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

Gradient-based Aerodynamic Shape Optimization Assisted by the Transformation of the Parameterization –Studies and Assessment

Diploma Thesis

Sotirios Damianos

Supervisor: Kyriakos C. Giannakoglou, Professor NTUA

Athens, 2023

Acknowledgments

First, I would like to express my deepest gratitude to my professor, Dr. K.C. Giannakoglou, for his trust, guidance, and support throughout the journey of completing this thesis. His expertise, insightful feedback, and encouragement played an instrumental role in shaping the direction and quality of my research. I am truly grateful for the opportunity to work under his supervision.

I would also like to extend my heartfelt appreciation to Mr. E.M. Papoutsis-Kiachagias. He was an invaluable resource, always ready to lend a helping hand whenever I encountered technical difficulties or had questions regarding the intricate details of my study. His patience, knowledge, and willingness to assist me greatly contributed to the successful completion of this thesis.

Last, I would like to express my sincere gratitude to my family and friends, whose unwavering support, encouragement, and understanding sustained me throughout this academic endeavor. Their belief in my abilities, motivation during challenging times, and constant presence in my life provided the necessary foundation for me to persevere and excel in my research.

National Technical University Of Athens
school of Mechanical engineering
Fluids Section
Parallel CFD & Optimization Unit

Gradient-based Aerodynamic Shape Optimization Assisted by the Transformation of the Parameterization –Studies and Assessment

Diploma Thesis

Sotirios Damianos

supervisor: Kyriakos C. Giannakoglou, Professor NTUA

Abstract

This thesis explores two methodologies for improving the optimization process in gradient-based optimization method using the Quasi-Newton method for updating the design variables. Both methodologies are applied solely to the parameterization of the aerodynamic surface.

The first technique focuses on the impact of the Jacobian-surface matrix on the optimization problem. It investigates how the angles between the column vectors of the Jacobian, affect the optimization. Afterward, scaling the design space is examined, exploring the effect of varying magnitudes among the gradients of design variables on the optimization process. Three scaling approaches are compared, aiming to identify the optimal design space for improved optimization outcomes. All of them are generated from the variations on the singular value matrix which is obtained from performing an SVD decomposition on the Jacobian matrix. The second technique aims to reduce the number of design variables while maintaining an optimal or near-optimal solution. This is achieved by selecting the optimum subset of a design space that is produced form the original one through sensitivity-energy analysis. The energy term represents the information contained within the Jacobian matrix. In this way, the design variables with less energy can be eliminated, resulting in simplifying the optimization problem without a significant loss of surface flexibility.

Various optimization problems with diverse geometries and aerodynamic simulations were tested to evaluate the effectiveness of each technique. Three key parameters, including the quality of the solution, the computational cost, and the algorithm robustness, were considered for assessment. Results indicate that the angle between the columns of the Jacobian matrix does not impact the optimization problem, while scaling the design space using singular values significantly affects optimization performance. More precisely scaling with the square root of the design variables appears to perform better, improving optimization performance and robustness. On the other hand, the sensitivity-energy approach effectively reduces the design space, particularly in problems with a lot of design variables. It significantly simplifies the optimization process making it possible to reduce the computational time without deviating significantly from the optimal solution. However, excessively low energy amounts negatively impact optimization and are not recommended. Finally, combining both techniques does not yield improved results. Therefore, separatelly employing each method is preferable to enhance the optimization process.

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

Αιτιοκρατική Βελτιστοποίηση Μορφής στην Αεροδυναμική Υποβοηθούμενη από Μετασχηματισμό της Παραμετροποίησης –Μελέτες και Αξιολόγηση

Διπλωματική Εργασία

Σωτήριος Δαμιανός

Επιβλέπων: Κυριάκος Χ. Γιαννάκογλου, Καθηγητής ΕΜΠ

Περίληψη

Αυτή η διπλωματική εργασία εξετάζει δυο ξεχωριστές μεθόδους με σκοπό την ενίσχυση της πορείας βελτιστοποίησης για αιτιοκρατικές μεθόδους. Και οι δύο μέθοδοι εφαρμόζονται μόνο στην παραμετροποίηση της γεωμετρίας, αφήνοντας ίδιο όλο το υπόλοιπο πρόβλημα βελτιστοποίησης.

Η πρώτη μέθοδος επικεντρώνεται στην επίδραση του Ιακωβιανού πίνακα της παραμετροποίησης στην πορεία βελτιστοποίησης. Ο Ιακωβιανός πίνακας αποτελεί τον πίνακα των παραγώγων των κόμβων της παραμετροποιημένης επιφάνειας ως προς τις μεταβλητές σχεδιαμού. Πιο συγκεκριμένα, η μέθοδος ερευνά το πως η γωνία ανάμεσα στα διανύσματα-στήλες του Ιακωβιανού πίνακα επιδρά στην πορεία βελτιστοποίησης. Στη συνέχεια, εξετάζεται η κλιμάκωση του χώρου σχεδιασμού. Δοκιμάζονται τρεις διαφορετικοί συντελεστές κλιμάκωσης, οι οποίοι παράγονται επιδρώντας στις ιδιάζουσες τιμές του Ιακωβιανού πίνακα. Η δεύτερη μέθοδος, σκοπεύει στη μείωση των μεταβλητών σχεδιασμού, χωρίς όμως να χάνεται η δυνατότητα εύρεσης της βέλτιστης λύσης. Αυτό πετυχαίνεται χρησιμοποιώντας μια ενεργειακή μέθοδο, η οποία έχει τη δυνατότητα να μειώσει σημαντικά τις μεταβλητές χωρίς όμως να επιδράσει έντονα στην ευελιξία της παραμετροποιημένης επιφάνειας. Για αυτόν τον σκοπό, κρατά τις πιο σημαντικές μεταβλητές σχεδιασμού, δηλαδή αυτές με τη μεγαλύτερη ενέργεια, ώστε να ικανοποιεί ένα κατώτατο όριο συνολικής ενέργειας σε σχέση με τον αρχικό χώρο.

Για να αποφανθεί η επίδραση αυτών των μεθόδων εξετάστηκαν προβλήματα αεροδυναμικής βελτιστοποίησης. Τα κριτήρια σύγκρισης των μεθόδων αφορούσαν την επίδραση του κάθε χώρου σχεδιασμού στη βέλτιστη λύση, στο υπολογιστικό κόστος και τέλος στη σταθερότητα του αλγόριθμου. Τα αποτελέσματα έδειξαν ότι η γωνία μεταξύ των στηλών του Ιακωβιανού πίνακα δεν παίζει κανένα ρόλο στην πορεία βελτιστοποίησης. Αντιθέτως, η κλιμάκωση του χώρου σχεδιασμού έδειξε να επηρεάζει έντονα την λύση. Πιο συγκεκριμένα, κλιμακώνοντας τις μεταβλητές σχεδιασμού με τη ρίζα των ιδιαζουσών τιμών τους, επιτεύχθηκε η μεγαλύτερη βελτίωση του χώρου σχεδιασμού βελτιώνοντας και τα τρία κριτήρια σε σχέση με τον πρωτότυπο χώρο. Όσο αφορά τη

δεύτερη μέθοδο, παρατηρήθηκε ότι, για μεγάλες τιμές ενέργειας είναι ικανή να μειώσει σημαντικά το υπολογιστικό κόστος, χωρίς να αποκλίνει από τη βέλτιστη λύση. Παρόλα αυτά, μετά από ένα σημείο, χαμηλές τιμές ενέργειας κατέληγαν να επιδρούν αρνητικά στο πρόβλημα. Τέλος, δοκιμάστηκε ένας συνδυασμός των δύο αυτών μεθόδων, ο οποίος δεν απέφερε θετικά αποτελέσματα. Συνεπώς, κάθε μια από τις μεθόδους ξεχωριστά, μπορεί να προσφέρει θετικά στοιχεία στην πορεία βελτιστοποίησης.

Acronyms

CFD	Computational Fluid Dynamics
NTUA	National Technical University of Athens
PCopt	Parallel CFD & Optimization unit
NS	Navier Stokes
RANS	Reynolds Averaged Navier Stokes
AE	Adjoint Equations
NURBS	Non-Uniform Rational Basis Spline
PARSEC	Parameterization for Airfoil Shape Representation
BC	Boundary Conditions

ΕΜΠ	Εθνικό Μετσόβιο Πολυτεχνείο
ΜΠΥΡ&Β	Μονάδα Παράλληλης Υπολογιστικής Ρευστοδυναμικής & Βελτιστοποίησης
ΥΡΔ	Υπολογιστική Ρευστοδυναμική

Contents

Chapter 1

Introduction

This thesis falls within the general research area of optimization methods and, in specific, aerodynamic shape optimization. Aerodynamic shape optimization plays a crucial role in the design and improvement of various engineering systems, ranging from aircraft and automobiles to wind turbines and sports equipment [14], [13], [26]. It involves the exploration and refinement of shapes to enhance the aerodynamic performance of these systems [5].

The pursuit of aerodynamic shape optimization can be traced back to the early days of aviation when engineers and designers sought to improve the efficiency and performance of aircrafts [11]. In those early years, shape optimization was primarily a manual and iterative process, relying on the intuition and experience of designers to modify and refine the geometry of aircraft components.

With the advent of computational techniques in the latter half of the 20th century, shape optimization took a significant leap forward [15]. The development of numerical methods and computing technology enabled engineers to simulate and analyze the flow of fluids around/inside complex geometries, providing valuable insights into the behavior of different designs. This marked the beginning of a more systematic and scientific approach to shape optimization.

1.1 Optimization Techniques: An Overview

Optimization techniques form the backbone of aerodynamic design, enabling engineers to search for the most favorable design configurations. These techniques aim to minimize or maximize specific objective functions, subject to a set of constraints. In the context of aerodynamics, the objectives typically involve measures of performance, such as drag reduction, lift enhancement, or pressure distribution optimization

[14], [5]. Two broad categories of optimization techniques are widely employed in aerodynamic shape optimization: stochastic methods and gradient-based methods [21], [15].

1.1.1 Stochastic Optimization Methods

The roots of stochastic optimization can be traced back to the mid-20th century when researchers began exploring the application of statistical and probabilistic approaches to optimization problems. The field has since evolved, with significant advancements in algorithmic development and computational power, enabling the optimization of complex systems.

These optimization methods are based on principles derived from probability theory and theory of evolution [6]. These methods explore the design space through random sampling, seeking optimal designs by iteratively refining the search based on the feedback received from the evaluated designs [4], [6], [8]. Stochastic techniques, such as evolutionary algorithms, have gained popularity due to their ability to handle complex design spaces with multiple objectives and constraints [13]. They offer robustness and the potential for discovering novel design solutions that may be missed by deterministic methods [21].

1.1.2 Gradient-Based Optimization Methods

The development of numerical methods and computing technology has significantly contributed to also the advancement and extended usage of gradient-based techniques [11].

Gradient-based optimization methods, also known as deterministic optimization methods, rely on the computation of derivatives to guide the search for optimal designs [16]. These methods employ mathematical optimization algorithms, such as the steepest descent, Newton, and quasi-Newton methods, to iteratively update the design variables based on the gradient of the objective function [7], [1], [18]. Gradient-based techniques are well-suited for problems with smooth and continuous design spaces and offer efficient convergence to optimal or near-optimal designs [10].

1.2 Purpose of this Thesis

Aerodynamic shape optimization using deterministic optimization methods has become a well-established process, with designers routinely performing large-scale optimizations with hundreds of geometric design variables, which had led to enormous simulation times [10]. Therefore, the simulation cost has now become one of the major challenges faced by gradient-based optimization method [11].

As the complexity and size of engineering systems continue to grow, the computational resources required for performing shape optimization increase significantly. This poses a significant hurdle in achieving efficient and practical optimization solutions. More precisely, in engineering applications where low-cost design iterations are necessary, such as in the aerospace industry, it is crucial to minimize the time required for optimization methods. Lengthy computational times can hinder the practicality of using gradient-based methods for shape optimization.

One crucial aspect that impacts the computational cost of aerodynamic shape optimization is the parameterization of the aerodynamic shape itself. The choice of parameterization method plays a significant role in defining the effectivness and the efficiency of the optimization [2].

Parameterization refers to the representation of the shape in terms of a set of design variables or parameters. These define the geometric features that can be modified during the optimization process. The selection of an appropriate parameterization scheme is essential for achieving good optimization results.

A common parameterization approach is by a set of control points or control curves to define the shape. These control points or curves can be changed to modify the geometry of the aerodynamic surface [27]. The number and distribution of control points or curves directly influence the design space and the accuracy of the optimization process. However, a higher number of them generally leads to a more demanding optimization process, as it requires a greater number of evaluations [11], [20].

This thesis investigates the application of two distinct methods associated with the parameterization technique in order to assess their potential benefits in gradient-based optimization. In essence, alterations are made to the manner in which the geometry is parameterized, to make the optimization process faster and more robust.

The first method involves mapping the design space onto another space, while maintaining the same number of design variables but with different characteristics. This makes use of the Jacobian matrix of the parameterization. This matrix essentially represents the derivatives of the parameterized surface with respect to (w.r.t.) the design variables, and is defined as follows:

$$\mathbf{J} = \begin{bmatrix} \frac{dX_1}{db_1} & \frac{dX_1}{db_2} & \cdots & \frac{dX_1}{db_N} \\ \frac{dY_1}{db_1} & \frac{dY_1}{db_2} & \cdots & \frac{dY_1}{db_N} \\ \frac{dX_2}{db_1} & \frac{dX_2}{db_2} & \cdots & \frac{dX_2}{db_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{dY_m}{db_1} & \frac{dY_m}{db_2} & \cdots & \frac{dY_m}{db_N} \end{bmatrix} \tag{1.1}$$

$\vec{X}_i = (X_i, Y_i), (i = 1, ..., m)$ are the coordinates of the surface mesh points and $b_n (n = 1, ..., N)$ are the design variables. The number of columns is equal to the number of active design variables and the number of rows is equal to the active coordinates of the surface-mesh points.

The Jacobian matrix plays a vital role in gradient-based optimizations, as the sensitivity derivatives of the objective function depend on it [1]. This approach seeks to reformulate the design space in a manner that may be more suitable for gradient-based optimization. It achieve this by enforcing orthogonality among the columns of the Jacobian matrix by appropriately reformulating the design variables. Moreover, this method determines appropriate scaling factors for the design variables, ensuring similar magnitudes among the Jacobian columns. By doing so, it provides an opportunity for a fair comparison among all design variables which expedites the solution of the optimization problem [3].

The second method revolves around reducing the number of design variables. It first transforms the design space into one with an equal number of variables, and subsequently extracting a subset from it, with the aim of simplifying the parameterization technique. This reduction holds the potential to decrease the number of optimization cycles required to converge [27]. However, it is important to acknowledge that reducing the design space has also some risks, as it may lead to suboptimal solutions. Nonetheless, the approach presented in this diploma thesis is devised in a manner that enables the elimination of the least influential design variables, thereby simplifying the optimization problem while retaining the ideal subset, necessary for attaining the same numerical optimum as in the original design space.

It is crucial to emphasize that there is no mathematical proof to substantiate the benefits of these approaches to the optimization problem. Nevertheless, they are based on intriguing ideas found in engineering journals, which motivated their examination using the adjoint optimization tool of the PCOpt/NTUA developed in the OpenFOAM environment. The objective is to assess the impact of these parameters on the optimization problem and determine if they can really enhance the optimization process.

1.3 The OpenFOAM software

OpenFOAM (Open Field Operation and Manipulation) was initially released in 2004 by the OpenCFD company and is a widely used open-source computational fluid dynamics (CFD) software package. It offers a comprehensive set of tools and solvers for simulating and analyzing fluid flows. It employs a finite volume method for discretizing the governing equations of fluid flow, enabling accurate and efficient simulations. The software supports a wide range of turbulence models, boundary conditions, and meshing techniques, allowing users to simulate complex flow phenomena encountered in aerodynamic applications.

Moreover, OpenFOAM's open-source nature provides users with the flexibility to customize and extend the software to suit their specific needs. The PCOpt/NTUA, with expertise in the adjoint method, has developed an adjoint optimization tool inside the OpenFOAM software. This tool basically takes advantage of the OpenFOAM capacity of solving the primal problem and then solves the adjoint equations to compute the sensitivity derivatives of the objective function.

The two methods described in this thesis were developed in a new class into the existing OpenFOAM optimization tool.

1.4 Thesis Layout

The thesis layout is as follows:

- **Chapter 2** : The basic theory of gradient-based optimization. This includes the CFD solver for the solution of the primal problem, the adjoint equations, the parameterization techniques, and the process of updating the design variables.
- **Chapter 3** : The theory underlying the first method is elucidated in this section. Initially, the Jacobian matrix of the parameterization is defined. Then the mapping of the design space is explained.
- **Chapter 4** : Results of the method of Chapter 3 in selected optimization problems with comments on the efficacy of this method are presented.
- **Chapter 5** : The theory underpinning the approach of reducing the design space is delineated with results obtained using this method, along with a comparative analysis with the previous approach.
- **Chapter 6** : A summary of the aforementioned findings is provided, accompanied by potential avenues for future research.

Chapter 2

Optimization and CFD

2.1 Generalities

Aerodynamic shape optimization entails designing an aerodynamic surface to optimize specific performance characteristics. The process involves adjusting the shape of the surface to maximize or minimize objective functions, such as drag, lift, or pressure losses. This is typically accomplished through gradient-based optimization, which iteratively modifies the design variables till the best performance is achieved [7].

To begin the process of aerodynamic shape optimization, the aerodynamic surface must firstly be parameterized. This involves defining a set of design variables that can modify the shape of the surface. Once the design variables have been defined, they can be used to generate a surface geometry which will later be used for the CFD simulation.

Next step is to perform a CFD simulation of the aerodynamic body, which is called the primal problem. This involves solving the Navier-Stokes equations to obtain a numerical solution for the airflow around the body. From this simulation, various aerodynamic quantities are computed which are, then usefull, for evaluating the aerodynamic performance of the body.

To optimize the aerodynamic surface using gradient-based methods, it is necessary to compute the sensitivities of the objective function w.r.t. the design variables. This is done using the adjoint method, which involves solving a set of adjoint to the Navier-Stokes equations. The adjoint sensitivities can be used to determine how changes to the design variables affect the aerodynamic performance of the surface.

Finally, the design variables are updated according to the gradient of the objective function w.r.t. them. Their values change in a way that improves the performance of the flow developed around/inside the surface and this is repeated iteratively.

2.2 Parameterization

The parameterization of the aerodynamic surface is the first step in an aerodynamic shape optimization problem. The main objective of parameterization is to provide a flexible and efficient way of representing complex shapes that can easily be modified for the purpose of the optimization [38].

Most of the time, the equations that describe the geometry depend on the, as called, Control Points. By moving them, the geometry changes automatically, and so their coordinates form the design variables. Therefore, the optimization algorithm comes down to moving the control points in the right direction in order to get the optimal geometry. In this thesis, two different types of parameterizations are used, NURBS and PARSEC.

NURBS (Non-Uniform Rational B-Splines) parameterization is a widely used technique that provides a flexible way of defining complex shapes [28]. This method is based on the B-Spline theory allowing smooth transitions between different regions and providing an accurate representation of the shape. Detailed informations for the mathematical background of this method can be found in [38], [39], [28].

Controversally, the PARSEC parameterization is exclusively utilized for airfoils and relies on a set of geometric principles. The process defines the shape of the airfoil based on geometric properties using 11 parameters, as illustrated in Figure 2.1.

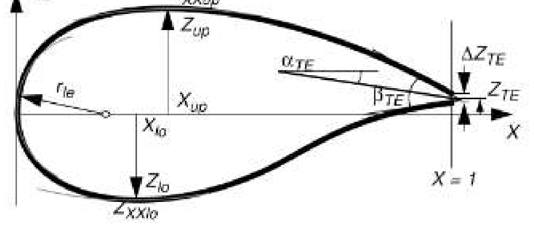


Figure 2.1: *PARSEC airfoil parameterization. From [29]*

In [1] and [29], the way that the airfoil contour is generated from these properties is provided. These geometric parameters are the design variables in the airfoil optimization.

2.3 Primal Problem

The CFD simulation solves the Navier-Stokes equations to compute the airflow around the aerodynamic surface.

First, the domain is typically discretized into a mesh with triangles or quadrilaterals [35]. The mesh can be generated using various techniques, such as structured or unstructured meshing, depending on the complexity of the geometry.

Once the mesh has been generated, numerical methods such as the finite volume method are employed to solve the Navier-Stokes equations. These methods partition the domain into small control volumes and implement the governing equations at every one of them [36]. The ensuing system of equations is commonly solved through iterative methods such as the Gauss-Seidel or the conjugate gradient method [32], [27].

The SIMPLE solver is employed for the solution of the incompressible fluid flow equations. Once the numerical solution has been obtained, various aerodynamic quantities can be computed guiding the shape optimization.

2.3.1 Primary Field Equations

The finite volume method is employed to solve the flow equations. This thesis is dealing with steady aerodynamic problems, of incompressible fluids. In other words, the density ρ remains constant across the domain, and all quantities are time-independent.

The governing equation, namely the Reynolds-Averaged-Navier-Stokes (RANS) equations, particularly the conservation of mass and momentum equation for incompressible flows, expressed in tensor notation, can be stated as follows:

$$R^p = \frac{\partial v_j}{\partial x_j} = 0 \tag{2.1}$$

$$R_i^v = v_j \frac{\partial v_i}{\partial x_j} - \frac{\partial}{\partial x_j}\left[(\nu + \nu_t)\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right)\right] + \frac{\partial p}{\partial x_i} = 0 \quad i = 1, 2 \tag{2.2}$$

the components of the velocity vector are denoted by v_j, p refers to the pressure divided by the constant density ρ, ν represents the kinematic viscosity and ν_t is known as eddy viscosity, which for laminar flows is equal to zero. This approach enables the additional turbulence stresses to be determined by augmenting the molecular viscosity with an eddy viscosity [34]. Equations 2.1 and 2.2 adhere to the Einstein notation, signifying a summation over a set of indexed terms in a formula.

2.3.2 The Spalart-Almaras turbulence model

The Spalart-Almaras turbulence model is a widely used one-equation model in the aerodynamics community due to its simplicity and computational efficiency [32], [36]. This model is based on the concept of eddy viscosity and assumes that a single scalar variable at each point, the Spalart-Allmaras variable $\tilde{\nu}$, can describe the turbulence in the flow field. The turbulent viscosity is computed using the equation:

$$\nu_t = \tilde{\nu} f_{v1} \tag{2.3}$$

where

$$f_{v1} = \frac{X^3}{X^3 + C_v^3 1}, \qquad X = \frac{\tilde{\nu}}{\nu_t} \tag{2.4}$$

The Spalart-Allmaras variable $\tilde{\nu}$ is obtained from a single differential equation, which takes the form:

$$\begin{aligned} R^{\tilde{\nu}} = v_j \frac{\partial \tilde{\nu}}{\partial x_j} + \frac{1}{\sigma}\left[\frac{\partial}{\partial x_j}\left((\nu + \tilde{\nu})\frac{\partial \tilde{\nu}}{\partial x_j}\right) - C_{b2}\left(\frac{\partial \tilde{\nu}}{\partial x_i}\right)^2\right] \\ + C_{b1}\left(1 + f_{t2}\right)\tilde{S}\tilde{\nu} + C_{w1}\left(\frac{\tilde{\nu}}{\Delta}\right)^2 = 0 \end{aligned} \tag{2.5}$$

where the last two terms are the production and dissipation terms, usually denoted by P and D respectively. The constants of equation 2.5 are $\sigma = 2/3$, $C_{b1} = 0.1355$, $C_{b2} = 0.1355$, $C_{w1} = 3.239$, $C_{v1} = 7.1$, $k = 0.41$, and Δ represents the distance from the nearest wall. Moreover, $\tilde{S}$, f_{t2}, and f_w are given by:

$$\tilde{S} = S + \frac{\tilde{\nu}}{k^2\Delta^2}, \qquad S = \sqrt{2\Omega_{ij}\Omega_{ij}}, \qquad \Omega_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i}\right) \tag{2.6}$$

$$f_w = g\left(\frac{1 + C_{w3}^6}{g^6 + C_{w3}^6}\right), \qquad g = r + C_{w2}\left(r^6 - r\right), \qquad r = \min\left[\frac{\tilde{\nu}}{\tilde{S}k^2\Delta^2}, 10\right] \tag{2.7}$$

Equations 2.6 - 2.7 utilize a distinct set of constants, except from the one defined earlier, which are detailed below:

$$C_{w2} = 0.3, \qquad C_{w3} = 2, \qquad C_{t3} = 1.2$$

The simpleFoam solver employs a variation of the model proposed by Spalart. In this formulation, a new variable, C_s, is introduced along with a new expression for $\tilde{S}$ that involves this new variable. In the original model, $\tilde{S}$ can take on negative values, which can hinder the convergence process of variables dependent on it [37]. By incorporating the C_s variable and the new $\tilde{S}$ equation, the non-negative value of $\tilde{S}$ is guaranteed. Mathematically,

$$\tilde{S} = max\left[S + \frac{\tilde{\nu}}{k^2\Delta^2}, \quad C_s S\right] \tag{2.8}$$

2.3.3 Primary Field Boundary Conditions

In order to numerically solve equations 2.1, 2.2 and 2.5 boundary conditions for velocity, pressure and turbulence quantities, must be defined.

