will be used. An EDO process which is composed of three steps: the formulation, the resolution and the validation. An EDO problem is mainly composed of variables, objectives and constraints. A numerical resolution of an EDO problem is a run of optimization. During a run, the algorithm will solve the problem through iterations in which one or several evaluations are made. The algorithm will attempt to find the values of the variables maximizing the satisfaction, which represents the designers' satisfaction toward a solution. The satisfaction is computed through a process designated as an evaluation. The algorithm will follow an iterative process until a stopping criterion is reached. A run of optimization could be evaluated in terms of value, convergence or efficiency. Many optimization algorithms, which could be classified in different ways, exists. For this thesis, it has been chosen to focus on meta-heuristic, continuous, global search, multi-modal, non-linear, single-objective, Stochastic, population-based, derivate-free algorithm. The algorithm used in this thesis are PSO, CMAES, GA, Cuttlefish and SA. It has be chosen to focus on PSO to develop methods in optimization. Finally, The robustness and fiability has been explained.

The explanations given in the following chapters will rely on the notions developed here. For instance, understanding chapter 2 will require the problem formulation notions developed in section 1.2. The state of the art presented in this chapter is not comprehensive but consistent as a requirement for the understanding of this thesis. More information about EDO in general could be found in [67, 68].

2

A framework to formulate engineering design optimization problems

Contents

2.1/ Introduction

As mentioned in chapter 1, engineering design optimization (EDO) takes 50% of the design process timespan[8]. This timespan is considered too long with regards of the industrial constraints. It is necessary to reduce this phase which is one goal of this thesis.

An optimization process could be decomposed into three phases as defined in [9]. The first one, the formulation, consists in formulating the optimization problem to solve. The designer, who is designing the product, and the optimizer, who is in charge of the optimization of the product, will communicate to allow the optimizer to mathematically formulate the problem. The second phase, the resolution, consists in setting up a resolution method to use it to solve the problem. These tasks will potentially be iterated until the optimizer finds a suitable solution or if the problem is considered unsolvable without modifications. The third phase, the validation, consists in checking if the solution meets the designer's expectations and if it doesn't, anticipate on what should be done. Usually, if the designer is not satisfied, the optimizer should come back to a previous phase of the optimization process in order to make modifications.

During the validation phase, solutions could be rejected by the designer because of ambiguities made during the formulation phase. Ambiguity [21] corresponds to the deviation between the costumer's expectations and the problem to be solved. When a solution is rejected due to ambiguities, both designer and optimizer would have to go back to the formulation phase of the process. In order to reduce the EDO time span, ambiguities must be avoided.

Currently, the formulation phase is usually performed through conversations between designers and optimizers. These conversations are performed according to the optimizer expertise. There exists no established method nor guidelines to help. Still, the Observation-Interpretation-Aggregation (OIA) method [21] could be mentioned as it tends to solve optimization problems according to the designer's will. This is done by using an advanced solution evaluation method. For the formulation phase, the OIA method relies on interviews but does not provide any framework to conduct them.

The aim of this chapter is to present a framework based on interviews to prevent ambiguities. Defining a protocol to properly test this framework is a considerable work which has to be done independently. Thus, the proposed framework will not be tested here.

The proposed framework relies on interviews which are conducted thanks to a questionnaire and some synthesis documents. An overview of the framework is done in section 2.2. Each of the interviews composing this framework have different goals. The interviews will be detailed in section 2.3.

2.2/ Overview of the proposed framework

This section intends to present the formulation framework. This framework is based on a succession of interviews, presented in sub-section 2.2.1. To conduct them, the optimizer will use a questionnaire, composed of three parts, introduced in sub-section 2.2.2. The synthesis documents, described in sub-section 2.2.3, will be filled with the designer's answers.

2.2.1/ Interviews presentation

The proposed framework is composed of three interviews which each lasts approximately two hours. The first one explores the problem environment and studies the problem context by thematic. The second defines the optimization problem elements by investigating thematics of interest. The third one is a discussion to correct and then validate the formulated problem.

The first interview explores the problem environment and serves several goals:

- Providing to the optimizer the knowledge to understand the designer's explanations and the opportunity to ask for details.
- Determining thematics of interest. Some might not have been considered by the designer yet.
- Providing to the designer the confidence towards the optimization and giving him a wider understanding of the problem.

To guide the exploration and avoid forgetting a topic, two documents are used: the thematic board and the first part of the questionnaire. The thematic board, presented in sub-section A.1, is a board depicting thematics related to the product. It provides both the designer and the optimizer a visual reminder of thematics that could be explored. On the opposite, the questionnaire is to be used only by the optimizer. It is a non-exhaustive collection of questions, structured by main thematics. The questions are written so that they require no knowledge in optimization to be understood.

The second interview investigates elements of an optimization problem. This investigation follows two steps:

1. Identification of aforementioned elements, such as objectives, constraints or variables.
2. Definition of the elements. For instance, what are the bounds of the i-th variable.

For the first step, the thematic board and the second part of the questionnaire will be used. The second part of the questionnaire is structured first by types of elements and then by main thematics. As for the first interview, a non-exhaustive collection of questions is used. They are written to be understandable without any particular knowledge in optimization. One should keep in mind that there is no need to investigate every thematic. How much each thematic should be investigated depends on topics of interest and information gathered during the first interview. For the second step, the third part of the questionnaire provides, for each type of element, a list of specific understandable questions. For this step, every question should be addressed. They are meant to collect data to solve the problem according to the normalized evaluations method, described in chapter 3.

The last interview is needed to discuss, correct and validate the formulated problem. This interview gives the opportunity to raise questions left unsolved during the two previous interviews. Questions might have been left unsolved as it requires the designer to investigate some topics or because some piece of information have to be queried. The conduction of the interviews should remain flexible and depends on how the first two interviews have been done. Therefore, no instructions will be given on how to perform it. However, two points should be kept in mind.

- It should be an open discussion between the designer and the optimizer.

- At the end of the third interview, the optimization problem which has been formulated should be validated.

2.2.2/ Questionnaire

This sub-section presents the questionnaire used to guide interviews and also its structure and its key points. Some elements will be provided as examples to illustrate points. However, it will not be presented in detail here as it could be found in sub-section A.2. The first part is used in the first interview whereas the second and third parts are used in the second interview.

The first part contains questions helping to explore the product environment. It is sub-divided into sections corresponding to the main thematics besides another one for the product itself. The sections are ordered to reproduce an informal conversation going from general information to technical details. First questions will be a company overview, for instance: 'What are the company main activities?' or 'How many employees?'. The questions concern several topics such as investigating product, customer, working scheme and production. Finally, environmental impact, usage, costs and design thematics will be explored. The last questions will be more technical and focused such as 'Are you using some simulation, model or software?' or 'What technology does this product use?'. For every sections, the questions, at first, concern the main thematic and then become more specific broaching on sub-thematics. From time to time, this rule can broken to improve the fluidity of the conversation. It should be noted that no matter the order of the sections and questions, the optimizer should adapt to the designer: questions are just a guide not a process to follow step by step.

The second part contains topic suggestions to determine the elements of the optimization problem. This part is sub-divided into three sections: objectives, constraints and variables. The section order has been chosen as it is the easiest one for the designer to express item. In each section, a pattern is repeated. First a general question to help the designer to figure out what is expected. For instance, in constraint case, 'What should be good enough for the product to be accepted?'. Then, for each main thematic, an alternative version of this question is followed by a few suggestions. The suggestions of the questionnaire are general ones based on sub-thematics which should be completed thanks to the information gathered during the first interview. For instance in a usage thematic, a question to ask in order to identify the objectives can be 'The most efficient product?' and suggestions are 'The finest sizing', 'the best performances' and 'The most confortable'. Deciding how much main thematics should be explored depends on information collected during the first interview and the designer's answers.

The third part contains specific questions to precisely define the elements of an optimization problem. They are organized by element type: objectives, constraints and variables. For each element type, a list of questions is written in the designer's language. Each one of them is meant to get a specific piece of information. They are listed to gather information from the most to the less general ones. For instance to ask about an objective if its value is to be minimized or maximized, the question is 'Do you want the most or the less of this?'.

2.2.3/ Synthesis documents

To formulate the optimization problem, the information given by the designer to the optimizer should be carefully noted. To avoid omissions and to help the optimizer in this task, several synthesis documents are provided along with the framework. One should know that using synthesis documents is not mandatory. Any means available at the optimizer's convenience could be used. The answer documents are the thematic board, the answer form and the optimization bill of specifications (OBS).

Thematic board, detailed in sub-section A.1, is a board on which thematics are depicted by labeled icons. It is used to remind thematics to the designer and to the optimizer. Also, annotations could be made on this document to highlight the thematics of interest and add useful information to prepare the second interview.

The answer form is meant to keep a record of answers given during the first interview. It will be used to prepare the second interview and it could be referred to. Its structure mimics the one of the questionnaire. The fields of the answers are adapted to the questions. For instance, for open questions, like product manufacturing, a blank space will be left whereas for a company position on the market multiple choices are proposed. Still, the answer form is designed to be as wide as possible.

The OBS is a short and technical document summarizing requirements for optimization-related specifications. The information contained in this document will be used, during the validation phase of the optimization process to decide whether the solution found is accepted or not. If it respects the OBS, it will be rejected only due to ambiguities or due to additional post-optimization analysis. This document is filled with the designer's answers to the third part of the questionnaire. In this thesis, some topics, for instance multi-objective handling method choice, are not considered. The topics which are out of this thesis scope are not considered by the proposed framework. Therefore, the OBS presented in sub-section A.3 is not comprehensive.

2.3/ Interviews in detail

This section intends to detail how the interviews are conducted based on questionnaire. The first interview which aims to learn more about the product will be detailed in sub-section 2.3.1. The second interview which purpose is to define the elements of the optimization problem will be detailed in sub-section 2.3.2. The third interview which is a review of the problem formulated in the second interview is presented in sub-section 2.3.3.

2.3.1/ First interview: Learn about the product

This sub-section will detail the first interview whose aim is to learn more about the product. First, some indications about the content of the interview will be given. Then, thematics will be motivated. Finally, the questions set will be explained.

The first interview is conducted by the optimizer using the first part of the questionnaire. This part is structured by sections following the main thematic of the thematic board. The idea of using thematics to help the designer and the optimizer to share knowledge originates from Charrier's knowledge guides [17]. They are meant to help experts from different fields to distribute knowledge about a product by exploring pre-defined thematics.

For the first interview, the sections are ordered to reproduce an informal conversation.

To define a comprehensive and properly structured list of thematics, they would have to be defined thanks to a review of application papers and then wisely organized. This task will require a considerable amount of work to be done by a research group composed of experts in both applied optimization and cognitive science. Thus, for this thesis, expectations will be lowered to the definition of a motivated list of thematics. One should note that this list of thematic could be amended by the optimizer and the thematic board modified in consequence.

The thematics have been thought as in a multi-level pie chart:

- Level 0: The product thematic: gathers all basic information about the product itself.
- Level 1: 8 main thematics which have been chosen to introduce the product environment to an optimizer.
- Level 2: 24 Sub-thematics: For each main thematics three sub-thematics have been chosen. They are meant to explore the main thematic related topics.

The list of thematics, divided by main thematic, is given hereunder. For each thematic is given, a non-exhaustive list of motivation and a reference of a work using this thematic.

1. Entreprise (Company) [21]: Showing interest for the company might make the designer feel confident. Also, company-related information could be used for commercial aspects. Knowing better the company could help to adapt the resolution method. For instance, by knowing the design process timespan, the allowed optimization process timespan could be estimated.

- Modularité (modularity) [69]: The product modularity is defined by how easily a product could be modified by changing some of its sub-products. This thematic explores the possibility to optimize a product part by part, as in some multi-level optimization methods.
- Bénéfices (Profit) [70]: This thematic is about optimizing the company benefits.
- Polyvalence (Versatility) [71]: A versatile product, which could be used for different tasks, could be sought by a company.

2. Produit (Product) [72]: The questions of this section will serve as an introduction to the product itself. They are meant to know more about the product itself. This could help the optimizer to conduct the interview.

3. Client (Customer) [73, 74]: The questions of this section were designed to be more acquainted about the customer's expectations towards the product.

- Ergonomie (Ergonomy) [75]: If the ergonomy of the product matters, it could impose restriction over the product characteristic that might be considered during optimization.
- Prix (Price) [76]: Price is a potential factor in the customer's choice. Therefore, the price of the product could be one indicator of the optimization problem.

- Esthétique (Aesthetic) [77, 78]: Aesthetic is one of the factor motivating customer choices. It could be included inside the optimization process.

4. Fonctionnement (Working scheme) [79]: For the optimizer to understand the designer's technical explanations, it is important to clarify the working scheme.

- Conditions d'utilisation (operating conditions) [80]: The designer could specify the operating conditions that could become constraints. Also, the designer could be willing to optimize the range of the operating conditions.
- Cahier des charges (bill of specifications) [81]: The bill of specifications of the design process could be helpful to define the OBS.
- Sécurité (safety) [82]: Safety is a potential source of constraint.

5. Production (Production) [83]: Production is the complete process to produce a product, including sourcing, manufacturing and packaging. Some uncertainties are due to the production, therefore this thematic should be discussed.

- Usinage (Machining) [84]: Machining is the process to use machines to make parts from raw materials. The choice of a machining strategy could be optimized. Also machining is a source of uncertainties.
- Matériaux (Materials) [85]: Materials could be part of an optimization problem. For instance, if the material of the part of a product is to be defined, the choice for material will become a variable.
- Entretien (Maintenance) [86, 87]: Maintenance operations could be a critical operation for some products. Therefore, maintenance-related elements of the product could be optimized.

6. Impacte énergétique et environemental (Energy and environment) [88, 89]: Nowadays, energy and environment topics are considered during the design process.

- Recyclage (Recycling) [90, 91]: Some products could be challenging to recycle and thus can be optimized to ease such an operation.
- Autonomie (Autonomy) [92]: Autonomy could be a challenge for some products, such as electric cars, which would be optimized to be competitive with regards to this problematic.
- Consommation (Consumption) [93]: Energy consumption, such as fuel consumption in air transport, could be an important problematic.

7. Usage (Usage) [94]: How the product is used should be asked as it could help to identify objectives, constraints and variables.

- Caractéristiques (Characteristics) [95]: The technical characteristics of a product are elements describing how the product has been conceived, such as geometries, materials or energy source. These characteristics are potential variables influencing the performances of the product.
- Performances (Performances) [96]: The technical performances of a product are elements that could be used to judge the product, such as physical and technical properties or could measure how well a task is performed. They are potential objectives and constraints.

- Confort (comfort) [97, 98]: If a customer has to use a product often or over extended periods of time, comfort will be a key topic.

8. Coûts (Costs) [99]: An usual objective in optimization is cost reduction [8].

- Chute (Waste) [100, 101]: The product costs could be decreased by reducing the wastes.
- Rebut (Failed parts) [64]: Diminishing the failed parts ratio could be a goal for the designer as the higher the ratio the higher the cost of a product.
- Fabrication (Manufacturing) [102]: Manufacturing is the process of machining parts, assembling parts and components to get the product and testing it. Manufacturing means could be integrated to the cost and could be optimized.

9. Conception (Design) [103]: Explanations about how the product is designed could highlight constraints due to physics or technology. It also introduces possible evaluation tools. Finally it could be used to set a more efficient resolution method by using human knowledge effectively [8].

- Adaptabilité (adaptability): A designer might be willing to design an adaptable product which will perform a single task in a large variety of environments.
- Technique (Technique): To correctly understand the performances and characteristics of the product, it might be important to clarify the technologies used by the product.
- Physique (Physics) [104]: The Physics behind the problem could have to be investigated. In some cases, the optimizer could have to implement the computation of some indicators, requiring to understand the physics behind the problem.

Questions are organized by main thematics. For each main thematic, a few questions are linked to the main thematic itself and one or a few questions are given for each related sub-thematics. The goals of the questions and their implications are:

- Providing information to understand the designer and to conduct interviews: Some questions should be meant to ask the explanation of a topic.
- Providing information to prepare the second interview (topic of interests, key information): Some questions should be meant to ask about the expectations of the designer or about the reference and order of magnitude.
- Being easy to understand: Use the designer's language and not the optimizer's one.
- Being a relatively restricted number and not redundant: No more than a few questions by thematics.

Based on these remarks, for every main thematics, a list of questions has been written.

2.3.2/ Second interview: Optimization elements definition

This sub-section will present the second interview whose purpose is to define the elements of the optimization problem. This interview is composed of two phases. The first one aims to find the optimization elements by using the second part of the questionnaire. The second phase focuses on detailing the optimization elements by using the third part of the questionnaire. The answers to this phase will be noted in the OBS.

For the first phase, the elements will be looked for in the following order: objectives, constraints, variables and evaluation tools. For each element types, a list of suggestions organized by main thematics could be used. Also, all suggestions are thematic-related answers to a generic question. For instance, the objective generic question is 'What should be the best possible?' and the Production-related suggestion is 'The easiest way to manufacture the product?'. Which suggestions should be used or not depends on the thematics defined as interesting during the first interview. For each main thematic, is given: a suggestion for the main thematic itself and one suggestion by related sub-thematic. If the designer agrees with the suggestion, the optimizer would ask what would make the suggestion true. The answers to that question are supposed to be the elements of the optimization problem. For instance:

- If the designer is looking for 'The easiest way to manufacture the product'
- The optimizer would have to ask 'what would make the product the easiest to produce?'
- The designer would, for instance, give answers such as 'reducing material consumption as much as possible'.

During the second phase, the elements will be detailed one by one. The third part of the questionnaire provides, for each type of element, a list of questions to gather information. For every piece of information that should be collected for an element to be defined, a corresponding question is given. The optimizer simply has to ask every questions and must fill the OBS according to the answers. The list of questions is 'comprehensive' with regards to the context of this thesis excluding some difficulties such as multi-objective method choices. In optimization, considering additional difficulties requires additional questions to be added. In this case, it would be consistent to keep the initial idea: one piece of information, one question.

2.3.3/ Third interview: Final discussion and corrections

This sub-section will present the third interview, which is a review of the problem formulated during the second interview. This interview, on the opposite of the two previous ones, is not guided. It is a discussion between the designer and the optimizer to serve several purposes:

- Giving a time to both the designer and the optimizer to ask technical questions.
- Clarifying some points.
- Correcting and then validate the OBS.

- Discussing the possible reasons of failure and what should be done to face them.
- Eventually, discuss some commercial aspects.

2.4/ Conclusion

In order to reduce the timespan of an EDO process, ambiguities should be avoided. A framework based on interviews has been developed in order to do so. This framework is based on three interviews conducted by the optimizer thanks to several documents: the thematic board, the questionnaire and the optimization bill of specifications. At last, during the first interview, the optimizer will learn more about the product by asking questions to the designer. During the second interview the designer will define the elements of the problem, by exploring thematics of interest. During, the third interview, the designer and the optimizer will discuss the formulated problem.

The choice of thematics has been motivated and a method to improve it has been proposed. How the questions of the questionnaire have been defined has been explained. The lists of questions used to precisely define the elements of the problem are comprehensive with regards to the context study of this thesis. To test this framework, a consistent evaluation protocol should be designed. This represents a consequent work which should be done apart.

The development of the framework requires knowledge in cognitive science and is addressing a topic that as not been investigated much. This explains why proposing a framework, that has not been tested yet, could be considered as a complete research work in itself. To improve the framework, it could be redesigned with the help of an expert in cognitive science whom could for instance consider other approaches such as the ontology one.

3

The normalized evaluations approach

Contents

3.1/ Introduction

Two difficulties can slow down a large-scale deployment of optimization techniques in an industrial context: the design process time span and the quality of the solution. As mentioned in chapter 1, the engineering design optimization (EDO) process timespan is half of the one of the design process [8]. During an EDO process, the resolution phase [9] might be performed several times thus increasing the process time span.

Moreover, several resolutions can be necessary if the obtained solution is biased. In this case, the algorithm converges to a solution which does not satisfies the designer. To take into account the designer's preferences and avoid biased solutions, two challenges, among others, should be faced.

- The first one is to have mixed variables with different orders of magnitude which might lure some mechanisms of the algorithm. Value domains can be continuous or discrete. Their boundaries are defined by designers from design requirements documents and also from their expertise.
- The second one is the introduction of the concept of level of satisfaction. Decision-makers seek to achieve a satisfactory trade-off between the risk and the expected performance in their decisions. Classical satisfaction computation method, such as the Lagrange multiplier one, while correct from a mathematical point of view, might lead to bias satisfaction. Indeed, such methods relies on penalties computed from the values of the constraints using linear or quadratic function. Those functions might not represent correctly the design's un-satisfaction toward an unsatisfied constraint. It may lead to a solution which is not the best from a mathematical point of view but is not from the designer's perspective.

The variables challenge has not been solved yet. However, some interesting works have been proposed to efficiently manipulate the search space in order to ease the task of the algorithm. For instance, a regression-based sensitivity analysis in order to reduce the search space has been performed by [105]. Yet, no work considers using a transformed search space in which optimum research will be eased. On the opposite, an interesting way to face the satisfaction challenge has been proposed by [21]: the OIA method (Observation-Interpretation-Aggregation). This method computes a satisfaction by aggregating normalized measures of performance. Some product specifications are observed. Then, from these observed specifications, performances are interpreted. Finally, performances are aggregated to produce a satisfaction.

In this chapter, a new approach is proposed to attempt to reduce these problems: the normalized evaluations (NE) one. In this approach, the solution of the optimization problem is evaluated in an original way. First, it uses normalized variables to allow different nature of design variables. This point facilitates the algorithm search for an optimum. Second, a new satisfaction indicator is proposed based on a 'normalized objective function'. This method uses normalized constraints and objectives as in OIA method. On the other hand, it uses the Lagrange multipliers framework. The evaluation is 'normalized' because its inputs, the variables, are continuous between 0 and 1 and because its output, the satisfaction, is continuous between -1 and 1 for all solutions inside the search space. The NE approach novelty relies in the use of an intermediary normalized variables space and in the process of satisfaction computation which normalizes every constraints and objectives independently in addition of using a specific aggregation process.

With the NE approach the classical evaluation process and the classic formulation are modified. The NE approach overview, done in section 3.2, explains these modifications. These modifications will be required to face the variables and satisfaction challenges. How variables are normalized and transformed to face the variables challenge will be explained in section 3.3. How the satisfaction is computed to face the satisfaction challenge will be detailed in section 3.4. Finally, the NE approach will be tested in section 3.5.

3.2/ NE approach overview

This section makes an overview of the NE approach. This overview aims to help understanding how the variable and satisfaction challenges are faced in sections 3.3 and 3.4. To do so the classical evaluation process, described in sub-section 1.3.2, has been modified leading to the working scheme presented in sub-section 3.2.1. This new process requires some additional information about the problem, which are presented in sub-section 3.2.2. Moreover, as explained in sub-section 3.2.3, the problem formulation has been updated.

3.2.1/ NE workflow

The NE approach is illustrated in figure 3.1 and includes two different steps which are explained below : variables challenge and satisfaction challenge.

- Variables challenge (section 3.3):
  - Determination of the normalized variables (sub-section 3.3.1).
  - Determination of the bounded variables by using bound handling techniques (sub-section 3.3.2).
  - Transformation of the bounded variables into the initial ones (sub-section 3.3.3).
- Satisfaction challenge (section 3.4):
  - Determination of the normalized expression of the objectives (respectively constraints) to obtain a bonus (respectively penalties). A bonus represents the designer's satisfactions towards a particular objective. A penalty represents the designer's un-satisfaction towards an un-respected constraint (sub-sections 3.4.1 and 3.4.2).
  - Aggregation of bonuses and penalties to produce an overall normalized satisfaction. This satisfaction represents the designer's global satisfaction towards a solution (sub-section 3.4.3).

3.2.2/ NE approach requirements

The NE approach uses normalization and aggregation methods [21] to face the satisfaction challenge. These methods should be set to reflect the designer's satisfaction towards a solution. To this end, optimizer should have some specific pieces of information, in addition of the usual ones. Collecting these information, listed in table 3.1, requires an advance formulation method. This method, developed in chapter 2, is labeled as advanced as it considers the designer's satisfaction.

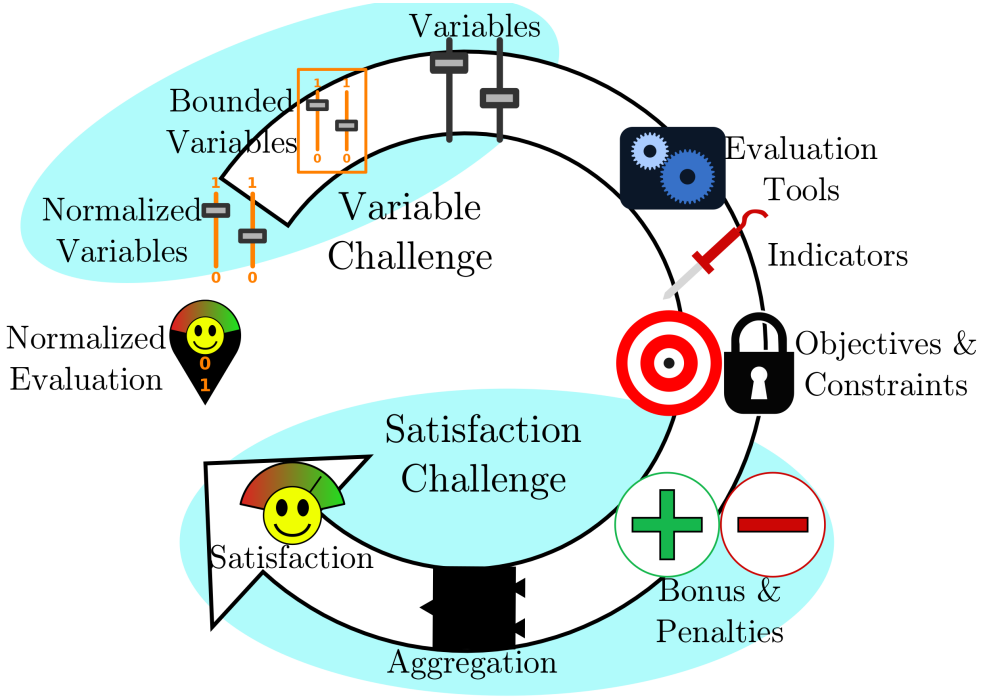


Figure 3.1: NE approach workflow

Table 3.1: Additional formulation information required to face the satisfaction challenge

Name	Notation	Usage	Description
Constraint limit	g^l	Penalty	Constraint value from which the penalty is not computed anymore but set to 1
Objective range	f^r	Bonus	Objective-function's values range.
Satisfying objective value	f^s	Bonus	Objective value for which the bonus is satisfying.
Satisfying bonus value	b^s	Bonus	Bonus value associated to the satisfying objective value.
Un-satisfying objective value	f^u	Bonus	Objective value for which the bonus is unsatisfying.
Un-satisfying bonus value	b^u	Bonus	Bonus value associated to the unsatisfying objective value.
Bonus-function	$b(f)$	Bonus	Desirability function [21] used to compute bonus from objective.
Objectives weights	w	Satisfaction	Weight given to each bonus for the satisfaction computation.

3.2.3/ Description of the new problem

With NE method, the evaluation consists in computing a normalized satisfaction from normalized variables. Therefore, the formulation of the problem changed from the classical one given in equation 1.1. By using the NE method the problem is formulated as in equation 3.1. In this equation s is the satisfaction and z is the vector of normalized variables. How s is computed from z will be explained in sections 3.3 and 3.4.

$$\begin{cases} \quad min \ -s(z) \\ 0 \leq \mathbf{z}_i \leq 1 \quad with \quad i = 1 \ldots D \\ \quad s \in [-1;1] \quad if \quad X \in \mathcal{S} \end{cases} \tag{3.1}$$

By using the NE method, the optimization engine [9] and the algorithm should be tuned to consider that:

- all variables are continuous ones bounded between 0 and 1.
- the satisfaction value is between −1 and 1 for all solutions inside the search space.

3.3/ Variables challenge

This section presents how the variable challenge is faced by NE approach. What are the normalized variables used by the NE approach will be explained in sub-section 3.3.1. These normalized variables should be bounded so that the algorithm explore solution into the search space. how variables are bounded is discussed in sub-section 3.3.2. Once the variables bounded they are transformed into the initial one, which is presented in sub-section 3.3.3.

3.3.1/ Normalized variables

As explained in sub-section 3.2.3, with NE method, the algorithm considers that variables are continuous between 0 and 1. However, to match with the initial formulation of the problem variable and to be used by evaluation tools, they should be converted into the initial variables. In addition, as normalized variables are bounded, bound handling techniques (BHT) should be used for the algorithm not to explore outside of the search space.

3.3.2/ Bound Handling Techniques used by the NE approach

During an optimzation run, it might happen that the meta-heuristic generates individuals where one or more components fall outside the corresponding bounds, espacially with PSO [106]. This could happen, for instance if an algorithm focuses on exploration over exploitation [59]. Although the meta-heuristic was initially proposed for unconstrained problems, several subsequent publications have been addressing adaptations in the original algorithms in order to tackle problems with constraints. In particular, BHT have been developed to face the variables bounds which are a particular kind of constraints. Indeed, they define the search space, by limiting the possible values for continuous constraints. In addition, the importance of BHT and its considerable impact on the final results is not negligible [106]. This sub-section, will introduce BHT which are used with the NE method. For the algorithm to keep looking for solutions inside the variables bounds, BHT must be used [26]. The solution inside the search space will be referred as inside-solutions while solutions outside the search space will be referred as outside-solutions. The BHT can be classified into four groups [26]:

- **Penalty [107]:** The outside-solutions are given satisfaction values so that the algorithm is encouraged to navigate inside the search space.
- **Repair [107]:** The outside-solutions are repaired so that they stay inside the search space.
- **Prevention [26]:** The generation of outside-solutions is prevented.
- **Other [26]:** These methods could not be classified into one of the previous groups. For instance, as mentioned by [106], bound violation could be formulated as an additional objective.

The prevention and other BHT are related to the algorithm mechanism and thus depend on the algorithm itself. The NE approach is designed to be used with any algorithm and thus BHT used by this approach should be independent of the algorithm. Therefore, only penalty and repair BHT could be used by the NE approach. Some of the BHT that could be used with NE approach, inspired by [26, 107], are presented in section A.4. This list is not exhaustive and could be increased, for instance by evolutionary BHT or probabilistic evolutionary BHT techniques [106].

The repair techniques might require to tune the algorithm to be properly used. For instance, in the PSO algorithm case, the particle velocities should be modified, as discussed in [26]. On the opposite, penalty techniques do not require to tune the algorithm. Therefore, the default BHT used by NE approach is a penalty technique. In section A.4, two penalty techniques are presented: the death penalty and the smooth penalty. A death penalty might lure some mechanisms of algorithms, such as gradient-based approximation of the slope, by introducing discontinuity and flat landscape. It should be noted that the death penalty, which is a widely used BHT, is not the most efficient one [26]. Therefore, the smooth penalty, given in equation 3.2, has been chosen as default technique for the proposed method. In equation 3.2, z is the normalized variable vector, s, the satisfaction and α and β are intermediate values.

$$\begin{cases} s(z) = -\frac{1}{D} \sum_{i=1}^{D} \beta_i^2 & with \quad \beta_i = \begin{cases} = 1 & if \quad \alpha_i \leq 1 \\ = \alpha_i & otherwise \end{cases} \\ \alpha_i = 0.5 + |z_i - 0.5| \quad \forall i = 1, \cdots, D \end{cases} \tag{3.2}$$

As mentioned in sub-section 3.2.1 the normalized variables should be converted into bounded ones. With the default BHT, the smooth penalty, as a penalty technique is used, no transformation is made. Hence, in this case, the bounded variables vector (Y) is equal to the normalized variables one (Z), as in equation 3.3. With other BHT presented in section A.4, the relation between bounded and normalized variables is not necessarily trivial as it is the case with the default BHT.

$$Y = Z \tag{3.3}$$

3.3.3/ Transformation of the bounded variables

Once normalized variables (Z) are converted in bounded ones (Y), bounded variables should be transformed into the initial ones (X). Continuous and discrete variables will be transformed by using different formulas.

Equation 3.4 presents how a single bounded variable (y) is transformed into a continuous variable (x). In this equation, x^{min} (respectively x^{max}) is the variable lower (respectively upper) bound. This method, which is a linear regression, has been chosen as it is a simple method establishing a linear relation between the normalized variable and the initial one.

$$x = x^{max} + y \cdot (x^{max} - x^{min}) \tag{3.4}$$

Equation 3.5 presents how a single bounded variable (y) is transformed into a discrete variable (x). In this equation, $\mathfrak{s}$ is the discrete variable scale, which holds $\mathfrak{l}$ items. This formulation has been chosen so that, every item of the scale possesses the same fraction of the normalized search space. Thus, if a solution is randomly selected in the normalized search space, according to an uniform distribution law, every item of the scale has the same probability.

$$\begin{aligned} x &= x^{\alpha} \quad with \quad x \in \mathfrak{s} = [x^1, x^{\mathfrak{l}}] \\ \alpha &= \lfloor y \cdot \mathfrak{l} \rfloor \end{aligned} \tag{3.5}$$

3.4/ Satisfaction challenge

This section intends to explain how satisfaction is computed in the NE approach. First all penalties are computed from constraints and all bonuses are computed from objectives. How penalties are computed will be introduced in sub-section 3.4.1 while how bonuses are computed will be presented in sub-section 3.4.2. How satisfaction is computed from penalties and bonuses will be explained in sub-section 3.4.3.

3.4.1/ Penalties computation

To properly compute satisfaction, to each constraint (g) a penalty (p) is associated. A penalty is assessing how un-satisfied the designer is by the corresponding un-satisfied constraint. Thus, the farther the un-satisfied constraint is from 0, the higher the penalty will be. A quadratic law for penalty variation will have a first order derivative proportional to the constraint un-satisfaction. This could help derivative based algorithm identifying the feasible search space. At some point, the constraint un-satisfaction is so that designer is totally un-satisfied. Therefore, when the constraint limit, defined in table 3.1 and noted g^l, is reached, the penalties will remain constant. Normalized penalties would be more convenient to use for normalized satisfaction computation, so the the bounds of a penalty should be 0 and 1. Once all these remarks are taken into account, the variation of the penalties should be as presented in figure 3.2.

How penalties are computed for 'inferior to' constraints is given in equation 3.6.

$$\begin{aligned} &if \quad (g < 0) \quad && (p = 0) \\ &if \quad (g > g^l) \quad && (p = 1) \\ &else && (p = \left(\frac{g}{g^l}\right)^2) \end{aligned} \tag{3.6}$$

How penalties are computed from equal constraints is given in equation 3.7. In this

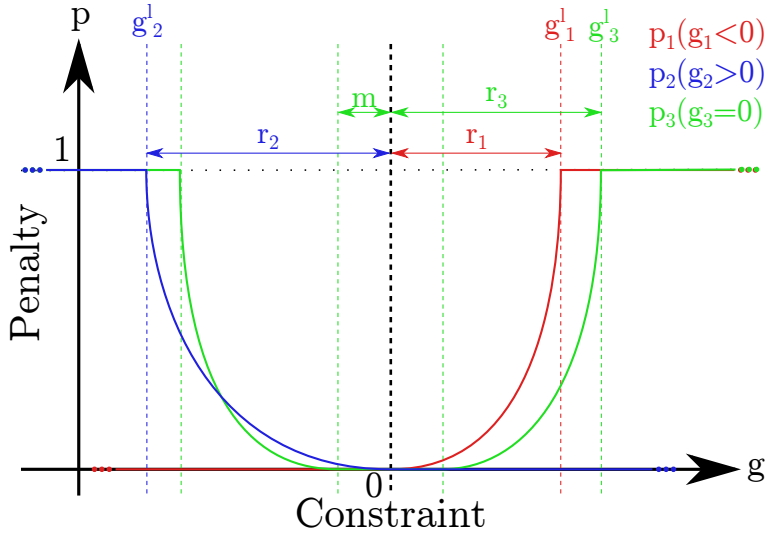


Figure 3.2: Penalty in function of constraint for different penalty types: Lower, Higher, Equal

equation, h^m is the allowed error margin to zero and g^l is the constraint limit from which penalty remains equal to 1. For equality constraints, both h^m and g^l are positive value.

$$\begin{array}{lll} if & |h| \leq h^m & p = 0 \\ if & |h| > g^l & p = 1 \\ otherwise & & p = \left(\frac{h}{g^l}\right)^2 \end{array} \tag{3.7}$$

This sections presented how the penalties (P) are computed. For each penalty, if the associated constraint is respected, the penalty is equal to zero. Thus, if all constraints are respected the vector P is composed only of zero. The penalties will be used to compute the overall penalty as explained in sub-section 3.4.3.

3.4.2/ Bonus computation

To properly compute satisfaction, bonuses are assessing how satisfied the designer is considering an objective value. For an objective to be minimized (respectively maximized), the lower (respectively higher) the objective value, the higher the satisfaction. To every possible value of the objective, a satisfaction, from 0 to 1 is associated. To perform this task, a bonus-function, which is a satisfaction function [21], is used.

Several satisfaction functions could be used as a bonus-function. These satisfaction functions, Harrington [108], Derringer [109] and Wood [110], are presented in figure 3.3. To adjust the satisfaction function to the designer's desire, two correspondence points, noted c_s and c_u, are required. The satisfying correspondence point, c_s, associates the satisfying bonus value, b^s, to the satisfying objective value f^s. The un-satisfying correspondence point, c_u, associates the un-satisfying bonus value, b^u, to the un-satisfying objective value f^u.

These three equations of satisfaction-function are presented here:

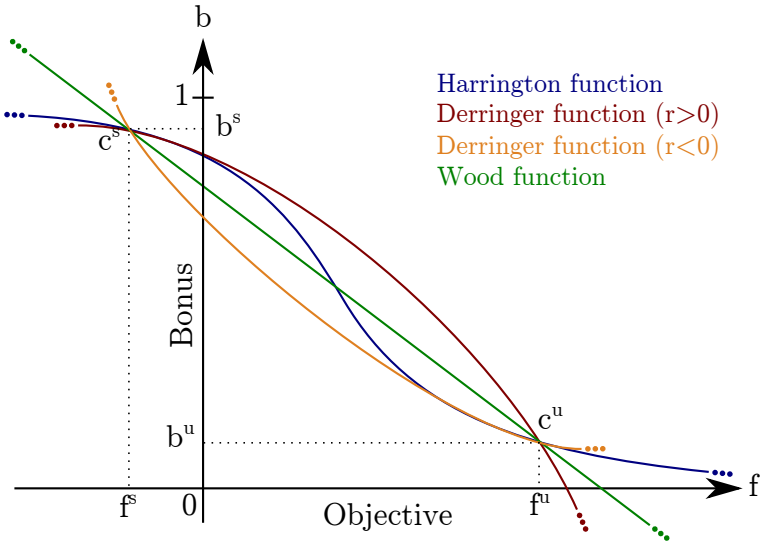


Figure 3.3: Satisfaction functions in an objective minimization case with a linear scale

- Wood function has a linear variation and can be given by:

$$\begin{cases} b = \alpha \cdot x + \beta \\ \alpha = \dfrac{b^u - b^s}{f^u - f^s} \\ \beta = b^s - \alpha \cdot f^s \end{cases} \tag{3.8}$$

- On the opposite, Derringer function, in has a quadratic variation, that could be adjusted by tuning the parameter γ.

$$\begin{cases} Min : f \begin{cases} b = (b^s - b^u) \cdot \alpha + b^u \\ \alpha = \left(\dfrac{f - f^s}{f^u - f^s} \right)^{\gamma} \end{cases} \\ Max : f \begin{cases} b = (b^u - b^s) \cdot \alpha + b^s \\ \alpha = \left(\dfrac{f - f^s}{f^u - f^s} \right)^{\gamma} \end{cases} \end{cases} \tag{3.9}$$

- Finally, Harrington function, in equation 3.10, has a variation stiffening far from the

correspondence points.

$$\begin{cases} Min : f \begin{cases} b = exp(-exp(\alpha + \beta \cdot f)) \\ \beta = \dfrac{ln(ln(b^u)/ln(b^s))}{f^u - f^s} \\ \alpha = ln(-ln(b^s)) - (\beta \cdot f^s) \end{cases} \\ Max : f \begin{cases} b = exp(-exp(\alpha + \beta \cdot f)) \\ \beta = \dfrac{ln(ln(b^s)/ln(b^u))}{f^s - f^u} \\ \alpha = ln(-ln(b^s)) - (\beta \cdot f^s) \end{cases} \end{cases} \tag{3.10}$$

In minimization (respectively maximization) cases, when the objective value is already low (respectively high), the designer is already satisfied and will focus on other criterion. Thus, when the objective value is 'satisfying' enough, a small objective variation will not change the designer's satisfaction much. In minimization (respectively maximization) cases, when the objective value is already high (respectively low), the designer will be unsatisfied no matter how high (respectively low) the objective is. Thus, when objective value is too 'un-satisfying', a small objective variation will not change the designer's satisfaction much. For those reasons, the Harrington function is chosen as the default bonus-function.

It could happen that the range of the values of the objective-function covers many orders of magnitude. In this case, the objective should be evaluated according to a logarithmic scale. To do so, the objective values of the correspondence points, f^s and f^u, should be substituted by their logarithmic scale equivalents. This substitution, of the initial value f by its logarithmic equivalent f', is done thanks to equation 3.11. In this equation f_{min} (respectively f_{max}) is the supposed minimum (respectively maximum) value that the objective can reach. As for f_{em} it is the strictly positive value under which the minimum is consider reached. In addition, the logarithmic function used in this equation, noted Log, is the logarithm to base 10.

$$f' = \frac{Log((f - f_{min})/f_{em})}{Log((f_{max} - f_{min})/f_{em})} \tag{3.11}$$

3.4.3/ Satisfaction computation

Once the bonuses and penalties are computed, the satisfaction could be computed too. As mentioned in this chapter's introduction, the idea is to use normalized constraints and objective, like in OIA method, into the Lagrange multiplier framework. However, the Lagrange multiplier framework has been modified so that a solution satisfying constraints will always be preferred to one not satisfying them.

Indeed, the satisfaction will not be computed the same way if the constraints are satisfied or not. If all constraint are satisfied the satisfaction will be equal to the overall bonus whereas if at least one constraint is un-satisfied, the satisfaction will be equal to the opposite of the overall penalty. This is summarized by equation 3.12. Both the

overall penalty and the overall bonus are positive normalized values. Thus, a solution not satisfying constraints will always have a positive satisfaction, between 0 and 1, whereas a solution not satisfying them will always have a negative satisfaction, between −1 and 0.

$$\begin{array}{ll} if & (\mathcal{P} > 0) \quad (s = -\mathcal{P}) \\ otherwise & (s = \mathcal{B}) \end{array} \tag{3.12}$$

If at least one constraint is un-satisfied, the satisfaction will be the opposite of the overall penalty, denoted $\mathcal{P}$. The overall penalty, given in equation 3.13, is the average of the N_j penalties. If all constraints are satisfied, the satisfaction will be the overall bonus, denoted $\mathcal{B}$. The overall bonus, given in equation 3.14, is the aggregation of the N_b bonuses. The aggregation of bonuses could be done by different aggregation methods [21]. However, the weighted sum has been chosen as the default method. With a weighted sum it is possible to take every bonus into account independently and with different weight. The different bonuses are weighted according to the weights of the objectives, w, which are defined during the formulation phase.

$$\mathcal{P} = \frac{1}{N_j} \cdot \sum_{j=1}^{N_j} p_j \tag{3.13}$$

$$\mathcal{B} = \sum_{j=1}^{N_b} w_j \cdot b_j \qquad with \quad \sum_{j=1}^{N_b} w_j = 1 \tag{3.14}$$

3.5/ Results and discussion

In order to test the benefit of using normalized evaluations, the normalized evaluations have been compared with the classical ones. Classical evaluations refer to evaluations done without using the NE approach. However, in the studied case, the classical evaluations will be performed using the water-fall objectives function method [37]. This method, which is detailed in sub-section 7.3.2, consists in using a particular sub-objective-function depending on which of the constraints, preliminary ordered, are fulfilled or not. How methods are compared is presented in sub-section 3.5.1. The results are discussed in sub-section 3.5.2.

3.5.1/ Evaluation methods review

To compare the classical and NE approach, 5 algorithms using both methods have been used to solve an industrial problem.

The algorithms used to test the evaluation method are the global stochastic population-based meta-heuristic algorithms chosen in sub-section 1.4.2 to test methods developped in this thesis. They represent different well-known widely used families of algorithms. The chosen algorithms are particle swarm optimization (PSO) [54], covariance matrix adaptation evolution strategy (CMAES) [52], genetic algorithm (GA) [55], Cuttlefish [56] and simulated annealing (SA) [57]. Detailed information about these algorithms and their settings are given in sub-section A.7.

To test the method, the algorithms have been used on an industrial optimization problem. It consists in designing a piezoelectric device, a surface acoustic wave (SAW) filter. This optimization problem is composed of 1 objective, 18 variables and 5 inequality constraints. Computations are done by a numeric model developed by the SMART-INN partners (see chapter 7).

3.5.2/ Optimization runs results

The 5 algorithms have been evaluated with classical and normalized evaluation, according to three performance measures [46]:

- Value quality: measures the quality of the objective-function at the end of an optimization case. As a comparison point, for the case, an increase of value quality by 0.1 point is equivalent to respecting an additional constraint.
- Convergence quality: measures how fast an algorithm converges to a solution. As a comparison point, an increase of convergence quality by 0.1 point is equivalent to reducing the running time by 10%. For the test case, it corresponds to a reduction of the running time of almost one hour.
- Efficiency: is a combined measure of value and convergence qualities. An increase of the efficiency by 0.1 point, is equivalent to an increase of both value and convergence by 0.1. In the particular case, it would mean than one more constraint would have been respected while the running time would have been shorten by almost an hour.

The results are presented in figure 3.4. Value quality of PSO, CMAES and SA has been improved by more than 0.1 point, while Cuttlefish one remains the same and GA one was reduced of 0.2 point. Thus, value quality remains the same or significantly increases for 60% of the tested algorithms. Convergence quality remains at 0 for Cuttlefish and SA. As they have not been provided a convergence criterion, their convergence could be non zero only if they reach the optimum. Once set apart, it can be seen that PSO and CMAES convergence increases by at least 0.1 point while GA goes from 0 to 0.8. Therefore, it can be stated that for the tested algorithms with a convergence ability, convergence has been increased at least significantly if not drastically. Finally, Cuttlefish efficiency remains the same and SA one has been increased by less than 0.1 point. On the opposite PSO, CMAES and GA efficiencies increased by more than 0.1 point. Thereby, efficiency has been significantly improved for 60% of the tested algorithms.

From the previous statement, a few remarks could be made about the proposed evaluation method. First, it globally improves an algorithm robustness. Second, it improves the algorithms convergence ability. Third, when used, the efficiency is either as good or better. Finally, apart from GA loss in term of value quality, for every algorithm, for every performance measure, the score is at least equal if not better.

3.6/ Conclusion

In order to reduce the EDO timespan, an evaluation method taking the designer's desires into account was sought. A normalized evaluation approach has been proposed. It uses normalized variables and a novel satisfaction computation method. With normalized

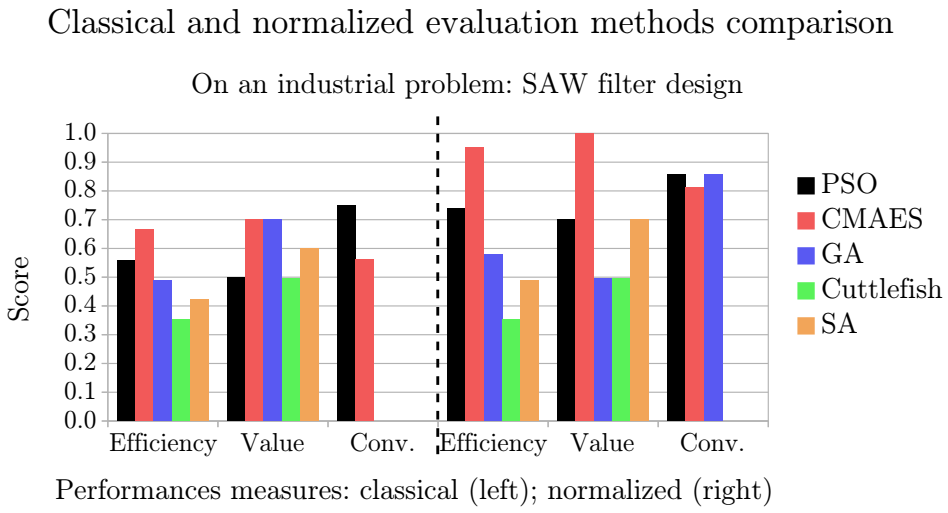


Figure 3.4: Evaluation methods review on an industrial case

evaluations, normalized variables are bounded and then transformed into the initial ones. Also, the objectives and constraints are normalized to obtain bonuses and penalties that will be aggregated to produce a satisfaction.

The NE approach has been compared to the classical one. The comparison has been done by using five meta-heuristic algorithms to solve an industrial problem using both methods. On this test, the NE approach have higher performances in terms of value quality, convergence quality and efficiency.

As the value quality has been improved it is likely that the NE approach will obtain more satisfying results. Thus, the resolution phase should be performed a fewer number of times to get a satisfying result. Moreover, the convergence has been improved, meaning that the resolution phase will be shortened. In addition, as efficiency also increases, it means that the combination of both aspect, value and convergence, is improved. Therefore, value improvement and convergence improvement are not in opposition. From the previous remarks, it could be deduced that the NE approach should fulfill its objective of reducing the timespan of the resolution phase.

However, some points should be explored. First, GA value quality has been degraded with the normalized evaluations method. Therefore, this method possesses limits to be explored. Second, the combination of the default sub-methods has been used to test the NE approach. The influence of sub-methods combination may be explored. In order to further investigate this method, more complete sets of tests must be conducted. These tests would use algorithms from a larger number of families and being of different degrees of evolution. They also must be conducted using different settings for the NE approach, for instance by changing the BHT.

4

A Benchmark for Meta-heuristic, Single-Objective, Continuous Optimization Algorithm

Contents

4.1/ Introduction

From a theoretical point of view, it is accepted that no metaheuristic can be considered as being better than another: this is the 'no free lunch' theorem [111]. However in practice, important performance differences can be observed depending on the algorithm mechanisms' quality and the problem's structure.

When a new algorithm is developed, most of the time, it is compared with few other ones on means and standard deviations of a few test functions final results [13]. The other algorithms may belong to the same family. Furthermore, the retained test functions are often part of well-known function test suites, such as the one proposed by De Jong in [55], being one of the oldest. This may lead to wrong assessment of the algorithm efficiency. To avoid false conclusions, algorithms should be tested on benchmarks [13]. Since 2005, some reference benchmarks have been proposed in the literature [13]. Two of them are quite renowned and have been presented for special sessions during international congresses: the CEC [112] in IEEE Congress on Evolutionary Computation (CEC) and the Black-Box Optimization Benchmarking (BBOB)[113] in Genetic and Evolutionary Computation Conference (GECCO). More recently, a platform designed to compare continuous optimizers in a black-box has been developed: COCO (Comparing Continuous Optimizers). More details on this approach can be found in [43, 114]. CEC is composed of functions with higher dimensionality than BBOB but without noise [41]. CEC functions' landscapes are very different from each other. CEC is a fixed-budget scenario: the problem solving 'time' is fixed. Usually, it is expressed in function evaluation. It can also be expressed in CPU time or in clock time [19]. This approach might not be relevant for a real problem. Calls can be high CPU-time consuming and should be reduced to the minimum. BBOB is a fixed-target scenario [113]. This approach raises an important issue: how to deal with an algorithm that does not reach the target at all or in a too-long time? Moreover, a fixed-target scenario can be an inadequate choice when associated with some methaheuristic [45]. Indeed they can modify the number function calls during the search. Recently a new benchmark has been proposed [115].

