National Technical University of Athens
School of Mechanical Engineering
Fluids Section
Parallel CFD & Optimization Unit

Gradient-based Multi-Objective Optimization, for Discontinuous and 3D Pareto Fronts, with Applications in Aerodynamics

Diploma Thesis

Eleftherios Kokalis

Advisor: Kyriakos C. Giannakoglou, Professor NTUA

Athens, 2024

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor, Professor Kyriakos C. Giannakoglou, for entrusting me with the subject of my Diploma Thesis. His guidance and support have been invaluable throughout this journey. I am also profoundly appreciative of the time he dedicated to mentoring me during both my Diploma Thesis and my semester project in Spring 2023.

I would like to extend my heartfelt gratitude to Dr. Varvara Asouti for the theoretical and technical discussions we have had over these months. Her eagerness to help and willingness to address any inquiries I had have been truly invaluable. Additionally, I would like to thank Dr. Andreas Margetis for his constant support and suggestions for improvement, which were essential for completing this Thesis.

Last but not least, I would like to express my gratitude to my family and friends for their support during my years at NTUA. I am particularly grateful to my partner, Ellie, for standing by me and helping me overcome the challenges I faced throughout these years.

National Technical University of Athens
School of Mechanical Engineering
Fluids Section
Parallel CFD & Optimization Unit

Gradient-based Multi-Objective Optimization, for Discontinuous and 3D Pareto Fronts, with Applications in Aerodynamics

Diploma Thesis

Eleftherios Kokalis

Advisor: Kyriakos C. Giannakoglou, Professor NTUA

Athens, 2024

Abstract

This Diploma Thesis presents a method for solving Multi Objective Optimization (MOO) problems using Gradient-Based (GB) methods and tracking the Pareto front, with applications in Computational Fluid Dynamics (CFD). The proposed method efficiently tracks the Pareto front, starting from a random point or the baseline geometry. The corresponding software programmed in C++, can be applied to two or three-objective optimization problems. Additionally, new algorithms are suggested to address existing challenges in GB MOO, such as tracking discontinuous Pareto fronts and three-objective Pareto fronts.

The method utilizes the Prediction-Correction scheme, as it can compute a number of elite points at a specific range of values, with minimal cost. This method involves two steps: the Go-To-Pareto Step, which computes an initial optimal point, and the Move-on-Pareto Step, which continues by computing all the other non-dominated points through the aforementioned Prediction-Correction scheme.

The Correction-step utilizes the Augmented Lagrangian Method (ALM) equipped with Quasi Newton (Broyden–Fletcher–Goldfarb–Shanno, (BFGS) method), and Steepest descent, modified to handle general equality and inequality constraints. The Sequential Quadratic Programming (SQP) algorithm is also adapted, using Quasi Newton methods to approximate (instead of exactly computing) the hessian matrix, thus reducing computational cost in CFD applications. The accuracy of the Prediction-step in SQP is improved by using the last hessian approximation from the previous Correction-step.

The proposed method is applied in two mathematical applications and the computational cost and efficiency of the SQP and ALM algorithms are compared. Then, the method is used to optimize the shape of an isolated airfoil, for two objectives. The first derivatives of the flow-related objectives function(s) w.r.t. the design variables are computed using the continuous adjoint method by running the CFD code PUMA, developed by Parallel CFD & Optimization Unit of the National Technical University of Athens.

New ways to detect discontinuous Pareto fronts are proposed, linking the residuals of the Karush Kuhn Tucker conditions of the Prediction-step to curvature change of the front. In this way the prediction of discontinuous regions is enabled, for the following target point to be tracked on the front. Welford's online algorithm is used to detect potential discontinuities. A method to solve problems with discontinuous fronts is proposed, which is supported by three algorithms: Target-Objective jump, Back-tracking, and Swap Target-Objective. The method is tested on three benchmark cases.

For three-objective problems, the Scan by-Layers algorithm is proposed, extending the GB method by varying one target objective while keeping the other two constant. The target points set are efficiently tracked as demonstrated in two benchmark problems.

Finally, a CFD application is presented, which optimizes the shape of the same airfoil, for three aerodynamic objectives. The results of the Scan by-Layers algorithm are compared with those of an Evolutionary Algorithm regarding quality of the front and computational cost.

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Μηχανολόγων Μηχανικών
Τομέας Ρευστών
Μονάδα Παράλληλης Υπολογιστικής Ρευστοδυναμικής & Βελτιστοποίησης

Αιτιοκρατική Πολυκριτηριακή Βελτιστοποίηση, για Ασυνεχή και 3Δ μέτωπα Pareto, με εφαρμογές στην Αεροδυναμική

Διπλωματική Εργασία

Ελευθέριος Κοκάλης

Επιβλέπων: Κυριάκος Χ. Γιαννάκογλου, Καθηγητής ΕΜΠ

Αθήνα, 2024

Περίληψη

Η διπλωματική αυτή εργασία παρουσιάζει μια αιτιοκρατική μέθοδο που ανιχνεύει το μέτωπο Pareto, επεκτείνοντάς την σε εφαρμογές CFD. Η μέθοδος ανιχνεύει αποτελεσματικά το μέτωπο Pareto, με "λογικό" υπολογιστικό κόστος, έχοντας ως αφετηρία ένα τυχαίο σημείο ή αρχική γεωμετρία. Το σχετικό λογισμικό που προγραμματίστηκε σε C++ επιλύει MOO προβλήματα δύο ή τριών στόχων. Επιπροσθέτως, προτείνονται νέοι αλγόριθμοι για την αντιμετώπιση υπαρκτών προκλήσεων όπως κατά την ανίχνευση ασυνεχών μετώπων Pareto, ή τη σάρωση μετώπων Pareto τριών στόχων.

Η μέθοδος που αναπτύσσεται χρησιμοποιεί το σχήμα Πρόβλεψης-Διόρθωσης, καθώς μπορεί να υπολογίσει έναν αριθμό σημείων στο μέτωπο Pareto, για ένα συγκεκριμένο εύρος τιμών, με ελάχιστο υπολογιστικό κόστος. Αυτή η μέθοδος περιλαμβάνει δύο βήματα: το βήμα Go-to-Pareto, το οποίο εντοπίζει ένα αρχικό βέλτιστο σημείο, και το βήμα Move-on-Pareto, το οποίο εντοπίζει τα υπόλοιπα μη-κυριαρχούμενα σημεία σαρώνοντας το μέτωπο. Το τελευταίο είναι αυτό που εφαρμόζει το προαναφερθέν σχήμα Πρόβλεψης-Διόρθωσης.

Το βήμα διόρθωσης χρησιμοποιεί τη μέθοδο ALM με την παραλλαγή της Quasi-Newton μεθόδου (BFGS) και της απότομης καθόδου, τροποποιημένη για να χειρίζεται περιορισμούς ισότητας και ανισότητας. Προσαρμόστηκε επίσης η SQP χρησιμοποιώντας Quasi Newton μεθόδους για την εκτίμηση του εσσιανού μητρώου, μειώνοντας το υπολογιστικό κόστος στις εφαρμογές CFD, καθώς αποφεύγεται ο ευθύς υπολογισμός των δεύτερων παραγώγων. Επιπλέον, βελτιώθηκε η ακρίβεια του βήματος πρόβλεψης στην SQP χρησιμοποιώντας την τελευταία προσέγγιση του εσσιανού από το προηγούμενο βήμα διόρθωσης.

Παρουσιάζονται δύο μαθηματικές εφαρμογές του λογισμικού και συγκρίνεται το υπολογιστικό κόστος και η αποδοτικότητα των αλγορίθμων SQP και ALM. Στη συνέχεια, παρουσιάζεται μία εφαρμογή CFD για τη βελτιστοποίηση σχήματος μιας μεμονωμένης αεροτομής, με δύο στόχους. Οι πρώτοι παράγωγοι των αεροδυναμικών ποσοτήτων υπολογίζονται μέσω της συνεχούς συζυγούς μεθόδου, με χρήση του λογισμικού επίλυσης

ροών (κώδικας PUMA) που αναπτύχθηκε από τη Μονάδα Παράλληλης Υπολογιστικής Ρευστοδυναμικής & Βελτιστοποίησης του Εθνικού Μετσόβιου Πολυτεχνείου.

Στη συνέχεια, διατυπώνονται νέες προτάσεις για την ανίχνευση ασυνεχειών στα μέτωπα Pareto, συσχετίζοντας τα υπόλοιπα (residuals) των συνθηκών Karush Kuhn Tucker, μετά το βήμα της πρόβλεψης με αλλαγή της καμπυλότητας του μετώπου. Με αυτόν τον τρόπο προβλέπονται περιοχές ασυνέχειας που περιλαμβάνουν το επόμενο σημείο του μετώπου. Επίσης, ενσωματώνεται ο στατιστικός online αλγόριθμος του Welford για την ανίχνευση ασυνεχειών. Προτείνεται, επιπλέον, μια μέθοδος αντιμετώπισης των ασυνεχειών που περιλαμβάνει τους αλγορίθμους: Target-Objective jump, Back-tracking και Swap Target-Objective και δοκιμάζεται σε τρεις μαθηματικές εφαρμογές.

Ο αλγόριθμος Scan by-Layers επεκτείνει τη μέθοδο Πρόβλεψης-Διόρθωσης για προβλήματα βελτιστοποίησης τριών στόχων, μεταβάλλοντας εναλλάξ τον ένα στόχο ενώ οι δύο άλλοι διατηρούνται σταθεροί. Για τους στόχους που έχουν τεθεί, ανιχνεύονται αποτελεσματικά τα σημεία του μετώπου μέσω αυτής της μεθόδου. Η ακρίβειά της επικυρώνεται σε δύο ενδεικτικά προβλήματα αναφοράς.

Τέλος παρουσιάζεται μια εφαρμογή CFD που βελτιστοποιεί το σχήμα μιας μεμονωμένης αεροτομής για τρεις αεροδυναμικούς στόχους. Τα αποτελέσματα του αλγορίθμου Scan by-Layers παρατίθενται δίπλα σε αυτά ενός εξελικτικού αλγορίθμου, πραγματοποιώντας συγκρίσεις ως προς την ποιότητα του μετώπου και το κόστος υπολογισμού.

Abbreviations

EA	Evolutionary Algorithm
SOO	Single Objective Optimization
MOO	Multi Objective Optimization
GB	Gradient-Based
ALM	Augmented Lagrangian Method
CFD	Computational Fluid Dynamics
EASY	Evolutionary Algorithm SYstem
PCOpt	Parallel CFD & Optimization unit
NTUA	National Technical University of Athens
PUMA	Parallel solver, for Unstructured grids, for Multi-blade row computations, including Adjoint
SQP	Sequential Quadratic Programming
EFS	Equivalent Flow Solutions
KKT	Karush Kuhn Tucker
BFGS	Broyden–Fletcher–Goldfarb–Shanno
SR-1	Symmetric Ranking One
NURBS	Non Uniform Rational B-Splines
DFP	Davidon-Fletcher-Powell
ShpO	Shape Optimization
BP	Benchmark Problem

Contents

Chapter 1

Introduction

1.1 Introduction to optimization theory

Optimization is the process of finding the best possible solution under a set of given circumstances. Since its development, optimization theory has been used in all engineering branches as a method to minimize costs and effort of a certain procedure, or maximize the desired output and performance of a product. As a branch of mathematics, optimization is concerned with finding a set of design variables $\vec{x} : \{x_1, x_2, \ldots, x_n\}$ that maximize or minimize an objective function f. Given that maximizing f is equivalent to minimizing $-f$, it has been 'de facto' established (without loss of generality) that optimization stands for minimization.

In aeronautical engineering, optimization is concerned with the minimization of drag of an airfoil or an airplane, the maximization of the corresponding lift, etc. In mechanical and aeronautical engineering problems a set of constraints is often imposed at the desired solution point, and this increases the complexity of the optimization problem.

Optimization methods can be categorized based on different criteria. They are distinguished depending on the imposition or not of constraints (Constrained and Unconstrained Optimization), the use of one or many targets (Single Objective Optimization (SOO), Multi Objective Optimization (MOO)), or the type of method used to find the target set of design variables (Stochastic Methods and Gradient-Based Methods (GB)). In the following section, the different categories of optimization methods are briefly discussed.

1.2 Categories of optimization methods

A popular distinction among optimization methods is that between gradient-based (GB) and stochastic methods:

A GB optimization method makes use of the derivatives of the objective function. The most significant advantage of such methods is that the optimization converges faster and can efficiently handle a large number of design variables, while retaining minimal computation cost. Nonetheless, GB methods have the major downside of occasionally converging to a local extremum.

Stochastic optimization algorithms are generally more easily adaptable to different optimization problems, in comparison to their counterparts, [11]. They deal with the search of the optimal solution utilizing randomized sets of design variables. The major advantage of stochastic algorithms is that they can easily be programmed and always converge to the global minimum of the objective function, if a great number of function calls is allowed. On the other hand, they tend to perform poorly with a great number of design variables and they, generally, converge in a much slower rate compared to GB methods, (even if assisted by surrogate evaluation models).

1.2.1 Line search methods

Line Search methods are a class of iterative GB methods that update in each step the current set design of variables $\vec{x}^k$, that approaches the optimal solution, by a vector search direction $\vec{p}^k$, multiplied by a coefficient η^k controlling the step size of each update.

Steepest descent

Steepest Descent is one of the most popular Line Search methods that, at a given iteration k of the optimization process, uses as search direction $\vec{p}^k$, the opposite to the local gradient of the objective function F$(\vec{x}^k)$. This can be expressed as:

$$\vec{p}^k = -\nabla F(\vec{x}^k) \tag{1.1}$$

The design variables $\vec{x}^{k+1}$ at each step of the method are obtained by:

$$\vec{x}^{k+1} = \vec{x}^k + \eta^k \vec{p}^k \tag{1.2}$$

Quasi Newton methods

Another class of Line Search Methods is that of Quasi-Newton methods. These methods approximate the hessian matrix, by updating an initial approximation of it. The approximated hessian is symbolized as B^k for the rest of the section, while H^k represents the inverse of the approximated hessian matrix. The effectiveness of Quasi-Newton methods largely depends on the accuracy of the initial hessian matrix. They are known for their rapid convergence rate and robustness when the initial point is close to the sought optimal solution.

The hessian of $F(\vec{x}^k)$ is approximated using the Taylor expansion of $F(\vec{x}^k + \vec{p}^k)$, which leads to the following expression (secant method), [11]:

$$\nabla^2 F(\vec{x}^{k+1})(\vec{x}^{k+1} - \vec{x}^k) \approx \nabla F(\vec{x}^{k+1}) - \nabla F(\vec{x}^k) \tag{1.3}$$

The following terms can be defined:

$$\begin{aligned} \vec{s}^k &= \vec{x}^{k+1} - \vec{x}^k \\ \vec{y}^k &= \nabla F(\vec{x}^{k+1}) - \nabla F(\vec{x}^k) \end{aligned} \tag{1.4}$$

the most popular Quasi-Newton methods are Symmetric Ranking-One (SR-1) and Broyden–Fletcher–Goldfarb–Shanno (BFGS) methods, which are formulated as:

1. **SR-1 method**:

$$B^{k+1} = B^k + \frac{(\vec{y}^k - B^k\vec{s}^k)(\vec{y}^k - B^k\vec{s}^k)^T}{(\vec{y}^k - B^k\vec{s}^k)^T\vec{s}^k} \tag{1.5}$$

2. **BFGS method**:

$$B^{k+1} = B^k + \frac{\vec{y}^k(\vec{y}^k)^T}{(\vec{y}^k)^T\vec{s}^k} - \frac{B^k\vec{s}^k(B^k\vec{s}^k)^T}{(\vec{s}^k)^T B^k\vec{s}^k} \tag{1.6}$$

Both methods require inverting the matrix B^{k+1}, after each iteration in order to compute its inverse H^{k+1}. The search direction $\vec{p}^k$ is then defined as:

$$\vec{p}^k = -H^k \cdot \nabla F(\vec{x}^k) \tag{1.7}$$

The update of the inverse hessian matrix H^{k+1} can also be approximated, without inverting the B^{k+1} matrix (DFP formula), as:

$$H^{k+1} = (I - \rho^k\vec{s}^k\vec{y}^{k^T})H^k(I - \rho^k\vec{y}^k\vec{s}^{k^T}) + \rho^k\vec{s}^k(\vec{s}^k)^T \tag{1.8}$$

where:

$$\rho^{k+1} = \frac{1}{(\vec{y}^k)^T\vec{s}^k} \tag{1.9}$$

In case its value is smaller than a predetermined threshold, the optimization skips the update step of the hessian (B^k), retaining its current values.

1.3 Basic Terminology of MOO

1.3.1 Introduction

Another distinction between different types of optimization methods is that between SOO and MOO. SOO methods deal with the optimization of a single objective function, for a single objective. SOO methods can also be applied to multi-objective problems, as long as the objectives are formulated into a single objective function, likely using weights. MOO methods are concerned with the optimization of more than one objective functions or the components of an objective vector function. The optimal (non-dominated) solutions in the multi-dimensional objective space form the Pareto front or front of non-dominated solutions.

1.3.2 Mathematical definition, the concept of Non-Dominance

MOO methods are recently increasing in popularity, due to the fact that they can compute a set of solutions that offers trade-offs among contradicting objectives. MOO problems are formulated as:

$$\min \vec{f}(\vec{x}) = min \begin{bmatrix} f_1(\vec{x}) \\ f_2(\vec{x}) \\ \vdots \\ f_m(\vec{x}) \end{bmatrix} \tag{1.10}$$

subject to the following constraints:

$$\begin{aligned} h_j(\vec{x}) &= 0, && \text{for } j = 1, \ldots, M_h \\ g_i(\vec{x}) &\leq 0, && \text{for } i = 1, \ldots, M_g \end{aligned}$$

Where $f : R^n \to R^m$, $g : R^n \to R^{M_g}$, and $h : R^n \to R^{M_h}$ are all sufficiently differentiable functions. The feasible solutions set is denoted as: $D := \{\vec{x} \in R^n \mid h(\vec{x}) = 0, g(\vec{x}) \leq 0\}$

The Pareto front represents the optimal solutions for MOO problems and, with two objectives is formed by a number of discrete points that tend to form a continuous or discontinuous curve. The curve can either be convex or non-convex; different approaches to compute the front have been formulated.

Another integral concept associated with the Pareto front is that of non-dominance, which refers to points in the objective space that are not dominated by other points. Mathematically, this can be expressed as:

$$f_i(\vec{x}^{\,1}) \leq \; f_i(\vec{x}^{\,2}), \forall i \in [1, m] \tag{1.11}$$

Provided that, for at least one objective function k, $k \in [1, m]$

$$f_k(\vec{x}^{\,1}) \;<\; f_k(\vec{x}^{\,2}) \tag{1.12}$$

where $\vec{x}^{\,1}$ is the set of design variables that corresponds to the optimal, non-dominated solution, also called "Pareto Optimal", and $\vec{x}^{\,2}$ represents the design variables of the dominated value of f.

In Figure 1.1, gray points are some dominated solutions by the Pareto front, from which horizontal and vertical lines are drawn, connecting each point with the horizontal and vertical axis, respectively. The set of points included within the square that is formed from the two perpendicular lines, represents all the solutions that dominate this point.

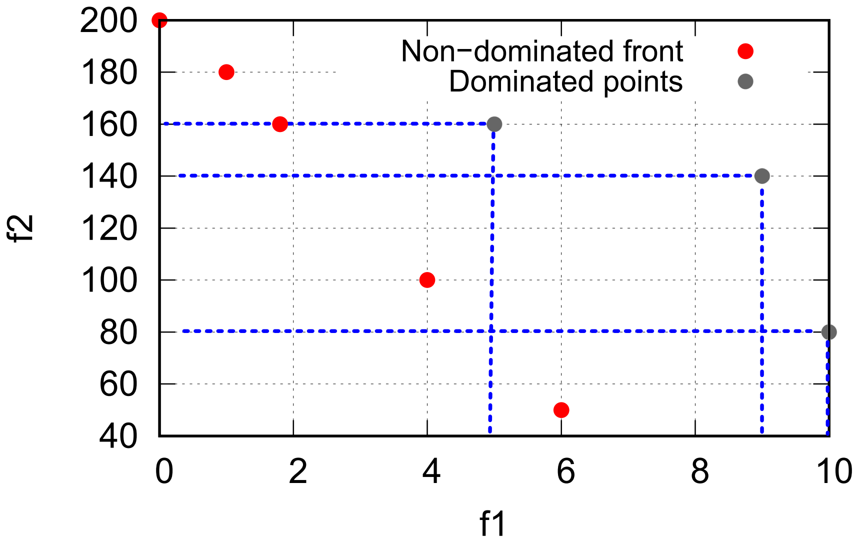


Figure 1.1: Pareto front, of a (min f_1, min f_2) problem, with non-dominated points shown in red and dominated solutions shown in gray.

From Figure 1.1, it is noted that any point could be Pareto-optimal, if and only if there are no other points inside the area defined by the vertical and horizontal lines. The horizontal and vertical lines at an optimal point would invert their direction, to include all solutions that are dominated by that point.

1.4 Pareto Points Tracking

GB Pareto tracking strategies aim at moving from one Pareto point to a neighbouring one by taking into account the local front curvature. Such strategies require the a priori input of a target step and of an initial Pareto point, in order to proceed to tracking the front.

A GB Pareto tracking scheme is that of the Prediction-Correction, [1]. The Prediction-step offers an initial approximation of the new Pareto point. The Correction-step consists of computing the exact optimal solution of the MOO problem for the target set. Research on the Prediction-Correction scheme and its application in CFD has already been performed in [22], [21], [8], [13], but there are still open areas of research.

An important challenge is that of reducing the computational cost of the scheme in CFD optimization. To this end, alternative and less computationally intense optimization methods have to be utilized. Another issue is that of tracking discontinuous (2D) Pareto fronts. Discontinuities in Pareto fronts can arise due to constraints imposed on either f_1 or f_2 over a specific interval, or from changes in the local curvature of the front. These factors result in regions where solutions are either non-feasible or dominated. The aforementioned works have addressed the need to extend the GB method for three-objective problems or an even higher dimension.

1.5 Purpose of the Thesis

The purpose of this Diploma Thesis is to extend the Prediction-Correction method for solving MOO problems and tracking the Pareto Front, and demonstrate its use in CFD applications. The necessary software was programmed in C++. This thesis is an extension of research presented by, [22], [21], [8], [13]. The following novel contributions are made:

- The SQP algorithm is integrated into the Correction-step of the Prediction-Correction scheme on par with the ALM, that was used in previous research for solving equality and inequality constrained problems. A comparison of the two methods in terms of their computational cost is made.

- A method to detect and track discontinuous Pareto fronts is proposed. This method is integrated into the software and tested on three mathematical Benchmark Problems, (BPs).

- An algorithm (Scan by-Layers) is formulated for expanding the GB method to tracking 3D Pareto fronts, at a low computational cost. The Scan by-Layers algorithm is applied to two mathematical problems and a CFD three-objective ShpO application of an isolated airfoil, successfully tracking the Pareto front.

1.6 Thesis Outline

The Thesis layout is organised as following:

Chapter 2: A brief introduction to constrained GB optimization methods is made. After the Karush Kuhn Tucker (KKT) conditions are presented, the Sequential Quadratic Programming (SQP) method is introduced for both equality and inequality constraints.

Chapter 3: This chapter lays the groundwork for the GB method used throughout the thesis. After introducing the Go-to-Pareto and Move-on-Pareto steps, which track an initial point and the front and move along it respectively, the Prediction-step and Correction-step are introduced and analyzed. The GB method is applied in two mathematical problems and its effectiviness in tracking the Pareto fronts is assessed. The ALM and SQP variants of the Correction-step are both used and their computational cost is compared.

Chapter 4: The GB method is used for the shape optimization (ShpO) of an isolated airfoil. The flow around the airfoil is inviscid, and the results of the GB method are compared to those of EASY, [12]. A variant of this problem constrained zero-pitch coefficient C_M is also optimized and presented.

Chapter 5: New methods are proposed to tackle challenges when tracking discontinuous Pareto fronts. A method to detect discontinuities is suggested based on the residuals of the KKT conditions of the Prediction-step. The algorithms of Target objective jump, Back-tracking and Swap Target-Objective are combined, constituting a method to track discontinuous fronts. Three mathematical applications of the methods are presented, investigating the effectiveness of the algorithms to detect discontinuities and track discontinuous fronts.

Chapter 6: Methods to track three-objective MOO problems are explored leading to the Scan by-Layers algorithm developed in this Diploma Thesis. This algorithm is applied in two mathematical BPs.

Chapter 7: The Scan by-Layers three-objective optimization algorithm is applied to a variant of the airfoil ShpO problem discussed in Chapter 5, with the third objective being the maximization of C_L at take-off conditions. The results of the GB algorithm optimization are, then, compared with those of EASY.

Chapter 8: This chapter presents the conclusions drawn from the research performed in the thesis, along with suggestions for future work.

Appendix A: The distinction between the concept of weak and strong dominance is discussed and made clear in Pareto fronts.

Appendix B: The Augmented Lagrangian method is presented.

Appendix C: The implicit function theorem is defined and applied to perform the Prediction-step.

Chapter 2

Constrained Optimization Methods

2.1 The KKT Conditions

When considering equality and inequality constraints, the generalized constrained optimization problem is:

$$\begin{aligned} &\text{Minimize } f(\vec{x}) \\ &\text{subject to} \quad g_i(\vec{x}) \leq 0, \quad i = 1, 2, \ldots, M_g \\ &\qquad\qquad\quad h_j(\vec{x}) = 0, \quad j = 1, 2, \ldots, M_h \end{aligned}$$

To take constraints into account, the concept of the Lagrange function is introduced, which for M_h equality and M_g inequality constraints takes the form:

$$L(\vec{x}, \vec{\lambda}, \vec{\mu}) = f(\vec{x}) - \sum_{j=1}^{M_h} \lambda_j h_j(\vec{x}) - \sum_{i=1}^{M_g} \mu_i g_i(\vec{x}) \tag{2.1}$$

Here, λ_j and μ_i denote the scaling coefficients of the equality and inequality constraints, respectively. These coefficients are used to penalize the objective function when the constraints are violated. λ_j, μ_i are also referred to as Lagrange multipliers.

In a minimization problem, two necessary conditions that must be met by the solution x^* are the stationarity of the first gradient w.r.t. $\vec{x}$ and the feasibility, (primal and dual) of constraints:

$$\nabla L(x^*, \lambda_j^*, \mu_i^*) = 0. \tag{2.2}$$

$$h_j(x^*) = 0. \tag{2.3}$$

$$g_i(x^*) \leq 0. \tag{2.4}$$

$$\mu_i^* \leq 0. \tag{2.5}$$

Furthermore, the first-order optimality conditions must account for the impact of active inequality constraints at the optimal solution, while disregarding the impact of inactive constraints (slackness condition). Thus, an additional condition is introduced:

$$\lambda_j^* h_j(x^*) = 0. \tag{2.6}$$

Eqs. (2.2), (2.3), (2.4), (2.5) and (2.6) are necessary conditions that must be satisfied by the optimal solution x^*, λ_j^*, μ_i^*, and are known as the Karush-Kuhn-Tucker (KKT) conditions.

2.2 The SQP Algorithm

The SQP method algorithm is one of the most robust and efficient iterative methods for solving constrained optimization problems. It works by decomposing the original nonlinear problem into a series of quadratic sub-problems, which are independently solved in place of the original problem. The algorithm can be used for solving problems with equality and inequality constraints, [6].

2.2.1 The SQP Algorithm for equality constraints

The first subproblem is defined by the second-order Taylor expansion of the objective function f around the current iterate $\vec{x}_k$. It involves minimizing a quadratic function of the step $\Delta\vec{x}$, which approximates the change in the objective function near $\vec{x}_k$. If equality constraints are applied, they are also expanded around $\vec{x}_k$ using a first-order Taylor expansion. Thus, the problem is formulated as follows:

$$\begin{aligned}&\min_{\Delta\vec{x}} f(\vec{x}_k) + (\nabla f(\vec{x}_k))^T \Delta\vec{x} + \frac{1}{2}\Delta\vec{x}^T H_{xx} \Delta\vec{x} \\ &\text{subject to } h(\vec{x}_k) + J\Delta\vec{x} = 0\end{aligned} \tag{2.7}$$

In order to satisfy the KKT conditions, eqs. (2.7) are formulated as:

$$\nabla^2_{xx} L(\vec{x}_k, \vec{\lambda}_k)\Delta\vec{x} + \nabla f(\vec{x}_k) - J^T\vec{\lambda}_k = 0, \tag{2.8}$$

$$J\Delta\vec{x} + h(\vec{x}_k) = 0$$

where $f(\vec{x}_k)$ is the objective function at the current iteration $\vec{x}_k$ while term H_{xx} notates its hessian. The Jacobian matrix of constraints, denoted by J, is defined as:

$$J^T = [\nabla h_1(\vec{x}_k), \nabla h_2(\vec{x}_k), \ldots, \nabla h_{M_h}(\vec{x}_k)] \tag{2.9}$$

Eq. (2.8) can be cast in a matrix form, thus forming the Jacobian of the quadratic sub-problem at iteration k as:

$$\begin{bmatrix} \nabla^2_{xx} L(\vec{x}_k, \vec{\lambda}_k) & -J_k^T \\ J_k & 0 \end{bmatrix} \begin{bmatrix} \Delta\vec{x} \\ \vec{\lambda}_{k+1} \end{bmatrix} = \begin{bmatrix} -\nabla f(\vec{x}_k) \\ -h(\vec{x}_k) \end{bmatrix} \tag{2.10}$$

The dimension of Jacobian matrix is $[n + M_h \times n + M_h]$. The matrix system can be solved by using any iterative method. For the purposes of this thesis, the Newton method is used in order to compute the perturbations vector P:

$$\mathbf{P} = \begin{bmatrix} \Delta\vec{x} \\ \vec{\lambda}_{k+1} \end{bmatrix}$$

After each iteration, the design variables and the hessian matrix $\nabla^2_{xx} L(\vec{x}_k, \vec{\lambda}_k)$ are updated. It is worth noting that this formulation of the quadratic sub-problem allows for the direct computation of $\vec{\lambda}_{k+1}$, while $\vec{x}_{k+1}$ at the next iteration is updated as:

$$\vec{x}_{k+1} = \vec{x}_k + \Delta x \tag{2.11}$$

The hessian is approximately updated using Quasi-Newton methods (BFGS, SR-1). These methods update only positive definite symmetric matrices. Hence, $\nabla^2_{xx} L(\vec{x}_k, \vec{\lambda}_k)$ is required to possess these characteristics which help avoiding a non-singular solution vector **P** for the system in eq. (2.10).

2.2.2 Handling of Inequality Constraints Using SQP, (Active-Set Method)

The active-set method is an algorithm that identifies which inequality constraints are equal to zero at the optimal solution (x^*, λ^*). These active constraints are then treated as equalities, transforming the original problem with inequality constraints into a simpler subproblem consisting solely of equality constraints. During iteration k of the SQP algorithm, inequality constraints are classified as either active or inactive:

$$\text{Active} := \{i \in \{1, \dots, M_g\} : g_i(\vec{x_k}) \geq 0\} \tag{2.11}$$

$$\text{Inactive} := \{i \in \{1, \dots, M_g\} : g_i(\vec{x_k}) < 0\} \tag{2.12}$$

If a set of inequality constraints is determined to be inactive at iteration k, they are not minimized in eq. (2.1) and are thus disregarded from the Lagrangian for the current iteration. However, these constraints can be re-activated in a subsequent iteration if eq. (2.11) holds true for them. Therefore, only the active inequality constraints are considered in eq. (2.1), denoted as $g_i(\vec{x}_k)$.

The algorithm of active-set method is thus formulated as:

Algorithm 1 Active-set method, SQP Algorithm

```
Require: Choose an initial pair $(\vec{x}_{s0}, \vec{\lambda}_0)$, choose a convergence tolerance $\epsilon$.
Ensure: Solution $(x^*, \lambda^*)$
 1: Set $k \leftarrow 0$
 2: Set convergence criteria: $\Delta x_k < \epsilon$
 3: while convergence criteria not met do
 4:     Estimate the Active-set of constraints: $J_{\text{ineqcoeff}} \Delta\vec{x}_k + g(\vec{x}_k) \geq 0$ or $g(\vec{x}_k) \geq 0$
 5:     Consider active inequality constraints into, eq. (2.1), as $\sum_{i=1}^{M_g} \mu_i g_i(\vec{x})$.
 6:     Compute $f(\vec{x}_k)$, $\nabla f(\vec{x}_k)$, $\nabla^2_{xx} L(\vec{x}_k, \vec{\lambda}_k)$, $h(\vec{x}_k)$, and $J_k$
 7:     Solve eq.(2.10) to get $P$
 8:     Update design variables $\vec{x}_{k+1} \leftarrow \vec{x}_k + \Delta\vec{x}_k$
 9:     Obtain $\vec{\lambda}_{k+1}$ from $P$
10:     if covergence criteria met then
11:         $(x^*, \lambda^*) \leftarrow (\vec{x}_{k+1}, \vec{\lambda}_{k+1})$
12:         stop
13:     end if
14:     Update the hessian matrix $\nabla^2_{xx} L(\vec{x}_k, \vec{\lambda}_k)$ using BFGS or SR-1
15:     $k \leftarrow k + 1$
16: end while
```

The active-set method is utilized in the applications presented in the next chapters of the thesis, when constrained optimization problems are solved.

Chapter 3

Tracking the Pareto Front Using GB Methods

3.1 Introduction

The basic layout for MOO problems has been presented in section 1.3. In this Diploma Thesis, the GB method developed uses the "Prediction-Correction" scheme, [1].