Inlet Conditions
Dirichlet condition are imposed on the velocity v_i and the Spallart-Almaras variable $\tilde{\nu}$, while a zero Neumann condition is imposed on the pressure.

$$v_I = \text{Const}, \qquad \left(\frac{\partial p}{\partial x_i} n_i\right)_I = 0, \qquad \tilde{\nu}_I = \text{Const}$$

Outlet Conditions
Controversially, with the inlet conditions, a zero Neumann condition is imposed on the velocity and the Spallart-Almaras variable $\tilde{\nu}$, while a Dirichlet codition is imposed on pressure,

$$\left(\frac{\partial v_i}{\partial x_j}\right)_o = 0, \qquad p_o = 0, \qquad \left(\frac{\partial \tilde{\nu}}{\partial x_j}\right)_o = 0$$

Walls
The velocity on the walls, as well as $\tilde{\nu}$ are set to zero. On the other hand, a zero pressure Neumann condition is imposed,

$$v_{i,W} = 0, \qquad \left(\frac{\partial p}{\partial x_i}\right)_W = 0, \qquad \tilde{\nu}_W = 0$$

2.4 The Adjoint Method

The adjoint problem must be solved to compute the sensitivities $\frac{\delta f}{\delta b}$ of the objective function w.r.t. the design variables, which are then used to update the aerodynamic surface. This method is a powerful tool that allows for the efficient computation of sensitivities, even for large-scale problems with many design variables [16], [18].

It works by solving a dual problem, called the adjoint problem, which is closely related to the original primal problem of the CFD simulation [9]. The adjoint problem involves solving a set of equations that are derived from the Navier-Stokes equations and the objective function [24].

The problem can be solved using two kinds of methods, the discrete or the continuous adjoint method [10], [42]. In discrete adjoint, the first thing to do is to linearize and discretize the objective function and the governing equations. Then, the adjoint equations are derived from the discretized primiry field equations and the objective function, and these equations are solved numerically [12].

On the other hand, using the continuous adjoint, the adjoint equations are mathematically defined from the governing PDEs before discretizing them. More precisely, firstly the augmented objective function is defined by adding the objective function to the integral of the governing equations multiplied by the adjoint variables. After, by taking advantage of the Green-Gauss theory, the adjoint equations, the boundary conditions, and the sensitivity gradients are derived. Lastly, the adjoint partial differential equations are discretized and solved iteratively.

The advantage of the adjoint method is the fact that the time needed for computing the gradient is independent of the total number of variables [1]. The total time for computing the gradients is solely dependent on the time needed to solve the primal and the adjoint problem [25]. Additionally, the cost for solving the adjoint equations is almost the same as the one needed for solving the primal problem.

2.4.1 The Continuous Adjoint Method

The RANS equations as shown in equations 2.1 and 2.2 together with the turbulence equation developed in section 2.3.2 and the boundary conditions comprise the primal problem. The variables of the primal field $v_i \quad i = 1, 2,\ p$, and the one from the turbulence model $\tilde{\nu}$ constitute the vector $\vec{U}$.

The gradient-based methods use the derivative of the objective function F w.r.t. the design variables $\vec{b}$, $\delta F/\delta\vec{b}$, to minimize the objective function. The objective function is dependent on the vector $\vec{U}$ and the design variables vector $\vec{b} = (b_1, b_2, ..., b_N)$. Additionally, the primal field $\vec{U}$ is also influenced by the design variables. Overall, the objective function can be formulated as:

$$F = F\left(\vec{U}, \vec{U}\left(\vec{b}\right)\right) \tag{2.9}$$

This diploma thesis is dealing with two objective functions: (1) the volume-average total pressure losses between the inlet S_I and the outlet S_o of the domain, (2) The drag force on the solid walls S_w. The corrresponding objective functions are:

$$\begin{aligned} F_1 &= \int_{S_I} F_{S_I} \mathrm{d}S + \int_{S_O} F_{S_O} \mathrm{d}S \\ &= -\int_{S_I} \left(p + \frac{1}{2}v^2\right) v_i n_i \mathrm{d}S - \int_{S_O} \left(p + \frac{1}{2}v^2\right) v_i n_i \mathrm{d}S \end{aligned} \tag{2.10}$$

$$F_2 = \int_{S_W} F_{S_W} \mathrm{d}S = \left[-(\nu + \nu_t)\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right) + p\delta_i^j\right] n_i r_j \mathrm{d}S \tag{2.11}$$

In equation 2.10 and 2.11 n_i is the outward normal vector and, in equation 2.11, r_j is a unitary vector parallel to the direction of the flow.

The first step in the development of the continuous adjoint method is to define the augmented objective function which includes the residual of the flow equations of the primal solution and the adjoint fields as shown:

$$F_{aug} = F + \int_\Omega u_i R_i^v \mathrm{d}\Omega + \int_\Omega q R^p \mathrm{d}\Omega + \int_\Omega \tilde{\nu}_\alpha R^{\tilde{\nu}} \mathrm{d}\Omega + \int_\Omega \Delta_\alpha R^\Delta \mathrm{d}\Omega \tag{2.12}$$

where Ω is the computational domain u_i, q, $\tilde{\nu}_\alpha$ are the adjoint velocity, pressure and viscosity respectively, R^Δ is the adjoint to the eikonal equation for distance computations and Δ_α is the corresponding adjoint variable [30]. The purpose of introducing these adjoint variables is to ensure that the derivative of the objective function will be computed without first computing the derivatives of the flow variables w.r.t. the design variables. Therefore, according to [1], the field adjoint equations are defined as:

$$R^q = -\frac{\partial u_j}{\partial x_j} = 0 \tag{2.13}$$

$$\begin{aligned} R_i^u &= u_j \frac{\partial v_j}{\partial x_i} - \frac{\partial v_j u_i}{\partial x_j} - \frac{\partial}{\partial x_j}\left[(\nu + \nu_t)\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right] + \frac{\partial q}{\partial x_i} \\ &+ \tilde{\nu}_\alpha \frac{\partial \tilde{\nu}}{\partial x_i} = 0, \qquad i = 1,2 \end{aligned} \tag{2.14}$$

$$\begin{aligned} R^{\tilde{\nu}_\alpha} &= -\frac{\partial (v_j \tilde{\nu}_\alpha)}{\partial x_j} - \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\tilde{\nu}}{\sigma}\right)\frac{\partial \tilde{\nu}_\alpha}{\partial x_j}\right] + \frac{1}{\sigma}\frac{\partial \tilde{\nu}_\alpha}{\partial x_j}\frac{\partial \tilde{\nu}}{\partial x_j} + 2\frac{C_{b_2}}{\sigma}\frac{\partial}{\partial x_j}\left(\tilde{\nu}_\alpha \frac{\partial \tilde{\nu}}{\partial x_j}\right) \\ &+ \tilde{\nu}_\alpha \tilde{\nu} C_{\tilde{\nu}} + \frac{\partial \nu_t}{\partial \tilde{\nu}}\frac{\partial u_i}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + (-P + D)\,\tilde{\nu}_\alpha = 0 \end{aligned} \tag{2.15}$$

$$R^\Delta = k^s(\tilde{\nu}, \Delta)\,\tilde{\nu}_\alpha + \frac{\partial}{\partial x_i}\left(\Delta_\alpha \frac{\partial \Delta}{\partial x_i}\right) = 0 \tag{2.16}$$

where k^s in equation 2.16 is defined as in [30]. Thus, the sensitivity of the augmented objective function takes the form:

$$\begin{aligned} \frac{\delta F_{aug}}{\delta b_n} &= \int_S \mathbf{BC}_i^u \frac{\partial v_i}{\partial b_n}\mathrm{d}S + \int_S \mathbf{BC}^{\tilde{\nu}_\alpha}\frac{\partial \tilde{\nu}}{\partial b_n}\mathrm{d}S + \int_S \left(u_j n_j + \frac{\partial F_{S_i}}{\partial p} n_i\right)\frac{\partial p}{\partial b_n}\mathrm{d}S \\ &+ \int_S \left(-u_i n_j + \frac{\partial F_{S_k}}{\partial \tau_{ij}} n_k\right)\frac{\partial \tau_{ij}}{\partial b_n}\mathrm{d}S - \int_S \tilde{\nu}_\alpha \left(\nu + \frac{\partial \tilde{\nu}}{\sigma}\right)\frac{\partial}{\partial b_n}\left(\frac{\partial \tilde{\nu}}{\partial x_j}\right) n_j \mathrm{d}S \\ &+ \int_{S_{W_p}} n_i \frac{\partial F_{S_{W_p,i}}}{\partial x_m} n_m \frac{\delta x_k}{\delta b_n} n_k \mathrm{d}S + \int_{S_{W_p,i}} \frac{\delta n_i}{\delta b_n}\mathrm{d}S + \int_{S_{W_p}} F_{S_{W_p,i}} n_i \frac{\delta (dS)}{\delta b_n} \\ &+ \int_{S_{W_p}} \left(u_i R_i^v + q R^p + \tilde{\nu}_\alpha R^{\tilde{\nu}}\right)\frac{\delta x_k}{\delta b_n} n_k \mathrm{d}S - \int_{S_{W_p}} 2\Delta_\alpha \frac{\partial \Delta}{\partial x_j} n_j \frac{\partial \Delta}{\partial x_m} n_m n_k \frac{\delta x_k}{\delta b_n}\mathrm{d}S \end{aligned} \tag{2.17}$$

where:

$$\mathbf{BC}_i^u = u_i v_j n_j + (\nu + \nu_t)\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) n_j - q n_i + \frac{\partial F_{S_k}}{\partial v_i} n_k \tag{2.18}$$

$$\mathbf{BC}^{\tilde{\nu}_\alpha} = \tilde{\nu}_\alpha v_j n_j + \left(\nu + \frac{\nu}{\sigma}\right)\frac{\partial \tilde{\nu}_\alpha}{\partial x_j} n_j - \frac{\tilde{\nu}_\alpha}{\sigma}(1 + 2C_{b2})\frac{\partial \tilde{\nu}}{\partial x_j} n_j + \frac{\partial F_{S_k}}{\partial \tilde{\nu}} n_k \tag{2.19}$$

In equation 2.17, $S = S_I \cup S_O \cup S_W \cup S_{W_p}$, where S_{W_p} are the parameterized walls (those controlled by the design variables) whereas S_W is any other solid wall which cannot alter during the optimization.

2.4.2 Adjoint Boundary Conditions

Inlet Conditions

A zero Neumann condition is imposed on the adjoint pressure q and a zero Dirichlet condition is imposed on $\tilde{\nu}_\alpha$. Moreover, the inlet conditions imposed on the adjoint velocity are

$$u_i v_j = u_{<n>} = \frac{\partial F_{S_{I,i}}}{\partial p} n_i, \qquad u_{<t>} = \frac{\partial F_{S_{I,k}}}{\partial \tau_{ij}} n_k t_i n_j + \frac{\partial F_{S_{I,k}}}{\partial \tau_{ij}} n_k t_j n_i$$

where t_i is the tangential unitary vector component, and $u_{<t>}$ is the adjoint velocity, which is parallel to the unitary vector t_i.

Outlet Conditions

For the ν_a, in order to eliminate the multiplier of $\frac{\partial \nu}{\partial b_n}$, a Robin-type condition is applied as:

$$BC^{\tilde{\nu}_\alpha} = \tilde{\nu}_\alpha v_j n_j + \left(\nu + \frac{\nu}{\sigma}\right) \frac{\partial \tilde{\nu}_\alpha}{\partial x_j} n_j + \frac{\partial F_{S_o,k}}{\partial \tilde{\nu}} n_k = 0 \tag{2.20}$$

To eliminate term $\frac{\partial v_i}{\partial b_n}$, the following condition is imposed:

$$BC_i^u = u_i v_j n_j + (\nu + \nu_t) \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) n_j - q n_i + \tilde{\nu}_\alpha \tilde{\nu} + \frac{\partial F_{s_o,k}}{\partial v_i} n_k \tag{2.21}$$

Finally, for $u_j n_j$ a zero Neumann condition is imposed.

Wall Boundaries

In order to eliminate the term $\frac{\partial}{\partial b_n}\left(\frac{\partial \tilde{\nu}}{\partial x_j}\right) n_j$ from equation 2.17, a zero Dirichlet condition is applied for the $\tilde{\nu}_\alpha$, ($\tilde{\nu}_{\alpha,W} = 0$). The normal and the tagential adjoint velocity are imposed to be:

$$u_{<n>} = -\frac{\partial F_{S_{W,i}}}{\partial p} n_i, \qquad u_{<t>} = \frac{\partial F_{S_{W,k}}}{\partial \tau_{ij}} n_k t_i n_j + \frac{\partial F_{S_{W,k}}}{\partial \tau_{ij}} n_k t_j n_i$$

2.4.3 Sensitivity Derivatives

After applying the boundary conditions to the adjoint equations, the sensitivity derivatives can finally be computed from the following equation:

$$
\begin{aligned}
\frac{\delta F_{aug}}{\delta b_n} = & - \int_{S_{W_p}} \left[(\nu_t + \nu) \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) n_j - q n_i + \frac{\partial F_{S_{W_p}}}{\partial v_i} n_k \right] \frac{\partial v_i}{\partial x_k} n_k \frac{\delta x_m}{\delta b_n} n_m \mathrm{d}S \\
& - \int_{S_{W_p}} \left[\left(\nu + \frac{\tilde{\nu}}{\sigma} \right) \frac{\partial \tilde{\nu}_\alpha}{\partial x_j} n_j + \frac{\partial F_{S_K}}{\partial \tilde{\nu}} \right] \frac{\partial \nu}{\partial x_m} n_m n_k \frac{\delta x_k}{\delta b_n} \mathrm{d}S \\
& + \int_{S_{W_p}} n_i \frac{\partial F_{S_{W_p}}}{\partial x_m} n_m \frac{\delta x_k}{\delta b_n} n_k \mathrm{d}S + \int_{S_{W_p}} F_{S_{W_p,i}} \frac{\delta n_i}{\delta b_n} \mathrm{d}S + \int_{S_{W_p}} F_{S_{W_p,i}} n_i \frac{\delta \left(\mathrm{d}S \right)}{\delta b_n} \\
& + \int_{S_{W_p}} \left(u_i R_i^u + q R^p + \tilde{\nu}_\alpha R^{\tilde{\nu}} + \Delta_\alpha R^\Delta \right) \frac{\delta x_k}{\delta b_n} n_k \mathrm{d}S \\
& - \int_{S_{W_p}} 2 \Delta_\alpha \frac{\partial \Delta}{\partial x_j} n_j \frac{\partial \Delta}{\partial x_m} n_m n_k \frac{\delta x_k}{\delta b_n} \mathrm{d}S \\
& - \int_{S_{W_p}} \left[\left(-u_{<n>} + \frac{\partial F_{S_{W_p,k}}}{\partial \tau_{lm}} n_k n_l n_m \right) \left(\tau_{ij} \frac{\delta \left(n_i n_j \right)}{\delta b_n} + \frac{\partial \tau_{ij}}{\partial x_m} n_m \frac{\partial x_k}{\partial b_n} n_k n_i n_j \right) \right] \mathrm{d}S \\
& - \int_{S_{W_p}} \left[\frac{\partial F_{S_{W_p,k}}}{\partial \tau_{lm}} n_k t_l t_m \left(\tau_{ij} \frac{\delta \left(t_i t_j \right)}{\delta b_n} + \frac{\partial \tau_{ij}}{\partial x_m} n_m \frac{\delta x_k}{\delta b_n} n_k t_i t_j \right) \right] \mathrm{d}S
\end{aligned}
\tag{2.22}
$$

This is the complete form of the sensitivity derivatives that can be used for any objective function. However, for a specific objective function, it can be simplified. For example, for the total pressure losses objective function presented in equation 2.10, $\frac{\partial F_S}{\partial \tau_{lm}} = 0$ and thus the final two terms are eliminated. On the other hand, for the drag objective function shown in equation 2.11, $\frac{\partial F_S}{\partial u_i} = 0$ and $\frac{\partial F_S}{\partial \tau_{lm}} = p \delta_j^i$ so equation 2.22 is updated accordingly.

2.5 Update Method

The line search method assumes that every iteration produces a new set of design variables that occurs from the previous ones added with the search direction p [7]. Thus, the new set of design variables is according to:

$$b_j^{n+1} = b_j^n + \eta^n p_j^n \quad j = 1, ..., N \tag{2.23}$$

where η is a positive number called step length. In this diploma thesis the quasi-Newton method is used for computing the search direction p_n and the Armijo condition for defining the step length η.

2.5.1 Quasi-Newton BFGS

The quasi-Newton method is one of the most widely used methods for calculating the search direction [3]. This method redefines the Newton method aiming to take advantage of its fast convergence characteristics while minimizing its computational cost.

Starting from the original Newton method, the search direction is given as:

$$p_j^n = -\left[\nabla^2 F\left(b_j^n\right)\right]^{-1} \nabla F\left(b_j^n\right) \tag{2.24}$$

where $\nabla^2 F\left(b_j^n\right)$ is also known as the Hessian matrix H_{ij}^n, and it is a very important quantity for the optimization process. However, computing it, significantly increases the computational cost, making it impossible in large-scale optimization problems [23]. For this reason, the quasi-Newton method proposes to compute an approximate Hessian, avoiding the calculation of the exact Hessian. Therefore, using a finite difference scheme, can be computed as:

$$\nabla^2 F\left(b_j^{n+1}\right)\left(b_j^{n+1} - b_j^n\right) \approx \nabla F\left(b_j^{n+1}\right) - \nabla F\left(b_j^n\right) \tag{2.25}$$

A common approach for solving equation 2.25 is the BFGS (Broyden-Fletcher-Goldfarb-Shano) method which can analitically be found in [22].

Concluding, the quasi-Newton method designates the search direction equal to:

$$p_j^n = -H_{ij}^n \nabla F\left(b_j^n\right) \tag{2.26}$$

2.5.2 Armijo Conditions

The Armijo condition is a type of step size control that ensures that the objective function decreases sufficiently during the optimization process [1]. It works by first computing a descent direction using a quasi-Newton method such as BFGS. Then, the step size is iteratively decreased until the Armijo condition is satisfied. In other words, it defines an appropriate step length η which satisfies the following inequality

$$f(b_n + \eta p_n) \leq f(b_n) + c\eta \nabla f(b_n) \tag{2.27}$$

where f is the objective function, b_n is the current set of design variables, p_n is the search direction, c is a small positive constant, and ∇f is the gradient of the objective function.

The inequality states that the objective function at the new point $b_n + \eta p_n$ must be less than or equal to the objective function value at the current point x plus a small positive constant times the gradient of the objective function multiplied by the descent direction.

If the inequality is satisfied, the step length is accepted, and the design variables are updated accordingly. Otherwise, the step length is decreased, and the process is repeated until the Armijo condition is satisfied. Whenever the inequality 2.27 is not satisfied, the primal solution must be repeated leading to high computational cost.

This condition is very useful when dealing with constrained optimization problems. These types of problems reform the objective function in such a way that if any constrained quantity goes out of bounds, the new objective function significantly increases. This solution is rejected by the Armijo condition, and a new step length is tested in order to satisfy all constraints.

2.6 Optimization Algorithm

Concluding, the optimization process can be described as follows:

1. Define the geometry and create the appropriate mesh using a CFD tool.
2. Perform parameterization of the geometry and select the design variables, as was described in section 2.2. During this step, a grid displacement procedure is performed to adapt the internal grid nodes to the approximate geometry that occurs from the parameterization [18].
3. Solve the RANS equations using the primal solver to compute the aerodynamic quantities of the geometry, as was mentioned in section 2.3.
4. Check the convergence criteria, which compare the optimization quantities of the current and the previous cycle against predefined thresholds. If the convergence criteria are satisfied, the optimization process ends. Otherwise, moves on the next step.
5. Solve the adjoint equations defined in section 2.4.1 to compute the sensitivity derivatives through equation 2.22.
6. Determine the search direction using equation 2.24 and find the length direction by satisfying the Armijo condition.
7. Update the design variables according to equation 2.23 and generate the new geometry.
8. Repeat steps 3-7 until the optimization problem converges.

Chapter 3

Rotating And Scaling the Design Space

3.1 The Jacobian Matrix

The Jacobian Matrix **J** of the parameterization w.r.t. the surface mesh nodes is a critical component used in a shape optimization problem. Comprised of multiple columns, each column signifies the extent to which the surface coordinates react to a small change to one design variable, thereby capturing the sensitivity of the geometry to this variable. The expression for the Jacobian is displayed in equation 1.1.

To illustrate, let's consider the geometry depicted in Figure 3.1, which is discretized with 5 points $\vec{X}_i(i = 1, ..., 5)$. This curve is generated from as a NURBS surface with 4 control points $\vec{b}_n(i = 1, ..., 4)$.

However, the first and last control points, along with their corresponding surface nodes, are deemed fixed. Furthermore, the displacement of control points is only allowed in the y-direction. Consequently, the parameterization involves 2 active control points and 3 active surface nodes moving only in one direction. In this scenario, the Jacobian matrix will take the following form:

$$\mathbf{J} = \begin{bmatrix} dY_2/db_{2,y} & dY_2/db_{3,y} \\ dY_3/db_{2,y} & dY_3/db_{3,y} \\ dY_4/db_{2,y} & dY_4/db_{3,y} \end{bmatrix} \tag{3.1}$$

This matrix can be computationally defined by applying a finite difference scheme

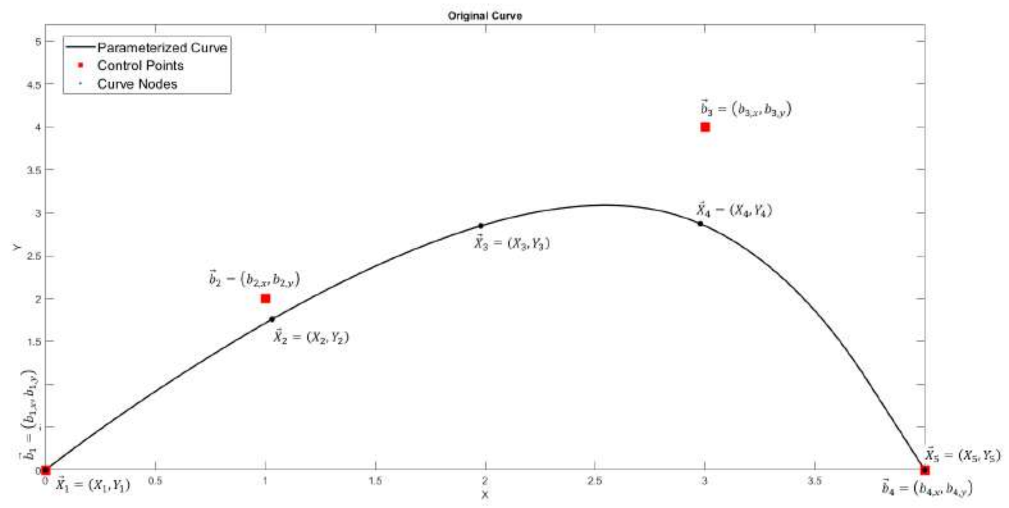


Figure 3.1: *A Curve with 5 nodes $\vec{X}_i$ generated by a NURBS parameterization consisting of 4 Control Points $\vec{b}_i$*

for a minute change in the corresponding design variable ($dy \to 0$):

$$\frac{dY_i}{db_{n,y}} = \frac{Y_{dy,i} - Y_i}{dy_n} \tag{3.2}$$

where $Y_{dy,i}$ is the y coordinate of the $i - th$ curve node, after moving the $n - th$ control point by a dy.