The proposed benchmark works according to the same pattern as other ones. An algorithm is run on a set of test functions from which data are extracted to provide performance measures. The main performance measure, which is the main score, is a general efficiency measure [46]. It may be exploited by researchers who wish to compare, tune or improve their algorithms. This score emphasizes the ability to fairly evaluate algorithms with criteria matching industry problematics. This is due to the computation technics used. To summarize raw data into a simple indicator, aggregation techniques [21] are used. To do this, data must be comparable and therefore it is necessary to convert them into the same metric. This metric is a level of desirability [116, 117] obtained by using the Harrington desirability function [118]. A level of desirability represents how desirable a solution is. It is computed thanks to a desirability function converting a scalar, which is the objective-function value in the benchmark case, into a level of desirability. The concept of desirability is similar to the one of satisfaction presented in chapter 3. Using a global level of satisfaction based on different metrics has already been done by [27] to find the optimal parameters of evolutionary algorithms. The innovative idea of this work is to introduce this concept to evaluate the main score of an algorithm.

In addition of the main score, other performance measures are defined, the sub-scores. Those are an advantage of this benchmark over the other ones. Indeed, they lead to a better understanding of an algorithm with more precise and exploitable data such

as: speed of convergence or the ability to solve a particular type of problem. Their main objective is to be able to provide all the characteristics of an algorithm.

This chapter aims to define a new benchmarking technology evaluating the performance and robustness of optimization algorithms and applying it to a set of significant selected optimization processes. This benchmark structure will be explained in section 4.2. To process scores, run results should be computed first, which is explained in section 4.3. Then, from run results, case results are computed, which is presented in section 4.4. Finally, from cases results scores are computed, as detailed in section 4.5. The proposed benchmark results are analyzed in section 4.6.

4.2/ Architecture of the proposed benchmark

In this section, the architecture of the proposed benchmark is presented. To understand the benchmark structure, a few general remarks, given in sub-section 4.2.1, should be stated. In addition of these remarks, the benchmark goals and objectives, introduced in sub-section 4.2.2, should be defined. Once done, the benchmark test set, explicited in sub-section 4.2.3, is designed. The following step is to define the set of scores, which is done in sub-section 4.2.4. In order to design a proper benchmark, [46]'s guideline has been followed.

4.2.1/ General remarks

In this sub-section, a few general remarks are made to help understanding:

- Upper case letters refer to fix elements while lower case letters refer to instantiated elements.
- An optimization case is an optimization problem associated with an algorithm, it can be run several times. At the end of a run, after a gross number of evaluations, a gross value of the objective function is achieved. Those two quantities may be normalized in order to compute a run result and two run sub-results.
- To obtain the global score of the proposed benchmark, the run results are aggregated [21] step by step. Different aggregation methods are used for the different steps. They need elements such as a vector of values, weights and indexes. Intermediate results are computed during the scores computation. The term 'value' will refer to the end of optimization objective-function value. The term 'score' will refer to end of benchmark performance measure. The term 'result' will refer to an intermediate result required to compute scores.
- The maximal number of evaluation allowed to solve a case, FEs, is computed as in equation 4.1. To compute FEs, the problem dimension, D, and $MaxFEs$ a coefficient are used [119].

$$FEs = 10000 \times D \times MaxFEs \tag{4.1}$$

4.2.2/ Goals and objectives

As this benchmark project is developed in an industrial context, it has to be able to evaluate algorithms according to problems faced by medium-sized entreprises in order to obtain the most efficient algorithm. An efficient algorithm must find a good objective function value in a short time. A suitable compromise between those two wills has to be found [8]. The main score should be similar to a return on investment measure. Also, the algorithm task is to find a good global optimum more than refining a local solution.

To this end, optimization cases were selected to try to constitute a representative sample of applied cases potentially found in a real industrial settings. In this context, the provided tool must be convenient to use and must provide value calculations which are necessary for the evaluation of the global score.

Two main different objectives can be distinguished. The first one is to provide an efficiency measure in order to compare and rank algorithms. The second one is to provide a set of measures helping designers to select an algorithm. To rank algorithms, this benchmark should be a generic tool that allows standard comparisons. In order to do so, the way optimization cases information are used to compute scores must be settled.

Scores computation relies on methods weighted with regards of this benchmark context. The weights' choice, while motivated by this benchmark's useage context, could be discussed, improved and tuned for a specific purpose. Though, one should remind that benchmarking always relies on subjective choices [46] which implies that its results only makes sense in it's context and with consideration of its bias.

4.2.3/ Test set

Before beginning an experiment, a benchmark dataset must be chosen. The main idea is to run an algorithm with a fixed set of parameters on a collection of problems in order to measure its performance. Specifically, an optimization case is defined by three quantities (figure 4.1):

1. an objective function F.
2. a dimension D.
3. a maximum number of fitness function evaluations' coefficient $MaxFEs$.

4.2.3.1/ Cases generation

Our benchmark's functions and dimensions are based on CEC 2015 competition [120].

- Benchmark functions are used by the algorithm as black boxes and are explicitly known by the scientific community. In order to test the algorithm efficiency in a large context, various categories of functions are considered to represent a wide range of difficulties: 15 functions were selected according to their characteristics. The details and the expressions of this set of functions can be found in section A.5.
- Four different search space dimensionalities are used for all functions.
- Seven different values of $MaxFEs$ are defined in the benchmark.

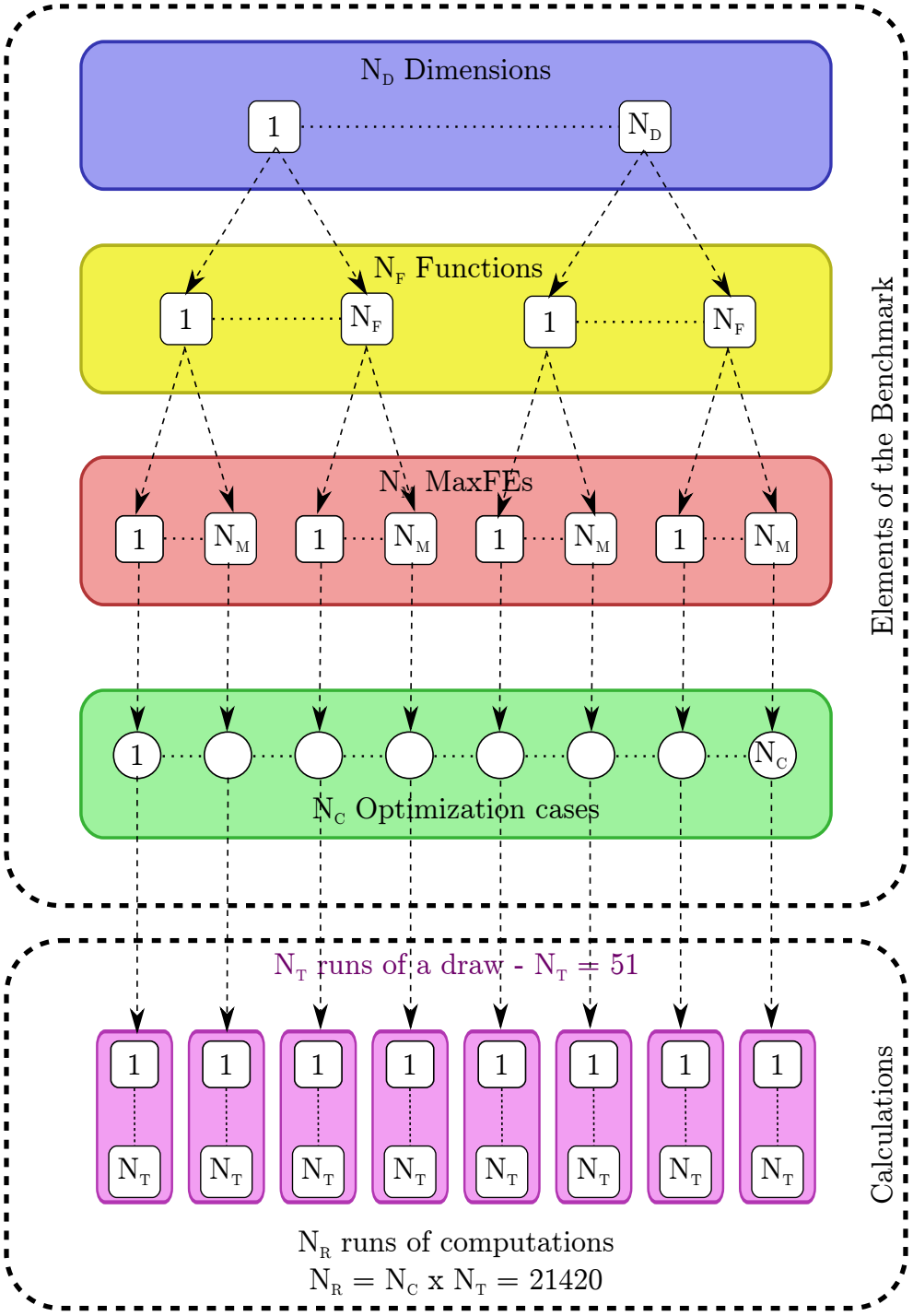


Figure 4.1: Global architecture of the benchmark - optimization cases and runs generation

4.2.3.2/ Details of the elements

Finally, this benchmark is composed of 420 optimization cases. All the elements are listed in table 4.1. Due to the usual stochastic nature of the tested algorithms, several independent runs are needed. So, it is obvious that the significance of the results should be tested with an appropriate statistical measure. In this way, several evaluations are conducted on the same optimization case. It has been proved that $N_T = 51$ evaluations

are enough to make relevant performance differences with a statistical significance [121]. This set of runs of the same optimization case is called a draw.

Table 4.1: Benchmark generation elements detail

Element	Number	Value
Dimension	$N_D = 4$	$D \in \{10, 20, 30, 50\}$
Function	$N_F = 15$	$N_{uni} = 2$ $N_{multi} = 7$ $N_{hybrid} = 3$ $N_{comp} = 3$
MaxFEs	$N_M = 7$	*MaxFEs* $\in$ $\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1\}$
Optimization cases	$N_C = 420$	$N_C = N_D \times N_M \times N_F$

In building design industry, most of the optimization problems have between 8 and 24 variables, with an average of 15 [9]. In other fields the average number of variables may be higher. Still, considering our context, it seems wise to choose dimensions inferior to 50. The choice of functions is of great importance in the design of a benchmark [46]. CEC's and BBOB's functions set have issues [41]. Another set of functions could be used. But this topic is complex enough to justify another research work and this point should be discussed in another article. As advised by [46], a standard test set will be chosen, even if it is not perfect. The CEC 2015's test functions suite is chosen.

4.2.4/ Presentation of the different scores

As specified previously, this benchmark has two main objectives. On one hand, it is interesting to standardize comparisons and to be able to define a generic tool. To this end, the main score is introduced to provide the user the most representative overview of the performances of an algorithm. The influences of all dimensions, functions and *MaxFEs* are taken into account when evaluating this quantity.

On the other hand, it is fundamental to understand more precisely with which problems an algorithm will be efficient. To this end, sub-scores are introduced. Information sub-scores provide information on the general behavior of an algorithm. Sub-set sub-scores highlight the influence of the dimensions, functions and *MaxFEs* on the algorithm efficiency. sub-scores show if an algorithm works correctly on a specific problem or under specific conditions, like being able to solve a problem multiple times. They highlight strengths and weaknesses of an algorithm. They may be used to modify or tune an algorithm in order to increase its performance. This could be done either for a particular kind of problem or in a general way. Moreover, by knowing a set of algorithm's sub-scores, it is possible to choose the algorithm that best suits the problem to be solved.

4.2.4.1/ Main score

In an industrial context, an algebraic description of the objective function is most of the time impossible. In general, the objective function is a black box based on a simulation. In addition, many large-scale and/or detailed simulations may be too expensive to run, time-wise. As a result, the efforts required to solve an optimization problem strongly depends

on the number of evaluations of the objective function. This depends on the optimization algorithm used and should be reduced as much as possible. In this context, it appears fundamental to take into account the number of evaluations in the computation of the run results (section 4.3). Moreover, if the computation time of the objective function is high, the associated solving time will also be important thus the optimization case will probably be run a limited number of times. On the opposite, if objective function computation time is low, the case could be run many times. Yet, for a stochastic optimization algorithm, the higher the number of runs, the greater the chance to find the best result. So, the main score must take into account both configurations.

4.2.4.2/ SUB-SCORES

This benchmark uses two kinds of sub-scores: the information ones and the sub-set ones.

A sub-set sub-score is computed by using only the corresponding sub-set of optimization cases. Sub-set sub-scores are divided into three categories: dimensions, functions and *MaxFEs*. These sub-sets sub-scores allow to measure the efficiency of an algorithm on a set of functions sharing a fixed particular property. This property is one of the three elements that characterizes an optimization case: dimensions, functions and *MaxFEs*.

Information sub-scores use the sub-results of the cases. These scores take into account the stochastic nature of the tested algorithm. They underline the algorithm's efficiency, how fast it converges and how good the results are considering the number of evaluations. To this end, they are divided into two categories: quality sub-scores and reliability sub-scores. Two measures of quality are introduced: a value score and a convergence score. The value score measures the quality of the objective function at the end of an optimization case. The convergence score measures how fast an algorithm converges to a solution. Two reliability measures are introduced: the alpha score and the omega score. The alpha score is the best result achieved in a draw. It represents the best possible outcome of an algorithm. On the opposite, the omega score is the score achieved by a large majority of the runs of a draw. It represents what, at least, can be expected by the algorithm. For both alpha and omega scores of a run are efficiency scores, meaning they take into account end value and convergence speed.

All scores information are summarized in table 4.2.

4.2.4.3/ SCORES COMPUTATION

This benchmark works on solving multiple times N_C different optimization cases, giving N_C case results and N_C set of sub-results. To extract performance measures from the generated data and to make them easily usable for the user, they need to be summarized. To this end, some sets of scores are defined based on the aggregation over a collection of values. This is presented in figure 4.2. In order for such a method to work correctly, it is fundamental to have comparable data, hence the necessity to normalize these data and to define metrics. These aspects are described in details in the next sections.

Table 4.2: Benchmark scores summary

Kind	Category	Name	Performance measure	Set	Details
Global	Main score	Score	Efficiency	All	How efficient an algorithm is
Information	Quality	Value	Quality	All	How good end of optimization value is
		Convergence	Quality	All	How fast an algorithm converges
	Reliability	Alpha	Reliability	All	Best score achieved in many runs
		Omega	Reliability	All	Score achieved by a large majority of runs. Computed thank to a 95% confidence limit
Sub-set	Dimension	D_{10} *to* D_{50}	Efficiency	Cases where $D = d$	Algorithm efficiency on a D-dimensions case
	Function	F_1 *to* F_{15}	Efficiency	Cases where $F = f$	Algorithm efficiency on a f-th function case.
	MaxFEs	$M_{0,01}$ *to* M_1	Efficiency	Cases where $M = m$	Algorithm efficiency if *MaxFEs* $= m$

4.3/ Results of a run

This section will present the computation of run results. Which strategy will be used to compute the run results depends on the stopping conditions used, which are defined in sub-section 4.3.1. Then, the run result computation process has to be defined as it is the case in sub-section 4.3.2. This process relies on gross value normalization and gross number of evaluations normalization which will be examined in sub-sections 4.3.3 and 4.3.4. Finally, how normalized value and normalized number of evaluations are aggregated into the run result will be explained in sub-section 4.3.5.

4.3.1/ Stopping conditions

Two kind of scenarios are currently used as stopping conditions: the fixed-target and fixed-budget scenario [122]. As explained by [44], both of these approaches, while of merit and justified by practical needs, have limits making them inappropriate for real-world if used individually. For instance, designers are not always able to define a target or a budget that makes sense. Some of these approaches' drawbacks might be solved. For instance, in fixed-target scenario, if the target value can not be reached, a success rate can be calculated as in [48]. Still as not all drawbacks cannot be tackled at one time, an interesting approch is to combine both scenario as suggested by [19]. In this case, target and costs are no longer defined as goals but as limits, which is an easier task to achieve for the designer. This is why, in the studied context, the best method is to use several stopping criteria. Article [9] presents several stopping criteria, from which three have been chosen. This article also mentions the possibility of using several of them alltogether. That is why, a run is stopped

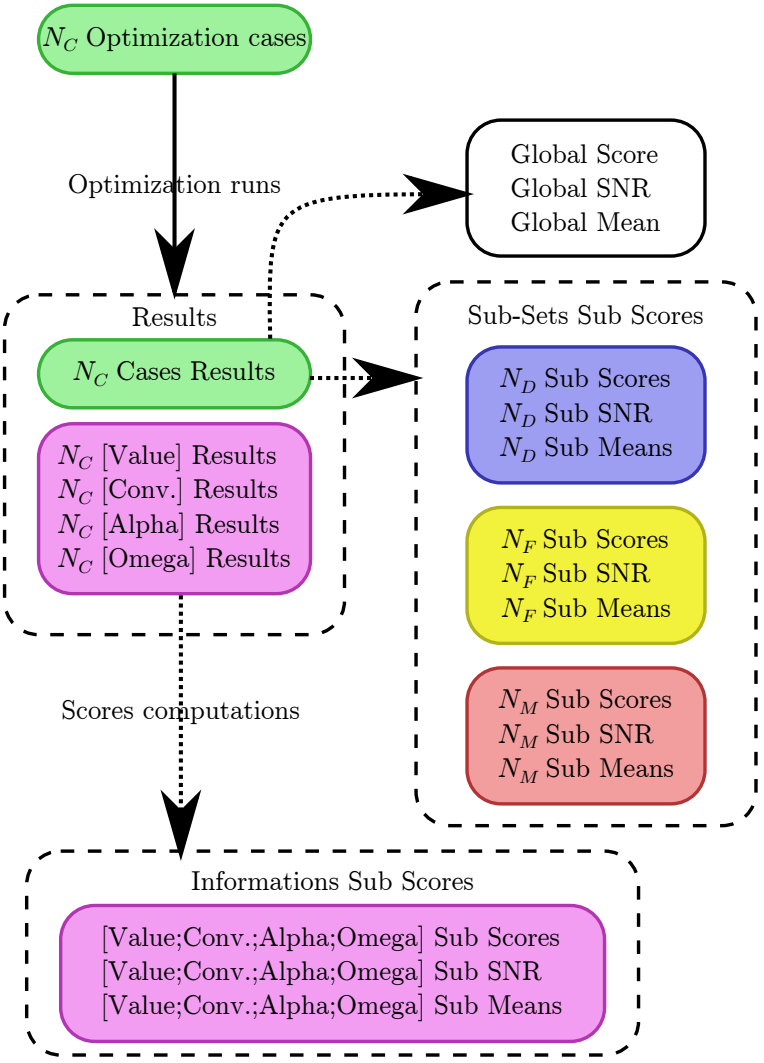


Figure 4.2: Benchmark scores work-flow

if at least one of these three situations occurs:

- The maximum number of evaluations of the objective function (*FEs*) is reached.
- The target (f_{min}) is reached.
- One of the algorithm's potential stopping criterion is reached.

Algorithms' own stopping criteria are described in section A.7.

4.3.2/ Run result constitution

The run result represents the performances of an algorithm on a particular run. Obviously, this result should be based on the value of the objective function at the end of the optimization (V_G). However, it should also consider the number of calls made by the algorithm (E_G). Indeed, the aim of this metric is to mimic a return on investment by measuring the

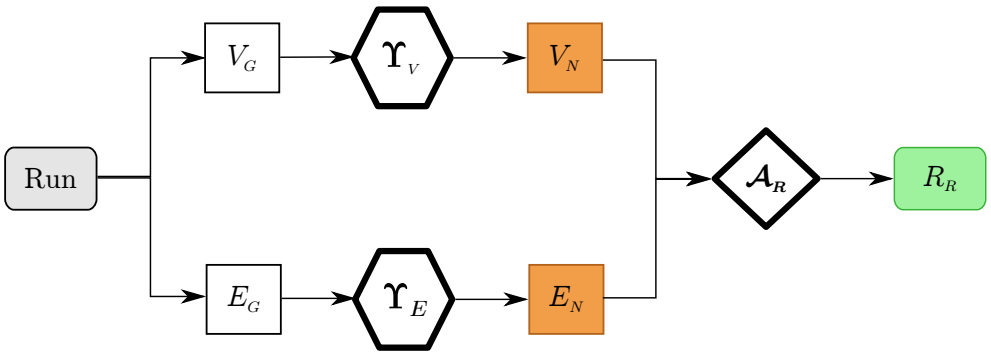


Figure 4.3: Run results computation workflow

quality of the value obtained for the invested solving time. How V_G and E_G are used to make a run result is explained by figure 4.3.

The final result of the run is obtained through the aggregation of the normalized value (V_N) and the normalized number of evaluations (E_N) given in equation 4.2.

$$R_R = \mathcal{A}_R\left(\Upsilon_E(E_G);\ \Upsilon_V(V_G)\right) \tag{4.2}$$

In this expression, several mathematical operators are introduced: aggregation operators and normalization which are explained in the following sub-sections. The normalized value (V_N) and the normalized number of evaluations (E_N) are run sub-results that will be used to compute sub-scores.

4.3.3/ Normalization of the gross operating value

To translate the performance of the algorithm, our main idea is to introduce a level of desirability, as in [27]. When approaching zero it reflects a value very far from the minimum of the function and have a higher chance not to lead to a correct solution. On the opposite, a level of desirability close to one reflects a minimum considered as acceptable. This aspect is accurately modeled by the Harrington function [118], called here Υ_V. This function is drawn on figure 4.4. Harrington function has been selected in the case 'the lower, the better' and a logarithmic scale has been chosen, leading to equation 4.3.

In this equation f_{min} (respectively f_{max}) is the minimum (respectively maximum) value of the objective function. In our case of eminent mathematical functions, f_{min} is well-known: it is the offset o of the function. f_{max} has been numerically estimated for each test function (table 4.3). Finally, the permitted error margin, f_{em}, is the strictly positive value under which the minimum, f_{min}, is considered as reached. As in CEC 2014 [121], f_{em} has been set to 1e−8.

This normalization method uses a logarithmic scale as the functions' order of magnitude are important. Indeed, let's consider a function so that $f_{min} = 0$, $f_{max} = 1e{+}10$ and $f_{em} = 1e{-}8$ and that $f_{max}/10$ is a small value. With a linear scale, the small value will be 1e+9 whereas with a logarithmic scale 1e+0 will be a small value. For this example, it is consistent to say that 1e+0 is a small value but not 1e+9. That is why a logarithmic scale should be used to normalize functions having a wide order of magnitude. Harrington method has been setted using a semantic scale as proposed by [21]. In this approach, two objective function values are labeled in terms of satisfaction, from extremely low to

extremely high. A satisfaction value is associated to every label. The two couples of value/satisfaction are used to set up a normalization method. In this case, the value are considered after being transformed by logarithmic scale (γ). Article [123] expresses that relative precisions of order 10-6, or even smaller, do not always reflect the needs of real-world optimization problems. That is why $\gamma = 0.1$ is associated to 'very high', 0.95. On the opposite, when the result is too high to be exploited, no matter how high it is, the associated satisfaction will remain very low and will not vary much. That is why $\gamma = 0.9$ is associated with 'very low', 0.05.

Tremendous gaps can occur between the maximum values of the different test functions. In this situation, if the values are aggregated without being normalized, the influence of some functions on the evaluation of the score will be nearly ineffective. This bias could lead to important mistakes as demonstrated in sub-section 4.6.2.1.

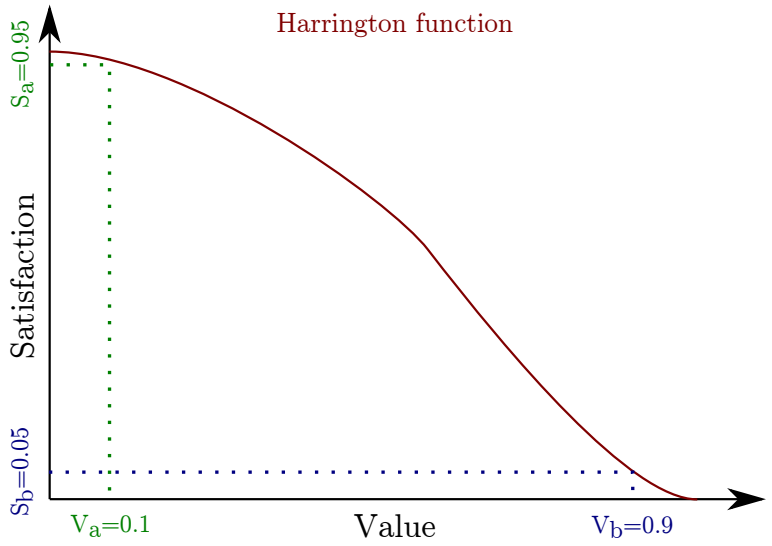


Figure 4.4: Harrington normalization function in 'the lower the better case' with predefined settings

$$\Upsilon_R : \begin{cases} V_N = exp(-exp(\alpha + \beta \cdot \gamma)) \\ \beta = \dfrac{ln(ln(0.95)/ln(0.05))}{0.1 - 0.9} \\ \alpha = ln(-ln(0.95)) - \beta \cdot 0.1 \\ if \quad (V_G - f_{min} > 0) \quad (\gamma = \dfrac{Log_{10}((V_G - f_{min})/f_{em})}{Log_{10}((f_{max} - f_{min})/f_{em})}) \\ else \qquad\qquad\qquad\qquad (\gamma = 0) \end{cases} \tag{4.3}$$

4.3.4/ Number of evaluations normalization

'The lower, the better' variation has been selected as the lower the number of evaluations, the higher the satisfaction. A linear scale has been chosen here as it works adequately where there are not many orders of magnitude between the minimum and the maximum. Besides, using a steady method leads to a steadily satisfaction increase with the reduction of number of evaluations. its normalization is presented in equation 4.4.

Table 4.3: Estimated functions maximums - given without offset

F	D_{10}	D_{20}	D_{30}	D_{50}
f_1	9.00e+10	1.90e+11	2.90e+11	3.90e+11
f_2	1.00e+10	1.00e+10	1.00e+10	1.00e+10
f_3	4.00e+01	7.50e+01	1.00e+02	1.50e+02
f_4	8.05e+02	1.61e+04	2.42e+04	4.03e+04
f_5	1.00e+03	1.00e+03	1.00e+03	1.00e+03
f_6	2.50e+01	2.50e+01	2.50e+01	2.50e+01
f_7	3.00e+02	5.00e+02	7.00e+02	1.00e+03
f_8	2.00e+07	3.50e+07	4.00e+07	5.00e+07
f_9	1.00e+01	2.00e+01	3.00e+01	5.00e+01
f_{10}	1.02e+10	1.20e+10	1.40e+10	2.00e+10
f_{11}	2.05e+10	5.20e+10	8.10e+10	1.41e+11
f_{12}	5.30e+16	1.05e+17	1.55e+17	2.57e+17
f_{13}	3.70e+06	1.05e+07	2.00e+07	3.70e+07
f_{14}	7.00e+01	9.50e+01	1.15e+02	1.45e+02
f_{15}	9.75e+02	7.15e+02	8.55e+02	1.07e+03

$$\Upsilon_E(E_G) = E_N = 1 - \frac{E_G}{FEs} \tag{4.4}$$

4.3.5/ Aggregation of a run result

When an optimization is performed in an industrial context, finding the best result in the shortest time is essential. An aggregation method with specific properties has been used: a non-annihilating version of the weighted product [21] has been chosen. The mathematical expression of this method is given in equation 4.5. In this equation, w_1 and w_2 are used to weigh the quality of the result and the amount of time.

If the benchmark is used for personal application, the user can tune w according to its optimization context. The more the gross value matters the more the quality of value matters and the higher w_1's value should be set. On the opposite, the higher the computer resources are the less convergence speed matters and the lower w_2's value should be set. Furthermore, if only the value quality (respectively convergence speed) is important, the value of w_1 (respectively w_2) should be set to 1. If $w_1 = 1$ (respectively $w_2 = 1$), only the value (respectively convergence) quality is taken into account for the run result computation. In this case, the run result is a measure of value (respectively convergence) quality, instead of efficiency measure. Furthermore, if the run results measures quality then the overall score also measures a quality instead of an efficiency. Sub-section A.6.1 shows the weights' impact on the main score and can help the user to set w's values.

If the benchmark is used to guide the designer in the choice of an algorithm, by comparing them in a competition, w values should be fixed. In this case, w should be tuned in a general context so that main score remains the general mesure of efficiency as it is meant to be. The weights have been chosen according to the following remarks:

- Weights should be balanced; Article [8]: The majority of the companies try to achieve optimized design as a balance between cost, quality and time but the design may not be the best. There are three major areas of improvement for engineering design optimization: efficiency, speed and human knowledge use.

- Value should be weighted more than convergence; Article [115]: The optimizer which found the best solution is considered the best performer unless other optimizers reached a matching best solution. Then the fastest of them is considered the best performer.
- Convergence speed is important; Article [8]: Optimization needs at least 50% of design life cycle.
- Finally, sub-section A.6.1, according to the previous remarks, motivates the choice of w's values.

The impact of these weights' choice on run results is presented in sub-section A.6.2.

$$\mathcal{A}_{\boldsymbol{R}} : \begin{cases} R_R = 2\left(\prod_{i=1}^{2}\left(\frac{1+V_i}{2}\right)^{w_i}\right) - 1 \\ V = \{V_N; E_N\} \\ w = \{0.75; 0.25\} \end{cases} \tag{4.5}$$

4.4/ Optimization case result and sub-results

The N_T run results and sub-results of a draw must be synthesized into a case result (R_C) and sub-results. Generally, results are based on mean computations but this is not specifically the case in our situation. In this benchmark, case result is the mean of alpha and omega sub-results, respectively noted R_A and R_O. The alpha sub-score measures the performance when an optimization case can be run multiple times. The omega sub-score measures the performance when an optimization case can be run a few times. It is considered that both configurations are equally probable. That is why the result case is the mean of those two sub-scores. Two other run sub-results exist, the value and convergence sub-results, respectively noted R_V and R_K. The workflow of the run result and sub-results computation is presented in figure 4.5. This process will be detailed in the next sub-sections.

First, how alpha sub-result is computed will be presented in sub-section 4.4.1 while how omega sub-result is computed will be presented in sub-section 4.4.2. Then, the case result computation will be presented in sub-section 4.4.3. Finally, value and convergence sub-results computation will be presented in sub-section 4.4.4.

4.4.1/ Alpha sub-result computation

This result corresponds to the one that can be achieved if the optimization case can be solved many times. In this configuration, the best result, or 'alpha' result, will be kept. To this end, the best value of all the runs is retained. The best sub-result,R_A, is defined as the maximum value obtained during all the runs (see equation 4.6):

$$\mathcal{A}_{\boldsymbol{A}} \begin{cases} R_A = \max\{\Re_R\} \\ \Re_R = (R_R^1, \cdots, R_R^{N_T}) \end{cases} \tag{4.6}$$

In this expression $\Re_R$ is the set of results of N_T runs of an optimization case.

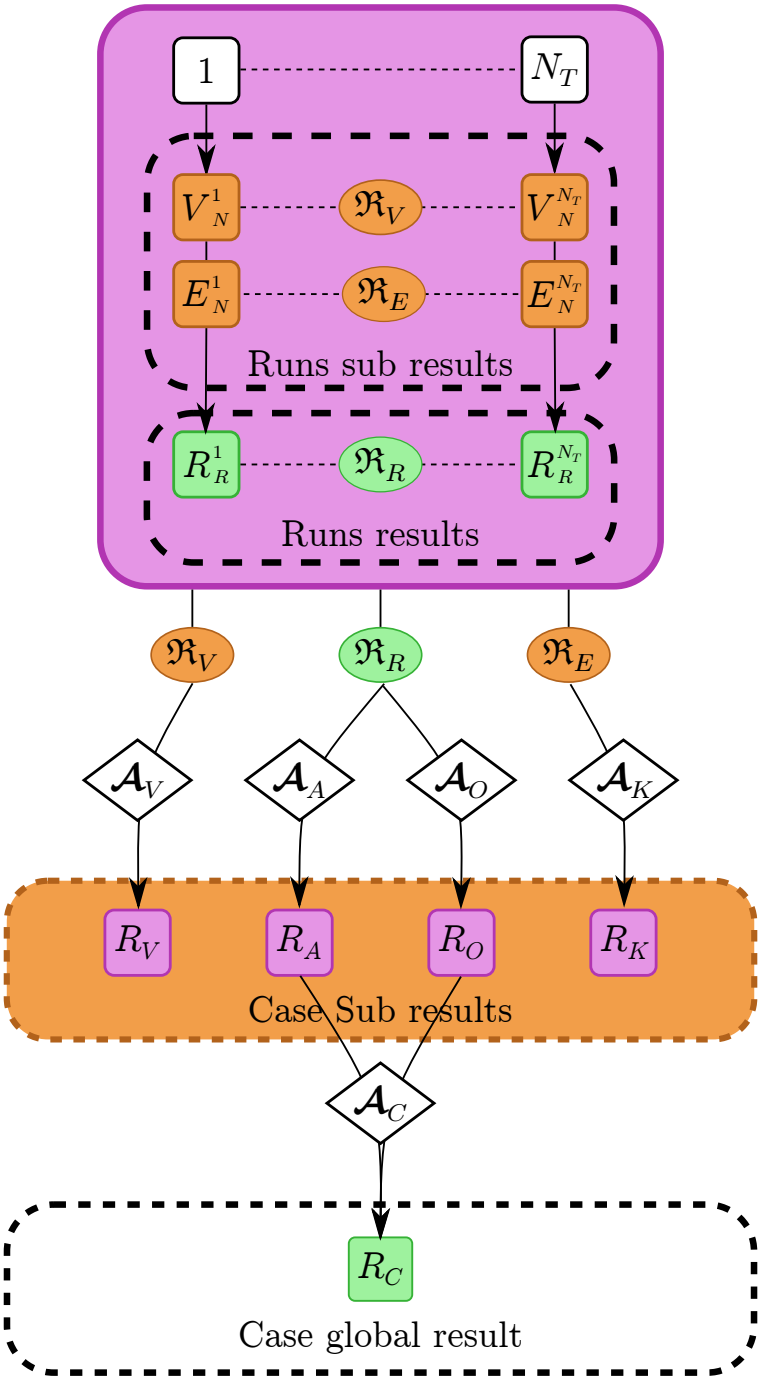


Figure 4.5: Optimization case results workflow

4.4.2/ Omega sub-result computation

In the situation of optimization and co-simulation, a run can be very CPU time-consuming. Last minute change in design specification puts a lot of strain on the human resources [8]. In this situation, it is hardly possible to perform several runs to find a better solution than the previous one: only one run is done. The omega sub-result is the 95% confidence limit of the draw's runs results' distribution. The omega sub-results, R_O, is defined in equation 4.7. Confidence limit ratio has been set to 95% according to sub-section A.6.3's remarks.

$$\mathcal{A}_O \begin{cases} R_O = \mu_{\Re_R} - y\,\sigma_{\Re_R} \\ \text{with } y \text{ such as } [\mu_{\Re_R} - y\,\sigma_{\Re_R}, +\infty] = 95\% \end{cases} \tag{4.7}$$

In this expression $\mu_{\Re_R}$ (respectively $\sigma_{\Re_R}$) is the arithmetic mean (respectively the standard deviation) of the sample $\Re_R$.

4.4.3/ Case result computation

The best and worst cases are aggregated by using the arithmetic mean of R_A and R_O as follows:

$$\mathcal{A}_C \left\{ R_C = \mu(R_A, R_O) \right. \tag{4.8}$$

4.4.4/ Value and Convergence sub-results

The value (respectively convergence) sub-result is the equivalent of the case result, except that only the end of optimization value (respectively number of evaluations) is considered. The value (respectively convergence) sub-result is computed the same way as the case result except that runs' results are replaced by runs' normalized value sub-results (respectively normalized evaluation sub-results). The computation of those sub-results are given by equation 4.9 and equation 4.10.

$$\mathcal{A}_V \begin{cases} R_V = \mu(a, b) \\ a = \max\{\Re_V\} \\ b = \mu_{\Re_V} - y\,\sigma_{\Re_V} \\ \text{with } y \text{ such as } [\mu_{\Re_V} - y\,\sigma_{\Re_V}, +\infty] = 95\% \\ \Re_V = (V_N^1, \cdots, V_N^{N_T}) \end{cases} \tag{4.9}$$

$$\mathcal{A}_K \begin{cases} R_K = \mu(a, b) \\ a = \max\{\Re_E\} \\ b = \mu_{\Re_E} - y\,\sigma_{\Re_E} \\ \text{with } y \text{ such as } [\mu_{\Re_E} - y\,\sigma_{\Re_E}, +\infty] = 95\% \\ \Re_E = (E_N^1, \cdots, E_N^{N_T}) \end{cases} \tag{4.10}$$

4.5/ Global score computation

In order to produce the global score, cases results should be summarized. Aggregation summarizes a set of data into a single indicator. Global score computation process is composed of three steps of aggregation:

- Step 1: All cases results are aggregated over *MaxFEs*, giving $N_D \times N_F$ *MaxFEs* results (R_M);
- Step 2: All *MaxFEs* results are aggregated over functions, giving N_D functions results (R_F);

- Step 3: To finish, all functions results are aggregated over dimensions, giving the main score (S).

A symbolic summary of this process is presented in equation (4.11) and in figure 4.6.

$$S = \mathcal{A}_D\left(\mathcal{A}_F\left(\mathcal{A}_M(R_C)\right)\right) \tag{4.11}$$

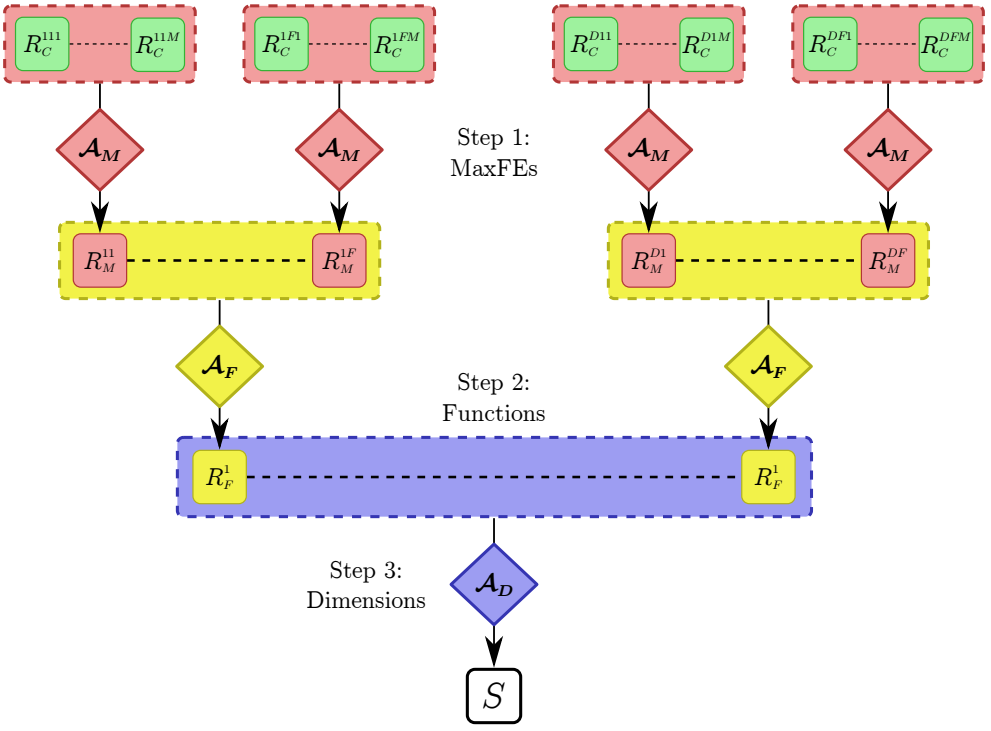


Figure 4.6: Global score computation flowchart

First, Aggregations by *MaxFEs*, by test functions and by dimensions will be presented in sub-sections 4.5.1, 4.5.2 and 4.5.3. Secondly, how other scores are computed will be explained in sub-section 4.5.4. Thirdly, how sub-scores are computed is detailed in sub-section 4.5.5.

4.5.1/ Aggregation of *MaxFEs*

In an industral context, if the solution is not sufficient to justify the investment of time, it is not worthy to optimize. Thus, the higher the *MaxFEs* the higher the chance to have a useful result. That is why, in this context, the higher the *MaxFEs*, the higher the weight should be. The aggregation method retained for *MaxFEs* is the weighted sum as given in equation 4.12. In this expression, the superscript f refers to the fixed function and the superscript d refers to the fixed dimension.

$$\mathcal{A}_{\boldsymbol{M}} : \begin{cases} R_M^{df} = \sum_{m=1}^{m=N_M} w_m \, R_C^{dfm} \\ w_m = W_m / \sum_{m=1}^{m=N_M} W_m \\ with \quad W = \{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1\} \end{cases} \tag{4.12}$$

In equation 4.12, W's values have been chosen with consideration of the following remarks. Article [44] mentions that, as we know, metaheuristics cannot successfully compete with mathematical programming methods when the computational budget is lower than a few hundred times the problem dimensionality. From this remark, it can be deduced that $100 \times D$ is the budget lower limit of meta-heuristics usage. As the firsts *MaxFEs*, from 0.01 to 0.05, are just above the lower limit they should have very low weights. Article [44] uses six computational budgets differing by two orders of magnitude. A similar choice has been made here by using seven *MaxFEs* over two orders of magnitude. From this, it can be considered that the last *MaxFEs*, 0.5 and 1, are relevant choices for metha-heuristic usage, as confirmed by [124] which only uses those Budgets to compute scores. *MaxFEs* with value of 0.5 and 1 should be heavily weighted. By weighting every *MaxFEs* its own value, the higher *MaxFEs* represent a little more than 50% and 25% of the final result and the lower ones only a few percent of this result. This choice of weight is consistent with the previously stated remarks.

4.5.2/ Aggregation of test functions

The problems faced by industrial companies are hard, multi-modal and probably close to hybrid and composite functions. Thus, test functions should be weighted differently. Moreover, optimization is meaningless if the performance increase of the result is not high enough to justify the time and energy costs. Consequently, as no algorithm can obtain convincing results on every problems topology, an algorithm is considered interesting if it performs well on some specific functions. An algorithm with both good or bad function results should have a better score than another one with only average function results. That is why a continuum method, with exponent superior to 1, has been chosen to aggregate over the test functions [21]. The mathematical expression of this method is given in equation 4.13. The choice of the weigth values is based on experimental observations.

$$\mathcal{A}_{\boldsymbol{F}} : \begin{cases} R_F^d = \sqrt{\sum_{f=1}^{f=N_F} w_f \left(R_M^{df}\right)^2} \\ w_f = W_f / (\sum_{f=1}^{f=N_F} W_f) \\ with \begin{cases} For \quad f = 1, 2; \quad W_f = 5/N_{uni} \\ For \quad f = 3, \cdots, 9; \quad W_f = 6/N_{multi} \\ For \quad f = 10, \cdots, 12; \quad W_f = 8/N_{hybrid} \\ For \quad f = 13, \cdots, 15; \quad W_f = 9/N_{composition} \end{cases} \end{cases} \tag{4.13}$$

4.5.3/ Aggregation of dimensions

In an industrial context, it is more important for an algorithm be efficient on high dimension problems than on low dimensional ones. The results from high dimensional problems should be weighted more than the results from low dimensional problems. In fact, an algorithm efficient for every dimension between 1 and 50 is looked after. An algorithm with average results for each problem dimension should have a better score than an algorithm with both good and bad results. This is the reason why a continuum method with an exponent inferior to −1 has been chosen [21]. The mathematical expression of this method is given in equation 4.14.

$$\mathcal{A}_D : \begin{cases} S = \sqrt[-2]{\sum_{d=1}^{d=N_D} w_d \left(R_F^d\right)^{-2}} \\ w_d = W_d / (\sum_{d=1}^{d=N_d} W_d) \\ with \quad W = \{10, 20, 30, 50\} \end{cases} \tag{4.14}$$

4.5.4/ Other results

In order to complete the benchmark, the global mean and global standard deviation are determined. To this end, all the results of the optimization cases are considered. This set is called $\Re_C$ and the mean is simply called $\mu_{\Re_C}$ (respectively standard deviation $\sigma_{\Re_C}$). Another quantity is introduced, the global SNR (Signal-to-noise ratio):

$$SNR = -10\ log\left(\frac{1}{\mu_{\Re_C}}\left(1 + \frac{3\sigma_{\Re_C}^2}{\mu_{\Re_C}}\right)\right) \tag{4.15}$$

4.5.5/ Sub-scores

The computation of the global score is highly important as it gives an overall idea of the tested algorithm's behavior. But this overall score can also be limiting. Indeed, the global score does not allow to classify the algorithms according to their intrinsic characteristics. A new issue arises: how can one detect the most efficient algorithm for a particular problem? To answer this question, one should use the sub-scores.

Dimension, function and *MaxFEs* sub-scores are sub-set sub-scores, meaning that they are computed the same way as the global score but by with a sub-set of cases results. For example, $D10$ sub-score is computed the same way as the global score using only the cases of dimension $D = 10$. Of course, in this example, dimension aggregation step is not achieved as only one dimension is represented.

Information sub-scores are computed the same way as the global score except that the cases' results are replaced by corresponding sub-results.

Sub-SNRs and sub-means are computed using the same values as the corresponding sub-scores. In the presented score computation method, it has been considered to use a

sub-mean (limited to a sub-score) instead of a mean, and respectively a sub-SNR instead of a SNR.

4.6/ Scoring method analysis through algorithms testing

In order to analyze and compare the proposed benchmark to the CEC one, different algorithms have been tested. The idea is to assess if the general conclusions from the literature are verified. For this task, a set of algorithms should be chosen. Indeed, thoroughly designing a consistent set of algorithms in order to test a benchmark is a work in itself. Accordingly, the aforementioned set is not completely justified and may include a bias. The motivation behind the choice of algorithms made is explained hereinafter. In order to fit the context of this work, the chosen algorithms should be classified as global search meta-heuristics. The algorithms represent different families of meta-heuristics as this benchmark is a generic tool. There is no guarantee that an up-to-date variant of an algorithm is a good representative of its family. Hence its initial version is used. Moreover, the hybrid algorithms are not considered in this article because there are too many references in the literature. A limited number of algorithms has been selected to avoid burdening the reader with too much information. From articles [9, 13, 125], particle swarm optimization, evolution strategy, genetic algorithm, swarm intelligence and differential evolution seem to be efficient families of algorithms. Respectively for each of those families, a representative has been chosen: The inertial PSO [45], the CMAES [52], an unpublished in-house Genetic algorithm [126], the Cuttlefish [56] and a population-based version of the simulated annealing [57]. More details about the algorithms and their settings can be found in section A.7.

In order to test the proposed benchmark several scores analysis will be done. First, the global score consistency with regards of the literature will be verified in sub-section 4.6.1. Then, the possibility of using different scoring strategy will be explored in sub-section 4.6.2. Finally, sub-scores will be analyzed in sub-section 4.6.3.

4.6.1/ Global score review

First, the proposed benchmark score is compared to the CEC one, considered as a reference and called here the classic benchmark. The algorithms results with both scoring methods are presented in figure 4.7.

From figure 4.7, the CEC Benchmark shows very high scores for CMAES and Cuttlefish and very low scores for the other algorithms. From the literature, this seems surprising. According to [9], Evolutionary algorithms, such as CMAES, PSO and Cuttlefish should obtain good results and PSO seems to be the best one. Conforming to [13], PSO and GA are the most prolific families of algorithms in terms of publications since 2004. It suggests that the scientific community considers that they perform well. X.-S. Yang confirms this assertion and recognizes that PSO and GA have become the hotspots of the current research in optimization and computational intelligence [125]. The benchmark proposed presents more homogeneous scores with limited maximal values. From the results obtained, this benchmark is in agreement with the literature unlike the CEC benchmark. CMAES algorithms show strong potential [52]. For both benchmarks, the assertion is confirmed and the CMAES is the reference algorithm. Due to their architecture, SA and Genetic

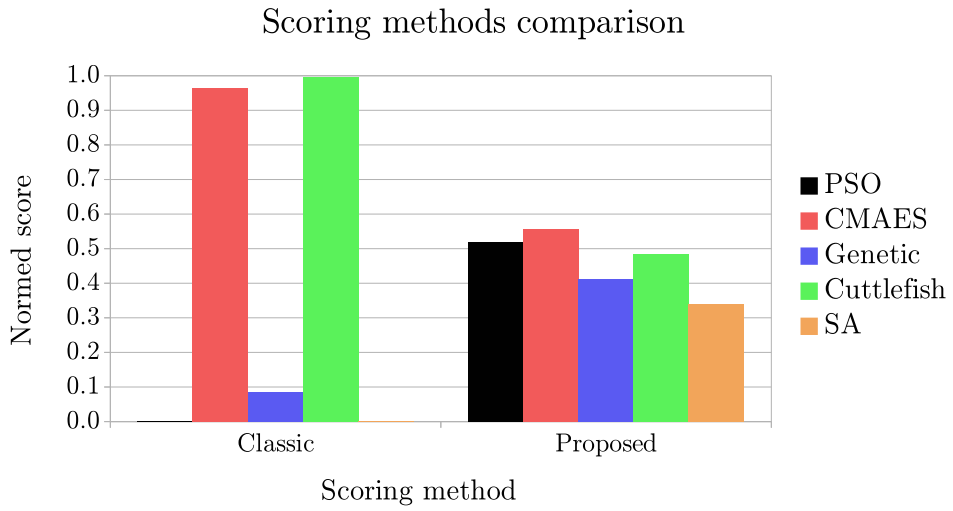


Figure 4.7: Scores from the CEC benchmark and the proposed benchmark for classical metaheuristic algorithms

algorithms are based on a large part of random. This explains why they have the lower results. Cuttlefish and PSO have close results because they both use displacement to the local and global best result. In fact, they are based on the well-balance between random and logic. Finally, CMAES is the best algorithm because it uses the most sophisticated mechanisms to collect information about iterated values of the objective function. Hence the wise choice of the next points to evaluate. An important source of improvement in optimization is the algorithm hybridization. To go further in this work, testing hybrid algorithms should be considered with regards to the results obtained for the initial algorithms. For instance, a PSO-CMAES hybridization, as proposed by [127], could obtain better results.

The lack of consistancy of the classic benchmark is explained in sub-section 4.6.2.1.

4.6.2/ Discussion about other scoring method

In this section other scoring methods will be discussed. In sub-section 4.6.2.1, CEC scoring method will be reviewed and in sub-section 4.6.2.2 classical statistical method will be examined.

4.6.2.1/ Issues with CEC scoring

First of all, CEC2017 scoring method is considered. This scoring method uses two scores: one based on the mean and median of optimizations raw values and the second one, based on the rank of considered algorithm on all optimization cases compared to those on CEC database.

The second score is not relevant in the case of a researcher willing to develop, improve or tune optimization methods. First, such a method is not convenient to set for an individual. Indeed, it requires the researcher to collect data on performances of other algorithms on a set of optimization problems and to set an interactive ranking method, such

as [128]. Here, a ranking method is qualified as interactive if it allows to evaluate an algorithm with a set of algorithms previously evaluated. Second, in this context, a researcher will look for an evaluation method whose result does not depend on external factors. With an interactive ranking method, the scores of every algorithms will change every time a new algorithm is evaluated. With a non-interactive ranking method, the result depends on the chosen algorithms for comparison. On the opposite, an interactive ranking method is a consistent choice to help the researcher identying up-to-date algorithms by establishing a global competition. Finally, it may be considered that ranking and competition encourage playing the 'up-the-wall game' which is a poor habit as explained by [12].