The MOO problem can be formulated using the Lagrangian function. In order to track the set of Pareto points for a problem with M_t objectives, f_1 is minimized and the rest of the M_t -1 objectives are treated as additional equality constraints that must be met at the optimal solution: $(f_k - \hat{f_k}) = 0$. Therefore, the Lagrangian function becomes:

$$L(\vec{x}, \vec{\lambda_f}, \vec{\lambda_h}, \vec{\mu}) = f_1(\vec{x}) - \sum_{k=2}^{M_t} \lambda_{f_k}(f_k - \hat{f_k}) - \sum_{j=1}^{M_h} \lambda_{h_j} h_j(\vec{x}) - \sum_{i=1}^{M_g} \mu_i g_i(\vec{x}) \tag{3.1}$$

3.2 Pareto Tracking GB Method

3.2.1 Tracking the Pareto Front

Tracking the Pareto front, given an initial non-optimal point consists of locating a first point of the front (Go-to-Pareto step) and moving along the front by tracking neighbouring points (Move-on-Pareto step). In a CFD MOO problem, the baseline geometry can be provided as input to the algorithm, or alternatively an initial Pareto point, in which case, the Go-to-Pareto step is unnecessary. The main steps of the algorithm are outlined below.

3.2.2 Go-to-Pareto step

The Go-to-Pareto Step is applied, when an initial set of design variables $\vec{x}$ is provided as input to the problem and, thus, an initial Pareto point has to be found. The first Pareto Point can be found by using the weighted sum method, which tracks the point for either convex or non-convex Pareto fronts and is formulated as:

$$\min_{x \in D} f(\vec{x}) = \vec{w} \cdot \vec{f}(\vec{x}) = \sum_{i=1}^{M} w_i f_i(\vec{x}) \tag{3.2}$$

By solving eq. (3.2) w.r.t. either for w_1=0 or w_2=0, in two-objective problems, the first and last points at the edge of the Pareto front can be tracked. An alternative way is treating the Go-to-Pareto step as an initial Correction-step, and thus optimizing eq. (3.1), by using SQP or ALM. In the following applications of the Thesis, the latter approach is used.

3.2.3 Move-on-Pareto steps

If the initial Pareto point is provided as input to the method or the Go-to-Pareto step has been completed, the Move-on-Pareto steps are executed consecutively in order to track the Pareto front. This is achieved by selecting an appropriate target $\hat{f}_k$ after a Pareto point has been tracked and perform a Prediction-step which provides an initial approximation of optimal design variables and Lagrangian multipliers. Then, the Correction-step allows for the precise identification of optimal Pareto points. In both steps, the Lagrangian function eq. (3.1) is used to model the MOO problem.

3.2.4 Prediction-Step

The Prediction-step approximates the optimal solution at the selected target $\hat{f}_k$. This is realized by expanding expressions, $x^* = x^*(f_k)$, $\lambda^* = \lambda^*(f_k)$ and $\mu^* = \mu^*(f_k)$, which are collectively denoted with $\vec{y}^* = h(f_k)$, as first-order Taylor series around the target $\hat{f}_k$. Therefore, $\vec{y}^*_{i+1}$ is approximated as:

$$\vec{y}^*_{\text{predicted},i+1} = \vec{y}^*_i + \frac{\partial \vec{h}}{\partial \hat{f}_k} \delta \hat{f}_k \tag{3.3}$$

$$\vec{z} = \begin{pmatrix} \hat{f}_2 \\ \vdots \\ \hat{f}_{Mt} \end{pmatrix}, \quad \vec{y} = \begin{pmatrix} \vec{x} \\ \vec{\lambda}_f \\ \vec{\lambda}_h \\ \vec{\mu} \end{pmatrix}, \quad H(\vec{z}, \vec{y}) = \begin{pmatrix} \nabla L(\vec{x}, \vec{\lambda}_f, \vec{\lambda}_h, \vec{\mu}) \\ f_k(\vec{x}) - \hat{f}_k \\ h(\vec{x}) \\ g(\vec{x}) \end{pmatrix} \tag{3.4}$$

At an optimal solution x^*, matrix H is cast from KKT conditions, specifically eqs. (2.2), (2.3), (2.4) and M_t-1 equations, $(f_k - \hat{f}_k)$. To compute the derivatives $\frac{\partial \vec{h}}{\partial \vec{\hat{f}}_\text{k}}$, H is differentiated w.r.t. $\hat{f}_\text{k}$, leading to:

$$\frac{\partial H}{\partial \vec{\hat{f}}_k} + \frac{\partial H}{\partial \vec{y}} \frac{\partial \vec{h}}{\partial \vec{\hat{f}}_k} = 0 \tag{3.5}$$

Given that $\frac{\partial H}{\partial \vec{\hat{f}}_\text{k}}$ is zero for eqs. (2.2),(2.3), (2.4) and -I for $\frac{\partial (f_k - \hat{f}_k)}{\partial \vec{\hat{f}}_\text{k}}$, the derivatives $\frac{\partial \vec{h}}{\partial \vec{\hat{f}}_\text{k}}$ can be obtained by reformulating eq.(3.5) as:

$$\frac{\partial H}{\partial \vec{y}} \frac{\partial \vec{h}}{\partial \vec{\hat{f}}_\text{k}} = \overbrace{\begin{bmatrix} \vec{0} \\ -I \end{bmatrix}}^{B} \tag{3.6}$$

By applying similar reasoning, derivatives $\frac{\partial \vec{h}}{\partial \vec{\lambda_k}}$ are computed.

An alternative way to compute $\frac{\partial \vec{h}}{\partial \vec{\hat{f}}_k}$, $\frac{\partial \vec{h}}{\partial \vec{\lambda_k}}$ is by applying the implicit function theorem to $y^* = h(\hat{f}_k)$. This method was derived by O. Schmidt and V. Schulz and is presented in Appendix C, [19].

For two-objective problems, the next optimal point x^* is approximated from eq.(3.3) as:

$$\mathbf{x}^*{}_{\text{predicted},i+1} = \mathbf{x}_i + \frac{\partial \vec{x}}{\partial \hat{f}_2} \delta \hat{f}_2 \tag{3.7}$$

3.2.5 Correction-Step

The Correction-step follows the Prediction-Step to retrieve the Pareto point, corresponding to the $\hat{f}_k$ value. Thus, the Correction-step is formulated in accordance to the minimization problem set in eqs. (3.1).

f_1 must be minimized and $f_k - \hat{f}_k$ is treated as a constraint that must be met at the optimal solution. In a two-objective problem, the target is set as $\hat{f}_2$ and with no other constraints imposed, the Correction-step is:

$$L(\vec{x}, \vec{\lambda}_{f_k}) = f_1(\vec{x}) - \lambda_{f_2}(f_2(\vec{x}) - \hat{f}_2) \tag{3.8}$$

The Correction-step, can be implemented by either the ALM or the SQP method as described in Appendix B and section 2.2.

3.2.6 Formulation of the Algorithm for Two Objectives

The algorithm of the GB method will be first formulated, for 2 objectives, which entails one target constraint $\hat{f}_2$. The two-objectives optimization algorithm reads as:

Algorithm 2 GB method's Algorithm, in a two-objective optimization problem

```
Require: Initial Point coordinates $\vec{x_{so}}$, SQP initialization, Variable Boundaries,
    Initial $\hat{f}_{2\text{INITIAL}}$ and $\delta\hat{f}_2$, number of Elite Pareto Points
 1: Counter of Elite Points, im ← 0
 2: Go-to-Pareto step Applied:
 3: |  Solve Optimization Problem, eq. (3.8) → $\vec{x_1}^*$
 4: Save First Pareto Point
 5: while Counter of Elite Points < nElite do
 6:     Counter of Elite Points, im ← im + 1
 7:     Move-on-Pareto step Applied:
 8:     |  Estimate $\hat{f}_{2,i+1} \leftarrow \hat{f}_{2,i+1} + \delta\hat{f}_2$
 9:     |  Prediction-step, eq. (3.7) → $\vec{x}^*_{i+1,\text{predicted}}$
10:     |  Correction-step, solve eq. (3.8) → $\vec{x}^*_{i+1}$
11:     |  Save Correction-Step Pareto point
12: end while
```

A visual representation of the Prediction-Correction Method is shown in Figure 3.2.

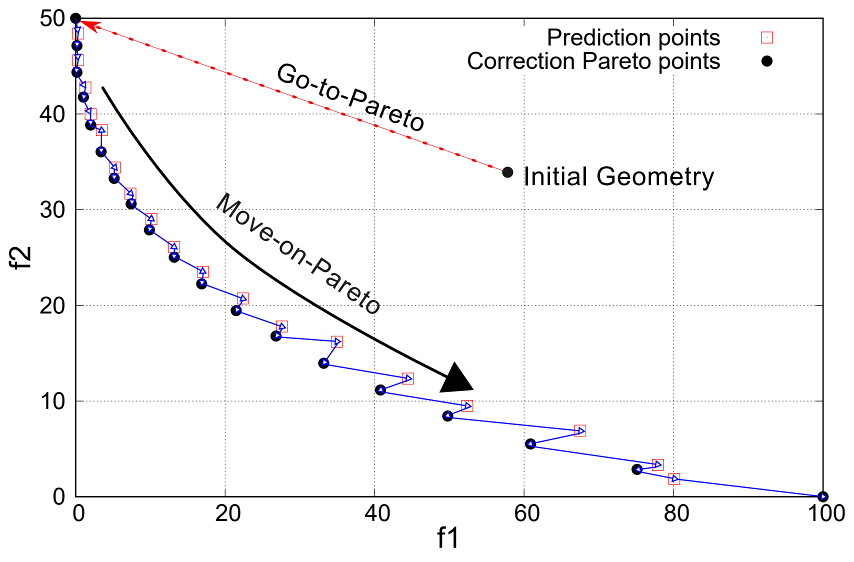


Figure 3.1: Visualization of the GB Method for a two-objective optimization problem. Blue arrows demonstrate the gaps between each Correction point and the corresponding Prediction point.

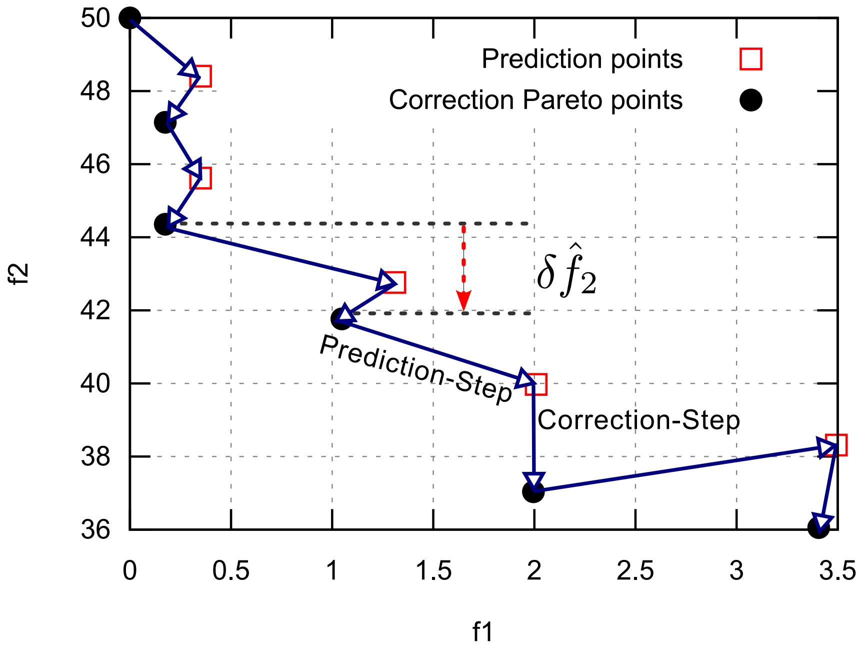


Figure 3.2: Visualization of the Prediction-Correction scheme in a two-objective optimization problem.

3.3 Mathematical Applications of the GB Method

In the remaining of this chapter, the GB method will be applied to track the Pareto front for a group of mathematical BPs. The efficiency of this method will be assessed when equality and inequality constraints are also set.

3.3.1 BP 1: Bihn and Korn Problem

The first mathematical BP, for assessing the efficiency of the GB method is the Bihn and Korn problem, [20] which is defined as following:

$$\begin{aligned} \text{Minimize:} \quad & f_1(x_1,x_2) = 4x_1^2 + 4x_2^2, \\ & f_2(x_1,x_2) = (x_1-5)^2 + (x_2-5)^2, \end{aligned} \tag{3.9}$$

The problem is subject to the following inequality constraints:

$$\begin{aligned} g_{1,2}(x_1,x_2) &= x_1 \in [0,5] \\ g_{3,4}(x_1,x_2) &= x_2 \in [0,3] \\ g_5(x_1,x_2) &= (x_1-5)^2 + x_2^2 \leq 25, \\ g_6(x_1,x_2) &= (x_1-8)^2 + (x_2+3)^2 \geq 7.7. \end{aligned} \tag{3.10}$$

The first derivatives ∇f_1 and ∇f_2 of the function are analytically computed as:

$$\frac{\partial f_1}{\partial x_1} = 8x_1, \quad \frac{\partial f_1}{\partial x_2} = 8x_2, \tag{3.11}$$

$$\frac{\partial f_2}{\partial x_1} = 2(x_1-5), \quad \frac{\partial f_2}{\partial x_2} = 2(x_2-5). \tag{3.12}$$

Both the ALM and the SQP method are used to track the Pareto front. In both cases, all inequality constraints defined in eqs. (3.10) are treated with the generalized ALM Algorithm as defined in Appendix B and the active-set SQP method in subsection 2.2.2.

After retrieving the initial Pareto point, (Go-to-Pareto Step) at (f_1, $\hat{f}_{2\text{INITIAL}}$)= (0.0, 50.0) by minimizing f_1, the Pareto front is tracked using the Move-on-Pareto step. The initialization of algorithm parameters set to track the Pareto Front is displayed in Table 1.

Parameter	Value
$\delta\hat{f}_2$	-2.5
Hessian Diagonals	1.0

Table 1: BP 1: Initialization of Optimization Parameters.

By minimizing f_2, the last extreme point on the Pareto front can be tracked; however, this incurs additional computational cost, as a second "Go-to-Pareto" step must be solved. The optimization process terminates when the optimal points between the two extreme points of the front are identified for the user-defined step $\delta\hat{f}_2$. Nevertheless, in all case studies of this Diploma Thesis, the last value of $\hat{f}_2$ to be tracked and the maximum number of elite points are user-defined, as the Pareto fronts are known a priori from the results provided by EASY. For BP 1 a total of 19 Pareto points are tracked.

For the ALM method:

The Augmented Lagrangian function is defined as following:

$$L(\vec{x}, \vec{\lambda_f}, \vec{\lambda_h}, \vec{\mu}) = f_1(\vec{x}) - \lambda_{f_2}(f_2 - \hat{f}_2) - \sum_{i=1}^{M_g} \mu_j g_j(\vec{x}) \tag{3.13}$$

where the set of λ_k or μ_k and ω_k selected to initialize the algorithm is displayed in Table 2.

λ_k	**Value**	ω_k	**Value**
λ_1	50.0	ω_1	500.0
λ_2	0.0	ω_2	50000.0
λ_3	0.0	ω_3	50000.0
λ_4	0.0	ω_4	500.0
λ_5	0.0	ω_5	0.5
λ_6	0.0	ω_6	0.5

Table 2: BP 1: Initialization Parameters for the ALM applied to the Correction-step.

The update of the design variables $\vec{x}$ using the ALM is performed by using BFGS. The hessian of the Prediction-Step $\nabla^2_{xx}L$ in eqs. (3.4), (3.5) is approximated using the SR-1 method, in which case S and Y are computed as:

$$S_i = \vec{x}_i - \vec{x}_{i,\text{prediction}} \quad , \quad Y_i = \nabla L_{\text{Correction-step}} - \nabla L_{\text{Prediction-step}} \tag{3.14}$$

The Pareto Front obtained by applying the GB method, with the ALM used in the Correction-step is displayed in Figure 3.3.

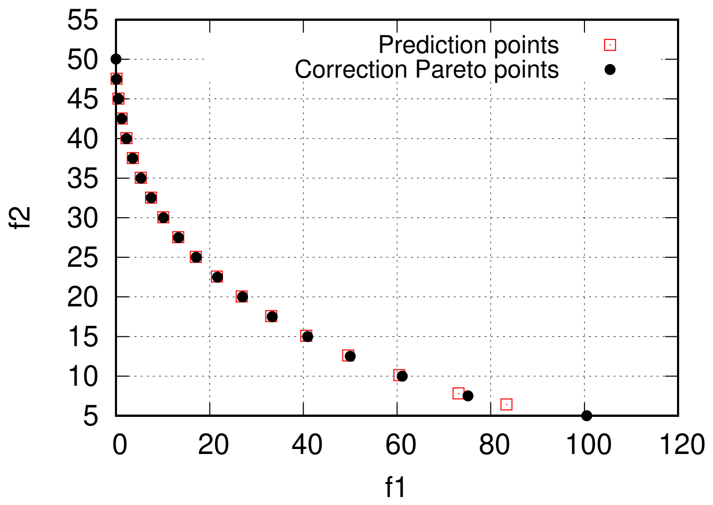


Figure 3.3: BP 1: Pareto points using the ALM algorithm in the Correction-step.

For the SQP method:
For the application of the SQP method, the second derivatives of eq. (3.9) are computed by the SR-1 method, Figure 3.4.

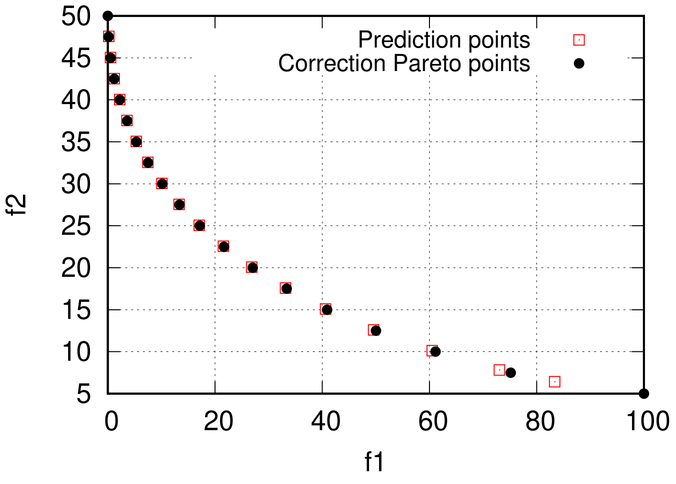


Figure 3.4: BP 1: Pareto points using the SQP algorithm in the Correction-step, with SR-1 to approximate the hessian.

Optimization using the Exact Hessian:
The SQP method can also be applied in such a mathematical problem in which all derivatives of any order can be computed analytically. The exact hessian at each optimization cycle is computed as:

$$H = \begin{bmatrix} \frac{\partial^2 f_1}{\partial x_1^2} - \lambda_{f_2}\frac{\partial^2 f_2}{\partial x_1^2} - \mu_5\frac{\partial^2 g_5}{\partial x_1^2} - \mu_6\frac{\partial^2 g_6}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} - \lambda_{f_2}\frac{\partial^2 f_2}{\partial x_1 \partial x_2} - \mu_5\frac{\partial^2 g_5}{\partial x_1 \partial x_2} - \mu_6\frac{\partial^2 g_6}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} - \lambda_{f_2}\frac{\partial^2 f_2}{\partial x_2 \partial x_1} - \mu_5\frac{\partial^2 g_5}{\partial x_2 \partial x_1} - \mu_6\frac{\partial^2 g_6}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} - \lambda_{f_2}\frac{\partial^2 f_2}{\partial x_2^2} - \mu_5\frac{\partial^2 g_5}{\partial x_2^2} - \mu_6\frac{\partial^2 g_6}{\partial x_2^2} \end{bmatrix}$$

or:

$$H = \begin{bmatrix} 8 - 2\lambda_{f_2} - 2\mu_5 - 2\mu_6 & 0 \\ 0 & 8 - 2\lambda_{f_2} - 2\mu_5 - 2\mu_6 \end{bmatrix}$$

The Pareto front tracked, using the exact hessian in (2.10) is shown in Figure 3.5.

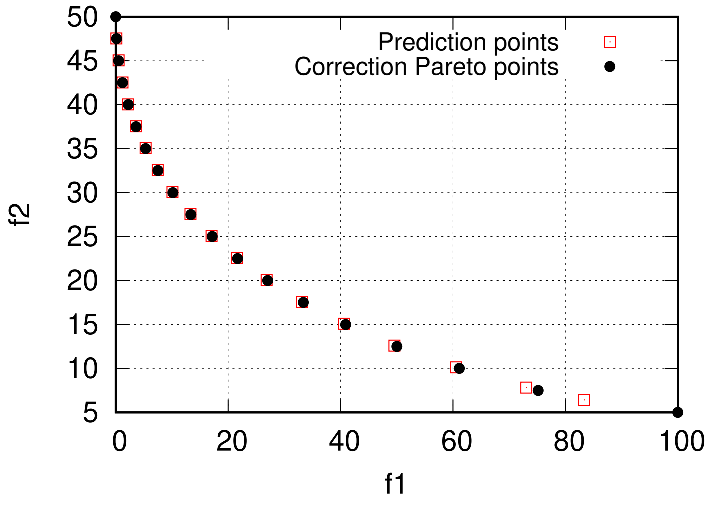


Figure 3.5: BP 1: Pareto points using the SQP algorithm, with exact hessian computation.

As shown in Figures 3.4, 3.5, the optimal points tracked using the exact hessian in the SQP are almost identical to those tracked using the SR-1.

Comparison of the SQP and ALM used in BP 1:

The computational cost of the SQP and ALM method in BP 1 is assessed based on the number of optimization cycles used in the Correction-step.

The comparison between the two methods is shown in Table 3.

Method Used	**Optimization Cycles**
SQP with the exact hessian	52
SQP method with hessian updated from SR-1	79
ALM with BFGS update step	288

Table 3: BP 1: Comparison of SQP method and ALM with BFGS used in the update step of the design variables.

Table 3 demonstrates that the SQP method, when using the exact hessian, requires fewer optimization cycles compared to approximating the Hessian using the SR-1 method. However, this comparison does not take into account the computational cost associated with the computation of exact second derivatives in CFD problems. As a result, the accuracy of the Quasi-Newton method for approximating the hessian is considered satisfactory, since both approaches yield identical Pareto fronts.

In BP 1, the SQP method with SR-1 yields a more accurate result in less optimization cycles, than ALM. Hence, in the following chapters, the SQP method is used in the Correction-step.

3.3.2 BP 2: Fonseca and Fleming Problem

The second BP to be optimized is the Fonseca and Fleming problem, [4]. This problem yields a non-convex Pareto front.

It is defined as:

$$\begin{aligned}\text{Minimize:}\quad & f_1(\vec{x}) = 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}}\right)^2\right), \\ & f_2(\vec{x}) = 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}}\right)^2\right) \\ \text{where}\quad & \vec{x} = (x_1, x_2), \\ & -4 \leq x_i \leq 4 \quad \forall \quad i = 1, 2.\end{aligned} \tag{3.15}$$

The first derivatives ∇f_1 and ∇f_2 of the functions are:

$$\frac{\partial f_1}{\partial x_i} = 2\left(x_i - \frac{1}{\sqrt{n}}\right)\exp\left(-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}}\right)^2\right), \tag{3.16}$$

$$\frac{\partial f_2}{\partial x_i} = 2\left(x_i + \frac{1}{\sqrt{n}}\right)\exp\left(-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}}\right)^2\right) \tag{3.17}$$

The bounds of design variables in eq. (3.15) are treated as constraints and thus are integrated within the objective function using the active constraints SQP method, as described in subsection 2.2.2. The second derivatives of the function, eq. (3.15) are approximated using the SR-1 method.

After retrieving the initial Pareto point (Go-to-Pareto step), at $(f_1,\hat{f}_{2\text{INITIAL}})=$ (0.001, 0.99) by minimizing f_1, the Pareto front is tracked using the Move-on-Pareto step. The initialization of the hessian for the SQP algorithm and the rest of the optimization parameters are displayed in Table 4. The final optimal point of the front is found by minimizing f_2, and is located at (1.0,0.0). Therefore, given the user-defined $\hat{\delta}f_2$, a maximum number of tracking 39 Pareto points was set.

Parameter	**Value**
$\delta\hat{f}_2$	-0.025
Hessian Diagonals	1.0

Table 4: BP 2: Initial Optimization parameters.

The Pareto front of BP 2, tracked by the GB method, is displayed in Figure 3.6.

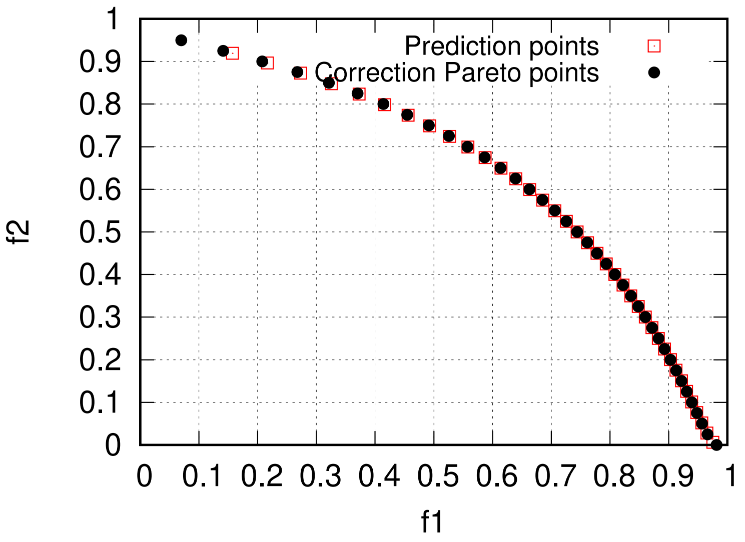


Figure 3.6: BP 2: Pareto points using the SQP algorithm in the Correction-step, with SR-1 method.

Optimization using the Exact Hessian:
The SQP method can also be applied as a test to see the role of the accuracy in computing the hessian matrix, by using the exact hessian at each optimization cycle, as the second derivatives of this function are computed as follows:

$$H = \begin{bmatrix} \frac{\partial^2 f_1}{\partial x_1^2} - \lambda_{f_2}\frac{\partial^2 f_2}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} - \lambda_{f_2}\frac{\partial^2 f_2}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} - \lambda_{f_2}\frac{\partial^2 f_2}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} - \lambda_{f_2}\frac{\partial^2 f_2}{\partial x_2^2} \end{bmatrix}$$

Using the exact hessian in the SQP yields the following Pareto front presented in Figure 3.7. The non-convex Pareto front tracked with the SR-1 method is accurate in locating the same front tracked with the use of the exact hessian matrix when applying the SQP.

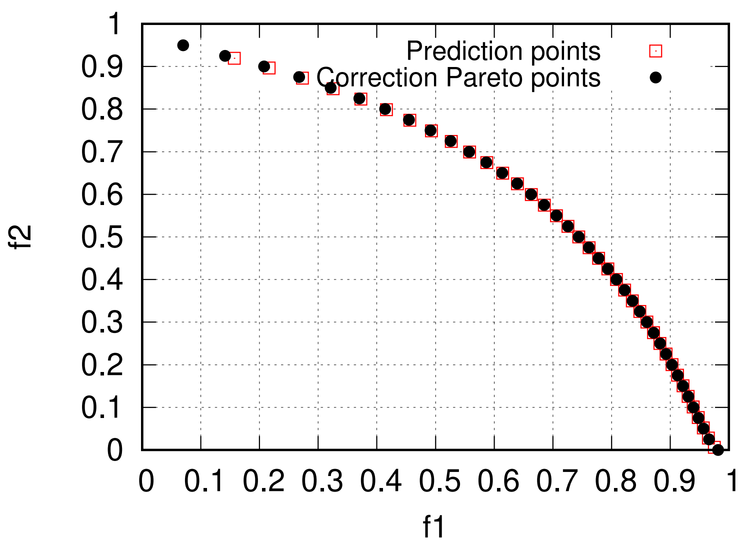


Figure 3.7: BP 2: Pareto points using the SQP algorithm in the Correction-step, with exact hessian.

Chapter 4

Application of GB method in External Aerodynamic ShpO

4.1 Case Description

This CFD case is concerned with the ShpO of an isolated airfoil in inviscid flow, starting from the NACA 4415 profile, for maximum lift and minimum drag (coefficients). A second variant of this case is presented in section 4.3, where the pitching moment coefficient (C_M) must be equal to zero and it is treated as an additional equality constraint.

The Pareto front of the objective function is tracked using the GB method. The GB method's results are compared with those of EAs (obtained using the EASY software), and its accuracy and computational cost are assessed. The sensitivity derivatives of the objective functions are computed using the continuous adjoint method by running the flow (CFD) and adjoint solver PUMA developed by PCOpt/NTUA. The external flow is inviscid and, thus, the Euler equations and their adjoints are numerically solved.

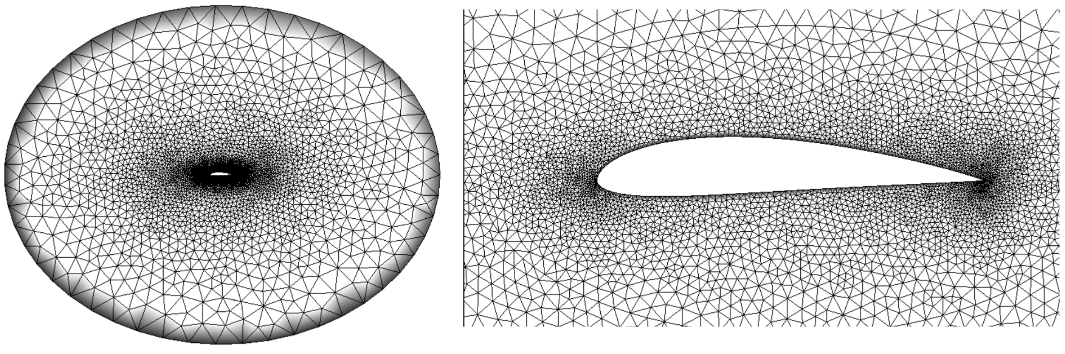


Figure 4.1: Airfoil case two-objectives: CFD mesh (left), and close up view around the airfoil (right picture).

The mesh around the airfoil is unstructured, consisting of approximately 6500 nodes, Figure. The farfield boundary is located at the distance of approximately 10 airfoil chords. The farfield or free-stream flow conditions are: flow angle $\alpha_\infty = 2°$, and Mach number $M_\infty = 0.8$.

The airfoil shape is parameterized using a NURBS lattice, consisting of 10x7 control points, 4x3 of which are free to be displaced, Figure 4.2.

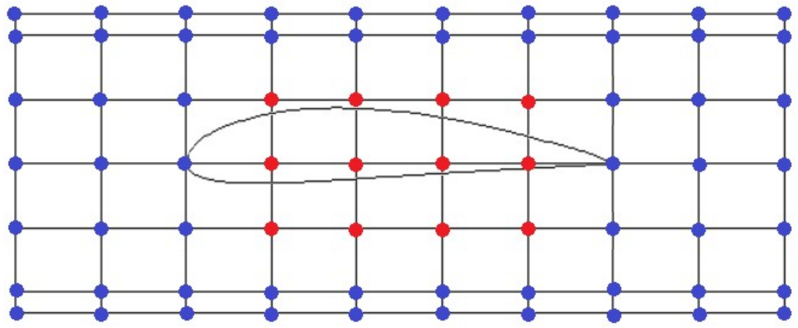


Figure 4.2: Airfoil case two-objectives: NURBS control lattice, blue points remain fixed while red control points are allowed to move to normal-to-the-chordwise direction.

The y (vertical) coordinates of the 12 red control points of Figure 4.2 constitute the design variables. Therefore, these points are allowed to move only in the vertical (normal-to-the chord) direction. In order for NURBS lattice points not to overlap during the optimization process, design variables' bounds within $\pm 0.05c$ around their initial values are introduced.

4.2 Optimization without the C_M Constraint

The ShpO case without a zero C_M constraint will be presented first. The GB method begins with the Go-to-Pareto step to track the first point on the front. Starting with non-optimal aerodynamic coefficients of the airfoil, $C_D = 0.06$ and $C_L = 0.985$, the Go-to-Pareto step tracks a first point on the front, from which the Move-on-Pareto step will be applied with a step size $\delta \hat{f}_2$. Based on the results of the EA an initial target value of $\hat{f}_2 = 0.35$ is chosen with a step size of $\delta \hat{f}_2 = 0.10$. Therefore, the GB method tracks optimal points across the same range of values as the EA.

The Move-on-Pareto step then sets to track the front by using the Prediction-Correction scheme. Since the value range of f_2 for non-dominated solutions is obtained from EASY, a threshold of 15 points is set for tracking by the GB method. These points correspond to values of C_L ranging from 0.33 to 1.63, beyond which drag losses render optimized airfoils as unacceptable. The SQP algorithm is used in the Correction-step. The problem to be minimized is formulated as:

minimize f:

$$f = \begin{cases} f_1(\vec{x}) = C_D(\vec{x}) \\ f_2(\vec{x}) = -C_L(\vec{x}) \end{cases} \tag{4.2}$$

The objective function to be minimized is the Lagrangian, reading:

$$L(\vec{x}, \vec{\lambda}_{f_2}) = f_1(\vec{x}) - \lambda_{f_2}(f_2(\vec{x}) - \hat{f}_2) \tag{4.3}$$

The initial values set for the parameters of the GB method are displayed in Table 5.

Parameter	Value
$\delta\hat{f}_2$	-0.10
Hessian Diagonals	10.0

Table 5: Airfoil case two-objectives: Initial Values for tracking the Pareto front.