The displacements of each surface node and their corresponding gradients are illustrated in Table 3.1.

Control Point	Curve Nodes	dy	$Y_{dy,i} - Y_i$	$dY_i/db_{n,y}$
$\vec{b}_2$	$\vec{X}_2$	10^{-5}	$6 \cdot 10^{-6}$	0.6
	$\vec{X}_2$	10^{-5}	$2 \cdot 10^{-6}$	0.2
	$\vec{X}_2$	10^{-5}	$1 \cdot 10^{-8}$	0.001
$\vec{b}_3$	$\vec{X}_3$	10^{-5}	$2 \cdot 10^{-6}$	0.2
	$\vec{X}_3$	10^{-5}	$7 \cdot 10^{-6}$	0.7
	$\vec{X}_3$	10^{-5}	$2 \cdot 10^{-6}$	0.2

Table 3.1: *The displacement of each curve node across the y-axis for a dy displacement of each active control point and their derivatives computed using a finite difference scheme.*

Consequently, the Jacobian matrix of the parameterized surface is:

$$\mathbf{J} = \begin{bmatrix} 0.6 & 0.2 \\ 0.2 & 0.7 \\ 0.001 & 0.2 \end{bmatrix} \tag{3.3}$$

Assuming that this matrix consists of two column vectors, each one representing a distinct control point, it serves to define two pivotal parameters of the parameterized surface.

The first parameter pertains to the angle between the two columns. This angle can be calculated using the following equation:

$$\theta_{ij} = arcos\Big(\frac{\vec{u}_i^T \vec{u}_j}{|\vec{u}_i| \cdot |\vec{u}_j|}\Big) \tag{3.4}$$

where $\vec{u}_i$ and $\vec{u}_j$ are the $i - th$ and $j - th$ column of the Jacobian $\mathbf{J}$, and of course $i \neq j$.

For the example depicted in Figure 3.1, utilizing the previously defined Jacobian matrix of equation 3.3, the angle between the two control points can be determined from equation 3.4 as:

$$\theta_{12} = arcos\Big(\frac{\vec{u}_1^T \vec{u}_2}{|\vec{u}_1| \cdot |\vec{u}_2|}\Big) = 56.92^o \tag{3.5}$$

The second parameter derived from the Jacobian matrix is the magnitude of each column vector. This magnitude signifies the sensitivity of the parameterized geometry to movements of the corresponding control points. In essence, it quantifies the extent to which the geometry is influenced by a change in a design variable. Consequently, this parameter is commonly referred to as ”sensitivity”.

For the example illustrated in Figure 3.1, the sensitivity of each design variable is presented in Table 3.2, clearly demonstrating that each design variable affects the parameterized surface with varying degrees of intensity.

$\lvert d\vec{X}/db_1 \rvert$	$\lvert d\vec{X}/db_2 \rvert$
0.6324	0.75498

Table 3.2: *The magnitude of each column vector of the jacobian Matrix*

This disparity in sensitivity can be visually seen in Figure 3.2, where the control points are moved not by a very small dy, but by a larger increment of $\Delta Y = 1$. It is evident that shifting the second active control point results in a more pronounced displacement of the entire curve compared to the same movement of the first active control point. This veryfies that the design variable with higher magnitude, as shown in Table 3.2, has bigger influence on the generated curve.

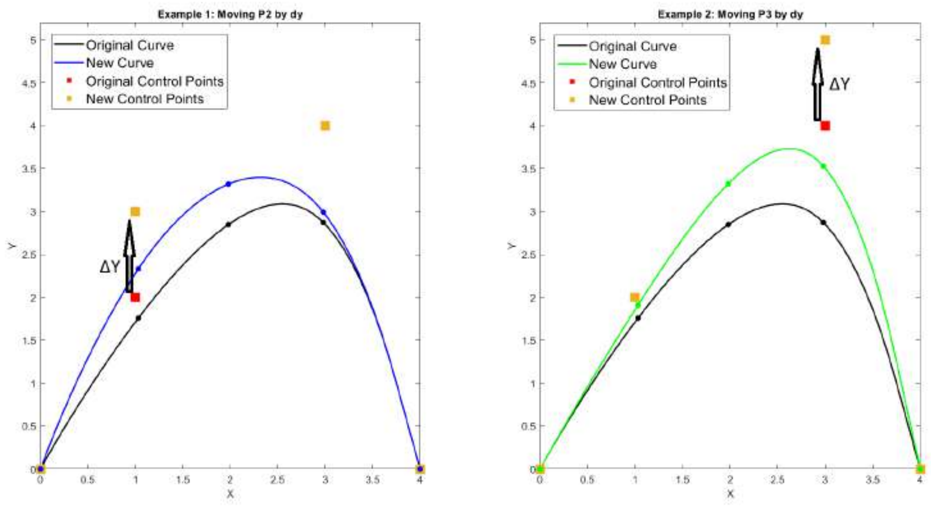


Figure 3.2: *A comparison between the sensitivity of the curve for equal displacement of the control points by a $\Delta Y = 1$.*

It is of utmost importance to highlight that the intensity of the effect of each design variable on the generated curve can be better characterized by another value that may not be immediately apparent. This value can be obtained by performing a Singular Value Decomposition (SVD) [43], [40] on the Jacobian matrix. The SVD decomposition expresses the Jacobian as the product of three matrices:

$$\mathbf{J} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathbf{T}} \tag{3.6}$$

where $\mathbf{U}$ is an $m \times m$ unitary[1] orthogonal matrix, $\mathbf{V}$ is an $N \times N$ unitary orthogonal matrix and $\boldsymbol{\Sigma}$ is $m \times N$ non-negative diagonal matrix containing the singular values. Each singular value corresponds to a specific design variable. The singular value matrix is a very useful tool for determining the variety of sensitivities of the Jacobian as it showcases how sensitive the contour/surface is to changes in that design variable in comparison with changes in the other variables. Thus, it is very usefull to use it instead of directly using the columns' magnitude of the Jacobian.

For instance, for the case presented in Figure 3.1, the singular value matrix is:

$$\boldsymbol{\Sigma} = \begin{bmatrix} 0.4596 & 0 \\ 0 & 0.8710 \end{bmatrix} \tag{3.7}$$

[1]A unitary matrix is a square matrix whose columns (or rows) have magnitudes equal to one, and the conjugate transpose of the matrix is its inverse.

verifying that the second design variable has a bigger impact on the surface than the first one.

At this point it must be clarify that, from now on, in this thesis, the term ”sensitivity” will exclusively refer to geometrical sensitivity which is defined by the magnitude of the column vector of the Jacobian matrix or, even better, by the singular values associated with each design variable. If there arises a necessity to discuss sensitivity derivatives of the objective function w.r.t. the design variables, it will be explicitly stated.

3.2 The Jacobian Matrix for Real Optimization Problems

Transitioning from a toy parameterization case to a reallistic shape optimization problem, the Jacobian Matrix assumes a considerably more substantial role. As elucidated in Chapter 2, in optimization problems, the active coordinates of the control points serve as the design variables. Consequently, the Jacobian Matrix is directly employed to determine the sensitivity derivatives of the objective function, given by the following equation:

$$\frac{\partial F}{\partial \vec{b}} = \frac{\partial F}{\partial \vec{X}_s}\frac{\partial \vec{X}_s}{\partial \vec{b}} = \frac{\partial F}{\partial \vec{X}_s}\mathbf{J} \tag{3.8}$$

Therefore, it is likely that the previously mentioned parameters will have a significant impact on the optimization problem. To gain a deeper understanding, a real-world optimization problem is introduced. This particular problem entails the minimization of the drag coefficient for the NACA0012 airfoil at a specific angle of attack, while concurrently constraining the lift, moment and volume coefficient. The volume coefficient (C_v) is defined as $C_v = \frac{V-V_{init}}{V_{init}}$, where V_{int} is the initial volume/area and V the volume after the shape optimization. The exact flow conditions and the imposed constraints are displayed in section 4.1, however there are not needed for the following application, as it focuses solely on the parameterization of the geometry.

The geometry of the airfoil is parameterized using NURBS curves, comprising a total of 32 control points. These control points are evenly distributed along the pressure and suction surfaces, as depicted in Figure 3.3.

Consequently, the Jacobian matrix takes the following form:

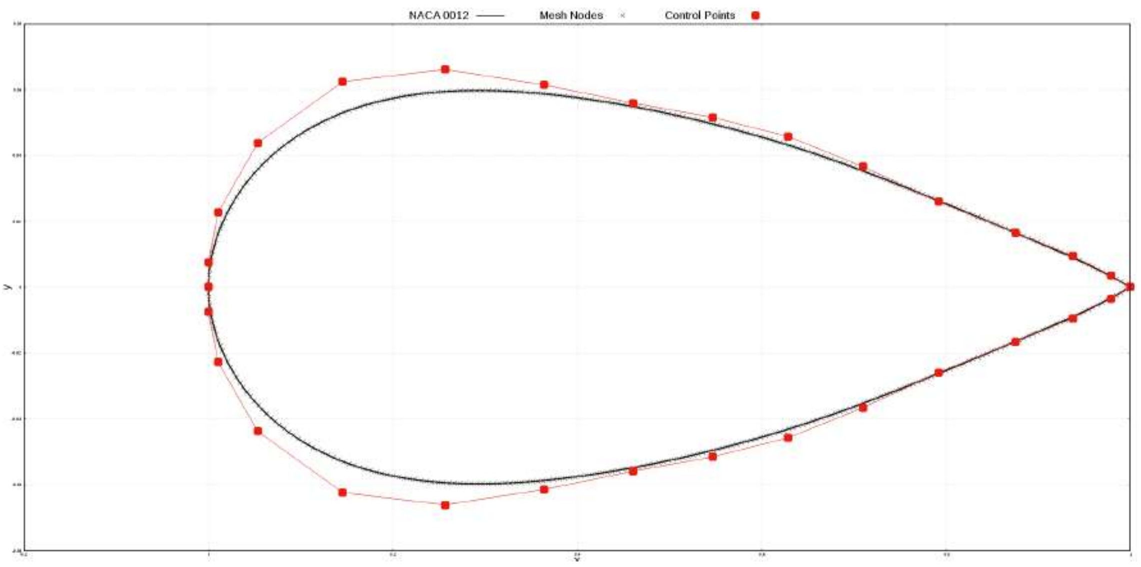


Figure 3.3: *The NACA 0012 airfoil parameterized with NURBS consisting of 32 control points.*

$$\mathbf{J} = \begin{bmatrix} \frac{dX_1}{db_1} & \frac{dX_1}{db_2} & \cdots & \frac{dX_1}{db_N} \\ \frac{dY_1}{db_1} & \frac{dY_1}{db_2} & \cdots & \frac{dY_1}{db_N} \\ \frac{dX_2}{db_1} & \frac{dX_2}{db_2} & \cdots & \frac{dX_2}{db_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{dY_m}{db_1} & \frac{dY_m}{db_2} & \cdots & \frac{dY_m}{db_N} \end{bmatrix}$$

Here, $\vec{X}_i = (X_i, Y_i)$ are the coordinates of each mesh point on the 2D airfoil contour. Moreover, m represents the number of rows and N represents the number of columns. Since every active control point and mesh node is allowed to move in both the x and y directions within the 2D design space, it follows that $m = 2\times$(Active Airfoil Nodes) and $N = 2\times$(Active Control Points).

Now, let us delve into the analysis of how the two parameters of the Jacobian matrix, as discussed earlier, vary within real optimization problems. Figure 3.4 showcases the distribution of angles between the column vectors of the Jacobian matrix for the geometry presented in Figure 3.3.

This visualization reveals that while the majority of design variable pairs exhibit an almost perpendicular relationship, there exist certain pairs that are closer to being parallel to each other. The smallest angle observed in this particular case is 22.35 degrees.

Considering that the Jacobian matrix is a component of complex numerical systems utilized throughout the optimization process, this lack of orthogonal columns could lead to ill-conditioned matrices and potentially have a detrimental impact on the numerical solution and algorithm robustness of the optimization problem.

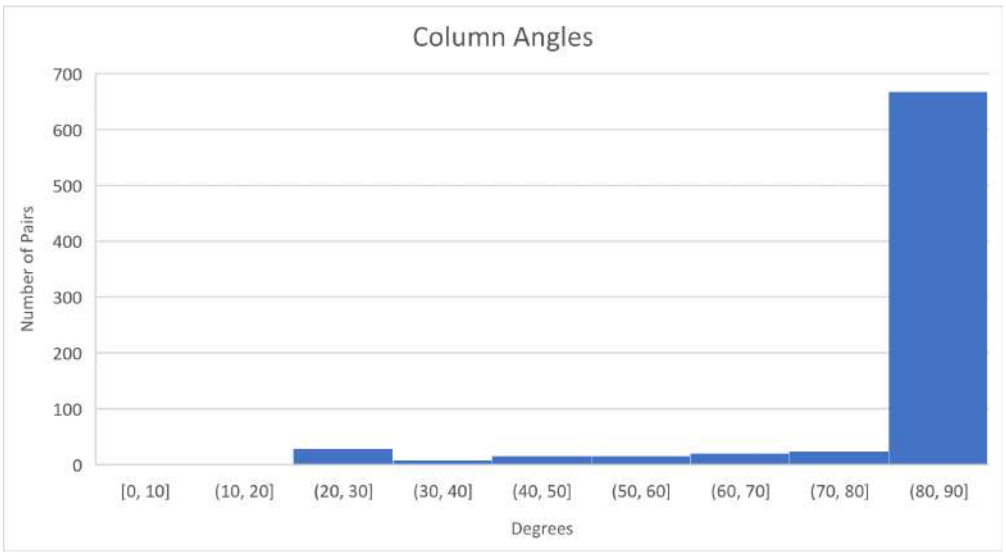


Figure 3.4: *A histogram showing the distribution of angles between pairwise geometric design variable gradients.*

Furthermore, Figure 3.5 provides insight into the distribution of singular values for each design variable, obtained through an SVD decomposition of the Jacobian matrix. This analysis aims to comprehend the sensitivity of the geometry w.r.t. each design variable.

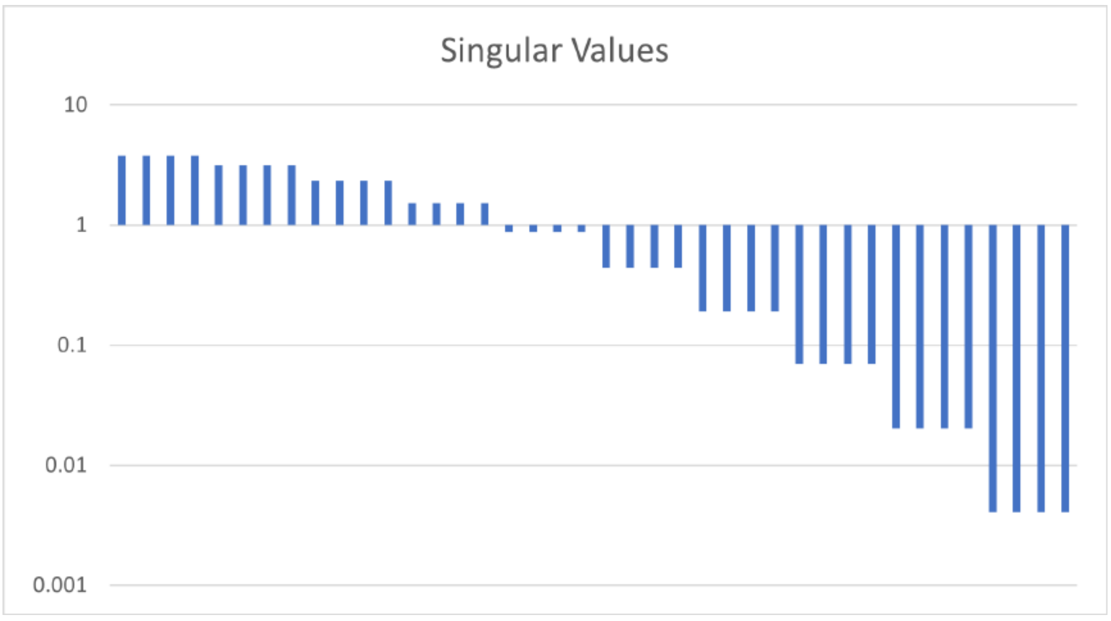


Figure 3.5: *A histogram showing the distribution of the singular values of the jacobian matrix for the NACA 0012 case with 40 Design Variables for each one of them. Y-axis uses a logarithmic scale.*

It is evident that there exists a significant variation in sensitivity across the design variables, with certain variables exhibiting a difference of more than two orders of magnitude. Such pronounced variations can pose challenges to the optimization process, potentially impeding the optimization algorithm's capacity to search the design space effectively.

Based on this data, an intriguing question emerges: How does the optimization problem fare considering these variations in angles and sensitivities between the gradients of the design variables? Moreover, could the optimization process benefit from demanding orthogonality among the gradients of the design variables and determining specific sensitivities that enable a fair comparison between them? Both questions are answered in Chapter 4 after describing the methodology for generating the new design space.

3.3 Generating New Design Variables

The primary objective is to transform the existing design variables, denoted as $\vec{b}$, into alternative variables that exhibit orthogonal gradients between them while maintaining a comparable impact on the parameterized geometry. Importantly, this transformation solely affects the parameterization, leaving the optimization problem unchanged.

To achieve this goal, the following transformation can be used,

$$\vec{\hat{b}} = \mathbf{A}\vec{b} \tag{3.9}$$

where $\vec{\hat{b}} = (\hat{b}_1, \hat{b}_2, \ldots, \hat{b}_N)$ is the new design variables' vector and $\mathbf{A}$ is the $N \times N$ transformation matrix. The matrix A is assumed to be the product of two other matrices:

$$\mathbf{A} = \mathbf{S\Phi} \tag{3.10}$$

where $\mathbf{\Phi}$ is a $N \times N$ matrix and $\mathbf{S}$ is a $N \times N$ diagonal matrix. To compute matrix $\mathbf{A}$, two steps are involved. Firstly, the appropriate $\mathbf{\Phi}$ matrix needs to be computed, which ensures orthogonal gradients for the design variables. Subsequently, this matrix is scaled by a diagonal matrix, $\mathbf{S}$, to achieve the desired magnitude for each Jacobian's column vector.

3.3.1 Defining Matrix Φ

This $N \times N$ matrix $\mathbf{\Phi}$ is chosen such that the Jacobian $\hat{\mathbf{J}}$ in the new design space, computed as:

$$\hat{\mathbf{J}} = \frac{d\vec{X}_s}{d\vec{\hat{b}}} \tag{3.11}$$

has orthogonal columns. This means that, for two distinct design variables $\hat{b}_i$ $\hat{b}_j$, the dot product of the gradients should pracically be zero:

$$\frac{d\vec{X}_s^T}{d\hat{b}_i} \cdot \frac{d\vec{X}_s}{d\hat{b}_j} = 0 \qquad \forall i \neq j. \tag{3.12}$$

To define the matrix $\mathbf{\Phi}$, is supposed that

$$\frac{d\vec{X}_s}{d\vec{\hat{b}}} = \frac{d\vec{X}_s}{d\vec{b}} \cdot \frac{d\vec{b}}{d\vec{\hat{b}}} \tag{3.13}$$

After having computed the geometric Jacobian for the original design variables, singular value decomposition (SVD) is performed. This will rewrite the Jacobian as the product of three matrices,

$$\frac{d\vec{X}_s}{d\vec{b}} = \mathbf{U\Sigma V^T} \tag{3.14}$$

In aerodynamic shape optimization applications, the number of nodes is substantially greater than the number of design variables ($m >> N$). Consequently, the last $m-N$ lines of matrix $\mathbf{\Sigma}$ are zero-lines making the last $m - N$ column of matrix $\mathbf{U}$ uselles. Therefore, after eliminating these columns, $\mathbf{U}$ becomes an $m \times N$ orthogonal unitary matrix, $\mathbf{\Sigma}$ a $N \times N$ diagonal matrix and $\mathbf{V}$ an $N \times N$ unitary orthogonal matrix.

Also, from equation 3.9,

$$\frac{d\vec{\hat{b}}}{d\vec{b}} = \mathbf{A} = \mathbf{S\Phi} \tag{3.15}$$

Rewriting equation 3.13, using equations 3.14 and 3.15 yields

$$\frac{d\vec{X}_s}{d\vec{\hat{b}}} = \mathbf{U\Sigma V^T\Phi}^{-1}\mathbf{S}^{-1} \tag{3.16}$$

by selecting,

$$\mathbf{\Phi} = \mathbf{V^T} \tag{3.17}$$

the Jacobian of the new design variables will be equal to:

$$\frac{d\vec{X}_s}{d\hat{\vec{b}}} = \mathbf{U\Sigma S}^{-1} \tag{3.18}$$

Through the linear transformation of equation 3.18, the requirement for an orthogonal Jacobian is met, given that the product of matrices $\mathbf{U}$, $\mathbf{S}$ and $\mathbf{\Sigma}$ yields an $m \times N$ non-unitary orthogonal matrix. Moreover, since matrix $\mathbf{\Phi}$ is orthonormal, the mapping of the design space corresponds to a rotation and, potentially, a reflection of the original design space.

3.3.2 Defining Matrix S

The matrix $\mathbf{S}$, known as the scaling factor, is responsible for determining the magnitude of the column vectors of the Jacobian matrix in the new design space. Notably, in Equation 3.18, the magnitude of each column vector is directly influenced by the product of its corresponding singular value and the matrix $\mathbf{S}$, given that $\mathbf{U}$ is a unitary orthogonal matrix. Considering this, the natural choice is to take advantage of the singular values Σ_i that correspond to each design variable b_i. More precisely, by defining matrix $\mathbf{S}$ as a function of the singular value matrix $\mathbf{\Sigma}$

$$\mathbf{S} = \mathbf{S}\left(\mathbf{\Sigma}\right) \tag{3.19}$$

can generate a range of scaling factors, which can be employed to easily attain the desired magnitude.

It is preferable to decrease the geometric sensitivity of the design variables that have a more significant influence and increase it for those that have a lesser impact on the 2D geometry. This approach guarantees that all design variables have comparable sensitivities. This also yields a better-conditioned Hessian from a numerical optimization perspective, with similar curvatures along diverse axes and consequently, the design variables could achieve similar convergence rates [3].

In this work, four different choices are compared and assessed. These are listed below:

1. $S_i = 1 \quad$ [no scaling]
   This means, that the new design variables correspond to only a rotation of the original ones. The sensitivity of each design variable will depend on the corresponding singular value.

2. $S_i = \Sigma_i$ [linear]
   By scaling the design variables with their singular values, it results that the new jacobian will be
$$\frac{d\vec{X}_s}{d\hat{\vec{b}}} = \mathbf{U}$$
   Because U is a unitary orthogonal matrix, all the design variables will have exactly the same geometrical infuence on the parameterized body.

3. $S_i = \sqrt{\Sigma_i}$ [sqrt]
   In the ideal scenario, a well-scaled optimization should have entries of the Hessian matrix with similar magnitudes along the diagonal, which indicates similar curvatures across design variables [3]. When scaling variables by a factor of $\mathbf{S}$
$$\hat{\vec{b}} = \mathbf{S}\vec{b},$$
   this causes the derivatives to be scaled by the same factor $\mathbf{S}$. Since the Hessian is the matrix of the second derivatives, this causes the entries of the Hessian to be scaled by $\mathbf{S}^2$. If the aim is to scale those Hessian entries by $\mathbf{\Sigma}$, then scaling the design variables by their square root would be the logical choice. Thus, this scaling factor is expected to produce a uniform distribution in Hessian.

4. $S_i = 1/\Sigma_i$ [reciprocal]
   This type of scaling is used as a counterexample of the linear one. In order to verify the above theory, this should behave in opposite ways from the other scaling factors.