The first score is computed thanks to equation 4.16. In section 4.3.3, it has been mentioned that using unnormalized values to compute a global score can lead to some errors. To prove this error is made by CEC and not by the proposed benchmark, a thought experiment is proposed. An algorithm which perfectly performed on all the functions except one is considered. Perfectly performing means reaching f_{min} in 0 evaluation. For the parasitic function, the algorithm reachs f_{max} with FEs evaluations. Alternately, every function is considered as a parasitic function, leading to 15 cases. Table 4.4 shows for each case what result is achieved by both scoring methods.

$$\begin{cases} Score1 = (1 - \frac{SE - SE_{min}}{SE}) \times 50 \\ SE = 0.1 \times \sum_{f=1}^{15} ef_{D=10,F=f} + 0.2 \times \sum_{f=1}^{15} ef_{D=20,F=f} \\ +0.3 \times \sum_{f=1}^{15} ef_{D=30,F=f} + 0.4 \times \sum_{f=1}^{15} ef_{D=50,F=f} \end{cases} \tag{4.16}$$

Table 4.4: Scoring comparison for the thought experiment

Case	Detail	SE	CEC	Proposed Benchmark
1	$F1$ = 9.00e+10	9.00e+10	0.000	0.954
2	$F2$ = 1.00e+10	1.00e+10	0.000	0.954
3	$F3$ = 3.40e+02	1.20e+04	0.997	0.985
4	$F4$ = 8.45e+03	2.01e+04	0.599	0.985
5	$F5$ = 1.50e+03	1.30e+04	0.923	0.985
6	$F6$ = 6.25e+02	1.20e+04	0.998	0.985
7	$F7$ = 1.00e+03	1.23e+04	0.976	0.985
8	$F8$ = 2.00e+07	2.00e+07	0.001	0.985
9	$F9$ = 9.10e+02	1.20e+04	0.999	0.985
10	$F10$ = 1.00e+10	1.00e+10	0.000	0.951
11	$F11$ = 2.00e+10	2.00e+10	0.000	0.951
12	$F12$ = 5.00e+16	5.00e+16	0.000	0.951
13	$F13$ = 3.00e+07	3.00e+07	0.000	0.945
14	$F14$ = 1.47e+03	1.21e+04	0.994	0.945
15	$F15$ = 2.48e+04	1.30e+04	0.925	0.945

It is obvious that some functions, such as 3 and 6, have no impact on CEC score 1. An algorithm with a poor performance on these functions will have an amazing CEC score. On the opposite, an algorithm with an insufficient performance on Function 1 or Function 11 will be considered as the worst one ever. Once taking this into account, the CEC score seems to deeply depend on this fact and it can be declared as unstable. On the

opposite, the results from the proposed benchmark remains stable for all the cases. The only difference between the cases is due to functions weights. This explains the results from sub-section 4.6.1. Let us consider PSO and CMAES results. In sub-section 4.6.3, it is shown that PSO obtains weak results on Function 1 and satisfying ones on Function 15. On the opposite, CMAES obtains good results on Function 1 and poor ones on Function 15. According to table 4.4, Function 1 is important for the computation of the CEC score and Function 15 almost not. So it is easy to understand why one has good score whereas the other one has poor one.

4.6.2.2/ Alternatives performances characteristics choice

The proposed benchmark comes with an unusual set of performance characteristics used to evaluate an algorithm on a case. This choice leads to a set of scores that researchers are not familiar with. Sub-section 4.6.3 will help understanding proposed scores. Still, one could question the choice of not using 'classical' statistical metrics as for instance those presented in [48].

First of all, the proposed set of performance characteristics, while providing consistant results within the context, is not the only one possible. However other performance characteristics have to be considered as replacements of the proposed ones, they should respect some basic rules. They should be relevant to the benchmark goals and context [46]. They should cover the three categories defined in [46]: efficiency, quality and reliability. They should follow the advice of benchmarking guide articles, such as [19, 46].

Common statistical performance characteristics [48], will now be discussed in order to guide researchers willing to modify this benchmark either for this context or another one. First, some of the proposed metrics rely on running time which can only be considered in a context in which evaluations are short enough not to be neglicted. In enginering design optimization, as advised by [19] it should be avoided. If algorithm complexity is to be evaluated, CEC procedure [121] could be used. A set of six performance characteristics, inspired by [48], has been chosen. The performance characteristics are:

- E_{hit}, the mean number of evaluation done when the optimization algorithm has reach the target value.
- V_{mean}, the mean of the gross values of the draw.
- E_{mean}, the mean of the gross evaluations of the draw.
- E_{best}, the minimum of the gross values of the draw.
- H_r, The ratio of runs reaching the target value.
- V_{std}, the standard deviation of the gross values of the draw.

These performance characteristics have been used to compute scores by the same normalization and aggregation procedure as the one used for information sub-scores. Computed scores, for every algorithms, are presented in Table 4.5. Results based on the target reaching, E_{hit} and H_r, should not be used. Indeed, the proposed benchmark is built with the idea that designers are not looking for refined results. Yet, those scores depend on the ability of the algorithm to thoroughly refine the result to find the target with the high precision. This explains why those results are not consistant with literature. Another issue

comes for V_{std}, This result, while useful for a single optimization case, lose all sense while normalized and aggregated. Indeed, this measure is to be used in addition of V_{mean}, in order to estimate how much the gross value is allowed to deviate. To properly apply this performance characteristic, all case results should be reported in a table. However, doing so is in opposition to synthesising information which is important for benchmark. V_{best} measures the best result of a run in terms of end of optimization value while *Alpha* does the same in terms of efficiency. V_{best}, is a logical choice to measure reliability in a context in which running time does not matter. As it it not the case here, reliability must be measured by taking convergence speed into account. V_{mean} (respectively E_{mean}) is almost equivalent to *Value* (respectively *Convergence*). The two differences are normalization and aggregation order. The first one is the mean of the results whereas the other one is the mean of the best and worst of the results. *Value* and *Convergence* tend to be a little lower than their equivalent. The highest measured difference is 0.024 for Cutllefish value.

The performance characteristics of the studied statistical methods are not relevant within the context of this work which corresponds to building a benchmark consistant with designers' needs. Some lead to inconsistent results, others do not fit the designers' needs and some are equivalent to the proposed ones. To conclude, using statistical methods' performance characteristics is not recommended for this article context.

Table 4.5: Alternatives performances characteristics scores

Kind	Category	Name	PSO	CMAES	Genetic	Cuttlefish	SA
Global	Main score	E_{hit}	0.000	0.000	0.000	0.000	0.000
Infos	Quality	V_{mean}	0.426	0.622	0.482	0.677	0.481
		E_{mean}	0.884	0.435	0.350	0.000	0.000
	Reliability	V_{best}	0.497	0.654	0.545	0.747	0.502
		H_r	0.000	0.000	0.000	0.528	0.000
		V_{std}	0.443	0.652	0.592	0.697	0.556

4.6.3/ Sub-scores analysis for the proposed method

Table 4.6 presents the global score and the sub-scores obtained for the tested algorithms with the proposed benchmark. Let us note that the CEC benchmark does not propose sub-scores in order to be aware of the strenghts and weaknesses of each algorithm. Sub-sections 4.6.3.1 and 4.6.3.2 will discuss algorithms strengths and weaknesses by using table 4.6's data ploted into graphs. Sub-section 4.6.3.3 presents how this table scores could be used to select algorithms for a given problem.

4.6.3.1/ Information sub-scores analysis

Information scores are presented by figure 4.8. First, from value and convergence results, some conclusions can be made, for instance, PSO converges very fast to a value satisfying enough. This can be explained by its settings which promote a global search. Cuttlefish finds a good value but does not converge. This is due to the use of a large part of random and a lack of an own stopping criterion. SA also does not converge because it does not find the global minimum and does not possess a stopping criterion by itself. CMAES is very interesting in an industrial context to find a good value in a reasonable computation time. Secondly, from the alpha and omega results, some other conclusions can be made. If

Table 4.6: Scores for the selected metaheuristic algorithms from the benchmark proposed

Kind	Category	Name	PSO	CMAES	Genetic	Cuttlefish	SA
Global	Main score	Score	0.518	0.556	0.412	0.484	0.340
Infos	Quality	Value	0.445	0.629	0.500	0.701	0.485
		Convergence	0.870	0.439	0.353	0.000	0.000
	Reliability	Alpha	0.568	0.577	0.459	0.514	0.351
		Omega	0.471	0.539	0.372	0.469	0.329
Sub-set	Dimension	10	0.581	0.625	0.422	0.480	0.359
		20	0.549	0.578	0.423	0.482	0.344
		30	0.526	0.557	0.417	0.484	0.339
		50	0.494	0.537	0.404	0.485	0.335
	Function	1	0.303	0.929	0.248	0.681	0.138
		2	0.437	0.888	0.285	0.678	0.225
		3	0.207	0.189	0.109	0.670	0.018
		4	0.203	0.942	0.045	0.194	0.024
		5	0.307	0.422	0.242	0.182	0.112
		6	0.229	0.221	0.128	0.087	0.105
		7	0.398	0.355	0.261	0.246	0.262
		8	0.542	0.391	0.337	0.382	0.366
		9	0.179	0.163	0.015	0.022	0.014
		10	0.381	0.479	0.189	0.617	0.129
		11	0.638	0.575	0.412	0.489	0.398
		12	0.801	0.723	0.545	0.572	0.547
		13	0.695	0.302	0.526	0.423	0.464
		14	0.216	0.016	0.245	0.030	0.079
		15	0.658	0.038	0.761	0.406	0.549
	MaxFEs	0.01	0.315	0.375	0.231	0.433	0.238
		0.02	0.378	0.394	0.257	0.460	0.267
		0.05	0.448	0.440	0.288	0.477	0.300
		0.1	0.478	0.485	0.326	0.480	0.318
		0.2	0.504	0.520	0.365	0.485	0.331
		0.5	0.523	0.555	0.406	0.484	0.343
		1	0.532	0.584	0.448	0.486	0.348

a case can be run for a long time, PSO and CMAES are the best options. Indeed, they are stochastic algorithms and have greater chance to find an interesting result if run several times. PSO is a more unsafe choice as it is likely to obtain poorest results if the case can be run only a few times. It is not necessary to run SA several times because its alpha and omega result are very close. This could be due to its settings that gives lots of importance to random. In this case, this algorithm spends a lot of time on global search leading to similar results through runs. On the opposite, Genetic algorithm has to be run several times. This could be explained by the fact that genes are not modified enough through a run. In this case, the algorithm depends a lot on the initial genes randomly obtained.

4.6.3.2/ Sub-set sub-scores analysis

Figure 4.9 presents the scores of the algorithms for different dimensions. The scores of the CMAES, PSO and SA algorithms decrease when dimension increases. The ones of Genetic and Cuttlefish algorithms remain steady and seem to be better designed to explore large search spaces. It can be deduced that the more an algorithm is based on random, the steadier the obtained solutions remain with regards of the dimension of the problem.

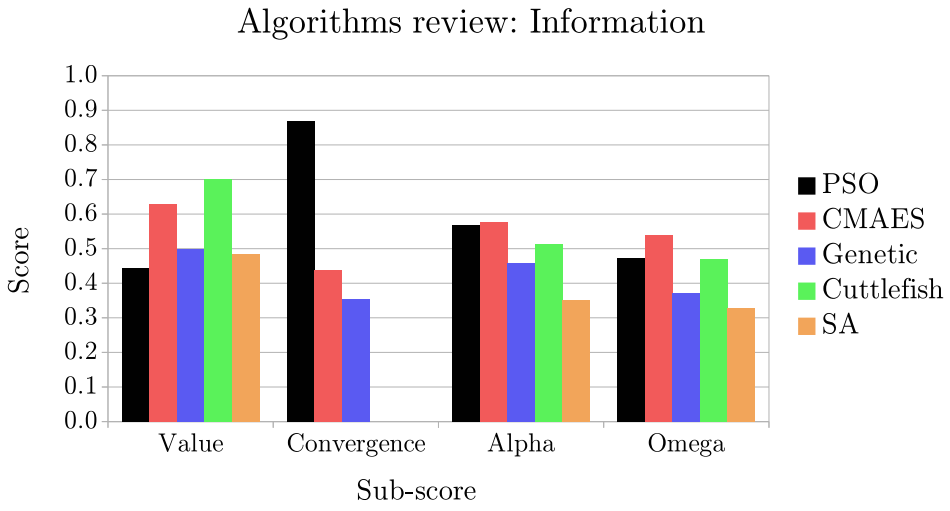


Figure 4.8: Meta-heuristic algorithms analysis: Information sub-scores

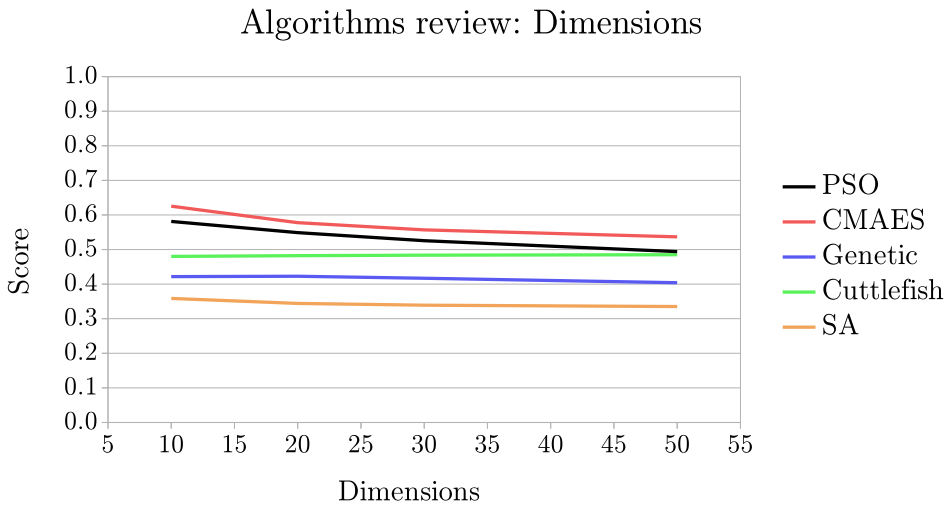


Figure 4.9: Meta-heuristic algorithms analysis: Dimensions scores

Figure 4.10 presents the function scores of the algorithms. A first observation can be made: the 'no free lunch' theorem is respected. Some tendencies can be observed. PSO and Genetic Algorithm are good on hybrid functions while CMAES and Cuttlefish perform well on uni-modal functions. No absolute rule concerning efficiency can be established. This could mean that the functions' choice and classification inherited from CEC are not perfectly consistent. This observation is partially explained thanks to [41]. The reasons why some functions such as Function 9 or Function 14, are harder to overtake than other, as Function 12, are quite difficult to explain. To do so, a proper study of both the mechanisms of the algorithms [12] and the landscapes of the functions [41] should be conducted.

To conclude, *MaxFEs* scores presented in figure 4.11 show that whether the number of evlauations is small or not, the appropriate algorithms will not be the same, as mentioned by [44]. According to [44], PSO performs well when evaluations are restricted unlike GA

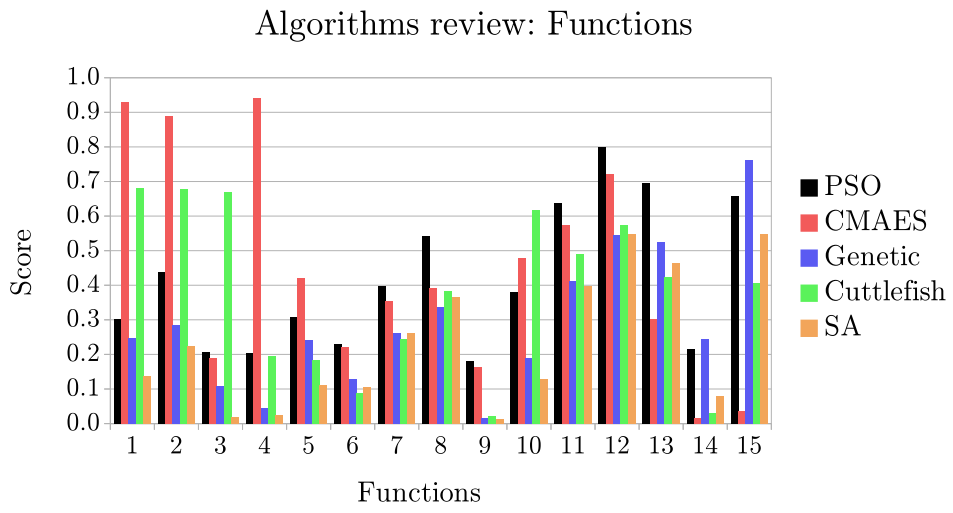


Figure 4.10: Meta-heuristic algorithms analysis: Functions scores

which is more efficient when evaluations are allowed in a large amount. In the case of PSO and GA, it is confirmed by figure 4.11.

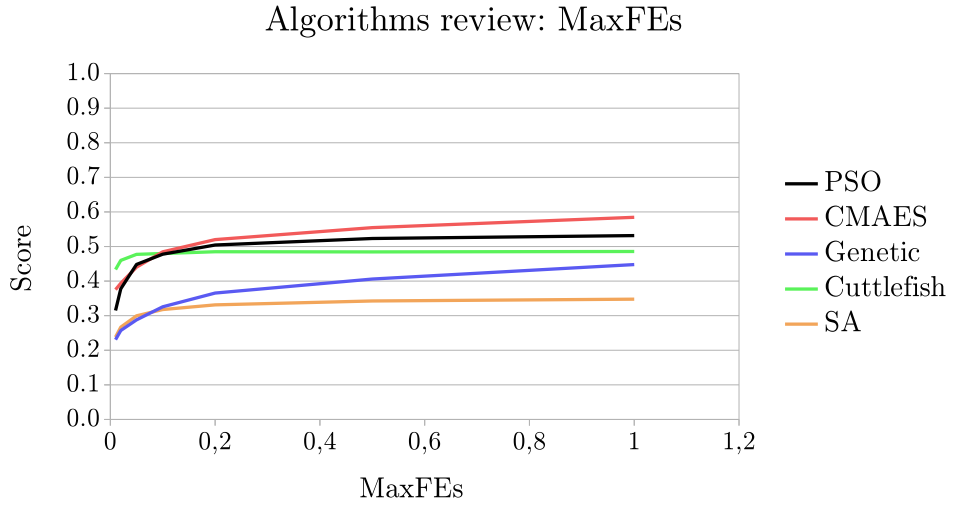


Figure 4.11: Meta-heuristic algorithms analysis: *MaxFEs* scores

4.6.3.3/ Using Scores to select suitable algorithms for a problem

In this sub-section, the way how scores could be used to select suitable algorithms for a given problem will be presented. The idea is to discuss a set of relevant scores as in fuzzy logic methods in order to define the suitable algorithms. This method will be presented throughout a practical case.

This practical case is a real-world optimization problem, described in [72]. It consists in optimizing an electrical device: a safety transformer. This problem has been considered

in its single-objective form, in which only the device mass is to be minimized. As the proposed benchmark is designed to the evaluate algorithms dedicated to solve single-objective problems, it has been chosen to use the single-objective form of the test problem. The proposed benchmark is not suited to select a proper algorithm for a multi-objective problem. In this specific case, one should modify the proposed benchmark for it to use multi-objective problems. It has seven mixed bounded variables and seven inequality constraints. This constrainted problem has been solved with normalized constraints and a normalized objective, as in OIA methods described in [21]. Constraints have been taken into account by using Lagrange multipliers methods. Objective and constraints are computed by a numerical code directly implemented in the optimization engine as a function.

Only the scores relevant to the problem characteristics will be considered. Some indications on how to choose relevant scores are presented here:

- First, the main score is always to be taken into account, as it gives a global efficiency measure. In the end, this score will be used as a reference to judge other scores.
- If the designer is balancing the product efficiency and the optimization time, as mentioned by [8], both value and convergence sub-scores should be used.
- If only the product quality matters, only value sub-score will be considered. It can be the case if the problem is not CPU demanding with regards of the resources or if the problem is very competitive.
- If the designer is using optimization as a continuous improvement tool, as mentioned by [8], only the convergence sub-score should be considered. Indeed, in this case, the optimization is used to quickly improve the efficiency and its main objective will be a short running time.
- If the designer could run many times the optimization case, alpha sub-score should be used. For instance, in the case of a new product with a long design phase.
- If the designer could run the optimization case only a few times, the Omega sub-score should be used. For instance, if the optimization has to be included in a softaware as a solving method.
- Dimension sub-scores around the problem dimension should be used.
- Relevant function sub-scores should be considered. The functions may be relevant depending on the properties shared with the problem. Those links could be made using function metrics [41] or by using the function categories defined in table 4.1.
- *MaxFEs* sub-scores around the optimization case should be used.

This problem is not CPU demanding. Indeed, with the numerical model inside the optimization engine, it takes one second to perform ten thousand evaluations on the laptop used for the experiment[1]. In this case, it cannot be considered that the design phase is short as it is a research project including development. The numerical model of

[1]Micro-Star International GL62 6QF (Processor: Intel(R) Core(TM) i7-6700HQ CPU @ 2.6GHz, 4 cores, 8 procesors; RAM: 8Go; Graphic Card: NVIDIA GeForce GTX 960M; OS: Microsoft Windows 10 Familly)

this problem is based on function compositions as in hybrid functions. In this case, the designer is looking for a final optimal solution and not a first approximation. *MaxFEs* would be set to 1.

Which is why chosen scores are: main, value, alpha, $D10$, $F10$, $F11$, $F12$ and $M1$. According to the semantic scale presented in [21], scores will be labeled as 'low' if under 0.35, 'average' if between 0.35 and 0.65 and 'high' if above 0.65.

With those assumptions, one can consider that:

- All PSO scores are average, except for $F12$ which is high. PSO should have average to high results.
- All CMAES scores are average, except for $F12$ which is high. CMAES should have average to high results.
- All GA scores are average, excepted for $F10$ which is low. GA should have average to low results.
- All Cuttlefish scores are average except for value which is high. Cuttlefish should have average to high results.
- All SA scores are average, except for main, $F10$ and $M1$ ones which are low. At first, SA should have average to low results. However as most of its sub-scores are better than its main score, SA is probably better for this case than on a general approach. It is not possible to conclude how SA will work.

From the previous remarks, it could be considered that PSO, CMAES and Cuttlefish are suitable choices. GA and SA are not recommended but might obtain good results.

All algorithms have been tested on the safety transformer problem. Algorithms had to minimize the normalized mass of the device. Mass has been normalized using the formula given in equation 4.17, in which 2.5Kg correponsds to a 0.9 level of satisfaction and 5Kg to 0.5. Each algorithm has 51 runs, with *MaxFEs* = 1, to find the best objective-function value while respecting the constraints. Running time is not an important criteria in this case. Still it could be used to discriminate algorithms with close values.

$$\begin{cases} f_{normalized}(x) = \alpha \cdot f(x) + \beta \\ \alpha = \frac{0.5-0.9}{5-2.5} = -0.16 \\ \beta = 0.9 + 0.16 \cdot 2.5 = 1.3 \end{cases} \tag{4.17}$$

Table 4.7 presents algorithms results on this problem. PSO, CMAES and Cuttlefish have close and very high V_{best}. The best value of the CMAES, 0.930, corresponds to a mass of 2.311Kg which is the best result obtained in [72]. SA performed as well as Cuttlefish. GA performed a little poorer than the other algorithms. The results confirmed that the method correctly defines the suitable algorithms. The proposed benchmark sub-scores could be used to correctly choose algorithms to solve a given problem.

4.7/ Conclusion

In the last few years, a great number of new algorithms, most of them are metaheuristic, has been developed. A hot topic in the community is to be able to measure the efficiency

Table 4.7: Algorithm results on Safety transformer problem

Result	PSO	CMAES	GA	Cuttlefish	SA
V_{best}	0.929	0.930	0.843	0.918	0.920
V_{mean}	0.919	0.418	0.500	0.829	0.856
E_{mean}	0.318	0.237	0.000	0.001	0.000
R_C	0.694	0.426	0.384	0.553	0.580
R_V	0.917	0.465	0.438	0.799	0.840
R_K	0.306	0.439	0.290	0.001	0.000
R_A	0.843	0.852	0.736	0.630	0.631
R_O	0.545	0.000	0.032	0.476	0.528

of these algorithms, to rank them and to select the right one for a given problem. As demonstrated by [13], benchmarking is the correct way to reach these objectives. Two benchmarks are quite renowned: CEC and BBOB. Their results are interesting but their scoring strategy could be improved. On one hand, the way of computing the main score can be enhanced. On the other hand, the computation of sub-scores could give more information concerning the algorithm's behavior and its efficiency with respect to the type of problem.

A benchmark, based on the CEC one, has been proposed. It is composed of 420 optimization cases with 51 runs per case. The cases are generated by the variation of three quantities: an objective function, a dimension and a *MaxFEs* coefficient. A scoring strategy has been developed to consider industrial needs and to avoid some errors in the algorithms' analysis. Several stopping criteria are used: target, *FEs* and algorithm convergence criteria. The run results take into account a normalized value for the objective function at the end of optimization and a normalized convergence speed. The results for the optimization cases are computed using statistics over the run results. The cases result goes through three steps of aggregation to produce the scores.

Finally, the scoring method has been analyzed. First, several metaheuristic algorithms have been tested on both selected scoring methods. These algorithms are PSO, CMAES, Genetic, Cuttlefish and SA. The idea is, using both benchmarks, to investigate if some conclusions, found in the literature, are confirmed by the algorithms scores. It appears that, with the proposed benchmark, the concluding comments are verified whereas with the CEC method they are not. Secondly, a thought experiment has been conducted to see the impact of using un-normalized results. It revealed that using the CEC scoring method is biased. Thirdly, the presented sub-scores have been analyzed. They have been used to made comments and remarks concerning the tested algorithms. Some of these comments are confirmed by literature. It has been demonstrated that some algorithms are efficient with few evaluations and some with more. The 'no free lunch' theorem has also been verified.

To properly use this benchmark, one should keep in mind its limits. It has been designed for global search metaheuristic algorithms. The main score is a measure of efficiency and not a measure of quality [46]. This benchmark has been designed for industrial needs. Thus, its design is centered around the designer's point of view.

The choice of the functions tested, inherited from CEC, is not perfect, thus the scores are not. For the proposed benchmark to evaluate algorithms efficiency on multi-objective

or uncertain problems, it should be modified.

5

Meta-optimization by design of experiment on benchmark used to tune PSO parameters

Contents

5.1/ Introduction

Optimization problems are solved more and more often by algorithms [8]. In engineering design optimization (EDO) [8], meta-heuristic optimization algorithms are widely used [8, 9, 129]. Even though there is not any mathematical evidence of their convergence to the global optimum, the metaheuristic methods have been shown to be very efficient. Efficiency being a measure of performance [46] balacing the result quality and the convergence speed [44]. Meta-heuristics are generally inspired by physical or biological phenomena [12]. For instance the Particle Swarm Optimization (PSO) draws its inspiration from the observation of some flock of birds [130]. The inertial version of this algorithm [54] is efficient and commonly used [9, 13, 125].

In EDO, efficiency is one of the three major areas of improvement defined by [8]. This topic is widely adressed by developping and improving algorithms leading to an ever increasing number of algorithms [13]. Due to the enthousiasm for algorithms, other approaches to improve efficiency, such as the tuning of the parameters of an algorithm, are not investigated enough [12].

Parameters are the coefficients of an algorithm driving its behaviour and adapting it to the problem to solve. The set of values given to parameters is referred to as a setting. Tuning parameters have always been a topic of interest [27], especially as the efficiency of a setting is strongly dependent on the problem. A common practice is to tune parameters manually [12] using empirical values, rules of thumb and the experience of the optimization expert. This method is not fully satisfying. Firstly, it does not guarantee the best performance. Secondly, it could not be done by a designer with no experience in optimization.

To avoid the drawbacks of this method, different solutions exist: using automatic setting tools such as IRACE [131] or AutoML [132], developping a self-tuning algorithm, as in [133], or performing meta-optimization (MO) [134]. According to [131], automatic setting tools such as IRACE may be inapropriate for problem requiring computational time superior or equal to several hours, which is the case in this thesis. Automatic setting method like AutoML require a databasis of similar problems solved with different settings, which is not the case in this thesis. Meta-optimization is the optimization of an algorithm by tuning its parameters. Performing MO requires less knowledge of the algorithm than developping a self-tuning algorithm. With regards of the ever-increasing number of new algorithms [12, 13] which are not well-known and studied yet, working on MO could be considered.

A MO approach requires an optimization method [8] and a tool to evaluate tested settings. Two optimization methods could be used: algorithm and design of experiment (DoE). DoE gives a good comprehension of the problem in a small number of evaluations [61]. Algorithmic optimization is more efficient but is a black box method and requires a large amount of evaluations, leading to a running time not many institutions could afford. Two tools are currently used to evaluate settings, objective-function test suites (OFTS) and real-case test suites (RCTS). As explained by [13], though test suites (TS) are still the usual way to evaluate and compare algorithms, to avoid false conclusion they should be tested on a benchmark. A benchmark is a tool evaluating an algorithm over a collection of an optimization problem [41]. As the benchmark is advised to evaluate the efficiency of an algorithm, it should also be used in MO. Current MO approaches seek for a single setting. According to the 'no free lunch' theorem [111], no algorithm could perform well on

every problem. Therefore it may be interesting to perform MO in order to obtain different settings suiting different problems. By choosing a benchmark with sub-scores, it may be possible to do so.

This article introduces a novel MO approach: a design of experiment on a benchmark. This approach will be presented through its application to a PSO [54]. DoE will use benchmark global score as performance measure to compute the main setting. Benchmark sub-scores will be used in a similar way to obtain sub-settings suiting different problems. Two goals are sought: first, finding an efficient setting. Second, determining information to adapt this setting to a particular problem.

The MO approach proposed in this chapter relies on the PSO and a benchmark which are introduced in section 5.2. Moreover, this approach is based on a design of experiment which is detailed in section 5.3. The results of the proposed MO approach will be discussed in section 5.4.

5.2/ Meta-optimization approach linked elements

This section presents the elements used by the proposed MO approach. This approach is applied to the PSO algorithm parameters which are detailed in sub-section 5.2.1. The DoE used by the proposed MO approach used a benchmark, presented in sub-section 5.2.2, as evaluation tool.

5.2.1/ Particle swarm optimization algorithm parameters

The proposed MO approach will be presented through its application. The algorithm chosen for application is the inertial version of PSO algorithm [54]. It has been chosen as it is efficient and commonly used. In addition, several MO have already been performed on this algorithm providing points of comparison. As the PSO algorithm itself has already been described in chapter 1, this section will focus on PSO parameters.

Parameters can be classified into two categories: algorithmic parameters and adaptation parameters. Algorithmic parameters, presented in table 5.1, control algorithm behaviour and thus have a direct effect on its efficiency. Indeed, they are used in the equations controlling the swarm, as for instance equation 1.4. Some studies attempt to find settings for the PSO to converge [135]. Currently, only empirical rules such as the Poli criterion exist [136]. Adaptation parameters, presented in table 5.2, adapt the algorithm to the problem. The settings of those parameters are strongly dependent on the problem and are often defined with the designer's help. The tuning scope of those parameters is narrow. Only algorithmic parameters are considered in this article. From now on, unless specified otherwise, parameters will refer to algorithmic parameters.

5.2.2/ Benchmark

In the past, DoE were used to determine a set of good parameters by using TS. Most studies were limited to a presentation of standard deviations of final TS results. In this article, MO is perfomed by using a benchmark. This benchmark should be chosen to evaluate tested settings with regards of the final user's expectations towards optimization. The majority of the companies try to achieve optimized design as a balance between cost,

Table 5.1: PSO algorithmic parameters description

Name	Details	Typical Values [61, 63, 133, 135]
N	Swarm particles number	$10 + 2\sqrt{D}$; $[10; 150]$
it_{max}	Iterate number	$10000\frac{D}{N}$; $[1000; 100000]$
c_1	Cognition parameter	$[05; 3]$
c_2	Social parameter	$[0.5; 3]$
w_{min}	Inertia factor minimal value	$[0; 0.5]$
w_{max}	Inertia factor maximal value	$[0.5; 2]$
V_{factor}	Speed control coefficient	$[0.01, 0.5]$

Table 5.2: PSO adaptation parameters description

Name	Details	Remarks
D	Problem dimension	Usually $D \in [1, 50]$ [8, 9]
x_{min}	Lower bound	Depends on the problem
x_{max}	Upper bound	Depends on the problem
v_{min}	Minimum velocity	Depends on the problem and V_{factor}
v_{max}	maximum velocity	Depends on the problem and V_{factor}

quality and time even if the design may not be the best [8]. From this remark, to test the presented approach, a benchmark evaluating algorithm in terms of efficiency should be chosen. In addition to fulfill the second goal of this work, having several sub-settings, the chosen benchmark should have several sub-scores. Two benchmarks, presented for special sessions during international congresses, are considered as reference [13]: the CEC [112] in IEEE Congress on Evolutionary Computation (CEC) and the Black-Box Optimization Benchmarking (BBOB)[113] in Genetic and Evolutionary Computation Conference (GECCO). CEC evaluates algorithms in terms of value quality while GECCO does it in terms of convergence quality. A new benchmark evaluating algorithms in terms of efficiency and proposing sub-scores has just been published [137]. Therefore, this benchmark, which is also detailed in chapter 4, has been chosen to test the presented MO approach. CEC and GECCO could perfectly be used to optimize algorithms in terms of value or convergence quality.

The chosen benchmark runs an algorithm on a set of test functions and provides a set of scores from the generated data. Its main score, defined as the 'global score', is a measure of efficiency. This score is a good indicator of the performance of the algorithm and it can be exploited by researchers in order to give them a general idea on the efficiency of different algorithms. Sub-scores provide different information on tested algorithms, for instance efficiency with regards of dimension. This benchmark has an original way to compute scores.

Raw data from optimization runs are used to compute value and convergence levels of desirability [116, 117]. To do so, normalization function, such as Harrington one [118], are used. Through several steps of aggregation, levels of desirability will be used to compute scores. Computing global levels of satisfactions from several metrics has already been done by [27] to handle multi-objective optimization problems. How the chosen benchmark works is detailed in chapter 4.

5.3/ Design of experiment

The objective of this DoE is to determine the setting maximizing the main score of the benchmark. The same exact method will be used for every other benchmark sub-scores. A full factorial experiment would need $3^6 = 729$ experiments, whose cost is considerable. Indeed, an experiment corresponds to a run of the benchmark, which is very expensive : a great number of optimization cases are solved. Our goal is to obtain the maximum amount of information by doing the minimum number of tests. In this context, a fractional factorial design is necessary [138]. The idea of such a design is to run only a sub-set of the full factorial experiments. The DoE modeling will be presented in sub-section 5.3.1 while the DoE results processing will be presented in sub-section 5.3.2.

5.3.1/ Design of experiment modeling

The hypothesis made in the elaboration of fractional designs is that some interactions are insignificant and they can be confused with factors whose influence is significant. The used experimental layout is presented in figure 5.1. It consists of two designs: one for the control factors (inner array) and one for the noise factors (outer array).

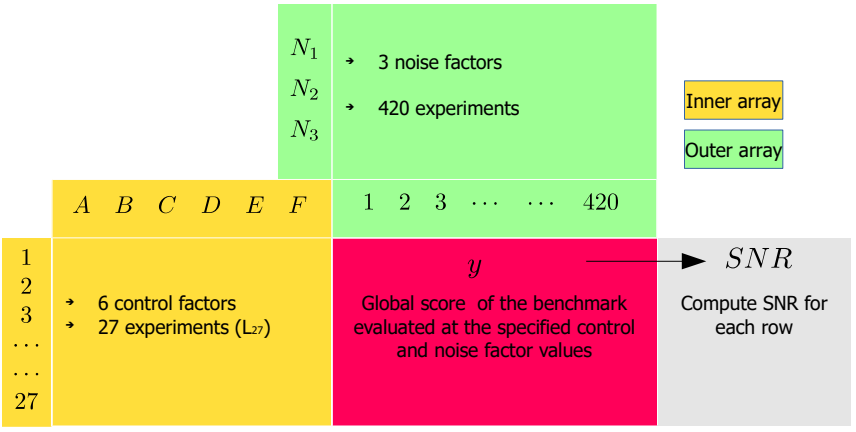


Figure 5.1: Summary of experimental layout

The combination of the inner array and outer array constitutes a complete parameter design layout (product design). The outer array is replicated for each of the runs in the inner array. A signal-to-noise ratio (SNR) is computed over the replicate. The performance measure thus reflects variations due to changes in the noise factors. Our experiment applies Taguchi method [139], which distinguishes control factors and noise factors.

- Inner Array and control factors : only the parameters of the algorithm are considered in this set (table 5.3). In our context, 27 runs are enough for the investigation of the six factors with the L_{27} orthogonal array.
- Outer array and noise factors. These factors are difficult or impossible or too expensive to control. This is the case for parameters which depend on the optimization

problems solved: function and dimension of the optimization problem. That is the reason why the physical parameters are defined as noise factors (table 5.4). For each experience of the outer array, a complete design of the noise factors is carried out. That is $4 \times 15 \times 7 = 420$ tests.

Table 5.3: Control factors and their levels for the experiment

Factors	Labels	Number of levels	Level 1	Level 2	Level 3
N	A	3	20	50	100
c_1	B	3	0.5	1	1.5
c_2	C	3	0.5	1	1.5
w_{min}	D	3	0.2	0.3	0.4
w_{max}	E	3	0.7	0.8	0.9
V_{factor}	F	3	0.05	0.1	0.3

Table 5.4: Noise factors and their levels for the experiment

Factors	Labels	Number of levels	Levels
D	N_1	4	$\{10, 20, 30, 50\}$
$Function$	N_2	15	$f_1 \cdots f_{15}$
$MaxFEs$	N_3	7	$\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1\}$

5.3.2/ Design of experiment results processing

For each experiment of the outer array, the following quantities are evaluated.

- The score of the benchmark is noted y.
- $\bar{y}$ (respectively s) is the arithmetic mean (respectively standard deviation) of the 420 measured values.
- The measured values y of the response of the process are transformed into SNR which provides a measure of the impact of noise factors on the performance. SNR is also calculated knowing that the criterion of the result y is to be maximized. The ratio is characteristic of the obtained dispersion around average. SNR is an indicator of robustness to noise. The larger the algebraic value of SNR the highest the performance. SNR is computed as in equation 5.1.

$$SNR(dB) = -10log\left[\left(\frac{1}{\bar{y}^2}\right)\left(1 + \frac{3s^2}{\bar{y}^2}\right)\right] \tag{5.1}$$

5.4/ Results and discussion

This section intends to discuss the results of the MO approach . First, DoE results for benchmark main score will be presented in sub-section 5.4.1. Then DoEs on the sub-scores of the benchmark will be analyzed through principal component analysis (PCA) in

sub-section 5.4.2. Finally, the proposed MO approach will be compared with other MO approaches in sub-section 5.4.3.

5.4.1/ Global study : analysis of the signal-to-noise-ratio

In order to find the optimal value, DoE will first study the influence of factors on the results of the tests to deduce the supposed optimal set of values, as in sub-section 5.4.1.1. Then, a validation test will be conducted on the deduced set of values, as in sub-section 5.4.1.2.

5.4.1.1/ Predicted best strategy parameters

SNR is indicative of the quality. The purpose of the Taguchi experiment is to find the best level for each operating parameter in order to minimize *SNR*. From the response graph (figure 5.2), the optimum level of each factor can be predicted as the level which has the lowest value of *SNR*. Thus, the optimal PSO setting for maximizing the global score of the benchmark is presented in table 5.5.

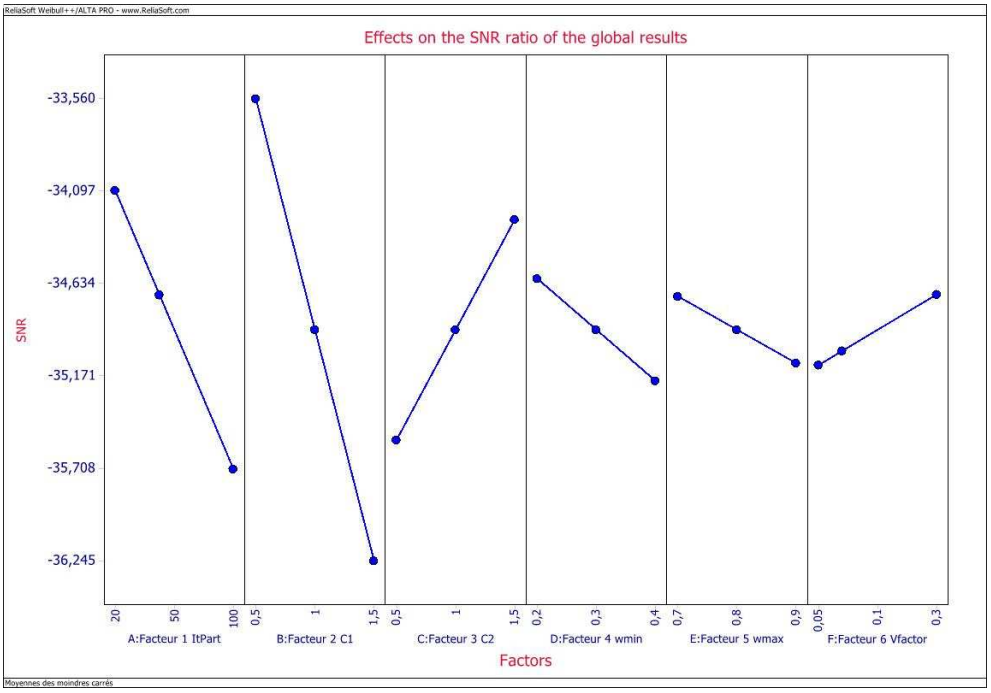


Figure 5.2: DOE factors' SNR Response graph

Table 5.5: DoE predicted PSO setting maximizing the benchmark main score

N	c_1	c_2	w_{min}	w_{max}	V_{factor}
A	B	C	D	E	F
20	0.5	1.5	0.2	0.7	0.3

5.4.1.2/ Validation test

The proposed combination of parameters is not part of the setting tested by the DoE. It is therefore necessary to perform a validation test. The results of this test, using the proposed combination of factors, are analyzed and compared to the predicted value theoretically. Since the theoretical result is close to the result of the validation test, the hypothesis of the additivity of the effects of all the factors and interactions is verified.

5.4.2/ Study by sub-sets and principal component analysis

In the previous part, an optimal setting maximizing the global score of the benchmark was obtained. This setting may appear as an efficient starting point for running PSO but is not suited for every problem. Indeed, it is clear that the setting strongly depends on the problem to deal with: landscape functions, dimension. These aspects are not taken into account with only a study on the global score. Further study is needed to better understand the impact of the parameters on the behavior of the PSO. To this end, three additional sets of DoEs were realized as specified in table 5.6.

Table 5.6: Study by sub-set - sets of DoEs for alternatives settings

	Case 1	Case 2	Case 3
Inner Array	6 control factors 27 experiments (L_{27})		
Outer Array	Dimension fixed (N_1) 2 noise factors 105 experiments	Function fixed (N_2) 2 noise factors 28 experiments	MaxFes fixed (N_3) 2 noise factors 60 experiments

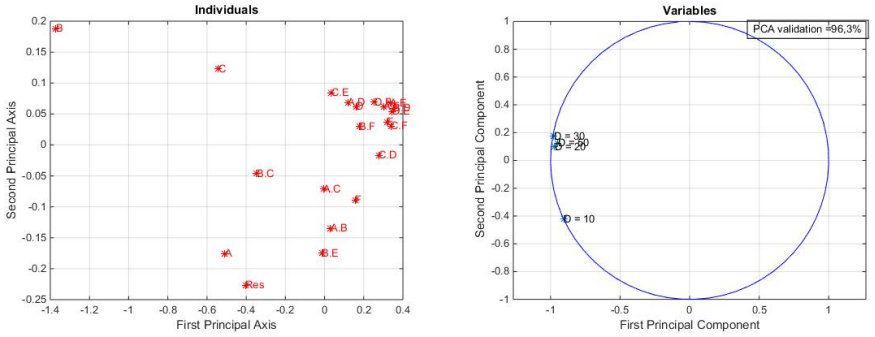
5.4.2.1/ Introduction to principal components analysis

Through sub-section 5.4.2, PCAs are performed. The general principle of the PCA is to reduce the size of a data set which depends on a large number of interdependent variables, while retaining all the information describing how the data varies between them. This result is obtained by the definition of a new set of variables, called principal components (PC), decorrelated from each other representing all the initial variables. The PCs are ordered so that most of the information of variation between the initial variables is present in the first PCs. PCA is performed by using orthogonal linear transformation such that a set of observations of possibly correlated variables is converted into a set of values of linearly uncorrelated ordered variables. These variables are such that each one has the highest variance possible under the constraint that it is orthogonal to the preceding ones. Further information on the PCA tool is detailed in [140].

With the change and reduction of variables, the PCA allows a graphical representation of the variation of the data of the DoE (table 5.6). Beyond the graphical representation, through the reduction of the variables, the PCA makes it possible to compress the data, reducing the number of dimensions without losing too much information. This technique is used in many areas including signal processing. Historically, the PCA was introduced by Pearson in 1901, and subsequently developed by Hotelling in 1933. Like many multi-dimensional methods, it was not widely used until the advent of the digital area, but is now well anchored in statistical analysis.

5.4.2.2/ Dimensions

From the figure 5.3, the variables are particularly well correlated because they are collected almost at the same location in the case where the dimension is equal to 20, 30 or 50. This means that the variation of the dimension (between 20 and 50) has little effect on the choice of the levels of the parameters. When the dimension is equal to 10, the value of A is substantially greater than in the other three cases.



(a) Individuals (b) Variables

Figure 5.3: PCA in case 1 - DoEs with fixed dimensions

5.4.2.3/ Functions

From figure 5.4, the variables are much less correlated. This means that the choice of function clearly influences the effects of the factors. Functions can be assigned to group. the most obvious one is the one made of functions 13, 14 and 15. These functions will share the same optimal setting. There are 3 other strong groups of 2 functions 3 - 12, 6 - 7 and 5 - 8. The other ones are not so far from each other and form a large group.

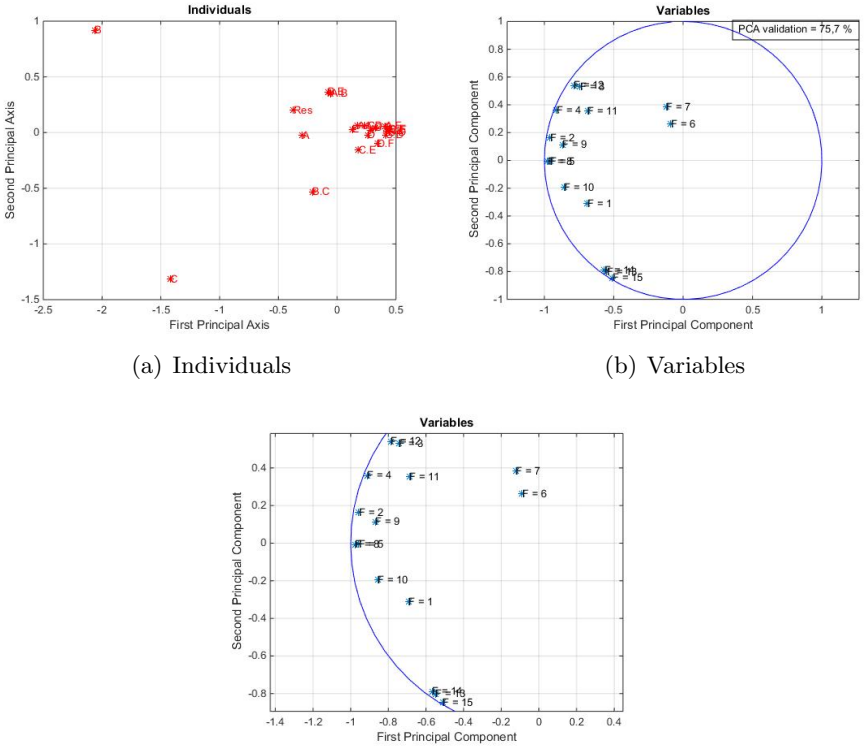
The functions of the benchmark are grouped into categories which have been settled by the CEC [124]. The repartition of the functions into thoses categories is supposed relevant to reflect the variety of optimization problems. It was assumed that, in PCA, functions will be grouped according to thoses categories. However, it is absolutely not the case. Indeed, they are based on few empirical criteria rather than some metrics caracterizing the properties of a function. [41] has investigated that CEC and BBOB benchmark functions properties. This article concludes that, when considering their properties, used functions are not so different one from another.

5.4.2.4/ MaxFEs

From figure 5.5, the variables are not correlated. Yet, there is a relationship between the variables and MaxFEs as they are aligned along the PCA curve.

5.4.2.5/ Designs of experiments exploitation

From the exploitation of other DoEs, optimal settings for every sub-score of the benchmark can be deduced. Those optimal settings are presented in table 5.7. For most of the subscores the optimal setting is the same as the one of the global score. However, it is not the

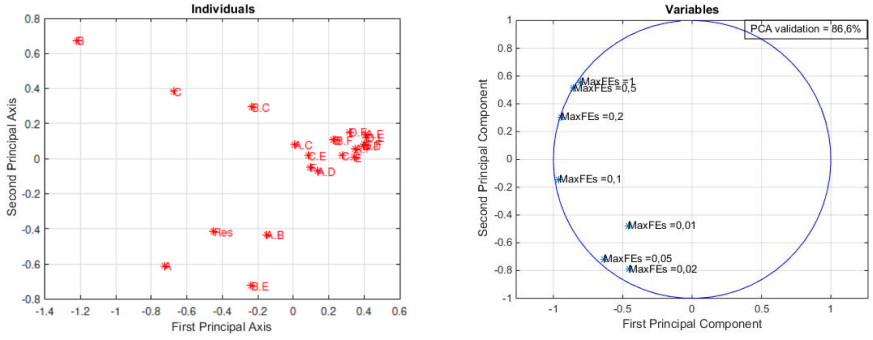


(a) Individuals

(b) Variables

(c) Variables Zoom

Figure 5.4: PCA in case 2 - DoEs with fixed functions



(a) Individuals

(b) Variables

Figure 5.5: PCA in case 3 - DoEs with fixed MaxFEs

case for some functions sub-scores. For these ones, the changes of parameter values seem to encourage global search. This makes sense as most of them are multi-modal functions.

5.4.3/ Meta-optimization approach review

To conclude on the proposed MO approach efficiency, its main setting has been compared with settings from other MO approaches. They have been compared through a benchmark. Approaches and corresponding settings are presented in table 5.8.

The setting of the expert based approach has been obtained through experience

Table 5.7: Optimal settings for the benchmark's sub-scores

Score	N	c_1	c_2	w_{min}	w_{max}	V_{factor}
Global	20	0.5	1.5	0.2	0.7	0.3
Alpha / Omega				-		
All Dimensions				-		
All MaxFEs				-		
F(3,4,8,10,11,12,13,14,15)				-		
F(5,6,9)	-	-	-	0.4	-	-
F1	-	1.5	-	0.4	-	0.05
F2	-	-	0.5	-	-	-
F7	-	-	0.5	0.4	-	-

Table 5.8: Reviewed meta-optimization approaches and associated settings

Approach	N	c_1	c_2	w_{min}	w_{max}	V_{factor}
DoE on Benchmark	20	0.5	1.5	0.2	0.7	0.3
Expert based	20	1	1	0.4	0.9	0.1
DoE on Function (Khosla) [61]	100	2	2	0.2	0.9	0.1
DoE on OFTS (Wang) [62]	25	2	1	0.1	0.3	0.1
DoE on RCTS (Das) [63]	50	2	2	0.9	1.2	0.1

and empirical observation made by expert in optimization of My-OCCS. Khosla's setting [61] have been obtained through the Taguchi method using Rosenbrock and Griewank functions in dimension 30. Wang's setting [62] have been obtained through the Taguchi method using four mathematical functions, such as the Sphere and Ackley ones, in five dimensions from 20 to 100. Das's setting [63] have been obtained through the Taguchi method using a single optimization real case in dimension 4.