The Algorithm used for the GB optimization is:

Algorithm 3 GB Method Algorithm, applied to Inviscid External Airflow case

```
Require: Initial Point coordinates $\vec{x}$, Hessian matrix initialization for the
    Correction-step, design Variable Bounds, Initial $\hat{f}_{2,\text{INITIAL}}$ and $\delta\hat{f}_2$, number of
    Elite Pareto Points, Convergence tolerance $\epsilon$
 1: Set counter of Elite Points, im ← 0
 2: Apply Go-to-Pareto step:
 3: while Convergence condition == FALSE do
 4:     | Adapt mesh to the current design variables $\vec{x}$
 5:     | Solve primal equations
 6:     | Solve adjoint equations for Lift and Drag
 7:     | Update $\vec{x}$ and $\lambda_2$
 8: end while
 9: Apply Move-on-Pareto step:
10: while im < nElite do
11:     | im ← im + 1
12:     | Apply Pareto Prediction-Step
13:     | Apply Pareto Correction-Step
14:     while Convergence condition == FALSE do
15:         | Adapt mesh to the current design variables $\vec{x}$
16:         | Solve primal equations
17:         | Solve adjoint equations for Lift and Drag
18:         | Update $\vec{x}$ and $\lambda_2$
19:     end while
20: end while
```

Computational cost is measured in terms of Equivalent Flow Solutions (EFS), which stands for the number of times the CFD solver or its adjoint (primal and adjoint are considered to have the same cost) must be called.

The convergence criterion for the Correction-step is set w.r.t. KKT residuals, given the nature of the CFD problem the convergence threshold is set as 10^{-3} for $\hat{f}_2$. The optimization process is performed with a step-size of $\delta\hat{f}_2 = 0.10$ until $\hat{f}_2 = 0.70$. The next target tracked is $\hat{f}_2 = 0.75$, from which the Move-on-Pareto step resumes, and the front is tracked with the $\delta\hat{f}_2$ set. The step size was adjusted to effectively track points corresponding to the previously mentioned range of values, also identified by the EA, facilitating a comparison between the two approaches. The GB method yields the Pareto points of Figure 4.3, and the computational cost of the algorithm is summarized in Table 6, (in terms of EFS).

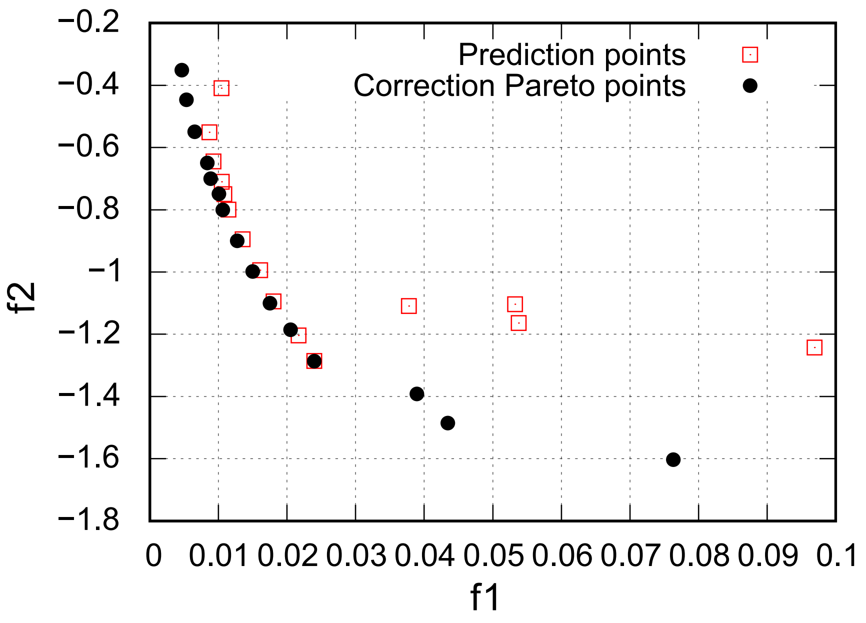


Figure 4.3: Airfoil case, two-objectives: Prediction points and Pareto Points obtained by the GB method.

Differential Equations	**EFS**
Primal	45
Adjoint	90
Total	135

Table 6: Airfoil case, two objectives: Computational cost (in EFS).

For the EA, 50 elite points were traced with the use of the built-in "RBF IF" metamodels. The computational cost of Algorithm 3 is compared with that of the EA, using EASY to compute the Pareto Front, in Table 8. The EA algorithm's results are displayed in Figure 4.4:

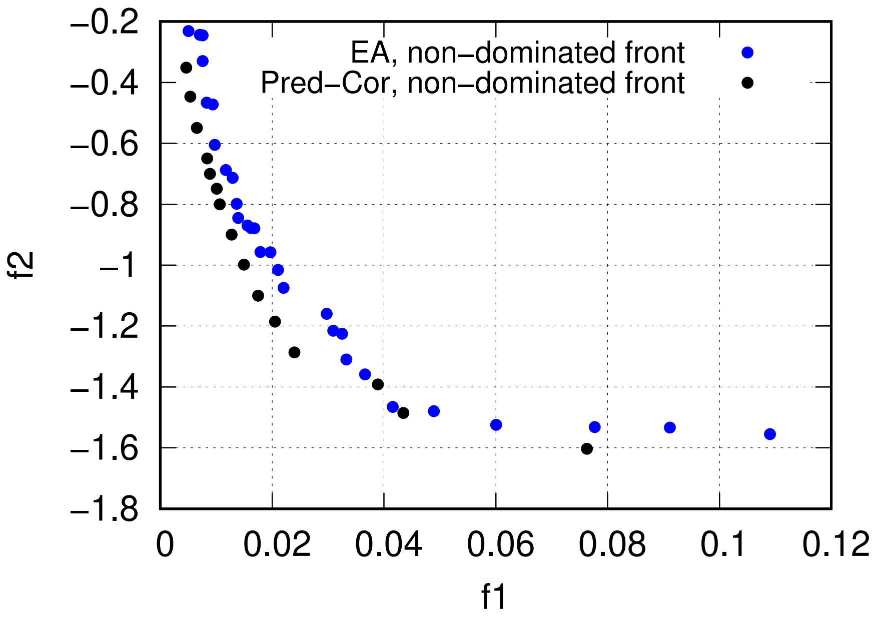


Figure 4.4: Airfoil case, two-objectives: Pareto front for EA Points and GB method.

Algorithm	**EFS**
EA with Metamodels	1000
GB Method	135

Table 7: GB method and EAs cost evaluation, (in terms of EFS).

As it can be seen, the GB method costs less evaluations; however, the EA algorithm computes 50 points, with a given threshold of 1000 evaluations and therefore it remains inconclusive whether the GB method developed requires less computational evaluations compared to EAs. Nonetheless, it can be deduced that the GB method has successfully tracked the Pareto front points, while retaining a reasonable computational cost.

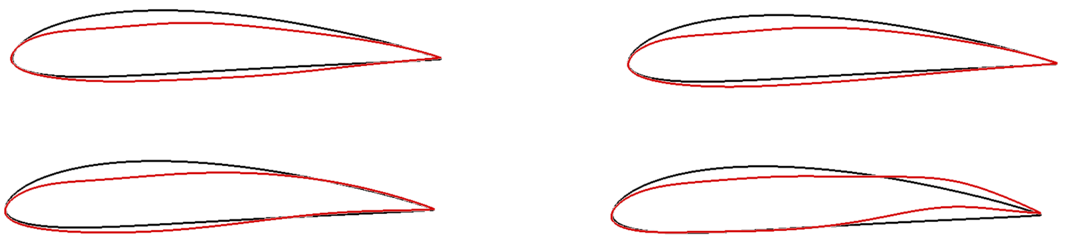


Figure 4.5: Airfoil case, two-objectives: ShpO, (black): baseline, from left to right, up to down (red): $(C_D, C_L) : (0.0119, 0.7989)$, $(C_D, C_L) : (0.0136, 0.8992)$, $(C_D, C_L) : (0.0187, 1.1100)$, $(C_D, C_L) : (0.0435, 1.4856)$.

The ShpO results of the GB method are presented in Figure 4.5. To achieve a higher C_L value curvature of both the suction and pressure side increase. Since the flow is assumed to be inviscid, the drag present is primarily due to losses caused by the formation of shock waves. By optimizing the shape of the airfoils, the intensity of these shock waves on the suction side is reduced. This minimizes drag losses, as demonstrated in Figure 4.7.

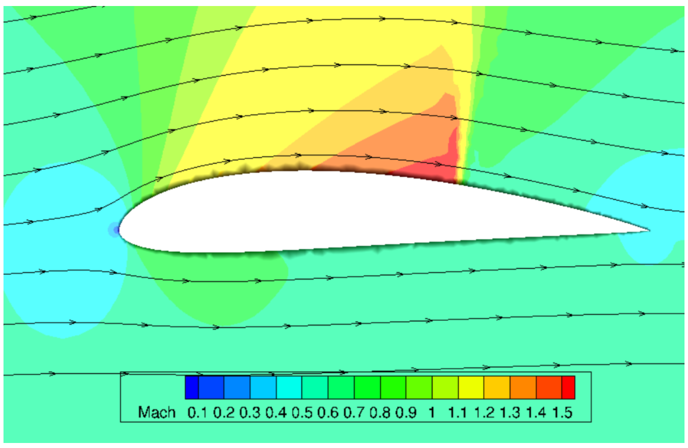


Figure 4.6: Airfoil case, two-objectives: Mach field around the baseline airfoil $(C_D, C_L) : (0.06, 0.985)$.

In Figure 4.7, Mach number fields of the optimized airfoils are presented.

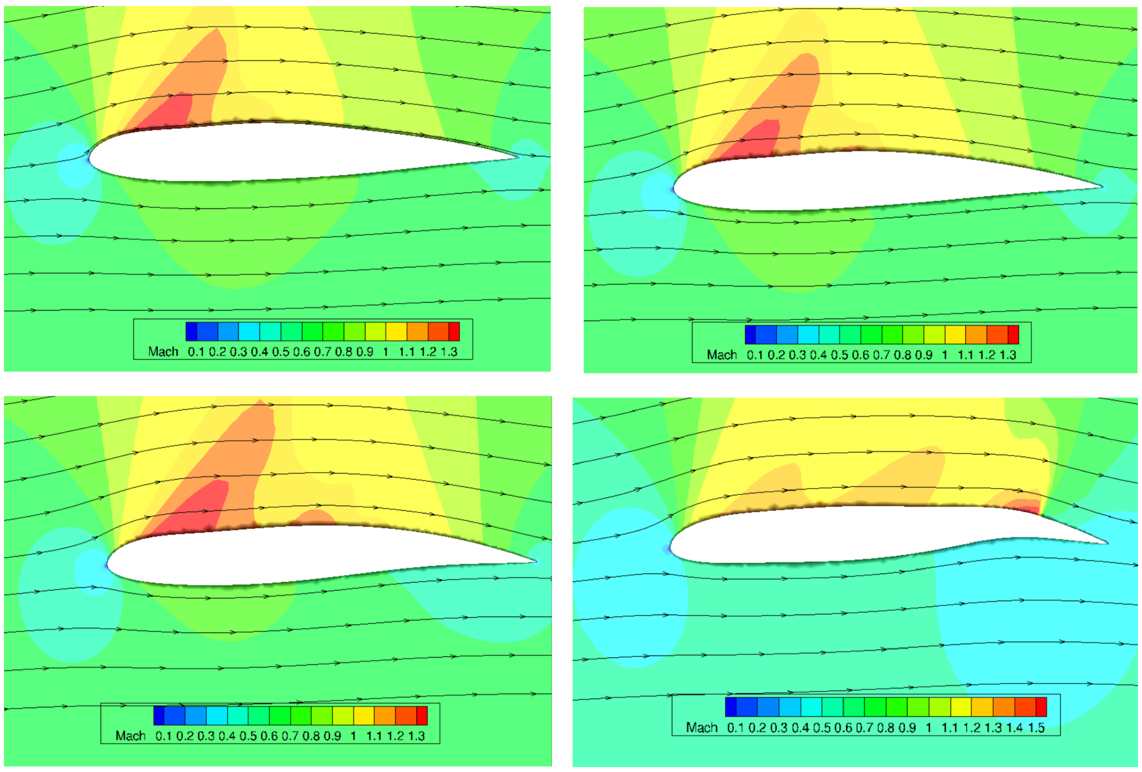


Figure 4.7: Airfoil case, two-objectives: ShpO, Mach flow fields, from left to right: $(C_D, C_L) : (0.0119, 0.7989)$, $(C_D, C_L) : (0.0136, 0.8992)$, $(C_D, C_L) : (0.0187, 1.1100)$, $(C_D, C_L) : (0.0435, 1.4856)$.

4.3 Optimization Constrained by C_M=0

The MOO problem presented in section 4.1 will be solved with the additional equality constraint of zero pitching moment coefficient (C_M=0). This is mathematically expressed by altering the Lagrangian as:

$$L(\vec{x}, \vec{\lambda}_{f_k}) = f_1(\vec{x}) - \lambda_{f_2}(f_2(\vec{x}) - \hat{f}_2) - \lambda_{C_M}(C_M(\vec{x})) \tag{4.5}$$

The additional equality constraint of the problem increases the computational cost per optimization cycle to 4 EFS. Particularly, computing first derivatives of C_M entails the solution of an additional set of adjoint equations of the function F_z, accounting for the pitching moment. Thus, at each optimization cycle 1 set of primal equations and 3 sets of adjoint equations are solved. For 7 Pareto points, the computational cost is displayed in Table 8:

Differential Equations	**EFS**
Primal	30
Adjoint	90
Total	120

Table 8: Airfoil case, two-objectives, $C_M = 0$: EFS cost evaluation of GB method.

The first target $\hat{f}_{2\text{INITIAL}}$=-0.33 is the same as in the case without the C_M constraint. This is done, in order to compare the results of the problem variants, computed both by the GB method. Initialization data, for the optimization parameters of this variation of the case are presented in Table 9, while the Pareto front obtained is shown in Figure 4.8.

Parameter	Value
$\delta\hat{f}_2$	-0.10
Hessian Diagonals	100.0

Table 9: Airfoil case, two-objectives, $C_M = 0$: Optimization Parameters for tracking the front.

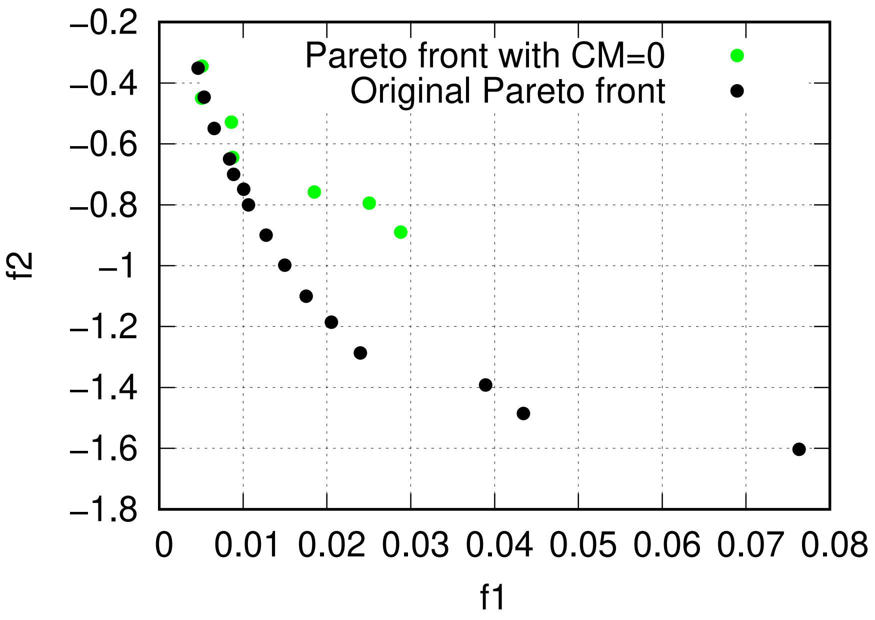


Figure 4.8: Airfoil case, two-objectives, $C_M = 0$: Pareto points with and without Pitching Coefficient constraint.

Finally, a comparison of the obtained shapes for the optimized airfoils is presented.

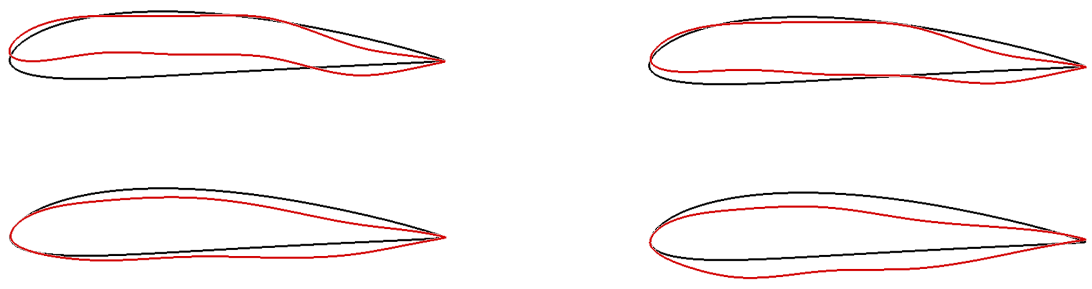


Figure 4.9: Airfoil case, two-objectives, $C_M = 0$: ShpO, (black): baseline geometry (C_D, C_L), from left to right, up to down, (red) optimized shape (Zero pitching moment coefficient): $(C_D, C_L) : (0.0288, 0.8903)$, $(C_D, C_L) : (0.0185, 0.7579)$, $(C_D, C_L) : (0.0086, 0.5284)$, $(C_D, C_L) : (0.0051, 0.3446)$.

One observation that can be made from the airfoils shown in Figure 4.9 is that, for higher C_L, they tend to alternate the curvature of both the suction and pressure side, while reducing the total area of the airfoil. This is made in order to counteract the negative pitching moment coefficient.

From the Pareto points obtained, it can be seen that optimizing the problem for a zero C_M coefficient comes at the cost of not significantly reducing C_D, for airfoils with a higher C_L value.

Chapter 5

Tracking Discontinuous 2-D Pareto Fronts Using GB Methods

5.1 Introduction

Tracking the Pareto front using GB Methods can pose certain challenges that need to be tackled during the optimization process. The Pareto front which represents solution points in most cases can be interpolated by a continuous curve as shown in subsection 1.3.2. When this Pareto curve abruptly terminates and then resumes at a subsequent range of values for $(f_1,\hat{f}_2)$, the Pareto front is referred to as discontinuous. This abrupt termination can be attributed to a change of the local curvature of the front, infeasibility within the function's domain or due to a binding constraint within a specified range of (f_1, f_2).

The major challenge in CFD applications when tracking Pareto is that it is not possible to be aware of discontinuous regions of it beforehand. Thus, the issue of excessive computational cost arises from conducting evaluations to track target points of a selected $\hat{f}_2$ which lies in a discontinuous region of it.

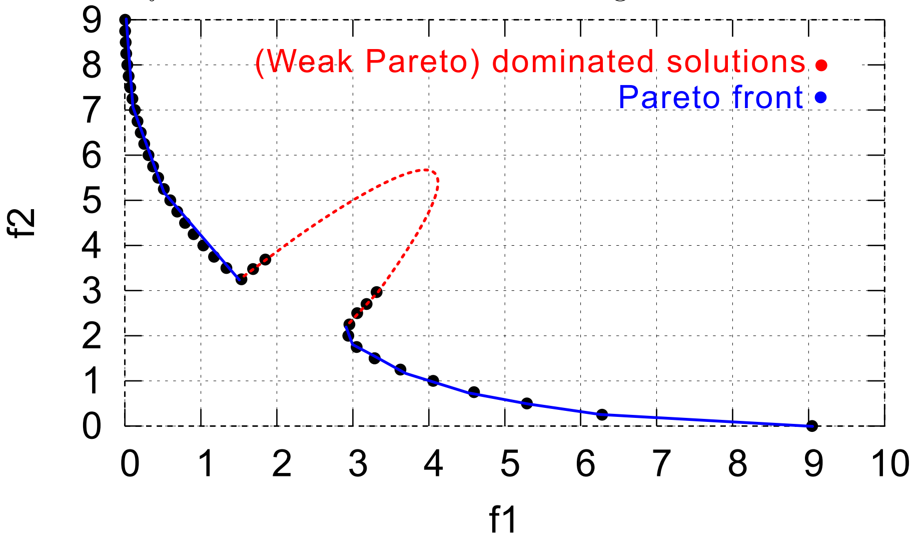


Figure 5.1: Visualization of a 2D discontinuous Pareto front.

Discontinuous Pareto fronts leading to divergence of the tracking method can be associated with the selected step size $\delta\hat{f}_2$. Specifically, if $\delta\hat{f}_2$ is large enough to bypass a range of (f_1,$\hat{f}_2$) values where the front is discontinuous, the Pareto front can be often tracked without being affected by the discontinuity.

An existing method to tackle this challenge is that of applying a Pareto filter after tracking every new Pareto point, which is presented in the next subsection, [5]. In the subsequent sections of this chapter, a new method will be presented and assessed for successfully tracking two-objective discontinuous Pareto Fronts while avoiding excessive objective function evaluations. The method consists of detecting a potential discontinuity and successfully tracking the front. Mathematical BPs have been solved using the proposed method.

5.1.1 Applying a Pareto Filter

An algorithm that can be used to detect discontinuous Pareto fronts is the Pareto filter which detects solutions that are dominated by other Pareto points, [5].

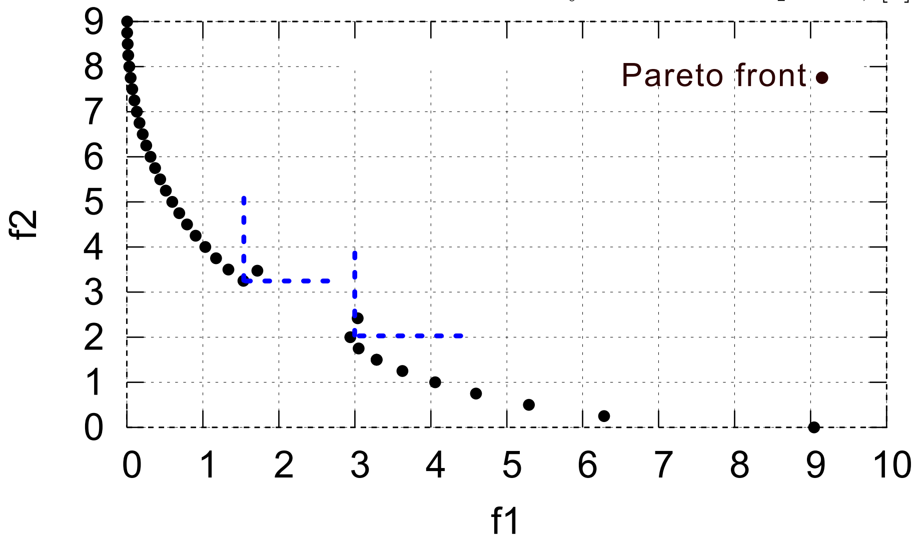


Figure 5.2: Visualization of the Pareto Filter, where solutions located within the area bounded by the blue dotted horizontal and vertical lines are dominated by the corresponding point, from which the lines are drawn.

The Pareto filter algorithm is used after tracking a new point to check if it is dominated and thus omitted from the Pareto front, or if it dominates an existing point that has been previously added to the front. The filter recursively checks whether each identified solution, when compared to the others, satisfies eq. (1.12). In the developed software, the Pareto filter acts as a minimum precaution, as it is applied after a new point is tracked, to ensure that any dominated solution is removed from the front.

5.2 A New Method to Detect Discontinuous Pareto Fronts

The first step in successfully tracking a discontinuous front is detecting a potential region where the front is discontinuous. The proposed method makes use of insufficient approximations to Pareto points yielded by the Prediction-step and correlates them with potential discontinuous regions of the front.

The GB method utilizes the Prediction-Correction scheme, as formulated in section 3.2, to track points along the Pareto front. The Prediction-step yields a satisfactory approximation of the optimal point given that $x^* = x^*(\hat{f}_2)$ can be approximated using a first order Taylor Series, eq. (3.7). Therefore, when the Taylor expansion of $x^* = x^*(\hat{f}_2)$ is sensitive over its higher-order terms, a first-order approximation of the next optimal point would yield inaccurate results.

This occurs in discontinuous regions of the front, where local curvature changes. Therefore, starting from this inaccurate and dominated Prediction point, the Correction-step will not be able to refine it to the optimal solution for the selected $\hat{f}_2$ target, as all solutions in a discontinuous region are either dominated or infeasible.

A correlation between the curvature of $(f_1,\ f_2)$ at each optimal point and the success of the Prediction's step approximation can be established. Indeed, eq. (3.7) underlines the need for function $x^* = x^*(\hat{f}_2)$ to be sufficiently approximated using a first-order Taylor series approach over a specified range $\delta\hat{f}_2$. Since every optimal solution x^* is a function of $x^* = x^*(f_1, f_2)$, any significant change in the curvature of f_1 or f_2 will affect it. Consider the intervals $(\hat{f}_2, \hat{f}_2 + \delta\hat{f}_2)$ or $(\hat{f}_1, \hat{f}_1 + \delta\hat{f}_1)$, which represent the range between two consecutive target points. If the curvature of f_1 or f_2 undergoes a notable change within these intervals, it follows from eq. (3.7) that the derivative $\frac{\partial\vec{x}}{\partial\hat{f}_2}$ will also exhibit a significant change, as:

$$\frac{\partial\vec{x}}{\partial\vec{\hat{f}_2}} = \left(\frac{\partial H}{\partial\vec{x}}\right)^{-1} \overbrace{\begin{bmatrix} 0 \\ -I \end{bmatrix}}^{B} \tag{5.1}$$

From eq.(5.1), it is deduced that the Prediction's step first order accuracy in a region where curvature of the front $(f_1^*,\ f_2^*)$ changes, will be poor. Therefore, a linear approximation eq. (3.7) is insufficient as $\frac{\partial\vec{x}}{\partial\hat{f}_2}$ significantly fluctuates.

A way to deem whether an approximated point is close to the optimal x^* or not is through the KKT conditions, eq.((2.2), (2.3), (2.4), (2.5), (2.6)), that must hold true for all optimal points. The residuals of KKT conditions are defined as:

$$\text{Residuals}_{predict,_{i+1}} = \begin{bmatrix} \nabla L(x^*_{\text{predict}_{i+1}}, \lambda^*_{f2,\text{predict}_{i+1}}, \mu^*_{\text{predict}_{i+1}}) \\ h(\mathbf{x}^*{}_{\text{predict}_{i+1}}) \\ g(\mathbf{x}^*{}_{\text{predict}_{i+1}}) \end{bmatrix} \tag{5.2}$$

Eq. (5.2), is updated after each Prediction-step, thus yielding new Prediction residuals of the KKT conditions. Unexpectedly high residuals signify a poor Prediction solution for the $\hat{f}_2$ and thus, a region of discontinuity is signified. Therefore, there is no need to realize the Correction-step for $\hat{f}_2$ and in this way SQP evaluations are avoided, which in CFD translates into EFS cost.

In order to determine whether the Prediction-step KKT residuals of eq.(5.2) are inadmissibly high, a method to calculate their variance at every optimization cycle, must be used. For this reason, the concept of online algorithms is proposed. In particular, the Welford's online algorithm is used, which is presented in the next section, [2], [3].

5.3 Computing Variance of KKT Residuals Sample

Computing the variance of a given sample is of fundamental role in computational statistics. Online algorithms refer to a class of algorithms that process data in a sequential manner. By definition, these algorithms do not have access to the entire data set at the outset. Online algorithms are particularly useful when the complete input is initially unknown or when memory limitations prevent storing the entire data set to be processed.

A widely known online Algorithm used to compute variance of a sample is Welford's algorithm. For a new element x_n, the following formulas are defined to update the mean and (estimated) variance of the sequence. Here, $\overline{x}_n = \frac{1}{n}\sum_{i=1}^{n} x_i$ denotes the sample mean of the first n samples $(x_1, \ldots, x_n)$, $\sigma_n^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x}_n)^2$ their biased sample variance, and $s_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x}_n)^2$ their unbiased sample variance.

$$\overline{x}_n = \frac{(n-1)\overline{x}_{n-1} + x_n}{n} = \overline{x}_{n-1} + \frac{x_n - \overline{x}_{n-1}}{n}$$

$$\sigma_n^2 = \frac{(n-1)\sigma_{n-1}^2 + (x_n - \overline{x}_{n-1})(x_n - \overline{x}_n)}{n} = \sigma_{n-1}^2 + \frac{(x_n - \overline{x}_{n-1})(x_n - \overline{x}_n) - \sigma_{n-1}^2}{n}$$

$$s_n^2 = \frac{n-2}{n-1}s_{n-1}^2 + \frac{(x_n - \overline{x}_{n-1})^2}{n} = s_{n-1}^2 + \frac{(x_n - \overline{x}_{n-1})^2}{n} - \frac{s_{n-1}^2}{n-1}, \quad n > 1$$

A better quantity for updating is the sum of squares of differences from the current mean, $\sum_{i=1}^{n}(x_i - \overline{x}_n)^2$, denoted, $M_{2,n}$:

$$M_{2,n} = M_{2,n-1} + (x_n - \overline{x}_{n-1})(x_n - \overline{x}_n)$$

$$\sigma_n^2 = \frac{M_{2,n}}{n}$$

$$s_n^2 = \frac{M_{2,n}}{n-1}$$

It is also common to denote $M_k = \overline{x}_k$ and $S_k = M_{2,k}$.

5.3.1 Applying Welford's Algorithm, to calculate Variance of sample of KKT Residuals

The GB method developed for tracking the Pareto front can be adapted to collect KKT condition residuals, as shown in eq. (5.2), after each Prediction-step approximates a new optimal point. To determine the initial mean and variance of the sample, it is recommended to collect at least the first 5 tracked prediction points. This process does not incur any additional computational cost. Each time a new Prediction point is approximated, its KKT residuals are added to the sample, and Welford's algorithm is used to update the sample's mean and variance accordingly.

If the sample of points is large enough, a Z-score test, [10] is then conducted to determine whether the newly added predicted point is an outlier in the sample, by using the following relation:

$$z > \frac{x_n - \overline{x_n}}{s_n} \tag{5.3}$$

If the Prediction point is identified as an outlier to the sample of collected KKT residuals, then the Correction-step applied to refine this solution to the optimal point is skipped. In such a case, it is recommended to use the methods described in section 5.4. If the sample consists of a small number of points a t-score test is recommended, [10]. Other non-parametric statistical tests, can also be applied, [14]

5.3.2 Algorithm Formulation for Detecting Discontinuities

The new method proposed in sections 5.2, 5.3 can be formulated into the following algorithm:

Algorithm 4 Detecting Discontinuities Algorithm

```
Require: Elite point im, Hessian matrix and Prediction-step Input
1: Perform Prediction-step
2: Estimate first-order KKT conditions' residuals, eq. (5.2)
3: Perform Welford's algorithm, section 5.3 and statistical test, 5.3.1, to identify
   outlier.
4: if Prediction Point is Outlier then
5:     Do not proceed to the Correction-Step
6:     Apply an algorithm to track discontinuous Fronts, section 5.4
7: end if
8: Iteration: $im \leftarrow im + 1$
9: Move-on-Pareto
```

5.4 Proposed Method to Move-on Discontinuous Fronts

After detecting a potential discontinuous region of the front at the selected $\hat{f}_2$ from algorithm 4, a method to track the front is suggested. The following three algorithms: Target-Objective jump, Swap Target-Objective and Back-tracking can be combined to constitute a method that can track such fronts.

5.4.1 Target-Objective jump

The first algorithm that can be applied after detecting a discontinuous region is that of Target-Objective jump. It is realized by selecting a different $\delta\hat{f}_2$. A new Correction-step is performed for the selected $\hat{f}_2$ from either the starting non-optimal point of the algorithm (baseline geometry in CFD) or from the most recent Pareto point tracked. The GB method for tracking the Pareto Front continues until the next discontinuous area be located or the selected number of optimal points has been tracked.

Using the Target-Objective jump algorithm presupposes that the new target $\hat{f}_2$ selected should be far from the interval where the discontinuity of the Pareto front occurs, and thus a $\delta\hat{f}_2$ large enough is selected to avoid yielding a point that falls into the discontinuous region. If this is not the case the jump will be repeated with a larger $\delta\hat{f}_2$, which in CFD amounts to the cost of solving primal and adjoint equations for all the iterations of the Correction-step.

In what follows, the Target objective jump are selected as a function of the current target-step $\delta\hat{f}_2$ that is used. The scaling of these jump must be selected, beforehand.

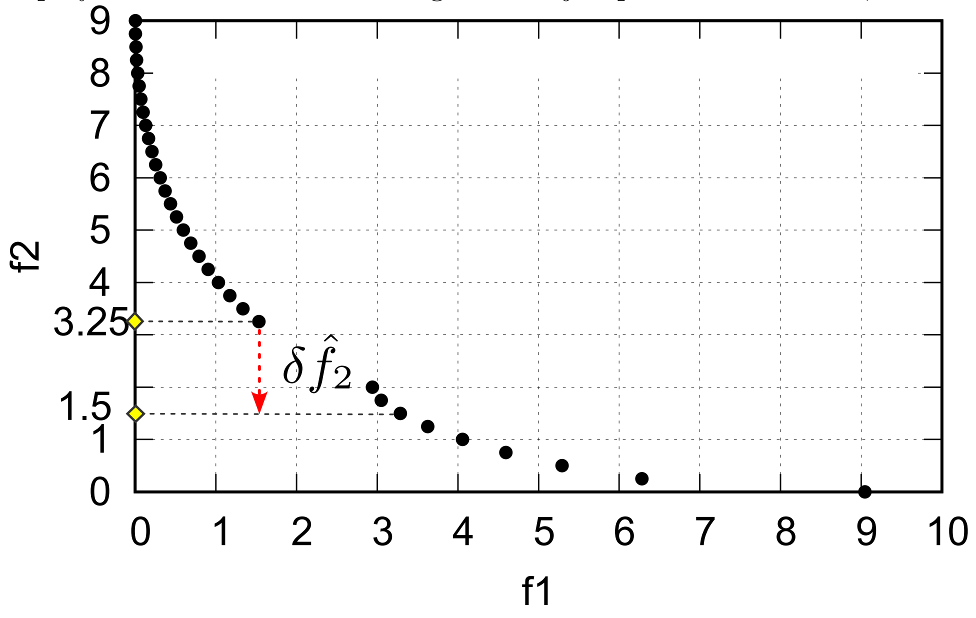


Figure 5.3: A demonstrative example of the Target-Objective jump algorithm technique to track Discontinuous Pareto Fronts. From $\hat{f}_2$= 3.25, to $\hat{f}_2$= 1.5, by selecting a $\delta\hat{f}_{2\text{Jump}} = 7\ \delta\hat{f}_2$.