As a larger scaling factor will decrease the sensitivity of a design variable, it is expected that monotonically increasing choices for $\mathbf{S}(\mathbf{\Sigma})$, such as 2 and 3, to perform well.

The presented method is a geometric approach that relies exclusively on the sensitivities of the design variables w.r.t. the embedded points. For example, the derivative of the objective function w.r.t. the (geometric) design variables is expressed as:

$$\frac{dC_d}{d\vec{b}} = \frac{dC_d}{d\vec{X}_s}\frac{d\vec{X}_s}{d\vec{b}} \tag{3.20}$$

This approach modifies only the second term which is based on the parameterization approach. The first term, which describes the physical correlation between the objective function and the geometric shape, remains unchanged. This provides an opportunity for a fair comparison among all design variables. In other words, if a design variable has a large first term, it is desirable for it to move accordingly, but if the second term is small, this movement may be nullified. Using scaling this effect is reduced or completely eliminated, and the sensitivity derivatives of the objective function w.r.t. the design variables solely depend on the physical correlation between them.

3.3.3 Verification

The scope is to verify that the Jacobian matrix of the new design space, for the NACA 0012 airfoil presented in Figure 3.3 will, indeed, obtain orthogonal columns and comparable sensitivity values.

The computation of angles θ_{ij} between column pairs of $\hat{\mathbf{J}}$ in the new design space is performed using equation 3.4, like previously. Recall that the w.r.t. alignment in the original design space results in an angle of $\theta_{ij} = 22.35^o$. The SVD method is employed to ensure the orthogonality of all gradient vectors, which is verified from Table 3.3, showing that this linear transformation does indeed yield an orthogonal Jacobian, with the maximum deviation close to machine accuracy.

	$\Delta\theta_{max}$ (deg)
$\vec{b}$	67.65
$\vec{\hat{b}}$	$2.82 \cdot 10^{-12}$

Table 3.3: *Verification of orthogonality of geometric sensitivities*

where

$$\Delta\theta_{max} = \max_{ij} |\theta_{ij} - 90^o| \qquad \text{for all } i \neq j \tag{3.21}$$

When comparing the magnitudes of each column vector, the singular value approach is employed. Figure 3.6 demonstrates that when scaling linearly, each design variable exhibits equal sensitivity, with a magnitude of one. Furthermore, scaling with the square root of the design variables yields a set of sensitivities that are more comparable to each other. Lastly, scaling with the reciprocal function reveals that the relative impact of each design variable on the surface geometry becomes even more diverse than in the original problem.

3.3.4 Reformulating the Optimization Problem

Once the mapping matrix has been computed, the optimization problem has to be rewritten in these new design variables $\vec{\hat{b}}$. This involves mapping three aspects of the optimization problem :

- Design Variables

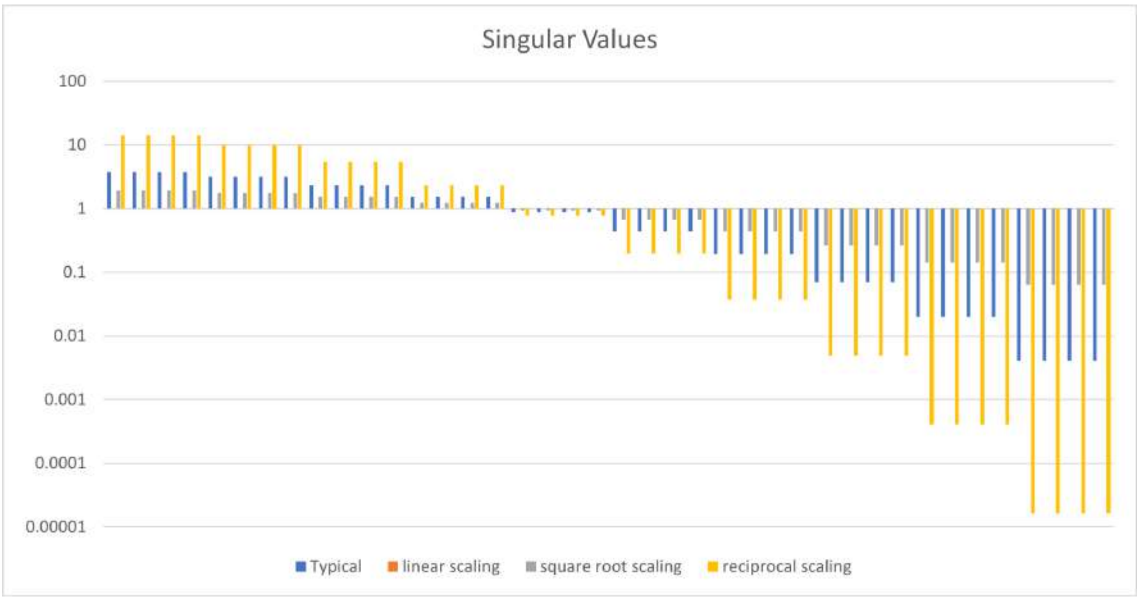


Figure 3.6: *A histogram showing the distribution of the singular values of the jacobian matrix for the NACA 0012 case with 40 Design Variables for different scaling techniques. The linear scaling is not evident as all sensitivities are equal to 1. Y-axis uses a logarithmic scale.*

- Linear and nonlinear constraints
- Design variables bounds

The procedure entails the multiplication of the design variables and Jacobians with $\mathbf{A}$ or its inverse, which is relatively simple for the first two. Nevertheless, it is imperative to adjust the optimization problem to include the design variable restrictions. These constraints primarily involve preserving the relative positions of consecutive control points to ensure that the order of the points remains unchanged, thus safeguarding the quality of the mesh. The boundaries in the original space $\vec{b}$ are provided as:

$$\vec{b}_L \leq \vec{b} \leq \vec{b}_U \tag{3.22}$$

but in the new design space, these bound constraints become linear inequality constraints

$$\vec{b}_L \leq \mathbf{A}^{-1}\vec{b} \leq \vec{b}_U \tag{3.23}$$

These constraints pose no significant challenges for gradient-based optimizers since they are linear.

3.4 Computational Framework

3.4.1 General

The adjointOptimizationFoam library, which is included in the open-source OpenFOAM software, was employed for the computational evaluation of this method. This library essentially operates as detailed in Chapter 2, solving the primal solution for incompressible aerodynamic problems and subsequently computing the adjoint sensitivities and updating the design variables.

To map the design space into the new one a new class was developed in the directory, named as *transformedDesignVariables*. More precisely, this class is responsible for defining the new design variables by utilizing the old ones. It initially defines the Jacobian Matrix based on the mesh and the parameterization data. It then calculates the transformation matrix, which isobtained from equation 3.10.

3.4.2 Define the Transformation Matrix

It is important to highlight that defining **S** and **V** directly from an SVD decomposition, as described earlier, is computationally challenging to achieve in a multi-processor solution. The Jacobian is a matrix that is divided into various processors, which presents a significant challenge for programmers. To overcome this issue without excessively increasing the programming requirements, it is recommended to perform a QR decomposition initially, followed by an SVD decomposition.

The QR decomposition is a matrix factorization technique used in linear algebra which is generally easier to parallelize than the SVD decomposition. That is because the QR decomposition involves only orthogonal and triangular matrix operations, which are well-suited for parallelization [27]. In contrast, the SVD decomposition involves more complex operations such as matrix diagonalization, which can be difficult to parallelize. Moreover, QR is a more stable and robust technique for matrix factorization than SVD. SVD can be numerically unstable [43], which means that small errors in the input data can lead to large errors in the output. This can cause problems when attempting to parallelize the algorithm, as errors can accumulate and cause inconsistent results between processors.

Therefore, after completing the QR decomposition on the Jacobian, an SVD decomposition is done at the R matrix which is a square matrix and it's the same for every processor, so it is much easier to parallelize. The last decomposition will end up giving the desired matrixes for mapping the design space. This method is mathematically explained below.

By applying a QR decomposition at the intial $(m \times N)$ Jacobian matrix, it takes the form of:

$$\frac{d\vec{X}_s}{d\vec{b}} = \mathbf{qR} \tag{3.24}$$

where $\mathbf{q}$ is an $m \times N$ matrix with orthogonal columns and $\mathbf{R}$ is an $N \times N$ upper triangular matrix. Then, matrix $\mathbf{R}$ is defined as the multiplication of three matrices, by applying SVD decomposition. Thus,

$$\mathbf{R} = \mathbf{U_R \Sigma_R V_R}^T, \tag{3.25}$$

where $\mathbf{U_R}$ and $\mathbf{V_R}$ are both $N \times N$ orthogonal unitary matrixes, whereas $\mathbf{\Sigma_R}$ is a $N \times N$ diagonal matrix containing the singular values of $\mathbf{R}$. So, replacing equation 3.24 with equation 3.25,

$$\frac{d\vec{X}_s}{d\vec{b}} = \mathbf{qU_R\Sigma_R V_R}^T = \mathbf{U\Sigma_R V_R}^T \tag{3.26}$$

$\mathbf{U}$ is $m \times N$ a unitary orthogonal matrix as it is the product of two unitary orthogonal matrixes with dimensions $m \times N$ and $N \times N$ respectively. Concluding, equation 3.4.2 is identical to the result of an SVD decomposition at the Jacobian matrix, satisfying every requirement. So,

$$\frac{d\vec{X}_s}{d\vec{b}} = \mathbf{U\Sigma V}^T \tag{3.27}$$

where

$$\mathbf{U} = \mathbf{qU_R} \tag{3.28}$$
$$\mathbf{\Sigma} = \mathbf{\Sigma_R} \tag{3.29}$$
$$\mathbf{V}^T = \mathbf{V_R}^T \tag{3.30}$$
$$\tag{3.31}$$

3.4.3 Other Functions

It is essential to note that this class, also consists of other two important functions. These functions are accountable for the conversion of the design space from its original form to a new one and vice versa. They operate by multiplying the transformation matrix or its inverse with the design variables, respectively. The primary objective is to avoid the complete transformation of the primal problem to the new design space. Consequently, the primal problem and the sensitivity derivatives of the design variables are initially calculated in the original design space. Subsequently, the transformation matrix is employed to convert the calculated sensitivity into the new design space. Following this, the update of the new design variable is carried out, which is then converted back to the original design space using the inverse of the transformation matrix. Finally, the new geometry is defined before commencing the next optimization cycle. This methodology is thoroughly elucidated in Figure 3.7.

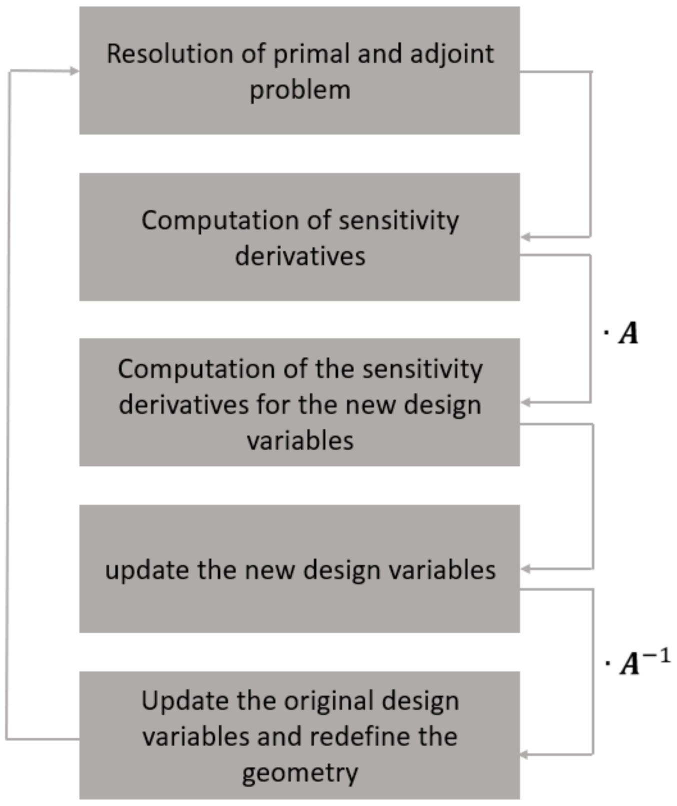

Resolution of primal and adjoint problem
Computation of sensitivity derivatives
$\cdot \boldsymbol{A}$
Computation of the sensitivity derivatives for the new design variables
update the new design variables
$\cdot \boldsymbol{A}^{-1}$
Update the original design variables and redefine the geometry

Figure 3.7: *The applied methodology for renewing the design variables in the new design space while while computing the sensitivities to the original one.*

Last, a lot of smaller functions are included. Most of them give information to the user, like the column angles before and after the transformation, the singular values, or maybe the adjoint sensitivity of each design variable, while others are responsible for checking that the algorithm is proceeding w.r.t. the mathematical methodology.

Chapter 4

Rotation And Scaling Results

This chapter evaluates the aforementioned technique by testing it on different types of applications and parameterizations to comprehend its impact on actual optimization problems.

The general characteristics of each case that will later be employed, are displayed in Table 4.1, while more details can be found when presenting each one of them.

Name	Description	Paramete-rization	DesVar	Flow	Reynold's Number
NACA 0012	Drag minimization in a Naca 0012 airfoil	NURBS or PARSEC	40/16//9	Laminar	1000
NACA4412	Drag minimization in a Naca 4412 airfoil	NURBS	44	Turbulent	$1.5 \cdot 10^6$
Draft-Airfoil	Drag minimization in a Draft airfoil	NURBS	16	Laminar	1000
S-Type Tube	Total Pressure losses minimization in a S-Type Tube	NURBS	20	Laminar	1482
Stator Cascade	Total Pressure losses minimization in a Stator airfoil	NURBS	44	Turbulent	$1 \cdot 10^6$

Table 4.1: *General characteristics for each case that will be employed in the next sections.*

Moreover, the flow is incompressible in all cases, and the quasi-Newton method is employed to update the design variables. The convergence criterion is defined as the residual of the objective function and is generally set to $1 \cdot 10^{-4}$, unless explicitly specified otherwise for specific cases.

The performance of an optimization solver comprises three fundamental parameters. First, the quality of the solution it produces; second, the computational cost required to obtain the solution; and third, the stability of the entire optimization process. These parameters will be used to define the effectiveness of each design space.

The results consist of three subsections. The first one examines solely the impact of orthogonality on the design space. The second demonstrates the effect of design variable scaling, while the final subsection draws final conclusions.

4.1 Impact Of Orthogonality

The NACA 0012 presented in Figure 3.3 is employed to examine the influence of orthogonality. The angle of attack is set to $\alpha = 1.89^o$ and the optimization problem can be succinctly described from:

$$\begin{aligned} &\min_{\vec{b}} C_D(\vec{b}) \\ 0 \leq\ &C_L \leq 0.1 \\ -0.03 \leq\ &C_m \leq 0.03 \\ &\frac{V - V_{intial}}{V_{intial}} \geq -0.15 \end{aligned} \tag{4.1}$$

The convergence of the optimization can be seen in Figure 4.1, showing the evolution of the objective function. It is essential to note that the standard technique stands for the original design space, and that in this section, only the objective function diagrams are depicted.

The optimization process for the standard design space and the rotated one appears to be indistinguishable, as evident from the results. More precisely, each iteration produces exact the same result for every aerodynamic quantity. Table 4.2 displays the optimized objectve function for each optimization, substantiating that orthogonality neither alters the optimization process nor confers any advantage.

This proposition can be mathematically demonstrated. However, prior to this, it is crucial to recall what was mentioned in section 3.4.3 regarding the method's working principle. Specifically, the primal problem and the sensitivity derivatives of the design variables are always computed in the original design space before being transferred to the new one. If the update of the original one remains unaltered, it is

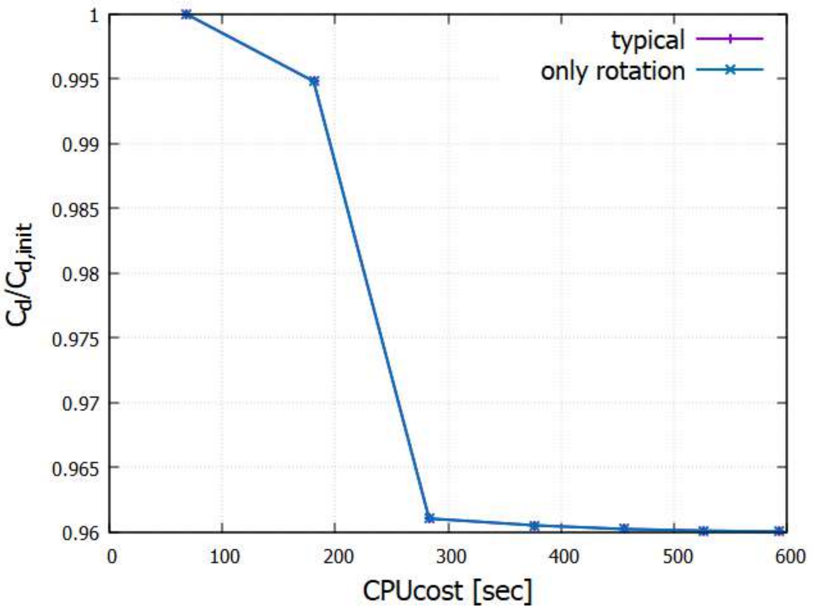


Figure 4.1: *NACA 0012: Comparison of the rotated design space with standard.*

Design Space	Total Simulation Time [sec]	Optimized $C_d/C_{d,init}$
Typical	592	0.9600
Rotated	592	0.9600

Table 4.2: *NACA 0012: Comparison of the final solution in terms of computational cost and objective function improvement for each design space.*

clear that the primal solution will continue to solve the same problem in each design space, rendering the mapping redundant.

Proof. The objective is to demonstrate that the rotation of the design space has no impact on the update of the design variables in each iteration.

The theory developed in Chapter 2 establishes that the update of design variables in the original space, without using rotation, is equivalent to:

$$\Delta\vec{b} = -\eta\frac{\delta f}{\delta\vec{b}} \tag{4.2}$$

This equation involves the step length η, search directory $\frac{\delta f}{\delta\vec{b}_j}$ and the objective function f. On the other hand, the update of the original design variables in the rotated design space is described as:

$$\Delta\vec{b} = \mathbf{\Phi}^T\Delta\vec{\hat{b}} \tag{4.3}$$

,where

$$\Delta\vec{\hat{b}} = -\eta\frac{\delta f}{\delta\vec{\hat{b}}} = -\eta\frac{\delta f}{\delta\vec{b}}\frac{\delta\vec{b}}{\delta\vec{\hat{b}}} = -\eta\frac{\delta f}{\delta\vec{b}}\mathbf{\Phi}^T \tag{4.4}$$

So, from equation 4.4, equation 4.3 is written:

$$\Delta\vec{b} = -\eta\mathbf{\Phi}^T\frac{\delta f}{\delta\vec{b}}\mathbf{\Phi}^T = -\eta\frac{\delta f}{\delta\vec{b}}\mathbf{\Phi}\mathbf{\Phi}^T \tag{4.5}$$

but $\mathbf{\Phi} = \mathbf{V}^T$ and $\mathbf{V}^T = \mathbf{V}^{-1}$, thus equation 4.5 becomes:

$$\Delta\vec{b} = -\eta\frac{\delta f}{\delta\vec{b}} \tag{4.6}$$

, which is the exact same to equation 4.2. Therefore, the rotation of the design space does not affect the update of the original design variables, and thus, it has no impact on the optimization process.

Although the rotation alone does not confer any advantage in the optimization process, it still proves beneficial in determining the scaling factors. In essence, it is significantly less complex to derive the desired factors through rotation compared to computing them in the original design space. The latter approach would necessitate additional functions to solve a more intricate system, which arises from the requirement of desired sensitivity to each design variable. This is precisely why the use of rotation will not be limited to its current application.

4.2 Impact Of Scaling

Unlike orthogonality, scaling offers several benefits in the optimization process, as will be demonstrated below. Therefore, this section provides more examples to support this claim. Initially, the conventional NACA 0012 case is employed twice, using different numbers of design variables each time to ascertain the impact of the number of design variables on scaling behavior. Subsequently, the same case is employed using a distinct kind of parameterization technique to confirm that the benefits of scaling are not subject to a specific parameterization method. Following this, the four additional cases are presented to demonstrate how scaling can have a positive influence on various problem types.

It is important to highlight that, from now on, the “only-rotation” transformation will not be displayed in the results, for the reasons exposed before.

4.2.1 The NACA 0012 Airfoil Case

The following three examples are utilized, which differ solely in airfoil parameterization. More precisely,

- NURBS Curves with 40 Design Variables [NACA 0012A Case]
- NURBS Curves with 16 Design Variables [NACA 0012B Case]
- PARSEC with 9 Design Variables [NACA 0012C Case]

The primary focus of this study is not to identify which of these methods yields a better solution. Rather, it is to establish that properly scaling the design space assists the optimization process, regardless of the chosen parameterization technique and the number of design variables.

NACA 0012A

In this parameterization, a total of 32 Control Points are equally distributed between the pressure and suction surfaces. From them 12 are fixed, thus forming a total of 40 Design variables.

Using Figure 4.2 and Table 4.3, it is apparent that scaling plays a crucial role in the optimization process. Although there are differences in the optimized solutions generated by each method, they are very similar, emphasizing the importance of comparing the computational time. Both linear and square root scaling could achieve faster convergence than the standard design space while generating superior solutions. Among these two, linear scaling holds the upper hand, as it outperforms square root scaling in both performance parameters. Conversely, reciprocal scaling performs the worst, requiring more time to converge than the standard approach and yielding the most inferior solution. Thus, this example demonstrates that linear and square root scaling enhance the optimization process, while reciprocal scaling, as discussed in section 3.3.2, has the opposite effect, diminishing the effectiveness of the optimization process.

Design Space	Total Simulation Time [sec]	Optimized $C_d/C_{d,init}$
standard	592	0.9600
linear	501	0.9592
sqrt	564	0.9594
reciprocal	825	0.9607

Table 4.3: *NACA 0012: Comparison of the final solution in terms of computational cost and objective function improvement for each scaling factor.*

NACA 0012B

This application employs 16 control points instead of 32, which are evenly distributed along both the pressure and suction surfaces. This results in a reduction in design

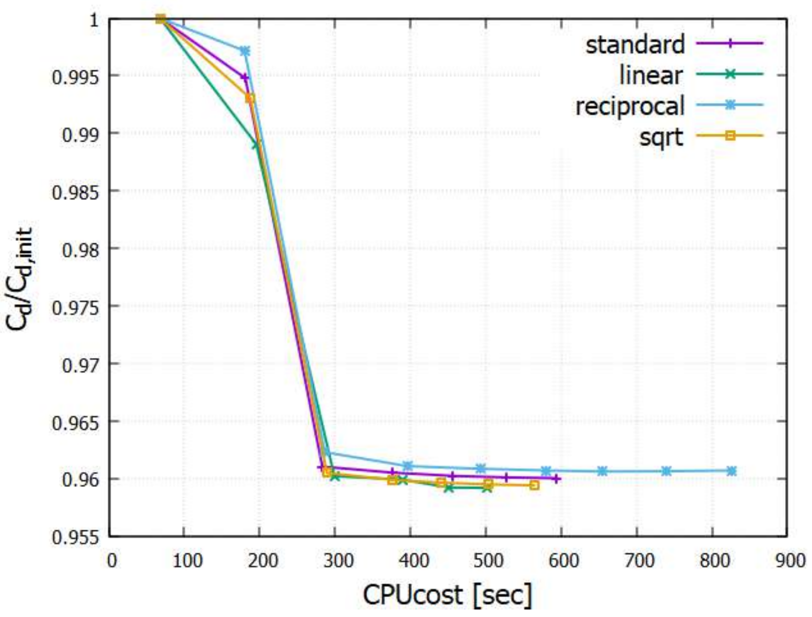


Figure 4.2: *NACA 0012A: The evolution of the objective function of the standard approach compared to the scaled design spaces. The entire optimization process can be found in A.1.*

variables which may have an impact on how scaling affects the optimization process. By decreasing the number of design variables, their sensitivities become more comparable, and the range of values is reduced. This can be observed in Figure 4.3, where the maximum difference is approximately one order of magnitude, compared to the previous example where it was almost three, as shown in Figure 3.5. This change may result in a decreased need for scaling and subsequently diminish its returns.