Figure 5.6 shows that the proposed approach outperformed all other tested approaches on benchmark global score. The difference between the tested approach and the expert-based one is less than 5%, meaning that the tested approach could not improve much the setting found by an expert on a well studied algorithm for which good settings are known.

From figure 5.7, it can be seen that the proposed approach is better than any other one in terms of quality and reliability. The only exception being the value quality for which the expert-based approach obtains better results by 5% percent.

Figure 5.8 represents the efficiency with regards of dimension. The proposed approach has better results than any other tested approaches for every studied dimension. The efficiency variation should also be considered. For Das' approach [63], the efficiency increases from dimension 30 to 50 which is unusual. Meaning that Das' approach has found a setting efficient in high dimension. Except from this results, all tested approaches obtain an usual decreasing slope. The proposed approach, equal with expert-based one, obtains the softest slope. Therefore, the proposed approach should obtain good results even in higher dimensions.

Figure 5.9 presents the efficiencies of approaches for functions of the benchmark. The proposed approach outperforms every other ones for every functions except for f_1

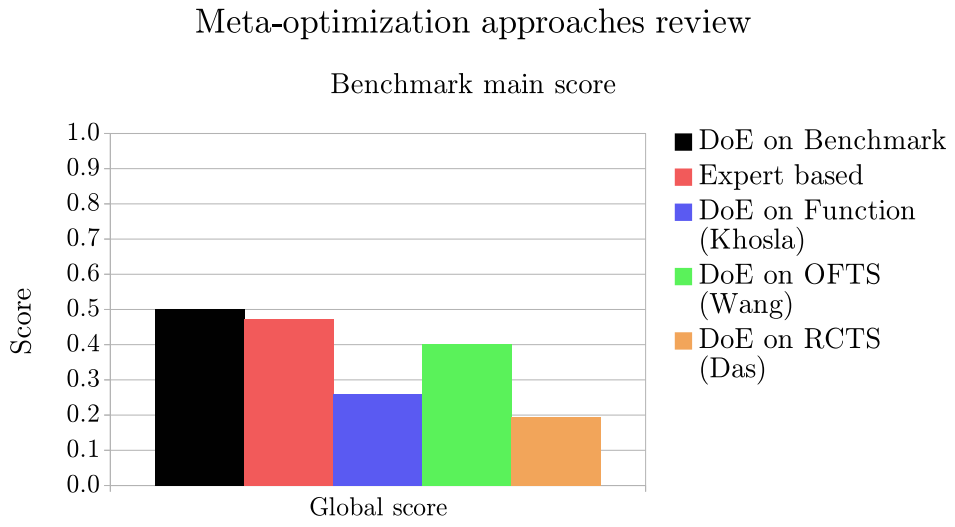


Figure 5.6: Settings review: global score

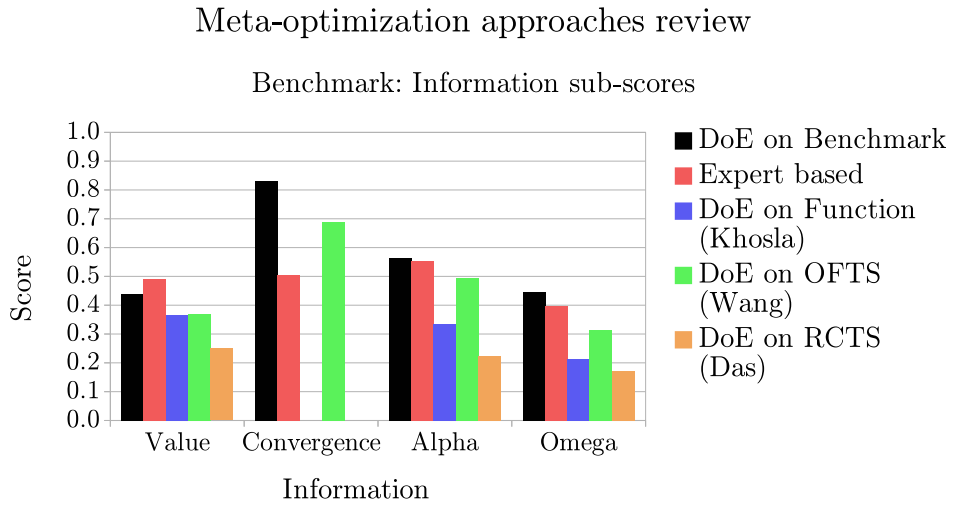


Figure 5.7: Settings review: information sub-scores

and f_{15}. When considering the no free lunch theorem [111], it would have been highly unlikely that one approach outperforms every other one on each functions. The proposed approach is outperformed by 23% on f_1 and by less than 1% on f_{15}. For other functions, the proposed approach surpasses the second better approach by 1% to 14% and the worst one by 13% to 69%. From those remarks, we can state that the proposed approach outruns significantly other approaches for functions f_2 to f_{14} and is significantly outperformed on function f_1.

Figure 5.10 represents the efficiency with regards of *MaxFEs*. It can be seen that the proposed method has the best efficiency for every *MaxFEs*. All approaches display an increasing variation with different rates. As the curves do not cross it is possible to rank them. The second approach, the expert-based one, is outperformed by 1% to 5%. The third one, Wang's approach, is surpassed by 1% to 10%. The fourth and fifth approaches

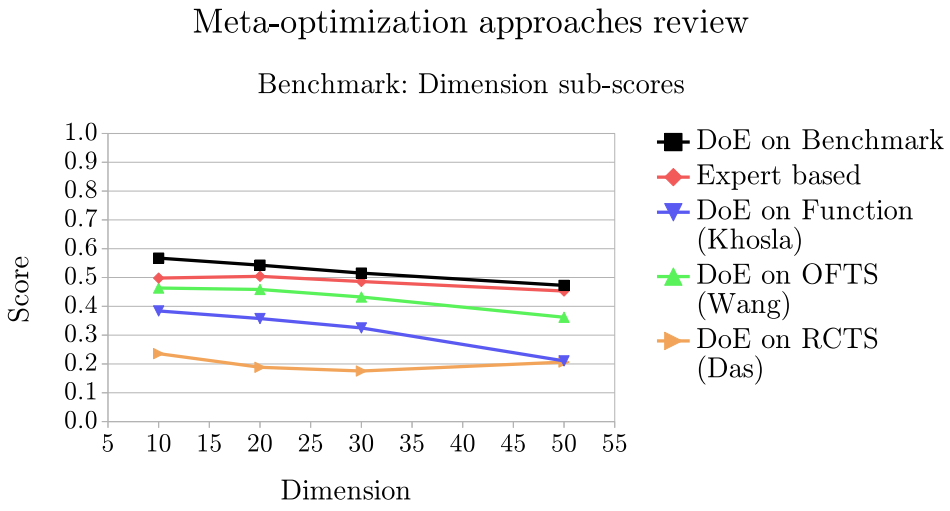


Figure 5.8: Settings review: dimensions sub-scores

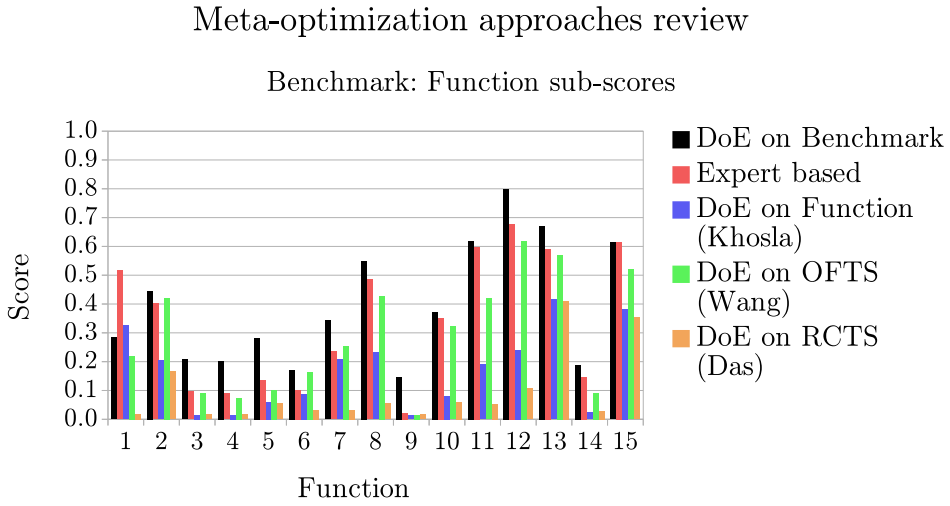


Figure 5.9: Settings review: functions sub-scores

are exceeded by at least 11%. It can be concluded that that the approaches of Khosla, Wang and Das are significantly outperformed while the expert-based one is a competitor.

Finally, from the presented results it can be concluded that, with the chosen benchmark, the proposed MO approach which is a DoE on a benchmark has outperformed state-of-the-art MO approaches and that only expert-based MO has a chance to be competitive.

5.5/ Conclusion

In EDO, the efficiency is one of the major areas of improvement. Improving algorithm efficiency by tuning its parameters is an approach which is not investigated enough. Tuning

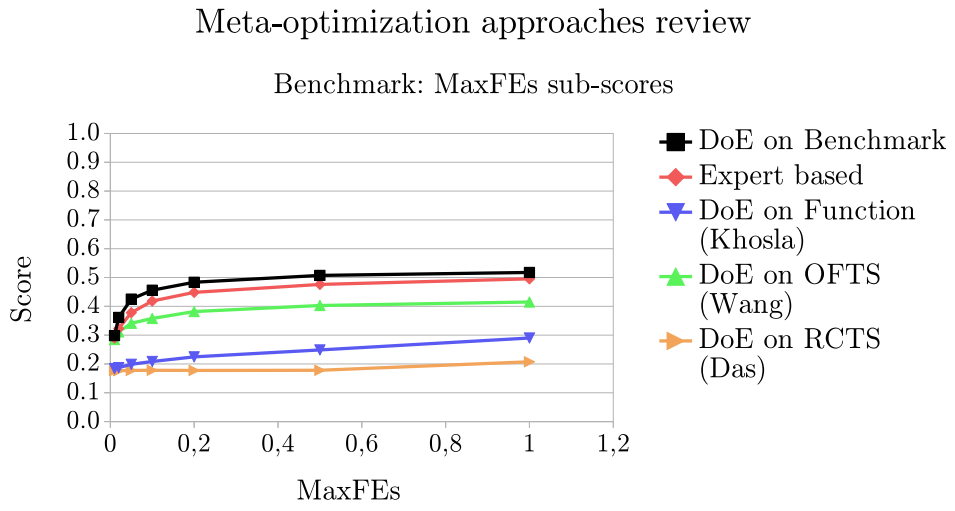


Figure 5.10: Settings review: MaxFEs sub-scores

parameters have always been a complex topic, especially as the efficiency of a setting is strongly dependent to the problem. MO is an interesting method to tune parameters. Currently, MO is mainly performed by DoE on TS. As it is advised to evaluate algorithm thanks to benchmarks and not TS, it has been proposed to perform MO by DoE on a benchmark. Two goals are sought: obtaining an efficient general setting and obtaining a collection of alternative settings to be able to set parameters for a specific problem. The PSO algorithm has been chosen to test this approach as it is efficient and commonly used.

The proposed MO approach is composed of a benchmark tool and a DoE method. The chosen benchmark evaluates tested settings according to industry expectations towards optimization: the efficiency. In addition, it delivers a set of sub-scores, which is necessary in order fulfill the second goal of this article. Concerning the DoE it needs for every scores to be paired with a mean and a *SNR* of used cases results. The DoE objective is to determine the setting maximizing a benchmark score. A fractional factorial DoE has been chosen to gather as much information as possible by doing the minimum number of tests. The chosen DoE uses Taguchi method. The control factors are the parameters of the algorithm while noise factors are elements generating optimization cases: dimension, function and MaxFEs. To investigate the control factors, a L_{27} orthogonal array of 27 tests is used.

An optimal setting maximizing the global score of the benchmark was obtained. Three additional sets of DoEs were realized to find optimal settings for every benchmark scores. For most of the sub-scores, the optimal setting is the same as for the global score. PCA were conducted on the results of those sets of DoE. PCA allows a graphical representation of sets of data variation of DoEs. From PCA several conclusions can be made. The dimension variation, between 20 and 50, has little effect on the choice of the values of the parameters. Different settings should be used for the different groups of the identified functions. CEC functions classification is not consistent for the selection of the values of the parameters. There is a relationship between *MaxFEs* and the optimal setting. The main setting of the proposed MO approach has been compared with settings from other MO approaches. They represent different current state-of-the-art MO approaches which have been tested with the benchmark used in this article. The proposed approach

outperforms all other tested approaches on the benchmark global score. Apart from few exceptions, it also obtains significantly better results for sub-scores. Only expert-based MO is truly competitive.

In a theoretical context, the interest of the proposed MO approach, a DoE on a benchmark, is validated. This approach has provided a significantly better than existing setting. In addition, this approach has yielded several alternative settings to adapt to different problems. To expand from this study theoretical context, it could be interesting to use a real-case benchmark. PCA results indicate that, for this case, not all benchmark optimization cases had an impact on the choice of the values of the parameters. By narrowing the benchmark onto useful cases, MO through the use of an algorithm instead of the DoE might be considered.

6

Sparing fiabilist optimization

Contents

6.1/ Introduction

As explained in the introduction, the SMART-INN partners are looking forward to reducing their production costs due to the manufacturing uncertainties and, consequently, due to the number of flaws. Therefore, a determinist optimization, not taking the uncertainties into account, is not sufficient. A fiabilist optimization method, taking the uncertainties into account, should be developed. Moreover, one of the challenges identified by [8] is the lack of optimization engine [9] taking uncertainties into account. This article has also stated that it is particularly true for the industrial companies dealing with a wide range of complex designs, which is the case of this thesis. However, taking uncertainties into account, which usually increases the running time [141], is not sufficient. Indeed, the majority of the companies tries to achieve optimized design as a balance between cost, quality and time [8]. Therefore, an optimization solution considering the uncertainties without degrading the resolution time nor the solution quality should be developed.

Each of those three aspects of optimization (quality [142], time [141] and cost [143]) is a topic of research by itself. Quality, being how satisfaying the solution is, is linked to the resolution method robustness. Robust optimization methods have been and are still widely investigated, as the number of new algorithms published proves [13]. Time corresponds to the time to run an optimization problem. The running time, though it could be reduced by techniques such as meta-model [144], is often measured in function evaluations [19]. The cost, among other things, depends on how the uncertainties are taken into account. Indeed, a solution sensitive to the uncertainties will lead to high failure rates during manufacturing, increasing production costs. This topic is addressed by techniques such as the ones tested in [145]. Addressing several of these three topics at a time is not an easy task. It has been done, for instance, by [144], which uses meta-models to perform optimization under uncertainties, in order to improve time and cost. To meet the industrial requirements, an optimization method should be both fast and efficient while considering uncertainties.

How to take the uncertainties into account is a wide topic. First, the uncertainties could have several sources and could impact several elements of the optimization problem [64]. Depending on which kind of uncertainties is considered, the determinist formulation of an optimization problem, in equation 1.1, will change in a different way. In the case of industrial problems faced during this thesis:

- The designers are looking for solutions so that $\mathcal{X}\%$ of the produced products respects the bill of specification. This corresponds to the Reliability-Based Design Optimization (RBDO) [146].

- Only uncertainties on variables will be considered.

This kind of problems, RBDO ones considering only variables, will be refereed as 'fiabilist optimization' (FO) ones. These problems could be formulated as in equation 6.1. This equation is based on the determinist one, equation 1.1. In equation 6.1, Δ is the vector of uncertainties, P is the probability function and $\mathcal{X}$ is a probability defined by the designer.

$$(P_{fia}) \begin{cases} \min_{X \in \mathcal{S}} \in \ F(X, \Delta) \\ P(G(X, \Delta) \leq 0) \geq \mathcal{X}\% \\ P(H(X, \Delta) = 0) \geq \mathcal{X}\% \\ X = (x_1, \cdots, x_D) \end{cases} \tag{6.1}$$

Solving a FO problem requires to use fiabilist evaluations. A fiabilist evaluation [64] could be classified into two categories: statistic evaluations and probabilist evaluations. In the statistic evaluation case, daughter evaluations are used. A daughter evaluation is a sub-evaluation which is determinist, performed in the uncertain area. The uncertain area is the part of the search space, centered around the solution to evaluate, defined by the uncertainties. In the statistic evaluation case, a set of daughter evaluations is done in the uncertain area. The satisfactions (see chapter 3) of daughter evaluations will be aggregated [21] to produce the satisfaction of the statistical evaluation. This method will drastically increase the number of evaluation because for every statistical evaluation, several daughter evaluations should be made. In the probabilist evaluation case, a constraint is added to the existing ones. The failure probability [64] should be inferior to a threshold, defined by the designer. This failure probability could be computed either by a Monte-Carlos method [147] or by an indicator-based method [64] such as Rjanitzyne-Cornell [148] or Hasofer-Lind [149] ones. In Monte-Carlos case, as for statistic evaluations, a set of daughter evaluations should be made. Concerning indicator-based methods, they require to know the distribution laws of the uncertainties which is not always the case. The fiabilist optimization methods, using fiabilist evaluations, increases drastically the number of evaluations.

To perform a fiabilist optimization without increasing radically the number of evaluations, a solution could be to use both fiabilist and determinist evaluations. As explained in the introduction, to solve an engineering design optimization (EDO) problem [8], it has been chosen to use population-based stochastic global meta-heuristic. They use a part of the solution for global search mechanisms. Therefore, a sparing fiabilist optimization (SFO) method, using fiabilist evaluations only for solutions used in global search mechanisms, is proposed. This method will be designed and tested with the particule swarm optimization (PSO) [54] algorithm, as this algorithm is both efficient (see chapter 1) and easy to modify.

The SFO method proposed in this chapter uses statistical evaluations which will be detailed in section 6.2. The SFO method, which uses both statistical and deterministic evaluations, will be explained in section 6.3. How the proposed SFO has been tested will be presented in section 6.4. Finally, the results will be presented and discussed in section 6.5.

6.2/ Statistical evaluation

This section intends to detail how the statistical evaluations used by the SFO method are performed. The statistical evaluation process is described in sub-section 6.2.1. In this process, daughter evaluations are drawn as explained in sub-section 6.2.2. The daughter evaluations sample size is fixed as explained in sub-section 6.2.3. Finally, how the satisfaction of the statistical evaluation is computed will be presented in sub-section 6.2.4.

6.2.1/ Statistical evaluation overview

The way how a statistical evaluation works is summarized by figure 6.1. A set of daughter evaluations are performed into the uncertain area. As a reminder of section 7.1, a daughter evaluation is a sub-evaluation which is deterministic. These daughters evaluations will produce deterministic satisfactions which will be aggregated by a $\mathcal{X}\%$ confidence limit to produce a fiabilist satisfaction. This fiabilist satisfaction represents the minimal satisfaction obtained by $\mathcal{X}\%$ of the solutions inside the uncertain area. The satisfaction of the statistical evaluation will be this fiabilist satisfaction.

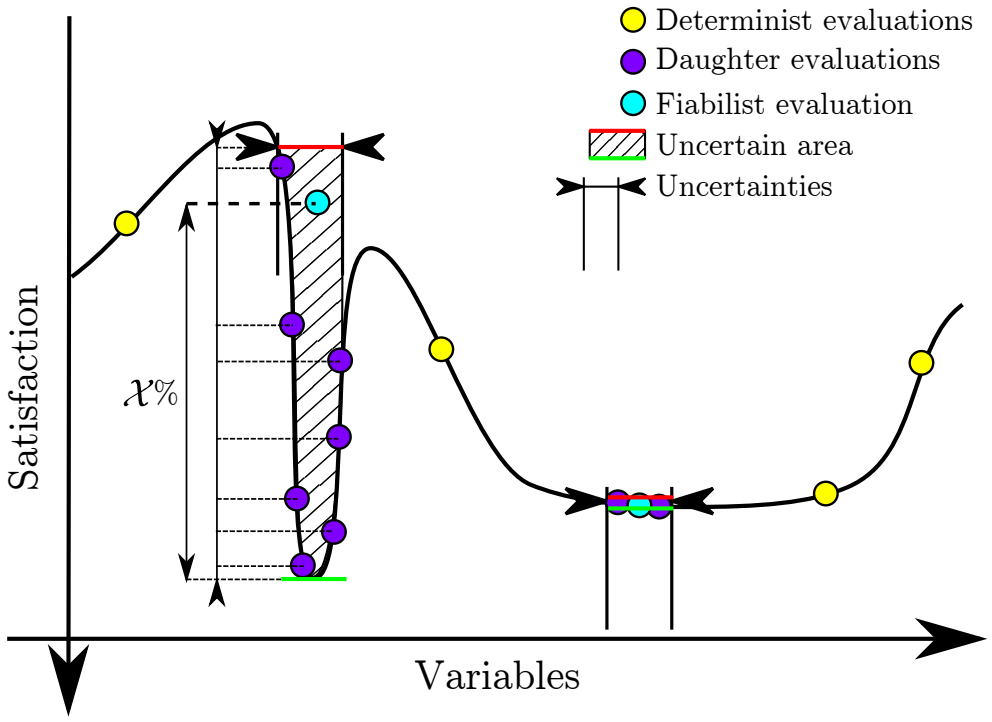


Figure 6.1: The fiabilist evaluation

6.2.2/ Daughter evaluations drawn

The uncertain area, denoted A, in which daughter evaluations are made, is defined by equation 6.2. In this equation, x_i is the the i-th variable of the statistical evaluation. Δ_i is the uncertainty on the i-th variable. The values of the Δ vector are defined by the designer during the formulation phase (see chapter 1). Finally, A_i is the uncertainty range for a given x_i.

$$\forall i, A_i = [x_i - \Delta_i, x_i + \Delta_i] \tag{6.2}$$

The coordinates of the daughter evaluations are computed according to equation 6.3. In this equation, x'_i (respectively x_i) is the i-th coordinate of the daughter (respectively statistical) evaluation. The random uncertainty vector, δ, is randomly drawn so that the coordinates of the daughter evaluation will be inside the uncertain area.

$$x'_i = x_i + \delta_i \quad with \quad \delta_i \in [-\Delta_i, \Delta_i] \tag{6.3}$$

The random selection of daughter solutions is realized according to a normal distribution law as it is one of the most commonly used laws in RBDO [64]. The normal distribution law has been set as in equation 6.4.

$$\delta_i \sim \mathcal{N}(0, \sigma) \qquad with \qquad \sigma = \frac{\Delta_i}{2.5} \tag{6.4}$$

In this equation, σ has been chosen so that 99% of the daughter solutions will be inside the uncertain area $[x_i - \Delta_i, x_i + \Delta_i]$. Indeed, the cumulative distribution function of the standard normal is equal to 99% for $\sigma = 2.5$ [150]. How many daughter evaluations are used is explained in sub-section 6.2.3.

6.2.3/ Sampling size

In order for the statistical evaluation to make sense, the number of daughter evaluations must be sufficient for the statistical analysis to be reliable. To determine the sampling size, the World Health Organization (WHO) recommendations [151] have been followed. In a confidence limit case, it leads to equation 6.5.

$$n = \frac{t^2 p(1-p)}{e^2} \tag{6.5}$$

Equation 6.5 notations are given in table 6.1.

Table 6.1: Sampling size equation notation

Notation	Description	Comment	Typical value
n	Sampling size	What is computed	20
t	Margin coefficient deduced from 'c'	Cf. [150]'s inverse cumulative distribution function	0
c	Confidence limit	Defined by the designer	50%
p	Supposed proportion of elements validating the test	Is equal to s in our case	50%
e	Relative error margin on the measure	The allowed error is defined by the optimizer	5%

A few comments about equation 6.5 are made here:

- First, it is considered that $p = c$: The sample is used to compute a $\mathcal{X}$% confidence limit, c. This means that $\mathcal{X}$% of the sample will have a better satisfaction than the statistical evaluation one. Thus, the percentage of the sample which succeeds in having a better satisfaction than the satisfaction of the statistic evaluation is $\mathcal{X}$%. Hence, by extension the supposed proportion of elements validating the test, p, is equal to $\mathcal{X}$%. Therefore, $p = \mathcal{X}\% = c$.
- By default, $e = 5\%$: The lower e is set the more accurate the confidence limit will be. For this reason, only values of e inferior to 10% are considered. On the opposite, the higher e is set the lower the population will be and the lower the impact on

resolution will be. Figure 4.11 shows that if $MaxFEs \leq 0.1$, the efficiency of the algorithms drops. For an algorithm using a population of $N = 100$ individual, if the sample size is $n = 1000$, at each iteration, the number of evaluations made will be at least multiplied by 10. This is equivalent to having $MaxFEs = 0.1$. Thus, with $MaxFEs \leq 1$, n should remain inferior to 1000 for the algorithm to stay efficient. Figure 6.2 shows that, with $e \leq 2\%$, $n \leq 1000$. So, the default value of e should be superior to 2%. On the opposite, with $e = 10\%$, n drops to 1 which is not consistent with the concept of the statistical evaluations. Thus, e should be strictly inferior to 10%.

Figure 6.2 shows sample size (n) with respect to the confidence limit (c) for different relative error margins (e). The sample size varies from a few samples to nearly a million. It can be observed that the lower the value of e is the smaller the sample is. With $e = 5\%$, the default value, $n \in [4, 116]$.

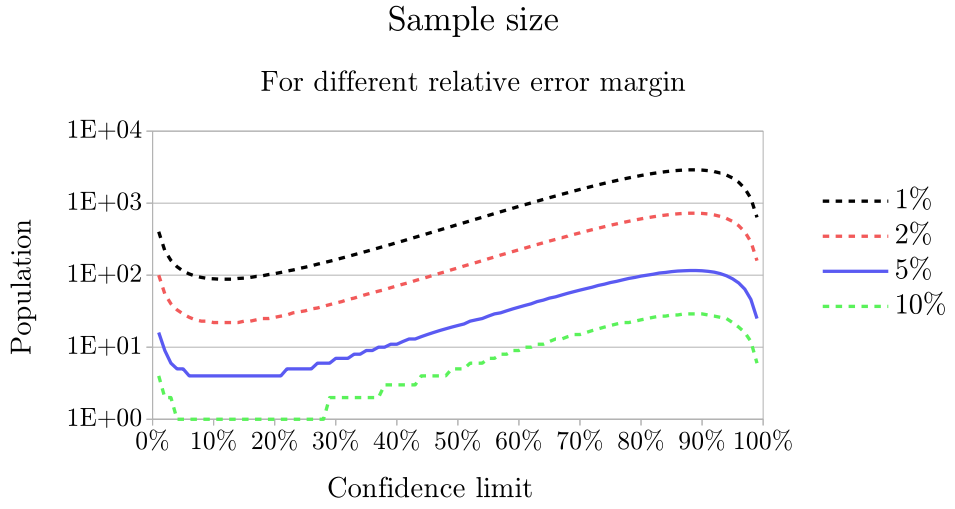


Figure 6.2: Sample size with respect to the confidence limit for different relative error margins

6.2.4/ Satisfaction computation of statistical evaluations

To compute the satisfaction of the statistical evaluation, noted s, a $\mathcal{X}\%$ confidence limit, given in equation 6.6, is used. This confidence limit is computed from the set of the daughters evaluations satisfactions, noted s^{Δ}. In this equation, the satisfaction of a single daughter evaluation is noted s^{δ}, μ is the mean function and σ is the standard deviation function. The confidence limit percentage, $\mathcal{X}$, is meant to be defined by the designer during the formulation phase. This equation relies on the hypothesis that the satisfaction of the daughter evaluation are following a Gaussian distribution law.

$$\begin{cases} s = \mu(s^{\Delta}) - \alpha \cdot \sigma(s^{\Delta}) \\ \text{with } \alpha \text{ such as } P(s^{\delta} \in [\mu(s^{\Delta}) - \alpha \cdot \sigma(s^{\Delta}), +\infty]) = \mathcal{X}\% \end{cases} \tag{6.6}$$

6.3/ Sparing fiabilist optimization method

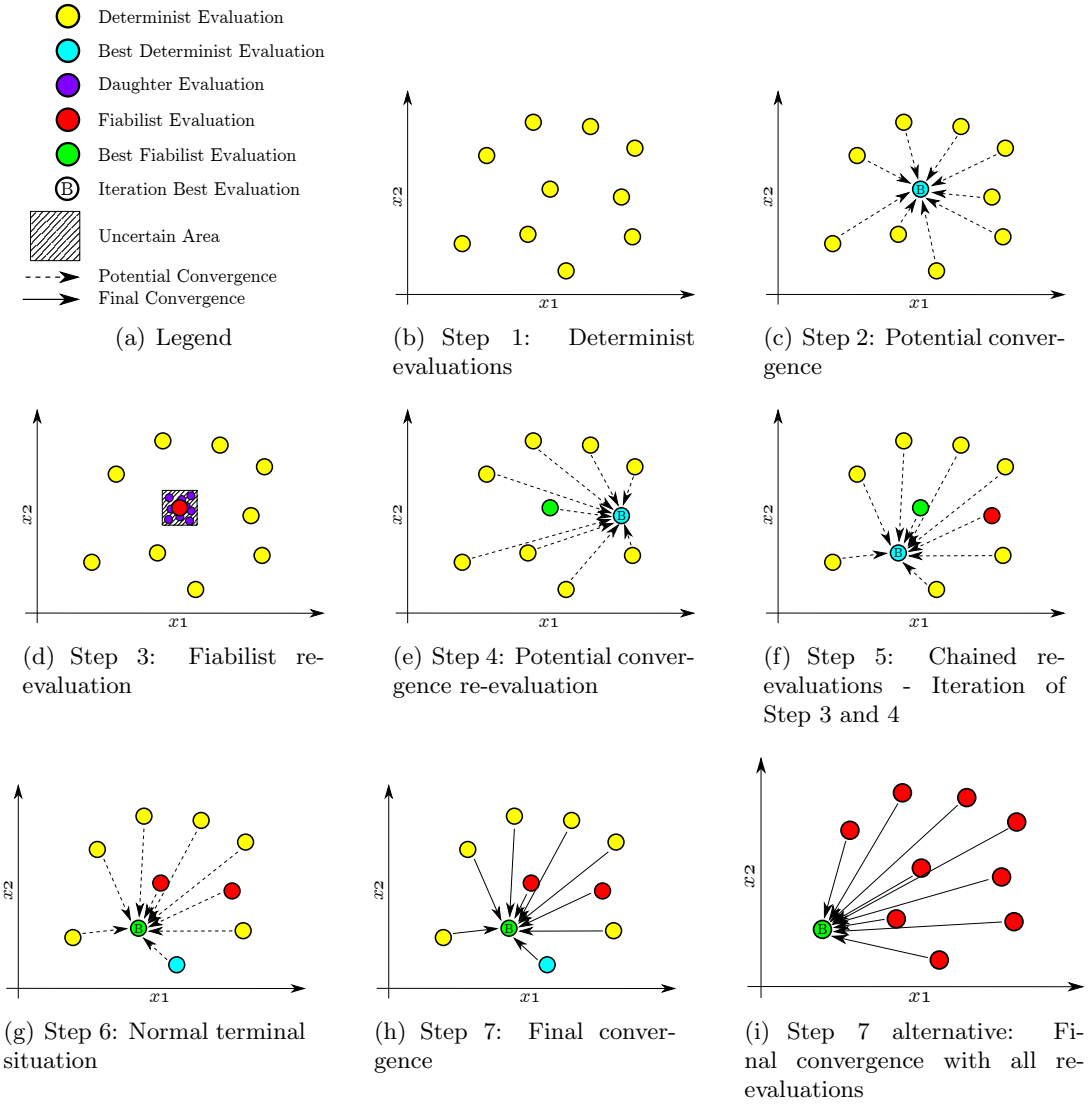
This section presents the SFO method through its application to PSO. For other algorithms this method should be adapted according to the remarks given at the end of this section. The method is explained by algorithm 2 which relies on figure 6.3. This figure displays a population of points in a two dimensional search space. In the SFO method, the evaluations which are considered the best ones are the ones having the highest satisfactions.

Algorithm 2 Sparing Fiabilist optimization method pseudo code - In PSO case

1: All population points are evaluated in a deterministic way (figure 6.3(b))
2: The best determinist evaluation is identified. At this point, this evaluation is also the iteration best evaluation. In a determinist resolution, the population is supposed to converge to this point. This defines the potential convergence (figure 6.3(c)).
3: The best determinist evaluation is re-evaluated in a fiabilistic way (figure 6.3(d)). The determinist satisfaction of this point is replaced by its statistical satisfaction. This evaluation is now a fiabilist one.
4: The potential convergence is re-evaluated (figure 6.3(e)): First, the best fiabilist evaluation and the best determinist evaluation are identified. Then, the iteration best evaluation, which could be determinist or fiabilist, is defined. Finally, the potential convergence, which heads towards the iteration best evaluation, is determined.
5: **while** Iteration best evaluation is not the Best fiabilist solution (figure 6.3(g)) **do**
6: The best determinist solution is re-evaluated in a fiabilistic way and then the potential convergence is re-evaluated (figure 6.3(f)). As statistical evaluation is likely to degrade the satisfaction of the solution, the iteration best evaluation has a great chance of being a determinist solution. Therefore, in order for the iteration best solution to the fiabilist best solution, this step will probably be iterated several times.
7: **end while**
8: The final convergence is set and the algorithm could proceed in converging the population towards the iteration best solution, which is a fiabilist one (figure 6.3(h)). It might happen that during the iterative part of this method, all points are re-evaluated. In this case, the final convergence will happen with all points evaluated in a fiabilistic way (figure 6.3(i)).

A few remarks could be made about the SFO method:

- To reduce the computation time, the percentage of fiabilist evaluations should be the lowest possible. The higher the population the higher the chance of having a small percentage of fiabilist evaluations. Therefore, the population should be large enough, typically more than 50 individuals.
- Step 3 consists in re-evaluating the best solution of the current iteration. For the PSO chosen as an example [54], only one solution is used for the global convergence. In the case of an algorithm using several solutions for global convergence, for instance a 4-cluster PSO [152] or a CMAES [52], the method should be adapted. In this case, all solutions used for the global convergence should be re-evaluated. In this case, it will be even more important to consider a large population.
- Steps 5 consists in chained re-evaluations. At this step, exactly as for step 3, it might happen that several solutions should be re-evaluated. The chain of re-evaluation will



(a) Legend

(b) Step 1: Determinist evaluations

(c) Step 2: Potential convergence

(d) Step 3: Fiabilist re-evaluation

(e) Step 4: Potential convergence re-evaluation

(f) Step 5: Chained re-evaluations - Iteration of Step 3 and 4

(g) Step 6: Normal terminal situation

(h) Step 7: Final convergence

(i) Step 7 alternative: Final convergence with all re-evaluations

Figure 6.3: The SFO method scheme - Pictured in the PSO algorithm case

then stop only if all solution used for global research mechanism have been evaluated in a fiabilist way. Hence, at step 6, all solutions used for global convergence are evaluated in a fiabilistic way.

6.4/ Sparing fiabilist optimization test method

This section will present how the proposed method has been tested. As mentioned the introduction of chapter 4, in order to avoid false conclusions, algorithms should be tested on benchmarks [13]. Moreover, there are two benchmarks of reference, the CEC [112] and the GECCO [113]. There is no version of the CEC benchmark with uncertainties. There is a Noisy version of the CEC benchmark [153]. However, in this version the uncertainties apply to the objective function and not the variables. So, neither the CEC nor the BBOB match our requirements. Therefore, the SFO method will be tested on a fiabilist version of the benchmark developed in chapter 4.

This fiabilist version is presented in sub-section 6.4.1. How the method has been tested thanks to this benchmark is described in sub-section 6.4.2.

6.4.1/ Fiabilist version of the benchmark

This sub-section presents how the benchmark detailed in chapter 4 has been modified to test fiabilist methods. Some modifications have been made to lower the difficulty of the problems in order to compensate for the addition of uncertainties and others have been made to avoid technical difficulties. The modifications are the following:

- Uncertainties on variables have been added: They have been added according the formulation of equation 6.1. For the industrial problems faced in this thesis, the order of magnitude of the uncertainties relative to the search space is about 1%. The search space of the problems composing the benchmark is $\mathcal{S} = [-100, 100]^D$. Therefore, for the uncertainties to have the same order of magnitude, relative to the search space, it has been chosen so that the range of the uncertainties will be: $\Delta = [-1, 1]^D$.

- The Shift vector (see section A.5), noted s, is now randomly drawn in $[-10, 10]^D$ instead of $[-100, 100]^D$: It might happen that the global fiabilist optimum corresponds to the global determinist optimum or may be close to it. It could happen, for instance, in the case of a function having a low dispersion [41]. In this case, the landscape of the function could be a funnel which is a global basin shape composed of clustered local optima [154]. Therefore, to avoid the inconvenience of having the global fiabilist optimum close to the search space edges, it has been chosen to narrow the shift vector to $s = [-10, 10]^D$.

- The set of dimensions used by the benchmark have been changed to $\{10, 20, 30\}$ instead of $\{10, 20, 30, 50\}$: $D50$ has been removed to face the increasing difficulty of FO problems.

- On the opposite, the set of functions remains unchanged for the method to be tested with functions having different sensitivities to uncertainties: Some functions having high ruggedness [155] or deep-valleys [8, 41], such as 'Katsuura' (equation A.12), will be sensitive to uncertainties. On the opposite, other functions having smooth landscapes with low-ruggedness [155], such as 'Bent Cigar' (equation A.8), will be less sensitive to uncertainties.

- The set of $MaxFEs$ used by the benchmark have been changed to $\{1, 2, 5, 10\}$: It is considered that $MaxFEs = 1$ is sufficient to solve deterministic problems [124]. However, to face the increasing difficulty of fiabilist problems, the running time and by extension the allowed number of evaluations (FEs) might be increased [141]. Thus, $MaxFEs \geq 1$ will be used. The SFO is meant to perform fiabilist optimization without increasing too much the running time. As mentioned in the introduction, in order to match the rapidity challenge of this thesis, it is wished to reduce the optimization process delay to a few days. With $MaxFEs = 1$, the typical running time of the problems faced in this thesis is 8 hours. With $MaxFEs = 10$, the running time should last about 3 days, which is more or less the time limit to respect. Thus $MaxFEs$ should remain lower or equal to 10.

- In *MaxFEs* aggregation ($\mathcal{A}_M$), the raw weights of *MaxFEs* (W) have been changed. $W_m = 1/MaxFEs$ and thus $W = \{1, 0.5, 0.2, 0.1\}$: $MaxFEs = 1$ is supposed to be sufficient to solve determinist problems [124]. As no other reference for fiabilist problems exists, it will be chosen for these problems as well. Therefore, if $MaxFEs \geq 1$, it would be considered that the algorithm is given 'additional' time to solve the problem. The higher the *MaxFEs*, the lower the designer's satisfaction because of the increasing running time. Thus, for $MaxFEs \geq 1$, the higher $MaxFEs$, the lower it should be weighted. In optimization, time could be measured in function evaluations (*FEs*) [19], which are proportional to *MaxFEs* (see equation 4.1). As in chapter 4, it would be considered that designer's satisfaction toward a run of optimization is inversely proportional to the time required to perform it. Therefore, an inverse law is used to weight *MaxFEs*.

6.4.2/ Test of the sparing fiabilist optimization

This section will explain how the SFO method has been tested. First, the resolution method will be presented. Then the conducted tests will be detailed.

To test the SFO method, the normalized evaluations approach, developed in chapter 3, has been used. The PSO [54] algorithm has been used as it is the algorithm chosen to develop method during this thesis. The chosen setting to test the method has been obtained by one of the additional meta-optimization realized in chapter 7. This setting, given in table 6.2, has been chosen for three reasons:

- A high N value: The higher the population, the lower the ratio of fiabilist evaluations over determinist ones. The lower this ratio, the lower the number of daughter evaluations made. The lower the number of daughter evaluations, the smallest the impact on resolution.

- A low V_{factor} value: Adding uncertainties to a problem will harden the task of finding the global optimum. Thus, the search space should be explored more carefully. This would be done by encouraging exploration over exploitation [44]. A low V_{factor} will avoid premature convergence [156] and encourage exploration by reducing convergence.

- This setting is the only one obtained through a meta-optimization using the NE approach (see chapter 3): A setting obtained by meta-optimization is optimized for the problem which is used as an evaluation tool. Therefore, settings found by a meta-optimization approach that has not used the NE approach might not be suited for the problem considered.

Table 6.2: PSO parameters setting chosen to test the SFO method

Parameters	N	c_1	c_2	w_{min}	w_{max}	V_{factor}
DoE test problem (NE)	50	0.8	1.2	0.25	0.75	0.1

To assess the benefit of using the SFO method, a comparison must be performed. It is done with two other FO methods using the same statistical evaluation method:

- The 'posterior' FO method [157]: In this case, only determinist evaluations are used to solve the problem. The solution found is re-evaluated in a fiabilist way at the very end of the run.
- The 'prior' FO method [157]: In this case, only fiabilist evaluations are used to solve the problem.

The three tested methods will be referred as 'posterior', 'sparing' and 'prior'. These methods have been chosen as they could be used with the statistical evaluations allowing standard comparisons with the sparing method. Those three methods have been tested with several confidence limits: $\{5\%, 10\%, 20\%, 50\%, 80\%, 90\%, 95\%\}$. These confidence limits have been chosen so that:

- A large part of the spectrum of the possible limits ($[0\%, 100\%]$) is covered: No assumption is made about designers' choice on the confidence limit.
- From the center of the range (50%) to the edges (0% and 100%), confidence limits are chosen along a logarithmic scale: It is assumed that, the closer the confidence limit is to a spectrum edge (0% or 100%), the fastest the the variation of the score of the benchmark will be. Thus, more confidence limits should be tested near the edges of the spectrum (0% and 100%) rather than near its center (50%).

6.5/ Results and discussion

This section will present and discuss results obtained by the fiabilist methods on the uncertain benchmark. First, sub-section 6.5.1 will introduce the raw results and discuss what should be of use from them. Then, sub-section 6.5.2 will discuss the confidence-limit-average of scores of the fiabilist methods. Finally, sub-section 6.5.3 will present the influence of the confidence limit on the scores of the benchmark.

6.5.1/ Raw results

All the scores of the benchmark obtained by the three tested fiabilist methods for each of the seven tested confidence limits are given in table 6.3. Vertically, the table is divided by fiabilist methods and then by confidence limits. Horizontally, the table displays: the global score, information sub-scores and sub-set sub-scores. Details about the computation of scores and their meaning could be found in chapter 4.

Considering the amount of data in this table, it has been decided not to discuss them directly but to identify topics of interest and analyze them. To ease these analysis, it is considered that a score variation inferior or equal to 0.1 is not 'consequent' enough to be of interest and analyzed. Based on this assumption, the score variation of posterior method, with respect of confidence limit, are not consequent. This could be explained as in the posterior method case, the algorithm will converge to a deterministic optimum no matter the confidence limit. In the sparing method case, the scores having a consequent variation, with respect of the confidence limit, are the following ones: convergence, alpha, $D30$, $F1$, $F5$, $F11$, $F12$, $F15$, $M5$ and $M10$. In this case, the information sub-scores (D,F and M) variations could be explained by the convergence sub-score one. In the Prior method case, the scores not having a consequent variation, with respect to the confidence limit, are

Table 6.3: The scores obtained by the tested fiabilist methods (×1000) on the uncertain benchmark, for the different confidence limits.

Scores			Posterior							Sparing							Prior						
Kind	Category	Name	5%	10%	20%	50%	80%	90%	95%	5%	10%	20%	50%	80%	90%	95%	5%	10%	20%	50%	80%	90%	95%
Global	Main score	Score	407	413	413	391	386	376	371	425	427	425	373	306	299	304	466	474	472	397	306	296	299
Infos	Quality	Value	314	315	318	299	290	280	275	417	418	417	412	407	400	396	416	418	418	412	403	396	392
		Convergence	974	975	974	973	974	973	974	549	554	549	321	060	046	066	773	809	799	426	060	041	070
	Reliability	Alpha	449	479	462	427	419	411	403	513	515	513	458	348	338	351	523	528	530	462	340	329	335
		Omega	368	359	370	357	354	342	340	342	343	342	293	266	261	258	411	422	416	334	272	265	266
Dimension		10	429	422	439	395	374	362	355	415	419	415	360	311	306	303	468	477	478	368	305	295	298
		20	410	417	425	395	389	382	377	421	424	421	365	301	295	299	463	472	471	391	301	293	295
		30	398	408	398	388	388	377	372	432	432	432	383	308	299	308	466	474	470	413	309	299	303
Sub-set	Function	1	282	282	282	283	280	277	276	131	131	131	141	100	097	110	258	262	262	221	095	086	098
		2	471	472	478	458	434	403	373	315	326	315	265	215	200	193	325	341	341	278	219	198	192
		3	202	204	204	202	201	199	200	153	148	153	107	051	040	050	194	195	196	147	046	040	048
		4	215	211	215	210	211	212	211	114	111	114	102	049	048	051	203	206	205	151	047	041	050
		5	281	284	283	271	264	260	252	219	228	219	182	116	102	101	282	289	290	216	112	097	100
		6	190	190	191	186	185	185	186	106	103	106	058	021	019	025	169	174	176	096	030	025	032
		7	187	189	189	185	185	188	185	104	095	104	055	018	017	021	167	173	176	097	028	025	031
		8	196	202	201	185	195	195	187	103	092	103	058	021	024	030	189	191	194	119	037	030	038
		9	202	203	202	200	201	200	200	118	125	118	080	027	024	024	142	151	147	088	026	020	020
		10	394	399	416	384	384	349	334	349	361	349	279	202	182	178	376	390	388	300	194	174	171
		11	639	635	631	596	609	571	567	586	589	586	531	423	401	416	629	645	638	565	418	392	399
		12	782	768	790	774	748	756	763	673	686	673	631	555	547	554	709	711	710	625	545	538	540
		13	207	250	213	185	185	185	185	567	560	567	499	417	420	427	639	651	645	532	428	424	431
		14	186	189	185	186	186	186	185	180	186	180	107	050	037	043	204	212	205	135	051	046	051
		15	207	260	243	185	186	185	186	598	584	598	500	408	404	410	629	639	639	549	418	410	414
MaxFEs		1	412	414	413	390	391	374	369	407	412	407	348	289	282	284	455	465	463	368	283	278	275
		2	402	413	410	392	374	379	372	437	432	437	385	307	301	309	472	479	478	418	302	289	299
		5	403	423	422	394	387	378	371	464	464	464	430	350	343	354	488	492	489	455	374	355	368
		10	402	410	420	397	392	376	376	475	479	475	453	388	374	384	493	497	493	470	420	399	411

F6-F9. In this case, global, alpha, omega, dimension and *MaxFEs* scores variation could be explained by value and convergence variation. Therefore, the scores which have to be investigated in sub-section 6.5.3 are: value, convergence and the function for the 'prior' method.

6.5.2/ Average scores obtained by the methods

This sub-section will compare fiabilist methods based on their scores on the benchmark. To ease the comprehension, for each method, the given scores are the average of scores obtained for the different confidence limits.

Fig 6.4 presents the average global score obtained by the fiabilist methods. It could be noted that the posterior method is the most efficient one, but by less than 0.1 point. Indeed, the efficiency of the methods remain between 0.35 and 0.4. Thus, as no method consequently outperforms one another in terms of efficiency, sub-scores should be carefully investigated.

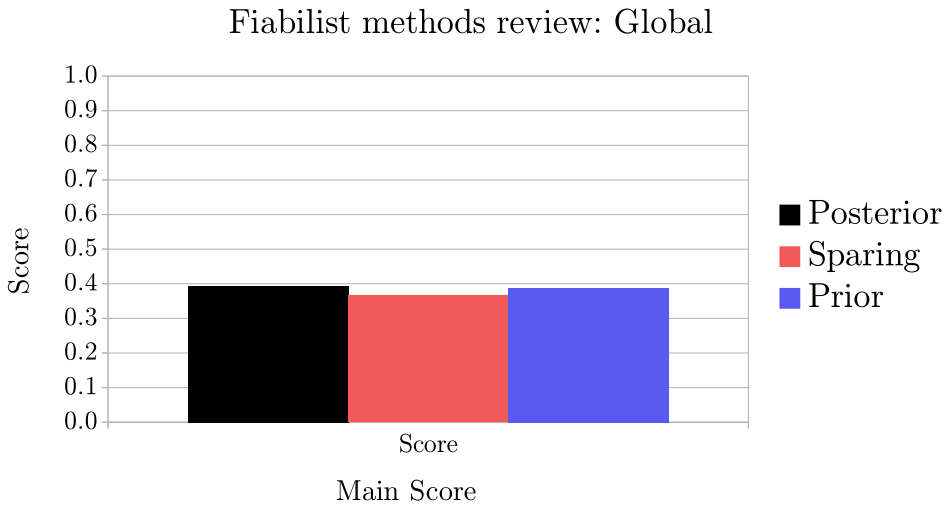


Figure 6.4: Fiabilist methods review: Average global scores

Figure 6.5 presents the average information sub-scores obtained by the fiabilist methods. About the value sub-score, it could be said that the sparing and prior methods, with a value quality of 0.4, are consequently superior to the posterior one which has a value quality of 0.3. The posterior method whose convergence sub-score is superior to 0.9, has a fast convergence. The convergence of the prior method, which is about 0.4, is consequently superior to sparing one whose value is about 0.3. Thus, if a good value is sought, the prior method, or eventually the sparing one, should be chosen. Whereas, if a fast convergence is sought, the posterior method should be preferred. The tested fiabilist methods have the same alpha sub-score, 0.4, but different omega sub-scores, all between 0.3 and 0.4. Thus no method consequently outperforms one another in terms of alpha or omega sub-scores.

Figure 6.6 presents the average dimension sub-scores obtained by the fiabilist methods. For each method, the scores remain steady and are similar to global score ones. The impact of the dimension on the choice of a fiabilist method could be neglected.

Figure 6.7 presents the average function sub-scores obtained by the fiabilist methods.