In Figure 5.3, after locating the discontinuity of the Pareto front, a jump of $\delta\hat{f}_{2\text{Jump}} = 7\ \delta\hat{f}_{2\text{old}}$ is performed and the tracking of the front continues from this point on.

Algorithm 5 Target-Objective jump Algorithm

```
Require: Starting Point before the Jump (baseline or previous Pareto point), $\hat{f}_2$,
    convergence criterion for Correction-step $\epsilon$
1: while Correction-step not converged do
2:     | The next Target-Objective jump $\delta\hat{f}_{2\text{Jump}}$ is selected.
3:     | $\hat{f}_{2\text{new}} = \delta\hat{f}_{2\text{Jump}} + \hat{f}_2$
4:     | Run Correction-step with $\hat{f}_{2\text{new}}$
5: end while
6: Iteration: $im \leftarrow im + 1$
7: Use Back-tracking to track omitted Pareto Points, subsection 5.4.3.
8: Proceed with Tracking The Pareto Front
```

5.4.2 Swap Target-Objective

Another algorithm that can be utilized when a discontinuous region is encountered is that of swapping the objective and target of the Lagrangian function, (Swap Target-Objective).

For a two-objective problem, the Lagrangian reads as:

$$L(\vec{x}, \vec{\lambda}_{f_1}, \vec{\lambda}_{h_j}, \vec{\mu}) = f_2(\vec{x}) - \lambda_{f_1}(f_1 - \hat{f}_1) - \sum_{j=1}^{M_h} \lambda_{h_j} h_j(\vec{x}) - \sum_{i=1}^{M_g} \mu_j g_j(\vec{x}) \tag{5.4}$$

This algorithm can be successfully applied in cases where a region of $\hat{f}_2$ yields solutions that are not included in the solution domain of the objective function or are placed outside the Pareto front, due to the binding of constraints.

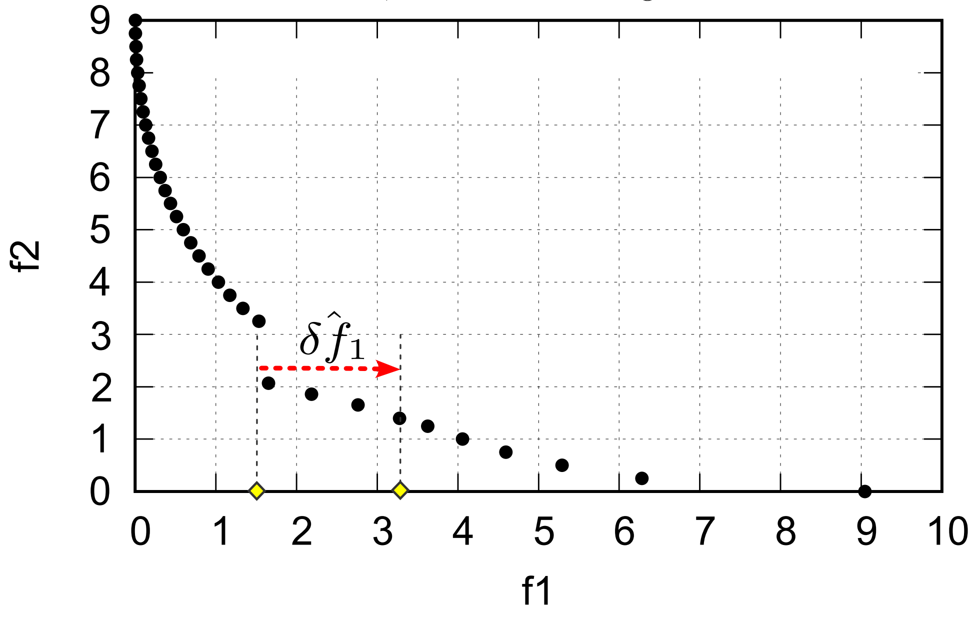


Figure 5.4: Implementation of Swap Target-Objective on a discontinuous front.

Similarly to the Target-Objective jump, the new $\hat{f}_1$ should be far from the discontinuity of the Pareto Front. Again, in case a new optimal point is not found after performing a Correction-step, the $\delta\hat{f}_1$ is doubled and the algorithm proceeds to perform once again a Correction-step.

5.4.3 Back-tracking

Back-tracking is a technique used when tracking the Pareto front and is applied after one of the two aforementioned algorithms of subsection 5.4.1, 5.4.2 have been used. Therefore, Back-tracking is used to track omitted Pareto points after a jump with a large $\delta\hat{f}_1$ or $\delta\hat{f}_2$ has been performed. It consists of inverting the tracking direction of the Pareto front in order to locate Pareto points for a range of $\hat{f}_2$ that was omitted, while performing a dis-proportionally large Target-Objective jump or selecting an $\hat{f}_1$ which omits an amount of Pareto points if the Swap Target-Objective algorithm is used.

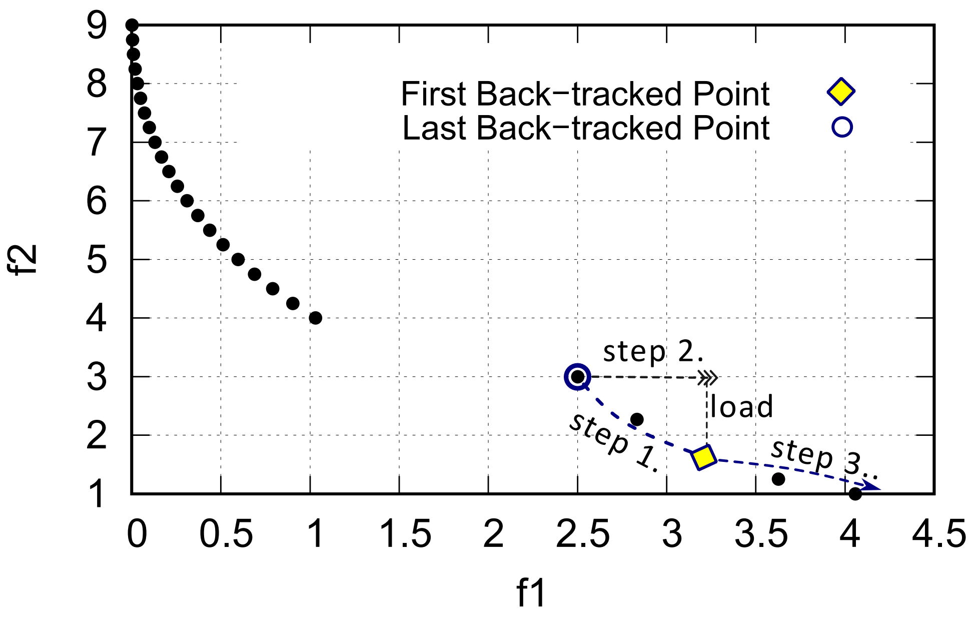


Figure 5.5: Demonstration of the Back-tracking technique for tracking the Pareto front after a Target-Objective jump has been performed. The symbols for the last Back-tracked Point and first Back-tracked Point will be used throughout the rest of the chapter.

The steps of the Back-tracking algorithm are depicted in Figure 5.5. The algorithm starts by transferring the data of its initial point after the jump (hessian matrix, first derivatives and (f_1, f_2) in a separate file). After that, the tracking direction is inverted until locating a new discontinuous region or reaching the old target $\hat{f}_2$, from which the jump was performed, (step 1.). The algorithm reverts back to its starting point and the hessian matrix as well as the derivatives of it are retrieved from the file they were stored (step 2.). In the final step, the tracking direction of the front is reverted to its original resuming the tracking of the front, (step 3.).

Algorithm 6 Back-tracking Algorithm

```
Require: Input Data: last tracked point ($f_1$, $\hat{f}_2$), Hessian matrix estimated by the
    SQP algorithm, $\delta\hat{f}_2$,
1: $\delta\hat{f}_2 \leftarrow -\delta\hat{f}_2$
2: Store Input Data in a separate file.
3: while Detecting discontinuity, algorithm 4 not satisfied do
4:     | Move-on-Pareto Step
5:     | Iteration: $im \leftarrow im + 1$
6: end while
7: Load Input Data, (step 2)
8: Proceed with Tracking The Pareto Front (step 3)
```

5.5 Algorithm Formulation of the Tracking Method

The proposed tracking method involves identifying potential discontinuities on the Pareto front and performing a jump to continue tracking solutions along the front. The algorithm for this method is outlined as follows:

Algorithm 7 Detecting and Tracking 2D Pareto fronts Algorithm

```
Require: Elite point im, Hessian matrix and Prediction-step Input
 1: Perform Prediction-step
 2: Estimate Prediction-step's KKT conditions' residuals, eq. (2.2),(2.3),(2.4)
 3: Perform statistical test, 5.3.1, to identify outlier Prediction point.
 4: if Prediction Point is Outlier then
 5:     Do not proceed to the Correction-Step
 6:     Starting from the last tracked Pareto point, apply either Target-Objective
        jump or Swap Target-Objective.
 7:     Apply Back-tracking algorithm to track omitted points due to a large jump.
 8: end if
 9: Iteration: $im \leftarrow im + 1$
10: Move-on-Pareto steps, from the point tracked after the jump.
```

5.6 Mathematical Applications, for Tracking Discontinuous Fronts

In order to assess the accuracy of the aforementioned method, three mathematical BPs are optimized, by using combinations of the algorithms presented in section 5.4. The three BPs and their optimization process are presented in the following subsections.

5.6.1 BP 3

The third BP to be minimized is defined as:

$$min : f_1(\vec{x}) = \sum_{i=1}^{3} \left((x_i - 2)^2 + \frac{\omega}{(x_1 - 3.5)^2 + \epsilon} \right)$$

$$f_2(\vec{x}) = \sum_{i=1}^{3} \left((x_i - 5)^2 + \frac{\omega}{(x_1 - 3.5)^2 + \epsilon} \right)$$

where $\vec{x}$ is a vector of 3 design variables, (x_1, x_2, x_3).

The gradients $\nabla f_1(\vec{x})$ and $\nabla f_2(\vec{x})$ w.r.t. $\vec{x}$ are:

$$\nabla f_1(\vec{x}) = \begin{bmatrix} 2(x_1 - 2.0) + \frac{-2\omega(x_1-3.5)}{(x_1-3.5)^2+\epsilon} \\ 2(x_2 - 2.0) \\ 2(x_3 - 2.0) \end{bmatrix}$$

$$\nabla f_2(\vec{x}) = \begin{bmatrix} 2(x_1 - 5) + \frac{-2\omega(x_1-3.5)}{(x_1-3.5)^2+\epsilon} \\ 2(x_2 - 5) \\ 2(x_3 - 5) \end{bmatrix}$$

Terms ω and ϵ are parameters of the problem, which regulate the span of the discontinuity region:

Parameter	Value
ϵ	10^{-3}
ω	10^{-1}

Table 10: BP 3: Values assigned to the parameters.

After retrieving the initial Pareto point (Go-to-Pareto step), at (f_1,$\hat{f}_{2\text{INITIAL}}$)= (0.00, 18.0) by minimizing f_1, the Pareto front is tracked using the Move-on-Pareto step, with a step of $\hat{\delta} f_2$= -0.5 tracking a total of 34 points.

Prediction-step's residuals of eq.(5.2) are collected and processed from the Detecting discontinuities algorithm, which estimates their variance and mean value after a new predicted point is added to the sample. After detecting an outlier to the sample, according to eq.(5.3), the GB method does not proceed to the Correction-step and executes a Target-Objective jump.

Initialization data for the BP are presented on Table 11.

Parameter	Value
$\delta \hat{f}_2$	-0.5
$\delta \hat{f}_{2\text{Jump}}$	6 $\delta \hat{f}_2$
$Zscore$	6.0
Hessian Initialization	1.0

Table 11: BP 3: Values of optimization parameters for tracking the Pareto front.

In Figure 5.6, the Prediction residuals of the KKT conditions are depicted for each Pareto Point tracked.

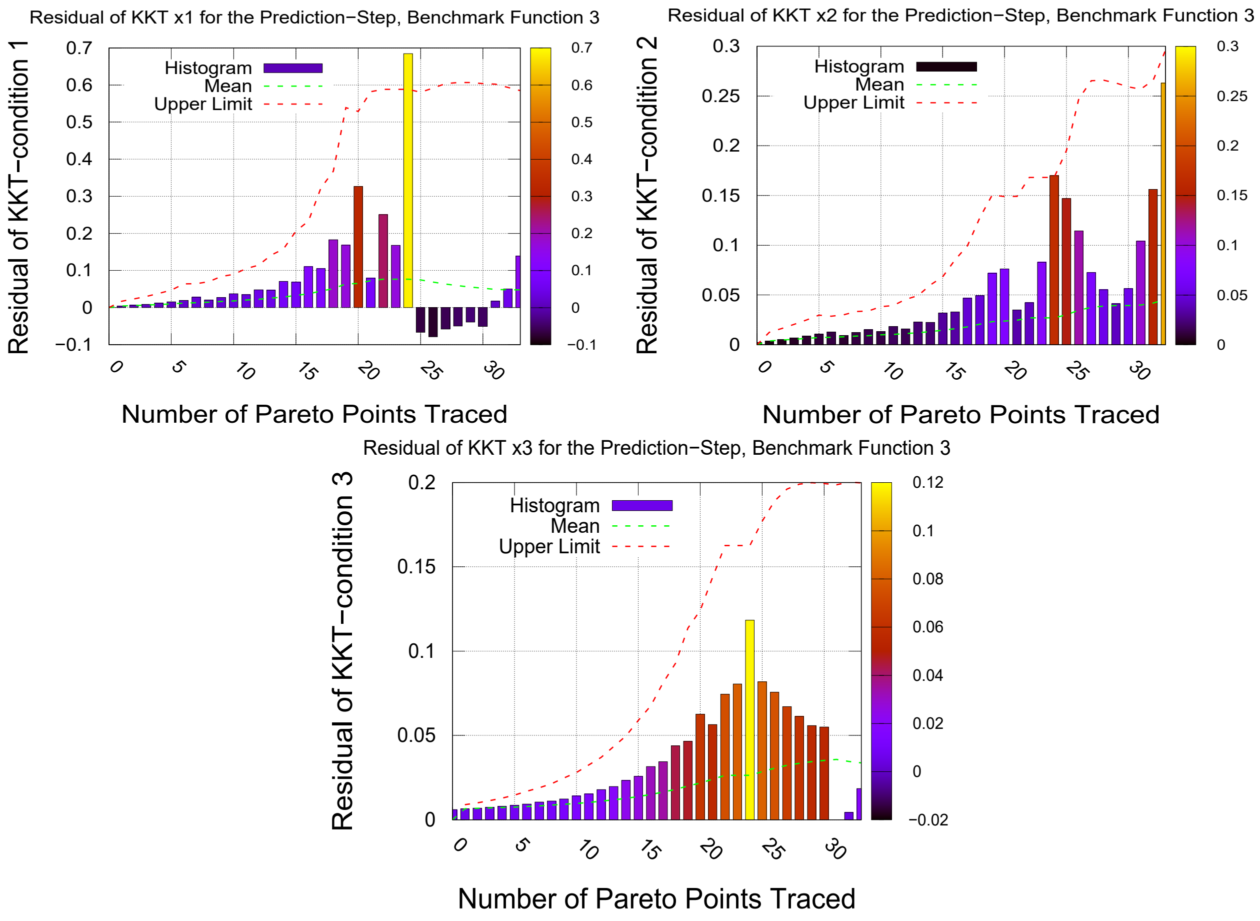


Figure 5.6: BP 3: Prediction-step residuals of KKT conditions.

In Figure 5.6, the Welford's algorithm detects an outlier residual at the KKT conditions imposed upon x_1 and x_2 at the 24th predicted Pareto point. This detection signals a discontinuous region in the Pareto front by the Detecting discontinuities algorithm, leading to the execution of the Target-Objective jump, with $\delta \hat{f}_{2\text{Jump}} = 6\delta \hat{f}_2$. After the jump, Back-tracking is used to retrieve points for the omitted $\hat{f}_2$ values.

The Pareto front tracked from the aforementioned initialization is displayed in Figure 5.7. Both Predicted points and Correction Pareto points are Displayed, while Figure 5.8 shows the Correction points tracked by the algorithm by utilizing the Target-Objective jump and the omitted points, due to the jump, located with Back-tracking.

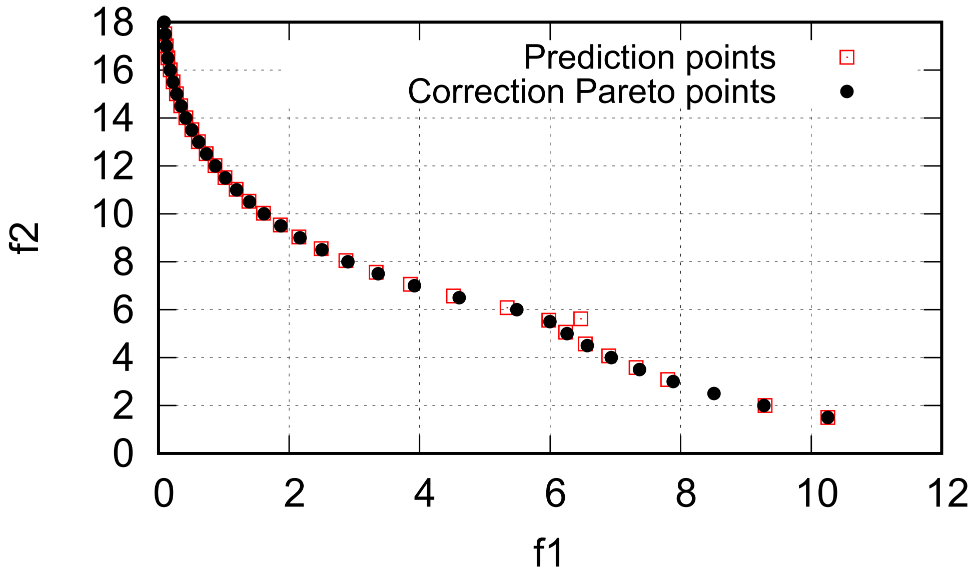


Figure 5.7 : BP 3: Prediction points and Correction Pareto points.

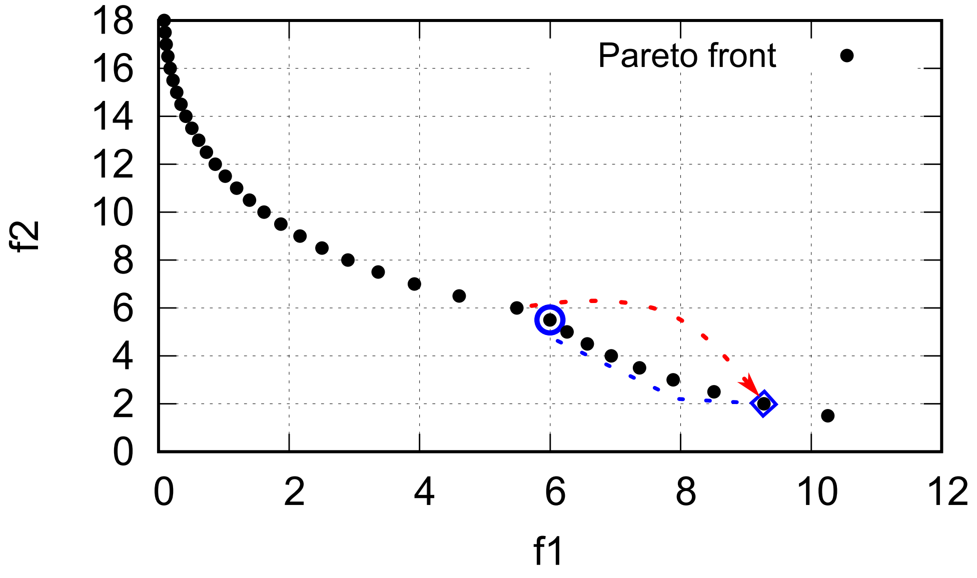


Figure 5.8: BP 3: Demonstration of Target-Objective jump (shown with red arrow) and Back-tracking (shown with blue arrow) algorithms for tracking the Pareto front, with Target-Objective jump of $\delta\hat{f}_{2\text{Jump}} = 6\delta\hat{f}_2$.

5.6.2 BP 4

The fourth BP to be minimized is:

$$minimize : f_1(\vec{x}) = \left(\frac{1}{2}(x_1 + x_2) - 2\right)^2 + \frac{\omega}{\left(\left(\frac{1}{2}(x_1 + x_2) - 3.5\right)^2 + \epsilon\right)} + (x_1 - x_2)^2$$

$$f_2(\vec{x}) = \left(\frac{1}{2}(x_1 + x_2) - 5\right)^2 + \frac{\omega}{\left(\left(\frac{1}{2}(x_1 + x_2) - 3.5\right)^2 + \epsilon\right)} + (x_1 - x_2)^2$$

The gradients $\nabla f_1(\vec{x})$ and $\nabla f_2(\vec{x})$ w.r.t. $\vec{x}$ are:

$$\nabla f_1(\vec{x}) = \begin{bmatrix} \left(\frac{1}{2}(x_1+x_2)-2\right) + \nabla_{x_1}P(\vec{x}) + 2(x_1-x_2) \\ \left(\frac{1}{2}(x_1+x_2)-2\right) + \nabla_{x_2}P(\vec{x}) - 2(x_1-x_2) \end{bmatrix}$$

$$\nabla f_2(\vec{x}) = \begin{bmatrix} \left(\frac{1}{2}(x_1+x_2)-5\right) + \nabla_{x_1}P(\vec{x}) + 2(x_1-x_2) \\ \left(\frac{1}{2}(x_1+x_2)-5\right) + \nabla_{x_2}P(\vec{x}) - 2(x_1-x_2) \end{bmatrix}$$

The design variables vector is $\overrightarrow{x} = (x_1, x_2)$, representing 2 design variables for this case.

The function $P(\vec{x})$ is called the Penalty term, as parameters ω and ϵ regulate the span of the discontinuous region of the front.

$$P(\vec{x}) = \frac{\omega}{\left(\left(\frac{1}{2}(x_1+x_2)-3.5\right)^2 + \epsilon\right)}$$

The values of the selected parameters are displayed in Table 12.

Parameter	Value
ϵ	10^{-3}
ω	$5\ 10^{-3}$

Table 12: BP 4: Values assigned to the parameters.

An interesting observation made when optimizing this BP, without applying the proposed method, is that the direction selected to scan the Pareto front yields Dominated points if $\hat{f}_2$ is selected to be decreasing, or track the border of the Weak Pareto Front (dominated solutions) if increasing.

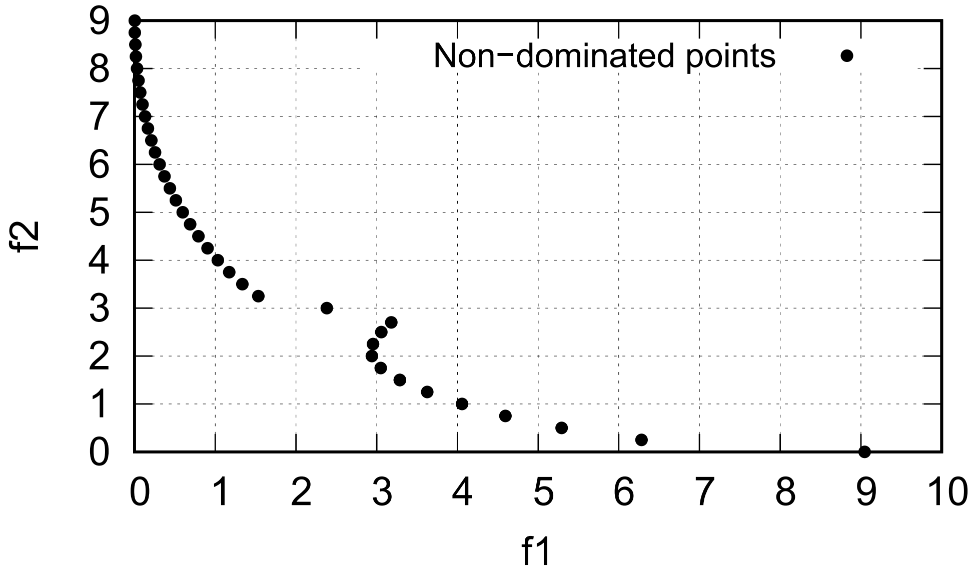


Figure 5.9: BP 4: Demonstration of generated points for $\hat{f}_2$ step-target decreasing.

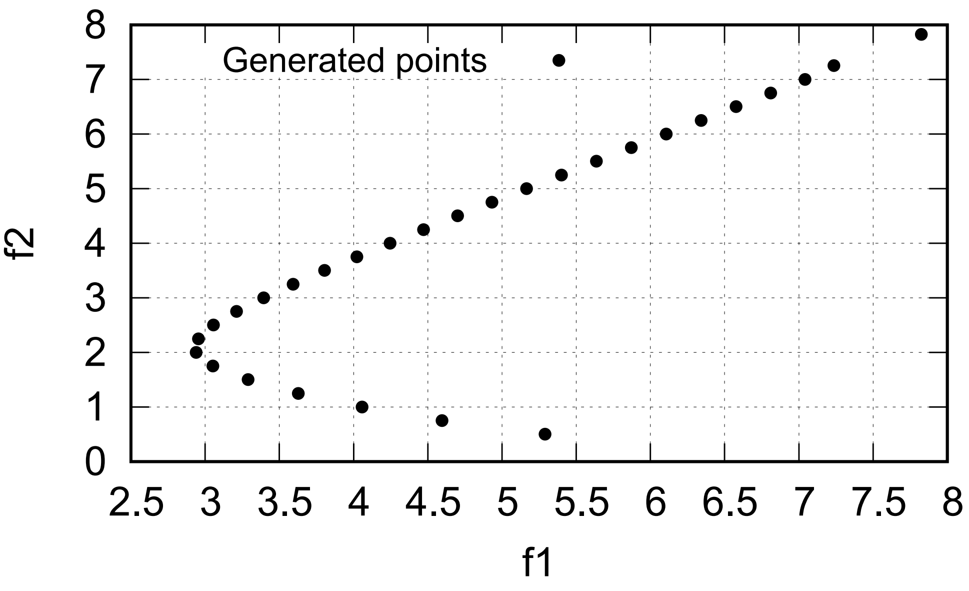


Figure 5.10: BP 4: Demonstration of generated points for $\hat{f}_2$ step-target increasing.

The shape of the generated points in Figure 5.10 is explained by the fact that λ is violating KKT conditions eq, (2.6), namely slackness condition by converging to positive values. Thus, in Figure 5.10 dominated points are tracked while positive Lagrangian coefficients indicate that the rate of change of the Lagrangian function L w.r.t. f_2 is positive. Consequently, the optimal Lagrangian function's value is increasing for an increase in f_2 instead of decreasing.

After retrieving the initial Pareto point (Go-to-Pareto step), at $(f_1,\hat{f}_{2\text{INITIAL}})=$ (0.00, 9.0) by minimizing f_1, the Pareto front is tracked with a step of $\hat{\delta} f_2=$ -0.25 tracking a total of 40 points.

For this case, the algorithms of Target-Objective jump, Back-tracking and of Pareto filter in the Prediction-step will be utilized in order to track the discontinuous front. The parameters used to initialize the optimization algorithm are shown in Table 13.

Parameter	**Value**
$\delta\hat{f}_2$	-0.25
$\delta\hat{f}_{2\text{Jump}}$	5 $\delta\hat{f}_2$
$Zscore$	10.0
Hessian Initialization	1.0

Table 13: BP 4: Values of optimization parameters.

The residuals of the KKT conditions for the Prediction-step, as well as Prediction points and Correction Pareto points are demonstrated in Figure 5.11.

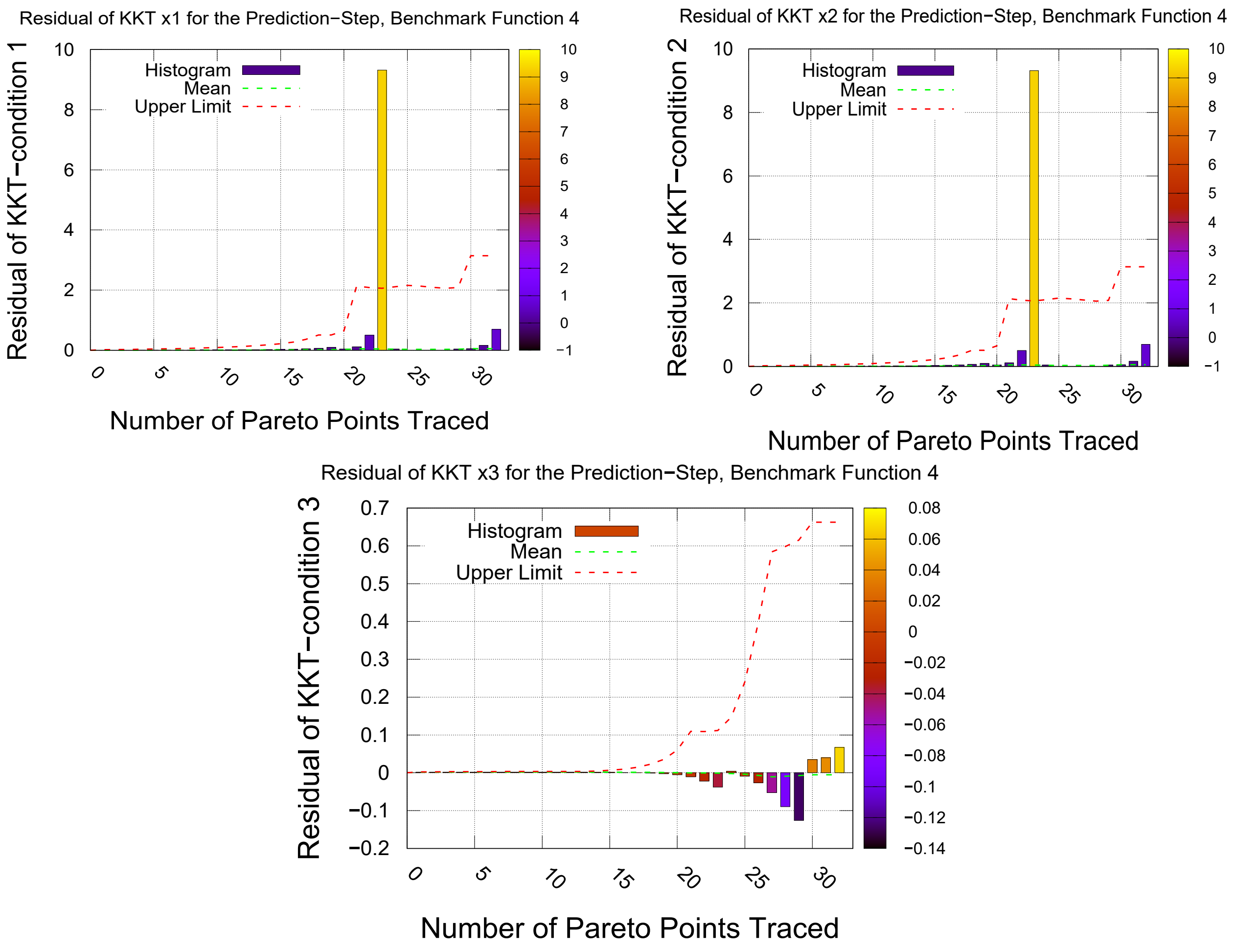


Figure 5.11: BP 4: Demonstration of Prediction-step KKT residuals.

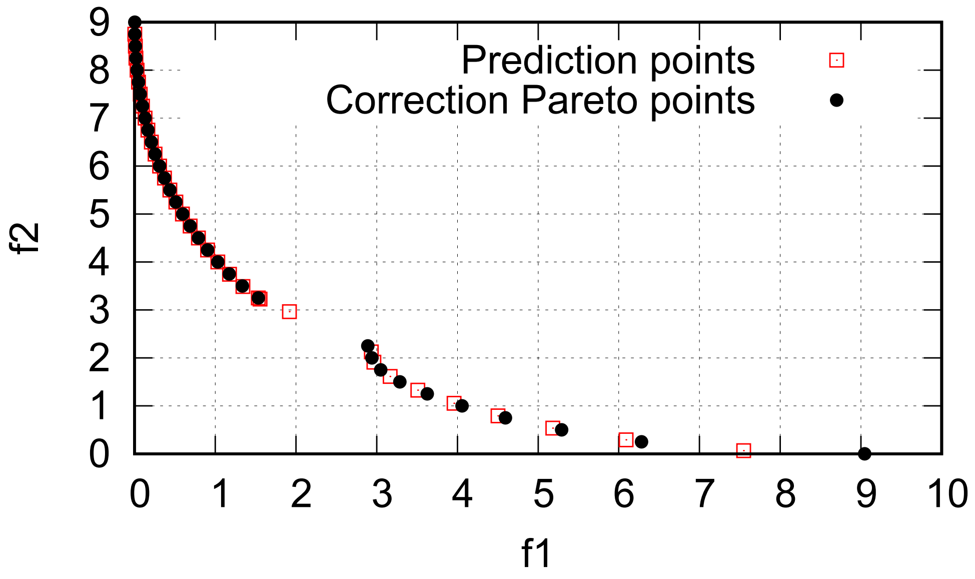


Figure 5.12: BP 4: Prediction points and Correction Pareto points.