However, based on Figure 4.4 and Table 4.4, it is evident that scaling can provide an advantage also in this case. Notably, these advantages persist despite the decreased range of sensitivity resulting from the reduction in design variables. Both linear and square root scaling techniques achieved a better optimized solution than standard scaling while also reducing the computational cost by 20%. This improvement in efficiency could lead to significant progress in large-scale optimization problems by saving time. Conversely, the reciprocal scaling technique produced a solution of the same quality as the standard approach, but at a higher computational cost.

In summary, scaling techniques offer benefits that are not highly dependent on the range of sensitivities and can be applied in both scenarios, regardless of the number of design variables used.

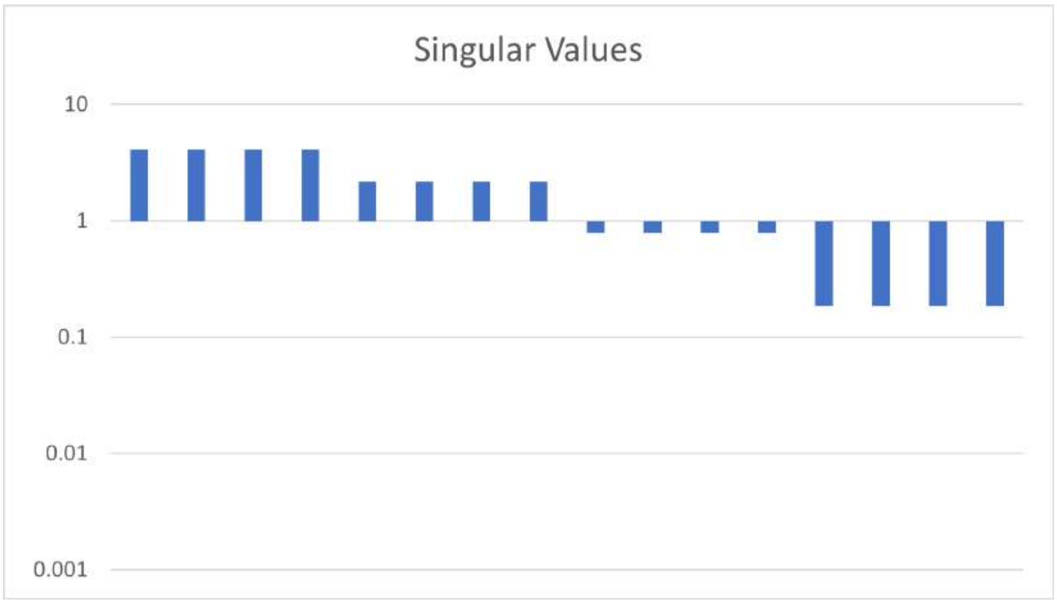


Figure 4.3: *NACA 0012B: A histogram showing the distribution of the singular values of the Jacobian matrix. Y-axis is in logarithmic scale.*

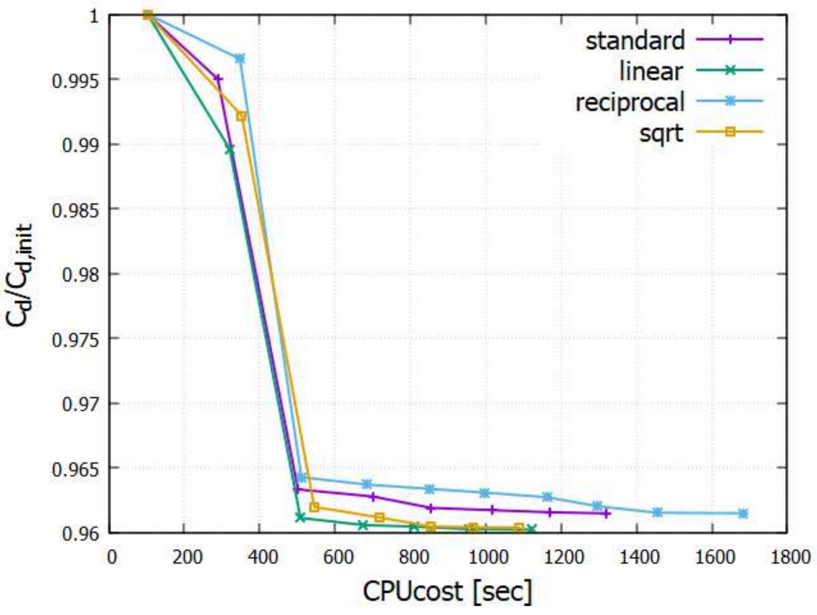


Figure 4.4: *NACA 0012B: The evolution of the objective function of the standard and scaled approaches.*

<u>NACA 0012C</u>

The total number of variables in the PARSEC parameterization are 11. Of those, two are frozen, while all the others compose the active design variable vector. The results are displayed in Figure 4.5 and Table 4.5.

Design Space	Total Simulation Time [sec]	Optimized $C_d/C_{d,init}$
standard	1320	0.9614
linear	1121	0.9602
sqrt	1087	0.9603
reciprocal	1682	0.9614

Table 4.4: *NACA 0012B: Comparison of the final solution in terms of computational cost and objective function improvement for each scaling factor.*

In general, the same conclusion can be drawn from this optimization problem. However, there is a significant difference: the reciprocal scaling technique converged first, but it was unable to produce a good result, as it became trapped into a local minimum and, thus, the fast convergence had no positive impact. In contrast, linear scaling not only converged before the standard method but also produced the best solution among the three scaling techniques.

Design Space	Total Simulation Time [sec]	Optimized $C_d/C_{d,init}$
Typical	621	0.9661
linear	378	0.9625
sqrt	427	0.9638
reciprocal	279	0.9712

Table 4.5: *NACA 0012C: Comparison of the final solution in terms of computational cost and objective function improvement for each scaling factor.*

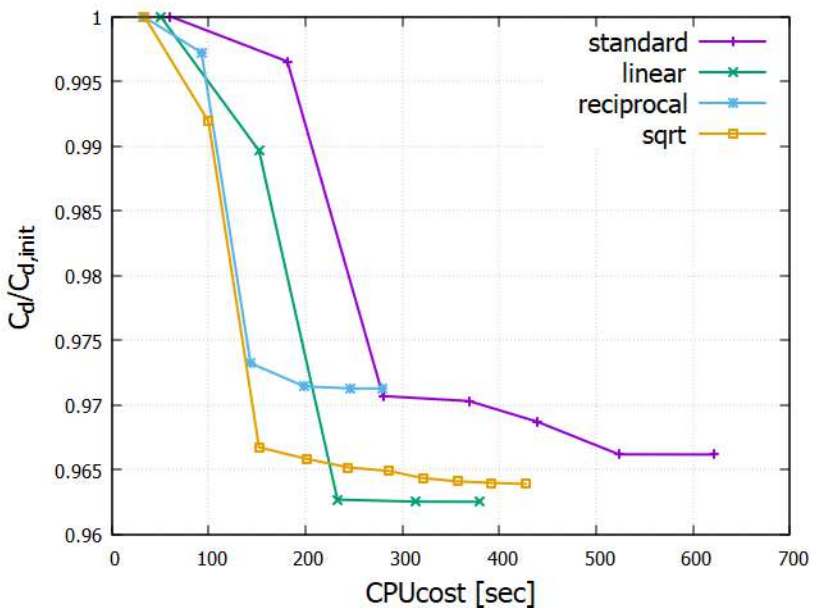


Figure 4.5: *NACA 0012C: The evolution of the objective function of the standard approach compared to the scaled design spaces.*

At this point, it is worthwhile to examine what would occur if the convergence criterion was set to 10^{-6}. While this criterion might be overly rigorous for real-world applications, it can yield intriguing findings for research purposes. Figure 4.6.a depicts the optimization process regarding the minimization of the objective function, whereas Figure 4.6.b illustrates the variation between two successive optimization iterations in terms of the total simulation time.

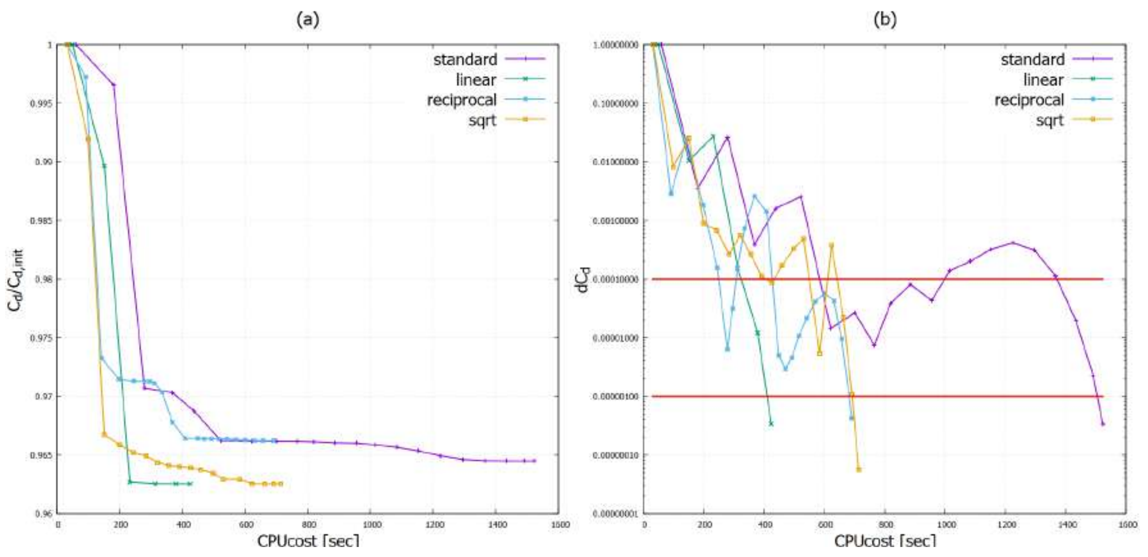


Figure 4.6: *NACA 0012C: The impact of each scaling factor compared to the standard for converge criterion set to 10^{-6}. Figure (a) shows the the evolution of the objective function C_D, whereas (b) shows the difference in the objective function between two consecutive iterations. The red lines represent the possible converge criteria.*

The results demonstrate that linear and square root scaling require fewer extra optimization cycles to meet the stricter convergence criterion compared to $s = 1/\Sigma$ and $s = 1$. Specifically, linear scaling requires only one more extra iteration to achieve convergence of the objective function to 10^{-6}. This rapid convergence rate is an indication that it was able to accumulate a sufficiently accurate approximate Hessian to be within the quadratic convergence region, likely due to a combination of orthogonal design variables and suitable scaling. Conversely, $s = 1/\Sigma$ and $s = 1$ have a much slower convergence rate, as they require many more iterations to go from a convergence of 10^{-4} to 10^{-6}, as shown in Figure 4.6.b. This indicates that they are having difficulties defining an accurate Hessian. Additionally, the evaluation of each optimized solution is identical to before, with linear producing the best solution and reciprocal producing the worst one. Therefore, it is crucial to emphasize that the positive effect of scaling can amplify as the convergence criterion becomes stricter.

<u>First Conclusion</u>

As a premature conclusion, it can be inferred that scaling has a consistent behavior regardless of the parameterization technique and the number of design variables used. Moreover, the optimized geometry for some of the design space previously presented, is observed in Figure 4.7, showing that every approach conclude in almost the same design.

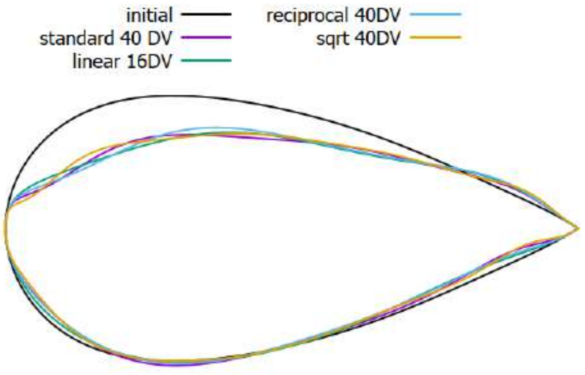


Figure 4.7: *NACA 0012: Final shape for a variety of design variables and scaling approaches. DV stands for design variables*

Although this specific example showed positive effects, it is essential to conduct further research using different optimization problems. Therefore, additional examples will be presented to demonstrate the advantages of scaling. Each application will be evaluated using only one parameterization technique. This approach will allow for a more accurate and reliable comparison of the performance of different scaling factors.

4.2.2 The NACA 4412 Airfoil Case

The application presented in this section is similar to the previous one, except that the flow is turbulent, and the angle of attack is significantly larger, $\alpha = 6.92^o$. These differences should make the optimization process more challenging and more difficult to converge. The optimization problem is defined as

$$\begin{gathered}
\min_{\vec{b}} C_D(\vec{b}) \\
1.035 \leq C_L \leq 1.235 \\
0 \leq C_m \leq 0.15 \\
\frac{V - V_{intial}}{V_{intial}} \geq -0.1
\end{gathered} \tag{4.7}$$

while the primal problem is depicted in Figure 4.8.

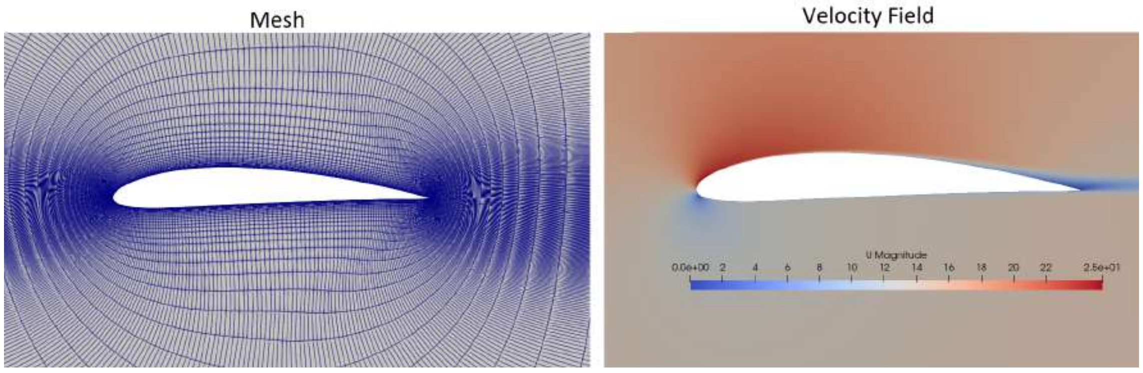


Figure 4.8: *NACA 4412: Mesh and velocity field of the initial airfoil.*

Design Space	Total Simulation Time [sec]	Optimized $C_d/C_{d,init}$
standard	14422	0.9187
linear	18733	0.9079
sqrt	17416	0.9079
reciprocal	10037	0.9247

Table 4.6: *NACA 4412: Comparison of the final solution in terms of computational cost and objective function improvement for each scaling factor.*

Figure 4.9 and Table 4.6 divulge highly intriguing findings. Firstly, the standart approach failed to converge, whereas both linear and square root scaling methods successfully converged. This suggests that scaling, apart from minimizing the simulation time, can enhance the optimization algorithm's robustness, underscoring the suitability of the new design space for gradient-based optimizations. Additionally, both square root and linear scaling techniques identified a solution of the same quality, which is considerably superior to the other two techniques. Of the two scaling methods, it appears that square root scaling maintains a slight advantage as it achieved convergence in a shorter duration with a higher converging rate than linear scaling, as depicted in Figure 4.10, meaning that for stricer convergence criteria is likely to converge first. This indicates that square root scaling was able to compute a more accurate approximate Hessian. In contrast, the scenario where $s = 1/\Sigma$, is once again the least favorable, deviating even earlier than the standard scenario, thereby compromising the optimization process.

Additionally, it is crucial to note that although the two scaling factors converged to a solution of the same quality regarding the objective function, the optimized geometries they produced have significant differences, as depicted in Figure 4.11.

4.2.3 Draft-Airfoil

This particular example is yet another airfoil simulation, similar to the previous ones. However, there is a fundamental difference that makes this example stand

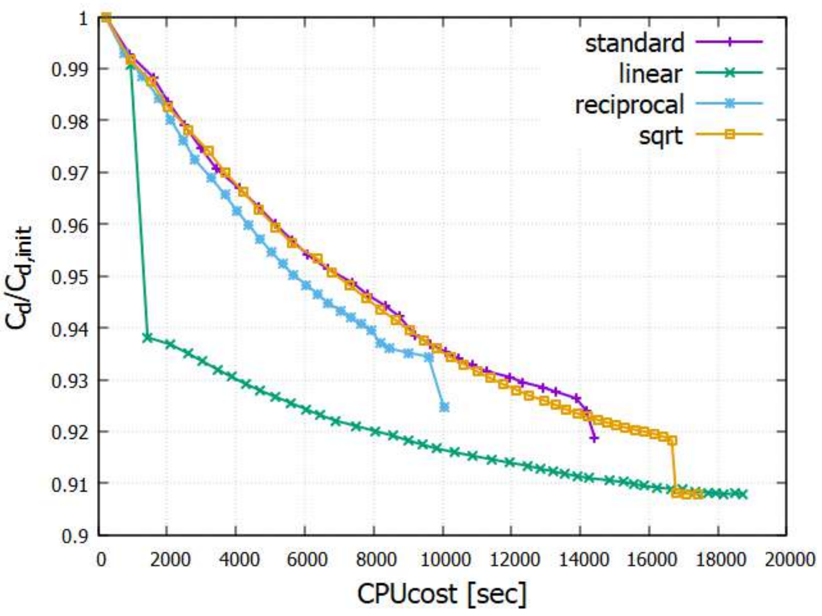


Figure 4.9: *NACA 4412: The evolution of the objective function of the standard approach compared to the scaled design spaces. The entire optimization process can be found in A.2.*

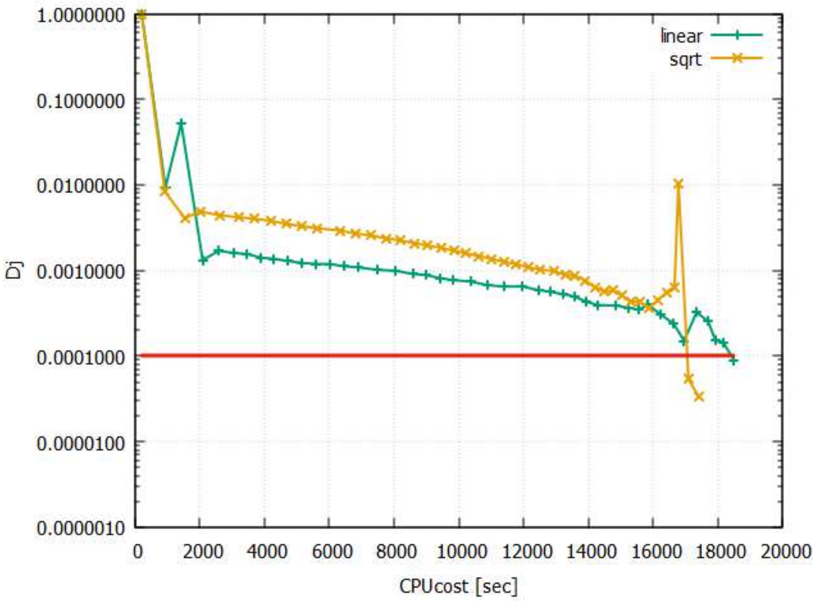


Figure 4.10: *NACA 4412: Comparison of the convergence rate between linear and square root scaling. The red line represents the convergence criterion.*

out. The airfoil has been specifically designed for testing this optimization method, and thus provides a larger space for reducing the objective function. This increased potential for optimization could lead to different results than the previous examples. The angle of attack is set to $\alpha = 1.95^o$ and the optimization problem is described as

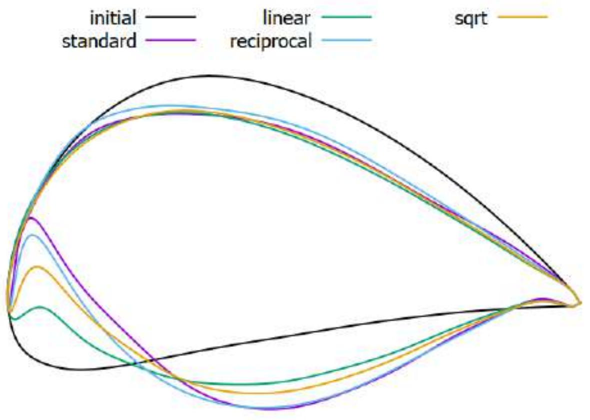


Figure 4.11: *NACA 4412: Final shape for different scaling approaches compared to the standard one. Airfoils not in (x,y) scale.*

$$\begin{gathered}\min_{\vec{b}} C_D(\vec{b}) \\ -0.01 \leq C_L \leq 0.01 \\ -0.01 \leq C_m \leq 0.01 \\ \frac{V - V_{intial}}{V_{intial}} \geq -0.15\end{gathered} \tag{4.8}$$

Design Space	Total Simulation Time [sec]	Optimized $C_d/C_{d,init}$
standard	12497	0.8849
linear	9496	0.8849
sqrt	9105	0.8850
reciprocal	-	-

Table 4.7: *Draft Airfoil: Comparison of the final solution in terms of computational cost and objective function improvement for each scaling factor.*

The results are exhibited in Figure 4.12 and Table 4.7 and generally speaking, are similar to the previous ones. Linear, square root and standard scaling factors reached a solution of the same quality but with different computational times. Specifically, linear and square root scaling factors performed better than the standard one as they required up to 25% less computational time to achieve this. Controverally, the reciprocal scaling factor failed to converge, thereby hindering the optimization process once again.

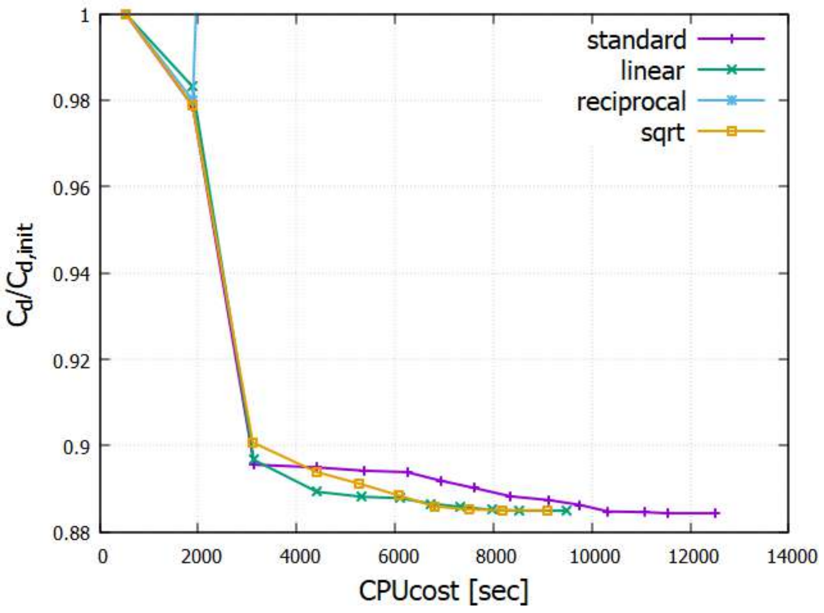


Figure 4.12: *Draft Airfoil: The evolution of the objective function of the standard approach compared to the scaled design spaces. The entire optimization process can be found in A.3.*

4.2.4 S-Type Tube

This application differs entirely from the previous examples as it involves minimizing total pressure losses in an S-Type tube. The optimization problem is defined by equation:

$$\min_{\vec{b}} P_{t,losses}(\vec{b}) \tag{4.9}$$

and the geometry is depicted in Figure 4.13.

Design Space	Total Simulation Time [sec]	Optimized $P_{t,losses}/P_{t,loss,init}$
standard	926	0.8752
linear	723	0.8751
sqrt	605	0.8750
reciprocal	533	0.8781

Table 4.8: *S-Type Tube: Comparison of the final solution in terms of computational cost and objective function improvement for each scaling factor.*

The findings of this case, as illustrated in Figure 4.14 and Table 4.8, deviate somewhat from the norm. Specifically, the reciprocal scaling approach yielded faster convergence, yet generated a suboptimal solution, revealing it was trapped into a local minimum.