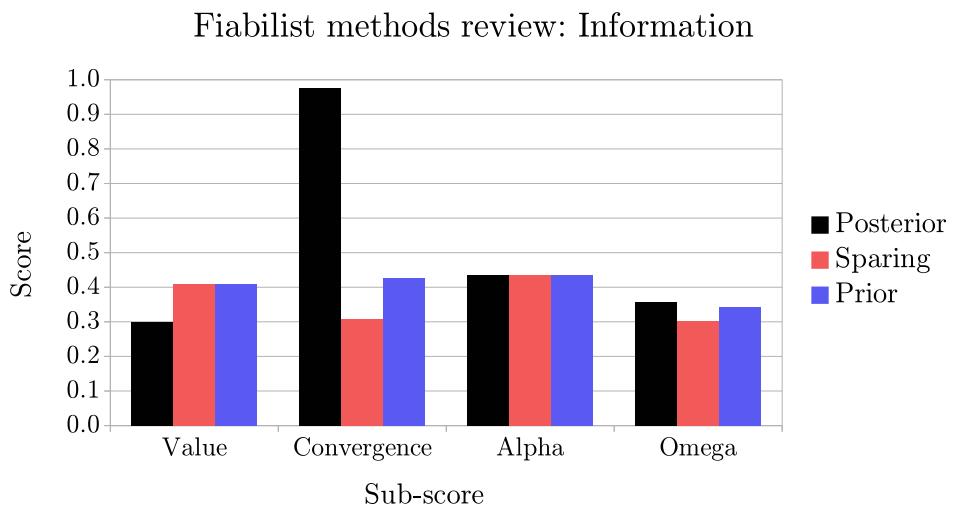


Figure 6.5: Fiabilist methods review: Average information sub-scores

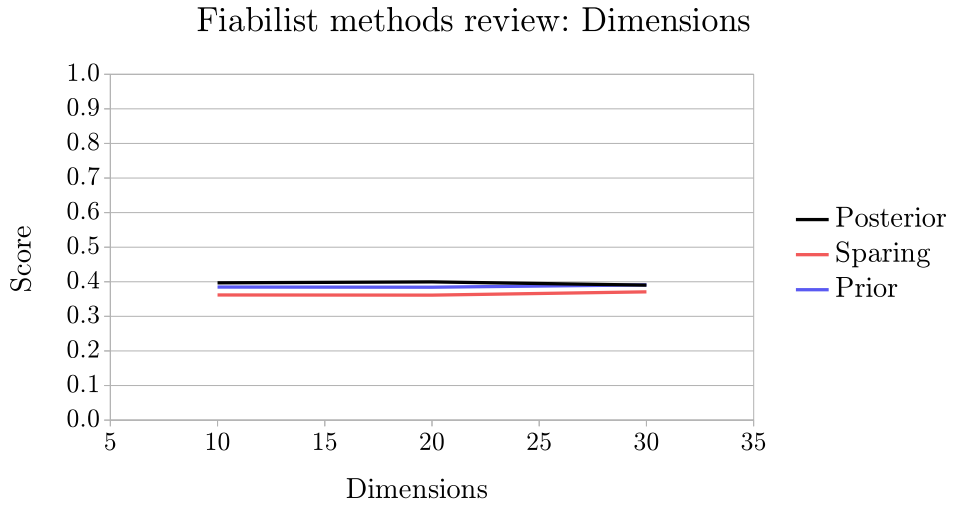


Figure 6.6: Fiabilist methods review: Average dimension sub-scores

For uni-modal functions ($F1$ and $F2$), the posterior method is consequently more efficient than the sparing and prior ones. Indeed, in a uni-modal case as there is only one optimum, a method using only determinist evaluations is advantaged. For multi-modal functions ($F3$-$F9$), the scores of the sparing method are equal to 0.2, or to 0.3 in $F5$ case. The sparing and prior methods obtain scores inferior or equal to 0.2, except for $F5$ in which case they are inferior to 0.2. For these functions, no methods could be considered truly efficient and it could be explained by the local minimum area which is too small compared to the uncertain area. Also, it is with these functions that algorithms tested in chapter 4 obtained the poorest results. Concerning hybrid functions ($F10$-$F12$), the posterior method is about 0.1 above the sparing and prior ones whose results are never different by more than 0.3. On these functions, their scores are going from 0.3 to 0.8 meaning that the methods are efficient. For composite function ($F13$-$F15$), their scores varies from

one function to another. On $F14$, no method obtains a score higher or equal to 0.2. On $F13$ and $F15$, the methods obtain almost exactly the same scores. On these functions, the scores of the sparing and prior methods are about 0.5 while the ones of the posterior method are consequently inferior being equal to 0.2. Based on previous remarks, it could be stated that the efficiency of the tested fiabilist methods strongly depends on the test functions. If efficiency is key, the sparing and the prior methods are to be preferred on composite functions. On the contrary, the posterior method is to be favored on uni-modal and hybrid functions. As for multi-modal ones, no method could be considered truly efficient and none is advised over another.

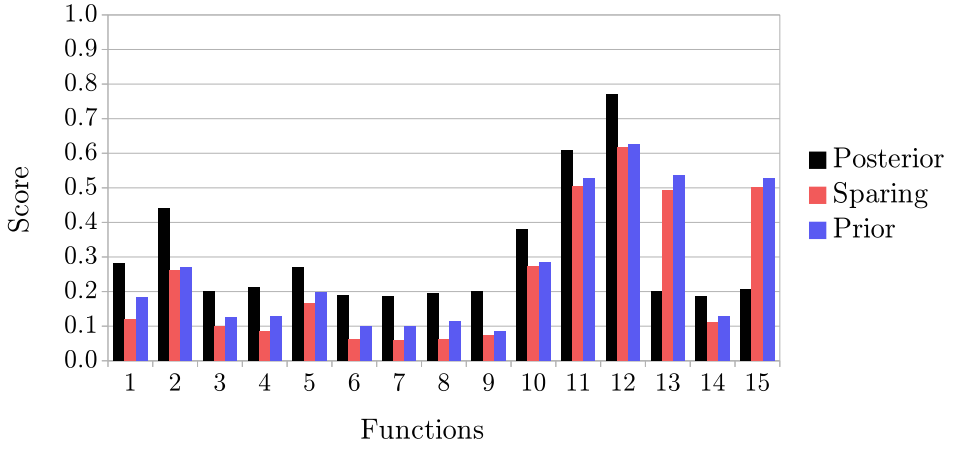


Figure 6.7: Fiabilist methods review: Average function sub-scores

Figure 6.8 presents the average *MaxFEs* sub-scores obtained by the fiabilist methods. The scores of the posterior method remain steady at 0.4 while the ones of the prior and sparing methods slowly increase from 0.35 to 0.45, which is not consequent. For $\mathit{MaxFEs} \leq 2$, the posterior method is not-consequently superior to the sparing and prior ones while it is the opposite for $\mathit{MaxFEs} \geq 2$.

For the study of the confidence-limit-average of the scores of the benchmark, a few conclusions can be stated:

- No method consequently outperforms another one in term of global efficiency.
- All methods have a convergence and value quality superior to 0.3. Therefore, no tested methods sacrifice either the value nor the convergence.
- The prior method, or eventually the sparing one, is to be preferred to have a high value quality while the posterior one is to be preferred in order to achieve a fast convergence. The sparing method is a competitor to the prior one in terms of value and convergence.
- Run reliability (which is linked to alpha and omega scores), Dimension and *MaxFEs* do not consequently impact the choice of a fiabilist method.
- The efficiency of a fiabilist method strongly depends on the test functions.

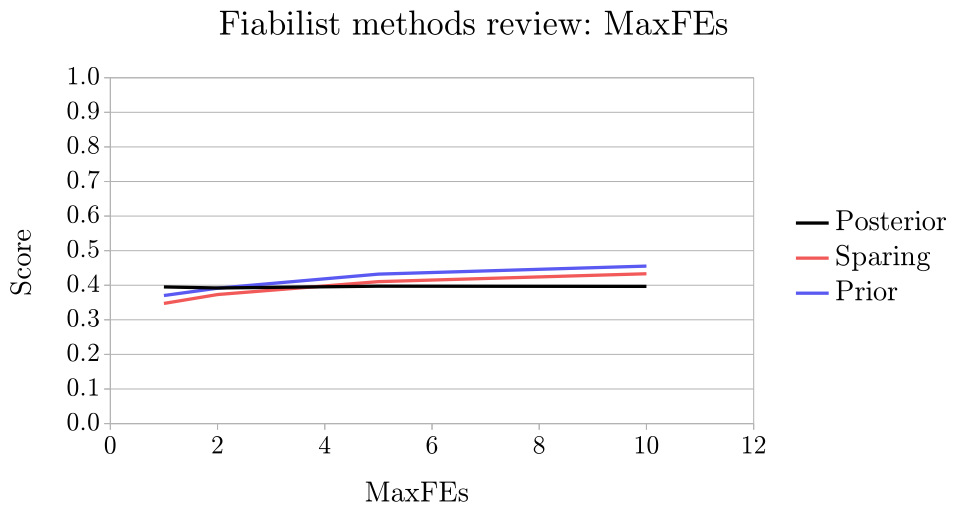


Figure 6.8: Fiabilist methods review: Average *MaxFEs* sub-scores

6.5.3/ Influence of the confidence limit

This section will present the influence of the confidence limit on the scores of the benchmark.

Figure 6.9 presents the global score variation with respect of the confidence limit for the different fiabilist methods. It can be observed that the efficiency of the posterior method remains almost equal to 0.4 no matter the confidence limit. As for the sparing (respectively the prior) method, the efficiency decreases from 0.4 (respectively 0.5) to 0.3. For $c \leq 50\%$, the prior and the sparing methods are superior to the posterior one, but not by more than 0.1. On the opposite, for $c \geq 50\%$, the prior and the sparing methods are inferior to the posterior one, but by less than 0.1. Thus, for tested confidence limits, no method is consequently more efficient than another.

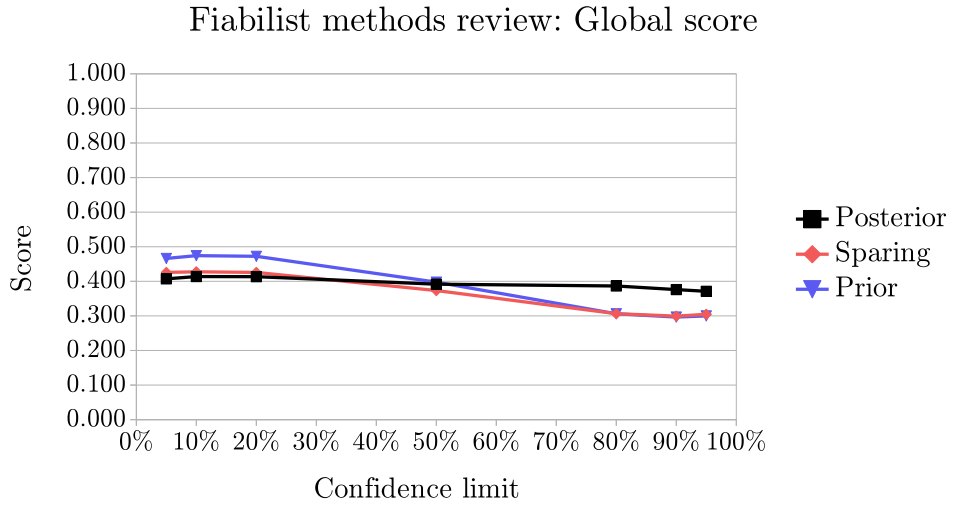


Figure 6.9: Fiabilist methods review: Global score with respect of the confidence limit

Figure 6.10 presents the value sub-score variation with respect of confidence limit for the different fiabilist methods. For all tested methods, the value remains the same for all tested confidence limits. The value of the prior and the sparing methods is equal to 0.4 while the value of the posterior one is equal to 0.3. In this case, confidence limits have a negligible influence on the quality of the value. Several explanations could be given to this surprising result. The uncertainties are small enough for the function variation to be negligible. The logarithmic scale used to normalized the objective-function masks the value decrease due to the increase of the confidence limit. Finally, the optimums of the functions used by the benchmark are wide compared to the uncertain area.

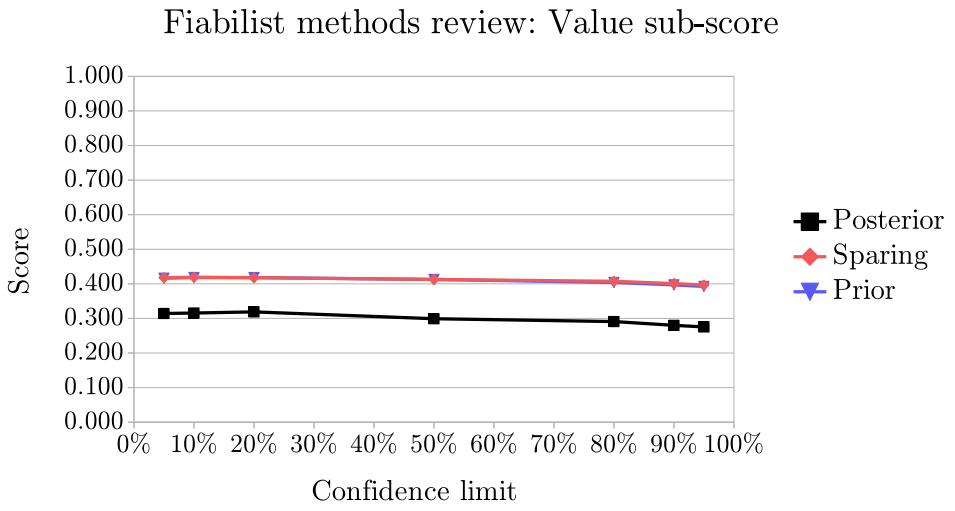


Figure 6.10: Fiabilist methods review: Value sub-score with respect of the confidence limit

Figure 6.11 presents the variation of the convergence sub-score with respect of the confidence limit for the different fiabilist methods. The convergence sub-score of the posterior method remains almost equal to 1. This could be explained because of different factors. First, with a posterior method, the problem is solved as if it was determinist, which explains why the confidence limit does not impact on the convergence. Then, *MaxFEs* is higher than 1 which is high for a 'pseudo-determinist' resolution. Finally, PSO possesses a fast convergence (see chapter 4). The convergence of the sparing and the prior methods, which decreases, seems to be inversely proportional to the sample size, depicted in figure 6.2. The convergence of the sparing (respectively the prior) method decreases from 0.6 (respectively 0.8) for c = 10% to 0.1 for c = 95%. It seems that there is an inversely proportional relation between the sample size and the convergence speed. Indeed, a sample size increases will induce an almost proportional increase in the number of evaluations done during an iteration. In addition, this is consistent with the almost equal to 1 convergence speed of the posterior method which does not use statistical evaluations to solve the problem.

In order to investigate the influence of the confidence limit on the function sub-scores, figure 6.12 presents the variation of the function sub-score with respect to the confidence limit for the prior method. It appears that, for all the functions, the variations are decreasing proportionally to the inverse of the sample size. In average, there is a function sub-score decrease of 0.2.

Some conclusions could be made based on this study:

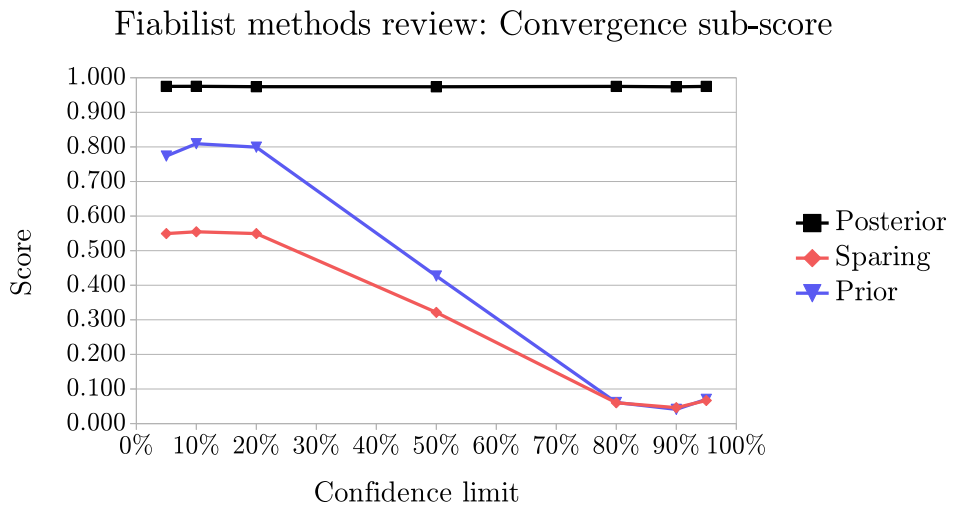


Figure 6.11: Fiabilist methods review: Convergence sub-score with respect of the confidence limit

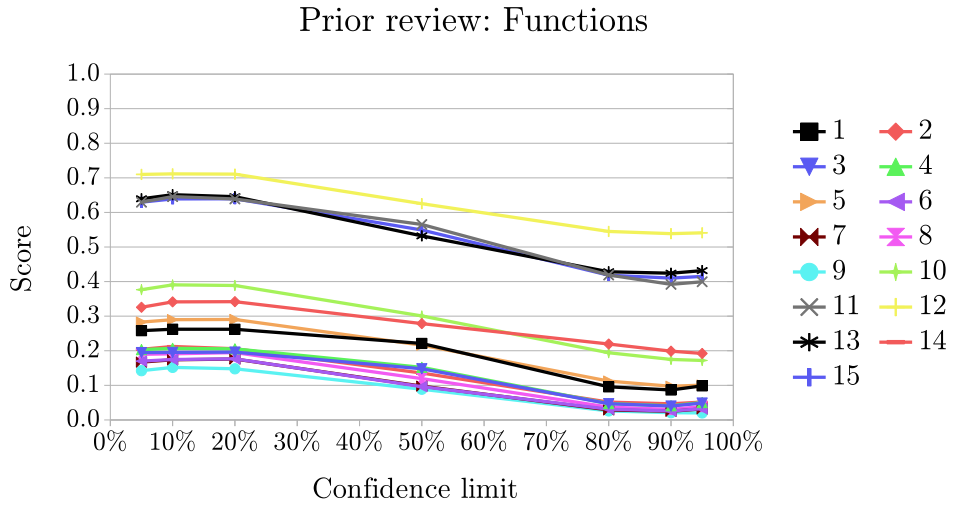


Figure 6.12: The influence of the confidence limit on the function sub-scores obtained by the prior function

- The tested methods have a similar efficiency.
- Thus, if a good value is sought, the prior method, or eventually the sparing one, should be chosen whereas if a fast convergence is sought, the posterior method should be preferred.
- The efficiency of the tested methods depends on the confidence limit and the function combination. Thus, some exceptions to the global remarks might occur.

6.6/ Conclusion

In order to take uncertainties into account without increasing too much the computational time, a SFO method has been developed. This method uses both determinist and statistical evaluations. A statistical evaluation is achieved by performing a sample of random determinist evaluations into the uncertain area. The sample is sized according to the WHO sampling rules. The SFO method is so that, after that all of the evaluations of an iteration have been performed in a determinist way, solutions serving for the global search mechanisms are re-evaluated in a fiabilist way.

The SFO method has been tested on the uncertain version of the benchmark, with different confidence limits, and compared with two other fiabilist methods. A few remarks could be made from the discussion of the results:

- The efficiency of the SFO method is equal to the posterior and prior ones.
- The convergence and value quality of the SFO method are both superior to 0.3.
- The SFO method obtained the highest value quality of 0.4 which is equal to the one obtained with the prior method.
- The efficiencies of the fiabilist methods depend on both the function and the confidence limit. Hence, the proposed method could be the most efficient one for some combinations of functions and confidence limits.

A remark about the results and the conclusion of this work has to be made. The set of functions used to test this method might be lowering the benefit of using it. The landscapes of the functions of the benchmark have been characterized and compared to the one of the test problem of this thesis in section A.10. It appears that the micro and macro ruggedness of the function of the benchmark are drastically lower than the one of the test problem. The higher the ruggedness, the higher the satisfaction degradation due to uncertainties is likely to be. Therefore, the FO methods have been tested on theoretical cases where the satisfaction degradation due to uncertainties is not as important as it is on the cases faced in this thesis. Therefore, the set of functions might be not sensitive enough to uncertainties for the SFO method to fully exhibit its full potential. It may be interesting to investigate the benefit of using the SFO method on a version of the benchmark with test functions having ruggedness values similar to the one of the real cases faced in this thesis.

The SFO method has been tested with different values of the confidence-limit, each time on 180 cases as the uncertain benchmark has been used as test method. The SFO method has been proven to be of interest in terms of value quality and efficiency. The results obtained in terms of value and convergence quality are consistent with the profitability challenge of this thesis. This method might be improved with further investigations. For instance, as the current sample sizing method induces a large population having a negative impact on the convergence speed, different sample strategies could be tested to improve the trade-off between the value quality and the convergence speed.

7

SAW FILTER OPTIMIZATION

Contents

7.1/ Introduction

The demand for Micro Electro-Mecanical Systems (MEMS) and Radio-Frequencies (RF) devices in particularly is increasing [1]. This is due to several mega-trends such as telecommunication, 5G or Smart automotive. Article [1] also states that 'Driven by the complexities associated with the move to 5G and the higher number of bands it brings, there is an increasing demand for RF filters in 4G/5G, making RF MEMS (BAW filters) the largest-growing MEMS segment.' and that 'This market will soar from US$2.3B in 2017 to US$15B in 2023.'.

To face the ever increasing needs of some markets, such as telecommunications with the 5G application, RF filters specifications should be improved. These specifications, detailed in sub-section 7.2.6, depend among other factors on the RF filter sizing.

As explained in the introduction, during an engineering process, once the product architecture has been defined, the product should be sized. The sizing phase is currently performed by an expert-based approach. The expert-based optimization does not satisfy the challenges of this thesis:

- **Rapidity**, which is how fast the optimization problem is solved. Currently, sizing a RF filter can last up to two weeks for an experienced designer. One of the goal of this thesis is to reduce this delay to a few days.
- **Efficiency**, which is how satisfying the product specifications are. Currently, when the bill of specifications is tough, even an experienced designer does not always find a sizing respecting all the required specifications. Another goal of this thesis is to find a solution fully satisfying the bill of specifications, even if it is a tough one.
- **Profitability**, which is how affordable the product is. Currently, some designs have a high failure rate due to manufacturing uncertainties. The last goal of this thesis is to reduce this rate by taking the manufacturing uncertainties into account during the sizing phase. In order to do so, the optimization process should lead to the selection of a reliable design.

To match with the challenges of this thesis, the sizing could be achieved through algorithmic optimization, see chapter 1. The RF filter sizing, which is an engineering design optimization (EDO) problem, will therefore be performed by an algorithm.

The optimization of RF devices has already been realized both by expert-based optimization [158, 159] and by algorithmic optimization [160, 161]. Article [161] concludes that better solutions could be found by improving the optimization means, such as the optimization algorithm. Therefore, to face the challenges of the RF market in terms of specifications, optimization methods adapted to RF filter should be developed.

This chapter aims to present how to use the EDO methods developed in this thesis on RF filter design optimization problems. The methods used in this chapter are the ones developed in chapters 2, 3, 4, 5 and 6. This chapter will focus on SAW filters as it corresponds to the industrial problems faced during this thesis.

To present the EDO application to SAW filter design, SAW filter design should be introduced first, which is done in section 7.2. Then, to solve a SAW filter design optimization problem, it should be formulated. Thus, how the methods linked to formulation, the framework and the NE approach, have been used is presented in section 7.3. Once the

problem has been formulated, a resolution method should be defined. How the methods related to the resolution, the benchmark and the MO, have been used is presented in section 7.4. Finally, the resolutions of the SAW filter design and the results will be discussed in section 7.5.

7.2/ SAW FILTER ENGINEERING

This section intends to present the SAW filter engineering process in order to understand the SAW filter optimization process. First, as the designer's goal is to design the filter in order to respect the specifications, they will be presented in sub-section 7.2.1. Then, an overview of the SAW filter working scheme will be detailed in sub-section 7.2.2. Finally, the following sections present the the steps of the SAW filter design process: The materials choice (sub-section 7.2.3), the architecture definition (sub-section 7.2.4), the modeling (sub-section 7.2.5), the sizing (sub-section 7.2.6) and the production (sub-section 7.2.7).

7.2.1/ SPECIFICATIONS OF THE SAW FILTER

This section will introduce the specifications of the SAW filter which are related to the transfer function. The designer's task is to design the filter in order to respect these specifications.

The specifications of the filter related to transfer function are given by figure 7.1. The transfer function should remain inside the template, meaning that this curve should be above the lower limit curve and under the upper limit curve. The part of the amplitude spectrum above lower limit curve will be referred as the bandwidth. The center frequency, f_0, is the frequency at the center of the lower limit. The insertion loss, which is the signal amplitude loss at the center frequency, should be lower than a threshold. The ripples, corresponding to the amplitude variation within bandwidth, should be lower than a designer-defined value. Finally the group delay should be inferior to a limit.

7.2.2/ WORKING SCHEME OVERVIEW OF THE SAW FILTER

The SAW devices use the piezo-electric effect [162] which converts electrical charges into mechanical stress and inversely. The SAW filter working scheme is explained through the delay line filter example, given in figure 7.2. It could be summarized like this:

1. The signal to filter reaches the inter-digited transducers (IDT) composed of periodic inter-digited electrodes. The electrical potential of electrodes will fluctuate.
2. Due to this potential fluctuation the piezo-electrical substrate in contact with the electrodes will be deformed. The deformation is periodical due to the periodicity of electrodes.
3. The local periodical substrate deformations will create an acoustic wave.
4. This wave will propagate through the substrate, in gaps, with electrical potential fluctuation linked to the piezo-electric substrate.
5. When the acoustic wave encounters a mirror, composed of periodic discontinuities, the wave is partially reflected and the rest of the wave keeps going.

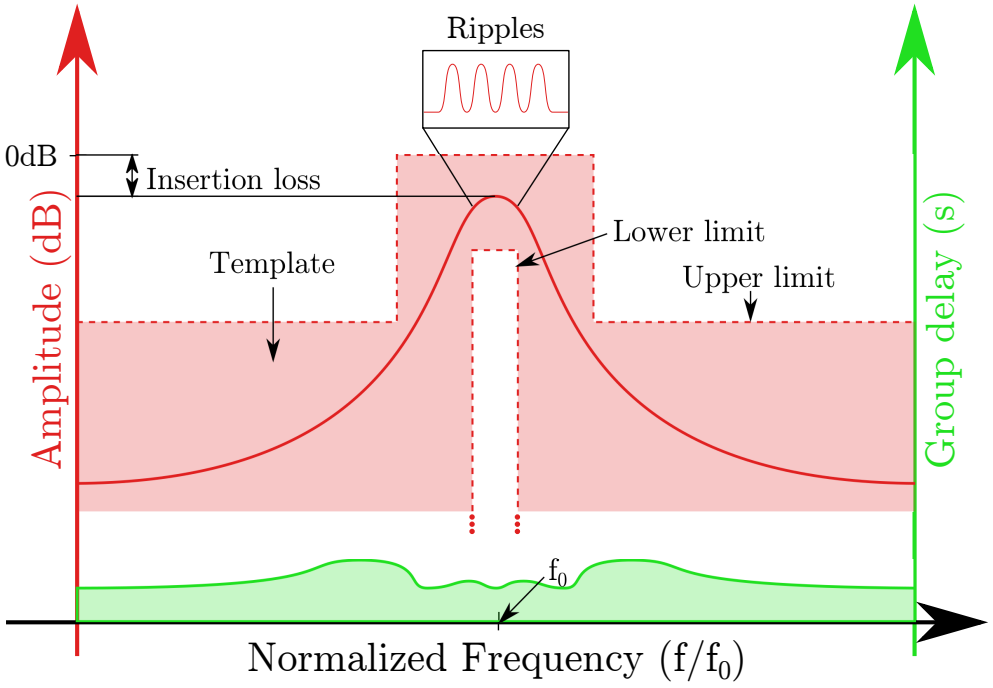


Figure 7.1: Filter specifications on the transfer function

6. Once the acoustic wave reaches back the electrodes, they will be exposed to electric potential fluctuations.
7. These fluctuations will generate a new filtered electrical signal.

With different filter architectures, the IDTs, the gaps and the mirrors will be organized differently leading to different working schemes. For instance, some filter architectures use mirrors at each side of the IDTs in order to create a cavity confining the SAW, which then becomes stationary.

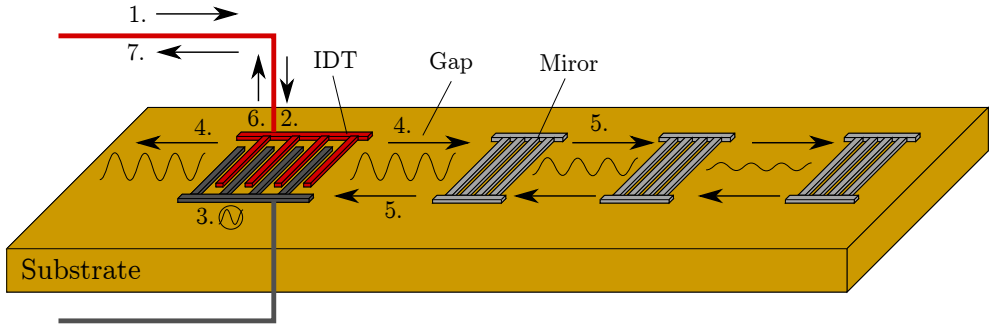


Figure 7.2: SAW filter working scheme through the delay line filter example

The SAW filters have a working frequency, which could be determined, at first order by equation 7.1. In this equation, V_ϕ depends on the substrate used while λ depends on the geometry of the electrodes. The working frequency will drive the filtering by defining the center frequency of the filter.

$$f = \frac{V_\phi}{\lambda} \tag{7.1}$$

A SAW filter is composed of mirrors, inter-digited transducers (IDT) and gaps, which will be used according to different architectures (see sub-section 7.2.4). A mirror is a network of electrodes or trenches whose period has been chosen so that, the acoustic wave is reflected when the device is used at the center frequency. An IDT [7] is a network of inter-digited electrodes converting electric signal into SAW as well as the opposite. A gap is a part of the substrate which is neither covered by an IDT nor by a mirror.

SAW filters could use different kinds of IDTs. In the application case of this thesis, a basic IDT composed of two fingers per wavelength having different potentials is used. An overview of IDTs is given in [163].

7.2.3/ Materials of the SAW filter

Commonly used materials for substrates are quartz (SiO_2), lithium niobate ($LiNbO_3$) and lithium tantalate ($LiTaO_3$) [2]. They are currently bought as mineral slices whose orientation has been carefully chosen [164]. These slices, also called wafers, could be found with different diameters, notably 100 and 150mm. For the electrodes, the materials usually chosen are aluminium and aluminium alloy [2]. The main criteria leading to the choice of materials, slice orientation and propagation direction are the following:

1. Electro-mechanical coupling coefficient (k^2) [165]: It traduces a piezo-electrical material ability to transform the electrical energy into a mechanical one, back and forth. This coefficient will impact the bandwidth. It could be computed thanks to the approximation given in equation 7.2, from [166]. It uses the wave velocity in the substrate (v_o) and in the electrodes (v_m).
2. The phase velocity (V_ϕ) [167]: This velocity, which depends on the geometry and the material of the electrodes, will affect the center frequency.
3. Thermal sensitivity coefficients [168]: These coefficients will induce perturbations as they traduce the variation of the wave frequency due to the temperature variation

$$k^2 = \frac{{v_o}^2 - {v_m}^2}{{v_o}^2} \tag{7.2}$$

For the application case of this thesis, the materials selected by the designer are:

- Substrate: quartz with a (YXl)/32 orientation.
- Electrode: Aluminum alloy

7.2.4/ Architecture of the SAW filter

This sub-section will introduce the topic of the SAW filter architecture. The architecture of the filter is defined by the designer, an expert in RF devices, for the product to be adapted to the specifications. This task relies on the designer's experience and is not subject to an optimization. Therefore, only basic knowledge related to this topic will be presented here. Once that the common architectures are presented, the one considered in this thesis will be detailed.

Different types of SAW filters architectures exist. They could be defined and classified according to different criteria. We will define the architecture of the filter of the application case from these criteria:

- Type of transducers - Classic filters [169]: Work as a network of transmitter and receiver on a propagation-free surface.
- Impedance elements [170, 171] usage - In lines filter: Uses an in-lines network of IDT, mirrors and gaps.
- Coupling type - Longitudinally coupled resonator filter (LCRF) [172]: These filters use coupling between two longitudinal resonating cavities.
- Specific properties - Double Mode SAW (DMS) [173]: An inline arrangement of input and output transducers backed by reflectors at the outer sides of the acoustic track.

The architecture chosen by the SMART-INN partner for the application case is a DMS. According to [174], they allow impedance transformation, they could have wide bandwidth, they offer a balun functionality for free and they exhibit an excellent far-off selectivity. This filter architecture is depicted in figure 7.3. This architecture exploits the coupling between two IDT, each linked to a different port. In this particular architecture as the coupling between the two IDTs is longitudinal, the filter is also considered as a LCRF. Each IDT is positioned between a mirror and a coupler. The mirrors are meant to totally reflect the acoustic waves while the couplers are meant to partially reflect it.

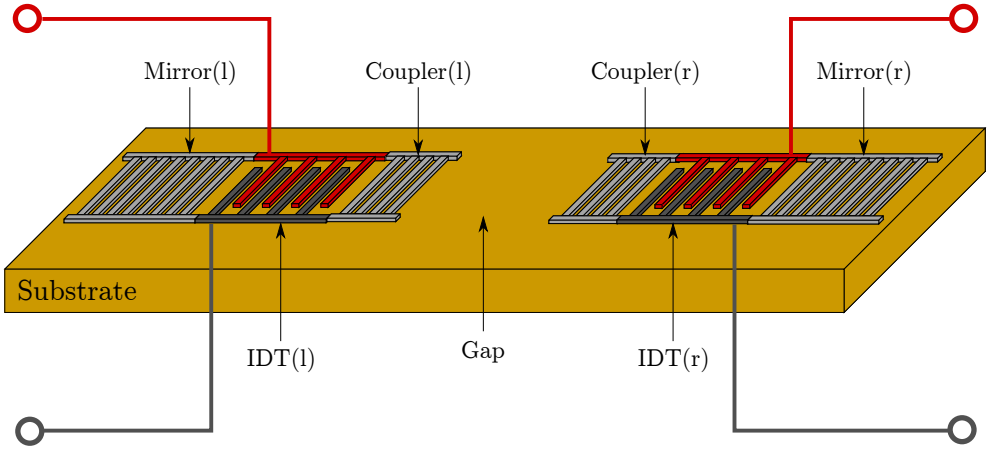


Figure 7.3: DMS filter architecture

A list of criteria influencing the architecture choice is given hereunder:

- The electrode reflection coefficient [175]: Measuring the part of the energy of the wave reflected by an electrode
- The directivity [2]: Which is the difference of phase between the emission and the reflection center of a periodic cell.
- The beam steering [176]: Which is the angle between the direction of the wave propagation and the wave pointing vector.
- The diffraction coefficient [176]: Which is defined as the derivative of the beam steering with respect of the preferred propagation direction [177].

7.2.5/ Modeling of the SAW Filter

This section aims to present how the SMART-INN partner modeled the SAW filter. Indeed this model will be used by the optimization engine as a co-simulation tool. A detailed explaination of the means used to model SAW filter could be found in [2]. The filter modeling considers subjects such as linear elasticity, materials an-isotropy, piezo-electricity an propagation in-homogeneity. This modeling is realized through three steps:

- *First step*: The behavior of a single periodic element, depicted in figure 7.4, is studied. This element is modeled as a two-dimension problem with a periodic limit condition. The period of the element (p) is equal to half of a wavelength (λ). The simulation of such an element will be done for several electrode geometries, defined by their heights (h) and their widths (a). The output will be a so-called 'grid' file containing, for different electrode geometries, periodic element behavior information such as the directivity, the phase velocity or the electro-mechanical coupling coefficient.

- *Second step*: the architecture of the filter is defined in a so-called 'dispo' file. This file defines the filter as a succession of periodic elements, such as the one defined in step 1. For each element, information such as the geometry of the electrodes or the materials used are defined. This file also contains general information such as simulation frequency range.

- *Third step*: The transfer function of the SAW filter is computed using a model whose inputs are design parameters, the 'dispo' and the 'grid' files. This model will simulate the filter behavior, for different frequencies, by multiplying a chain of matrices. The chain of matrices is defined by the 'dispo' file, and matrices values are defined thanks to 'grid' file information. The 'chained mixed matrix' multiplication will produce an overall admittance matrix, which is the response of the filter at a given frequency. The admittance matrices values, for the different frequencies, will be used by the model to produce the transfer function of the filter.

During this thesis, the designer is preforming the two first steps in order to produce a 'dispo' and a 'grid' files, both adapted to the specifications. These files are given to the optimizer for the optimization engine to be able to use the model described in step 3. The possibility of choosing among different 'grid' and 'dispo' files in order to include the architectural and the material choice into the optimization has not been investigated during this thesis.

7.2.6/ Sizing of the SAW Filter

This sub-section intends to present the filter sizing. As a reminder from chapter 1, the sizing phase of an engineering process consists in giving values to the design parameters of the product. First, the design parameters of the filter will be presented. Then, how the sizing phase is done will be discussed.

The design parameters of the filter, which were presented in figure 7.5, are detailed in table 7.1.

The sizing phase is commonly performed by the designer itself using its experience to adjust the values of the design parameters in order to meet the specifications. The

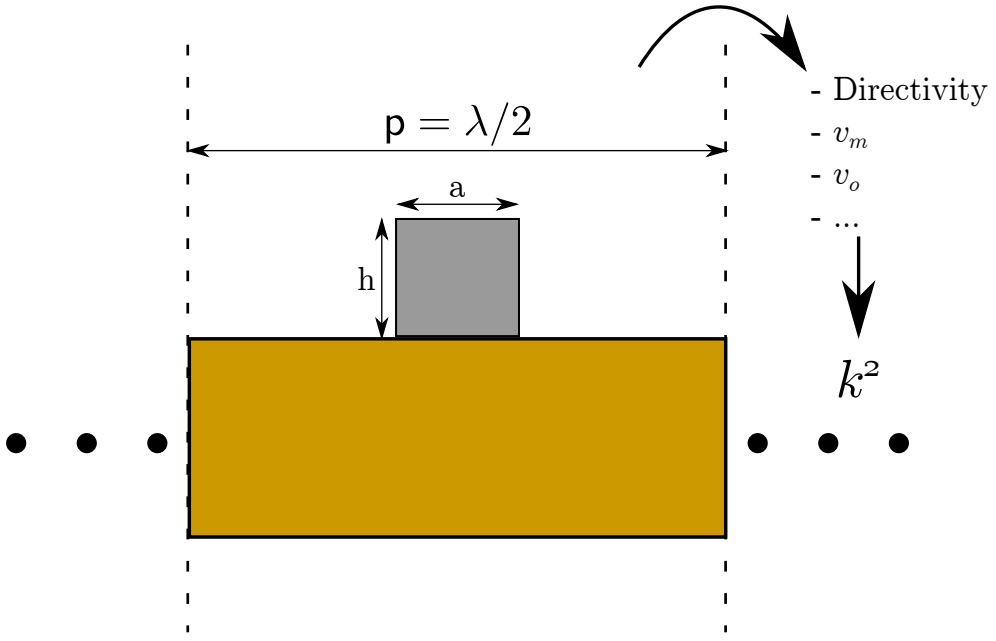


Figure 7.4: SAW filter modeling - unit element simulation

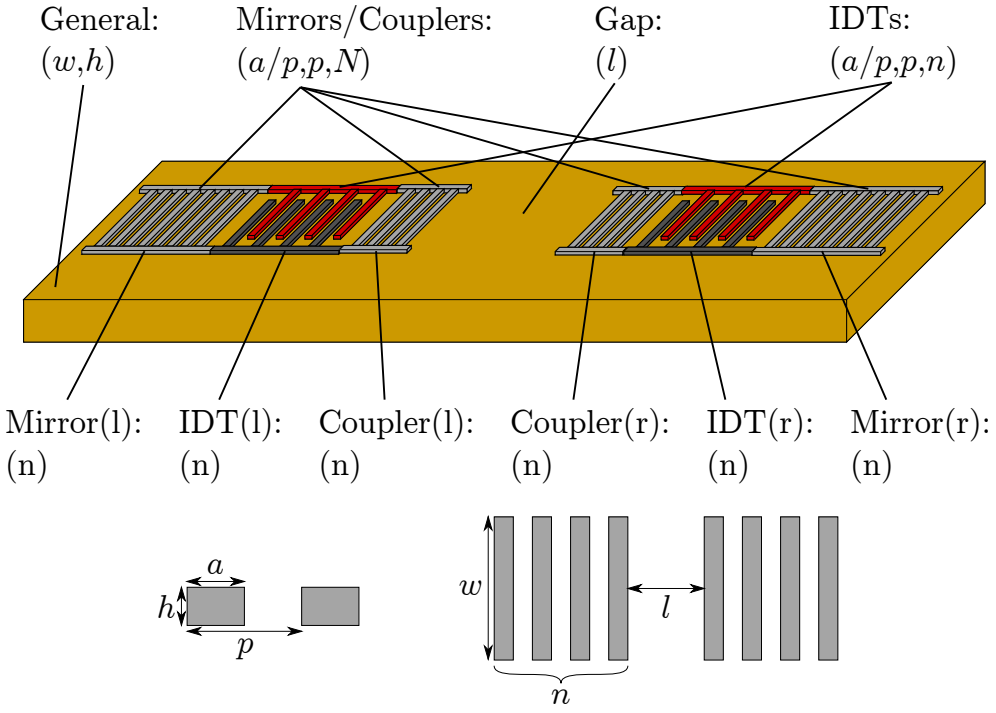


Figure 7.5: DMS design parameters

iterative process composed of design parameters adjustment, simulation and solution evaluation is realized by the designer to size its product. It can take up to two weeks for an experienced designer to meet a tough bill of specifications. Once the sizing validates the bill of specifications according to the simulation, the product is then manufactured, conditioned and measured. This thesis aims to develop methods to automate this task through an optimization process.

Table 7.1: Detail of DMS design parameters - given by element

Element	Parameter	Description
Overall	Electrodes height	Thickness of all electrodes
Overall	Electrodes aperture	Aperture of all electrodes
Transducer	Electrodes period	Electrodes Mechanical period
Transducer	Electrodes ratio	Ratio of an electrode width over the mechanical period of the electrodes
Transducer	Electrodes repetition	Number of digited electrodes forming the IDT
Mirror / Coupler	Electrodes period	Electrodes Mechanical period
Mirror / Coupler	Electrodes ratio	Ratio of an electrode width over the mechanical period of the electrodes
Mirror / Coupler	Electrodes repetition	Number of digited electrodes forming the mirror
Gap	Gap length	Gap length

7.2.7/ Manufacturing of the SAW Filter

This sub-section will present the SAW filter manufacturing process used in our application case. This process is introduced to present the uncertainties that should be considered in a fiabilist optimization. More information about the SAW filter manufacturing could be found in [2].

The filters, which are made in a clean room environment, could be manufactured using different processes. For the application case used in this thesis, the SAW filter is produced using lift-off. This process will be briefly described in order to introduce the sources of uncertainty. This process uses photolitography and metal deposition. Photolitography consists in using photo-sensitive resist to reproduce patterns coming from masks. In the application case, a positive resist [178] is used creating a positive copy of the mask as it dissolves once exposed to light. Metal deposition [179] consists of deposing a thin layer of metal material on a target. In the application case, the metal deposition is performed by a thermal evaporation technique.

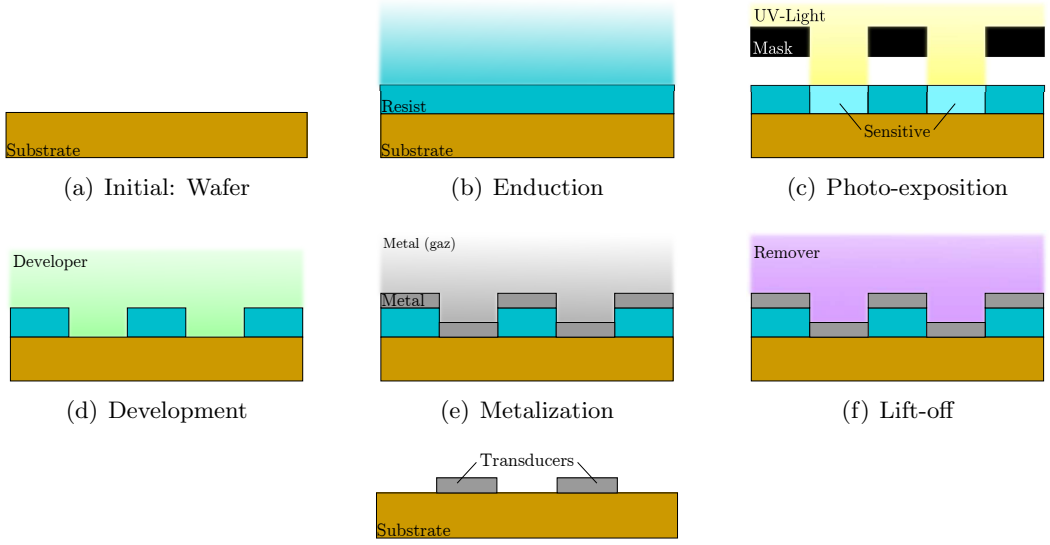
The different steps of the lift-off process used for the application case are summarized in figure 7.6.

The SAW filter manufacturing process steps, and how they induced uncertainties, will be detailed here:

1. Wafer (figure 7.6(a)): The properties and orientation of the wafer might be different from what was expected. This would induce uncertainties on the attenuations (l).

2. Enduction (figure 7.6(b)): The enduction is the deposition of resist on the wafer. The height and homogeneity of the deposition is subject to uncertainties that will impact the geometry of the electrodes.

3. Photo-exposition (figure 7.6(c)): The photo-exposition consists in insolating the resist for it to be sensitive to the developer. The mask is defined so that insolated part of the resist forms a pattern which corresponds to the architecture of the filter.



(a) Initial: Wafer (b) Enduction (c) Photo-exposition

(d) Development (e) Metalization (f) Lift-off

(g) Final: SAW filter

Figure 7.6: Lift-off process in the application case

However different phenomena creating uncertainties could be observed, such as the ones of the mask manufacturing or the light diffraction and diffusion from the mask openings. These phenomena will influence the geometry of the electrodes, notably a.

4. Development (figure 7.6(d)): The development consists in using developer to remove the sensitive resist. This operation will also influence a.

5. Metalization (figure 7.6(e)): A thin layer of metal is deposed over the wafer. The metal might not be homogeneously deposed and its average height might be different from the chosen one. Thus variations in h could be observed.

6. Lift-off (figure 7.6(f)): Lift-off consists in using a remover to remove resist an by extension the metal layer on in order for the metal to be deposed according to the architecture. During this operation, the metal at the edge of the resist might not be removed perfectly. This could induce variation in the filter behavior.

7. SAW filter (figure 7.6(g)): The filter is now manufactured and ready to be tested, conditioned and used. SAW filter uncertainties due to manufacturing that will be considered in this thesis are linked to the geometry of the electrodes (a, p, h) and to the attenuation (l). The ranges of these uncertainties, given in sub-section 7.3.1, depend on the means of production used by the Smart-Inn partners which won't be presented in this thesis.

7.3/ Formulation of the SAW filter optimization problem

This section presents how the methods developed in this thesis related to formulation have been applied to a SAW filter design problem. First, the framework, developed in

chapter 2, have been used to define the optimization problem given in sub-section 7.3.1. Then, the problem has been implemented inside an optimization engine [9] according to the normalized evaluations approach described in chapter 3.

7.3.1/ Framework usage

This section will present the formulation of the test problem which has been performed thanks to framework.

Table 7.2 presents the 15 mixed variables which has been identified.

Table 7.2: Variables of the test problem - Numerical data marked by an 'X' are confidential

#	Notations	Description	Type	Values	Unit	Accuracy	Uncertainty
1	asp1	Transducers a/p ratio	Continuous	$[0.4, 0.6]$		0.01	1e−03
2	p1	Transducers period	Continuous	$[0.2, 5]$	μm	0.1	1e−03
3	l1	Transducer attenuation	Continuous	X	dB/λ	X	X
4	asp2	Mirror a/p ratio	Continuous	$[0.4, 0.6]$		0.01	1e−03
5	p2	Mirror period	Continuous	$[0.2, 5]$	μm	0.1	1e−03
6	l2	Mirror attenuation	Continuous	X	dB/λ	X	X
7	p3	Gap length	Continuous	$[0.2, 5]$	μm	0.1	1e−03
8	l3	Gap attenuation	Continuous	X	dB/λ	X	X
9	em	Transducer and mirror electrodes thickness	Continuous	$[0, 0.2]$	μm	1e−03	1e−04
10	nrmg	Number of electrodes of the left mirror	Discrete	$\{300, 500\}$		1	0
11	nrtg	Number of pair of electrodes of the left transducer	Discrete	$\{40, 60\}$		1	0
12	nrcg	Number of pairs of electrodes of the left coupling mirror	Discrete	$\{50, 70\}$		1	0
13	nrcd	Number of pairs of electrodes of the right coupling mirror	Discrete	$\{50, 70\}$		1	0
14	nrtd	Number of pair of electrodes of the right transducer	Discrete	$\{40, 60\}$		1	0
15	nrmd	Number of electrodes of the right mirror	Discrete	$\{300, 500\}$		1	0

Table 7.3 presents the different identified constraints which are all inequality ones.

Table 7.3: Constraints of the test problem

#	Description	Expression
1	Insertion loss Ψ (dB) should be limited	$\Psi > -3$
2	Transfer function T (dB) should be superior to lower limit Lowerlimit	$\forall f, T(f) \geq \text{Lowerlimit}(f)$
3	Transfer function T (dB) should be inferior to upper limit Upperlimit	$\forall f, T(f) \leq \text{Upperlimit}(f)$
4	Ripples amplitudes R_a (dB) should be inferior to a limit R_l	$R_a \leq R_l$
5	Ripples number R_n should be inferior to a limit	$R_n \leq 1$

The single objective of this optimization problem is presented in table 7.4.

Table 7.4: Objectives of the test problem

#	Min/Max	Description	Notations	Unit	Acceptability
1	Min	Group delay	τ_g	s	1e−04

The properties of the single evaluation tool used for this problem, the Smart-inn partner simulation tool, are given in table 7.5. More information how this tool works could be found in [2]

Table 7.5: Evaluation tool of the test problem

#	Tool	Description	Inputs	Outputs
1	SAW Model	Filter behavior simulation	All variables, 'dispo' file, 'grid' file	Transfer function (Amplitude and phase)

Finally, the template information are summarized in table 7.6.

Table 7.6: SAW filter template

Specification	Range	Value
Center frequency	Confidential	f_0
template reference		f_0
Lower limit	$[-1e{-}04 \cdot f_0, 1e{-}04 \cdot f_0]$	−3dB
Upper limit	$[-\infty, -0.00104 \cdot f_0]U[0.00104 \cdot f_0, \infty]$	−30dB
Ripples amplitude	$[-1e{-}04 \cdot f_0, 1e{-}04 \cdot f_0]$	1dB

The described problem is a co-simulation, highly constrained, single-objective optimization problem with 15 mixed variables of different orders of magnitude. For this problem, the SMART-INN partners are willing to obtain a 95% success rate, which is consistent with litterature as some SAW devices could be process with a 96% reliability [180]. Therefore, for this SAW filter design problem, when uncertainties are considered, a 95% confidence limit will be used.

7.3.2/ Implementation of the normalized evaluations method

This sub-section will present how the problem is implemented in the new optimization engine using the normalized evaluations (NE) method, developped in chapter 3. How the problem is implemented should be discussed as the NE method is tuned by the optimizer according to the designer's indications. In addition, a waterfalls objective-function (WOF) [37] method which is meant to ease the resolution of highly constrained problems, is used.

WOF is a different way to take constraints into account, close to the constraints relaxation [35] and the constraints ordering methods [34]. Indeed, constraints are ordered and tested one after another. Depending on which constraints are fulfilled or not, a particular sub-objective-function, noted φ, is selected from a set. The set of sub-objective-functions holds one more element than the number of constraints, noted N_g. A workflow of WOF method, in the case of a constrained single-objective optimization problem, is given in figure 7.7.