The Pareto front tracked is shown in Figure 5.13, demonstrating, the Target-Objective jump and Back-tracking methods used in the optimization process.

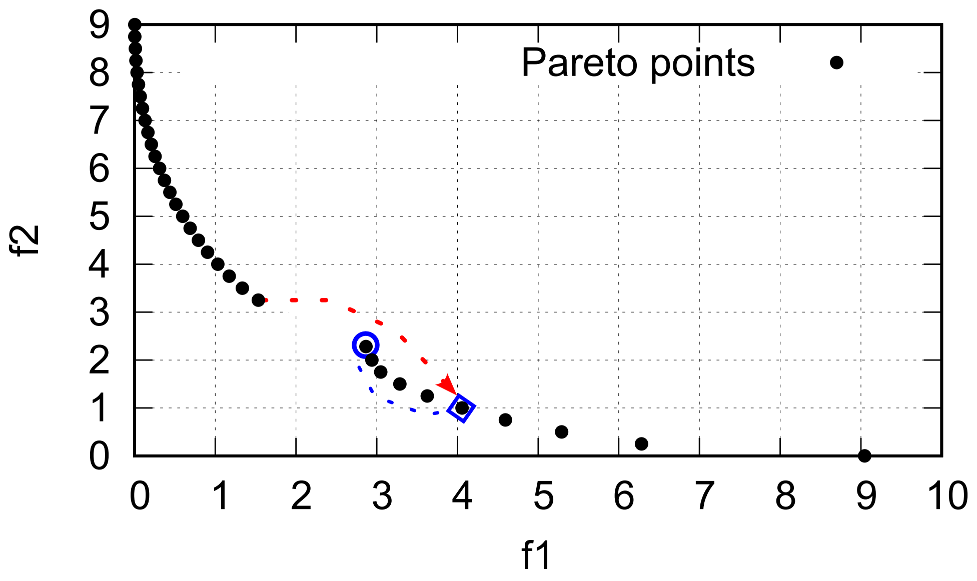


Figure 5.13: BP 4: Demonstration of Target-Objective jump and Back-tracking, for $\delta\hat{f}_{2\text{Jump}} = 5\delta\hat{f}_2$.

The Swap Target-Objective algorithm is also applied on this Benchmark. After Welford's algorithm detects the outlier point, a swapped $\delta\hat{f}_{1Jump} = 5.4\ \delta\hat{f}_1$ is chosen to perform the jump. After the jump, the problem's target and objective swap once again to min(f_1,$\hat{f}_2$) and tracking the front continues with the original step $\delta\hat{f}_2 = -0.25$.

The Pareto front tracked by this method is presented in Figure 5.14. The red arrow indicates the target set after the swapping, while the blue arrow, the tracking direction after reverting the initial objective-target functions.

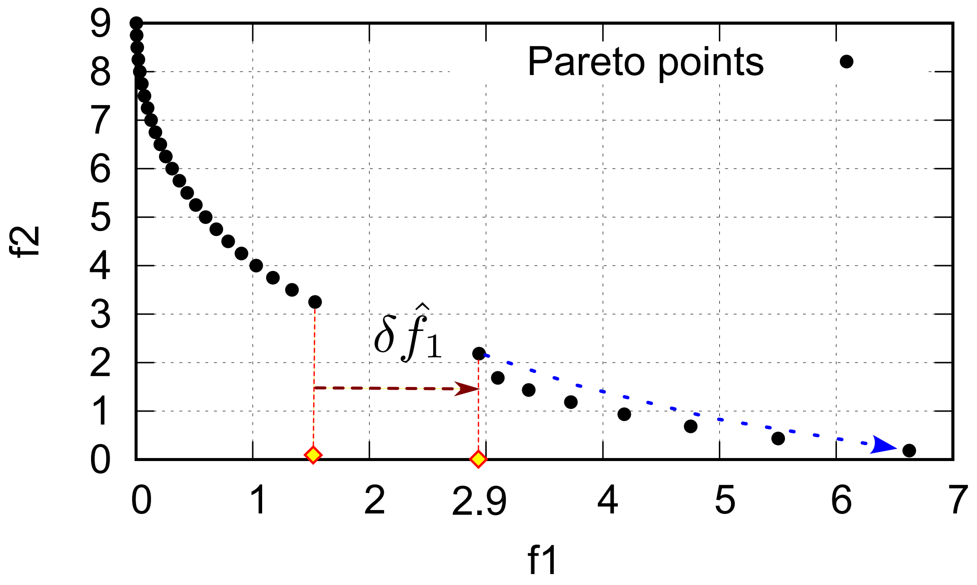


Figure 5.14: BP 4: Implementation of Swap Target-Objective.

5.6.3 BP 5

The fifth BP concerns the minimization of an alternative version of the ZDT3 function, [7]. For one design variable x_1, functions f_1 and f_2 are:

$$\begin{aligned} \text{Minimize:} f_1(\vec{x}) &= x_1 \\ f_2(\vec{x}) &= g(\vec{x}) \cdot h(f_1(\vec{x}), g(\vec{x})) \end{aligned}$$

Gradients of the objective function f, for f_1 and f_2 for BP 5:

$$\begin{aligned} \nabla f_1(\vec{x}) &= 1.0 \\ \nabla f_2(\vec{x}) &= \Bigg[g(\vec{x}) \cdot \Bigg(-0.5 \left(\frac{1.0}{\sqrt{g(\vec{x})}} \right) \sqrt{\frac{1}{f_1(\vec{x})}} \\ &\quad - \frac{1}{g(\vec{x})} \sin(10.0\pi f_1(\vec{x})) - \\ &\quad 10.0\pi \left(\frac{f_1(\vec{x})}{g(\vec{x})} \right) \cos(10.0\pi f_1(\vec{x})) \Bigg) \Bigg] \end{aligned}$$

Function $g(\vec{x})$ is defined for this version of ZDT3 as:

$$g(\vec{x}) = 1.0$$

whereas function $h(f_1, g)$ is:

$$h(f_1, g) = 1.0 - \sqrt{\frac{f_1}{g}} - \left(\frac{f_1}{g} \right) \sin(10.0\pi f_1)$$

The gradient of f_2 is computed using the chain rule based on the functions g and h, with $g(\vec{x})$ and $f_1(\vec{x})$ being part of the computation.

The design variable's bounds are:

$$0 \leq x_1 \leq 1$$

The optimization process parameters are presented in Table 14.

Parameter	**Value**
$\delta \hat{f}_2$	-0.025
Hessian Initialization	1.0
$Zscore$	10
$\delta \hat{f}_{2\text{Jump}}$	eq. (5.5).

Table 14: BP 5: Values of optimization parameters.

Given, that the Pareto front of ZDT3 is known, an internal point of the front is selected to be retrieved by the Go-to-Pareto step, with its target set as $\hat{f}_{2Initial}$=0.95. Moreover, after detecting discontinuous regions on the front, the Target-Objective jump algorithm is selected, to jump over them and track the front. A total of 74 Pareto points are tracked.

Several Target-Objective jump values are initialized in case one of them fails to track an optimal point. The Target-Objective jump vector is initialized as:

$$\delta\hat{f}_{2\text{Jump}} = \begin{bmatrix} \delta\hat{f}_2 \\ 5\delta\hat{f}_2 \\ 10\delta\hat{f}_2 \\ 12\delta\hat{f}_2 \end{bmatrix} \tag{5.5}$$

The jump values are initialized in a way that, when a discontinuous region is detected, an initial jump is attempted with a coefficient $\delta\hat{f}_{2\text{Jump}} = \delta\hat{f}_2$. If this attempt fails to converge to the target point, the jump is repeated with an increased coefficient $\delta\hat{f}_{2\text{Jump}} = 5\delta\hat{f}_2$, and so on, until convergence is achieved. The value $\delta\hat{f}_2$ concerns the initial target-step as selected in Table 14.

In this case, four discontinuous regions are encountered (it is known that ZDT3 is comprised by four discontinuous regions). The Pareto Front is tracked by combining the Target-Objective jump and Back-tracking algorithms. The residuals of the KKT conditions for the Prediction-step as well as Prediction points and Correction Pareto Points are demonstrated in Figure 5.15.

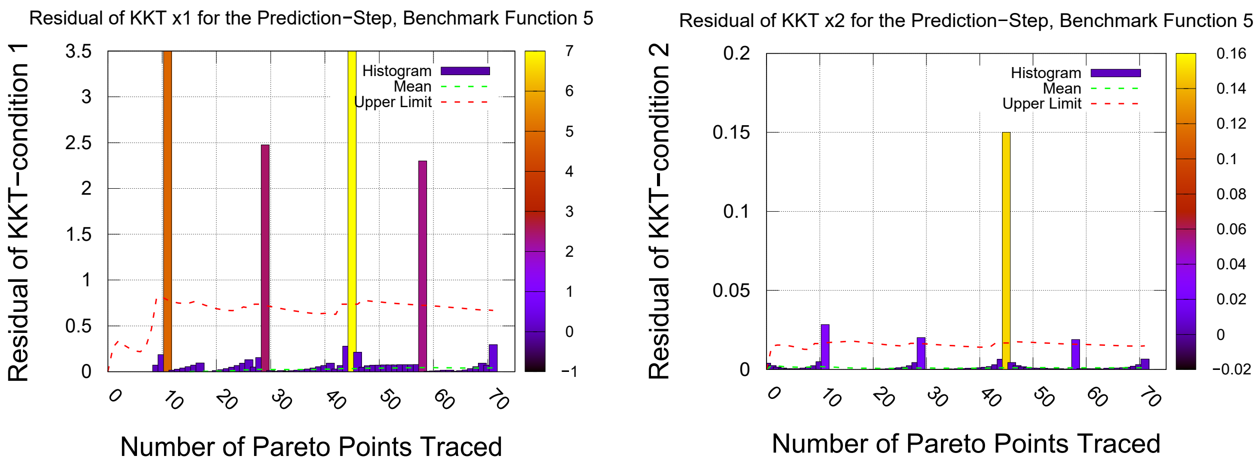


Figure 5.15: BP 5: Prediction-Step residuals.

From Figure 5.15 it can be seen that the upper limit set by the Z-score for Welford's algorithm is violated 4 times leading to the detection of the Pareto discontinuous regions.

Prediction points and Correction Pareto points of the objective function are shown in Figure 5.16, while the jumps performed and Back-tracking is displayed in Figure 5.17.

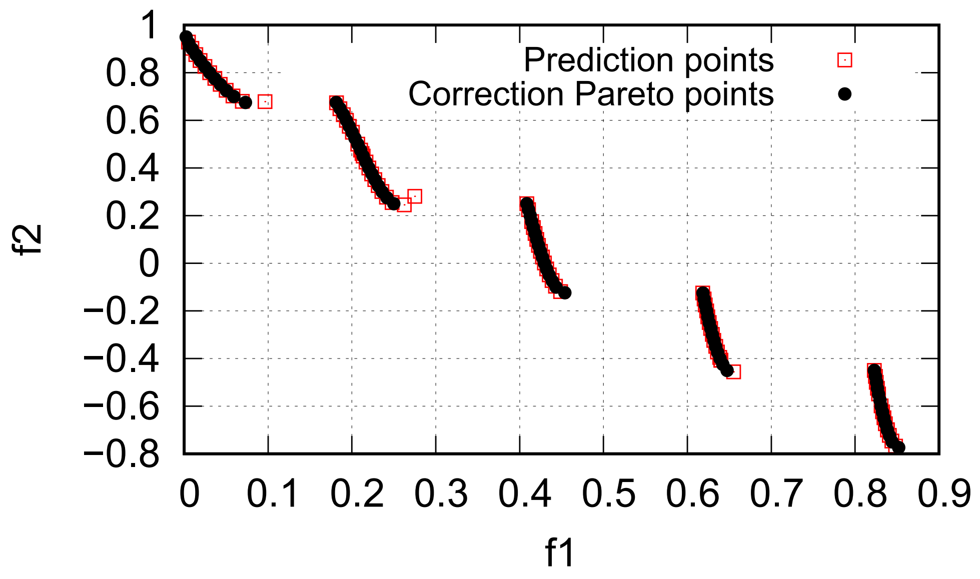


Figure 5.16: BP 5: Pareto points obtained by GB method.

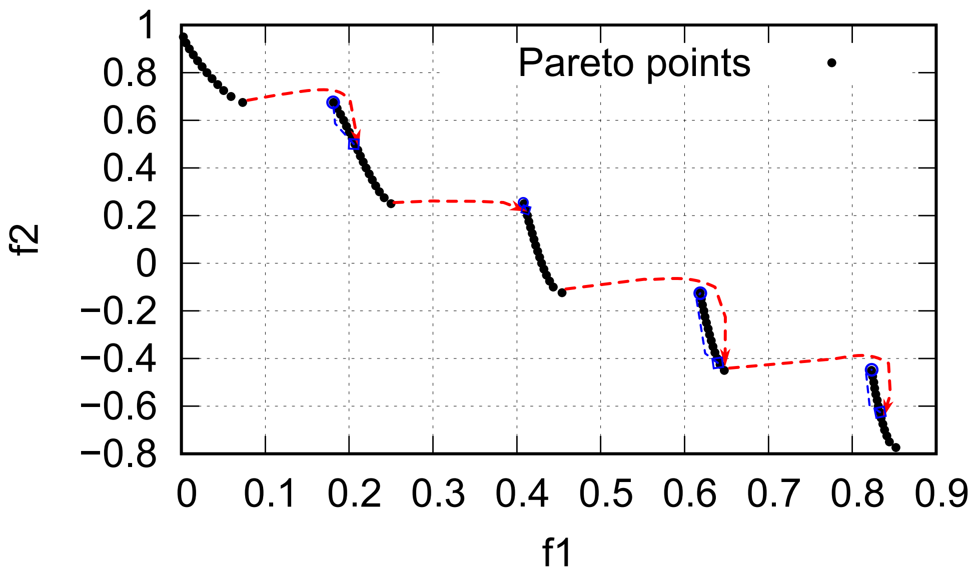


Figure 5.17: BP 5: Pareto points, with Target-Objective jump (red-arrow) and Back-tracking (blue-arrow) algorithms shown.

5.7 Conclusions

It can readily be concluded that the step size $\hat{\delta} f_2$ in the GB method can greatly impact how discontinuities are handled. Selecting a step size, $\delta \hat{f}_2$ that is larger than the discontinuous region might avoid yielding an outlier, when examining the pool of samples. In this case tracking the front will be easier as the discontinuity tackling method will not be used.

A Pareto filter is integrated into the software and can be used after tracking a point to ensure that it is not-dominated. However, the generation of dominated points rarely occurs. The computational cost of the algorithm, is thus maintained low.

Chapter 6

Tracking Three-Objective Pareto Fronts

6.1 Introduction

This chapter aims at extending the Prediction-Correction scheme as described in section 3.2 to three-objective optimization problems. The reason for extending the method for three objectives is its application in CFD. Visualization of the Pareto points is performed by three-axis plot. The major challenge in attempting to locate a 3D Pareto front is determining the algorithm that will effectively scan the front and identify all the target points. Locating the boundary points of the 3D Pareto front,and, thus altering the scanning direction to avoid evaluating non-feasible optimal points is also important and will be addressed given its significance in CFD simulations. An algorithm inspired by the box method, [1], is proposed. The accuracy, and the initialization parameters of the algorithm will be shown by optimizing two three-objective BPs, (DTLZ-1 and DTLZ-2).

6.2 Tracking the 3D Pareto front (Scan by-Layer Algorithm)

6.2.1 Formulating the Problem

Three-objective optimization problems can be formulated using eq. 3.3 for $M_t = 3$. Therefore, two-target constraints are applied with $\hat{f}_2$ and $\hat{f}_3$ being selected as constraints in each iteration of the algorithm.

$$L(\vec{x}, \vec{\lambda}_{f_k}, \vec{\lambda}_{h_j}, \vec{\mu}) = f_1(\vec{x}) - \lambda_{f_2}(f_2 - \hat{f}_2) - \lambda_{f_3}(f_3 - \hat{f}_3) - \sum_{j=1}^{M_h} \lambda_{h_j} h_j(\vec{x}) - \sum_{i=1}^{M_g} \mu_j g_j(\vec{x}) \tag{6.1}$$

The SQP method is used in the Correction-step treating the objective targets $\hat{f}_2$ and $\hat{f}_3$ as equality constraints.

The Prediction-step of the algorithm is based on the dependency of x^* on $\hat{f}_2$ and $\hat{f}_3$.

$$x^* = x^*(\hat{f}_2, \hat{f}_3) \tag{6.2}$$

Expanding eq.(6.2), as a first-order Taylor scheme yields:

$$x^*_{\text{predicted},i+1} = x_i + \frac{\partial \vec{x}}{\partial \hat{f}_2}\,\delta \hat{f}_2 + \frac{\partial \vec{x}}{\partial \hat{f}_3}\,\delta \hat{f}_3 \tag{6.3}$$

Eq. (6.3) is the basis of the Scan by-Layers, algorithm presented in order to track 3D Pareto fronts.

6.2.2 Scan by-Layers

The major challenge that is faced when tracking 3D Pareto fronts is the choice of a direction of scanning the front. In order to accurately track the front, the border points of the Pareto front must be known.

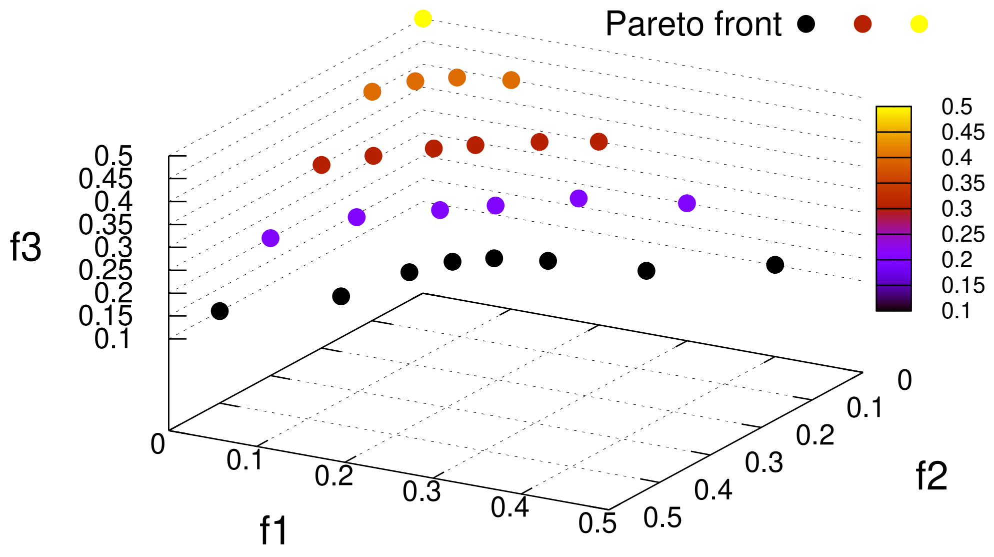


Figure 6.1: Visualization of a 3D- Pareto front.

In 3D Pareto fronts, as shown in Figure 6.1, choosing a single tracking direction is not feasible due to the complex shape of the front. Therefore, it is essential to develop a method that can dynamically adjust the tracking direction to ensure that Pareto points are accurately scanned without excessive evaluations accounting to scanning dominated points.

The Scan by-Layers method is a Prediction-Correction tracking scheme applied for two target objectives, and thus for a three-objective optimization problem.

The method consists of selecting initial values for $\hat{f}_2$ and $\hat{f}_3$ and tracking the 3D Pareto front by keeping one target constant while shifting the other. For the following applications, $\hat{f}_3$ is selected as the constant target of the layer while $\hat{f}_2$ is the changing target. This involves scanning in a layer of the 3D Pareto front where $\hat{f}_3$ remains constant while $\hat{f}_2$ varies.

The Scan by-Layers method is depicted in Figure 6.2.

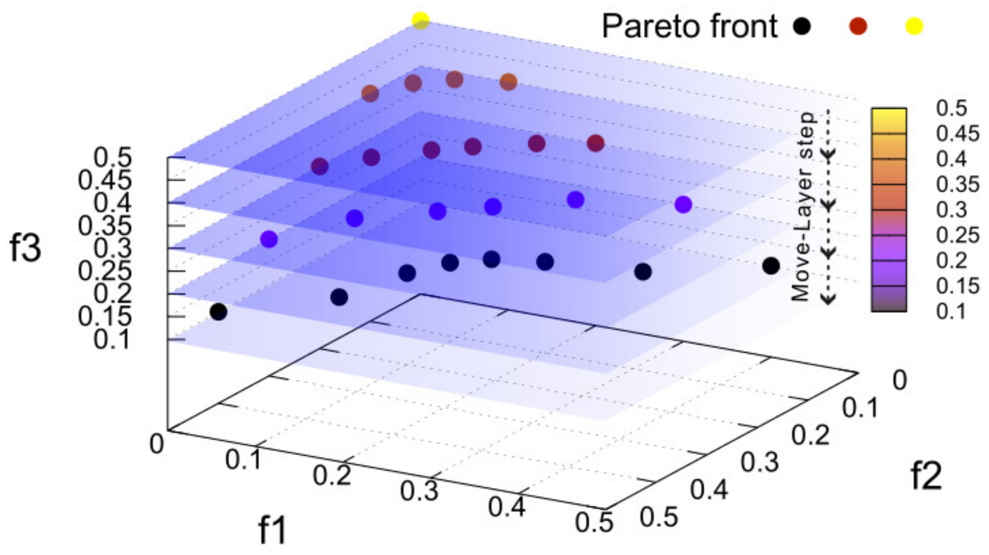


Figure 6.2: Demonstration of the Scan by-Layers method. Layers of $\delta\hat{f}_3 = 0$ are shown in blue. The Move-Layer step is depicted with black arrows.

The notion of areas of influence is fundamental for the aforementioned method. Areas of influence is an auxiliary term used to determine the scanning direction of the Prediction-step of the method and the need to move to the next layer of the 3D Pareto front thereby changing $\delta\hat{f}_3$ while $\delta\hat{f}_2$ is set as 0, from eq.(6.3).

An area of influence is a circular area of the (f_2, f_3) plane, centered around the target point set $(\hat{f}_2, \hat{f}_3)$, with radius:

$$r_{\text{influence}} = \frac{1}{2}\delta\hat{f}_2, \quad \text{if } \delta\hat{f}_3 = 0, \tag{6.4a}$$

$$r_{\text{influence}} = \frac{1}{2}\delta\hat{f}_3, \quad \text{if } \delta\hat{f}_2 = 0. \tag{6.4b}$$

In these equations the tolerance of each area, is set at the half distance between two consecutive target points $(f_1, \hat{f}_2)$. If appropriate, this tolerance can be adjusted to take values ranging between 0 and 1.

Figure 6.3, depicts the area of influence, of 3 target-points ($\hat{f}_2$, $\hat{f}_3$).

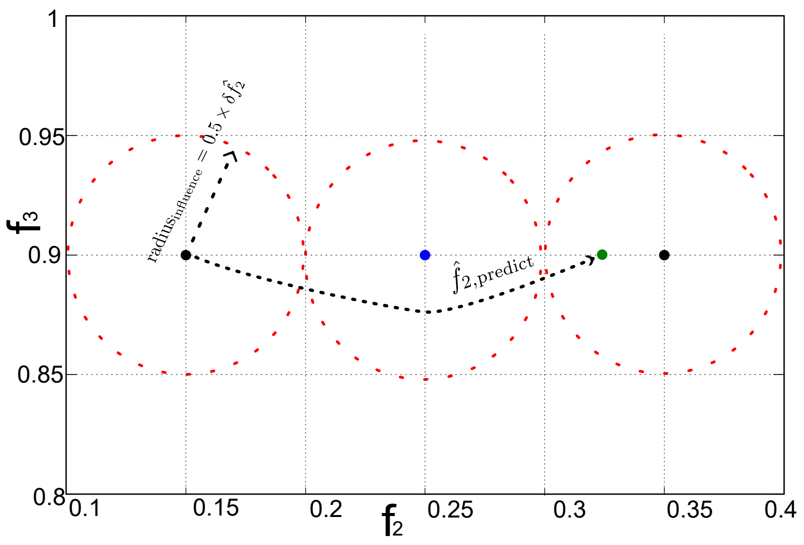


Figure 6.3: Demonstration of the selected Areas of Influence (shown in dotted red lines), around the target points:($\hat{f}_2$, $\hat{f}_3$). The green predicted point is outside the area of influence of the blue target point selected, therefore Back-tracking is used for the layer that is scanned.

The Prediction-step is determined as valid, if the Prediction point of eq.(6.3) is inside the area of Influence of the target point selected. Otherwise, Back-tracking is used on the layer, to scan it on the opposite direction as described in subsection 5.4.3. Yielding a second predicted Pareto point outside the area of influence of the target points will initiate a Move-Layer step. In this case, by using eq.(6.3) for a stable $\delta\hat{f}_2$ =0, and a shifting $\delta\hat{f}_3$, the layer is changed, triggering once again the scanning the layer-step.

A Pareto filter is integrated within the algorithm as described in subsection 5.1.1. The filter is applied after a new point is tracked, removing dominated points. The algorithm terminates once the optimal points corresponding to the $\delta\hat{f}_2$ and $\delta\hat{f}_3$ set have been tracked, covering the three extreme points: min f_1, min f_2, and min f_3.

6.2.3 Precise Tracking of the 3D Pareto's front Border

Another way of performing the Move-Layer step is by precise tracking of a surface's border elite point. The underlying idea of the concept inspired by [23], is that a single two-objective optimization Correction-step takes place with its Lagrangian to be minimized formulated as:

$$L(\vec{x}, \vec{\lambda}_{f_k}, \vec{\lambda}_{h_j}, \vec{\mu}) = f_1(\vec{x}) - \lambda_{f_3}(f_3 - \hat{f}_{3NEW}) - \sum_{j=1}^{M_h} \lambda_{h_j} h_j(\vec{x}) - \sum_{i=1}^{M_g} \mu_i g_i(\vec{x}) \tag{6.5}$$

or

$$L(\vec{x}, \vec{\lambda}_{f_k}, \vec{\lambda}_{h_j}, \vec{\mu}) = f_2(\vec{x}) - \lambda_{f_3}(f_3 - \hat{f}_{3NEW}) - \sum_{j=1}^{M_h} \lambda_{h_j} h_j(\vec{x}) - \sum_{i=1}^{M_g} \mu_i g_i(\vec{x}) \tag{6.6}$$

In this way, a border point of the new layer will be known without using a Prediction-step from the preceding layer. This method can be used as an alternative to using a Prediction-step in the Move-Layer step. The main advantage is that in irregular shaped 3D Pareto fronts the border point of the layer is unlikely to be tracked using the Prediction-step from the former layer, as it could lead to dominated Pareto Points adding to the computational cost of the problem.

6.2.4 The Scan by-Layer algorithm

The Scan by-Layer algorithm is formulated as:

Algorithm 8 Scan by-Layer Algorithm

```
Require: Number of Pareto Points $N_{\text{elite}}$, Hessian matrix initialization, $\delta\hat{f}_2$, $\delta\hat{f}_3$
    steps
Ensure: Elite Points tracked counter: $im$, $\delta\hat{f}_3$
 1: $im \leftarrow 0$
 2: $\delta\hat{f}_3 \leftarrow 0$
 3: Go-to-Pareto step
 4: while Elite Points tracked: $im < N_{\text{elite}}$ do
 5:     Store first point of the layer
 6:     Scan-the Layer step
 7:        | Prediction-step
 8:     if $X_{\text{Elite,Predicted}} < \text{Radius}_{\text{influence}}(\delta\hat{f}_2, \delta\hat{f}_3)$ then
 9:        |   Correction-step
10:     else if Back-tracking is not active then
11:        |   Load first point of the layer data
12:        |   Back-tracking → "ACTIVE"
13:        |   Algorithm 6, on layer
14:     else if Back-tracking is active then
15:        |   Back-tracking → "NOT ACTIVE"
16:        |   $\delta\hat{f}_3 \leftarrow 0$
17:        |   Move-Layer Step
18:        |   |   Prediction-Step with eq. (6.4b)
19:        |   |   Or
20:        |   |   Border-tracking of an elite Point, subsection 6.2.3
21:     end if
22:     |   Elite Points tracked: $im \leftarrow im + 1$
23: end while
```

The Scan by-Layer Algorithm will be tested in two BP, in three-objective optimization (DTLZ-1 and DTLZ-2).

6.2.5 BP 6

The sixth BP to be optimized is the DTLZ-1 function. This BP yields a 3D Pareto front, that coincides with the linear hyper-plane defined by $\sum_{m=1}^{M} f_m^* = 0.5$. The objective function has three components $\vec{f} = (f_1, f_2, f_3)$ and should be minimized for three design variables $\vec{x}$:

The function f_1 for DTLZ-1, is defined as:

$$\begin{aligned}
\text{Minimize: } f_1(\vec{x}) &= \frac{1}{2}(1+g(x_M))x_1x_2 \\
f_2(\vec{x}) &= \frac{1}{2}(1+g(x_M))x_1(1-x_2) \\
f_3(\vec{x}) &= \frac{1}{2}\,(1-x_1)\,(1+g(x_M))
\end{aligned}$$

Function $g(x_M)$ is:

$$g(\vec{x},k) = 100(k + \sum_{i=n_{\text{var}}-k}^{n_{\text{var}}-1} (x_i-0.5)^2 - \cos(20\pi(x_i-0.5)))$$

where $k = n - 3 + 1$. The variable X_M refers to the design variables exceeding k, that is $M > k$.

The gradient of objective function for DTLZ-1 is:

$$\nabla f_1(\vec{x}) = \begin{bmatrix} \nabla_{x_1} f_1 \\ \nabla_{x_2} f_1 \\ \nabla_{x_3} f_1 \end{bmatrix}, \nabla f_2(\vec{x}) = \begin{bmatrix} \nabla_{x_1} f_2 \\ \nabla_{x_2} f_2 \\ \nabla_{x_3} f_2 \end{bmatrix}, \nabla f_3(\vec{x},n) = \begin{bmatrix} \nabla_{x_1} f_3 \\ \nabla_{x_2} f_3 \\ \nabla_{x_3} f_3 \end{bmatrix}$$

The $\nabla f(\vec{x})$ terms are formulated as:

$$\begin{aligned}
\nabla_{x_1} f_1 &:= 0.5x_2(1+g(x_M)) + 0.5x_1x_2\nabla_{x_1} g(\vec{x_M}) \\
\nabla_{x_2} f_1 &:= 0.5x_1(1+g(x_M)) + 0.5x_1x_2\nabla_{x_2} g(\vec{x_M}) \\
\nabla_{x_3} f_1 &:= 0.5x_1x_2\nabla_{x_3} g(\vec{x_M}) \\
\nabla_{x_1} f_2 &:= 0.5(1-x_2)(1+g(x_M)) + 0.5x_1(1-x_2)\nabla_{x_1} g(\vec{x_M}) \\
\nabla_{x_2} f_2 &:= -0.5x_1(1+g(x_M)) + 0.5x_1(1-x_2)\nabla_{x_2} g(\vec{x_M}) \\
\nabla_{x_3} f_2 &:= 0.5x_1(1-x_2)\nabla_{x_3} g(\vec{x_M}) \\
\nabla_{x_1} f_3 &:= -0.5(1+g(x_M)) + 0.5(1-x_1)\nabla_{x_1} g(\vec{x_M}) \\
\nabla_{x_2} f_3 &:= 0.5(1-x_1)\nabla_{x_2} g(\vec{x_M}) \\
\nabla_{x_3} f_3 &:= 0.5(1-x_1)\nabla_{x_3} g(\vec{x_M})
\end{aligned}$$

The Scan by-Layers algorithm is initialized with the parameters shown in Table 15.

Parameter	**Value**
$\delta\hat{f}_2$	-0.05
$\delta\hat{f}_3$	-0.1
Hessian Diagonals Initialization	1.0

Table 15: BP 6: Initialized parameters of optimization process.

The initial Go-to-Pareto step, is performed by minimizing f_1,f_2 and thus the point:($\hat{f}_{1\text{INITIAL}}$,$\hat{f}_{2\text{INITIAL}}$,$\hat{f}_{3\text{INITIAL}}$)= (0.0, 0.0, 0.5) is retrieved. The GB method tracks a total of 36 points. Once again the SQP algorithm is used in the Correction-Step. As shown in Figures 6.4, 6.5 the Scan by-Layer algorithm successfully tracks the 3D Pareto front for the steps $\delta\hat{f}_2$, $\delta\hat{f}_3$, set.

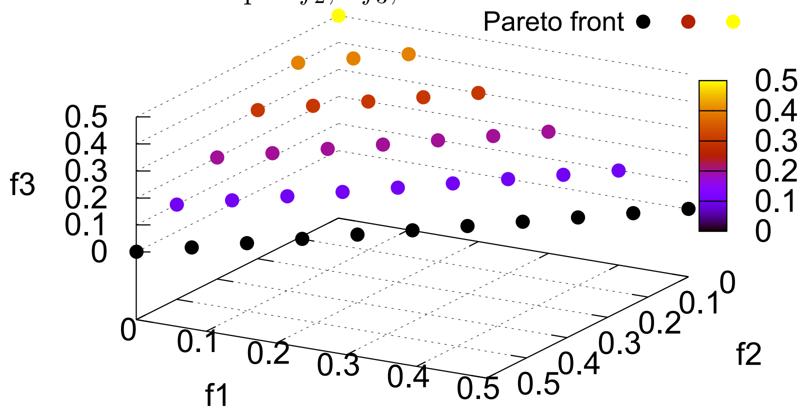


Figure 6.4: BP 6: Demonstration of the 3D Pareto front obtained by Scan by-Layers method.