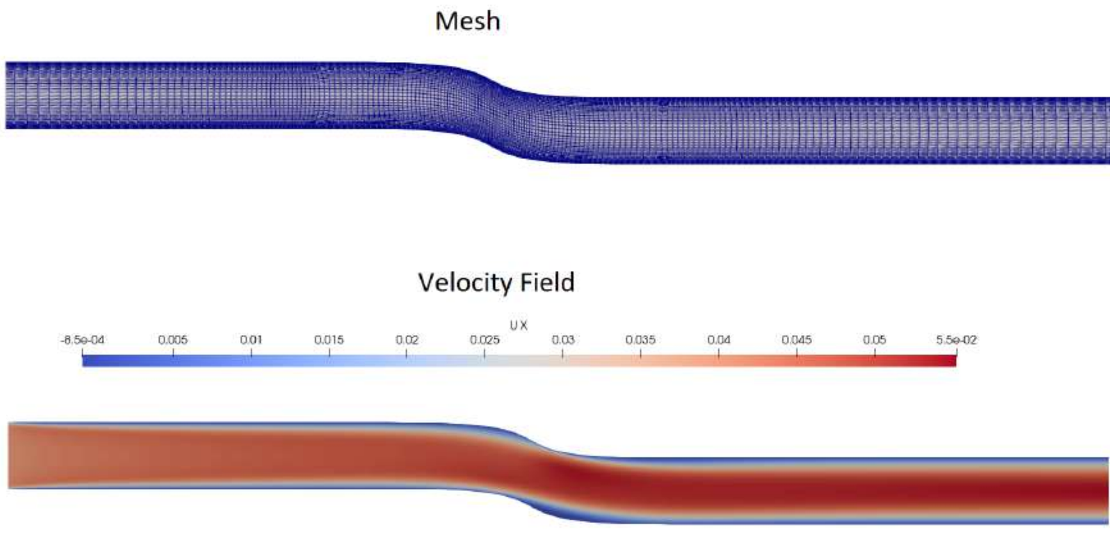


Figure 4.13: *S-Type Tube: Mesh and velocity field of the initial geometry.*

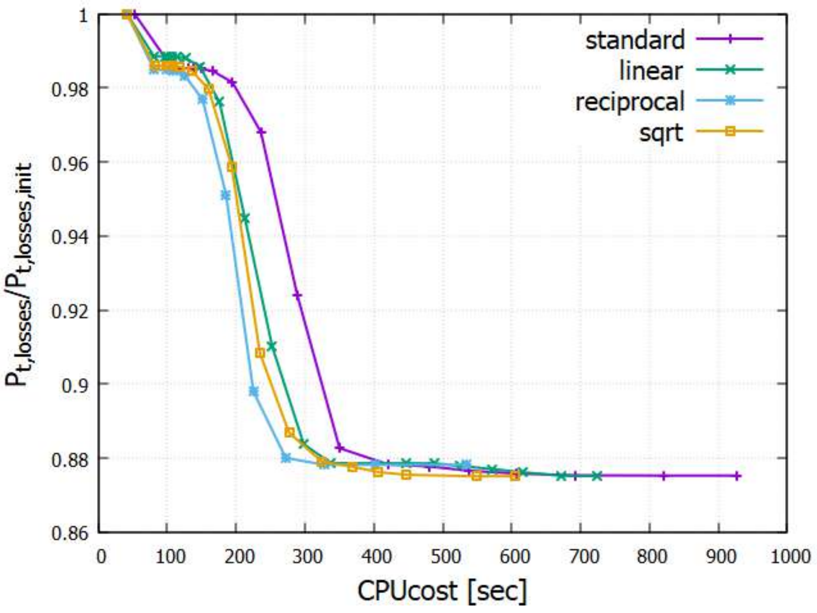


Figure 4.14: *S-Type Tube: The evolution of the objective function of standard approach compared to scaled design spaces.*

However, the difference between it and the others scaling factors is not that important, meaning that reciprocal scaling is effective for this type of problem. On the other hand, both square root and linear scaling exhibited equal improvements to the objective function, when compared to the standard method, while also significantly reducing simulation time. The square root scaling approach proved to be the optimal design space, achieving convergent results in a shorter time frame, thus reducing the simulation cost by 37%. Therefore, for the first time, reciprocal scaling can be recommended as a viable space mapping method. Nonetheless, the square root scaling method is still capable of guaranteeing the best improvement in the optimization process.

4.2.5 Stator Cascade

This case involves the minimization of total pressure losses in a 2D stator cascade. The inlet angle is $\alpha = -42^o$ and a volume constraint has been added along with the control points constraints. The primal problem and the geometry can be seen from Figure 4.15 and the optimization problem can be defined as:

$$\min_{\vec{b}} P_{t,losses}(\vec{b})$$
$$\frac{V - V_{intial}}{V_{intial}} \geq -0.1 \tag{4.10}$$

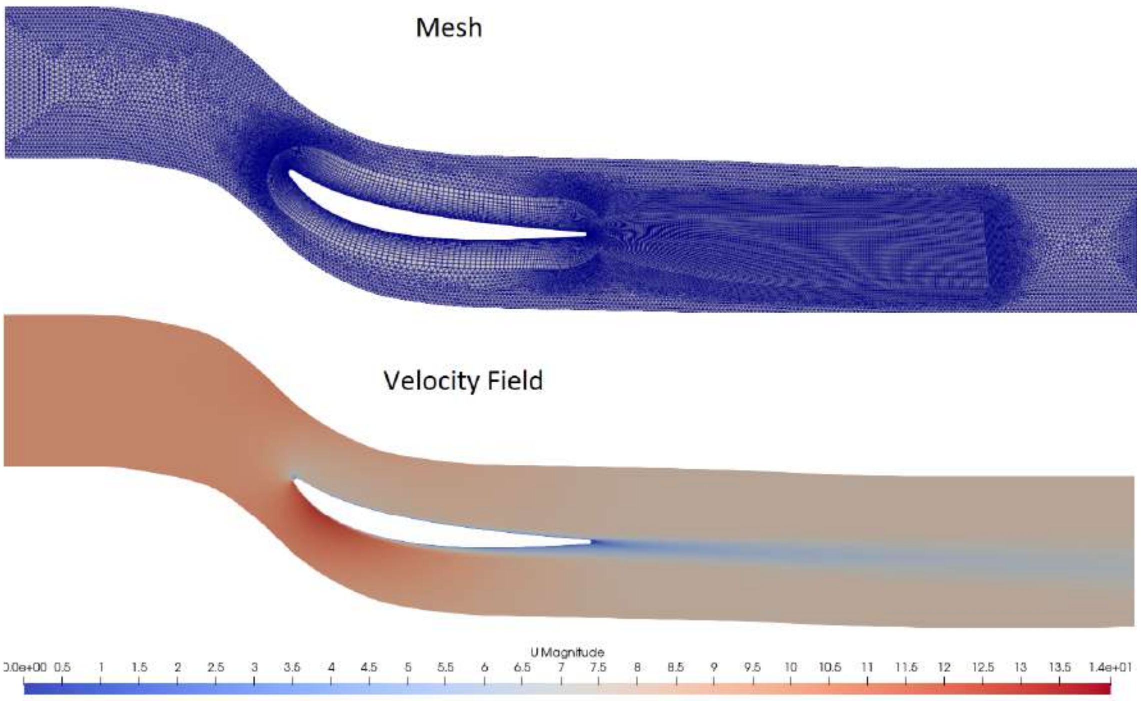


Figure 4.15: *Stator Cascade: Mesh and velocity field of the initial geometry.*

After analyzing Figure 4.16 and Table 4.9, it can be observed that the standard design space converged in the shortest time but produced a suboptimal solution. On the other hand, linear scaling generated the best solution but took the longest

Design Space	Total Simulation Time [sec]	Optimized $P_{t,losses}/P_{t,loss,init}$
standard	1948	0.9425
linear	3519	0.9347
sqrt	2101	0.9358
reciprocal	-	-

Table 4.9: *Stator Cascade: Comparison of the final solution in terms of computational cost and objective function improvement for each scaling factor.*

time to converge. Interestingly, square root scaling converged in a time similar to the standard design space, yet produced a solution that was very close to the optimal. Conversely, the reciprocal scaling failed to converge, highlighting its inability to map the design space effectively. Hence, it can be concluded that square root scaling is the most effective in defining the design space in this optimization problem.

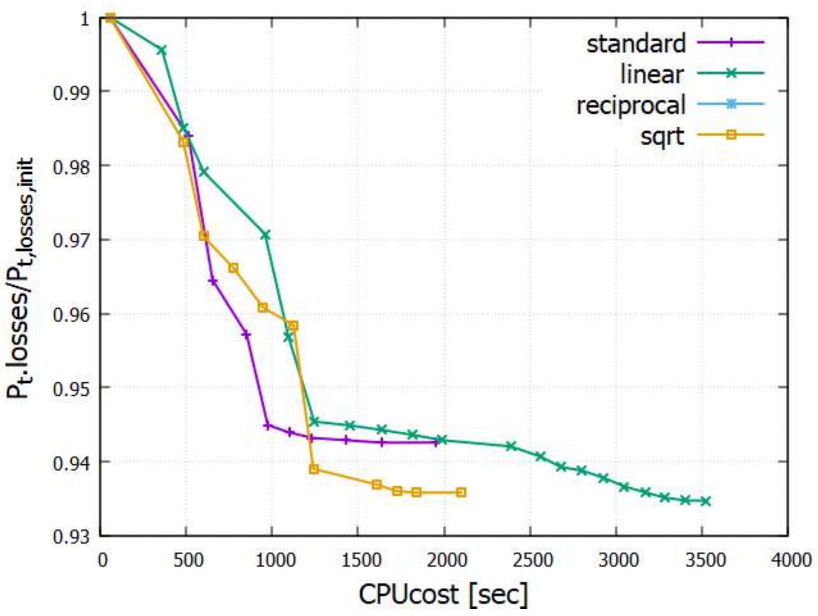


Figure 4.16: *Stator Cascade: The evolution of the objective function of standard approach compared to scaled design spaces. The entire optimization process can be bound in A.4.*

Lastly, from Figure 4.17 is evident that both the linear and square root scaling approaches converge in almost the same geometry.

4.3 Conclusions

The first conclusion that can be drawn from the analysis is that orthogonality between the columns of the Jacobian matrix has no impact on the optimization

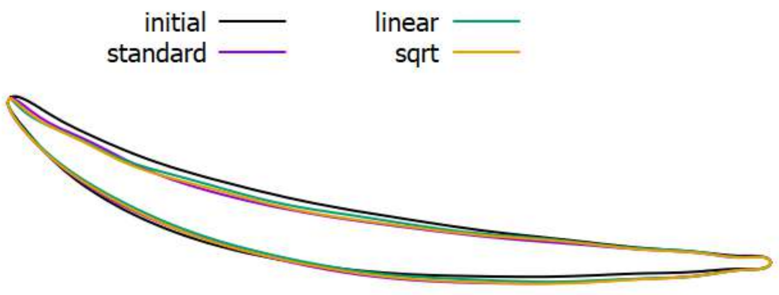


Figure 4.17: *Stator Cascade: Final shape for diiferent scaling approaches compared to the standard one. (x,y) axis not in scale.*

process. The rotated design space does not alter the optimization process and produces identical results to the standard design space. However, this rotation can provide valuable information about the suitable scaling factor from the singular value decomposition. This scaling factor plays a crucial role in determining the efficiency of the optimization process. In the subsequent paragraphs, each scaling factor is compared with the standard design space in all cases to evaluate its advantages. Finally, the most effective scaling factor is selected.

Table 4.10 presents a comparison between the linear scaled design space and the standard one for each application. In most cases, the new design space had a positive effect on the optimization algorithm. It consistently converged in a smaller amount of time than the standard one, except for one case. Moreover, it always converged in an equal or better solution. Specifically, in the NACA 0012C, and the Stator case, it was able to generate a much better solution than the standard one. It also significantly improved the convergence rate, leading to a notable reduction in the simulation time required to meet strict convergence criteria, as seen in the NACA 0012C case. Additionally, the effect of this type of scaling on the robustness of the optimization algorithm is also positive. It appears to increase the robustness, as seen in the NACA 4412 example where it achieved convergence while the standard one did not. Therefore, it is safe to conclude that, in general, linear scaling improves the optimization process as it usually outperforms the standard one in every aspect.

From Table 4.11 is apparent that square root scaling offers several advantages over the standard approach. Firstly, every example converges in less simulation time, with the exception of the Stator cascade where the difference is negligible. Secondly, it never converges to a worse solution than the standard one and in some cases significantly minimizes the objective function, such as in NACA 0012 and Stator cascade. Moreover, it converges in every simulation, improving the optimization algorithm's robustness, even in cases where the standard one fails to converge. Lastly, it significantly increases the convergence rate, reducing the simulation time

Case	Better improvement in the objective function	Less Simulation Time
NACA 0012A	linear	linear
NACA 0012B	linear	linear
NACA 0012C	linear	linear
NACA 4412	linear	linear
Draft-Airfoil	same	linear
S-Type Tube	same	linear
Stator	linear	standard

Table 4.10: *Actual comparison of the linear scaling technique with the standard one for every example presented earlier.*

required for strict convergence criteria. Thereby, it can be inferred that the mapping that square root scaling implies is better suited for gradient-based optimization than the standard one.

Case	Better Improvement in the Objective Function	Less Simulation Time
NACA 0012A	sqrt	sqrt
NACA 0012B	sqrt	sqrt
NACA 0012C	sqrt	sqrt
NACA 4412	sqrt	sqrt
Draft-Airfoil	same	sqrt
S-Type Tube	same	sqrt
Stator	sqrt	standard

Table 4.11: *Actual comparison of the square root scaling technique with the standard one for every example presented earlier.*

On the opposite side,Table 4.12 shows that reciprocal scaling never achieved the best improvement of the objective function. Instead, it converged to a sub-optimal solution in every case, indicating its inferior performance. Furthermore, it often required more time to converge than the standard approach. Additionally, the use of reciprocal scaling resulted in reduced robustness of the optimization algorithm, as it faced difficulties converging in several applications such as Draft-Airfoil, NACA 4412, and Stator case. All in all, it is clear that the use of reciprocal scaling impairs the optimization process in almost every aspect.

The selection of the best scaling factor is not so obvious. Of course, the selection should be between the square root and linear scaling as both are better than the standard one. Table 4.13 presents a comparison of these two in every case.

Assuming that NACA 0012 with the three different types of parameterizations is one case, it is evident that square root scaling converges faster in almost every case.

Case	Better Improvement in the Objective Function	Less Simulation Time
NACA 0012A	standard	standard
NACA 0012B	same	standard
NACA 0012C	standard	reciprocal
NACA 4412	-	-
Draft-Airfoil	standard	standard
S-Type Tube	standard	reciprocal
Stator	standard	standard

Table 4.12: *Actual comparison of the reciprocal scaling technique with the standard one for every example presented earlier.*

Case	Better Improvement in the Objective Function	Less Simulation Time
NACA 0012A	same	linear
NACA 0012B	same	same
NACA 0012C	linear	linear
NACA 4412	same	sqrt
Draft-Airfoil	same	same
S-Type Tube	same	sqrt
Stator	linear	sqrt

Table 4.13: *Actual comparison of the linear scaling technique with the square root one for every example presented earlier.*

Moreover, they find a solution of the same quality in almost every case, except for the Stator case and NACA 0012. In NACA 0012, linear scaling has the advantage of converging first yet finding a better optimized solution. Controversially in the Stator case, linear scaling required much more time to produce only a marginal improvement in the objective function, which was not worthwhile. Therefore, both square root and linear scaling can improve the optimization process in a similar way. However, square root scaling has the advantage of improving convergence time while producing a solution very close to the real optimal. On the other hand, the advantage of linear scaling is the possibility of finding the best solution, in term of objective function reduction, but it usually needs more time to converge. To conclude, scaling with the square root factor seems to map the design space into the most efficient one, as it practically never hurts the optimization process and usually improves it compared to the linear scaling factor.

In closing, it is crucial to remind that the general behavior of each scaling factor should remain the same independently of the surface parameterization that is used.

Chapter 5

Reducing the Design Space

5.1 Introduction

Another common technique used in aerodynamic shape optimization is reducing the number of design variables, which involves limiting the number of independent variables that define the shape of the surface. While this approach can have several advantages, it also has certain disadvantages that need to be carefully considered.

One of the significant advantages of reducing the number of design variables in aerodynamic shape optimization is that it can accelerate the optimization process. When fewer variables are involved, the optimization problem becomes less complex and easier to solve. This can lead to faster convergence and reduced computational time, which is especially important in real-time applications. Moreover, fewer variables can also improve the robustness of the optimization algorithm, as fewer degrees of freedom need to be optimized.

However, reducing the number of design variables can also have some disadvantages. One of the main disadvantages is that it can limit the design space, potentially leading to suboptimal solutions. When fewer variables are involved, the design space becomes smaller, which may prevent the optimization algorithm from finding the optimal solution. In some cases, reducing the number of variables may cause the algorithm to converge to a local minimum rather than the global minimum.

Another disadvantage of reducing the number of design variables is that it can limit the flexibility of the design. With fewer variables, there are fewer degrees of freedom to modify the shape of the surface, which may limit the potential performance gains that can be achieved through optimization. This is particularly important in complex geometries, where reducing the number of design variables may limit the optimization algorithm's ability to capture the intricate features of the geometry.

In this chapter, a different approach is developed, concerning the reduction of the design space in order to reduce the simulation time. This approach is based on eliminating the less sensitive variables as defined in a new design space that initially has the same number of design variables with the original one and is generated through the rotation that SVD decomposition implies. In that way, it is possible to reduce the design space and at the same time minimize the imported inflexibility of the geometry that occurs from this reduction.

5.2 Methodology

To make the most of the advantages out of this method while minimizing its disadvantages, it is crucial to select the most efficient subset of the design space. This subset must provide the same flexibility to the geometry as before, while significantly reducing the number of design variables.

As discussed, in detail, in Chapter 3 the SVD can define the sensitivity of each design variable of the rotated design space on the surface geometry. Taking this information into consideration, it is obvious that the design variables that will be eliminated are the ones that arise as less sensitive after the rotation. More specifically, the energy of the matrix is used in an effort to determine the mentioned subset.

The energy of a matrix is a measure of the magnitude of the matrix's singular values [43]. It represents the total amount of information contained in the matrix and is used to quantify the significance of each variable. The energy of a matrix can be decomposed into contributions from each design variable, allowing identifying which variables have the most significant impact on the optimization problem.

To select the most important variables, a cumulative energy approach can be used, where it ranks the variables by their contribution to the total energy of the matrix and select the top variables until a desired percentage of the total energy is reached. Typically, the desired energy has to belong between 95% and 99% for this traction to be harmless.

This is explained via Figure 5.1, which depicts the correlation between the overall energy of the Jacobian matrix and the number of active design variables in the NACA 0012 case. It is evidently discernible that diminishing the design variables by half results in the matrix's total energy persisting up to 99%, while utilizing 26 of the 40 design variables leads to the energy exceeding 99.9% of the overall energy. This provides the prospect of substantially simplifying the optimization process by significantly reducing the degrees of freedom, while at the same time retaining virtually the same optimization problem, as the Jacobian matrix remains practically unaffected.

The methodology for selecting the design variables is presented below. Starting, SVD is employed (equation 3.14) in the original design space. Then, the total

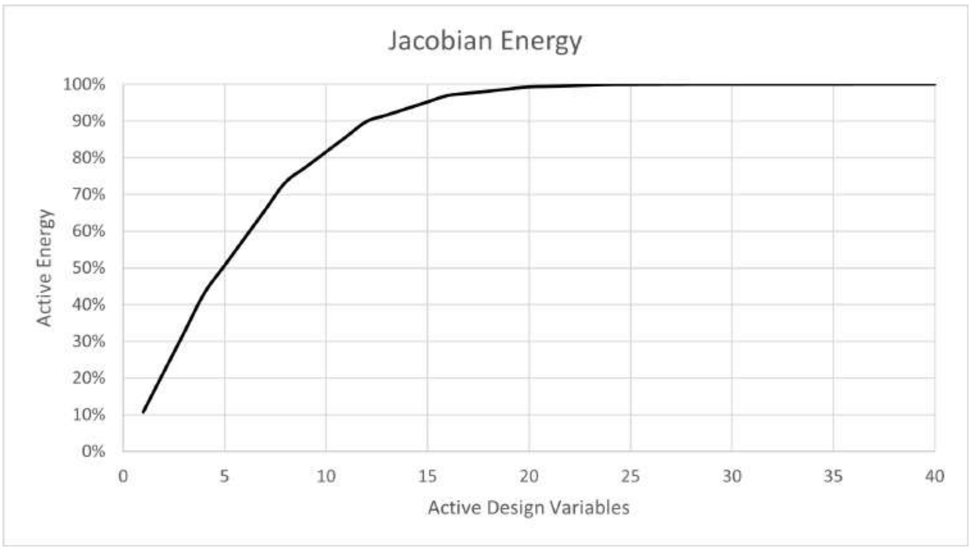


Figure 5.1: *NACA 0012A: The connection between the active energy of the jacobian matrix with the number of active design variables as a percentage of the total amount of energy.*

energy of the Jacobian must be defined. This is equal to

$$E_{total} = \sum_{i=1}^{N} \Sigma_i^2 \tag{5.1}$$

where Σ_i represent the singular values of the Jacobian matrix. Then, the energy of every design variable must be defined compared to the total energy of the matrix. Thus

$$E_i = \frac{\Sigma_i^2}{E_{total}} \tag{5.2}$$

where E_i is the energy corresponding to each singular value. At this point, the desired energy must be defined. The requirement of the new design space is to consist of the minimum number of design variables that will satisfy the inequality

$$E_r = \frac{1}{E_{total}} \sum_{j=1}^{r} {\Sigma_j}^2 \geq AE \tag{5.3}$$

r represents the new number of design variables, and $AE \in [0, 1]$ is the demanded energy of the new design space as a percentage of the original one.

To accomplish this computationally, the singular value matrix must be sorted in decreasing order. Subsequently, the columns of $\mathbf{U}$ and the lines of $\mathbf{V}^T$ must be reformulated in accordance with the reshaped $\mathbf{\Sigma}$ matrix. Afterward, commencing from the first singular value the number of the design variables, r, must be defined to satisfy the inequality 5.3. Last step is to eliminate the last $N-r$ lines and columns from $\mathbf{\Sigma}$, the last $N-r$ columns from $\mathbf{U}$, and the last $N-r$ rows from $\mathbf{V}^T$. Consequently, the dimensions of the three matrices will correspond to those specified in Table 5.1.

Matrix	Dimensions
U	$N \times r$
Σ	$r \times r$
V^T	$r \times N$

Table 5.1: *Dimensions of each component of the SVD decomposition after the reduction of the design space.*

For converting the design space, the original one is multiplied by the transformation matrix, as mentioned in Chapter 3,

$$\vec{b}_{new} = \mathbf{S}\mathbf{V}^T\vec{b}_{old} \tag{5.4}$$

because of the new dimensions of $\mathbf{\Sigma}$ and $\mathbf{V}$, the product $\vec{b}_{new}$ is a $r \times 1$ matrix, which indicates that the new design space will have r design variables instead of N. Moreover, it is crucial to acknowledge that initially $\mathbf{S} = \mathbf{I}$, unless explicitly stated otherwise. This implies that no scaling factor will be applied. Additionally, the reduced matrix $\mathbf{V}^T$ in order to reduce the design space, must first rotate it, as was analytically described in Chapter 3. Therefore, from equation 5.4 is evident that the reduced design space is a subset of the rotated one and not the original one. However, it has been proven that this rotation not impact the optimization process. Hence, the generated results will be influenced solely by the reduction in the design space.

5.3 Computational Framework

This methodology was developed in the same class as the previous one by integrating two additional functions. The first function is responsible for sorting the SVD and choosing the requisite quantity of design variables in the updated design space, whereas the other function verifies if the ordered $\mathbf{U}$, $\mathbf{\Sigma}$, and $\mathbf{V}$ matrices prior to the cutoff point result in the same design space as the unordered matrices. Consequently, the two technique operate on the same underlying principle, as outlined in section 3.4.3. The only distinction lies in the utilization of the new reduced transformation matrix instead of the transformation matrix presented in Chapter 3.