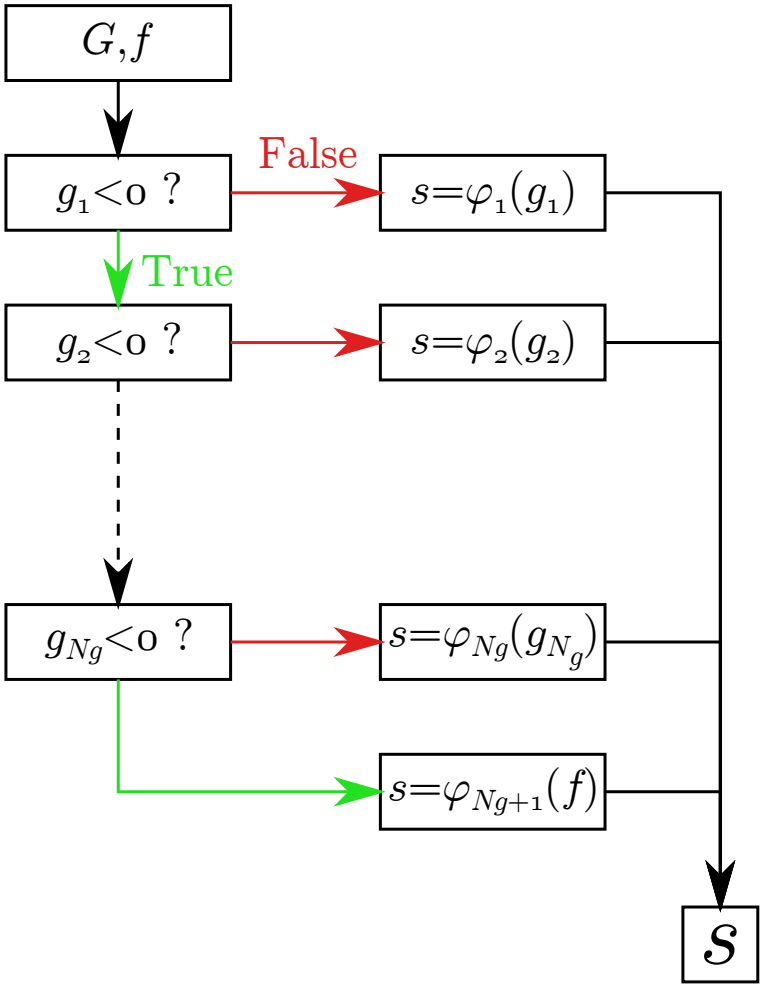


Figure 7.7: WOF workflow

As the NE approach is used, the sub-objective-function, noted φ, will be based on the penalties and the overall bonus which are normalized. The sub-objective-function used if all constraints are respected, φ_{N_g+1}, is the overall bonus, $\mathcal{B}$. The other sub-objective-functions are computed according to equation 7.3. In this equation, i is the WOF constraints index and p_i refers to the i-th penalty.

$$\varphi_i = -\frac{1}{N_g}(N_g - i + p_i) \tag{7.3}$$

As a reminder of chapter 3, penalties are computed according to equation 7.4. In this equation, the constraint g is considered respected if its value is below the constraint threshold, g^t. As for g^l, it represents the limit from which the penalty should be equal to 1.

$$g < g^t : \begin{cases} g < g^t \Rightarrow p = 0 \\ g > g^l \Rightarrow p = 1 \\ g \in [g^t, g^l] \Rightarrow p = ((g - g^t)/(g^l - g^t))^2 \end{cases} \tag{7.4}$$

Table 7.7 summarizes the information required to compute the penalties.

Table 7.7: Penalties computation elements

#	Constraints	Type	g	g^t	g^l
1	Insertion loss	Inferior	Ψ	3	100
2	Lower limit crossing	Inferior	P	0	1
3	Upper limit crossing	Inferior	S	0	1
4	Ripples amplitudes	Inferior	$\mathsf{R_a}$	1	2
5	Ripples number	Inferior	$\mathsf{R_n}$	1	100

The lower limit crossing evaluation is done as in equation 7.5. The upper limit crossing evaluation follows the same pattern.

$$\begin{cases} \mathsf{P} = \|\alpha(\mathsf{f})\|_{\mathsf{f}_{min}}^{\mathsf{f}_{max}} \\ \beta(\mathsf{f}) \in [0,1] \Rightarrow \alpha(\mathsf{f}) = \beta(\mathsf{f}) \\ \beta(\mathsf{f}) < 0 \Rightarrow \alpha(\mathsf{f}) = 0 \\ \beta(\mathsf{f}) > 1 \Rightarrow \alpha(\mathsf{f}) = 1 \\ \beta(\mathsf{f}) = (\mathit{Lowerlimit}(\mathsf{f}) - \mathsf{T}(\mathsf{f}))/(\mathit{Max}(\mathit{Lowerlimit}(\mathsf{f}) - \mathit{Min}(\mathit{Upperlimit}(\mathsf{f}))) \end{cases} \tag{7.5}$$

Finally, the overall bonus is equal to the group-delay objective bonus. This bonus is computed using a Wood function [110] in the lower the better case. This function has been tuned so that the satisfaction is equal to 1 (respectively 0) for Φ = 1E−06 (respectively Φ = 1E−04).

7.4/ A method to solve the SAW filter optimization problem

This section presents how the methods developed in this thesis related to resolution have been applied to the problem of the SAW filter design. First, how to choose an algorithm, thanks to the benchmark introduced in chapter 4, will be presented in sub-section 7.4.1. Then, how to tune the parameters of the algorithm, using the meta-optimization approach developed in chapter 5, will be detailed in sub-section 7.4.2.

7.4.1/ Choice of an algorithm thanks to the benchmark

This section will discuss how to choose an algorithm to perform robust optimization on the chosen problem. The same method as the one detailed in chapter 4 will be used. The idea is to discuss a set of relevant scores as in fuzzy logic methods in order to define the suitable algorithms. First, scores relevant to the problem to solve should be determined. In our case, the scores to use are:

- Global score: as it is always taken into account
- Value and convergence sub-score: as a balance between result quality and running time is seek

- Alpha sub-score: as the problem could be run several times
- $D10$ and $D20$ sub-score: as the dimension of the problem is 15
- $F13$, $F14$ and $F15$: as explained in A.10, the landscapes of these functions are consistent with the one of the application case.
- $M1$: as the designer is looking for a final solution and not an approximation, $MaxFEs = 1$ will be used as it corresponds to the full required running time.

As a reminder, in this method, scores will be labeled as 'low' if under 0.35, 'average' if between 0.35 and 0.65 and 'high' if above 0.65. According to this reminder, the following remarks could be made:

- Most of the scores of the PSO are average. Convergence, $F13$ and $F15$ ones are high and $F14$ one is low.
- All of the CMAES scores are average except $F13$, $F14$ and $F15$ ones which are low.
- Most of the genetic scores are average. $F14$ one is low whereas $F15$ one is high.
- Most of the Cuttlefish scores are average. Value one is high whereas convergence and $F14$ ones are low.
- SA scores are either average, such as value, alpha, $D10$, $F13$ and $F15$ ones or low, such as global, convergence, $D20$, $F14$ and $M1$ ones.

Based on the previous remarks, the PSO algorithm seems to be the wisest choice and the Genetic algorithm could obtain satisfying results. CMAES, Cuttlefish and SA are more likely to be less efficient on this problem. However, as mentioned in chapter 1, the PSO has been chosen to develop method during this thesis. So, between PSO and GA, the PSO will be used to solve the test problem.

7.4.2/ Tuning of the parameters of the algorithm by meta-optimization

This sub-section will present how the parameters of the PSO have been tuned using several meta-optimization (MO) approaches in order to solve the test problem. Indeed, as the MO approach developed in chapter 5 does not provide satisfying results on the application case, other MO approaches have been used. As a reminder of chapter 5, meta-optimization is the optimization of an algorithm by tuning its parameters. A meta-optimization approach relies on both an optimization method and an evaluation method. In chapter 5, the PSO has been optimized through a design of experiment (DoE) using the main score of the benchmark as an evaluation tool. First the different optimization approaches used will be presented. Then, their results will be discussed.

Several meta-optimization approaches, presented in table 7.8 could be used to tune the chosen algorithm. In this table, time-spans are given for the computer used during this thesis [1]. The expert-based approach (# 0) is accomplished by an expert in optimization

[1] Micro-Star International GL62 6QF (Processor: Intel(R) Core(TM) i7-6700HQ CPU @ 2.6GHz, 4 cores, 8 procesors; RAM: 8Go; Graphic Card: NVIDIA GeForce GTX 960M; OS: Microsoft Windows 10 Familly)

using its knowledge. The results of this approach will serve as a reference to evaluate the other approaches. The first tested approach (# 1) is the one described in chapter 5. From the principal component analysis (PCA) of this DoE, it appears that, in this particular case, the benchmark could be reduced. Indeed, from the PCA analysis realized in chapter 5, it seems that only a fraction of the problems of the benchmark could be used to perform meta-optimization with a reasonable loss of quality. Therefore a reduced benchmark, presented in table 7.9 has been used for approach # 2. As the time-span of this benchmark drops from 10h to 30s, an algorithm has been used as an optimization method. However, as demonstrated later in this sub-section, none of the two first approaches provides satisfying results on the test problem. Thus, a third approach (# 3) using a DoE on the test problem has been used. One should note that this approach has been tested with and without the normalized evaluation (NE) method described in chapter 3. Finally, the two last approaches (# 4 and # 5) have not been tested because their time-spans are not matching with Smart-Inn partners needs. The settings found by the different meta-optimization approaches are given in table 7.10.

Table 7.8: Meta-optimization approaches

#	Method	Tool	Timespan	Specific
0	Expert-based		*	General
1	DoE	Benchmark	1 week	General
2	Algorithm	Reduce Benchmark	2 days	General
3	DoE	Test problem	1 week	Specific
4	Algorithm	Benchmark	6 years	General
5	Algorithm	Test problem	5 years	Specific

Table 7.9: Reduced benchmark

Element	Number	Value
Dimension	$N_d = 1$	$D = 20$
Function	$N_f = 8$	$N_{uni} = 1$ Uni-modal (1) $N_{multi} = 5$ Multi-modal (3,4,7,8,9) $N_{hybrid} = 1$ Hybrid (11) $N_{comp} = 1$ Composite (13)
MaxFEs	$N_m = 1$	$MaxFEs \in \{0.2\}$
Draw	$N_t = 25$	
Runs	$N_r = 200$	$N_r = N_t \times N_c$

Meta-optimization approaches have been tested both on the benchmark and on the SAW filter problem. Their results are given in figure 7.8. First, on figure 7.8(a), it can be observed that all meta-optimization approaches obtain equal or better results on the benchmark than on the expert-based one. Meta-optimization approaches using the benchmark as an evaluation tool provide the best results on the benchmark. However, the global scores of the benchmark, remaining between 0.45 and 0.55, do not vary much. Figure 7.8(b) shows meta-optimization results on the SAW filter problem. Three different results are given, value, convergence and efficiency. Value corresponds to the end of the optimization result quality while convergence is the speed of the convergence quality.

Table 7.10: Settings of the PSO found by the tested Meta-optimization approaches

Parameters	N	c_1	c_2	w_{min}	w_{max}	V_{factor}
Expert-based	20	1	1	0.4	0.9	0.1
DoE benchmark	20	0.5	1.5	0.2	0.7	0.3
Algorithm reduced benchmark	20	0.2	1.3	0.3	0.8	0.4
DoE test problem	100	0.5	1	0.2	0.8	0.3
DoE test problem (NE)	50	0.8	1.2	0.25	0.75	0.1

Finally, efficiency is the combination of value and convergence quality. Meta-optimization using the benchmark results in a value quality inferior to 0.2 while the other one possesses the same value quality around 0.5. Hence, meta-optimization using benchmark will not provide satisfying results compared to the other one. This explains why two approaches using SAW filter as an evaluation tool have been performed. Those two approaches and the one using an algorithm on a reduced benchmark obtain a convergence speed superior to 0.5. It means that with these approaches the running time will be less than half of the allowed one. On the opposite, the expert-based approach and the DoE on the benchmark one have a convergence equal to 0, meaning that the algorithm does not converge within the given time. As for efficiency, the first remark is that it fluctuates greatly increasing from 0.1 to 0.6. Meta-optimization using the SAW filter as an evaluation tool provides significantly higher efficiencies (0.52 and 0.59) than the expert-based one (0.36). The approach giving the best result on the test problem is without any doubt, the DoE on test problem as it has the best value, convergence and quality. Therefore the setting found by this approach will be used to solve the test problem and any other similar problems.

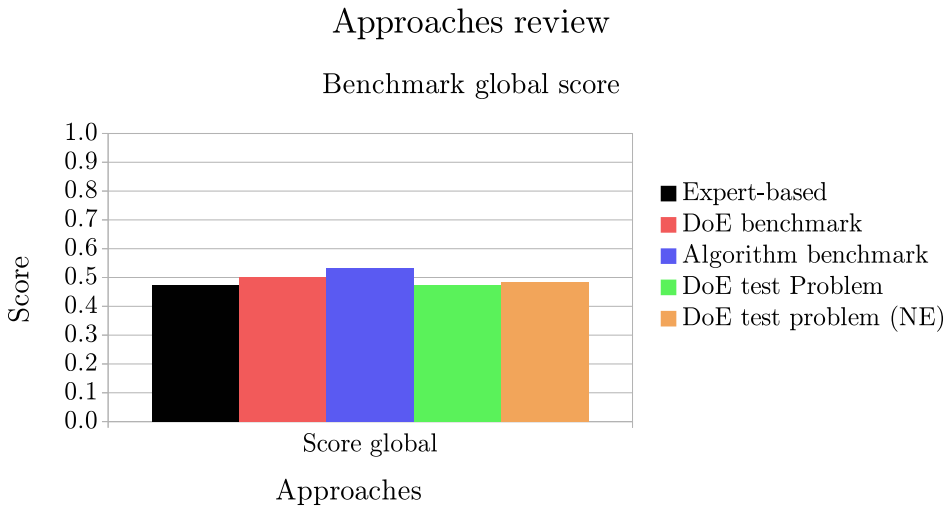
7.5/ Solving of the SAW filter optimization problem

This section will present how the SAW filter design optimization problem has been solved both in a determinist and in a fiabilist way. First, the problem is solved in its determinist form, not taking uncertainties into account, in sub-section 7.5.1. Then, in sub-section 7.5.2, the problem is solved in its fiabilist form meaning that uncertainties are considered.

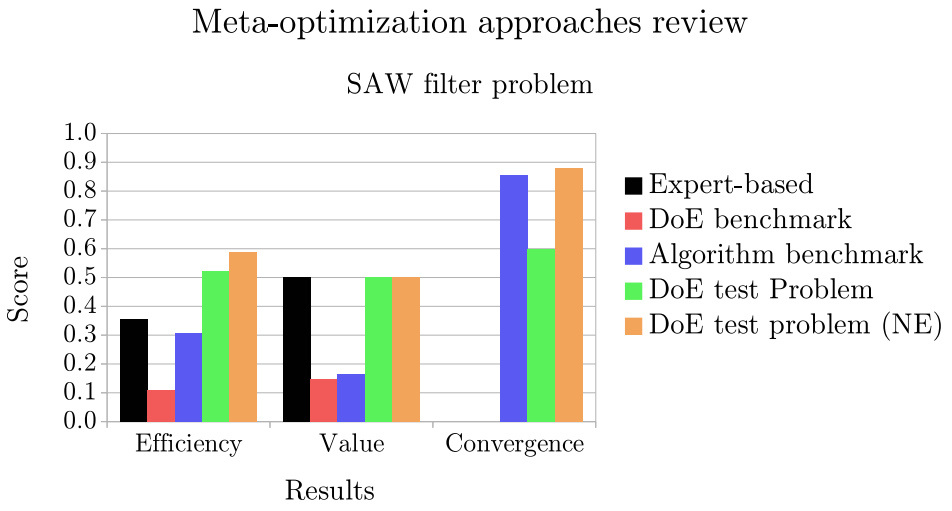
7.5.1/ Determinist resolution

This sub-section will discuss the determinist resolution of the test problem. To solve this problem, methods developed in previous chapters have been used. This resolution, labeled 'robust' has been compared to a resolution made before this thesis, labeled 'reference'. Figure 7.9 presents the transfer functions obtained with both resolution methods. The transfer functions in terms of amplitude are given in figure 7.9(a) while the transfer functions in terms of group delay are given in figure 7.9(b). For the amplitude transfer functions, specifications in terms of template have been added. As group delay is only considered in bandwidth, figure 7.9(b) only display group delay in bandwidth frequencies.

In figure 7.9(a), it can be observed that the reference solution has a 'spike' at $-4dB$ which is $1dB$ more than the allowed insertion loss. This 'spike' is at $1.001f_0$ which is outside the lower limit but inside the upper one. The reference solution respect the upper limit for most of the frequencies. Thus, the reference solution partially respects the constraints.



(a) Meta-optimization approaches review: Benchmark global score



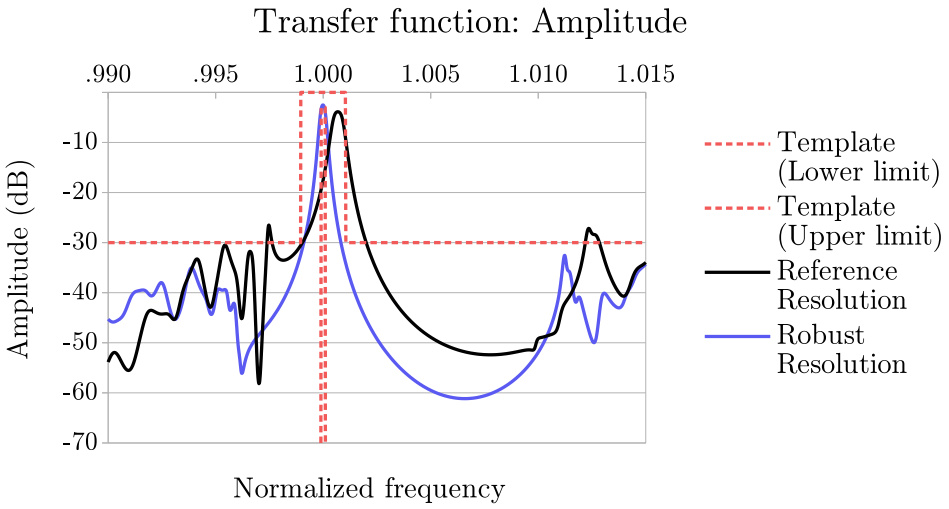
(b) Meta-optimization approaches review: SAW filter problem

Figure 7.8: Meta-optimization approaches review

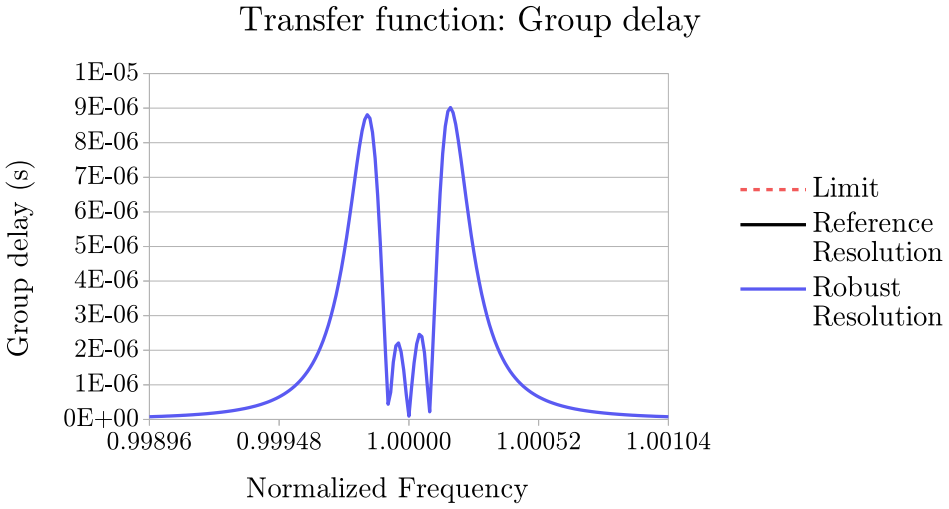
On the other side, the robust solution fully respects the problem constraints. Indeed, the 'spike', which is centered, reach $-2.5dB$, the template is respected and there is no ripples. Figure 7.9(b) present the group delay of the robust solution. It can bee seen that group delay remains inferior to the acceptability limit of $1e{-}4s$. Hence, the specifications obtained with the robust resolution fully satisfies the determinist optimization problem. Therefore, it can be concluded that, on the studied case, the methods developed during this thesis were necessary and sufficient to fully solve the problem in its determinist form.

7.5.2/ Fiabilist resolution

This sub-section will discuss the fiabilist resolution of the test problem. First, the robust resolution solution, from section 7.5.1, is re-evaluated in a fiabilist way. Indeed, it might



(a) Amplitude transfer function



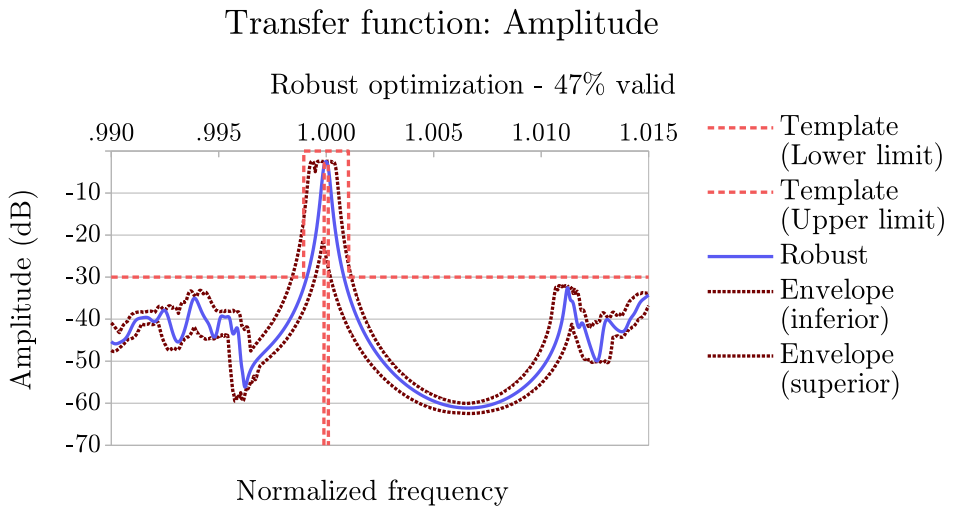
(b) Group delay transfer function

Figure 7.9: Determinist review

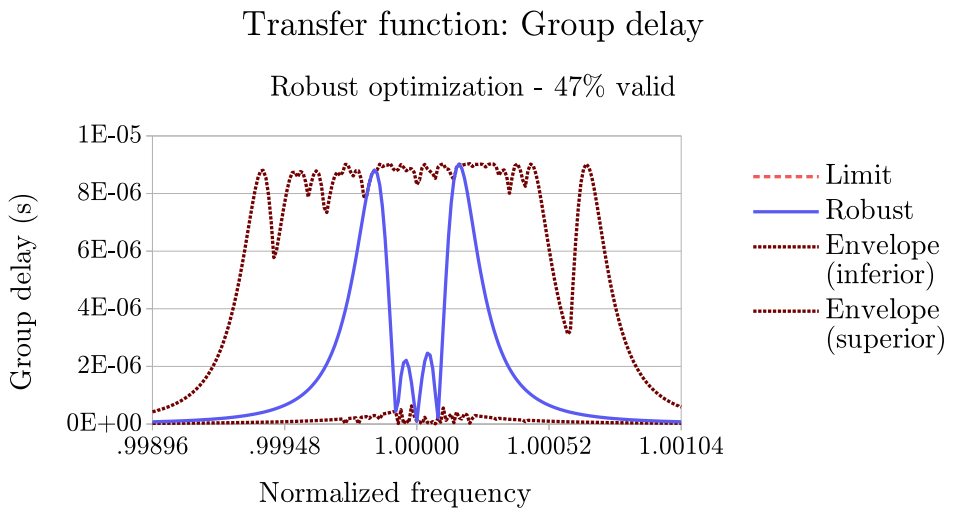
happen that the variation of this solution due to constraint is low enough for it to have a satisfying success rate. If it is not the case, a fiabilist resolution using the the sparing fiabilist optimization (SFO) method developed in chapter 6 will be done. This method, in its initial form, could be not suited for a given real case problem. In this case, the SFO method could be modified to the face real case issues.

Figure 7.10 present the transfer function of the robust resolution, from section 7.5.1, once re-evaluated in a fiabilist way. The amplitude is given in figure 7.10(a) and the group delay in 7.10(b). For amplitude and group delay curves, the envelope of the variation due to uncertainties, composed of an inferior and a superior curves, is added. This envelope gives for every frequency the min and max values of fiabilist sample's curves. This means that, for the given uncertainties, the transfer function is supposed to remains between these curves. For the group delay, the acceptability limit has been added. It represents

the threshold under which the group delay should remains for the designer to be fully satisfied by the group delay minimization objective.



(a) Amplitude transfer function



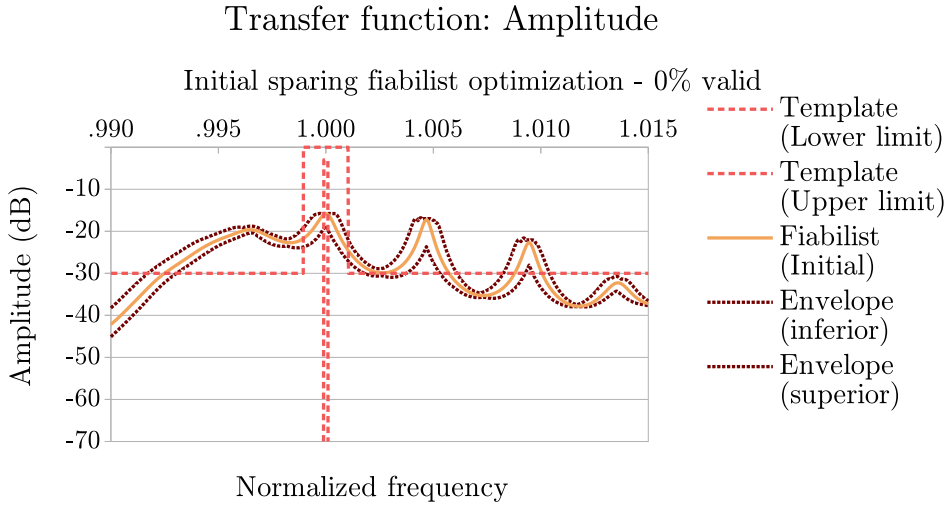
(b) Group delay transfer function

Figure 7.10: Robust optimization on SAW filter problem with uncertainties - transfer functions

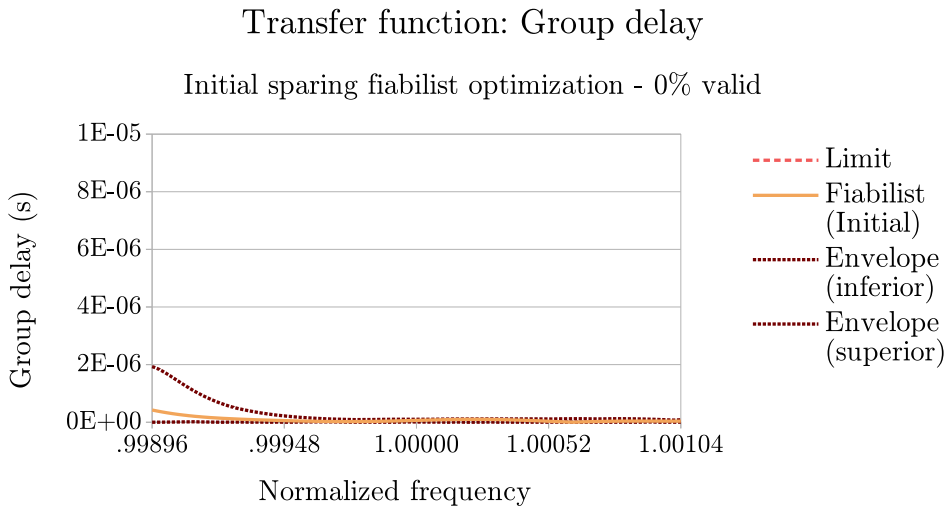
The robust solution present a 47% success rate, which is half of the 95% target. It could be observed on figure 7.10(a) that the transfer function variation seems to be mostly due to frequency shift. This frequency shift, which is about $0.001f_0$, is so that while the 'spike' remains in the upper limit, the template is crossed. On figure 7.10(b), it could be seen that the group phase deviation does not exceed $1e{-}5$ which is lower that the acceptability value of $1e{-}4$. Also it could be mentioned that the robust solution is close to a search space limit which lower its success rate. Indeed, some solutions inside the uncertain area does not respect the variables bounds. By extending the variables range, a higher success rate could be obtained for this solution. Thus, while not fully satisfying in

state, the robust solution could be validated by a designer willing to amend the problem. To seek for an even better solution, it has been decided to use the SFO method.

The SFO method method, presented in chapter 6, has been used to solve the SAW filter problem. The SFO method has been used in its initial form, meaning that no modification has been done. A 95% confidence limit has been used as it is the sought success rate. The transfer function obtained by this resolution is presented in figure 7.11.



(a) Sparing initial optimization - Amplitude transfer function



(b) Sparing initial optimization - Group delay transfer function

Figure 7.11: Initial fiabilist resolution of SAW filter problem with uncertainties using SFO method - transfer functions

The initial SFO method has not been able to find a 95% success rate solution, which might not exist. In figure 7.11(a) it could be observed that the initial SFO method found a solution with a $-15dB$ insertion loss. Hence, The initial SFO method failed to satisfy the first constraint of the WOF: the insertion loss reduction. Thus, it has been supposed that the test problem toughness is due to its constraints that are too restrictive. To verify

this hypothesis, how constrained the problem is has been checked. This has been done thanks to the method presented in section A.9. According to this method, the problem's feasible space ratio (FSR) is inferior to 1e−6. Thus, according to the constraint levels scale, presented in section A.9, the problem is excessively constrained. This means that the problem, in its determinist form, is almost impossible to solve in state. To face this problem it has been decided to modify the problem and the resolution method.

The optimization problem have been modified so that its goal is to find the solution maximizing the success rate. By doing so the designer could be able to estimate the success rate impact on the product price. In this case, the designer would choose wheter the product is validated or not depending on the trade-off between price and performance, which is a common practice in EDO [8]. The resolution method has also been modified to be face the issues of test problem. This new resolution method will be referred as modified SFO. The detail of the modifications and the associated reasons are listed here:

- Constraints have been relaxed: As one could notice on the transfer function obtained from the determinist resolution, presented in figure 7.9, the transfer function is at the edge of the template. In this case, to find a fiabilist solution, constraints should be relaxed so a solution slightly crossing template could be accepted.
- A center frequency shift is allowed but constrained: By observing the envelope of the amplitude transfer function of the initial fiabilist resolution, in figure 7.11(a), it could be suspected that a part of the variation of the transfer function is due to the center frequency shift. For a transfer function with a bandwidth which is short relative to the center frequency, if the template remains fixed at the desired center frequency, even a small center frequency shift could lead to a template crossing. Not to reject solutions that could respect the bill of specifications with a minor center frequency correction, a constrained center frequency shift has been added.
- The objective is to maximize the percentage of valid solutions: As there is no guarantee that a solution with a 95% success rate could be found, the objective has been updated to be the percentage of valid daughter solutions. The previous objective, the group phase minimization, has been transformed into a constraint, like in the ε-constraint method [31].
- The SFO method sample size has been decreased: As the goal is now to maximize the percentage of valid solutions, the confidence limit used by the SFO method is only applied to fiabilist evaluations having a success rate equal to zero. Therefore, the 95% confidence limit is no longer a criterion to adjust the sample size. As mentioned in chapter 6, the smaller the sample size the lower the negative impact on the efficiency of the algorithm. Therefore, the sample size has been lowered from its initial value of 89 to 20.
- The search space area has been restricted: From the landscape analysis of the SAW problem, made in section A.10, the values of ***FEM*** and ***DM*** indicate that this problem is likely to present multiple funnels. The search space has been restricted around the determinist solution to focus the search on the funnel containing the robust solution. Such a restriction could ease the research of a global fiabilist optimum.
- The CMAES algorithm has been used: In chapter 3, which is about the normalized evaluations approach, the test algorithms of this thesis have been tested on the problem considered in this chapter. It appears that, for this test, the CMAES is

the algorithm obtaining the best score in terms of efficiency and value. As the main criterion of the fiabilist resolution considered here is value, the CMAES has been chosen instead of the PSO for the fiabilist resolution.

- Robust value is used at the initialization: A common practice in optimization is to replace one or several initial solutions by the ones given by the optimization. This provides the algorithm a few good solutions to help it converging to the areas of interest. Thus, the algorithm is given the robust solution on initialization to ease its convergence towards the funnel containing potential fiabilist solutions.

The constraints, modified according to previous remarks, are presented in table 7.11.

Table 7.11: Relaxed constraints and attached penalties computation values

#	Constraints	Type	g	g^t	g^l
1	Center frequency shift	Inferior	$\delta(\mathsf{f_0})$	$2.604\text{e}{-04} \cdot \mathsf{f_0}$	$0.0156 \cdot \mathsf{f_0}$
2	Insertion loss	Inferior	Ψ	3	100
3	Lower limit crossing	Inferior	P	0.05	1
4	Upper limit crossing	Inferior	S	0.05	1
5	Ripples amplitudes	Inferior	$\mathsf{R_a}$	1	2
6	Ripples number	Inferior	$\mathsf{R_n}$	5	100
7	Group delay	Inferior	τ_g	1e−04	1e−02

The ranges of the variables once restricted are presented in table 7.12.

The transfer function obtained with the modified SFO method is presented in figure 7.12. Figure 7.12(a) presents the amplitude while figure 7.12(b) presents the group delay.

It could be seen in figure 7.12(a) that the resonance 'spike' is centered ($1.00008\mathsf{f_0}$) and that the insertion loss is inferior to $1dB$. As for the template, a 'spike' at $1.01 \cdot \mathsf{f_0}$ could be observed. This 'spike' could be explained by the constraint relaxation. The envelope indicates that the lower limit and upper limit could be crossed. The transfer function group delay, displayed in figure 7.12(b), remains under 1e−5 which means that the group delay constraint is respected. The fiabilist result as a succes rate of 71% which could be explained by the uncertainties influence leading to template crossing superior to the allowed threshold. This success rate is superior to the robust and initial SFO method ones. Therefore, the SFO method improved significantly the success rate to 71%. For this method to be fully satisfying a way to increase this rate to 95%, if possible, remains to be found.

In order to improve the success rate and reach the expected one of 95%, several solutions could be considered, among which:

- Increasing $MaxFEs$
- Improving the SFO method, for instance by optimizing the sample size
- Considering up-to-date algorithms such as the RB-IPOP-CMA-ES or the hybrid algorithms
- Improving the formulation of the problem by using the designer's knowledge and notions such as the alias.
- Discussing with the designer the possibility of lowering the values of the uncertainties

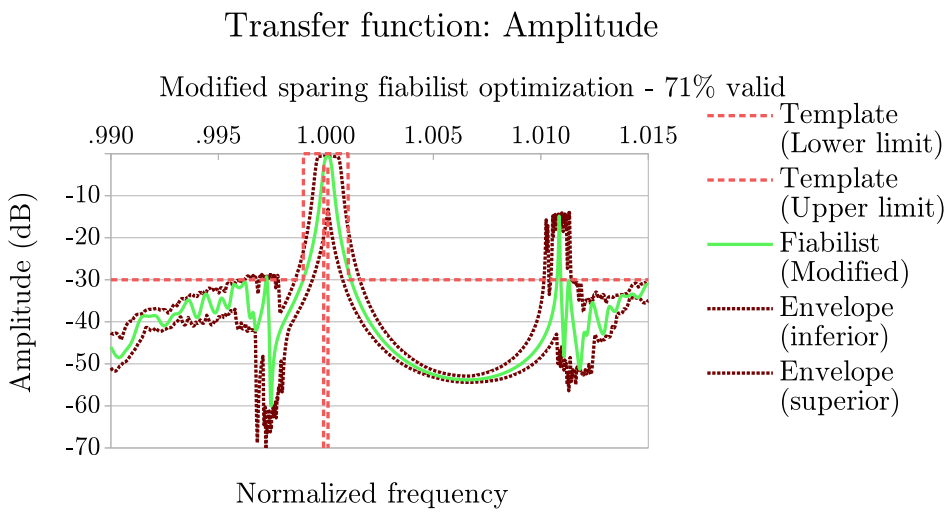
Table 7.12: The restricted variables - Numerical data marked by an 'X' are confidential

#	Notations	Description	Type	Restricted	Initial	Unit
1	asp1	Transducers a/p ratio	Continuous	[0.4, 0.5]	[0.4, 0.6]	
2	p1	Transducers period	Continuous	[1.5, 1.7]	[0.2, 5]	μm
3	l1	Transducer attenuation	Continuous	X	X	dB/λ
4	asp2	Mirror a/p ratio	Continuous	[0.4, 0.6]	[0.4, 0.6]	
5	p2	Mirror period	Continuous	[1.5, 1.7]	[0.2, 5]	μm
6	l2	Mirror attenuation	Continuous	X	X	dB/λ
7	p3	Gap length	Continuous	[2.5, 4.5]	[0.2, 5]	μm
8	l3	Gap attenuation	Continuous	X	X	dB/λ
9	em	Thickness of the electrodes of the transducer and the mirror	Continuous	[0.05, 0.15]	[0, 0.2]	μm
10	nrmg	Number of electrodes of the left mirror	Discrete	{300, 400}	{300, 500}	
11	nrtg	Number of pairs of electrodes of the left transducer	Discrete	{40, 50}	{40, 60}	
12	nrcg	Number of pairs of electrodes of the left coupling mirror	Discrete	{60, 70}	{50, 70}	
13	nrcd	Number of pairs of electrodes of the right coupling mirror	Discrete	{60, 70}	{50, 70}	
14	nrtd	Number of pairs of electrodes of the right transducer	Discrete	{40, 50}	{40, 60}	
15	nrmd	Number of electrodes of the right mirror	Discrete	{400, 500}	{300, 500}	

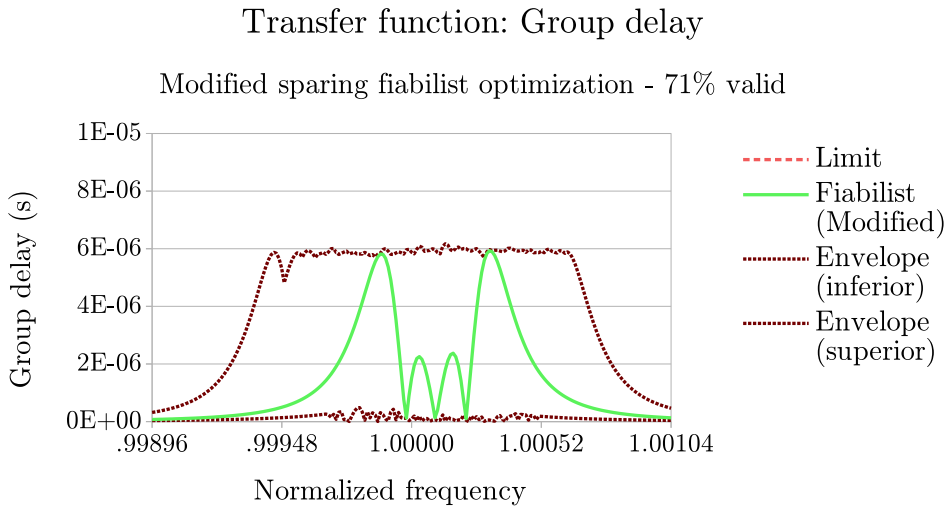
7.6/ Conclusion

The demand for high quality RF MEMS, especially for SAW filters, is increasing due to several mega-trends such as 5G in telecommunication. To meet the ever increasing specifications of the SAW filters, the sizing should be performed using optimization. To face the specific needs of the Smart-Inn project, some methods have been developed during this thesis. This chapter intends to test if the methods developed during this thesis are sufficient to solve the test problem of this thesis, the sizing of a DMS SAW filter. First, an introduction to the application domain, the SAW filter engineering have been made. Then, how to use the framework to formulate the problem and to tune the NE method in order to solve such a problem has been discussed. After that, how to choose a proper algorithm and to tune its parameters has been detailed. Finally, the determinist and the fiabilist resolution of this problem have been performed and their results were discussed.

Several comments could be made about the results of the section 7.5. First, for the determinist resolution, a solution fully satisfying the specifications has been found thanks to the methods developed in this thesis. Therefore, the benefit of using these methods is proven. For the fiabilist resolution, the sparing fiabilist optimization method developed in chapter 6, with an adequate resolution method adjustment, managed to found a solution with a 71% success rate which is superior to the one of the robust resolution being of 47%. Though, the targeted 95% success rate has not been reached, the solution might be accepted by a designer considering that the trade-off between product price and performance is so that product could be positioned on the market. Several solutions could be explored in order to raise this success rate to 95%, which is the profitability challenge



(a) Amplitude transfer function



(b) Group delay transfer function

Figure 7.12: Modified sparing fiabilist optimization on SAW filter problem with uncertainties using SFO method - transfer functions

criterion.

According to [8], one of the three major areas of improvement for EDO is the efficient use of human knowledge. During the formulation, how the RF filter expert is designing the filter has not been discussed though it could be investigated. Indeed, for a given specification, not all the parameters of the filter design do necessarily have a significant influence. With the resolution method used in this chapter, constraints are considered one by one. Therefore, a method to solve the problem step by step, by using only a fraction of the variables at a time, could be investigated. Another possibility is to include the material and the structural choices in the optimization problem. Indeed, both these elements have a strong influence on the specifications of the filter. By including them into the problem, un-explored efficient combinations of structure, material and sizing might be found.

Conclusion

This thesis, which is part of the SMART-INN project, aims to develop optimization methods to meet partners' needs in optimization. The challenges of this project are: rapidity, efficiency and profitability. Rapidity means how fast the optimization problem is solved. Efficiency relates to how good the specifications of the product are. Profitability is how inexpensive the product is. In order to face these three challenges, five research works have been conducted:

- The *Framework*: how to properly formulate an optimization problem.
- The *Normalized Evaluations Approach*: how to solve an optimization problem according to the designer's expectations.
- The *Benchmark*: how to assess the performances of algorithms to be able to select the appropriate one for a given problem.
- The *Meta-Optimization*: how to tune the parameters of an algorithm for it to be as efficient as possible.
- The *Sparing Fiabilist Optimization*: how to take uncertainties into account without increasing too much the numerical resolution timespan.

The methods developed during this thesis have been used to optimize a SAW filter.

Contribution, response to challenges and limits

The contribution to the robust and fiabilist optimization will be presented chapter by chapter. Afterwards, a synthesis about the achievements of this thesis will be given.

Framework

A framework to properly formulate an optimization problem has been presented in chapter 2. This framework is based on three interviews. Each interview lasts between two and three hours. If each interview is finished in a half-day period, a problem could be fully formulated in a day and half. If the framework, which has not been tested yet, allows to formulate a problem without ambiguities, the formulation phase is considered successful. Thus, except if the problem is too difficult to solve, the resolution and validation phase should proceed. Concerning optimization problems encountered during this thesis, the typical running time is about a few hours. By considering the time to set the resolution method and to perform a few runs, the resolution phase should be about one to three days. If the designer takes half a day for validation, the total optimization process timespan is three to five days. Therefore, by using the framework, the optimization process timespan

could be no more than a week. This timespan matches with the rapidity challenge of this thesis.

Normalized evaluations approach

A normalized evaluations approach, aiming to solve an optimization problem according to the designers' expectations, has been presented in chapter 3. This approach has been compared to a classical evaluation method on an industrial problem. The comparison has been performed by using five meta-heuristic algorithms. On this test, the normalized evaluations approach exhibit higher performances in terms of value quality, convergence quality and efficiency. On the test problem, the average value quality of the algorithms has been improved by 15%. This means that the algorithms are able to find better solutions. In term of convergence quality, the PSO and CMAES values have been increased by more than 10% and GA, which was not converging, has a convergence quality of 0.8. Therefore, normalized evaluations approach significantly reduces the running time. Consequently, as the running time and result quality have been improved, the resolution phase should be eased, which is confirmed by the 30% average increase of efficiency. As the resolution phase has been eased, the timespan could be reduced. A resolution timespan reduction will lower the optimization process timespan which is one of the challenges of this thesis. In addition, the CMAES has a quality value of 1, meaning that a solution fully respecting the bill of specifications has been found. As the tested problem is excessively constrained (see section A.9), the normalized evaluations approach enables to find a solution to a tough problem. This corresponds to the efficiency challenge definition. Thus, on an industrial problem case, the normalized evaluations approach has been sufficient to solve the efficiency challenge. Several bound handling techniques could be used with such an approach. However as their influence has not been investigated yet, the default choice might not be the optimal one. In addition, the proposed approach used an advanced evaluation method, whose limits and drawbacks are not known yet. For instance, the reason why the GA value quality has been degraded with normalized evaluations approach remains to be investigated.

Benchmark

A benchmark, which is a tool to assess the performances of algorithms, is proposed in chapter 4. Several algorithms, commonly used in EDO, have been tested with the proposed benchmark. The main scores of the algorithms are consistent with the literature, which is not the case when the CEC benchmark is used. The reason why such a scoring method could lead to inconsistent results and the reasons why using classical measures of performances is not sufficient in our case have been explained. The sub-scores of the algorithms have been discussed in such a way they highlight the strengths and weaknesses of the algorithms. Based on the scores of the benchmark, a method to select a suitable algorithm for a given problem is proposed. It has been tested on a practical case and identifies three possible algorithms in a panel of five ones. The five algorithms have been runned several times on the problem used for demonstration. Some remarks about the results of the algorithms could be made. First, the worst average value quality is less than half of the best one meaning that a poor choice of algorithm could drastically reduce the end of the optimization value. The second remark is that the best (respectively average) value quality of the algorithms identified as suitable are superior or equal to 98% (respectively 90%) of the best results of literature. Thus, the proposed benchmark is a valid tool

to assess the performance of the algorithms and to select a proper algorithm for a given problem. Using the proposed benchmark is essential to face the efficiency challenge of this thesis. The proposed benchmark uses the same objective-functions set as the CEC 2015 which suffers from a diversity bias [41]. Therefore, the proposed benchmark suffers from the same bias. Also, the selection of a proper algorithm for a given problem thanks to the scores obtained is explained but a formal method is not established yet.

Meta-optimization

A meta-optimization approach, tuning the parameters of an algorithm to improve its efficiency, is proposed in chapter 5. A principal component analysis (PCA) has been performed on the meta-optimization results. From this PCA, it appears that, on this particular case, some of the benchmark constitutive elements have a similar effect on the determination of an optimal setting. This observation means that, in this particular case, a reduced benchmark requiring less computation time, could be used to perform the meta-optimization. An additional design of experiments has been done on the sub-scores of the benchmark to provide for each one the optimal settings. These sub-settings could be used to adapt the optimal setting for a given problem. The main setting, which optimizes the global score of the benchmark, has been tested and used to solve both the benchmark developed in this thesis and the industrial problem of reference. Its results were compared to the ones obtained by using the settings of other meta-optimization approaches. They include the expert-based approach which is used as a reference. Concerning the global score of the benchmark, all the tested approaches were outperformed by the proposed one. In average, they were exceeded by 13%. This improvement helps to face the efficiency challenge of this thesis. Currently, the proposed meta-optimization approach provides a general setting and a set of sub-settings optimized for the sub-scores of the benchmark. These sub-settings could be used to adapt the main setting to a problem. However, the proposed meta-optimization approach could not be directly used to meta-optimize an algorithm for a given problem yet. For this to be possible, a method using sub-scores to compute an alternative score specific to a given problem, should be designed.

Sparing fiabilist optimization

A sparing fiabilist optimization method, which aims to take the uncertainties into account without increasing too much the numerical resolution timespan, is presented in chapter 6. The SFO method has been extensively tested on the uncertain version of the benchmark, with different confidence limits, and compared with two other fiabilist methods. The SFO method has been proven to be of interest in terms of value quality and efficiency. This method has an efficiency of 0.4, similar to posterior and prior ones. The SFO is competitive in terms of value quality as it shares the best value quality with the prior method. The efficiency of fiabilist methods depend on the function and on the confidence limit. Thus, the proposed method could be the most efficient one. It could be noted that the efficiency of the SFO method on hybrid and composite functions could be higher than 0.5. In addition, the set of functions used to test this method might be lowering its benefit meaning it could exhibit even better results on real case. Finally, the results obtained by the SFO method, in terms of value and convergence quality, are consistent with the profitability challenge this work is meant to address. This method might be improved through further investigation, notably on the sample size parameter which has a negative

impact on the convergence speed.

SAW filter optimization

The way to optimize the SAW filter is detailed in chapter 7. A test problem has been solved with and without considering the uncertainties by using the works of the previous chapters. The proposed meta-optimization approach does not provide good results on the test problem, therefore several other meta-optimization approaches were tried. These approaches were tested on the benchmark and on an industrial case by comparing their settings results. An expert-based setting is used as a reference. For the main score of the benchmark, the tested meta-optimization approaches obtain equal or better results on the benchmark than the expert-based one. On the industrial problem, it can be observed that meta-optimization approaches using the benchmark as an evaluation tool are less efficient than the expert-based one. On the opposite, the meta-optimization approaches using test problems as evaluation tools are more efficient, indicating an efficiency score rising from 0.35 to at least 0.5. Therefore it could be concluded that if the setting found with the meta-optimization approach proposed in chapter 5 is not adapted to the problem to solve, it is possible to perform the meta-optimization on an analog problem.

Thanks to these additional meta-optimizations, the resolution method was set and the test problem could be solved. For the determinist resolution, with the developed methods, a solution that fully respects the bill of specifications has been found whereas it is not the case of the optimization taken as reference. For the fiabilist resolution, the sparing fiabilist optimization method developed in chapter 6, with adequate resolution method adjustment, have found a solution with a 71% success rate which might be accepted by a designer considering that the trade-off between product price and performance is so that product could be positioned on the market. Several solutions could be explore to raise this success rate to 95%, which is the profitability challenge criteria. However, this rate might not be reached for this particular case. Indeed, after additional analysis, it appears that the test problem is excessively constrained ($FSR \leq 1E-6$), meaning that constraints are strong enough for the problem to be almost impossible to solve. This explains why the sparing fiabilist method could not find a fully satisfying solution.

Thus, this chapter demonstrates that the methods developed during this thesis are sufficient to face both rapidity and efficiency challenges on the test case. In addition it seems possible, for problems which are not excessively constrained, to face these challenges altogether. Thus, concerning this chapter, the rapidity and efficiency challenges were faced and it seems that the profitability one could be faced also if the problem is eased. Finally, additional meta-optimization approaches presented in this chapter could be investigated.

Synthesis

The challenges of this thesis are: rapidity, efficiency and profitability. Thanks to the methodology (chapter 2) an optimization process timespan will probably be inferior to a week. The normalized evaluations approach (chapter 3), improving both the end of optimization value and the convergence speed, will ease the resolution phase reducing the timespan of this phase. The benchmark (chapter 4) is a reliable tool to select a suitable algorithm for a given problem, which is a critical task as it could reduce the value quality by more than two. The meta-optimization (chapter 5) could improve the efficiency of the algorithm by 13% compared with to standard setting. The sparing fiabilist optimization

method (chapter 6) is of interest in terms of efficiency as it obtains a competitive value quality without sacrifying convergence speed. The methods developed during this thesis were applied to solve a test case both in a determinist and fiabilist way (chapter 7). For the determinist resolution, though the problem was excessively constrained, a robust solution fully respecting the bill of specifications has been found. For the fiabilist resolution, a solution with a 71% success rate which might be accepted by the designer has been found. Several solutions could be explore to raise this success rate to 95% which might not be possible due to how constrained the problem is. Therefore, on the tested case, the developed methods were sufficient to solve both the rapidity and the efficiency challenges together and it seems possible, for problems which are not excessively constrained, to solve the profitability challenge in addition.

As a conclusion, the rapidity and efficiency challenges could be faced at the same time. As long as the problem is not excessively constrained, there is a chance to face the profitability challenge in addition. Additional tests should be done to verify if the three challenges could be faced together at least on a problem not too much constrained. The goal of this thesis which is developing optimization methods to face SMART-INN partners needs in optimization is achieved.

Perspectives

The developments achieved during this thesis gave ideas for future research works. Such a work corresponds to an independent short-term (a few months) research subject. These research works have been used as a basis to define research projects. A research project is a mid-term (a few years) research subject composed of several research works. The research projects are meant to solve high-level industrial or scientific problematics. Four research projects will be presented in this section.