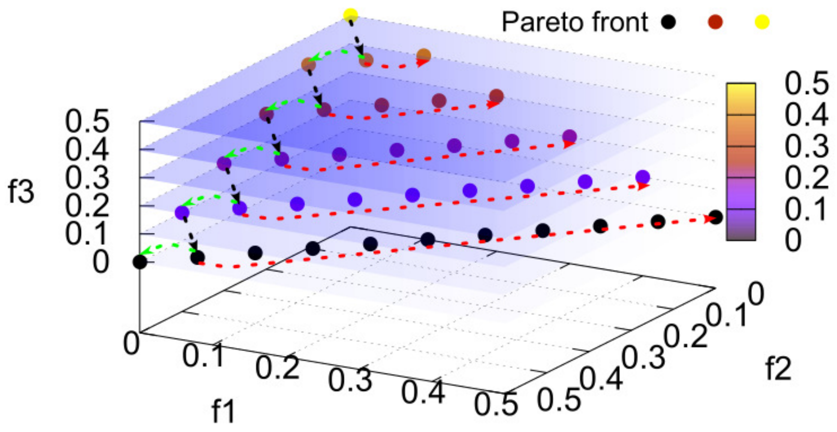


Figure 6.5: BP 6: Demonstration of the Scan by-Layers method, used to track 3D Pareto front, (red arrows): scanning direction on layer, (green arrows): points obtained by Back-tracking, (black arrows): Move-Layer step.

6.2.6 BP 7

The seventh BP to be optimized is the DTLZ-2 function. This function yields a convex surface shaped 3D Pareto front defined by $\sum_{m=1}^{M} f_m^{2*} = 1$. The function is defined for three design variables $\vec{x}$ and three objective functions $\vec{f} = (f_1, f_2, f_3)$, as:

$$\begin{aligned}\text{Minimize:} f_1(\vec{x}) &= (1 + g(x_M))cos(\frac{\pi}{2}x_1)cos(\frac{\pi}{2}x_2)\\ f_2(\vec{x}) &= (1 + g(x_M))cos(\frac{\pi}{2}x_1)sin(\frac{\pi}{2}x_2)\\ f_3(\vec{x}) &= (1 + g(x_M))sin(\frac{\pi}{2}x_1)\end{aligned}$$

Function $g(x_M)$ is:

$$g(\vec{x}, k) = \sum_{i=n-k}^{n-1} (x_i - 0.5)^2$$

Where X_M denotes the number of design variables that exceed k, defined as $k = n - 3 + 1$, implying that $M > k$. The gradient of Objective Function f is:

$$\nabla f_1(\vec{x}) = \begin{bmatrix} \nabla_{x_1} f_1 \\ \nabla_{x_2} f_1 \\ \nabla_{x_3} f_1 \end{bmatrix}, \nabla f_2(\vec{x}) = \begin{bmatrix} \nabla_{x_1} f_2 \\ \nabla_{x_2} f_2 \\ \nabla_{x_3} f_2 \end{bmatrix}, \nabla f_3(\vec{x}) = \begin{bmatrix} \nabla_{x_1} f_3 \\ \nabla_{x_2} f_3 \\ \nabla_{x_3} f_3 \end{bmatrix}$$

where $\nabla_{x_1} f_1 := -\pi 0.5 sin(\frac{\pi}{2}x_1)cos(\frac{\pi}{2}x_2)(1 + g(x_M)) + cos(\frac{\pi}{2}x_1)cos(\frac{\pi}{2}x_2)\nabla_{x_1} g(\vec{x_M})$

$$\nabla_{x_2} f_1 := -\pi 0.5 sin(\frac{\pi}{2}x_2)cos(\frac{\pi}{2}x_1)(1 + g(x_M)) + cos(\frac{\pi}{2}x_1)cos(\frac{\pi}{2}x_2)\nabla_{x_2} g(\vec{x_M})$$

$$\nabla_{x_3} f_1 := cos(\frac{\pi}{2}x_1)cos(\frac{\pi}{2}x_2)\nabla_{x_3} g(\vec{x_M})$$

$$\nabla_{x_1} f_2 := -\pi 0.5 sin(\frac{\pi}{2}x_1)sin(\frac{\pi}{2}x_2)(1 + g(x_M)) + cos(\frac{\pi}{2}x_1)sin(\frac{\pi}{2}x_2)\nabla_{x_1} g(\vec{x_M})$$

$$\nabla_{x_2} f_2 := \pi 0.5 cos(\frac{\pi}{2}x_1)cos(\frac{\pi}{2}x_2)(1 + g(x_M)) + sin(\frac{\pi}{2}x_2)cos(\frac{\pi}{2}x_1)\nabla_{x_2} g(\vec{x_M})$$

$$\nabla_{x_3} f_2 := cos(\frac{\pi}{2}x_1)sin(\frac{\pi}{2}x_2)\nabla_{x_3} g(\vec{x_M})$$

$$\nabla_{x_1} f_3 := -0.5\pi(1 + g(x_M))cos(\frac{\pi}{2}x_1) + sin(\frac{\pi}{2}x_1)\nabla_{x_1} g(\vec{x_M})$$

$$\nabla_{x_2} f_3 := sin(\frac{\pi}{2}x_1)\nabla_{x_2} g(\vec{x_M})$$

$$\nabla_{x_3} f_3 := sin(\frac{\pi}{2}x_1)\nabla_{x_3} g(\vec{x_M})$$

The Scan by-Layers algorithm is initialized with the parameters shown in Table 16:

Parameter	**Value**
$\delta\hat{f}_2$	-0.05
$\delta\hat{f}_3$	-0.1
Hessian Diagonals Initialization	1.0

Table 16: BP 7: Initialized Parameters for 3D Pareto front

The initial Go-to-Pareto step is performed by minimizing f_1,f_2 and thus the point: ($\hat{f}_{1\text{INITIAL}}$,$\hat{f}_{2\text{INITIAL}}$,$\hat{f}_{3\text{INITIAL}}$)= (0.0, 0.0, 1.0) and a total of 174 Pareto points are tracked, scanning 10 layers of $\hat{f}_3$. The front obtained by the Scan by-Layers algorithm is.

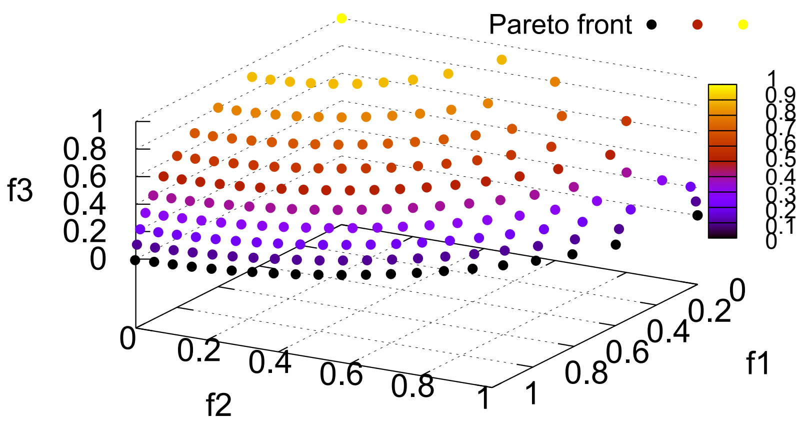


Figure 6.6: BP 7: Demonstration of the 3D Pareto front, obtained by Scan by-Layer method.

6.3 Conclusions

The proposed Scan by-Layers algorithm has successfully tracked both mathematical BPs. It is characterized by its low computational time and its adaptability as the user can define the number and the range of Pareto points that are to be tracked. However, using areas of influence to track the border of the 3D Pareto front may mistakenly identify a discontinuity region as the border potentially missing elite points on a layer. Integrating the precise tracking of the 3D Pareto front's border can facilitate tracking all the points of the layer at the expense of additional computational cost. Furthermore, the precise tracking of the 3D Pareto front's border can be combined with the method described in chapter 5 to track discontinuous 3D Pareto fronts.

Chapter 7

Three-Objective CFD Application

7.1 Introduction

The following application focuses on optimizing a three-objective CFD case. The aim is to optimize the shape of the airfoil under both the airflow conditions set in section 4.1 and take-off conditions, that will be defined in section 7.2. The three aerodynamic objectives to be optimized are: minimum drag, maximum lift (coefficients), both at cruise conditions and attaining a high target lift coefficient during take-off. Therefore, two operating points are involved. The above mentioned CFD problem can be solved using the GB Scan by-Layers algorithm as described in section 6.2.4.

7.2 Cruise and Take-Off Airflow Conditions

Cruise flight conditions are defined as the airflow conditions of the section 4.1. $M_\infty = 0.8$ while $a_\infty = 2°$.

During take-off, the free-stream Mach speed is set to $M_\infty = 0.23$, with $a_\infty = 9°$. The mesh around the airfoil is selected to remain the same as in section 4.1, comprised of 6500 nodes while the farfield boundary is located at approximately 10 airfoil chords away from the airfoil.

7.3 Scan by-Layers Algorithm Initialization

The Pareto front of the case is tracked by applying the Scan by-Layers algorithm as described in section 6.2 for three-objective problems.

The functions to be minimized read as:

$$f = \begin{cases} f_1(\vec{x}) = C_D(\vec{x}), & \text{at Cruise conditions} \\ f_2(\vec{x}) = -C_L(\vec{x}), & \text{at Cruise conditions} \\ f_3(\vec{x}) = -C_L(\vec{x}), & \text{at Take-off conditions} \end{cases} \tag{7.1}$$

The Lagrangian of the problem is:

$$L(\vec{x}, \vec{\lambda}_{f_k}) = f_1(\vec{x}) - \lambda_{f_2}(f_2(\vec{x}) - \hat{f}_2) - \lambda_{f_3}(f_3(\vec{x}) - \hat{f}_3) \tag{7.2}$$

Once again, the airfoil shape is parameterized using the same NURBS lattice and thus, the y (vertical) coordinates of the same 12 CPs constitute the design variables of the application. The Scan by-Layers algorithm for this CFD case is modified in order to handle the primal and adjoint equations and reads as:

Algorithm 9 Scan by-Layer Algorithm, applied to external inviscid flow

```
Require: Number of Pareto Points $N_{\text{elite}}$, Hessian matrix initialization, $\delta\hat{f}_2$, $\delta\hat{f}_3$
    steps, Initial Point coordinates
Ensure: Elite points tracked counter: $im$, $\delta\hat{f}_3$
 1: $im \leftarrow 0$
 2: $\delta\hat{f}_3 \leftarrow 0$
 3: Go-to-Pareto step
 4: | Adapt mesh to the current design variables
 5: | Solve primal equations
 6: | Solve adjoint equations
 7: | Update $F_{\text{obj}}$ and sensitivity derivatives
 8: while Elite points tracked: $im < N_{\text{elite}}$ do
 9:     Store first point of the layer
10:     Scan-the layer step
11:        | Prediction-step
12:        |    | Solve primal equations
13:        |    | Update $F_{\text{obj}}$ and sensitivity derivatives
14:     if $X_{\text{Elite,Predicted}} < \text{Radius}_{\text{influence}}(\delta\hat{f}_2, \delta\hat{f}_3)$ then
15:        |    | Correction-step
16:        |    |    | Adapt mesh to the current design variables
17:        |    |    | Solve primal equations
18:        |    |    | Solve adjoint equations
19:        |    |    | Update $F_{\text{obj}}$ and sensitivity derivatives
20:     else if Back-tracking is not active then
21:        |    | Load first point of the layer data
22:        |    | Back-tracking → "ACTIVE"
23:        |    | Algorithm 6, on Layer
24:     else if Back-tracking is active then
25:        |    | Back-tracking → "NOT ACTIVE"
26:        |    | $\delta\hat{f}_3 \leftarrow 0$
27:        |    | Move-Layer Step
28:        |    |    Prediction-Step with eq.(6.4b)
29:        |    |    Or
30:        |    |    Border-tracking of an elite Point, subsection 6.2.3
31:     end if
32:     |    Elite points tracked: $im \leftarrow im + 1$
33: end while
```

The convergence criterion of the algorithm adopted is selected for the Correction-step of the SQP method as 10^{-2} for $\hat{f}_2$, and $5\ 10^{-3}$ for $\hat{f}_3$.

The EFS of this CFD case can be measured by first estimating the number of times the primal and adjoint equations are solved in each optimization cycle. In each cycle two primal equations (one for the cruise conditions (f_1,f_2) and one for the take-off conditions f_3) are solved, while three adjoint equations (two for f_1=C_D and f_2=-C_L at flight conditions and one for the sensitivity derivatives of f_3=-C_L at take-off conditions) are solved. The lift adjoint equations are first solved to obtain the values of C_L. Subsequently, the sensitivity derivatives of these values are inverted, as they represent objectives to be maximized. The cost is 5 EFS at each optimization cycle.

Additionally, the use of Back-tracking involves solving two additional primal equations before reversing the scanning direction. This is the case, because Back-tracking is triggered after a Prediction-step point is found outside the radius of influence of the target solution. This accounts to the solution of an additional set of 2 primal equations. The same procedure is followed each time a Move-Layer step is applied.

The GB method begins with the Go-to-Pareto step to track the first point on the front. Starting from an airfoil geometry with non-optimal aerodynamic coefficients $C_D = 0.06$ and $C_L = 0.985$, at cruise conditions, the Go-to-Pareto step tracks an initial point on the front, from which the Move-on-Pareto step will be applied using a step size of $\delta\hat{f}_2$ on the first layer to be scanned. Based on the results of the EA, the initial target values of $\hat{f}_2 = -0.83$, $\hat{f}_3 = -1.33$ are chosen, with a step size of $\delta\hat{f}_2 = -0.15$, $\delta\hat{f}_3 = -0.1$ to track points across the same range as it. The GB method is set to track 10 Elite points

The parameterization data for the Scan by-Layers optimization method, is displayed in Table 17.

Parameter	**Value**
$\delta\hat{f}_2$	-0.15
$\delta\hat{f}_3$	-0.1
Hessian Diagonals Initialization	10.0

Table 17: Airfoil case three-objectives: Initial values for the parameters of 3D Pareto front.

Thus, for 10 Elite Points, the total cost in EFS is presented in Table 18.

Differential Equations	**EFS**
Primal	134
Adjoint	180
Total	314

Table 18: Airfoil case three-objectives: EFS cost evaluation of the Scan by-Layers Algorithm.

7.4 Results

The execution of the Scan by-Layers algorithm yields the following 3D Pareto front:

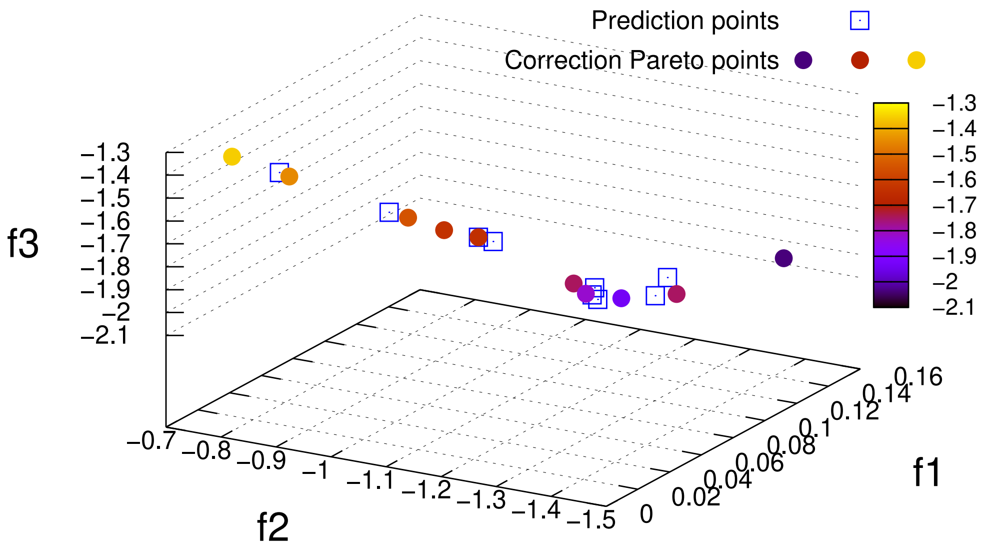


Figure 7.1: Airfoil case three-objectives: Demonstration of the 3D Pareto front tracked with the Scan by-Layers algorithm.

For the EA selected to track the 3D Pareto front the EASY software was once again utilized. The 3D Pareto front traced with the use of the EA, is comprised of 31 elite points in total. The use of "RBF IF" metamodels are enabled during the EA run.

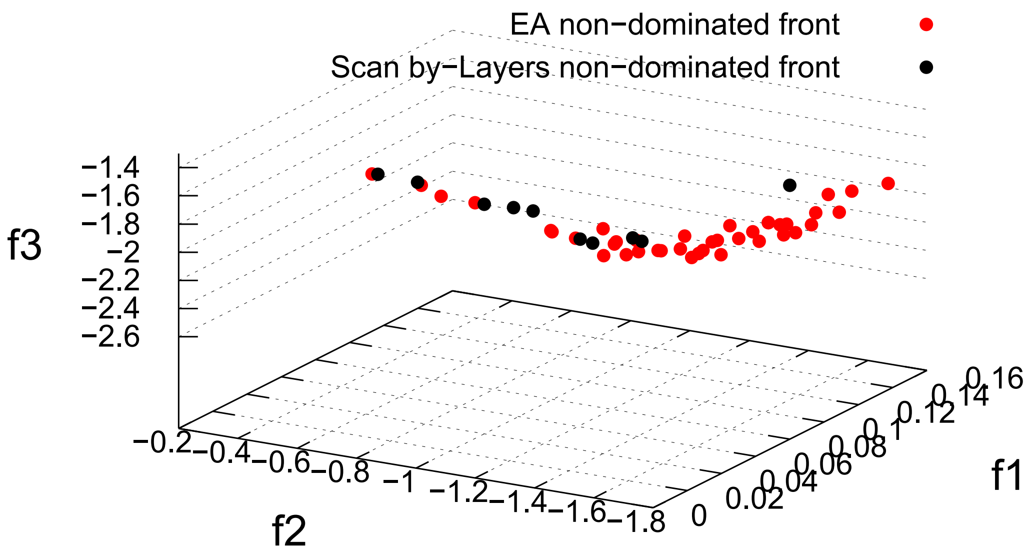


Figure 7.2: Airfoil case three-objectives: Pareto front tracked with Scan by-Layers algorithm (black) and EA (red).

The EFS cost of the EA has increased in comparison to the case described in section

4.1, as two sets of primal equations are solved at each evaluation instead of one thus increasing the algorithm's cost.

Algorithm	**EFS**
EA with Metamodels	2000
GB Method	314

Table 22: Airfoil case three-objectives: EFS evaluation for Scan by-Layers and EA.

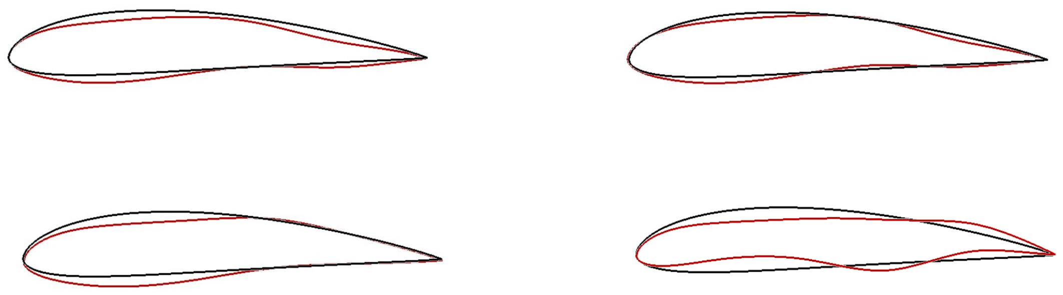


Figure 7.3: Airfoil case three-objectives: ShpO (black), (baseline geometry) from left to right, up to down (red): $(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0178, 0.768, 1.356)$,
$(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0282, 0.842, 1.450)$,
$(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0312, 1.049, 1.551)$,
$(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0332, 1.457, 1.745)$.

The ShpO results of the GB method are presented in Figure 7.3. It is shown that in order to achieve the target coefficient of Lift, $C_{L,9^\circ}$, at take-off conditions, as expected curvature of both the suction and pressure side increase, even yielding an extremely curved-shaped airfoil as the second one. Additionally, for higher $C_{L,9^\circ}$, the area around the trailing edge is flap-shaped, as this yields a higher lift coefficient.

In Figures 7.4 and 7.5, the Mach number fields of the optimized airfoils are compared for cruise and take-off flight conditions. Once again, the intensity of shock waves on the suction side is reduced thus, minimizing drag losses.

It is important to note that the shapes presented in Figure 7.3, particularly the shape corresponding to $(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0332, 1.457, 1.745)$, may be disregarded due to aerodynamic considerations and the total surface characteristics of the airfoil. However, these shapes arise from the parameterization of the problem.

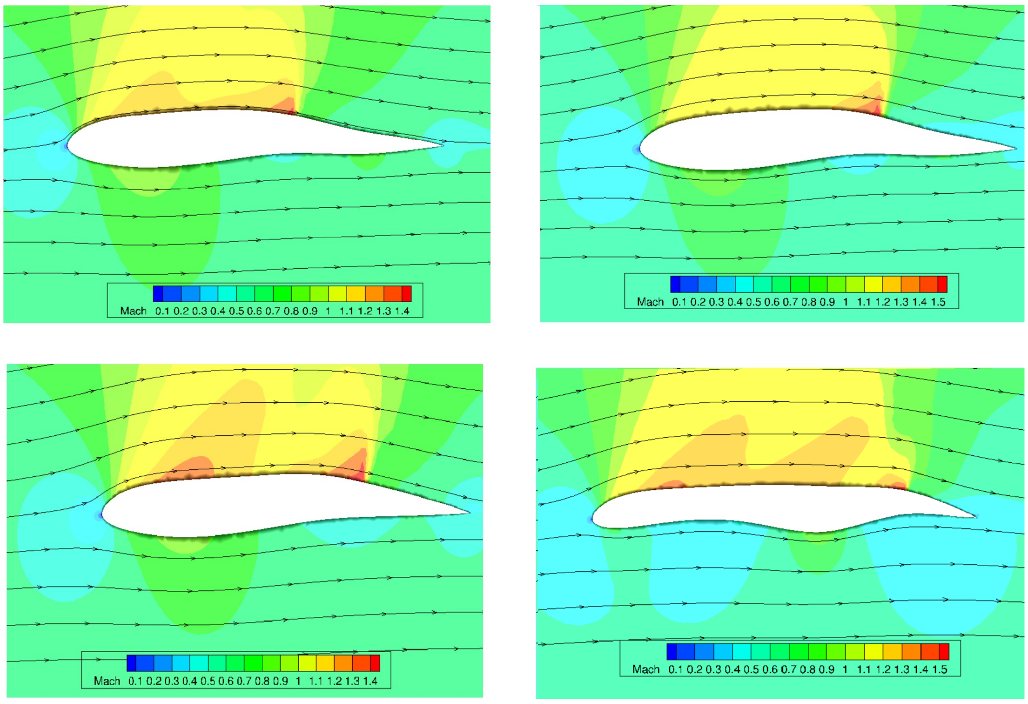


Figure 7.4: Airfoil case three-objectives: Mach flow fields, from left to right, **(cruise conditions)**: $(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0178, 0.768, 1.356)$,
$(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0282, 0.842, 1.450)$,
$(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0312, 1.049, 1.551)$,
$(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0332, 1.457, 1.745)$.

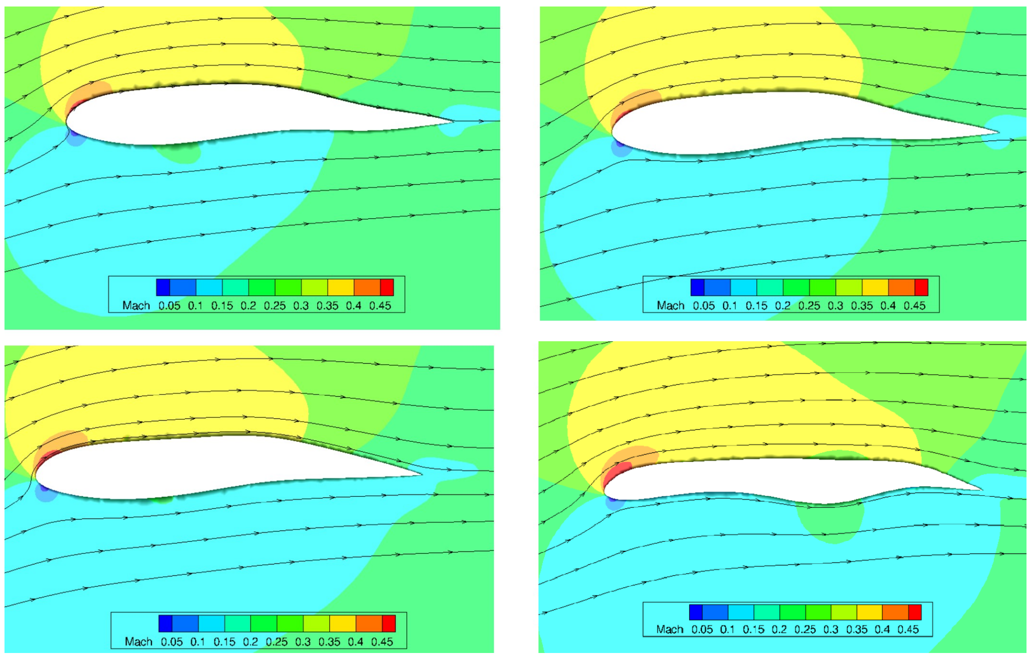


Figure 7.5: Airfoil case three-objectives: Mach flow fields, from left to right, **(take-off conditions)**: $(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0178, 0.768, 1.356)$,
$(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0282, 0.842, 1.450)$,
$(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0312, 1.049, 1.551)$,
$(C_{D,2^\circ}, C_{L,2^\circ}, C_{L,9^\circ}) = (0.0332, 1.457, 1.745)$.

Chapter 8

Summary and Future Work

8.1 Summary

In this Diploma Thesis, a method for solving MOO problems using GB techniques and tracking the Pareto front, extended also to CFD applications, was developed and programmed in C++. The underlying concept is that by providing a starting point, the method will be able to track the Pareto front of the MOO with a reasonable number of (CFD) evaluations. A range of methods are provided to optimize two and three-objective problems. New algorithms are developed and used to solve problems in which the purpose is to track discontinuous Pareto fronts and three-objective 3D Pareto fronts.

The Prediction-Correction scheme was integrated within the GB method for tracking the Pareto front as it can accommodate tracking a particular number of elite points, with a set target step. Briefly, the algorithm a priori receives as input the maximum number of Pareto points to be tracked, the step size between them, and an initialization in the design space. It comprises of two steps: the Go-to-Pareto Step and the Move-on-Pareto Step. The first step is responsible for locating the first optimal point of the Pareto front based on the given initial solution in the design space. The second step tracks the rest of points along the Pareto front. The Move-on-Pareto Step consists of a Prediction-step which utilizes the implicit function theorem to estimate the next optimal solution. This is followed by a Correction-step that accurately refines the solution obtained to the Pareto front.

In this thesis, the Correction-Step was applied utilizing the ALM method equipped with a steepest descent and a Quasi-Newton, BFGS variant. An additional modification was made to use the SQP method. In order to reduce the cost in CFD applications the hessian matrix needed in SQP can be estimated with Quasi-Newton methods either (BFGS,dBFGS or SR-1 method). The accuracy of the Prediction-step in the SQP method was enhanced by using the most recent hessian approximation from the previous Correction-step.

Two mathematical applications were presented to compare the computational cost of the SQP and ALM algorithms. Particularly, the Bihn and Korn BP was optimized using both methods in the Correction-step and it was shown that SQP minimizes the case in 88 optimization cycles while ALM in 288. By minimizing both BPs, with the exact computation of the hessian and their Quasi-Netwon, it was shown that Quasi-Newtons can be used to accurately track optimal point, as the Pareto fronts tracked were almost identical.

A CFD application aimed at the shape optimization of the NACA-4415 airfoil for minimum C_D and maximum C_L is solved under external airflow conditions of ($M_\infty = 0.8$, $a_\infty = 2$), and the software was integrated with the primal and adjoint solvers of PUMA/NTUA. The computed fronts and the computational cost (EFS) were compared to those of EASY, and it was shown that the Pareto front tracked by the GB method tracked a total of 15 points. A variant of this application was also presented where the same aerodynamic coefficients were optimized, while maintaining a zero-pitching moment coefficient C_M, tracking a total of 7 optimal points

New proposals were presented regarding discontinuous Pareto fronts. A connection was made between KKT conditions residuals obtained from the Prediction-step and local front curvature change. This allowed KKT residuals to be used as indicators of discontinuous regions in the Pareto front. Welford's Online algorithm was integrated into the software to calculate the variance of the sample of KKT residuals collected after each Prediction-step, and detect the outlier points. Furthermore, a method for tracking discontinuous fronts was proposed. In order to track discontinuous fronts three algorithms can be combined, constituting a robust method: Target-Objective jump, Back-tracking, and Swap Target-Objective. These methods were applied to three mathematical BPs, successfully detecting discontinuous regions and by performing the tracking method, they tracked the 2D fronts.

The GB method developed was expanded to three-objective optimization problems, (Scan by-Layers algorithm). The Scan by-Layers algorithm involves changing one of the two target objectives while keeping the other constant until the Prediction-step yields a solution point outside the area of influence of the next target point to be optimized. Back-tracking is then used to scan in the reverse direction until a solution point outside the area of influence of the next target point is reached again. After this the second target changes by a step size, and the process is repeated until all elite points are tracked. Two BPs were optimized and presented using this algorithm (DTLZ-1, DTLZ-2), which were both successfully tracked by the algorithm for 36 and 174 Pareto points respectively.

Finally, a three-objective CFD application was presented which optimized the shape of the airfoil under both cruise airflow conditions ($M_\infty = 0.8, a_\infty = 2$) and take-off conditions ($M_\infty = 0.23, a_\infty = 9$), while also binding a third objective of a target CL at a higher angle of attack during take-off conditions. The results of the Scan by-Layers algorithm were compared to those of EASY, with the GB method tracking 10 Pareto points at the cost of 314 flow solutions (in terms of EFS).

8.2 Conclusions

After having completed the development of the software and the algorithms, and having optimized the CFD applications in the Thesis the following conclusions can be made:

SQP and ALM, robustness and computational cost

The configuration of the SQP algorithm is considered to be easier and more robust than that of the ALM algorithm, given the latter's requirement for precise initialization of the Lagrangian variables λ_k and penalty constants ω_k which often leads to divergence of the method. The integration of Quasi-Newton methods in the SQP algorithm makes it less computationaly intense than its ALM counterpart even when the latter utilizes these methods. Therefore, the SQP method was used in all the applications of this thesis.

Quasi-Newton Methods in CFD applications

When tracking the Pareto front, Quasi-Newton methods can be integrated with the optimization software but they require a "good" initialization of the hessian. In all applications, multiple initializations of the hessian were given as input optimization process each at different orders of magnitude. This ensures that, if the first initialization fails to converge, another can be used allowing the tracking of the Pareto front to continue. Therefore, the optimization code should be provided with multiple values of different orders of magnitude for the hessian initialization.

Influence of Step-Size and Target-Objective jumps's accuracy, in Discontinuous Fronts

The step size of the Prediction-Correction algorithm can significantly influence the effect of discontinuities. Choosing a step size larger than the discontinuous region may prevent the generation of an outlier point, when processing the pool of samples, assuming the curvature of the front remains consistent. The Prediction-step by default cannot be used to locate the next point because the $x^* = x^*(\hat{f}_2)$ function can vary significantly after the discontinuous region. Therefore, it is recommended to initialize the algorithm with multiple values for Target-Objective jumps and Swap Target-Objective jump respectively in the algorithm. This ensures that if the first jump fails to yield an optimal point, the next can be employed, etc. Inevitably, failed jumps increase the computational cost of the GB method, as the Correction-step must be repeated.

Three-objective optimization, computational cost

The proposed Scan by-Layers algorithm is a low computational time GB method used to solve three-objective multi-objective optimization (MOO) problems. The main advantage of this method compared to the algorithm suggested in, [23], is that it incurs no additional EFS cost to evaluate three two-objective Pareto fronts (tracking all the border), rendering the Scan by-Layers algorithm a low-cost method. However, using areas of influence to track the border of the 3D Pareto front may mistakenly identify a discontinuity region as the border, if not combined with the exact tracking of the border method.

8.3 Proposals for Future Work

The following future research work is proposed:

First, the use of KKT Prediction residuals to track the discontinuous fronts could be used as an initial tracking tool for the Pareto Frontier. The gaps shaped on the front could then be analyzed using EA methods as suggested in, [17], in order to solidify whether a gap is a discontinuity or not (i.e Elite points existing inside the gap regions or not). A low cost, hybrid algorithm could be developed which also employs EA on top of GB techniques to track elite points in the gaps.

Moreover, the Scan by-Layers algorithm could be extended to discontinuous 3D Pareto fronts. This would require scanning the feasible border of the 3D Pareto front employing the algorithm of, [23]. The areas of influence would then be turned as a tool to detect discontinuities on the front, at the cost of increasing the computational cost of the Scan by-Layers algorithm.

Finally, the Scan by-Layers algorithm can be extended for more than three-objective optimization problems. This requires graphic visualization techniques to present Optimal Solutions for greater than three objectives.

Appendix A

Weak and Strong Dominance

In certain cases, the objective function's feasible border does not coincide with the Pareto Front of optimal solutions as demonstrated in Figure A.1. It can be deduced that the optimal solution of a given MOO problem could be feasible while being dominated by other elite points of the Pareto front.