5.4 Results

A comparable methodology to the preceding chapter is employed to assess the outcomes of this technique. Specifically, four cases previously introduced will be revisited utilizing this distinct approach. The aim is to scrutinize the impact that this method has on simulation time, optimal solution, and optimization algorithm robustness. It is noteworthy that some cases differ in the number of design variables compared to their respective cases presented earlier. Additionally, it should be emphasized that different quantities of energy will be utilized in each case, contingent on the total number of design variables and the optimization process of each case.

5.4.1 The NACA 0012 Airfoil Case

NACA 0012A

The problem is described from equation 4.1. In Figure 5.2 the optimization process is presented, while Table 5.2 depicts the number of control points used in every run, the simulation cost, and the optimized solution.

Active Energy	Active Design Variables	Total Simulation Cost [sec]	Optimized $C_d/C_{d,init}$
100%	40	592	0.9600
99.9%	26	653	0.9601
99%	20	576	0.9604
95%	15	468	0.9606
90%	13	846	0.9604

Table 5.2: *NACA 0012: The impact on the active design variables and the final solution in terms of computational cost and objective function improvement for different amounts of energy.*

It is evident that adjusting the energy applied can have a significant impact on the optimization process. As shown in Table 5.2, minimizing active energy results in a inferior solution. However, the differences between optimized solutions are negligible, highlighting that the effectiveness of each energy amount depends on the overall simulation cost. Though using 99.9% of energy increases simulation time, while using 99% decreases it, the difference is insignificant and has no practical impact on the optimization process. Conversely, using 95% of energy enables the optimization to converge more quickly compared to the standard design space while still generating a solution very close to the best one. Further decreasing energy starts to hinder the optimization process, as the 90% energy requires significantly more time to converge than any other energy amount. Therefore, using energy amounts between 95% and 99% proves beneficial for the optimization process, while using energy amounts below 95% can adversely affect the process.

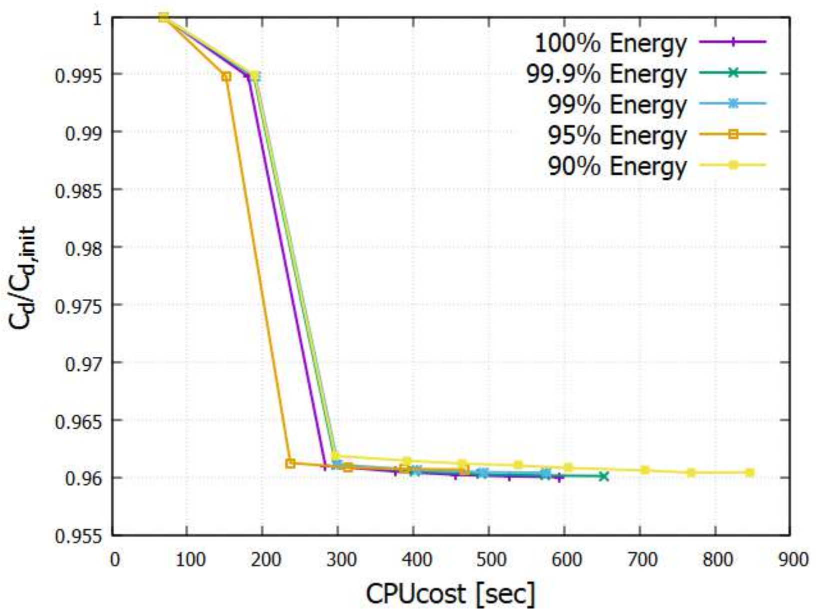


Figure 5.2: *NACA 0012: The evolution of the objective function for different amounts of energy. The entire optimization process can be found in A.5.*

<u>NACA 0012B</u>

The problem solely differs from the previous one on the number of design variables. Having fewer variables leads to more comparable sensitivity values, resulting in the deletion of fewer variables when reducing energy, as evidenced by Table 5.3. Additionally, each variable has a greater impact on the flexibility of the surface, which means that reducing variables could significantly compromise the quality of the generated solutions.

Active Energy	Active Design Variables Variables	Total Simulation Cost [sec]	Optimized $C_d/C_{d,init}$
100%	16	1320	0.9614
99.9%	14	1269	0.9609
99%	11	1382	0.9611
95%	8	1852	0.9613
90%	7	1514	0.9618
85%	6	1865	0.9622

Table 5.3: *NACA 0012B: The impact on the active design variables and the final solution in terms of computational cost and objective function improvement for different amounts of energy.*

Firstly, the optimized outcome for each energy level is analyzed. According to Table 5.3, the optimized solution is relatively insensitive to the amount of energy, with negligible deviations observed in each run. On the other hand, the total simulation time is more reliant on the Jacobian energy. Specifically, for high energy levels, such as 99.9% and 99%, the simulation cost is almost identical to the original problem. Nevertheless, it is important to note that these two approaches approximate the optimized solution much faster than the standard approach, thus simplifying the optimization problem. Conversely, if the energy level is reduced to below 95%, the optimization process performs worse than the standard approach. Therefore, in this case, the approach works solely for high energy levels, enabling the same outcome to be achieved at the same cost but with a quicker approach.

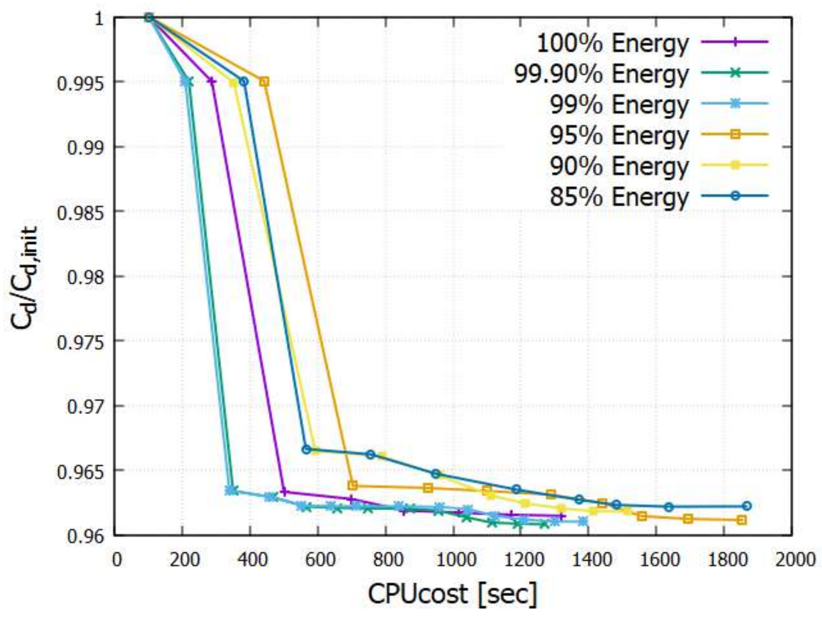


Figure 5.3: *NACA 0012B: The evolution of the objective function for different amounts of energy.*

By comparing the two NACA 0012 cases with different numbers of design variables, some important observations can be made. Firstly, it is apparent that the optimized objective function is not highly sensitive to the energy amount, regardless of the total number of design variables. Regarding the simulation time, the energy approach can benefit both cases. However, different energy amounts behave differently in the optimization process. Specifically, in the case with more control points, the effective energy amounts are from $95\% - 99\%$, while in the one with fewer control points, the effective energy amounts range from $99\% - 99.9\%$.

5.4.2 The NACA 4412 Airfoil Case

The present case is different than the one previously depicted in Chapter 4. The angle of attack is set to $\alpha = 1.5^o$ while Reynolds number remain the same. The new optimization problem at hand is expounded upon from

$$
\begin{gathered}
\min_{\vec{b}} C_D(\vec{b}) \\
0.4831 \leq C_L \leq 0.6831 \\
0 \leq C_m \leq 0.2 \\
\frac{V - V_{intial}}{V_{intial}} \geq -0.1
\end{gathered}
\tag{5.5}
$$

Active Energy	Active Design Variables	Total Simulation Cost [sec]	Optimized $C_d/C_{d,init}$
100%	44	4965	0.9088
99%	37	8205	0.9086
95%	27	3705	0.9085
90%	23	10880	0.9088

Table 5.4: *NACA 4412: The impact on the active design variables and the final solution in terms of computational cost and objective function improvement for different amounts of energy.*

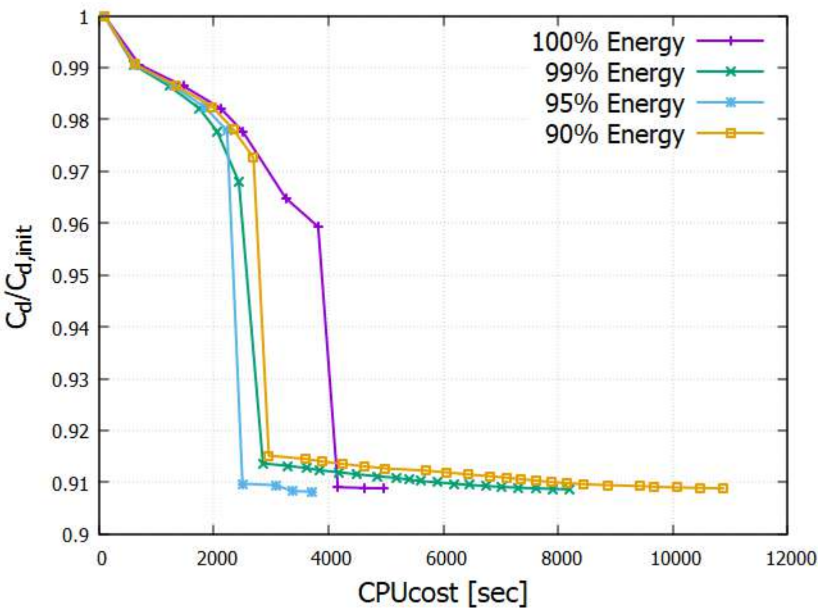


Figure 5.4: *NACA 4412: The evolution of the objective function for different amounts of energy. The entire optimization process can be found in A.6.*

From Figure 5.4 and Table 5.4 is evident that all the simulations ultimately reached a solution of the same quality but varied in the amount of time taken to converge. Notably, using 95% of the total energy proved to be the most efficient as it converged first. However, 99% and 90% were also effective, even if they required much more time to converge, as they were able to make a decent approximation of the optimized solution much faster than using the total energy amount. More precisely, these amounts had a slow converging rate due to their inability to approximate an accurate Hessian matrix. As a result, they needed a considerable amount of time to make solely minor adjustments to the objective function. On the other hand, the standard approach demonstrated a higher convergence rate, requiring only two iterations to converge after approaching the optimized solution.

Table 5.5 depicts the outcomes with the assumption that the initial approximation of the optimized solution is sufficient for every simulation. By reducing the design space, a satisfactory approximation of the optimized solution can be obtained while significantly reducing simulation time. Although this is a suboptimal solution, it is still very close to the total optimal. Thus, in some instances, it is acceptable to compromise a slight reduction in the objective function to minimize the simulation time.

Active Energy	Time for a decent approximation	approximation Value	Divergence from the total optimal	Divergence from total reduction of the objective
100%	4167	0.9088	0%	0%
99%	2861	0.9136	0.55%	5.47%
95%	2505	0.9097	0.16%	1.74%
90%	2964	0.9151	0.69%	7.56%

Table 5.5: *NACA4412: The impact on the simulation time for obtaining a decent approximation of the final solution for different amounts of energy.*

The last column of Table 5.5 represents the divergence of the reduction in the objective function using the approximate solution w.r.t. the same value using the converged solution, as described in equation

$$D[\%] = \frac{(1-\bar{C}_{D,optimal}) - (1-\bar{C}_{D,approximation})}{1-\bar{C}_{D,optimal}} \quad , \text{where} \quad \bar{C}_D = \frac{C_D}{C_{D,Init}} \tag{5.6}$$

Last, from Figure 5.5 is evident that every energy amount generated the same geometry.

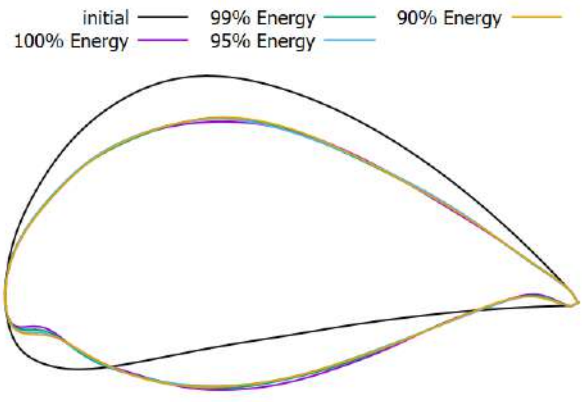


Figure 5.5: *NACA 4412: Final shape for a variety of energy amounts compared to the standard approach. (x,y) axis not in scale.*

5.4.3 S-Type Tube

This case is similar to the one expounded in Chapter 4, with minor differences in the number of design variables.Figure 5.6 and Table 5.6 depict the results.

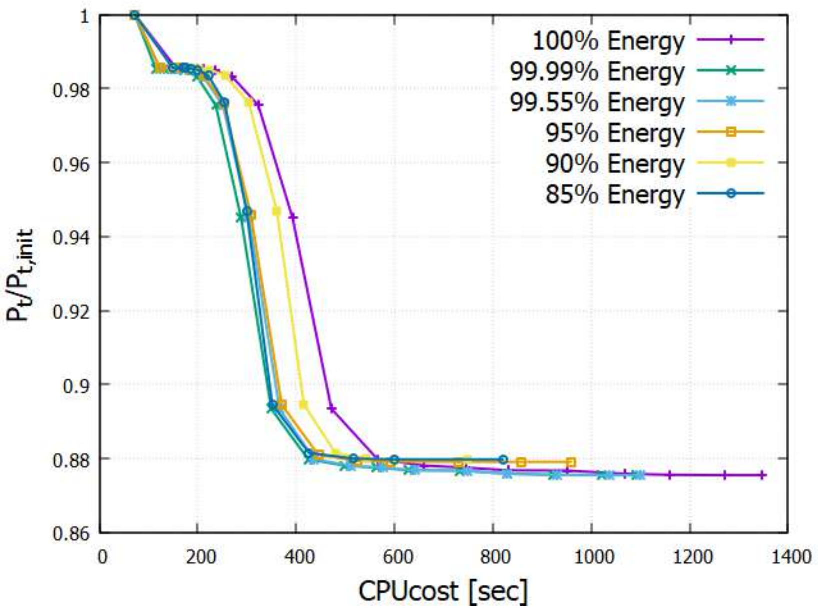


Figure 5.6: *S-Type Tube: The evolution of the objective function for different amounts of energy.*

The distinctions on the optimized solution are relatively insignificant for every amount of energy and with the utilization of over 99.55% results in exactly the same outcome as the standart approach. Furthermore, the utilization of only eleven design variables out of a total of 36 leads to a 45.3% decrease in simulation time, while

Active Energy	Active Design Variables	Total Simulation Cost [sec]	Optimized $P_{t,loss}/P_{t,loss,init}$
100%	36	1346	0.8754
99.99%	27	1091	0.8755
99.55%	20	1099	0.8754
95%	13	959	0.8790
90%	11	746	0.8797
85%	10	820	0.8797

Table 5.6: *S-Type Tube: The impact on the active design variables and the final solution in terms of computational cost and objective function improvement for different amounts of energy.*

still obtaining a solution with a divergence of just 3.3% of the entire optimization reduction. It is evident that the optimization process is negatively affected by reducing more than 10% of the energy, as the 85% approach requires a longer period to converge to the same solution as the 90% approach. Hence, in this scenario, reducing the energy by up to 10% can give a benefit to the optimization process, but reducing it by more than that would lead to diminishing returns.

5.4.4 Stator Cascade

The final case is the Stator case, which differs from the respective case presented in Chapter 4 solely in terms of the total number of design variables. Figure 5.7 and Table 5.7 show the effect that the reduction of the design space implies to the optimization process.

Active Energy	Active Design Variables	Total Simulation Cost [sec]	Optimized $P_{t,loss}/P_{t,loss,init}$
100%	32	6060	0.9554
99%	24	5790	0.9559
97%	20	6334	0.9566
95%	19	6628	0.9570

Table 5.7: *Stator Cascade: The impact on the active design variables and the final solution in terms of computational cost and objective function improvement for different amounts of energy.*

When considering the optimized solution, it is evident that a reduction in the design space results in a inferior objective function value. Nonetheless, the difference between the original and 99% energy simulations is nearly insignificant. In terms of simulation cost, the 99% energy approach holds a slight advantage over the original method. However, the disparity is negligible, leading to comparable efficacy in the optimization process. On the other hand, decreasing the energy by more than

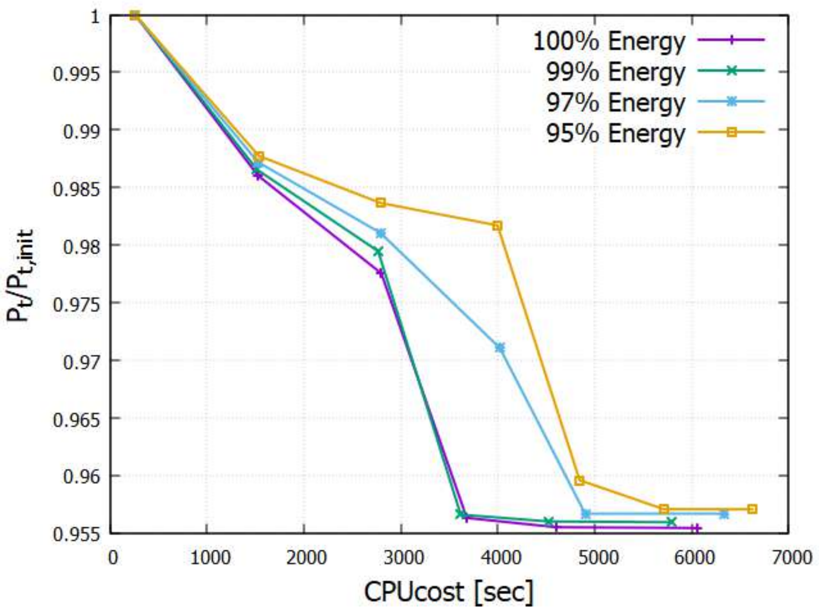


Figure 5.7: *Stator Cascade: The evolution of the objective function for different amounts of energy. The entire optimization process can be found in A.7.*

3% not only generates sub-optimal results but also prolongs the simulation time. Consequently, reducing the energy by only 1% provides an acceptable optimization process, whereas reducing it by more than 1% can impede it.

5.5 Conclusions

It is apparent that the application of the singular value energy approach for reducing the design space has a favorable impact on the optimization process. Generally, cases with a large number of design variables could benefit from reducing the energy by less than 0.1%, resulting in a substantial reduction of design variables. However, reducing more than $5 - 10\%$ of the energy begins to negatively impact the optimization process, producing a solution that could deviate significantly from the total optimal, and most important, time reduction is less effective. Moreover, in some cases, this approach has adversely affected the convergence rate while approaching the optimized solution.

More precisely, Table 5.8 shows the effect of the energy level on the optimization problem, bearing in mind that if a lower energy amount is worse than the consecutive higher, it negatively affects optimization. In summary, optimization for problems with a high number of design variables, such as the NACA 0012A, the NACA 4412, and the S-Type Tube, can be effectively enhanced while reducing the total energy by less than 5%. In contrast, cases with fewer design variables, such as the Stator and the NACA 0012B, are effective when reducing just a small part of the energy, up to 1%. However, this cannot be generalized for every case, but it can be assumed as a safe range for any optimization problem.

Case/Energy	NACA 0012	NACA 0012	NACA 4412	S-Type Tube	Stator
Number of DesVars	40	16	44	36	32
$\geq 99\%$	yes	yes	yes	yes	yes
$99\% > Energy \geq 95\%$	yes	no	yes	yes	no
$95\% > Energy \geq 90\%$	no	no	no	yes	no
$< 90\%$	no	no	no	no	no

Table 5.8: *The effect of the energy amount in each optimization case. Yes means possitive, no means negative.*

5.6 Comparison of Scaling with the Energy Approach

Upon careful analysis of the beneficial effects of scaling with the singular value matrix and the reduction in the number of design variables through the energy approach, an important query arises: what would be the outcome of combining these techniques, and ultimately, which method yields the optimal results?

Regarding the first question, it is unlikely that the combination of these techniques will produce favorable outcomes. This is because energy reduction takes place before scaling, which leads to changes in the energy of each design variable and results in a design space that fails to meet the desired energy requirements. Specifically, linear and square root scaling, which are deemed the most effective scaling factors, reduce the sensitivity of the most crucial variables, resulting in a design with less energy than required, thus generating an ineffective design space.

To test this hypothesis, the optimization process of the NACA 0012 and S-Type Tube cases was conducted using only square root scaling, which is generally the best choice. For each case, the best energy outcome was evaluated, and its performance was compared to that of the best scaling outcome. Figure 5.8 displays the optimization process of each combination, and the results are summarized in Table 5.9.

The findings reveal that combining the scaling factor with energy reduction results in inferior outcomes compared to using it with the total energy. In both scenarios, more time is required to converge, yet the solution is poorer. Therefore, the assumption that the scaling-energy combination would be ineffective is validated since scaling follows energy reduction, which changes the original energy of design variables.

Even if scaling occurred before the reduction, this combination is also unlikely to

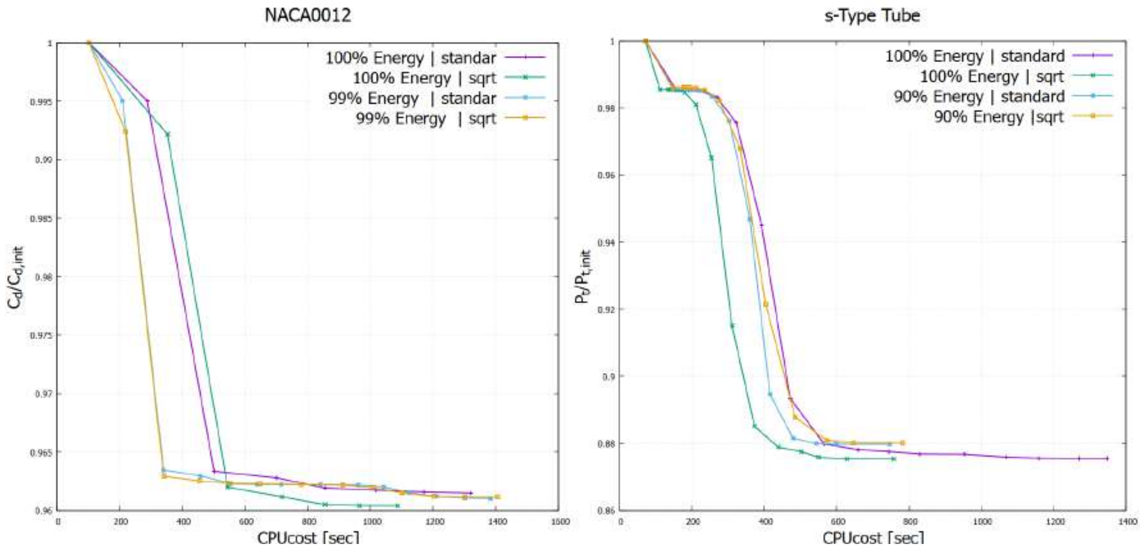


Figure 5.8: *A comparison of the optimization process for the NACA 0012 case (a) and the S-Type Tube case (b) combining different energy amounts with square root scaling*

Case	Scaling Factor	Energy amount	CPU Cost [sec]	Optimized Normalized Obj. Fun.
NACA 0012	no Scaling	100% Energy	1320	0.9614
		99% Energy	1382	0.9610
	sqrt Scaling	100% Energy	1087	0.9603
		99% Energy	1405	0.9611
S-Type Tube	no Scaling	100% Energy	1346	0.8754
		90% Energy	746	0.8797
	sqrt Scaling	100% Energy	757	0.8753
		90% Energy	783	0.8800

Table 5.9: *Results of the typical optimization space compared to the energy approach combined with the scaling technique for the NACA 0012 case and the S-Type Tube case .*

have a positive impact on the optimization process. Linear and square root scaling lead to comparable energies among the design variables, reducing the sensitivity of the most sensitive ones and increasing that of the less sensitive ones. Consequently, fewer design variables are reduced for the same amount of energy compared to the original design space, making the optimization process more complicated. To elaborate further, let us consider the S-Type Tube case for energy reduction in the original space and in the linearly scaled design space. In the former, a 10% energy reduction would reduce the design space by 26 design variables, as shown in Table 5.6. In the latter, where scaling implies unitary energy in each singular value, a 10% reduction would reduce the design space by only three design variables.