Research project 1: To an EDO acceptation

This research project aims to help the EDO to be accepted by the designers to be spread in the industry. To conduct the design process, designers are provided with a large panel of methodology, for instance the FAST diagram one. As no such methodologies exist to conduct the EDO process, the designers should either call for an expert in optimization or become experts themselves. This state of fact is one of the cause of the relative rareness of the EDO use in industry compared to simulation or finite element use. For the EDO to be accepted, designer-centered methodologies should be developed in order to conduct the EDO process. The research works composing this research project are:

1. A framework test protocol: In order to properly develop, correct and validate the framework proposed in chapter 2, a test protocol should be established. This protocol should define: the volunteer profile, the test conditions, what is measured and how to validate whether the framework is consistent or not. The idea is to determine if the framework improves the formulation, by reducing ambiguities, or not.

2. An improved framework ready for real-life use: The framework should be used in a large panel of real applications in order to detect and then correct its limits. The idea is to improve the methodology enough for optimizers and designers to have confidence in it and for it to be able to face real-case difficulties.

3. A designer centered optimization engine: The goal of this work is to define how an optimization engine should be interfaced to be as convenient as possible to use by a designer. To spread EDO into the industry field, it is essential that the designers have a user-friendly tool they could use on their own without being an expert in optimization.

4. Optimization case characterization (landscape function): This work is about how to fully characterize an optimization case. What information about an optimization case would matter in order to choose an appropriate resolution method should be defined. How to compute them should also be established.

5. Automatic algorithm choice: This work aims to automate the choice of an algorithm. First, the problem to solve is characterized. Secondly, based on the characterization, similar benchmark problems are selected. Finally, based the scores of the algorithms of the similar set of problems, the most appropriate algorithm for the problem to solve is selected.

6. Designer's solutions introduction into solving process: The purpose of this work is to discuss how designer's solutions could be used during the resolution phase. First, a state-of-the-art of designers solutions should be made. Then a method to use them into an EDO process should be defined. Finally, the exploitation method of the designers' solutions should be tested and discussed.

Research project 2: To the 3rd gen of meta-heuristic algorithms - Mechanism based ones

This research project aspires to contribute to the emergence of a 3rd generation of meta-heuristic algorithms, which would be designed from mechanisms. The term mechanism, used by [12, 13], could be defined as a group of algorithmic operations meant to perform a particular task. For instance, the L-SHADE [58] algorithm is an evolution of the SHADE [181] algorithm including an additional mechanism reducing the population during the run [13]. A large amount of second generation meta-heuristic algorithms, which are metaphor-based ones, does not outperform the previously published ones [13]. The authors of [12] highlight the necessity of explaining and thinking algorithms in terms of mechanisms. The performance of mechanisms will be independently investigated. It is expected that, from the gained knowledge, efficient mechanisms-based algorithms could be designed. The research works composing this research project are:

1. Optimization case characterization (landscape function): This work is about how to fully characterize an optimization case. What information about an optimization case would matter in order to choose an appropriate resolution method should be defined. How to compute them should be defined.

2. Benchmark function constitution: This work goal is to correct the benchmark function diversity bias. The idea is to constitute a set of functions which represents the diversity of real-life EDO problems.

3. Algorithm mechanism classification: The goal of this work is to reference and classify the common mechanisms which could be found in widely used meta-heuristic algorithms.

4. Algorithm nomenclature: The aim of this work is to develop a method to name, reference and classify algorithms according to their mechanisms. Many algorithms have been published with a large variety of names. This could lead to incredible situations, such as one described in [12] during which a recently published algorithm happens to be particular case of an already existing algorithm published 30 years before. To avoid such a situation, and to keep a record of the existing algorithms, their names should be codified and they should be referenced in a common database. Hence, considering the evolution phenomenon described in introduction, it could be wise to base the algorithm naming on mechanisms.

5. Algorithm mechanisms investigation: This work aims to investigate the performance of the mechanisms. The idea is to test with the benchmark, mechanisms one by one and/or in combination to gain knowledge on their strengths and weaknesses.

6. Algorithms designed from mechanisms: This work is to investigate whether basing the design of algorithms on mechanisms is consistent or not. The mechanisms having the best performance will be used to design algorithms. The resulting algorithms will be tested. Based on their results, whether it is consistent to design algorithms from mechanism or not will be discussed.

Research project 3: A global radio-frequencies devices optimization engine

This research project aims to develop an optimization engine able to efficiently solve any kind of RF devices design optimization problems. This research project will generalize this thesis work to all RF devices. By doing so, the SMART-INN partners would be able to optimize other structures than SAW filters. The research works composing this research project are:

1. An optimization strategy including structural choices: By including structural choices into the optimization problem, some additional difficulties are expected. For instance, in the impedance elements filters case, a multi-level optimization method will be investigated. Thanks to this work, no matter what kind of problem are encountered, a suited optimization solution will be provided.

2. An efficient strategy for sets of optimizations: It could be interesting for SMART-INN partners to design series of optimal RF devices, with only a few specifications changing, such as the center frequency or the relative bandwidth. This could be used as a strategy to efficiently respond to calls for bids as it increases the chance of having a suitable design to suggest. For this to be possible, several optimizations with close bills of specifications have to be performed. As these optimizations are similar from one another, it might be interesting to develop techniques, akin to fordism, to industrialize the optimization process. Optimization would therefore be performed in sets such as products are manufactured by batch. Multi-objective means could be used as inspiration sources to developed this 'sets of optimization' techniques.

3. An artificial intelligence (AI) based EDO for RF devices: In the case of a type of product regularly optimized, an AI could be used to improve the optimization performance. Indeed, in this case, former results could be used to train an AI which would submit solutions. Even if not fully satisfying, they could be used by an optimization algorithm as reference solutions.

4. A specification based optimization process: This work aims to develop a fully automated process which selects an architecture depending on the specifications and optimizes it. The choice of the architecture will require to formalize the knowledge of an RF expert and will implement it as an expert-rules-based choice process.

Research project 4: To RF filter analytic optimization

This research project aims to use algorithm-based optimization as a mean to gather enough knowledge about the RF filter design to be able to perform optimization by analytic means, at least partially. The idea is to go from the algorithm-based optimization approach to an analytic optimization one (see chapter 1). The research works composing this research project are:

1. An efficient strategy for a sets of optimizations: It could be interesting for SMART-INN partners to design series of optimal RF devices, with only a few specifications changing, such as the center frequency or the relative bandwidth. This could be used as a strategy to efficiently respond to calls for bids as it increases the chance of having a suitable design to suggest. For this to be possible, several optimizations with close bills of specifications have to be performed. As these optimizations are similar from one another, it might be interesting to develop techniques, akin to fordism, to industrialize the optimization process. Optimization would therefore be performed in sets such as products are manufactured by batch. Multi-objective means could be used as inspiration sources to developed this 'sets of optimization' techniques.

2. A method to analyze the results of a set of optimizations: This work aims to develop a method to analyze the results of a set of optimizations in order to expose rules which could be used for analytic optimization. It is wished to find a method exploring the relations between the variation of the bills of specification of the optimizations and the variables ones. The gist is to observe how a change in a bill of specification influences optimal design parameters.

3. From optimization set results to analytic optimization: Batteries of optimizations will be performed in order to investigate possible relationships between the specifications and the optimal design parameters. If such links are found they could be used to create expert-rules to design filters with optimal specifications. In this case, additional investigations could be achieved to find physical explanations for such rules.

References

Bibliography

[1] Eric Mounier. Status of the mems industry 2018, May 2018.

[2] Loic Braun. *Composants à ondes elastiques de surface pour le filtrage à gabarits maîtrises aux fréquences radios pour applications spatiales et professionnelles*. PhD thesis, Université de Franche-Comté,, 2015.

[3] Jean-Pierre Tanguy. *Théorie et pratique du signal: Signaux déterministes et aléatoires en continu et en discret*. Ellipses, 2007.

[4] Gérard Mangiante. *Analyse et synthèse des filtres actifs analogiques*. Lavoisier (Paris), 2005.

[5] Walter Appel and Emmanuel Kowalski. *Mathematics for physics and physicists*. Princeton University Press Princeton, NJ, USA; Oxford, UK, 2007.

[6] R. Aigner. Saw and baw technologies for rf filter applications: A review of the relative strengths and weaknesses. In *2008 IEEE Ultrasonics Symposium*, pages 582–589, Nov 2008.

[7] RM White and FW Voltmer. Direct piezoelectric coupling to surface elastic waves. *Applied physics letters*, 7(12):314–316, 1965.

[8] Rajkumar Roy, Srichand Hinduja, and Roberto Teti. Recent advances in engineering design optimisation: Challenges and future trends. *CIRP Annals*, 57(2):697 – 715, 2008.

[9] Anh-Tuan Nguyen, Sigrid Reiter, and Philippe Rigo. A review on simulation-based optimization methods applied to building performance analysis. *Applied Energy*, 113(Supplement C):1043 – 1058, 2014.

[10] Joseph Louis Lagrange. *Traité de la résolution des équations numériques de tous les degrés: avec des notes sur plusieurs points de la théorie des équations algébriques*. Chez Courcier, 1806.

[11] George B Dantzig. Maximization of a linear function of variables subject to linear inequalities, in activity analysis of production and allocation. 1951.

[12] Kenneth Sörensen. Metaheuristics - the metaphor exposed. *International Transactions in Operational Research*, 22(1):3–18, 2012.

[13] Carlos García-Martínez, Pablo D. Gutiérrez, Daniel Molina, Manuel Lozano, and Francisco Herrera. Since cec 2005 competition on real-parameter optimisation: a decade of research, progress and comparative analysis's weakness. *Soft Computing*, 21(19):5573–5583, October 2017.

[14] Ponnuthurai N Suganthan, Nikolaus Hansen, Jing J Liang, Kalyanmoy Deb, Ying-Ping Chen, Anne Auger, and Santosh Tiwari. Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. *KanGAL report*, 2005005:2005, 2005.

[15] Nikolaus Hansen, Steffen Finck, Raymond Ros, and Anne Auger. Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions. Research Report RR-6829, INRIA, 2009.

[16] Fred Glover. Heuristics for integer programming using surrogate constraints. *Decision sciences*, 8(1):156–166, 1977.

[17] Marjorie Barcella Charrier. *Ergonomics and design in a product design approach centered on the needs of persons.* Theses, Université de Technologie de Belfort-Montbeliard, November 2016.

[18] Steven D Eppinger and Karl T Ulrich. Product design and development.

[19] T. Weise, R. Chiong, J. Lassig, K. Tang, S. Tsutsui, W. Chen, Z. Michalewicz, and X. Yao. Benchmarking optimization algorithms: An open source framework for the traveling salesman problem. *IEEE Computational Intelligence Magazine*, 2014.

[20] Zbigniew Michalewicz. Ubiquity symposium: evolutionary computation and the processes of life: the emperor is naked: evolutionary algorithms for real-world applications. *Ubiquity*, 2012(November):3, 2012.

[21] Arnaud Collignan. *Méthode d'optimisation et d'aide à la décision en conception mécanique : application à une structure aéronautique.* PhD thesis, 2011.

[22] Carlo Poloni, Andrea Giurgevich, Luka Onesti, and Valentino Pediroda. Hybridization of a multi-objective genetic algorithm, a neural network and a classical optimizer for a complex design problem in fluid dynamics. *Computer Methods in Applied Mechanics and Engineering*, 186(2):403 – 420, 2000.

[23] Yeh-Liang Hsu, Shu-Gen Wang, and Chia-Chieh Yu. A sequential approximation method using neural networks for engineering design optimization problems. *Engineering Optimization*, 35(5):489–511, 2003.

[24] Sangmun Shin and Byung Rae Cho. Bias-specified robust design optimization and its analytical solutions. *Computers & Industrial Engineering*, 48(1):129 – 140, 2005. Selected Papers from the 31st. International Conference on Computers and Industrial Engineering.

[25] S. Sutanthavibul, E. Shragowitz, and J. B. Rosen. An analytical approach to floorplan design and optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(6):761–769, June 1991.

[26] S. Helwig, J. Branke, and S. Mostaghim. Experimental analysis of bound handling techniques in particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 17(2):259–271, April 2013.

[27] Mohammadsadegh Mobin, Seyed Mohsen Mousavi, Mohammad Komaki, and Madjid Tavana. A hybrid desirability function approach for tuning parameters in evolutionary optimization algorithms. *Measurement*, 114:417 – 427, 2018.

[28] Laurent Zimmer and Patricia Zablit. "global aircraft" pre-design based on constraint propagation and interval analysis. 06 2001.

[29] Clarisse DHAENENS. Optimisation multi-critère : approche par métaheuristiques.

[30] Chanan S. Syan and Geeta Ramsoobag. Maintenance applications of multi-criteria optimization: A review. *Reliability Engineering & System Safety*, 190:106520, 2019.

[31] Ali Esmaeel Nezhad, Mohammad Sadegh Javadi, and Ehsan Rahimi. Applying augmented e-constraint approach and lexicographic optimization to solve multi-objective hydrothermal generation scheduling considering the impacts of pumped-storage units. *International Journal of Electrical Power & Energy Systems*, 55:195 – 204, 2014.

[32] Eric Sandgren. Structural design optimization for latitude by nonlinear goal programming. *Computers & Structures*, 33(6):1395 – 1402, 1989.

[33] André A. Keller. Chapter 2 - glossary of mathematical optimization terminology. In André A. Keller, editor, *Mathematical Optimization Terminology*, pages 13 – 237. Academic Press, 2018.

[34] Miguel A. Salido. A non-binary constraint ordering heuristic for constraint satisfaction problems. *Applied Mathematics and Computation*, 198(1):280–295, April 2008.

[35] Christian Bessiere. Chapter 3 - constraint propagation. In Peter van Beek Francesca Rossi and Toby Walsh, editors, *Handbook of Constraint Programming*, volume Volume 2, pages 29–83. Elsevier, 2006.

[36] Vipin Kumar. Algorithms for constraint-satisfaction problems: A survey. *AI magazine*, 13(1):32–32, 1992.

[37] François Schott, Sébastien Salmon, Chamoret, Dominique, Thomas Baron, and Yann Meyer. A new loading problem: product size reduction. In *Congrès Français de Mécanique*, 2017.

[38] Donald R. Jones, Matthias Schonlau, and William J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, Dec 1998.

[39] Dimitri P Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

[40] Yanfeng Liu, Aimin Zhou, and Hu Zhang. Termination detection strategies in evolutionary algorithms: a survey. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1063–1070. ACM, 2018.

[41] A. P. Garden, R. W. ; Engelbrecht. Analysis and classification of optimisation benchmark functions and benchmark suites. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, 2014.

[42] Jorge Adán Fernández-Vargas, Adrián Bonilla-Petriciolet, Gade Pandu Rangaiah, and Seif-Eddeen K Fateen. Performance analysis of stopping criteria of population-based metaheuristics for global optimization in phase equilibrium calculations and modeling. *Fluid Phase Equilibria*, 427:104–125, 2016.

[43] Nikolaus Hansen, Anne Auger, Dimo Brockhoff, Dejan Tusar, and Tea Tusar. COCO: performance assessment. *CoRR*, abs/1605.03560, 2016.

[44] Adam P. Piotrowski, Maciej J. Napiorkowski, Jaroslaw J. Napiorkowski, and Pawel M. Rowinski. Swarm intelligence and evolutionary algorithms: Performance versus speed. *Information Sciences*, 384:34 – 85, 2017.

[45] Dominique Chamoret, Sébastien Salmon, Noelie Di Cesare, and Yingjie J. Xu. Bsg-starcraft radius improvements of particle swarm optimization algorithm: An application to ceramic matrix composites. In Patrick Siarry, Lhassane Idoumghar, and Julien Lepagnot, editors, *Swarm Intelligence Based Optimization*, pages 166–174, Cham, 2014. Springer International Publishing.

[46] Vahid Beiranvand, Warren Hare, and Yves Lucet. Best practices for comparing optimization algorithms. *Optimization and Engineering*, 18(4):815–848, 2017.

[47] Elizabeth D. Dolan and Jorge J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, Jan 2002.

[48] Borko Boškovic and Janez Brest. Protein folding optimization using differential evolution extended with local search and component reinitialization. *Information Sciences*, 454-455:178 – 199, 2018.

[49] R. Biedrzycki. A version of ipop-cma-es algorithm with midpoint for cec 2017 single objective bound constrained problems. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 1489–1494, June 2017.

[50] A. Auger and N. Hansen. A restart cma evolution strategy with increasing population size. In *2005 IEEE Congress on Evolutionary Computation*, volume 2, pages 1769–1776 Vol. 2, Sep. 2005.

[51] Nikolaus Hansen and Stefan Kern. Evaluating the cma evolution strategy on multimodal test functions. In Xin Yao, Edmund K. Burke, José A. Lozano, Jim Smith, Juan Julián Merelo-Guervós, John A. Bullinaria, Jonathan E. Rowe, Peter Tiňo, Ata Kabán, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature - PPSN VIII*, pages 282–291, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[52] Nikolaus Hansen. *The CMA Evolution Strategy: A Comparing Review*, pages 75–102. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[53] Hong-Ji Meng, Peng Zheng, Rong-Yang Wu, Xiao-Jing Hao, and Zhi Xie. A hybrid particle swarm algorithm with embedded chaotic search. In *IEEE Conference on Cybernetics and Intelligent Systems, 2004.*, volume 1, pages 367–371. IEEE, 2004.

[54] Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, pages 69–73, May 1998.

[55] Kenneth Alan De Jong. Analysis of the behavior of a class of genetic adaptive systems. 1975.

[56] Adel Sabry Eesa, Adnan Mohsin Abdulazeez Brifcani, and Zeynep Orman. Cuttlefish algorithm: A novel bio-inspired optimization algorithm. 2014.

[**57**] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[**58**] Ryoji Tanabe and Alex S Fukunaga. Improving the search performance of shade using linear population size reduction. In *2014 IEEE congress on evolutionary computation (CEC)*, pages 1658–1665. IEEE, 2014.

[**59**] Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. Exploration and exploitation in evolutionary algorithms: A survey. *ACM Comput. Surv.*, 45(3):35:1–35:33, July 2013.

[**60**] F. van den Bergh and A.P. Engelbrecht. A study of particle swarm optimization particle trajectories. *Information Sciences*, 176(8):937 – 971, 2006.

[**61**] Arun Khosla, Shakti Kumar, and K.K. Aggarwal. Identification of strategy parameters for particle swarm optimizer through taguchi method. *Journal of Zhejiang University-SCIENCE A*, 7(12):1989–1994, 2006.

[**62**] H. Wang, Q. Geng, and Z. Qiao. Parameter tuning of particle swarm optimization by using taguchi method and its application to motor design. In *2014 4th IEEE International Conference on Information Science and Technology*, pages 722–726, April 2014.

[**63**] Argha Das, Arindam Majumder, and Pankaj Kr. Das. Detection of apposite PSO parameters using taguchi based grey relational analysis: Optimization and implementation aspects on manufacturing related problem. *Procedia Materials Science*, 6(Complete):597–604, 2014.

[**64**] Bouchaîb Radi Abdelkhalak El Hami. *Incertitudes, optimisation et fiabilité des structures*. hermes sience, 2013.

[**65**] Christian Artigues. Optimisation discrète sous incertitudes, October 2013.

[**66**] Zhan Kang. *Robust design optimization of structures under uncertainties*. PhD thesis, University of Stuttgart, 2005.

[**67**] Noélie Di Césare. *Développement d'une nouvelle méthode metaheuristique pour l'optimisation topologique des structures et des metamatériaux*. PhD thesis, Université de Technologie de Belfort-Montbéliard, 2016.

[**68**] Thomas Quirante. *Modelling and numerical optimization methods for decision support in robust embodiment design of products and processes*. PhD thesis, 2012.

[**69**] Leila Moslemi Naeni, Regina Berretta, and Pablo Moscato. Ma-net: A reliable memetic algorithm for community detection by modularity optimization. In Hisashi Handa, Hisao Ishibuchi, Yew-Soon Ong, and Kay Chen Tan, editors, *Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems, Volume 1*, pages 311–323, Cham, 2015. Springer International Publishing.

[**70**] Jianfei Shen and Lincong Han. Design process optimization and profit calculation module development simulation analysis of financial accounting information system based on particle swarm optimization (pso). *Information Systems and e-Business Management*, Jan 2019.

[71] Carl Fredrik Rehn, Jose Jorge Garcia Agis, Stein Ove Erikstad, and Richard de Neufville. Versatility vs. retrofittability tradeoff in design of non-transport vessels. *Ocean Engineering*, 167:229 – 238, 2018.

[72] Arnaud Hubert, Pierre-Alain Yvars, Yann Meyer, and Laurent Zimmer. Conception préliminaire optimale des systèmes électriques. une approche par synthèse. In *Symposium de Genie Electrique*, 2016.

[73] Jürgen Wittmann. Customer-oriented optimization of the airplane boarding process. *Journal of Air Transport Management*, 76:31 – 39, 2019.

[74] Weiwei Cui and Lin Li. A game-theoretic approach to optimize the time-of-use pricing considering customer behaviors. *International Journal of Production Economics*, 201:75 – 88, 2018.

[75] Christos Georgoulas, Thomas Linner, and Thomas Bock. Towards a vision controlled robotic home environment. *Automation in Construction*, 39:106 – 116, 2014.

[76] Muhammad Arshad Shehzad Hassan, Minyou Chen, Houfei Lin, Mohammed Hassan Ahmed, Muhammad Zeeshan Khan, and Gohar Rehman Chughtai. Optimization modeling for dynamic price based demand response in microgrids. *Journal of Cleaner Production*, 222:231 – 241, 2019.

[77] António Gaspar-Cunha, Dirk Loyens, and Ferrie van Hattum. Aesthetic design using multi-objective evolutionary algorithms. In Ricardo H. C. Takahashi, Kalyanmoy Deb, Elizabeth F. Wanner, and Salvatore Greco, editors, *Evolutionary Multi-Criterion Optimization*, pages 374–388, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[78] Sachin Mishra and Pradeep Kumar. Aesthetic design optimization of a sustainable human cum electric powered vehicle through implication of golden ratio perspectives. In Amaresh Chakrabarti, editor, *ICoRD'15 – Research into Design Across Boundaries Volume 1*, pages 315–327, New Delhi, 2015. Springer India.

[79] Jun Yao, Shinobu Miwa, Hajime Shimada, and Shinji Tomita. A fine-grained runtime power/performance optimization method for processors with adaptive pipeline depth. *Journal of Computer Science and Technology*, 26(2):292–301, Mar 2011.

[80] Ruben Lostado, Ruben Escribano García, and Roberto Fernandez Martinez. Optimization of operating conditions for a double-row tapered roller bearing. *International Journal of Mechanics and Materials in Design*, 12(3):353–373, Sep 2016.

[81] S. Eleftheriadis, P. Duffour, B. Stephenson, and D. Mumovic. Automated specification of steel reinforcement to support the optimisation of rc floors. *Automation in Construction*, 96:366 – 377, 2018.

[82] Tahereh Korouzhdeh and Hamid Eskandari-Naddaf. Cost-safety optimization of steel-concrete composite beams using standardized formulation. *Engineering Science and Technology, an International Journal*, 22(2):523 – 532, 2019.

[83] A. A. Yershov, V. V. Kotov, and Yu. N. Loginov. Optimization of the initial form of a semifinished product in pam-stamp 2g. *Metallurgist*, 56(3):231–235, Jul 2012.

[**84**] Malgorzata Plaza, Wojciech Zebala, and Andrzej Matras. Decision system supporting optimization of machining strategy. *Computers & Industrial Engineering*, 127:21 – 38, 2019.

[**85**] Xingang Tang, David Hicham Bassir, and Weihong Zhang. Shape, sizing optimization and material selection based on mixed variables and genetic algorithm. *Optimization and Engineering*, 12(1):111–128, Mar 2011.

[**86**] Snehanshu Saha, Jyotirmoy Sarkar, Avantika Dwivedi, Nandita Dwivedi, Anand M. Narasimhamurthy, and Ranjan Roy. A novel revenue optimization model to address the operation and maintenance cost of a data center. *Journal of Cloud Computing*, 5(1):1, Jan 2016.

[**87**] Stein-Erik Fleten, Erik Haugom, Alois Pichler, and Carl J. Ullrich. Structural estimation of switching costs for peaking power plants. *European Journal of Operational Research*, 2019.

[**88**] Jixin Wang, Wanghao Shen, Zhongda Wang, Mingyao Yao, and Xiaohua Zeng. Multi-objective optimization of drive gears for power split device using surrogate models. *Journal of Mechanical Science and Technology*, 28(6):2205–2214, Jun 2014.

[**89**] Adam Chehouri, Rafic Younes, Adrian Ilinca, and Jean Perron. Review of performance optimization techniques applied to wind turbines. *Applied Energy*, 142:361 – 388, 2015.

[**90**] A. van Schaik and M. A. Reuter. The optimization of end-of-life vehicle recycling in the european union. *JOM*, 56(8):39–43, Aug 2004.

[**91**] N. A. Tyutin. The principle of recycling and optimization as applied to batch-operated thermal processing plants. *Refractories and Industrial Ceramics*, 43(3):157–160, Mar 2002.

[**92**] Yongli Wang, Xiaohai Wang, Haiyang Yu, Yujing Huang, Huanran Dong, ChengYuan Qi, and Niyigena Baptiste. Optimal design of integrated energy system considering economics, autonomy and carbon emissions. *Journal of Cleaner Production*, 225:563 – 578, 2019.

[**93**] Vedant Singh and Somesh Kumar Sharma. Fuel consumption optimization in air transport: a review, classification, critique, simple meta-analysis, and future research implications. *European Transport Research Review*, 7(2):12, Apr 2015.

[**94**] Alexandre Bertrand, Alessio Mastrucci, Nils Schüler, Riad Aggoune, and François Maréchal. Characterisation of domestic hot water end-uses for integrated urban thermal energy assessment and optimisation. *Applied Energy*, 186:152 – 166, 2017. Energy and Urban Systems.

[**95**] Miloš Madić, Miroslav Radovanović, Miodrag Manić, and Miroslav Trajanović. Optimization of ann models using different optimization methods for improving co2 laser cut quality characteristics. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 36(1):91–99, Jan 2014.

[**96**] Dong-Ming Fang, Yong Zhou, Xiang-Meng Jing, Xiao-Lin Zhao, and Xi-Ning Wang. Modeling, optimization and performance of high-q mems solenoid inductors. *Microsystem Technologies*, 14(2):185–191, Feb 2008.

[**97**] Lin Hu, Jing Huang, and Fangyi Li. Simulation and optimization of vehicle ride comfort on random road. In David Jin and Sally Lin, editors, *Advances in Mechanical and Electronic Engineering*, pages 495–501, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[**98**] Bérenger Favre and Bruno Peuportier. Using dynamic programming optimization to maintain comfort in building during summer periods. In Anne Hakansson, Mattias Höjer, Robert J. Howlett, and Lakhmi C Jain, editors, *Sustainability in Energy and Buildings*, pages 137–146, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[**99**] Yang Xu, C. Boone, and L. T. Pileggi. Metal-mask configurable rf front-end circuits. In *2004 IEE Radio Frequency Integrated Circuits (RFIC) Systems. Digest of Papers*, pages 547–550, June 2004.

[**100**] Yanira Gonzalez, Gara Miranda, and Coromoto Leon. Multi-objective multi-level filling evolutionary algorithm for the 3d cutting stock problem. *Procedia Computer Science*, 96:355–364, 2016.

[**101**] Gerhard Wascher, Heike Haussner, and Holger Schumann. An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183(3):1109–1130, December 2007.

[**102**] Timothy R. Brooks and Joaquim R.R.A. Martins. On manufacturing constraints for tow-steered composite design optimization. *Composite Structures*, 204:548 – 559, 2018.

[**103**] K.H. Jyothiprakash, J. Harshith, Abhimanyu Sharan, K.N. Seetharamu, and Y.T. Krishnegowda. Thermodynamic optimization of three-fluid cross-flow heat exchanger using ga and pso heuristics. *Thermal Science and Engineering Progress*, 11:289 – 301, 2019.

[**104**] Hamda Chagraoui, Mohamed Soula, and Mohamed Guedri. A robust multi-objective and multi-physics optimization of multi-physics behavior of microstructure. *Journal of Central South University*, 23(12):3225–3238, Dec 2016.

[**105**] Anh Tuan Nguyen. *Sustainable housing in Vietnam: Climate responsive design strategies to optimize thermal comfort.* PhD thesis, Université de Liège, Liège, Belgium, 2013.

[**106**] Amir H Gandomi and Ali R Kashani. Probabilistic evolutionary bound constraint handling for particle swarm optimization. *Operational Research*, 18(3):801–823, 2018.

[**107**] Rafa Biedrzycki, Jarosaw Arabas, and Dariusz Jagodziński. Bound constraints handling in differential evolution: An experimental study. *Swarm and Evolutionary Computation*, 2018.

[**108**] Edwin C Harrington. The desirability function. *Industrial quality control*, 21(10):494–498, 1965.

[**109**] George Derringer and Ronald Suich. Simultaneous optimization of several response variables. *Journal of quality technology*, 12(4):214–219, 1980.

[**110**] Kristin L Wood and Erik K Antonsson. Computations with imprecise parameters in engineering design: background and theory. *Journal of Mechanisms, Transmissions, and Automation in Design*, 111(4):616–625, 1989.

[**111**] D. H. Wolpert and W. G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, Apr 1997.

[**112**] R. Cheng, M. Li, Y. Tian, X. Zhang, S. Yang, Y. Jin, and X. Yao. Benchmark functions for the cec 2017 competition on many-objective optimization. Technical report, University of Birmingham, UK, 2017.

[**113**] Nikolaus Hansen, Anne Auger, Steffen Finck, and Raymond Ros. Real-parameter black-box optimization benchmarking 2010: Experimental setup. Research Report RR-7215, INRIA, 2010.

[**114**] Nikolaus Hansen, Anne Auger, Olaf Mersmann, Tea Tusar, and Dimo Brockhoff. Coco: A platform for comparing continuous optimizers in a black-box setting. ArXiv e-prints, arXiv:1603.08785, July 2016.

[**115**] Ahmed S. Tawfik, Amr A. Badr, and Ibrahim F. Abdel-Rahman. Comparative optimizer rank and score: A modern approach for performance analysis of optimization techniques. *Expert Systems with Applications*, 45(Supplement C):118 – 130, 2016.

[**116**] Wellison J.S. Gomes, André T. Beck, Rafael H. Lopez, and Leandro F.F. Miguel. A probabilistic metric for comparing metaheuristic optimization algorithms. *Structural Safety*, 70:59–70, 2018.

[**117**] Y. Huang, Y. Jin, and Y. Ding. New performance indicators for robust optimization over time. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1380–1387, May 2015.

[**118**] Thomas Quirante, Patrick Sebastian, and Yann Ledoux. A trade-off function to tackle robust design problems in engineering. *Journal of Engineering Design*, pages 64–81, 2013.

[**119**] N H Awad; M Z Ali; P N Suganthan; J J Liang; B Y Qu. Problem definitions and evaluation criteria for the cec 2017 special session and competition on single objective real-parameter numerical optimization. Technical report, Nanyang Technological University, Singapore And Jordan University of Science and Technology, Jordan And Zhengzhou University, Zhengzhou China, 2016.

[**120**] B.Y. Qu, J.J. Liang, Z.Y. Wang, Q. Chen, and P.N. Suganthan. Novel benchmark functions for continuous multimodal optimization with comparative results. *Swarm and Evolutionary Computation*, 26:23–34, 2016.

[**121**] JJ Liang, BY Qu, and PN Suganthan. Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization. 2013.

[**122**] Nikolaus Hansen, Anne Auger, Olaf Mersmann, Tea Tusar, and Dimo Brockhoff. COCO: A Platform for Comparing Continuous Optimizers in a Black-Box Setting. ArXiv e-prints, arXiv:1603.08785, July 2016.

[**123**] Michael Hellwig and Hans-Georg Beyer. Benchmarking evolutionary algorithms for single objective real-valued constrained optimization – a critical review. *Swarm and Evolutionary Computation*, 44:927 – 944, 2019.

[**124**] Q. Chen; B. Liu; Q. Zhang; J.J. Liang; P. N. Suganthan; B.Y. Qu. Problem definition and evaluation criteria for cec 2015 special session and competition on bound constrained single-objective computationally expensive numerical optimization. Technical report, Congress on Evolutionary Computation (CEC), October 2013.

[**125**] Xin-She Yang, Slawomir Koziel, and Leifur Leifsson. Computational Optimization, Modelling and Simulation: Past, Present and Future. *Procedia Computer Science*, 29(Supplement C):754 – 758, 2014.

[**126**] John H. Holland. *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA, USA, 1992.

[**127**] C. L. Muller, B. Baumgartner, and I. F. Sbalzarini. Particle swarm cma evolution strategy for the optimization of multi-funnel landscapes. In *2009 IEEE Congress on Evolutionary Computation*, pages 2685–2692, May 2009.

[**128**] Niki Vecek, Marjan Mernik, and Matej Crepinšek. A chess rating system for evolutionary algorithms: A new method for the comparison and ranking of evolutionary algorithms. *Information Sciences*, 277:656 – 679, 2014.

[**129**] Ilhem Boussaïd, Julien Lepagnot, and Patrick Siarry. A survey on optimization metaheuristics. *Information Sciences*, 237:82 – 117, 2013. Prediction, Control and Diagnosis using Advanced Neural Computations.

[**130**] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.*, 21(4):25–34, August 1987.

[**131**] Manuel López-Ibáñez, Jérémie Dubois-Lacoste, Leslie Pérez Cáceres, Mauro Birattari, and Thomas Stützle. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43 – 58, 2016.

[**132**] Ian Dewancker, Michael McCourt, Scott Clark, Patrick Hayes, Alexandra Johnson, and George Ke. A strategy for ranking optimization methods using multiple criteria. In *AutoML@ICML*, 2016.

[**133**] Marco S. Nobile, Paolo Cazzaniga, Daniela Besozzi, Riccardo Colombo, Giancarlo Mauri, and Gabriella Pasi. Fuzzy self-tuning pso: A settings-free algorithm for global optimization. *Swarm and Evolutionary Computation*, 2017.

[**134**] Robert E Mercer and JR Sampson. Adaptive search using a reproductive meta-plan. *Kybernetes*, 7(3):215–228, 1978.

[**135**] K. R. Harrison; B. M. Ombuki-Berman; A. P. Engelbrecht. Optimal parameter regions for particle swarm optimization algorithms. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, 2017.

[**136**] Riccardo Poli and David Broomhead. Exact analysis of the sampling distribution for the canonical particle swarm optimiser and its convergence during stagnation. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, GECCO '07, pages 134–141, New York, NY, USA, 2007. ACM.

[**137**] François Schott, Dominique Chamoret, Thomas Baron, Sébastien Salmon, and Yann Meyer. A new scoring strategy for meta-heuristic, single-objective, continuous optimization algorithm benchmark. Manuscript re-submitted to Applied soft computing, February 2019.

[**138**] Jacques Alexis. *Pratique industrielle de la méthode Taguchi.* Association Française de Normalisation (AFNOR), 1995.

[**139**] Genichi Taguchi, Seiso Konishi, and S Konishi. *Taguchi Methods: Orthogonal Arrays and Linear Graphs. Tools for Quality Engineering.* American Supplier Institute Dearborn, MI, 1987.

[**140**] I. T. Jolliffe. *Principal Component Analysis.* Springer, 2002.

[**141**] A. Mohsine, G. Kharmanda, and A. El-Hami. Improved hybrid method as a robust tool for reliability-based design optimization. *Structural and Multidisciplinary Optimization*, 32(3):203–213, Sep 2006.

[**142**] Wei Chen, Margaret M Wiecek, and Jinhuan Zhang. Quality utility: a compromise programming approach to robust design. *ASME DETC98/DAC5601*, 1998.

[**143**] D. Janz, W. Sihn, and H.-J. Warnecke. Product redesign using value-oriented life cycle costing. *CIRP Annals*, 54(1):9 – 12, 2005.

[**144**] Ruichen Jin, Xiaoping Du, and Wei Chen. The use of metamodeling techniques for optimization under uncertainty. *Structural and Multidisciplinary Optimization*, 25(2):99–116, 2003.

[**145**] Constantine Caramanis Dimitris Bertsimas, David B. Brown. Theory and applications of robust optimization. *SIAM REVIEW*, 2011.

[**146**] Jian Tu, Kyung K Choi, and Young H Park. A new study on reliability-based design optimization. *Journal of mechanical design*, 121(4):557–564, 1999.

[**147**] Nicholas Metropolis and Stanislaw Ulam. The monte carlo method. *Journal of the American statistical association*, 44(247):335–341, 1949.

[**148**] C Allin Cornell. A probability-based structural code. In *Journal Proceedings*, volume 66, pages 974–985, 1969.

[**149**] Abraham M Hasofer and Niels C Lind. Exact and invariant second-moment code format. *Journal of the Engineering Mechanics division*, 100(1):111–121, 1974.

[**150**] KM Aludaat and MT Alodat. A note on approximating the normal distribution function. *Applied Mathematical Sciences*, 2(9):425–429, 2008.

[**151**] S.K. Lwanga and S. Lemeshow. Détermination de la taille d'un échantillon dans les études sanométriques. Technical report, Organisation mondiale de la Sante, 1991.

[**152**] R. Mendes, J. Kennedy, and J. Neves. Watch thy neighbor or how the swarm can learn from its environment. In *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No.03EX706)*, pages 88–94, April 2003.

[**153**] Steffen Finck, Nikolaus Hansen, Raymond Ros, and Anne Auger. Real-parameter black-box optimization benchmarking 2010: Presentation of the noisy functions. Technical report, Citeseer, 2010.

[**154**] Andrew M Sutton, Darrell Whitley, Monte Lunacek, and Adele Howe. Pso and multi-funnel landscapes: how cooperation might limit exploration. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 75–82. ACM, 2006.

[**155**] Vesselin K Vassilev, Terence C Fogarty, and Julian F Miller. Smoothness, ruggedness and neutrality of fitness landscapes: from theory to application. In *Advances in evolutionary computing*, pages 3–44. Springer, 2003.

[**156**] Ioan Cristian Trelea. The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters*, 85(6):317 – 325, 2003.

[**157**] Sébastien Salmon. *Caractérisation, identification et optimisation des systèmes mécaniques complexes par mise en oeuvre de simulateurs hybrides matériels/logiciels.* PhD thesis, Belfort-Montbéliard, 2012.

[**158**] P. Nintanavongsa, U. Muncuk, D. R. Lewis, and K. R. Chowdhury. Design optimization and implementation for rf energy harvesting circuits. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2(1):24–33, March 2012.

[**159**] J. M. Lopez-Villegas, J. Samitier, C. Cane, P. Losantos, and J. Bausells. Improvement of the quality factor of rf integrated inductors by layout optimization. *IEEE Transactions on Microwave Theory and Techniques*, 48(1):76–83, Jan 2000.

[**160**] S. H. Yeung, W. S. Chan, K. T. Ng, and K. F. Man. Computational optimization algorithms for antennas and rf/microwave circuit designs: An overview. *IEEE Transactions on Industrial Informatics*, 8(2):216–227, May 2012.

[**161**] Yong Pang and Gary X. Shen. Improving excitation and inversion accuracy by optimized rf pulse using genetic algorithm. *Journal of Magnetic Resonance*, 186(1):86 – 93, 2007.

[**162**] Jan Tichỳ, Jirí Erhart, Erwin Kittinger, and Jana Privratska. *Fundamentals of piezoelectric sensorics: mechanical, dielectric, and thermodynamical properties of piezoelectric materials.* Springer Science & Business Media, 2010.

[**163**] D. C. Malocha. Evolution of the saw transducer for communication systems. In *IEEE Ultrasonics Symposium, 2004*, volume 1, pages 302–310 Vol.1, Aug 2004.

[**164**] Eugène Dieulesaint. Ondes élastiques dans les solides. 1970.

[**165**] C Rosen, Basavaraj V Hiremath, and Robert Newnham. *Piezoelectricity.* Number 5. Springer Science & Business Media, 1992.

[**166**] KA Ingebrigtsen. Surface waves in piezoelectrics. *Journal of Applied Physics*, 40(7):2681–2686, 1969.

[**167**] Eugène Dieulesaint and Daniel Royer. *Ondes élastiques dans les solides-Tome 1: Propagation libre et guidée.* Paris: Masson, 1996.

[**168**] BK Sinha and HF Tiersten. On the temperature dependence of the velocity of surface waves in quartz. *Journal of Applied Physics*, 51(9):4659–4665, 1980.

[**169**] RH Tancrell and MG Holland. Acoustic surface wave filters. *Proceedings of the IEEE*, 59(3):393–409, 1971.

[**170**] VP Plessky. Saw impedance elements. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 42(5):870–875, 1995.

[**171**] M Ueda, O Kawachi, K Hashimoto, O Ikata, and Y Satoh. Low loss ladder type saw filter in the range of 300 to 400 mhz. In *1994 Proceedings of IEEE Ultrasonics Symposium*, volume 1, pages 143–146. IEEE, 1994.

[**172**] L. Braun, O. Franquet, W. Daniau, T. Baron, E. Courjon, and S. Ballandras. Effects of a plasma etching process on a longitudinally coupled resonator filter. In *2014 European Frequency and Time Forum (EFTF)*, pages 9–11, June 2014.

[**173**] T. Morita, Y. Watanabe, M. Tanaka, and Y. Nakazawa. Wideband low loss double mode saw filters. In *IEEE 1992 Ultrasonics Symposium Proceedings*, pages 95–104 vol.1, Oct 1992.

[**174**] C. C. W. Ruppel. Acoustic wave filter technology–a review. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 64(9):1390–1400, Sep. 2017.

[**175**] David Morgan. *Surface acoustic wave filters: With applications to electronic communications and signal processing*. Academic Press, 2010.

[**176**] Vincent Laude and Sylvain Ballandras. Slowness curves and characteristics of surface acoustic waves propagating obliquely in periodic finite-thickness electrode gratings. *Journal of applied physics*, 94(2):1235–1242, 2003.

[**177**] Benjamin P Abbott and Leland Solie. A minimal diffraction cut of quartz for high performance saw filters. In *2000 IEEE Ultrasonics Symposium. Proceedings. An International Symposium (Cat. No. 00CH37121)*, volume 1, pages 235–240. IEEE, 2000.

[**178**] J.C. sabonnadière. *Techniques de fabrication des microsystèmes 1: Structures et microsystèmes électromécaniques en couches minces*. Hermes Science Publications, 2004.

[**179**] M de Labachelerie et al. Techniques de fabrication des microsystèmes 1: structures et microsystèmes électromécaniques en couches minces. *Edition Lavoisier*, 2004.

[**180**] G Tobolka, W Faber, G Albrecht, and D Pilz. High volume tv-if filter design, fabrication, and applications. In *IEEE 1984 Ultrasonics Symposium*, pages 1–12. IEEE, 1984.

[**181**] Ryoji Tanabe and Alex Fukunaga. Success-history based parameter adaptation for differential evolution. In *2013 IEEE congress on evolutionary computation*, pages 71–78. IEEE, 2013.

[**182**] Frédéric Héliodore, Amir Nakib, Boussaad Ismail, Salma Ouchraa, and Laurent Schmitt. *Métaheuristiques pour les réseaux électriques intelligents*, volume 3. ISTE editions, 2017.

[**183**] Lyle Pursell and SY Trimble. Gram-schmidt orthogonalization by gauss elimination. *The American Mathematical Monthly*, 98(6):544–549, 1991.

[**184**] INRIA. The cma evolution strategy, December 2016.

[**185**] I. Loshchilov. Cma-es with restarts for solving cec 2013 benchmark problems. In *2013 IEEE Congress on Evolutionary Computation*, pages 369–376, June 2013.

[**186**] Darrell Whitley. A genetic algorithm tutorial. *Statistics and Computing*, 4(2):65–85, Jun 1994.

[**187**] Sylvain Thouviot. *Multi-objective optimization of a mechatronic system considering vibro-acoustic phenomena*. Theses, Ecole Centrale Paris, February 2013.

[**188**] Michael D McKay, Richard J Beckman, and William J Conover. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.

[**189**] Nicolas Piegay. *Optimisation multi-objectif et aide à la déecision pour la conception robuste. : Application à une structure industrielle sur fondations superficielles*. PhD thesis, Université de Bordeaux, 2015.

List of Figures

List of Tables

A

Appendix

Contents

A.1/ Thematics board

This section presents the thematics and sub-thematics used by the formulation methodology. Thematics have been depicted and gathered on a board. This board, the 'thematics board', is used during interviews to guide both the designer and the optimizer. The thematics board, in figure A.1, is composed of main thematics and sub-thematics. Each sub-thematics is linked to a 'mother' main thematics. The product is depicted in the board center. The main thematics are gathered close to the product, while sub-thematics are revolving around in the same direction as their mother thematics. Every item of the board is labeled to guide designer in case of icone misunderstanding. The thematic board is presented in its initial language (french) to avoid poor translation misunderstanding.

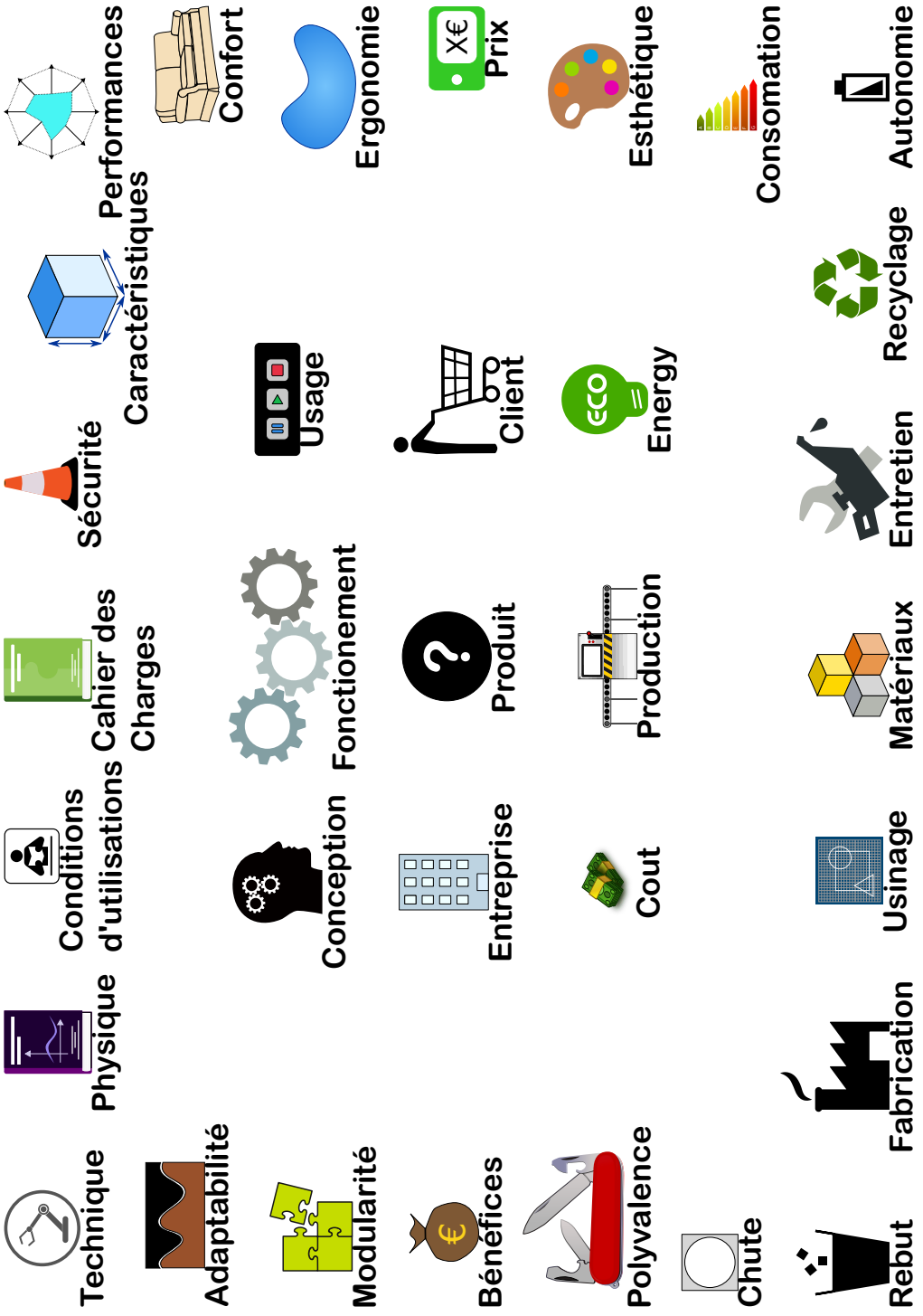


Figure A.1: Thematics board

A.2/ Questionnaire

The questionnaire is presented in its initial language (french) to avoid poor translation misunderstanding.

A.2.1/ Part 1: Learn about the case

A.2.1.1/ Entreprise

- Quelles sont les activités principales de votre entreprise ?
- Quelle est votre position sur le marché ?
- Où êtes-vous implanté ?
- Combien d'employés ?
- Quel est votre chiffre d'affaires ? À combien s'élèvent vos bénéfices ?
- Quel département/structure/équipe est en charge de la conception de ce produit ?
- Quelle est la taille de ce département/structure/équipe ?
- Avez-vous une charge de travail importante (en ce moment) ?
- Quel est le délai de mise sur le marché d'un nouveau produit ?
- Combien de temps avez-vous pour développer le produit ?

A.2.1.2/ Produit

- À quoi sert ce produit ?
- Est-ce que la gamme de produits dans laquelle s'inscrit ce produit est petite ou grande ?
- Est-ce un produit multiusage ?
- Est-ce un produit stratégique ? Pourquoi ?
- Quelles sont les tendances actuelles du marché ?
- Est-ce que votre entreprise à des attentes particulières pour ce produit ?

A.2.1.3/ Client

- Qui achète ce produit ? Quelle est la clientèle cible ?
- Comment votre client utilise-t-il votre produit ? Y a-t-il des usages détournés ?
- Ce produit fait-il partie d'une série ?
- Quels sont les prix du marché pour ce produit ?
- Qui vend ce genre de produit ?
- Quels sont les principaux concurrents (produits et entreprise) ?
- Qu'est-ce que le client trouve attractif dans ce produit ?
- Qu'est-ce qui fait que ce produit est facile d'usage ?

A.2.1.4/ Fonctionement

- Comment fonctionne le produit ?
- Quelles sont les différentes parties de l'architecture ?
- Quels sont les principaux techniques utilisés ?
- Quels sont les principes physiques mis en jeu ?
- Quelles sont les consignes d'utilisation ?
- Quels sont les sources de dysfonctionnement ?
- Y a-t-il un cahier des charges ?

A.2.1.5/ Production

- Comment produisez-vous ce produit ?
- Quelles sont les différentes étapes de la production ?
- Quels matériaux sont utilisés ?
- Quels sont les machines et outils utilisés pour la production ?
- Est-ce que le produit nécessite un entretien particulier ?
- Le produit est-il facile à réparer ?

A.2.1.6/ Impacte énergétique et environmental

- Quel est le cycle de vie du produit ?
- Quel type d'énergie utilise le produit ?
- Combien consomme-t-il ?
- Quel déchet le produit génère-til ?
- Quelle quantité de déchets émet-il ?
- Quelle est l'autonomie du produit ?
- Est-ce que la fabrication du produit consomme beaucoup d'énergie et de ressource ?
- Est-il possible de recycler le produit ?

A.2.1.7/ Usage

- Comment le client se sert-il du produit ? Quelle interface utilise-t-il pour se servir du produit ?
- Le produit est-il facile à utiliser ? Son usage demande-t-il des compétences particulières .
- Est-il confortable d'utiliser le produit ?
- Quelles sont les performances techniques du produit ?
- À quel point les produits compétiteurs sont-ils performants ?
- Quelles sont les principales caractéristiques du produit ?