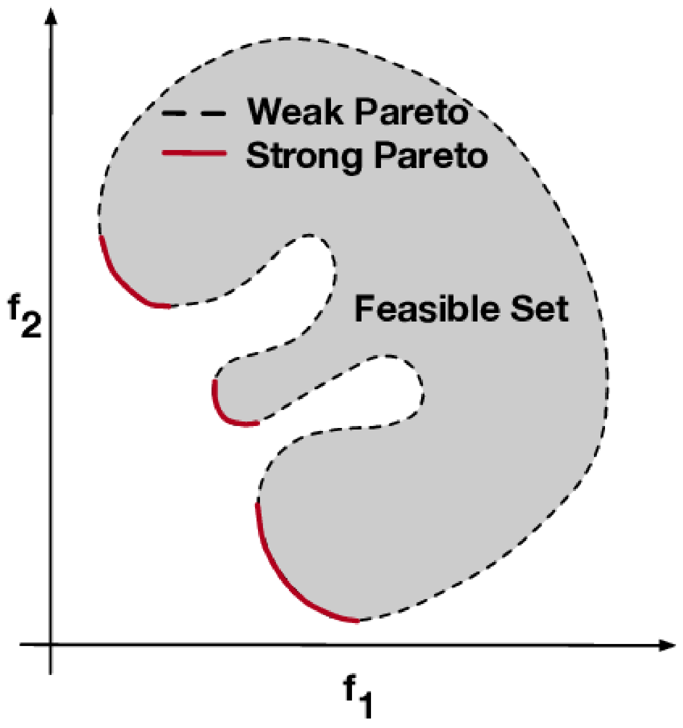


Figure A.1: An example of a border of the feasible set of solutions for an objective function that is not identical to the Pareto front of the function.

The members of the feasible border of the objective function that are not dominated constitute the strong Pareto front as displayed in Figure A.1, while the dominated points constitute the weak Pareto front.

For the mathematical definitions of weak Pareto front optimal and strong Pareto front optimal conditions for a given objective function F_{obj}, the vector of objective function values is represented as:

$$\vec{f}(\vec{\mathbf{x}}) = \begin{pmatrix} f_1(\vec{x}) & f_2(\vec{x}) & \cdots & f_m(\vec{x}) \end{pmatrix} \tag{A.1}$$

Strongly Pareto Optimal: The Pareto optimal solution x^* for (A.1) satisfies the conditions:

$$\begin{aligned} \forall j : f_i(x_j) \leq f_i(x^*),\ \text{for } i = [1, \dots, M] \\ \exists n : f_n(x_j) < f_n(x^*) \end{aligned} \tag{A.2}$$

This implies that for $x^* \in X$ to be strongly Pareto optimal, there does not exist another $x_j \in X$ such that $f_i(x_j) \leq f_i(x^*)$ for all functions f_i, $\forall i \in [1, m]$ and that, for at least one target function f_k, $f_k(x_j) < f_k(x^*)$. This definition is in accordance with that of subsection 1.3.2 and Pareto optimal conditions. Thus, strong Pareto Optimality will be referred to shortly as Pareto optimality.

Weakly Pareto Optimal: A weak Pareto point $\tilde{x}^*$ satisfies:

$$\forall j : f_i(x_j) < f_i(\tilde{x}^*),\ \text{for } i = [1, \dots, m] \tag{A.3}$$

A solution is weakly Pareto optimal if no other point exists that improves all of the objectives simultaneously. This is different from a strongly Pareto optimal point such that no point exists that improves at least one objective without being detriment to other objectives.

Appendix B

Augmented Lagrangian Method (ALM)

The Augmented Lagrangian Method (ALM) for short is an iterative algorithm developed, to handle constrained optimization problems, [15], [16].

ALM Algorithm for Equality Constraints

ALM combines the concept of the Lagrangian function with that of the Quadratic Penalty Function. The objective function is formulated mathematically as:

$$F_{\text{obj}}(\vec{x}) = f(\vec{x}) - \sum_{j=1}^{M_h} \lambda_j h_j(\vec{x}) + \frac{\omega}{2} \sum_{j=1}^{M_h} h_j^2(\vec{x}) \tag{B.1}$$

The gradient of the objective function is:

$$\nabla_x F_{\text{obj}}(\vec{x}) = \nabla_x f(\vec{x}) - \sum_{j=1}^{M_h} \lambda_j \nabla_x h_j(\vec{x}) + \omega_k \sum_{j=1}^{M_h} h_j(\vec{x}) \nabla_x h_j(\vec{x}) = 0 \tag{B.2}$$

From the above, it is deduced that for the optimal (KKT) conditions denoted by λ^* are binding for this optimization problem:

$$\lambda^* \approx \lambda_j^k - \omega_k \cdot h_k(\vec{x}) \tag{B.3}$$

This equation eliminates perturbations provided that the approximation of the Lagrange multiplier (λ^*) is satisfactory as follows:

$$h_k(\vec{x}) \approx \frac{1}{\omega_k}(\lambda^* - \lambda_j^k) \tag{B.4}$$

An iterative algorithm is thus set up, defined as:

Algorithm 10 Augmented Lagrange Method

```
Require: Given $\omega_0 > 0$, convergence tolerance $tol_0 > 0$, starting points $\vec{x_{s0}}$ and $\vec{\lambda_0}$
1: while not converged satisfied do
2:     Update design variables $x_{s(\vec{k}+1)} \leftarrow \vec{x_k}$
3:     if converging condition satisfied then
4:         stop with approximate solution $\vec{x_k}$
5:     end if
6:     Update Lagrange multipliers to obtain $\vec{\lambda_{k+1}}$
7:     Optional: Choose new penalty parameter $\omega_{k+1} \geq \omega_k$
8:     Use a Search line method (Steepest descent or BFGS)
9: end while
```

In the software developed as part of the diploma thesis, which uses the ALM method, the scheme selected to update ω, is:

$$\omega = \min(\gamma \cdot \omega_p, \omega_{p_{\max}}) \tag{B.5}$$

Respectively $\vec{\lambda}$ is updated using :

$$\vec{\lambda}^{\,k+1} = \vec{\lambda}^{\,k} - \zeta \cdot \omega \left(\nabla_x \bar{h}(x)\right) / M_h \tag{B.6}$$

where the average $\nabla_x \bar{h}(x)$ is estimated as:

$$\nabla_x \bar{h}(x) = \frac{\sum_{i=1}^{N} \nabla_x \bar{h}(x)}{N}$$

and ζ is an over-relaxation coefficient aimed at preserving $\omega \cdot \nabla_x \bar{h}(x)$ and $\vec{\lambda}$ at the same order of magnitude, when updating the Lagrange multipliers, [9].

Generalized ALM Algorithm

The Generalized ALM takes into account the effect of imposing inequality constraints to the optimization problem. The underlying concept is to incorporate the inequality constraints with additional terms in the ALM eq. (B.2). This concept can be realized by introducing slack variables z_j^2 to the inequality constraint functions $g_j(\vec{x})$ thereby transforming them into additional equality constraints. The inequality constraints are thus formulated:

$$\begin{aligned} g_1(\vec{x}) + {z_1}^2 &= 0 \\ g_2(\vec{x}) + {z_2}^2 &= 0 \\ &\vdots \\ g_{M_g}(\vec{x}) + {z_{M_g}}^2 &= 0 \end{aligned} \tag{B.7}$$

Slack variables can be treated as additional design variables that increase the dimension of the set of design variables from n to $n + M_g$. The objective function in this case is changed to:

$$F_{\text{obj}}(\vec{x}) = f(\vec{x}) - \sum_{j=1}^{M_h} \lambda_j h_j(\vec{x}) + \frac{\omega_c}{2} \sum_{j=1}^{M_h} h_j^2(\vec{x}) - \sum_{i=1}^{M_g} \mu_i (g_i(\vec{x}) + z_j^2) + \frac{\omega_g}{2} \sum_{i=1}^{M_g} (g_i(\vec{x}) + z_i^2)^2 \tag{B.8}$$

where ω_{h_j} are the penalty parameters for equality constraints and ω_{g_i} for inequality constraints. In eq (B.8), the set of design variables that leads to the global minimum is $\vec{x} = \{x_1, x_2, \ldots, x_n, z_1, \ldots, z_{mg}\}$:

$$\begin{aligned} \nabla_x F_{\text{obj}}(\vec{x}) = \nabla_x f(\vec{x}) - \sum_{j=1}^{M_h} \lambda_j \nabla_x h_j(\vec{x}) \\ + \omega_c \sum_{j=1}^{M_h} \nabla_x h_j(\vec{x}) \cdot h_j(\vec{x}) \\ - \sum_{i=1}^{M_g} \mu_i \nabla_x g_i(\vec{x}) + \omega_g \sum_{i=1}^{M_g} 2 \cdot (g_i(\vec{x}) + z_j^2) \cdot \nabla_x g_i(\vec{x}) \end{aligned} \tag{B.9}$$

From eq B.9, it is deduced that the relation used to optmize μ_i is:

$$\min \left\{ \mu_i + \omega_g [g_i(\vec{x}) + z_j^2] + \frac{1}{2}[g_i(\vec{x}) + z_j^2]^2 \right\} \tag{B.10}$$

which, by estimating the gradient of this term, is formulated into:

$$z_j^2 = -\{ \left[\frac{\mu_i}{c} + g_i(\vec{x}) \right] \} \tag{B.11}$$

Thus, the value of slack variables can be recomputed after each iteration. Given that slack variables have to be positive or zero the following expression can be given:

$$z_j^2 = \max\{0, -\left[\frac{\mu_i}{c} + g_i(\vec{x}) \right] \} \tag{B.12}$$

It is possible to reformulate (B.12) by adding the inequality constraint:

$$g_i(\vec{x}) + z_j^2 = \max\{g_i(\vec{x}), g_i(\vec{x}) - \left[\frac{\mu_i}{c} + g_i(\vec{x}) \right] \} \tag{B.13}$$

By setting $g_i^+(\vec{x}) = g_i(\vec{x}) + z_j^2$, (B.9) reads:

$$F_{\text{obj}}(\vec{x}) = f(\vec{x}) - \sum_{j=1}^{M_h} \lambda_j h_j(\vec{x}) + \frac{\omega_c}{2} \sum_{j=1}^{M_h} h_j^2(\vec{x}) - \sum_{i=1}^{M_g} \mu_i g_i^+(\vec{x}) + \frac{\omega_g}{2} \sum_{i=1}^{M_g} (g_i^+(\vec{x}))^2 \tag{B.14}$$

Using this expression both equality and inequality constraints can be handled and thus the ALM can be generalized.

Appendix C

The Implicit Function Theorem

Let $[H : R^{n+m} \rightarrow R^m]$ be a continuously differentiable function, and let R^{n+m} have coordinates $[(\vec{y}, \vec{z})]$.Where R^{n+m}, is defined as the Cartesian
product $R^n \cdot R^m$, [18].

Lets consider the point $[(\vec{y}, \vec{z}) = (a, b) \in R^{n+m}]$. For these values, [H(a,b)=0], where 0 is the Zero vector. The Jacobian w.r.t. $\vec{z}$ is denoted as:

$$J_{Hz}(a, b) = [\frac{\partial H_i}{\partial z_j}(a, b)] \tag{C.1}$$

And it is assumed to constitute the left-hand panel of the Jacobian matrix. Respectively, the Jacobian w.r.t. $\vec{z}$ is denoted as:

$$J_{Hy}(a, b) = [\frac{\partial H_i}{\partial y_j}(a, b)] \tag{C.2}$$

and this constitutes the right-hand panel of the Jacobian matrix.

The implicit function theorem states that If $J_f^z : R^n \rightarrow R^m$ is invertible, then there exists an open set $U \subset R^n$ that includes $\mathbf{a}$ and guarantees the existence of a unique function $g : U \rightarrow R^m$ such that $g(\mathbf{a}) = \mathbf{b}$ and $f(\mathbf{x}, g(\mathbf{x})) = \mathbf{0}$ for every $\mathbf{x} \in U$. Additionally, the function g is continuously differentiable, and the left-hand side of the Jacobian matrix from the earlier section is denoted as follows: :

$$\left[\frac{\partial g_i}{\partial z_j}(\vec{z})\right]_{m\times n} = -\left[J_{f_z}(\vec{z}, g(\vec{z}))\right]^{-1}_{m\times m} \left[J_{H_y}(\mathbf{y}, g(\vec{y}))\right]_{m\times n} \tag{C.3}$$

which can be also stated as:

$$\nabla_{\vec{z}} g(\vec{z}) = -\left[\nabla_{\vec{y}} H(\vec{z}, g(\vec{z}))\right]^{-1} \nabla_{\vec{z}} H(\vec{z}, g(\vec{z})) \tag{C.4}$$

Applying the Implicit Function Theorem to Track the Pareto Front

The implicit function theorem expresses the derivatives of the design variables and Lagrange Multipliers w.r.t. the target functions that are set as constraints. Eqs. (3.4) when differentiated with regards to $\vec{z}$ and $\vec{y}$ yield the following expressions:

$$\vec{\nabla}_{\vec{z}} \vec{y} = \left[\frac{\partial \vec{x}}{\partial \hat{f}_{2,\ldots,M}}, \frac{\partial \vec{x}}{\partial \lambda_{f2,\ldots,fM_t}}, \frac{\partial \vec{x}}{\partial \lambda_{h,\ldots,Mh}}, \frac{\partial \vec{x}}{\partial \mu_{1,\ldots,Mg}}\right] \tag{C.5}$$

$$\vec{\nabla}_{\vec{z}} H(\vec{z},\vec{y}) = \left[\nabla^2_{\vec{x},\hat{f}_k} L(\vec{x},\vec{\lambda_f},\vec{\lambda_h},\vec{\mu}), \nabla_{\hat{f}_k}(f_k(\vec{x}) - \hat{f}_k), \nabla_{\hat{f}_k} h(\vec{x}), \nabla_{\hat{f}_k} g(\vec{x})\right] \tag{C.6}$$

$$\vec{\nabla}\vec{y} H(\vec{z},\vec{y}) = \left[\nabla_{\vec{x}} H(\vec{z},\vec{y}), \nabla_{\vec{\lambda} f} H(\vec{z},\vec{y}), \nabla_{\vec{\lambda}_h} H(\vec{z},\vec{y}), \nabla_{\vec{\mu}} H(\vec{z},\vec{y})\right] \tag{C.7}$$

For eq. (C.5) the terms in the formulated matrix are defined as following:

$$\left[\frac{\partial \lambda_{f_{2,\dots,f_{Mt}}}}{\partial \hat{f}_{2,\dots,Mt}}\right] = \begin{bmatrix} \frac{\partial \lambda_{f2}}{\partial \hat{f}_2} & \frac{\partial \lambda_{f2}}{\partial \hat{f}_3} & \cdots & \frac{\partial \lambda_{f2}}{\partial \hat{f}_{Mt}} \\ \frac{\partial \lambda_{f3}}{\partial \hat{f}_2} & \frac{\partial \lambda_{f3}}{\partial \hat{f}_3} & \cdots & \frac{\partial \lambda_{f3}}{\partial \hat{f}_{Mt}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \lambda_{f_{Mt}}}{\partial \hat{f}_2} & \frac{\partial \lambda_{f_{Mt}}}{\partial \hat{f}_3} & \cdots & \frac{\partial \lambda_{f_{Mt}}}{\partial \hat{f}_{Mt}} \end{bmatrix} \tag{C.8}$$

$$\left[\frac{\partial \lambda_{h_{1,\dots,h_{Mt}}}}{\partial \hat{f}_{2,\dots,Mt}}\right] = \begin{bmatrix} \frac{\partial \lambda_{h1}}{\partial \hat{f}_2} & \frac{\partial \lambda_{h1}}{\partial \hat{f}_3} & \cdots & \frac{\partial \lambda_{h1}}{\partial \hat{f}_{Mt}} \\ \frac{\partial \lambda_{h2}}{\partial \hat{f}_2} & \frac{\partial \lambda_{h2}}{\partial \hat{f}_3} & \cdots & \frac{\partial \lambda_{h2}}{\partial \hat{f}_{Mt}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \lambda_{hMh}}{\partial \hat{f}_2} & \frac{\partial \lambda_{hMh}}{\partial \hat{f}_3} & \cdots & \frac{\partial \lambda_{hMh}}{\partial \hat{f}_{Mt}} \end{bmatrix} \tag{C.9}$$

$$\left[\frac{\partial \mu_{1,\dots,Mg}}{\partial \hat{f}_{2,\dots,Mt}}\right] = \begin{bmatrix} \frac{\partial \mu_1}{\partial \hat{f}_2} & \frac{\partial \mu_1}{\partial \hat{f}_3} & \cdots & \frac{\partial \mu_1}{\partial \hat{f}_{Mt}} \\ \frac{\partial \mu_2}{\partial \hat{f}_2} & \frac{\partial \mu_2}{\partial \hat{f}_3} & \cdots & \frac{\partial \mu_2}{\partial \hat{f}_{Mt}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \mu_{Mg}}{\partial \hat{f}_2} & \frac{\partial \mu_{Mg}}{\partial \hat{f}_3} & \cdots & \frac{\partial \mu_{Mg}}{\partial \hat{f}_{Mt}} \end{bmatrix} \tag{C.10}$$

It is obvious that in eq. (C.6), $\nabla^2_{\vec{x},\hat{f}_k} L(\vec{x},\vec{\lambda}_f,\vec{\lambda}_h,\vec{\mu}) = 0$ and the same applies for $\nabla_{\hat{f}_k} h(\vec{x}) = \vec{0}$ and $\nabla_{\hat{f}_k} g(\vec{x}) = \vec{0}$. When differentiating w.r.t. $\hat{f}_k$, the target constraint $\nabla_{\hat{f}_k}(f_k(\vec{x}) - \hat{f}_k) = -\vec{\mathbf{I}}$

Finally, the terms of eq. (C.7) are computed as:

$$\nabla_{\vec{x}} H(\vec{z},\vec{y}) = \begin{bmatrix} \nabla^2_{\vec{x}} L(\vec{x},\vec{\lambda_f},\vec{\lambda_h},\vec{\mu}) \\ \nabla_{\vec{x}} f_k(\vec{x}) \\ \nabla_{\vec{x}} h(\vec{x}) \\ \nabla_{\vec{x}} g(\vec{x}) \end{bmatrix} \tag{C.11}$$

$$\nabla_{\lambda_{fk}} H(\vec{z},\vec{y}) = \begin{bmatrix} \nabla^2_{x,\lambda_{fk}} L(\vec{x},\vec{\lambda_f},\vec{\lambda_h},\vec{\mu}) \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \end{bmatrix} \tag{C.12}$$

$$\nabla_{\lambda_h} H(\vec{z},\vec{y}) = \begin{bmatrix} \nabla^2_{\vec{x},\lambda_h} L(\vec{x},\vec{\lambda_f},\vec{\lambda_h},\vec{\mu}) \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \end{bmatrix} \tag{C.13}$$

$$\nabla_{\mu} H(\vec{z},\vec{y}) = \begin{bmatrix} \nabla^2_{\vec{x},\mu} L(\vec{x},\vec{\lambda_f},\vec{\lambda_h},\vec{\mu}) \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \end{bmatrix} \tag{C.14}$$

The right-hand sides of eqs. (C.11),(C.12),(C.13),(C.14) are written as:

$$\nabla_{\vec{x}} H(\vec{z},\vec{y}) = \begin{bmatrix} \nabla^2_{\vec{x}} L(\vec{x},\vec{\lambda}_f,\vec{\lambda}_h,\vec{\mu}) \\ \nabla_{\vec{x}} f_k(\vec{x}) \\ \nabla_{\vec{x}} h(\vec{x}) \\ \nabla_{\vec{x}} g(\vec{x}) \end{bmatrix} \tag{C.15}$$

$$\nabla_{\lambda_{fk}} H(\vec{z},\vec{y}) = \begin{bmatrix} -\nabla_{\vec{x}} f_k(\vec{x}) \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \end{bmatrix} \tag{C.16}$$

$$\nabla_{\lambda_h} H(\vec{z},\vec{y}) = \begin{bmatrix} -\nabla_{\vec{x}} h(\vec{x}) \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \end{bmatrix} \tag{C.17}$$

$$\nabla_{\mu} H(\vec{z},\vec{y}) = \begin{bmatrix} -\nabla_{\vec{x}} \mu(\vec{x}) \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \end{bmatrix} \tag{C.18}$$

The derivation of the Jacobian matrix of constraints can be made from eqs. (C.15),(C.16),(C.17),(C.18):

$$J(x)^T = [\nabla_{\vec{x}} f_k(\vec{x}), \nabla_{\vec{x}} h(\vec{x}), \nabla_{\vec{x}} \mu(\vec{x})] \tag{C.19}$$

Thus, the following system needs to be solved:

$$\left.\overbrace{\begin{bmatrix} \nabla^2_{xx} L & -J_k^T \\ J_k & 0 \end{bmatrix}}^{\tilde{A}}\right| \overbrace{\begin{bmatrix} \frac{\partial\ \vec{x}}{\partial \hat{f}_{2,\dots,Mt}} \\ \frac{\partial \lambda_{f_{2,\dots,Mt}}}{\partial \hat{f}_{2,\dots,Mt}} \\ \frac{\partial \lambda_{h_{1,\dots,Mh}}}{\partial \hat{f}_{2,\dots,Mt}} \\ \frac{\partial \mu^*_{1,\dots,M_g} g}{\partial \hat{f}_{2,\dots,Mt}} \end{bmatrix}}^{X} = \overbrace{\begin{bmatrix} \tilde{0} \\ \tilde{I} \\ \tilde{0} \\ \tilde{0} \end{bmatrix}}^{B} \tag{C.20}$$

The dimensions for the matrices in eq. (C.20) are the following:

Matrix A:	$[(n + (M_t - 1) + M_h + M_g) \times (n + (M_t - 1) + M_h + M_g)]$
Matrix X:	$[(n + (M_t - 1) + M_h + M_g) \times 1]$
Matrix B:	$[(n + (M_t - 1) + M_h + M_g) \times 1]$

Using the Implicit Function theorem's expression (C.4) presented in Appendix C, equation (C.20) is formulated as:

$$\frac{\partial \vec{h}}{\partial \vec{z}}(\vec{z}) = -\frac{\partial \vec{H}}{\partial \vec{z}}[\vec{h}(\vec{y}),\vec{y}]^{-1} \cdot \frac{\partial \vec{H}}{\partial \vec{y}}[\vec{h}(\vec{y}),\vec{y}] \tag{C.21}$$

From eq. (C.21) it becomes clear that this first-order accuracy scheme can effectively be employed to predict the value of an unknown Pareto point on the curve, provided a target value has been set for its M_t-1 variables. The primary challenge in employing eq. (C.20) lies in estimating the hessian $\nabla^2_{xx} L$ which due to its high computational cost will be estimated using BFGS or SR-1.

Bibliography

[1] A.Dell'Aere, O.Schütze, and M.Dellnitz. On continuation methods for the numerical treatment of multi-objective optimization problems. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2005.

[2] B.Einarsson. *Accuracy and reliability in scientific computing*. SIAM, 2005.

[3] B.Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

[4] C.Fonseca and P.Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. pages 416–423, July 17-22 1993.

[5] C.Mattson, A.Mullur, and A.Messac. Smart pareto filter: Obtaining a minimal representation of multiobjective design space. *Engineering Optimization*, 36(6):721–740, 2004.

[6] D.Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014 Chapter 3.

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

[8] I.Vasilopoulos. Cad-based and cad-free aerodynamic shape optimization of turbomachinery blade rows using the adjoint method. *PhD Thesis*, 2020.

[9] J.Nocedal and S.Wright. *Numerical optimization*. Springer, 1999, Chapter 17.

[10] J.Patel and C.Read. *Handbook of the normal distribution*, volume 150. CRC Press, 1996.

[11] K.Giannakoglou. *Optimization Methods in Aerodynamics*. NTUA, 2006.

[12] K.Giannakoglou. The easy (evolutionary algorithms system) software. *https://velos0.ltt.mech.ntua.gr*, 2008.

[13] K.Gkaragkounis. The continuous adjoint method in aerodynamic and conjugate heat transfer shape optimization, for turbulent flows. *PhD Thesis*, 2020.

[14] L.Wasserman. *All of nonparametric statistics*. Springer Science & Business Media, 2006.

[15] M.Hestenes. Multiplier and gradient methods. *Journal of optimization theory and applications*, 4(5):303–320, 1969.

[16] M.Powell. A method for nonlinear constraints in minimization problems. *Optimization*, pages 283–298, 1969.

[17] R.Lussier. Filling gaps on the pareto front in multi-and many-objective optimization. *Scholarly Horizons: University of Minnesota, Morris Undergraduate Journal*, 9(2):5, 2022.

[18] S.Krantz and H.Parks. *The implicit function theorem: history, theory, and applications.* Springer Science & Business Media, 2002.

[19] S.Schmidt and V.Schulz. Pareto-curve continuation in multi-objective optimization. *Pacific Journal of Optimization*, 4(2):243–258, 2008.

[20] T.Binh and U.Korn. Mobes: a multiobjective evolution strategy for constrained optimization problems. *Proceedings of the Third International Conference on Genetic Algorithms (MENDEL97)*, pages 176–182, 1997.

[21] Μ.Καρούζου. Επίλυση Προβλημάτων Βελτιστοποίησης Δύο Στόχων με Βάση τη Θεωρία της Συνέχισης του Μετώπου Pareto. *Διπλωματική Εργασία ΕΜΠ*, Σεπτέμβριος 2015.

[22] Μ.Καψής. Εντοπισμός του Μετώπου Pareto στην Πολυκριτηριακή Αεροδυναμική Βελτιστοποίηση με Μέθοδο Newton με Αποκοπή. *Διπλωματική Εργασία ΕΜΠ*, Ιούλιος 2014.

[23] Ν.Πατσαλίδης. Ανάπτυξη αλγορίθμου και λογισμικού υπολογισμού του μετώπου Pareto με αιτιοκρατική μέθοδο βελτιστοποίησης και εφαρμογές στην αεροδυναμική. *Διπλωματική Εργασία ΕΜΠ*, Φεβρουάριος 2020.

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Μηχανολόγων Μηχανικών
Τομέας Ρευστών
Μονάδα Παράλληλης Υπολογιστικής Ρευστοδυναμικής
& Βελτιστοποίησης

Αιτιοκρατική Πολυκριτηριακή Βελτιστοποίηση, για Ασυνεχή και 3Δ μέτωπα Pareto, με εφαρμογές στην Αεροδυναμική

Εκτενής Περίληψη στην Ελληνική

Ελευθέριος Κοκάλης

Επιβλέπων: Κυριάκος Χ. Γιαννάκογλου, καθηγητής ΕΜΠ

Αθήνα, 2024

Εισαγωγή

Στη διπλωματική αυτή εργασία, παρουσιάζεται μια αιτιοκρατική μέθοδος πολυκριτηριακής βελτιστοποίησης (MOO) για ανίχνευση του μετώπου Pareto και επεκτείνεται σε εφαρμογές CFD. Η μέθοδος ανιχνεύει αποτελεσματικά το μέτωπο Pareto με χαμηλό υπολογιστικό κόστος, για προβλήματα δύο ή τριών στόχων. Προτείνονται νέοι αλγόριθμοι για αντιμετώπιση προκλήσεων όπως τον εντοπισμό ασυνεχών περιοχών και σάρωση τέτοιων 2Δ μέτωπων Pareto, και τον εντοπισμό μετώπων τριών στόχων. Στο πλαίσιο της διπλωματικής εργασίας αναπτύσσεται σχετικό λογισμικό σε C++.

Η μέθοδος που αναπτύσσεται χρησιμοποιεί το σχήμα Πρόβλεψης-Διόρθωσης, [1] που επιλέχθηκε, καθώς μπορεί να υπολογίσει έναν αριθμό σημείων στο μέτωπο Pareto, για ένα συγκεκριμένο εύρος τιμών, με ελάχιστο υπολογιστικό κόστος. Αυτή η μέθοδος περιλαμβάνει δύο βήματα: το βήμα Go-to-Pareto, το οποίο εντοπίζει ένα αρχικό βέλτιστο σημείο, και το βήμα Move-on-Pareto, το οποίο εντοπίζει τα υπόλοιπα μη-κυριαρχούμενα σημεία σαρώνοντας το μέτωπο. Το τελευταίο είναι αυτό που εφαρμόζει το σχήμα Πρόβλεψης-Διόρθωσης που προαναφέρθηκε. Αρχικά μέσω του βήματος της πρόβλεψης είναι εφικτός ένας ικανοποιητικός πρώτος προσεγγιστικός υπολογισμός της βέλτιστης λύσης. Έπειτα το Βήμα Διόρθωσης ανιχνεύει επακριβώς τη βέλτιστη λύση και προσαρμόζεται, ώστε να λειτουργεί με τη μέθοδο ALM και τη μέθοδο SQP, προχωρώντας σε δεύτερο χρόνο στη σύγκριση των δύο. Για αυτόν τον λόγο, παρουσιάζονται δύο μαθηματικές εφαρμογές, συγκρίνοντας το υπολογιστικό κόστος και την αποδοτικότητα των ALM και SQP, [2], [3]. Το λογισμικό επεκτείνεται σε μια εφαρμογή CFD βελτιστοποίησης μορφης για μια μεμονωμένη αεροτομή, με και χωρίς περιορισμό μηδενικού συντελεστή ροπής C_M. Στη συνέχεια, προτείνεται μέθοδος ανίχνευσης ασυνεχειών στα μέτωπα Pareto, μέσω της στατιστικής επεξεργασίας των υπολοίπων σύγκλισης του βήματος πρόβλεψης, που ανιχνεύονται από τον στατιστικό αλγόριθμο του Welford, [4], και παρουσιάζεται μέθοδος για τη σάρωση τέτοιων μετώπων. Τέλος, αναπτύσσεται ο αλγόριθμος Scan by-Layers, για την επέκταση της μεθόδου-πρόβλεψης διόρθωσης σε προβλήματα τριών στόχων και επαληθεύεται σε μαθηματικές εφαρμογές, καθώς και σε εφαρμογή CFD.

Αιτιοκρατική Μέθοδος Σάρωσης του Μετώπου

Η Αιτιοκρατική μέθοδος που αναπτύσσεται στα πλαίσια της εργασίας συνίσταται σε δύο βήματα. Το πρώτο βήμα (Go-to-Pareto step), εκκινεί με βάση ένα αρχικό σημείο και προσδιορίζει το πρώτο σημείο του μετώπου. Αφού έχει προηγηθεί ο υπολογισμός ενός σημείου στο μέτωπο εκτελούνται διαδοχικά τα βήματα του Move-on-Pareto step, δηλαδή της σάρωσης του μετώπου, που χρησιμοποιεί το σχήμα Πρόβλεψης-Διόρθωσης.

Βήμα Πρόβλεψης

Το πρώτο βήμα του σχήματος είναι αυτό της πρόβλεψης. Σε μια περιοχή βέλτιστης λύσης x_i^*, η επόμενη εκτίμηση του βέλτιστου σημείου Pareto x_{i+1}^* δίνεται από:

$$\mathbf{x}^*{}_{\text{predict},i+1} = \mathbf{x}_i + \frac{\partial \vec{x}}{\partial \hat{f}_2} \delta \hat{f}_2$$

όπου η παράγωγος $\frac{\partial \vec{x}}{\partial \hat{f}_2}$ υπολογίζεται μέσω διαφόρισης των ΚΚΤ συνθηκών ως προς $\hat{f}_2$, και εντέλει καταλήγουν στον τύπο:

$$\frac{\partial H}{\partial \vec{x}} \frac{\partial x}{\partial \vec{\hat{f}}_2} = \overbrace{\begin{bmatrix} 0 \\ -I \end{bmatrix}}^{B}$$

Η μέθοδος Πρόβλεψης παρέχει μια ικανοποιητική αρχικοποίηση για το βήμα Διόρθωσης, με ελάχιστο υπολογιστικό κόστος.

Βήμα Διόρθωσης

Το Βήμα Διόρθωσης αξιολογεί το σημείο Pareto που προκύπτει από το Βήμα Πρόβλεψης. Ο στόχος είναι η ελαχιστοποίηση της συνάρτησης:

$$L(\vec{x}, \vec{\lambda}_{f_k}, \vec{\lambda}_h, \vec{\mu}) = f_1(\vec{x}) - \sum_{k=2}^{M_t} \lambda_{f_k}(f_k - \hat{f}_k) - \sum_{j=1}^{M_h} \lambda_{h_j} h_j(\vec{x}) - \sum_{i=1}^{M_g} \mu_i g_i(\vec{x})$$

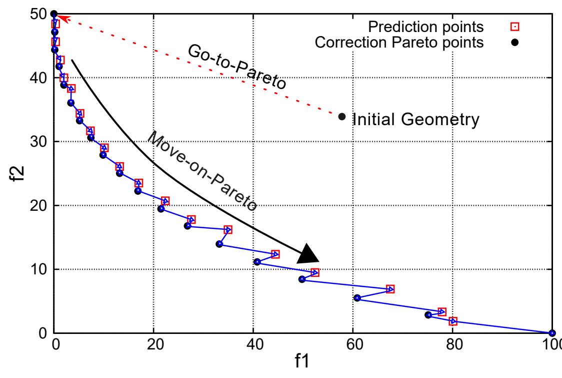
Σε προβλήματα δύο στόχων, χωρίς επιπλέον περιορισμούς με $\hat{f}_2$, το βήμα διόρθωσης γίνεται:

$$L(\vec{x}, \vec{\lambda}_{f_k}) = f_1(\vec{x}) - \lambda_{f_2}(f_2(\vec{x}) - \hat{f}_2)$$

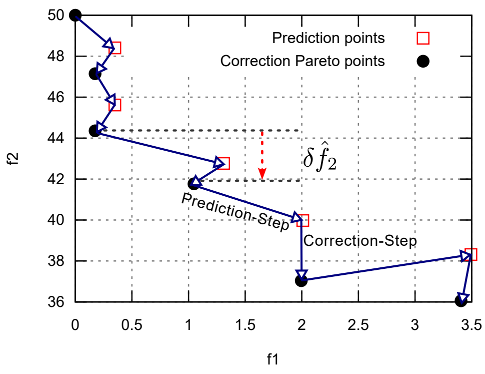
Η υλοποίηση του βήματος διόρθωσης μπορεί να γίνει και με ALM και με SQP.

Σάρωση του Μετώπου Pareto

Τα παραπάνω βήματα συνδυάζονται για να αποτελέσουν την αιτιοκρατική μέθοδο που στο πλαίσιο της εργασίας επεκτείνεται σε εφαρμογές CFD. Στα σχήματα 1 και 2 παρουσιάζεται ενδεικτικά η σάρωση του μετώπου με χρήση της μεθόδου, καθώς και το σχήμα Πρόβλεψης-Διόρθωσης.



(Σχημα 1) Σάρωση του Pareto με χρήση της αιτιοκρατικής μεθόδου.



(Σχήμα 2) Σχήμα Πρόβλεψης-Διόρθωσης.

Μαθηματικές Εφαρμογές

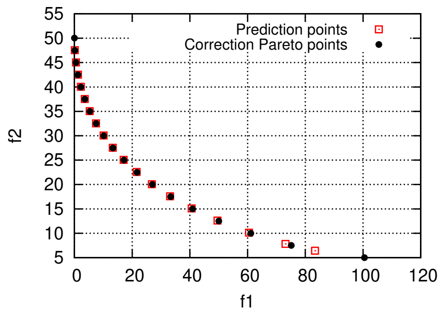
Παρουσιάζονται δύο μαθηματικές εφαρμογές, με την πρώτη να επιλύει το πρόβλημα Bihn and Korn, [5] το οποίο βελτιστοποιείται με την εφαρμογή του βήματος διόρθωσης και με SQP και με ALM. Το πρόβλημα ελαχιστοποίησης διατυπώνεται ως:

$$minf_1(x_1,x_2) = 4x_1^2 + 4x_2^2, \qquad f_2(x_1,x_2) = (x_1-5)^2 + (x_2-5)^2,$$

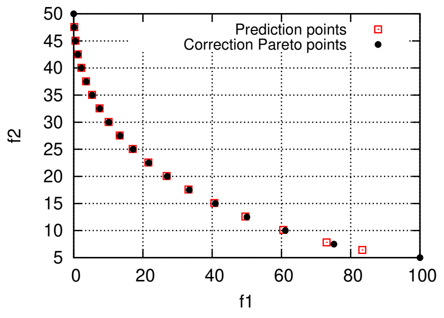
υπό τους περιορισμούς:

$$g_{1,2}(x_1,x_2) = x_1 \in [0,5] \qquad g_{3,4}(x_1,x_2) = x_2 \in [0,3]$$
$$g_5(x_1,x_2) = (x_1-5)^2 + x_2^2 \leq 25, \qquad g_6(x_1,x_2) = (x_1-8)^2 + (x_2+3)^2 \geq 7.7.$$

Τα βέλτιστα μέτωπα, για τις δύο μεθόδους προκύπτουν ως εξής (Σχήματα 1,2):



(Σχήμα 3):BP 1: Μέτωπο Pareto, με χρήση της ALM στο βήμα διόρθωσης.



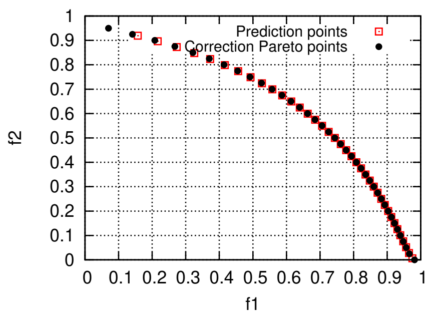
(Σχήμα 4): BP 1: Μέτωπο Pareto με χρήση του SQP στο βήμα διόρθωσης.

Η ALM, οδηγεί στην ανίχνευση του παραπάνω μετώπου, μετά από 288 κύκλους βελτιστοποίησης, σε αντίθεση με την SQP, που απαιτεί μόλις 78. Επιπλέον για 6 μεταβλητές σχεδιασμού, η ALM απαιτεί την ικανοποιητική αρχικοποίηση 12 πολλαπλασιαστών Lagrange, ενώ η SQP μια ικανοποιητική προσέγγιση του αρχικού εσσιανού.

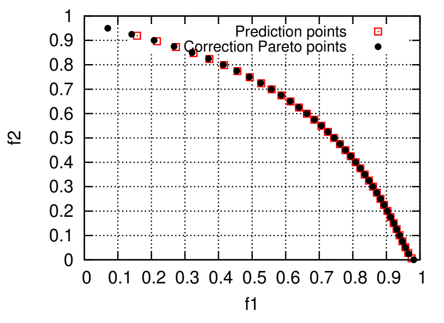
Η επόμενη μαθηματική εφαρμογή είναι του προβλήματος Fonseca and Fleming,που σχηματίζει μη κυρτό μέτωπο, και ορίζεται ως:

$$min \quad f_1(\mathbf{x}) = 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}}\right)^2\right), \qquad f_2(\mathbf{x}) = 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}}\right)^2\right),$$

$$\text{με} \quad \mathbf{x} = (x_1,x_2), \quad -4 \leq x_i \leq 4 \quad \forall \quad i = 1,2.$$



(Σχήμα 5):BP 2:Μέτωπο Pareto με τη χρήση της μεθόδου SQP, και SR-1.



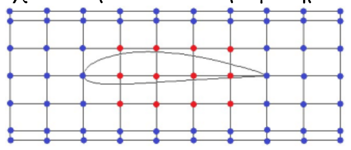
(Σχήμα 6): BP 2: Μέτωπο Pareto με χρήση του SQP, ακριβής υπολογισμός.

Το παραπάνω μέτωπο αποτελείται από 39 σημεία και σαρώνεται με βήμα $\delta f_{2,target}$ = -0.025. Παρατηρείται ότι η προσεγγιστική μέθοδος δίνει το ίδιο μέτωπο Pareto με αυτό που προκύπτει με ακριβή υπολογισμό του εσσιανού.

Εφαρμογή Δύο Στόχων: Εξωτερική ατριβής ροή, στην NACA-4415, με τη χρήση της μεθόδου Πρόβλεψης-Διόρθωσης

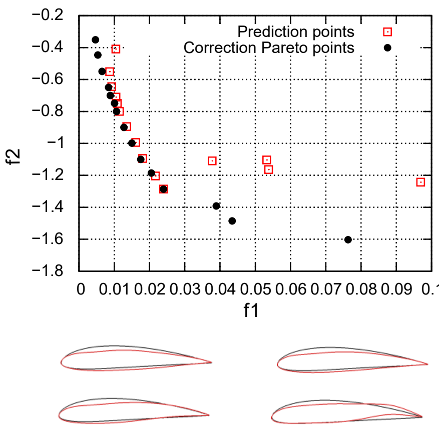
Στο επόμενο κεφάλαιο αναλύεται το πρώτο πρόβλημα CFD. Σχετίζεται με την βελτιστοποίηση του σχήματος της αεροτομής NACA 4415 σε εξωτερική ατριβή ροή, ώστε να μεγιστοποιηθεί η άνωση $f_2 = -C_L$ ενώ ελαχιστοποιείται ο συντελεστής αντίστασης $f_1 = C_D$. Η ανάλυση της παραπάνω εφαρμογής έγινε για θεώρηση συμπιεστού ρευστού ενσωματώνοντας το λογισμικό βελτιστοποίησης με το λογισμικό PUMA της Μονάδας Παράλληλης Υπολογιστικής Ρευστοδυναμικής & Βελτιστοποίησης που επιλύει της εξισώσεις ροής και τις συζυγείς εξισώσεις.

Η εξωτερική ροή αρχικοποιείται με $\alpha_\infty = 2°$ και $M_\infty = 0.8$. Η παραμετροποίηση και η αναδιάταξη του υπολογιστικού πλέγματος πραγματοποιείται χρησιμοποιώντας ένα πλέγμα μορφοποίησης NURBS 10x7, οπου οι τεταγμένες y του πλέγματος 4ξ3 (κόκκινο) είναι οι μεταβλητές σχεδιασμού του προβλήματος, (Σχήμα 7).

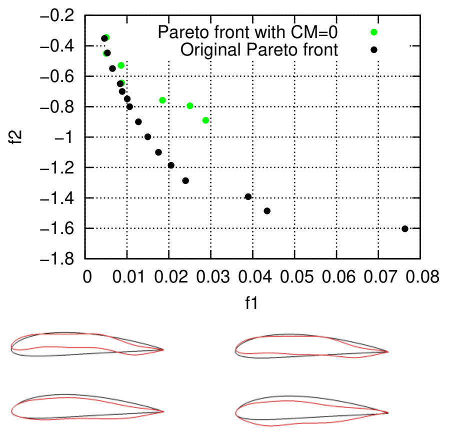


Σχήμα 7:Airfoil case two-objectives: Παραμετροποίηση NURBS, 10x7

Για την αρχικοποίηση του λογισμικού, δίνεται αρχική εκτίμηση $\hat{f}_2 = -0.35$, των διαγωνίων του εσσιανού 10 και το βήμα $\hat{\delta f}_2 = 0.1$, ανιχνεύονται 15 σημεία του μετώπου, με κόστος 135 επιλύσεων σε όρους EFS, που αντιστοιχεί σε 45 επιλύσεις των εξισώσεων ροής και 90 για τις επιλύσεις των συζυγών. Μια παραλλαγή της ίδιας εφαρμογής, παρουσιάζεται υπό τον επιπλέον περιορισμό μηδενικού συντελεστή ροπής C_M. Για την αρχικοποίηση της εφαρμογής, ανιχνεύονται 7 σημεία με αρχική εκτίμηση $\hat{f}_2 = -0.35$ και το ίδιο βήμα με πριν. (Σχήματα 8, 9)



(Σχ.8): Airfoil case two-objectives: Μέτωπο Pareto και σχετικές αεροτομές.



(Σχ.9):Airfoil case two-obj., No C_M: Μέτωπο Pareto και σχετικές αεροτομές.

Ανιχνεύοντας τις Ασυνέχειες στο Μέτωπο Pareto

Βασικό μειονέκτημα της αιτιοκρατικής μεθόδου είναι η σάρωση ασυνεχών μετώπων, καθώς δεν είναι εκ των προτέρων, γνωστή η ύπαρξη περιοχών, που δεν αντιστοιχούν σε βέλτιστες λύσεις, ή τις αποκλείουν λόγω περιορισμών. Επιπλέον η σάρωση κυριαρχούμενων περιοχών εντέλει οδηγεί και σε αύξηση του υπολογιστικού κόστους, καθώς τα σημεία που προκύπτουν δεν είναι στο μέτωπο.

Για την ανίχνευση των ασυνεχειών, συσχετίζεται η ικανοποιητικής ή μη προσέγγιση του βήματος της Πρόβλεψης με την αλλαγή καμπυλότητας του μετώπου Pareto, για (x_1*,x_2*). Με βάση την παραπάνω διαπίστωση, προτείνεται η παρακολούθηση των υπολοίπων σύγκλισης των KKT συνθηκών, μετά από κάθε βήμα Πρόβλεψης και η on-line στατιστική τους επεξεργασία, για τον υπολογισμό της μεταβολής του δείγματος. Ο αλγόριθμος στατιστικής επεξεργασίας που επιλέγεται είναι του Welford. Σε περίπτωση ανίχνευσης, υπολοίπων των KKT εκτός της αποδεκτής περιοχής, το βήμα της Διόρθωσης δεν πραγματοποιείται και προτείνεται η χρήση της μεθόδου παράκαμψης της ασυνέχειας, που παρουσιάζεται παρακάτω και συνδυάζει 3 αλγοριθμικές τεχνικές.

Σάρωση Ασυνεχών Pareto Μετώπων

- **Target-Objective jump**: Ο αλγόριθμος του Target-Objective jump υλοποιείται επιλέγοντας ένα διαφορετικό $\hat{f}_2$. Έπειτα, νέο βήμα διόρθωσης εκτελείται για το επιλεγμένο βήμα, με τις μεταβλητές σχεδιασμού που αρχικοποιείται είτε από το αρχικό σημείο του αλγορίθμου (βασική γεωμετρία στο CFD), είτε από το πιο πρόσφατο σημείο Pareto που ανιχνεύθηκε.
- **Swap Target-Objective**: Συνίσταται στην εναλλαγή της συνάρτησης στόχου $\hat{f}_2$ με την συνάρτηση προς ελαχστοποίηση f_1 της Lagrangian. Η μέθοδος μπορεί να εφαρμοστεί σε περιπτώσεις, όπου ένα εύρος τιμών του στόχου $\hat{f}_2$ αποδίδει κυριαρχούμενα σημεία ή δεν αποδίδει λύσεις λόγω περιορισμών.
- **Back-tracking**: Το Back-tracking εκτελείται αμέσως μετά το Target-Objective jump ή το Swap Target-Objective και συνίσταται στην αντιστροφή της φοράς σάρωσης του μετώπου. Με αυτόν τον τρόπο σαρώνονται σημεία που ενδεχομένως να είχαν παραλειφθεί από μια μεγάλη αύξηση στο βήμα σάρωσης κατά το άλμα. Ο αλγόριθμος διακόπτεται με την ανίχνευση νέας ασυνέχειας και επαναφέρεται στο αρχικό σημείο.

Εφαρμογή σε Μαθηματικές Συναρτήσεις

Η τρίτη μαθηματική εφαρμογή που βελτιστοποιείται, είναι η BP 3:

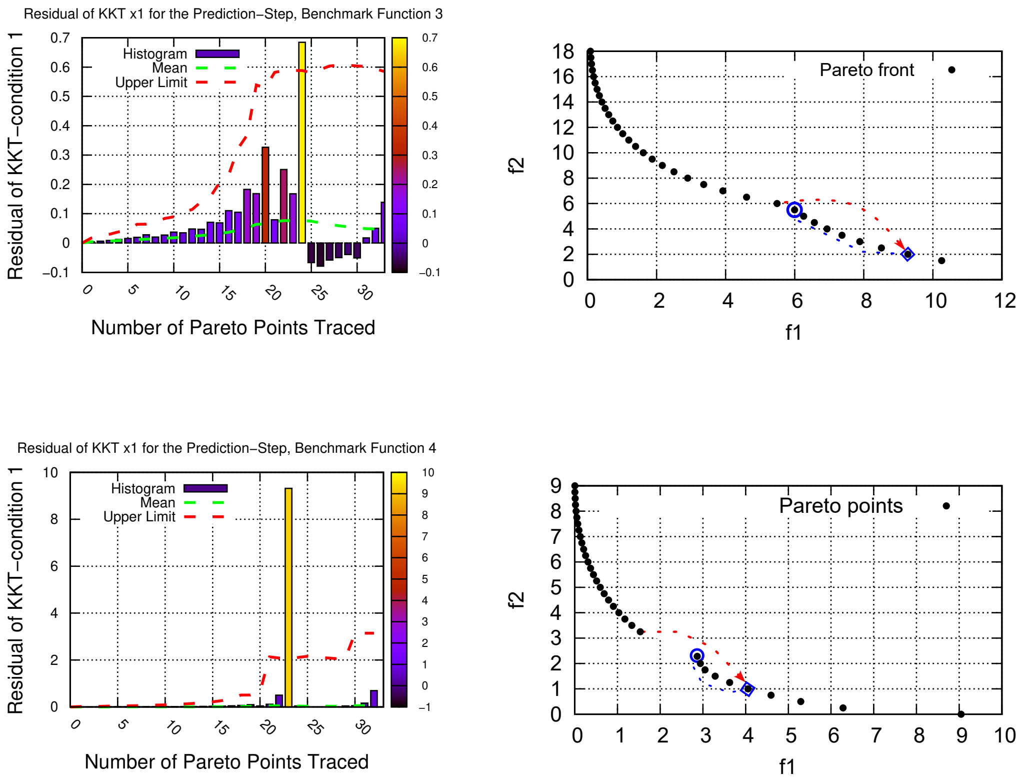
$$\min f_1(\vec{x}) = \sum_{i=1}^{n}\left((x_i-2)^2+\frac{0.1}{(x_1-3.5)^2+10^{-3}}\right) \quad f_2(\vec{x}) = \sum_{i=1}^{n}\left((x_i-5)^2+\frac{0.1}{(x_1-3.5)^2+10^{-3}}\right)$$

Οι συναρτήσεις ελαχιστοποιούνται για $\vec{x}=(x_1,x_2,x_3)$, 3 μεταβλητές σχεδιασμού. Οι όροι που είναι σε μορφή κλάσματος στις f_1, f_2 προκαλούν ασυνέχεια στο μέτωπο για τιμές του x_1 κοντά στο 3.5.

Η τέταρτη μαθηματική εφαρμογή που ελαχιστοποιείται, είναι η BP 4

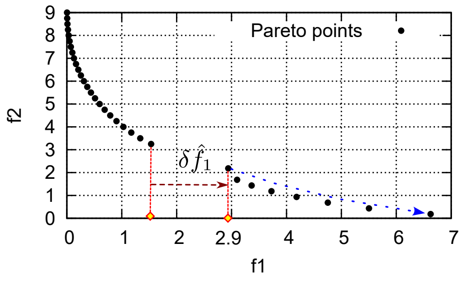
$$\min \begin{cases} f_1(\vec{x}) = \left(\frac{1}{2}(x_1+x_2)-2\right)^2 + \frac{5\times10^{-3}}{\left(\left(\frac{1}{2}(x_1+x_2)-3.5\right)^2+10^{-3}\right)} + (x_1-x_2)^2 \\ f_2(\vec{x}) = \left(\frac{1}{2}(x_1+x_2)-5\right)^2 + \frac{5\times10^{-3}}{\left(\left(\frac{1}{2}(x_1+x_2)-3.5\right)^2+10^{-3}\right)} + (x_1-x_2)^2 \end{cases}$$

Οι συναρτήσεις ελαχιστοποιούνται για δύο μεταβλητές σχεδιασμού $\vec{x}$=(x_1,x_2). Οι όροι σε μορφή κλάσματος θα προκαλέσουν μια περιοχή ασυνέχειας για ένα εύρος (x_1∗,x_2∗) γύρω από το 3.5.



Σχήμα 10: BPs 3,4: Διάγραμμα Υπολοίπων KKT συνθηκών για την μεταβλητή σχεδιασμού x_1(Αριστερά), και μέτωπα Pareto που προκύπτουν μετά την ανίχνευση της ασυνέχειας (Δεξιά).

Τα μέτωπα Pareto του σχήματος 10 σαρώνονται με συνδυασμό της χρήσης του jump για την πραγματοποίηση του άλματος και του Back-tracking για την εύρεση σημείων σε τιμές στόχων που παραλείφθηκαν από το άλμα. Σαρώνεται το μέτωπο Pareto του BP 4 και με χρήση του Swap Target-Objective για άλμα του $\delta f_{1Jump} = 5.4\delta f_1$, (Σχήμα 11):

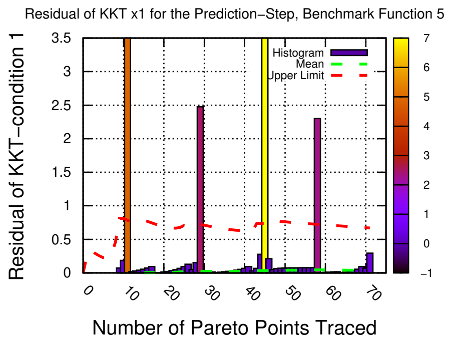


Σχήμα 11: BP 4: Χρήση Swap Target-Objective.

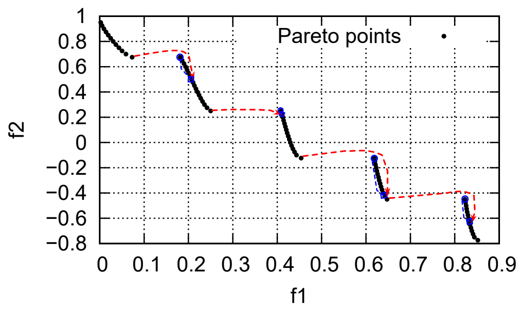
Το πέμπτο BP προς ελαχιστοποίηση είναι παραλλαγή του προβλήματος ZDT3, [6] για μια μεταβλητή σχεδιασμού x_1:

$$\min \quad \begin{cases} f_1(x) = x_1 \\ f_2(x) = (1.0)\left(1.0 - \sqrt{\frac{f_1}{1.0}} - \left(\frac{f_1}{1.0}\right)\sin(10.0\pi f_1)\right) \end{cases}$$

Το μέτωπο Pareto που προκυπτει (Σχήμα 13) αποτελείται από 4 περιοχές ασυνέχειας, οι οποίες αφού ανιχνεύονται με χρήση του αλγόριθμου ανίχνευσης των ασυνεχειών, παρακάπμτονται με τη χρήση αλμάτων Target-Objective jump. Αφού προηγηθούν τα άλματα γίνεται χρήση του Back-tracking και σαρώνονται τα σημεία που είχαν παραλειφθεί κατά το άλμα.



(Σχήμα 12): BP 5: Υπολοίπα των KKT συνθηκών.



(Σχήμα 13): BP 5: Μέτωπο Pareto

Σάρωση 3Δ Μετώπων Pareto

Η μεγάλη πρόκληση στη σάρωση μιας επιφάνειας Pareto για ένα πρόβλημα με τρεις στόχους είναι ο καθορισμός του αλγορίθμου που θα τη σαρώνει αποτελεσματικά εντοπίζοντας όλα τα σημεία στόχους. Η εύρεση των συνοριακών σημείων της επιφάνειας και κατά συνέπεια, η έγκαιρη αλλαγή της κατεύθυνσης σάρωσης, για την αποφυγή επιπλέον κύκλων βελτιστοποίησης είναι επίσης σημαντική και αντιμετωπίζεται, δεδομένης της σημασίας της στις προσομοιώσεις CFD.

Προτείνεται, ο αλγόριθμος Scan by-Layers, για την επέκταση της μεθόδου Πρόβλεψης-Διόρθωσης, για τρεις στόχους. Η βασική του ιδέα εδράζεται στην επέκταση του βήματος της Πρόβλεψης λόγω της εξάρτησης του x^* από τους $\hat{f}_2$, και $\hat{f}_3$,

$$x^* = x^*(\hat{f}_2, \hat{f}_3)$$

Επεκτείνοντας την παραπάνω εξίσωση ως σχήμα Taylor πρώτης τάξης:

$$x^*_{\text{predict},i+1} = x_i + \frac{\partial \vec{x}}{\partial \hat{f}_2}\,\delta \hat{f}_2 + \frac{\partial \vec{x}}{\partial \hat{f}_3}\,\delta \hat{f}_3$$

Εχοντας επιλέξει αρχικές τιμές για τα $\hat{f}_2$ και $\hat{f}_3$, το 3Δ-Pareto ανιχνεύεται διατηρώντας τον έναν στόχο σταθερό ενώ μεταβάλλεται ο άλλος. Για τις ακόλουθες εφαρμογές, χρησιμοποιείται ο $\hat{f}_3$ ως ο σταθερός στόχος του επιπέδου, ενώ ο $\hat{f}_2$ θα είναι ο μεταβλητός στόχος. Το βήμα της Πρόβλεψης θεωρείται έγκυρο εφόσον το σημείο που προκύπτει βρίσκεται εντός ακτίνας $\delta\rho = \delta\hat{f}_2\ /2$, από τον στόχο που έχει τεθεί. Ο χώρος που ορίζεται από την παραπάνω ακτίνα $\delta\rho$ ονομάζεται περιοχή επιρροής (area of influence). Σε διαφορετική περίπτωση, ο αλγόριθμος σαρώνει στην αντίθετη κατεύθυνση από την αρχική, αρχίζοντας απο το πρώτο σημείο έως ότου το επόμενο μη αποδεκτό σημείο ανιχνευθεί.

Με την ανίχνευση και του δεύτερου σημείου Πρόβλεψης εκτός της περιοχής επιρροής του σημείου στόχου που έχει τεθεί, το επίπεδο σάρωσης (layer) αλλάζει, δηλαδή $\hat{f}_2$, παραμένει σταθερός (είτε μεταβάλλεται) και ο $\hat{f}_3$, μεταβάλλεται Move-Layer step. Στο επόμενο επίπεδο σάρωσης (layer), η παραπάνω διαδικασία επαναλαμβάνεται μέχρι τη σάρωση όλου του αριθμού των σημείων που όρισε ο χρήστης.

Εναλλακτική προσέγγιση για την πραγματοποίηση του Move-Layer step είναι μέσω της ακριβής ανίχνευσης ενός συνοριακού σημείου του επιπέδου σάρωσης. Αυτό γίνεται μέσω βελτιστοποίησης ενός προβλήματος δύο-στόχων της Lagrangian

$$L(\vec{x}, \vec{\lambda_{f_k}}, \vec{\lambda_h}, \vec{\mu}) = f_1(\vec{x}) - \lambda_{f_3}(f_3 - \hat{f}_{3NEW}) - \sum_{j=1}^{M_h} \lambda_{h_j} h_j(\vec{x}) - \sum_{i=1}^{M_g} \mu_i g_i(\vec{x})$$
$$L(\vec{x}, \vec{\lambda_{f_k}}, \vec{\lambda_h}, \vec{\mu}) = f_2(\vec{x}) - \lambda_{f_3}(f_3 - \hat{f}_{3NEW}) - \sum_{j=1}^{M_h} \lambda_{h_j} h_j(\vec{x}) - \sum_{i=1}^{M_g} \mu_i g_i(\vec{x})$$

Το κύριο πλεονέκτημα είναι ότι σε επιφάνειες Pareto ακανόνιστου σχήματος, η εύρεση του πρώτου σημείου του επόμενου επιπέδου είναι απίθανο να επιτευχθεί από το βήμα Πρόβλεψης του προηγούμενου επιπέδου.

Επιλύονται δυο επιπλέον μαθηματικές εφαρμογές βελτιστοποίησης με τον Scan by-Layers. Η πρώτη (BP 6) είναι η συνάρτηση DTLZ-1 για τρεις μεταβλητές σχεδιασμού $\vec{x}$:

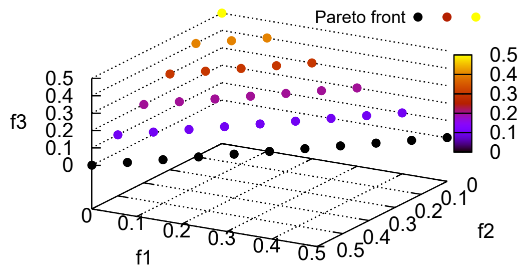
$$\min \quad (f_1, f_2, f_3)$$
$$f_1(\vec{x}) = \frac{1}{2}(1 + g(x_M))\, x_1\, x_2 \quad f_2(\vec{x}) = \frac{1}{2}(1 + g(x_M))\, x_1(1 - x_2)$$
$$f_3(\vec{x}) = \frac{1}{2}(1 - x_1)(1 + g(x_M)) \quad g(\vec{x}, k) = 100\left(k + \sum_{i=n-k}^{n-1}(x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5))\right)$$

Η επιφάνεια αποτελείται από 36 σημεία, με βήμα $\hat{\delta f_2}$ = -0.05, $\hat{\delta f_3}$ = -0.1. Οι διαγώνιοι του εσσιανού μητρώου αρχικοποιούνται με την τιμή 1, (Σχήμα 14)

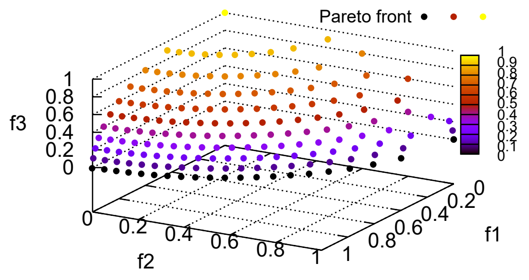
Αντίστοιχα, ελαχιστοποιείται και η DTLZ-2, η οποία ορίζεται ως:

$$\min \quad (f_1, f_2, f_3)$$
$$f_1(\vec{x}) = (1 + g(x_M))cos(\frac{\pi}{2}x_1)cos(\frac{\pi}{2}x_2) \quad f_2(\vec{x}) = (1 + g(x_M))cos(\frac{\pi}{2}x_1)\, sin(\frac{\pi}{2}x_2)$$
$$f_3(\vec{x}) = (1 + g(x_M))\, sin(\frac{\pi}{2}x_1) \qquad g(\vec{x}, k) = \sum_{i=n-k}^{n-1}(x_i - 0.5)^2$$

Η επιφάνεια αποτελείται από 174 σημεία, με βήμα $\hat{\delta f_2}$ = -0.05, $\hat{\delta f_3}$ = -0.1. Οι διαγώνιοι του εσσιανού μητρώου αρχικοποιούνται με την τιμή 1. (Σχήμα 17)



(Σχήμα 14): BP 6: 3Δ Pareto

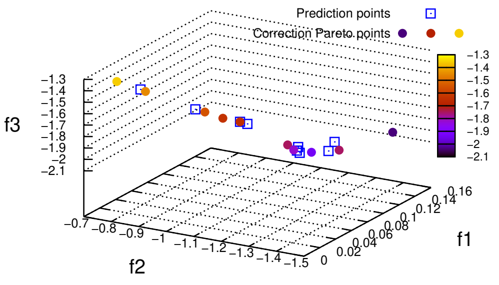


(Σχήμα 15): BP 7: 3Δ Pareto

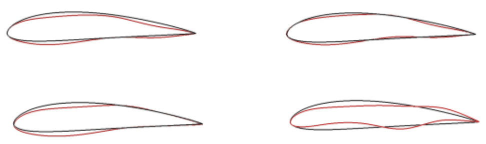
Εφαρμογή Τριών Στόχων: Εξωτερική Ατριβής Ροή σε αεροτομή

Η τελευταία εφαρμογή CFD είναι μια παραλλαγή της βελτιστοποίησης μορφής για την αεροτομή NACA-4415, για τρεις στόχους. Το πρόβλημα στοχεύει στο να βελτιστοποιηθεί στο σχήμα της αεροτομής, ώστε να μεγιστοποιηθεί η άνωση $f_2 = -C_L$ με ελαχιστοποίηση του συντελεστή αντίστασης $f_1 = C_D$, με ταυτόχρονη επίτευξη σε συνθήκες απογείωσης ($M_\infty = 0.23$, $a_\infty = 9°$), μέγιστου $f_{3,\text{target}} = -C_{L,\text{TO}}$. Η επίλυση ισοδύναμων εξισώσεων ροής, του προβλήματος με τη χρήση του Scan by-Layers, ανέρχεται σε 314, (134 για την επίλυση των εξισώσεων ροής και 180 για την επίλυση των συζυγών εξισώσεων).

Η αρχικοποίηση του αλγόριθμου γίνεται με δύο τιμές για τις διαγώνιες του εσσιανού (10 και 100), με τη δεύτερη να χρησιμοποιείται σε αποτυχία σύγκλισης του βήματος διόρθωση, με την πρώτη τιμή. Τα βήματα σάρωσης είναι, $\delta\hat{f}_2$= -0.05 και $\delta\hat{f}_3$= -0.1 για συνολικά 10 σημεία. (Σχήματα 16, 17).



(Σχήματα 16):Airfoil case three-objectives Pareto μέτωπο, μέσω του Scan by-Layers



(Σχήματα 17):Airfoil case three-objectives: Βελτιστοποιημένα σχήματα αεροτομών.

Συμπεράσματα

Με την περάτωση των εφαρμογών και της ανάπτυξης των αλγορίθμων προκύπτουν τα εξής συμπεράσματα. Ο αλγόριθμος SQP αποδείχθηκε πιο εύκολος στη ρύθμιση και πιο στιβαρός από τον ALM, λόγω των απαιτήσεων ακριβούς αρχικοποίησης των μεταβλητών Lagrange και των σταθερών ποινής του ALM. Σε εφαρμογές CFD, η σάρωση του μετώπου Pareto απαιτούσε πολλαπλές επαναρχικοποιήσεις του εσσιανού για να εξασφαλιστεί η σύγκλιση του. Για τον παραπάνω λόγο, προτείνεται η δυνατότητα χρήσης πολλών τιμών αρχικοποίησης σε περίπτωση αστοχίας σύγκλισης του βήματος διόρθωσης, με την αρχική τιμή. Το μέγεθος του βήματος στον αλγόριθμο Πρόβλεψης-Διόρθωσης επηρεάζει την ανίχνευση ή μη των ασυνεχειών, ενώ επιπλέον προτείνεται η χρήση πολλαπλών τιμών για τα Target-Objective jumps σε περίπτωση αστοχίας . Τέλος, ο αλγόριθμος Scan by-Layers αποδείχθηκε χαμηλού υπολογιστικού κόστους για την επίλυση προβλημάτων τριών στόχων, αν και η χρήση περιοχών επιρροής μπορεί να οδηγήσει σε λανθασμένη αναγνώριση ασυνεχειών ως συνοριακών σημείων, εάν τα τελευταία δεν έχουν προσδιοριστεί επακριβώς.

Βιβλιογραφία

[1] A.Dell'Aere, O.Schütze, and M.Dellnitz. On continuation methods for the numerical treatment of multi-objective optimization problems. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2005.

[2] M.Hestenes. Multiplier and gradient methods. *Journal of optimization theory and applications*, 4(5):303–320, 1969.

[3] J.Nocedal and S.Wright. *Numerical optimization*. Springer, 1999, Chapter 18.

[4] B.Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

[5] T.Binh and U.Korn. Mobes: a multiobjective evolution strategy for constrained optimization problems. *Proceedings of the Third International Conference on Genetic Algorithms (MENDEL97)*, pages 176–182, 1997.

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