A.2.1.8/ Couts

- Combien coûte la production de ce produit ?
- Qu'est-ce qui coute cher dans la production ? La masse salariale, les installations, les ressources, l'énergie ?
- Durant la fabrication, y a-t-il des opérations critiques ?
- Durant la fabrication, y a-t-il des opérations couteuses ?
- Y a-t-il un fort taux de rebut ?
- Y a-t-il beaucoup de chute et de déchets ?

A.2.1.9/ Conception

- Comment ce produit est-il conçut ?
- Utilisez-vous des simulations, des modèles ou des logiciels ?
- Comment faite vous ces calculs numériques ? Avez-vous besoin de données/informations en particulier ?
- Quelles sont les données que vous calculez ?
- Comment traitez-vous ces données ?
- Quelles sont les performances que vous calculez ?
- Comment les calculez-vous ?
- Quels sont les domaines scientifiques qui interviennent dans la conception ?
- Quels équations/lois/modèles physiques utilisez-vous ?
- Quelles technologies sont utilisées par ce produit ?
- Est-ce que la conception du produit doit être adaptée pour différents usages/situations/environement ?

A.2.2/ Part 2: Optimization element - identification

A.2.2.1/ Objectif

Qu'est-ce qui doit être le mieux possible ?

- Entreprise : Le produit phare de la marque ?
  - Le plus gros bénéfice ?
  - Le plus polyvalent ?
- Client : Le produit le plus attractif ?
  - Le produit le moins cher ?
  - Le plus facile d'utilisation ?
  - Le plus beau produit ?
- Fonctionement : Le produit qui fonctionne le mieux ?
  - Celui qui respecte le cahier des charge le plus exigeant ?
  - Les conditions d'utilisation les plus larges ?
  - Le produit le plus sécuritaire ?
- Production : Le plus facile à produire ?
  - Les meilleurs matériaux (cout, facilité à travailler) ?
  - Le plus facile à usiner ?
  - Le produit avec le moins de maintenace/entretien ?
- Energie : Le produit avec la meilleure efficience énergétique ?
  - La consommation la plus basse ?
  - La meilleure autonomie ?
  - Le produit le plus éco-responsable ?
- Usage : Le produit le plus efficient ?
  - Le meilleur dimensionnement ?
  - Les meilleures performances techniques ?
  - Le plus confortable ?
- Cout : Le moins cher à produire ?
  - Celui qui demande le moins d'investissement ?
  - Celui avec le taux de rebut le moins élevé ?
  - Celui qu génère le moins de chutes/déchet ?
- Conception : Le produit le mieux conçu ?
  - Celui qui a les meilleures propriétés physiques ?
  - Celui qui exploite les meilleures technologies ?
  - Celui qui s'adapte le mieux (environement, utilisation) ?

A.2.2.2/ Contraintes

Qu'est-ce qui doit être suffisamment bon pour que le produit soit accepté ?

- Entreprise : Le produit doit répondre à certaines attentes ?
  - Le produit doit dégager assez de bénéfices ?
  - Le produit doit s'inscrire dans au moins x gammes de produit ?
  - Le produit doit être suffisament facile à décliner ?
- Client : Le produit doit être suffisament attractif ?
  - Le produit doit être accessible financièrement ?
  - Le produit doit être suffisament facile d'utilisation ?
  - Le produit doit être aggréable à regarder ?
- Fonctionement : Le produit doit respecter certaines conditions ?
  - Le produit doit respecter certains points du cahier des charges ?
  - Le produit doit pouvoir être utilisé dans différentes situations ?
  - Le produit doit pouvoir être utilisé en toute sécurité ?
- Production : Le produit doit pouvoir être produit avec un équipement standard ?
  - Les matériaux doivent respecter des critères ?
  - Les opérations de maintenance ne doivent pas être trop contraignantes ?
  - La production doit respecter certains standards ?
- Energie : Le produit doit être éco-responsable ?
  - Le produit doit avoir une consommation maitrisée ?
  - Le produit doit avoir une autonomie suffisante ?
  - Le produit doit être recyclable, au moins partiellement ?
- Usage : Le produit doit être agréable/facile à utiliser ?
  - Les caractéristiques techniques du produit doivent respecter des contraintes ?
  - Les performances techniques du produit doivent être suffisamment élevées ?
  - Le produit doit être suffisamment confortable ?
- Cout : Le produit ne doit pas être trop cher à produire ?
  - Le cout des installations nécessaires à la production doit être maitrisé ?
  - Les émissions de déchets lors de la fabrication doivent être maitrisées ?
  - Le pourcentage de rebut doit être inférieur à un seuil ?
- Conception : Lors de la conception du produit certaines restrictions doivent être respectées ?
  - Certaines règles/contraintes physiques doivent être respectées ?
  - Certaines technologies utilisées doivent respecter des contraintes ?
  - La conception doit permettre au produit de s'adapter à différentes situations, utilisations et aux aléas ?

A.2.2.3/ Variables

Qu'est ce que vous réglez/ajustez ? Sur quels aspects travaillez/jouez vous ? :

- Entreprise : Quels sont les éléments de votre produit que vous achetez aux prés de fournisseurs ?
- Client : Quelles sont les formes de votre produit que vous pouvez modifier ?
- Fonctionement : Quels sont les éléments de votre architecture que vous pouvez modifier/régler ?
- Production : Quels sont les matériaux et géométries que vous pouvez modifier ?
- Energie : Quel sont les éléments liés à la source d'énergie sur lesquels vous pouvez agir ?
- Usage : Quelles sont les caractéristiques techniques de votre produit que vous pouvez régler ?
- Cout : Quels sont les éléments de votre processus de fabrication sur lesquels vous pouvez agir pour faire baisser les couts ?
- Conception : Quelles parties de la conception peuvent être changées ?

A.2.3/ Part 3: Optimization element – Definition

A.2.3.1/ Objectifs

Table A.1: Définition des objectifs

Information	Question
Description	Qu'est-ce que vous cherchez à améliorer ?
Nom	Comment appelez-vous cette grandeur ?
Notation	Quelle notation mathématique utilisez-vous pour cette grandeur ?
Unité	En quelle unité est-ce exprimé ?
Type	Est-ce que vous en voulez le plus ou le moins possible ?
Acceptabilité	Pour quelle valeur serez-vous raisonnablement satisfait du résultat ?
Satisfaction correspondance 1	A quel point trouvez-vous la valeur [valeur compétiteur 1] bonne ou mauvaise ?
Satisfaction correspondance 2	A quel point trouvez-vous la valeur [valeur compétiteur 2] bonne ou mauvaise ?

A.2.3.2/ Contraintes

Table A.2: Définition des contraintes

Information	Question
Description	Qu'est-ce que vous cherchez à maitriser ?
Nom	Comment appelez-vous cette grandeur ?
Notation	Quelle notation mathématique utilisez-vous pour cette grandeur ?
Unité	En quelle unité est-ce exprimé ?
Limite inférieure	À partir de quelle valeur êtes-vous pleinement satisfait ?
Limite inférieure marge	À partir de quelle valeur seriez-vous prêt à faire un compromis ?
Limite supérieure	En dessous de quelle valeur êtes-vous pleinement satisfait ?
Limite inférieure marge	En dessous de quelle valeur seriez-vous prêt à faire un compromis ?

A.2.3.3/ Variables

Table A.3: Définition des variables

Information	Question
Description	Qu'est-ce que vous réglez ?
Nom	Comment appelez-vous cette grandeur ?
Notation	Quelle notation mathématique utilisez-vous pour cette grandeur ?
Unité	En quelle unité est-ce exprimé ?
Type	Est-ce que vous pouvez donner à cette grandeur toutes les valeurs d'un intervalle ou seulement des valeurs dans un ensemble donné ?
Intervalle	Entre quelles valeurs pouvez-vous ajuster [Nom] ?
Gamme	Quelles sont les valeurs que peut prendre [Name] ?
Précision	Quelle est la précision numérique avec laquelle vous recherchez la valeur de [Nom] ?

A.3/ Optimization bill of specifications

Table A.4: OBS Objectives

#	Min/Max	Description	Notations	Unit	Acceptability

Table A.5: OBS Constraints

#	Description	Expression

Table A.6: OBS Variables

#	Notations	Description	Type	Values	Unit	Accuracy	Uncertainty

Table A.7: OBS Evaluation Tools

#	Tool	Description	Inputs	Outputs

A.4/ Bound handling techniques

This section presents some bound handling techniques (BHT) that could be used with the normalized evaluations (NE) approach. These BHT are inspired by [26, 107], which presented BHT methods for PSO and DE algorithms. To understand BHT equations, a few notations, given in table A.8, should be introduced.

Table A.8: Notation for bound handling techniques

Notation	Definition
D	Dimension of the optimization problem
z	Normalized variables vector
y	Bounded variables vector
s	Satisfaction
U	Random function (Uniform distribution law)

The list of BHT is the following:

- Penalty techniques:
  - Death penalty [107] (infinity method [26]): The solution is given a negative infinite satisfaction. In practice, due to numerical restriction infinity will be replace by an arbitrary large value.

$$s(x) = -\infty \tag{A.1}$$

  - Smooth penalty (substitution, quadratic, function penalty [107]): The solution satisfaction will be equal to penalty (substitution). This penalty will be computed by a function (function penalty). This function is the square (quadratic) of the distance between the solution and the nearest search space border.

$$\begin{cases} s(x) = -\frac{1}{D} \cdot \sum_{i=1}^{D} \beta_i^2 \\ \begin{cases} \alpha_i \leq 1 \Rightarrow \beta_i = 0 \\ \alpha_i \geq 1 \Rightarrow \beta_i = \alpha_i \end{cases} \\ \alpha_i = 0.5 + |z_i - 0.5| \end{cases} \tag{A.2}$$

- Repair techniques:
  - Teleportation (re-sampling [107]): The solution is randomly relocate into the search space

$$\forall z_i, y_i \sim U([0,1]) \tag{A.3}$$

  - Shift (re-initialization [107], equivalent to random forth[26]): The coordinate of the solution which are outside of the search space are randomly set between bounds.

$$\begin{cases} z_i \notin [0,1] \Rightarrow y_i \sim U([0,1]) \\ z_i \in [0,1] \Rightarrow y_i = z_i \end{cases} \tag{A.4}$$

– Wall (projection [107]): The solution is moved to the closest edge of the search space.

$$\begin{cases} z_i \geq 1 \Rightarrow y_i = 1 \\ z_i \leq 0 \Rightarrow y_i = 0 \\ z_i \in [0,1] \Rightarrow y_i = z_i \end{cases} \tag{A.5}$$

– Bounce (reflexion [107]): The solution is moved in the search space center direction proportionally to how far from the edge it is.

$$\begin{cases} z_i \leq 0 \Rightarrow y_i = ((-z_i) mod 1) \\ z_i \geq 1 \Rightarrow y_i = 1 - ((z_i - 1) mod 1) \\ z_i \in [0,1] \Rightarrow y_i = z_i \end{cases} \tag{A.6}$$

– Round (wrapping [107]): The search space is rounded by correcting the solution position using a modulo.

$$\begin{cases} z_i \leq 0 \Rightarrow y_i = 1 - ((-z_i) mod 1) \\ z_i \geq 1 \Rightarrow y_i = ((z_i - 1) mod 1) \\ z_i \in [0,1] \Rightarrow y_i = z_i \end{cases} \tag{A.7}$$

A.5/ Functions details

A.5.1/ Basic functions

Basic functions are support to make benchmark functions. Most of them are currently used in literature [182].

- Bent cigar function

$$f_1(x) = x_1^2 + 10^6 \cdot \sum_{i=2}^{D} x_i^2 \tag{A.8}$$

- Discuss function

$$f_2(x) = 10^6 \cdot x_1^2 + \sum_{i=2}^{D} x_i^2 \tag{A.9}$$

- Weierstrass function

$$\begin{cases} f_3(x) = \sum_{i=1}^{D} (\sum_{k=0}^{kmax} [a^k cos(2\pi b^k (x_i + 0.5))]) \\ -D \sum_{k=0}^{kmax} [a^k cos(2\pi b^k \cdot 0.5)] \\ a = 0.5; b = 3; kmax = 20 \end{cases} \tag{A.10}$$

- Modified Schwefel function

$$\begin{cases} f_4(x) = 418.9829 \cdot D - \sum_{i=1}^{D} g(z_i) \\ z_i = x_i + 420.9687462275036 \\ c = 500; b = 10000 \\ g(z_i) : \begin{cases} |z_i| \le c : z_i sin(\sqrt{|z_i|}) \\ z_i > c : (c - z_i \% c) sin(\sqrt{|c - z_i \% c|}) - \frac{z_i - c}{b \cdot D} \\ z_i < c : (-c + z_i \% c) sin(\sqrt{|-c + z_i \% c|}) - \frac{z_i + c}{b \cdot D} \end{cases} \end{cases} \tag{A.11}$$

- Katsuura function

$$f_5(x) = \frac{10}{D^2} \prod_{i=1}^{D} (1 + i \sum_{j=1}^{32} \frac{2^j x_i - round(2^j x_i)}{2^j})^{\frac{10}{D^{1.2}}} - \frac{10}{D^2} \tag{A.12}$$

- HappyCat function

$$f_6(x) = |\sum_{i=1}^{D} x_i^2 - D|^{1/4} + (0.5 \sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i)/D + 0.5 \tag{A.13}$$

- HGBat function

$$f_6(x) = |(\sum_{i=1}^{D} x_i^2)^2 - (\sum_{i=1}^{D} x_i)^2|^{1/2} + (0.5 \sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i)/D + 0.5 \tag{A.14}$$

- Expanded extended Griewank plus Rosenbrock function

$$f_8(x) = \sum_{i=1}^{D} f_{11}(f_{10}(x_i, x_{(i+1)\%D})) \tag{A.15}$$

- Expanded Schaffer function

$$\begin{cases} g(x,y) = 0.5 + \frac{sin^2(\sqrt{x^2+y^2}-0.5)}{1+0.001(x^2+y^2)} \\ f_9(x) = \sum_{i=1}^{D} g(x_i, x_{(i+1)\%D}) \end{cases} \tag{A.16}$$

- Rosenbrock function

$$f_{10}(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2) \tag{A.17}$$

- Griewank function

$$f_{11}(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} cos(\frac{x_i}{\sqrt{i}}) + 1 \tag{A.18}$$

- Rastigin function

$$f_{12}(x) = \sum_{i=1}^{D} (x_i^2 - 10cos(2\pi x_i) + 10) \tag{A.19}$$

- High conditioned elliptic function

$$f_{13}(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2 \tag{A.20}$$

- Ackley function

$$f_{14}(x) = -20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - exp(\frac{1}{D}\sum_{i=1}^{D} cos(2\pi x_i)) + 20 + e \tag{A.21}$$

A.5.2/ Uni-modal and multi-modal functions

In the benchmark presented in chapter 4, 2 uni-modal and 7 multi-modal function are used. The functions are based on the simple functions introduced in the previous section. They follow the pattern of equation A.22. This equation relies on some elements such as the offset. The elements composing the simple functions are detailed in table A.9. In this table, $\overline{\Phi}$ refers to a normed random vector and Φ to a randomly generated rotation matrix. Φ matrices are geenreated using the Gram-Schmidt process [183].

$$F_i(x) = f(M(c(x - s)) + p) + o \quad where \quad i = 1, \cdots, 9 \tag{A.22}$$

- $f(x)$: Basic function used
- M: Rotation matrix

- c: Scaling factor
- s: Shift vector
- p: Inside function offset
- o: Offset

Table A.9: Uni-modal and multi-modal functions

F_i	f	s	c	M	p	o
1	1	$\overline{\Phi}$	1	Φ	0	100
2	2	$\overline{\Phi}$	1	Φ	0	200
3	3	$\overline{\Phi}$	0.005	Φ	0	300
4	4	$\overline{\Phi}$	10	Φ	0	400
5	5	$\overline{\Phi}$	0.05	Φ	0	500
6	6	$\overline{\Phi}$	0.05	Φ	0	600
7	7	$\overline{\Phi}$	0.05	Φ	0	700
8	8	$\overline{\Phi}$	0.05	Φ	1	800
9	9	$\overline{\Phi}$	1	Φ	1	900

A.5.3/ Hybrid functions

In the benchmark presented in chapter 4, 3 hybrid functions are used. These functions are based on the basic functions introduced in section A.5.1. The hybrids functions follow the pattern of equation A.23 which relies on elements, such as a set of basic functions. Those elements are detailed in table A.10.

$$\begin{cases} F = (\sum_{i=1}^{N} g_i(M_i \cdot z_i)) + o \\ z = [z_i, ..., z_n] \\ z_i = [S_a, ..., S_b] \\ a = \sum_{j=1}^{j=i-1} (n_j) + 1; b = \sum_{j=1}^{j=i} (n_j) + 1 \\ S = \text{random-permutation}(y) \\ y = x - s \\ n_i = p_i \cdot D \\ \sum_{i=1}^{N} n_i = D \\ s = \overline{\Phi} \\ M_i = \Phi(n_i, n_i) \end{cases} \tag{A.23}$$

- N: Number of basic functions used
- $g_i(x)$: i-th basic function used
- p_i: Percentage of variable allocated to the i-th basic function
- n_i: Dimension of the i-th basic function

Table A.10: Hybrid functions elements

F_i	N	o	p	g
10	3	1000	[0.3, 0.3, 0.4]	[4, 12, 13]
11	4	1100	[0.2, 0.2, 0.3, 0.3]	[11, 3, 10, 9]
12	5	1200	[0.1, 0.2, 0.2, 0.2, 0.3]	[5, 6, 8, 4, 14]

A.5.4/ Composite functions

In the benchmark presented in chapter 4, 3 composite functions are used. Composite functions follow the pattern of equation A.24 which relies on elements, such as a set of origin shifts. Those elements are detailed in table A.11.

$$\begin{cases} F(x) = \sum_{i=1}^{N} [\omega_i(\lambda_i g_i(x) + o'_i)] + o \\ w_i = \frac{1}{\sqrt{\sum_{j=1}^{D} (x_j - s_{i,j})^2}} exp(-\frac{\sum_{j=1}^{D} (x_j - s_{i,j})^2}{2D\sigma_i^2}) \\ \omega_i = w_i / \sum_{i=1}^{n} w_i \\ o'_i = 100(i-1) \end{cases} \tag{A.24}$$

- N: Number of basic functions used
- $g_i(x)$: i-th basic function used
- s_i: Shift of the i-th basic function
- o'_i: Offset of the i-th basic function
- σ_i: Control factor of the i-th basic function
- λ_i: Scale factor of the i-th basic function
- w_i: Weight of the i-th function

Table A.11: Composite functions elements

F_i	N	σ	λ	g
13	5	$[1, 2, 3, 4, 5] \times 10$	[1, 1e−6, 1e−26, 1e−6, 1e−6]	[10, 13, 1, 2, 13]
14	3	$[1, 3, 5] \times 10$	[0.25, 1, 1e−7]	[4, 12, 13]
15	5	$[1, 1, 1, 2, 2] \times 10$	[10, 10, 2.5, 25, 1e−6]	[7, 12, 4, 3, 13]

A.6/ Results and Scores computation attached notes

This section presents notes explaining in more details how results and scores are computed.

A.6.1/ Impact of run result aggregation weights on main score

figure A.2 shows the impact of run result aggregation weights on the main score of the benchmark for tested algorithms. V_N's weight, w_1, is on X-axis. As weigths are normalized, $w_2 = 1 - w_1$. On the right, where $x = 1$, only the value matters and on the left side, where $x = 0$, only the convergence speed matters.

From the first remarks on weights made in sub-section 4.3.5, w_1 should be set between 0.6 and 0.9. In this range, it can be observed that PSO, CMAES and Cuttlefish ranks change around $x = 0.75$. There is still room to discuss w_1 value, especially as the importance given to value and convergence depends on the optimization context. However, for a 'general context' approach, w_1 should be set between 0.7 and 0.8. $w_1 = 0.75$ and $w_2 = 0.25$ can be considered a good starting point for run result aggregation weights.

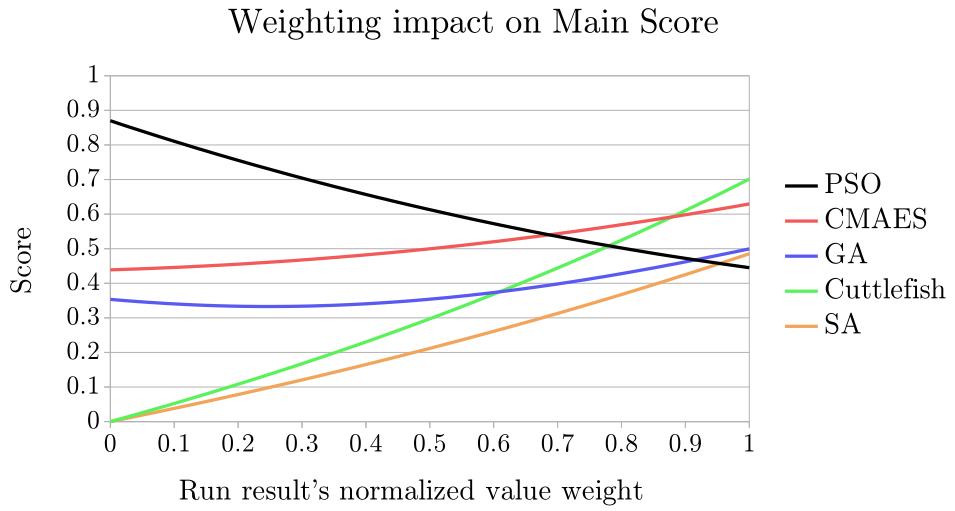


Figure A.2: Impact of run result aggregation's V_N's weight on Main score for tested algorithms

A.6.2/ Runs results of a draw

It is not easy to have a concrete idea of what run results could be achieved for a given couple of gross evaluations and gross values. To address this problematic some examples will be presented. Table A.12 presents PSO draw results for the case where $D = 10$, $F = 1$ and $M = 1$. This case has been chosen for the diversity of couples made by gross evaluations and gross values.

Table A.12: Draw results detail - PSO, D10, F1, M1

#	Run res	Norm res	Norm eval	Res	Eval

1	0.73	0.68	0.89	125	10960
2	0.62	0.54	0.88	1320	11760
3	0.55	0.46	0.88	9456	11740
4	0.60	0.52	0.87	2228	12220
5	0.59	0.51	0.87	2944	12040
6	0.58	0.50	0.86	3948	13920
7	0.57	0.49	0.86	5036	13140
8	0.36	0.52	0.00	2481	100000
9	0.67	0.61	0.87	283	12220
10	0.68	0.62	0.88	251	11900
11	0.67	0.60	0.88	327	11980
12	0.56	0.47	0.87	8289	12440
13	0.31	0.43	0.00	18109	100000
14	0.39	0.55	0.00	1021	100000
15	0.66	0.60	0.88	386	11720
16	0.68	0.62	0.87	235	12420
17	0.53	0.43	0.87	19844	12600
18	0.57	0.48	0.87	6326	12560
19	0.34	0.48	0.00	6227	100000
20	0.38	0.54	0.00	1372	100000
21	0.33	0.47	0.00	7757	100000
22	0.43	0.44	0.40	14344	59900
23	0.63	0.56	0.87	811	12520
24	0.57	0.49	0.86	4853	13840
25	0.57	0.47	0.88	6983	11680
26	0.78	0.75	0.86	101	13240
27	0.63	0.56	0.87	836	12140
28	0.73	0.69	0.86	116	13700
29	0.38	0.41	0.29	30432	70280
30	0.45	0.52	0.25	1990	74560
31	0.61	0.53	0.87	1672	12460
32	0.39	0.56	0.00	941	100000
33	0.33	0.47	0.00	7842	100000
34	0.34	0.47	0.00	6971	100000
35	0.59	0.50	0.87	3645	12220
36	0.65	0.58	0.87	580	12100
37	0.42	0.47	0.28	7879	71460
38	0.34	0.48	0.00	5695	100000
39	0.37	0.48	0.09	5459	90600
40	0.62	0.54	0.87	1364	12580
41	0.64	0.57	0.87	627	12240
42	0.62	0.54	0.88	1350	11160
43	0.34	0.47	0.00	6908	100000
44	0.30	0.43	0.00	21238	100000
45	0.38	0.49	0.08	4211	91100
46	0.37	0.52	0.00	2001	100000
47	0.31	0.43	0.00	19885	100000
48	0.33	0.47	0.00	7636	100000
49	0.65	0.58	0.88	557	11860
50	0.78	0.75	0.86	101	13160
51	0.73	0.69	0.88	118	11760

A.6.3/ Impact of confidence ratio

This sub-section will look for an acceptable value of the confidence limit ratio. The confidence limit goal is to measure how efficient the algorithm is when only one or a few runs are possible. The return value should be the worst of a large amount of the runs of a draw. This ratio has to be set at a high value such as 75%, 90% or even 99,9%. As a ratio under 50% would lead to a value higher than the average, the ratio will be explored from this point.

Figures A.3 and A.4 show the impact of confidence limit ratio on Benchmark's main score and omega sub-score for tested algorithm. Judging from the first figure, it can be noticed, that algorithms decrease almost steadily until 90%. Onwards, the curves slope drops. As one may suspect, in order for the confidence limit to have an actual impact on the main score, its ratio should be set above 90%. Given the results of the second figure, the same trend can be observed, except for Cuttlefish, which steadily decreases. On this figure, an interesting point can be noticed, at 95% where Cuttlefish an PSO curves are crossing each other. It seems that 95% is the minimum for which less sensitive algorithms may outperform sensitive ones. To conclude, the confidence limit ratio should be set at 95% or more.

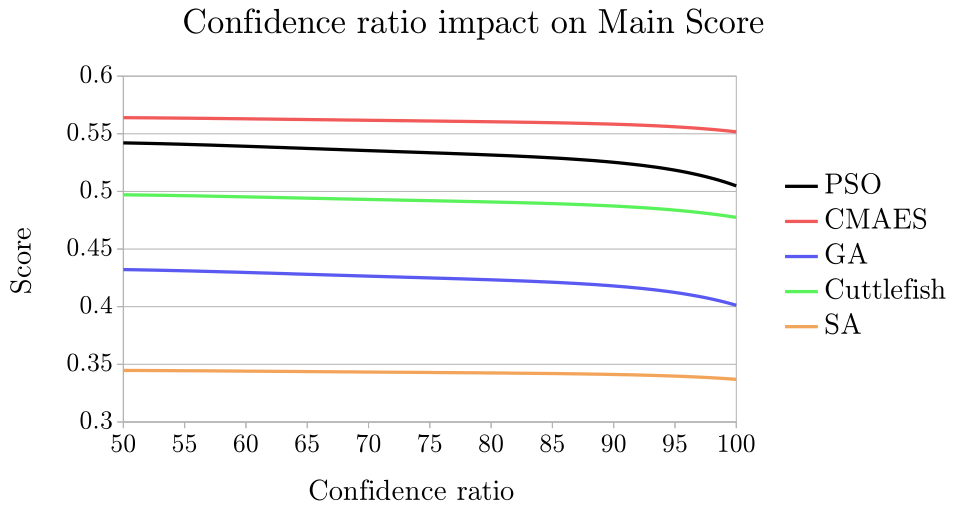


Figure A.3: Impact of confidence limit ratio on Main score for tested algorithms

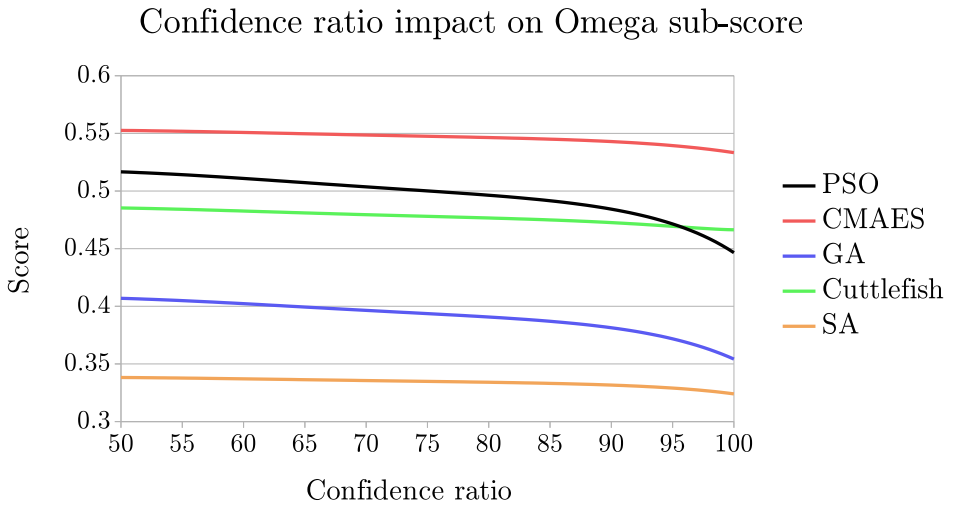


Figure A.4: Impact of confidence limit on Omega sub-score for tested algorithms

A.7/ Algorithms detail

A.7.1/ Covariance Matrix Adaptation Evolution Strategy (CMAES)

A.7.1.1/ Introduction

The CMAES [52] (Covariance Matrix Adaptation Evolution Strategy) is an evolutionary algorithm for difficult non-linear non-convex black-box optimization problems in continuous domains [184]. The CMAES is typically applied to unconstrained or bounded constraint optimization problems whose search space dimension lies between three and a hundred. Derived from the concept of self-adaptation in evolution strategies, the CMA (Covariance Matrix Adaptation) adapts the covariance matrix of a multi-variate normal search distribution. Two main principles for the adaptation of parameters of the search distribution are exploited in the CMAES algorithm.

The first one is the maximum-likelihood principle, increasing the probability of successful candidate solutions at each steps. It consists in updating the mean of the distribution and co-variance matrix, such that the likelihood of previously successful candidate solutions is maximized. The second one is the evolution paths. An evolution path is a sequence of successive steps. This evolution path is strategy over a number of generations. This strategy relies on the record of the two last steps of distribution mean's evolution. These paths contain significant information about the correlation between consecutive steps.

A.7.1.2/ Stopping condition

The CMAES stops if, for r_t iterations, all of the points evaluated are inside a radius. Meaning that, for each dimension, equation A.25 is verified. The CMAES also stops in case of an ill-conditioned matrix.

$$\forall < i, j >; point_i[d] - point_j[d] \leq 0.01 \cdot (\text{upper-limit}[d] - \text{lower-limit}[d]) \quad (A.25)$$

A.7.1.3/ Pseudo-code

CMAES pseudo code is given by algorithm 3.

Algorithm 3 CMAES pseudo code

Require: Initialize λ {Number of samples per iteration}
Require: Initialize m, σ, $C = I$, $p_\sigma = 0$, $p_c = 0$ {Initialize state variables}
while stopping criteria is not reach {iterate} **do**
  for $i \in \{1 \dots \lambda\}$ {Sample λ new solutions and evaluate them} **do**
    $x_i \sim N(m, \sigma^2 C)$
    $f_i = fitness(x_i)$
  end for
  $x_{i\dots\lambda} \leftarrow x_{s(i)\dots s(\lambda)}$ with $s(i) = argsort(f_{i\dots\lambda}, i)$
  $m' = m$ {Keep previous iteration mean in mind}
  $m \leftarrow$ update-m$(x_i, \dots, x_\lambda)$ {Move mean}
  $p_\sigma \leftarrow$ update-ps $(p_\sigma, \sigma^{-1} C^{-1/2}(m - m'))$ {Update isotropic evolution path}
  $p_c \leftarrow$ update-pc $(p_c, \sigma^{-1}(m - m'), \|p_\sigma\|)$ {Update anisotropic evolution path}
  $C \leftarrow$ update-C $(C, p_c, (x_1 - m')/\sigma, \dots, (x_\sigma - m')/\sigma)$ {Update covariance matrix}
  $\sigma \leftarrow$ update-sigma $(\sigma, \|p_\sigma\|)$ {Update step-size using isotropic path length}
end while
return m or x_1

A.7.1.4/ Setting

Table A.13 gives the setting of the CMAES used in this thesis. λ is defined according to the default strategy parameters [52]. The default step size is $\sigma = 0.5$ but [185] mentioned that a smaller initial step-size is especially useful on composition functions, which are highly weighted in the proposed benchmark. Therefore, σ has been lowered to 0.3. r_t and r_v values have been copied from PSO's radius [45].

Table A.13: CMAES setting

Parameter	Value	Remarks
λ	$\lambda = 4 + floor(3 \cdot log(D))$	Population size
σ	0.3	Coordinate wise standard deviation (step size)
r_t	10	Number of iteration inside radius to stop
r_v	1e−3	Radius threshold value

A.7.2/ Inhouse Genetic algorithm (GA)

A.7.2.1/ Introduction

In genetic algorithms, a population of candidate solutions, called individuals, is evolved towards better solutions. Each candidate solution possess a chromosome, which can be

mutated and altered [186]. A chromosome is a set of D genes.

The GA algorithm used in this thesis is an inhouse genetic algorithm. Its main difference relies in the way individuals are matched to produce offsprings. The idea is to simulate a scheme of animals forming couples. In addition, every individuals possess two chromosomes in order to be closer to how sexual reproduction works. The first chromosome is the one used for evaluation. The second chromosome only contains 'recessive' genes.

At each iteration, new individuals are created either by reproduction or by imitation. Both of those methods include mutation mechanisms. The generation of an individual includes mechanisms that randomly sets up genes.

- Reproduction: A new individual is generated from two parents. For each gene, each parent will give one of its corresponding genes. For both parents, which genes will be given is selected randomly. Which of the two parents genes will be on first chromosome is also chosen randomly. For each gene there is a probability P_m that the gene is mutated.
- Imitation: A new individual is generated thanks to the genes of the previous generation. For each genes there is a probability P_c that the gene is copied from a chromosome randomly selected in the previous generation population. For each copied gene there is a probability P_m that the gene is mutated. If the gene is not copied, it is randomly set.
- Mutation: If the gene is mutated it will be set randomly.
- Random set: Genes are randomly set inside the research limits of the algorithm. Indeed, the random selection is made over a portion of the search space. This portion is limited by upper and lower bounds over each variables. Limits are updated at each iteration in order to set genes in the supposedly interesting area of the search space. If algorithm converges, limits are shrunk with each new iteration.

The couple generation is based on the concept of attraction. Attraction of individual B on A ($A_{B\to A}$) expresses how much individual A want to mate with B. Attraction of B on A is computed, as in equation A.26, by three normed factors:

- Attractiveness of B (G_B): It represents how appealing individual B is. It is the fitness of the solution B relative to the current population.
- Affinity of A with B ($L_{A\to B}$): It represents how much A likes B. It is a random number set at each iteration. Affinity of A with B is different from affinity of B with A.
- Distance between A and B ($D_{A\leftrightarrow B}$): It represents what could prevent A and B to mate (distance in space, cultural and social barrier). It is the normed distance between A and B in search space.

$$A_{B\to A} = G_B + L_{A\to B} - D_{A\leftrightarrow B} \tag{A.26}$$

The generation of couples relies on two iterative procedures using attraction. The first one creates couples so that, among single individuals, the ones with each other greatest

attractions are matched. The second one creates couples so that, among single individuals, the ones with the highest average attractions are matched.

Each couple produces 2 childrens who are forbidden to breed together. The number of new individuals made by imitation depends solely on the fact that the population size must remain the same. The individual with the best fitness is called the patriarch. To avoid premature convergence, patriarch can be forbidden to reproduce.

A.7.2.2/ Stopping criteria

The GA stops if, for r_t iterations, all evaluated points during an iteration are contained inside a radius. Meaning that, for each dimension, equation A.27 is verified.

$$\forall < i, j >; point_i[d] - point_j[d] \leq 0.01 \cdot (\text{upper-limit}[d] - \text{lower-limit}[d]) \tag{A.27}$$

A.7.2.3/ Pseudo-code

GA pseudo code is given by algorithm 4.

Algorithm 4 Inhouse Genetic Algorithm pseudo code

```
Initialize population
Evaluate individuals
Compute distances between individuals
while Stopping criteria has not been reached do
    Compute individuals properties (Attractivenesses, Affinities and finally Attractions)
    Compute couples (two iterate procedure)
    Compute new limit (limits in which genes will be randomly drawn)
    Update population (by reproduction and imitation)
    Compute population fitness
    Update patriarch (individual with best fitness)
    Compute distances between individuals
end while
return Patriarch first gene
```

A.7.2.4/ Setting

Table A.14 gives the setting of GA used in this thesis. This setting has been found empirically and is considered as the default one.

A.7.3/ Simulated annealing (SA)

A.7.3.1/ Introduction

Simulated annealing [57] is a probabilistic technique for approximating the global optimum of a given function in a large search space. This approach is inspired by the process of annealing in metallurgy. Annealing involves heating and cooling a material to alter its

Table A.14: Genetic setting

Parameter	Value	Remarks
N	50	Population size
T_c	0.3	Probability to copy another gene during imitation procedure
T_m	0.1	Probability of mutation during reproduction procedure
A_p	False	Allow the patriarch to reproduce
r_t	10	Number of iteration inside radius to stop
r_v	1e−3	Radius threshold value

physical properties due to the changes in its internal structure. As the metal cools its new structure becomes stable, consequently causing the metal to retain its newly obtained properties.

This notion of slow cooling implemented in the simulated annealing algorithm is interpreted as a slow decrease in the probability of accepting worse solutions as the solution space is explored. Accepting worse solutions is a fundamental property of metaheuristics because it allows for a more extensive search for the global optimal solution. In general, the simulated annealing algorithm works as follows. At each time step, the algorithm randomly selects a solution close to the current one, measures its quality, and then decides to move onto it or to stay with the current solution. If the new position is better, the algorithm will move onto it. Otherwise, the algorithm may or may not move according to a probability. This probability, as the concept of temperature, will decrease with time.

A.7.3.2/ Pseudo-code

SA pseudo code is given by algorithm 5.

A.7.3.3/ Setting

Table A.15 gives the setting of SA used in this thesis. In this setting, Δ_t comes from [57] and the population size has been copied from PSO [45]. The Initial temperature, T, has been set after experimenting on objective-functions used by this benchmark.

Table A.15: SA setting

Parameter	Value	Remarks
N	20	Population size
T	1000	Initial temperature
Δ_t	0.9	Temperature decreasing geometric progression's commmon ratio

Algorithm 5 SA pseudo code

Require: Initialize x_i with $i \in \{1, \ldots, N\}$ {Initialize randomly the population}
Require: Initialize x_{best}
for $it = \{0, \ldots, it_{max}\}$ **do**
  $T \leftarrow \Delta_t \cdot T$ {Temperature is decreasing according to a geometric progression of common ratio Δ_t}
  for $i \in \{1, \ldots, N\}$ **do**
    $x'_i = neighbour(x_i)$ {A new solution (x'_i) close to the current one (x_i) is selected}
    if $fitness(x'_i) < fitness(x_i)$ **then**
      $x_i \leftarrow x'_i$ {Position is updated}
      if $x_i < x_{best}$ **then**
        $x_{best} \leftarrow x_i$ {Best known solution is updated}
      end if
    else
      if $\exp(-1 \cdot (fitness(x'_i) - fitness(x_i))/T) < rand()$ **then**
        $x_i \leftarrow x'_i$ {Position is updated}
      end if
    end if
  end for
end for
return x_{best}

A.7.4/ Cuttlefish

A.7.4.1/ Introduction

The Cuttlefish Algorithm is a meta-heuristic bio-inspired optimization algorithm. It mimics the mechanism of colour changing behavior used by the cuttlefish to solve numerical global optimization problems. Cuttlefish considers two main processes: reflection and visibility. The reflection process is proposed to simulate the light reflection mechanism used by these three layers, while the visibility is proposed to simulate the visibility of the matching pattern used by the cuttlefish [56]. The algorithm divides the population (cells) into four groups. Each group works independently sharing only the best solution. Two of them are used for global search, while the others are used for local search.

A.7.4.2/ Pseudo-code

Cuttlefish pseudo code is given by algorithm 6.

A.7.4.3/ Setting

Table A.16 gives the setting of the cuttlefish used in this thesis. The population is the one used in the initial article [56].

Algorithm 6 Cuttlefish pseudo code

Require: Initialize population ($P[N]$) with random solutions. Assign the values of $r1$, $r2$, $v1$, $v2$.

Evaluate fitness of the population, and keep the best solution in g_{best}.
Divide population into 4 Groups: G_1, G_2, G_3 and G_4
while Stopping criteria hasn't been reached **do**
  Calculate the average points of the best solution (g_{best}), and store it in $AVBest$
  for each cell in G_1 generate new solution using reflection and visibility, and calculate the fitness.
  for each cell in G_2 generate new solution using reflection and visibility, and calculate the fitness.
  for each cell in G_3 generate new solution using reflection and visibility, and calculate the fitness
  for each cell in G_4 generate a random solution, and calculate the fitness.
  if $fitness >$ current fitness **then**
    Current solution = new solution
  end if
  if fitness > G_{best} fitness **then**
    G_{best} = new solution
  end if
end while
return G_{best}

Table A.16: Cuttlefish setting

Parameter	Value	Remarks
N	60	population size

A.8/ Benchmark detailed scores

Table A.17 presents the scores obtained by the tested MO approaches. This table could be used to compare algorithms with more accuracy than with graphs. For instance, it shows that expert based approach obtain a better score than DoE on Benchmark one for f_{15} (with 0.615 for 0.614).

Table A.17: Benchmark scores obtained by the tested meta-optimization approaches

Kind	Category	Name	DoE on Bench-mark	Expert based	DoE on Function (Khosla)	DoE on OFTS (Wang)	DoE on RCTS (Das)
Global	Main score	Score	0.502	0.474	0.259	0.401	0.195
Infos	Quality	Value	0.439	0.491	0.366	0.368	0.250
		Convergence	0.830	0.503	0.000	0.690	0.000
	Reliability	Alpha	0.562	0.554	0.335	0.496	0.223
		Omega	0.447	0.398	0.212	0.313	0.172
Sub-set	Dimension	10	0.567	0.498	0.384	0.464	0.236
		20	0.543	0.504	0.358	0.458	0.188
		30	0.515	0.486	0.325	0.432	0.175
		50	0.473	0.453	0.211	0.362	0.206
	Function	1	0.284	0.517	0.325	0.220	0.018
		2	0.445	0.401	0.204	0.420	0.166
		3	0.208	0.097	0.016	0.092	0.017
		4	0.202	0.090	0.015	0.075	0.016
		5	0.281	0.136	0.061	0.101	0.054
		6	0.172	0.101	0.087	0.163	0.032
		7	0.343	0.235	0.207	0.255	0.033
		8	0.550	0.485	0.231	0.427	0.055
		9	0.148	0.022	0.013	0.015	0.016
		10	0.373	0.351	0.081	0.322	0.061
		11	0.619	0.598	0.191	0.419	0.053
		12	0.800	0.678	0.240	0.620	0.106
		13	0.670	0.592	0.416	0.569	0.411
		14	0.189	0.146	0.024	0.092	0.028
		15	0.614	0.615	0.383	0.520	0.353
	MaxFEs	0.01	0.298	0.290	0.184	0.285	0.177
		0.02	0.361	0.323	0.187	0.311	0.176
		0.05	0.425	0.378	0.199	0.341	0.178
		0.1	0.456	0.418	0.209	0.358	0.178
		0.2	0.483	0.448	0.224	0.382	0.177
		0.5	0.507	0.476	0.249	0.403	0.178
		1	0.517	0.495	0.290	0.415	0.208

A.9/ Highly constrained problems

How to efficiently solve an optimization problem will depends on how much constrained it is. Curently, optimization problem are considered un-constrained, constrained or highly constrained.

Many paper claim to face an highly constrained problem and using this statement to justify usage of this or that method. However, apart from few exceptions, no justification is ever given on why the problem should be qualified 'highly' constrained. In most of the case this assertion relies on empirical observation, discussion with the designer and optimizer experience.

A clear definition of highly constrained problem would benefit the community by helping researcher evaluating if their problem is truly highly constrained or not. In addition, a constraint level scale will be an interesting tool to develop as it would help researcher to chose an appropriate resolution method.

To acheive the previously stated goals, two interesting notions could be used. First, the feasible space [187], which is the part of the search space containing solutions respecting all the constraints. Second, the failure probability [64], which is the probability for product to not respect at least one constraint.

These two notions could be used to build a new metric: the feasible space ratio (FSR). This ratio is the probability of a solution, inside the search space, to be in the feasible space. A constrain level scale based on the FSR value is proposed in this section.

Several method could be used to compute the failure probability. The simplest one will be kept to compute the FSR. It consist of using random evaluations, according to Monte-carlos method, and to compute the percentage of random solutions respecting all constraints. This method could be improved, for instance by using latin hypercube sampling [188] instead of the Monte-Carlos [147], as advised by [189]. However, with the monte-carlos method it is easy to control the number of random evaluations made. The advised number of random evaluations to determine FSR is 1e6. This number has been chosen as it is the inverse of the scale's highest level's FSR, which is 1e−6 (for excessively constrained problem). For a first approximation, this number could be reduced, for instance to 1e2, the inverse of the highly constrained level's FSR.

The constraint level scale is given in Table A.18. This scales is based on levels defined by FSR values. To define the levels some assumption about EDO and algorithms used to solve them have been made:

- EDO problems could be solve in $FEs = D \cdot 10000 \cdot MaxFEs$ [119].
- MaxFEs should at least be superior to 0.01 for the problem to be solvable, as explained in sub-section 4.5.1) using [44].
- Typical EDO problem dimension is inferior to 100 [9, 119].
- Most of the population-based algorithm used for EDO have a typical population inferior or equal to 100 [44].

The levels' FSR values justifications are the following:

- Constrained: If $FSR < 1$, at least some part of the search space contain solutions that do not respect constraint.
- Highly Constrained: With $FSR \leq 1e{-}2$, if a population based algorithm is randomly initialize the odd of having at least one feasible solution are inferior to 50%.
- Extremely constrained: with $1e{-}4 \leq FSR \leq 1e{-}6$, once considered that $1e4 \leq FEs \leq 1e6$, if only random evaluations are used, the probability of find at least one feasible solution is not guarantee to be superior to 50%.
- Excessively constrained: With $FSR \leq 1e{-}6$, once considered that $FEs \leq 1e6$, if only random evaluations are used, the probability of find at least one feasible solution is inferior to 50%.

A few additional remarks about excessively constrained problem should be made. If a problem is considered excessively constrained, it is less an optimization problem than a constraints satisfaction problem [72]. In this case, it is not consistent to use this problem

Table A.18: Constraint levels scale

#	Level	Feasible space ratio	Comments
0	Unconstrained	1	All solutions inside the search space are feasible
1	Constrained	[1e−2, 1[	Constrains are not yet a major difficulty
2	Highly constained	[1e−4, 1e−2[	The optimization method should be oriented to solve constraints
3	Extremely constrained	[1e−6, 1e−4[	The optimization problem is about finding a feasible solution
4	Excessively constrained	]0, 1e−6[	As formulated, the problem does not make sense and is barely impossible to solve
5	Fully constrained	0	The problem could not be solve without modifying constraints

to test or compare optimization method, except if they are meant to solve excessively constrained problem. An optimization problem could be excessively constrained for different reasons such as:

- The specifications are almost impossible to reach due to physical or technological limitations
- The problem is poorly formulated
  - A different choice of variables and alias could have been made
  - Designer knowledge could have been used to simply the problem

In any case, if the problem could not be solved in state, the most obvious solution is to discuss problem with designer in order to modify it:

- Some constraint could be removed or loosen.
- What design parameters are used as variables or alias could be modified
- Maybe some of the methods used by the designer to tune its product could be used in optimization process.

A.10/ Landscape analysis

This section will briefly presents the function landscape [41] analysis of the Benchmark function and the application case. This analysis serves the selection of function sub-scores in the process of algorithm choice presented in chapter 4.
The landscape have been analyses thanks to the metrics presented in [41] which are given in table A.19.

Table A.19: Landscape metrics

Metric	Range	Interpretation
FEM (micro/macro)	$[0, 1]$	The first entropic measure (FEM) measures the landscape (micro/macro) ruggedness. 0 indicates a flat landscape and 1 indicates maximal ruggedness.
DM	$[0, 1]$	The dispersion metric (DM) measures the global topology of a function landscape. The higher DM the more disperse solutions in optimum basin are.
G_{avg}	$[0, +\infty]$	G_{avg} is the function average gradient.
G_{dev}	$[0, +\infty]$	G_{dev} is the standard deviation of gradient values.
FDC_s	$[-1, 1]$	The fitness-distance correlation (FDC_s) metric measures the performance of algorithms with unknown optima. 1 indicates the highest measure of searchability.
IL_{ns}	$[0, 1]$	The Information Landscape (IL_{ns}) measures functions searchability relative to sphere function one. 0 indicates maximum searchability.
FCI_{cog} / FCI_{soc}	$[0, 1]$	Fitness Cloud Index (FCI) indicates the proportion of fitness improving solutions after two PSO updates, using either cognitive (FCI_{cog}) or social (FCI_{soc}) mechanism.

The landscape metrics values of benchmark functions and SAW problem are given in table A.20. A few comments could be made about the values presented. First, only the SAW problem has a macro (respectively micro) ruggedness (FEM) superior to 0.5. This means that no benchmark function could be considered similar to SAW one in terms of ruggedness. The highest recorded average gradient (0.279) and gradient standard deviation (0.443) are obtained by the SAW problem. These value are at least to times superior to the second highest ones obtained by $F13$ and $F2$. Thus, in this case, ruggedness and gradient related metrics will not be considered in the choice of an algorithm. In addition, as all tested functions have a FCI_{soc} inferior to 0.1, this metric will also not be considered. Finally, only three functions, $F13$, $F14$ and $F15$, are harder on all the remaining metrics than the SAW one. Therefore, $F13$, $F14$ and $F15$ benchmark sub-scores will be selected to choose an algorithm suitable for the SAW problem.

Table A.20: Landscapes analysis

Function	FEM_{macro}	FEM_{micro}	DM	G_{avg}	G_{dev}	FDC_s	IL_{ns}	FCI_{cog}	FCI_{soc}
SAW	0.531	0.739	0.400	0.279	0.443	−0.004	0.405	0.329	0.000
1	0.000	0.000	0.334	0.002	0.002	0.471	0.339	0.466	0.076
2	0.150	0.112	0.399	0.076	0.198	0.026	0.484	0.342	0.014
3	0.000	0.000	0.401	0.003	0.003	0.065	0.483	0.357	0.003
4	0.000	0.000	0.396	0.008	0.007	0.049	0.484	0.385	0.008
5	0.000	0.314	0.404	0.074	0.070	−0.001	0.499	0.379	0.004
6	0.000	0.000	0.363	0.001	0.001	0.561	0.303	0.380	0.001
7	0.000	0.000	0.362	0.001	0.001	0.555	0.304	0.382	0.002
8	0.000	0.000	0.362	0.000	0.000	0.392	0.339	0.383	0.003
9	0.000	0.000	0.394	0.004	0.004	−0.001	0.502	0.373	0.006
10	0.008	0.008	0.391	0.022	0.041	0.064	0.476	0.353	0.009
11	0.018	0.009	0.377	0.022	0.039	0.156	0.442	0.384	0.015
12	0.048	0.026	0.375	0.042	0.073	0.142	0.445	0.369	0.007
13	0.117	0.086	0.443	0.095	0.106	−0.290	0.597	0.226	0.000
14	0.000	0.000	0.470	0.005	0.005	−0.216	0.574	0.208	0.000
15	0.017	0.042	0.468	0.087	0.069	−0.256	0.581	0.211	0.000

Document generated with LaTeX and:
the LaTeX style for PhD Thesis created by S. Galland — http://www.multiagent.fr/ThesisStyle
the tex-upmethodology package suite — http://www.arakhne.org/tex-upmethodology/