Thus, despite having the same energy, the optimization process is significantly more complex.

A viable solution to address the issue of scaling, irrespective of whether it occurs prior to or subsequent to the reduction process, is to implement the reciprocal scaling factor. Unlike linear scaling, this factor has the opposite effect on optimization. Specifically, it enhances the sensitivity of the most critical design variables, thereby enabling a more significant reduction of the design space for the same level of energy reduction. Figure 5.9 provides evidence of the benefits of this approach, indicating that combining reciprocal scaling with the energy reduciton can generate more effective outcomes compared to using the same scaling factor in the entire design space.

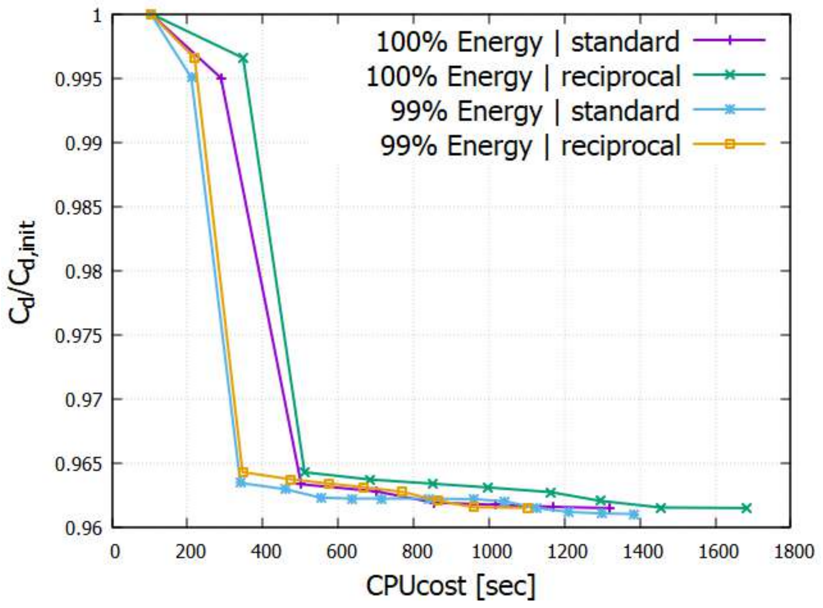


Figure 5.9: *A comparison of the optimization process for the NACA 0012 case combining different energy amounts with reciprocal scaling.*

However, it is important to note that while combining reciprocal scaling with energy reduction can generate better results than using the same scaling factor in the original design space, it is still not as effective as using square root scaling in terms of simulation time, optimized solution. Moreover, reciprocal scaling generally hurts the optimization process and deteriorates the algorithm's robustness.

In conclusion, it is recommended to use scaling and energy reduction independently, as both can generate great results for the optimization process. Between the two techniques, it appears that scaling has a slight advantage in simulation time, optimized solution, and algorithm robustness. However, this cannot be generalized for any optimization problem.

Chapter 6

Summary-Conclusion

6.1 Summary

The objective of this thesis was to assess two separate techniques applied to parameterization with the goal of improving the optimization problem. Specifically, the scope was to minimize the optimization cost while simultaneously achieving optimal or near-optimal solutions for gradient-based optimizations. This was achieved by employing the Quasi-Newton method for updating the design variables and the adjoint method for calculating sensitivity derivatives.

The first technique focused on examining the potential impact of the Jacobian-surface matrix on the optimization problem. This investigation involved two additional subsections. Firstly, the evaluation aimed to analyze the influence of the angles between the column vectors of the Jacobian. To conduct this evaluation, a new design space was created, ensuring orthogonal gradients of design variables. By comparing the results of the two design spaces, it became possible to determine whether the angles between the column vectors of the Jacobian affect the optimization process. In essence, if the new design space yielded better optimization results, it would indicate that the dependence between the gradients of design variables hinders the optimization process.

The second section of the first technique was about scaling the previously created design space. This scaling aimed to modify the magnitude of the column vectors in the Jacobian matrix, consequently influencing the geometrical sensitivity of the design variables. The purpose was to investigate whether the varying magnitudes observed among the gradients of the design variables hindered the optimization process. To accomplish this, the design space was appropriately scaled to create a more comparable range of magnitudes for the design variables. The singular value matrix was utilized for this purpose, as the singular values of the Jacobian matrix represent the normalized sensitivity of each design variable. Three distinct scaling approaches were employed. The first approach achieved equal magnitudes for every column vector, thereby employing linear scaling. The second approach reduced the variation in magnitude values while simultaneously scaling the Hessian matrix with the corresponding singular values, known as square root scaling. Conversely, the third approach served as a counter-example, intentionally increasing the variation of sensitivity between the design variables. The objective was to compare these three scaling factors with the original design space and assess whether the variation in sensitivity affected the optimization process, as well as whether a specific design space could guarantee superior optimization outcomes.

The second technique revolves around parameterization but takes a different approach. This method focused on reducing the number of design variables in order to define an easier to solve optimization problem, which could be solved faster. However, reducing the number of design variables decreases the flexibility of the parameterized surface which can lead to sub-optimal solutions. To tackle this issue, it is crucial to select a subset of the design space which can reduce the design space efficiently while minimizing the imported inflexibility of the aerodynamic surface. To achieve this a sensitivity-energy approach was employed. This approach involved computing the total energy of the Jacobian matrix as a function of its singular values. Subsequently, the design variables were sorted with respect to their sensitivities to the geometry, and only the minimum number of design variables was retained in order to satisfy a predefined number of energy. This approach, in problems with a lot of design variables, was able to eliminate almost half of the design variables, thus simplifying the optimization process, while keeping at least 99% of the Jacobian energy, resulting in a negligible loss of information.

6.2 Conclusions

To evaluate each technique, several optimization problems were tested. These problems exhibited variations not only in terms of geometry but also in the type of aerodynamic simulation (external/internal), parameterization methods, convergence criteria, and other factors. Each optimization process was evaluated based on three key parameters.

1. The optimized solution it produces
2. The computational cost needed
3. The algorithm robustness

Regarding the first technique, two distinct conclusions were drawn, one pertaining to orthogonality and the other to sensitivity. Firstly, it was demonstrated both mathematically and computationally that the angle between the columns of the

Jacobian matrix does not impact the optimization problem, yielding identical solutions. Conversely, when scaling the design space using values derived from the singular value matrix, was shown to significantly affect the optimization problems. Specifically, scaling with the original singular value matrix or with the square root of the singular values demonstrated improved performance. These scaling approaches consistently avoided suboptimal solutions and frequently achieved faster convergence times. Furthermore, they enhanced the robustness of the optimization algorithm when faced with challenging optimization tasks. The benefits were particularly prominent when strict convergence criteria were imposed, requiring high convergence ratios. In contrast, reciprocal scaling often had opposite effects on the optimization process. It deteriorated the robustness of the algorithm and hindered its ability to discover optimal or near-optimal solutions. Overall, the findings suggest that square root scaling provided the most favorable design space, as exhibited more consistent behavior than linear scaling.

On the other hand, the sensitivity-energy approach for reducing the design space also offered advantages in the optimization process. However, evaluating its overall results posed some challenges as each optimization problem utilized a different number of energies associated with the total design variables of the problem. Nevertheless, it was consistently beneficial for optimization problems to have a high amount of energy, surpassing 99%. For cases in which the energy was less than 97%, some applications, particularly those with many design variables, experienced further improvement. This resulted in a significant reduction in simulation time without deviating significantly from the optimized solution. However, for other problems, especially those with fewer design variables, the opportunity for further improvement from this additional reduction in energy was limited. Lastly, it is important to note that amounts less than 90% of the energy proved to be detrimental to every optimization problem and, therefore, are not recommended.

In the end, an attempt was made to combine both techniques. However, it was observed that the advantages of each method seemed to negate one another, resulting in a design space that was inferior to the original. In conclusion, when comparing the two techniques, the scaling approach holds an advantage over energy reduction.

6.3 Future Work

In light of the findings presented in this thesis, several promising areas for future research emerge, offering opportunities to further advance the field of optimization and address the remaining challenges.

One potential area for future research lies in exploring alternative scaling techniques for the design space in optimization problems. While the current study focused on linear scaling and square root scaling, there may be other scaling approaches that could offer unique advantages. Investigating different scaling methods, such as customized scaling based on problem characteristics, could provide valuable insights into further improving the optimization process.

Another promising avenue for future research involves expanding the sensitivity-energy approach for reducing the design space. While the current study demonstrated the benefits of this approach in various optimization problems, there is still potential for refinement and optimization. Further investigation could explore advanced techniques for selecting the subset of design variables based on energy thresholds.

Exploring these potential research directions holds the potential to deepen our understanding of optimization techniques, improve their effectiveness in various domains, and pave the way for more efficient and robust optimization algorithms in the future.

appendix A

Entire Optimization Process for Examples Peviously Displayed

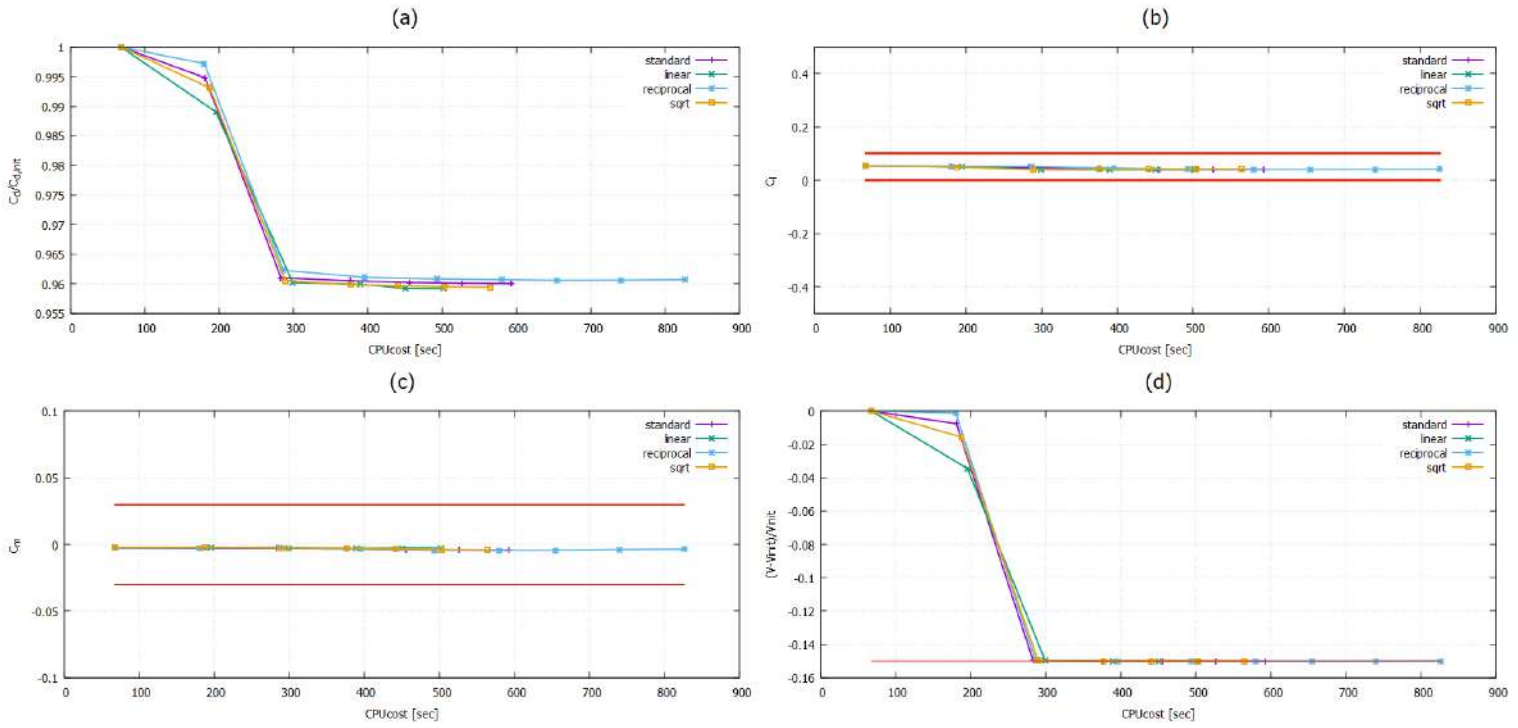


Figure A.1: *NACA 0012: The evolution of (a) normalized Drag Coefficient C_D, (b) Lift Coefficient C_L, (c) Moment Coefficient C_m and (d) total volume. The red lines represent the constraints of each aerodynamic quantity.*

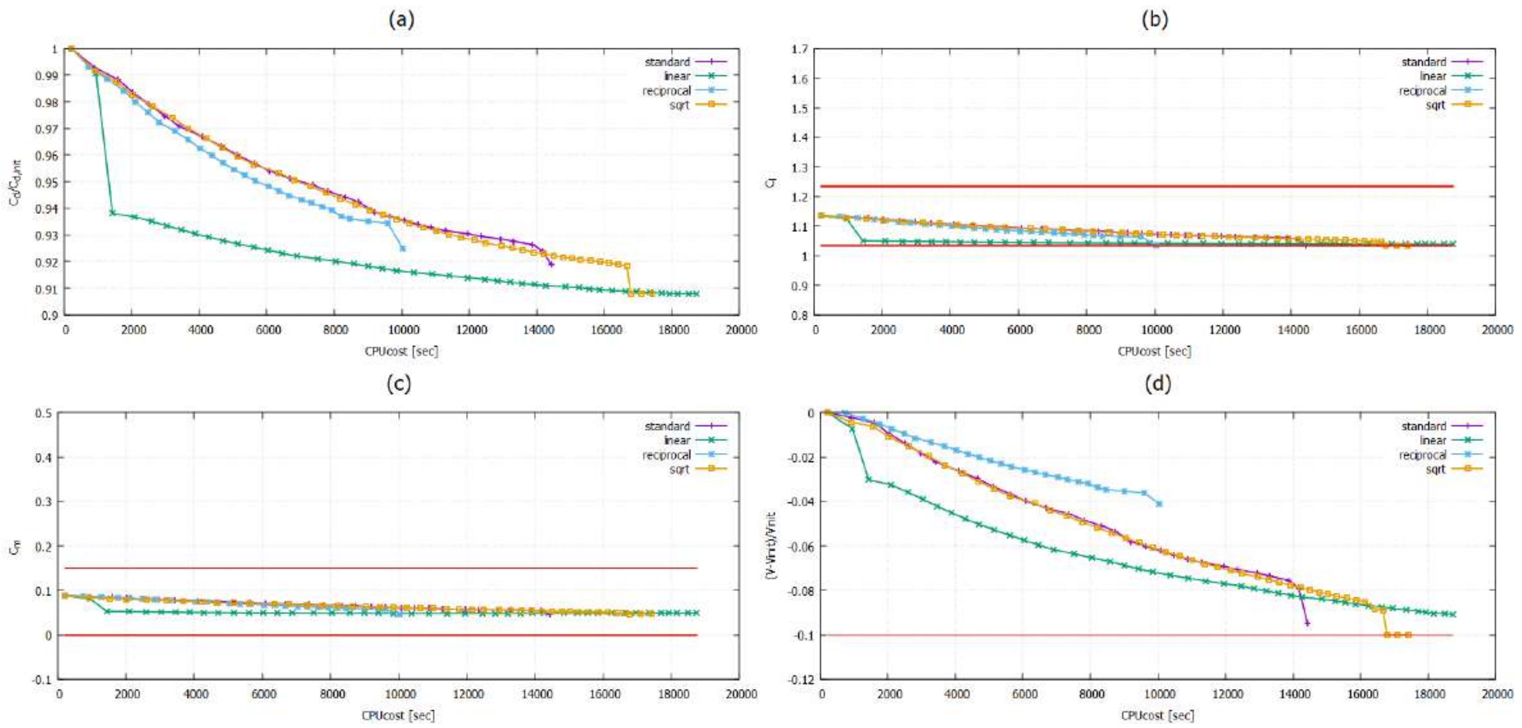


Figure A.2: *NACA 4412: The evolution of (a) normalized Drag Coefficient C_D, (b) Lift Coefficient C_L, (c) Moment Coefficient C_m and (d) total volume for different scaling approach. The red lines represent the constraints of each aerodynamic quantity.*

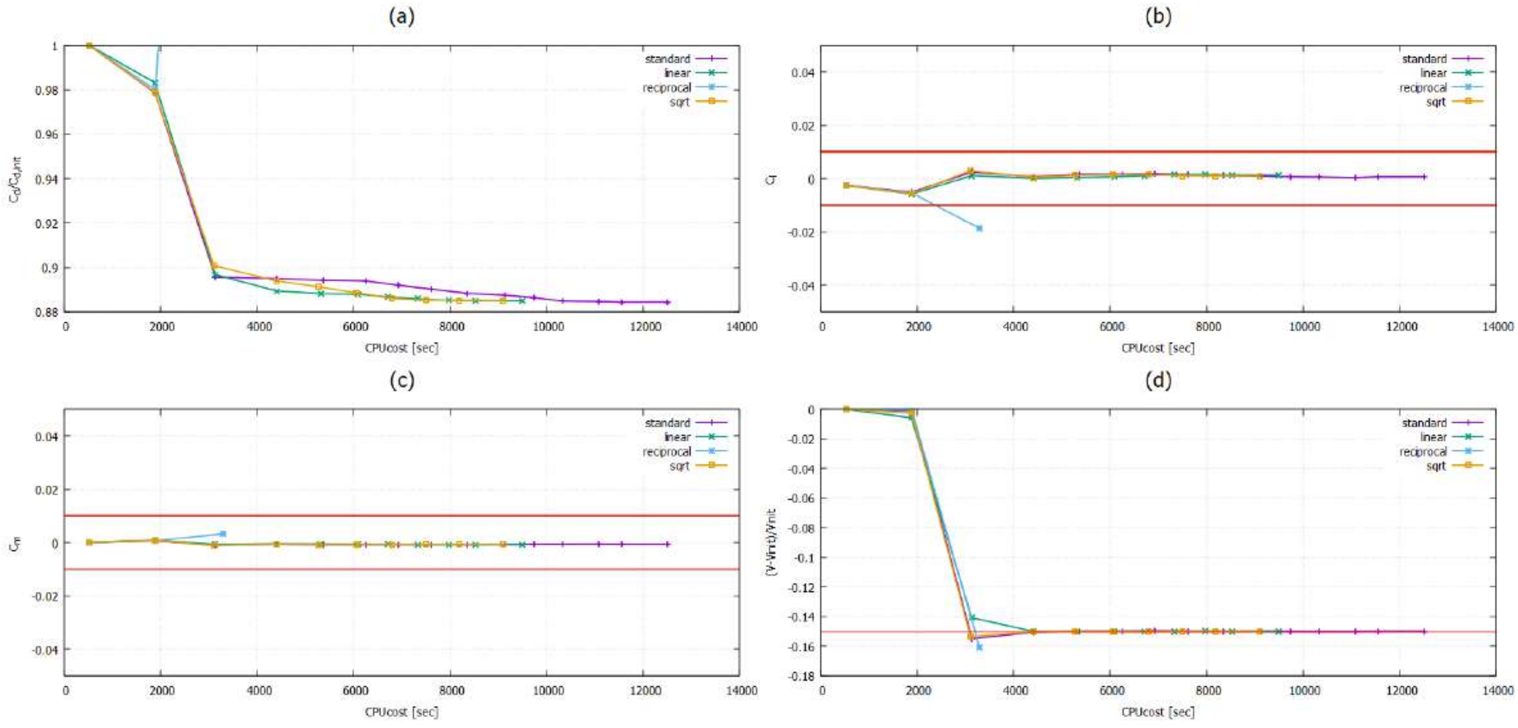


Figure A.3: *Costum Airfoil: The evolution of (a) normalized Drag Coefficient C_D, (b) Lift Coefficient C_L, (c) Moment Coefficient C_m and (d) total volume for different scaling approach. The red lines represent the constraints of each aerodynamic quantity.*

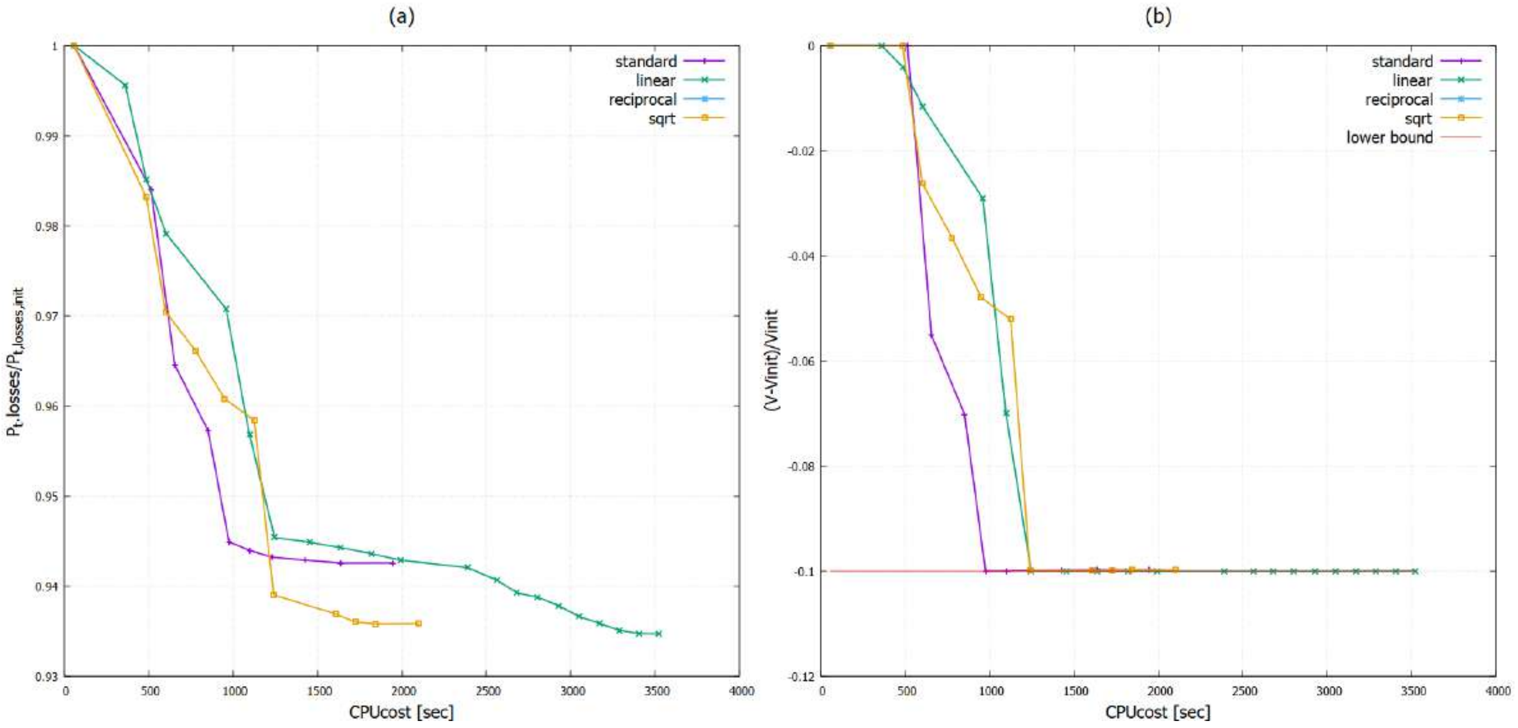


Figure A.4: *Stator Cascade: The evolution of (a) normalized total Pressure losses $P_{t,loss}$ and (d) total volume for diiferent scaling approach. The red lines represent the constraints of each aerodynamic quantity.*

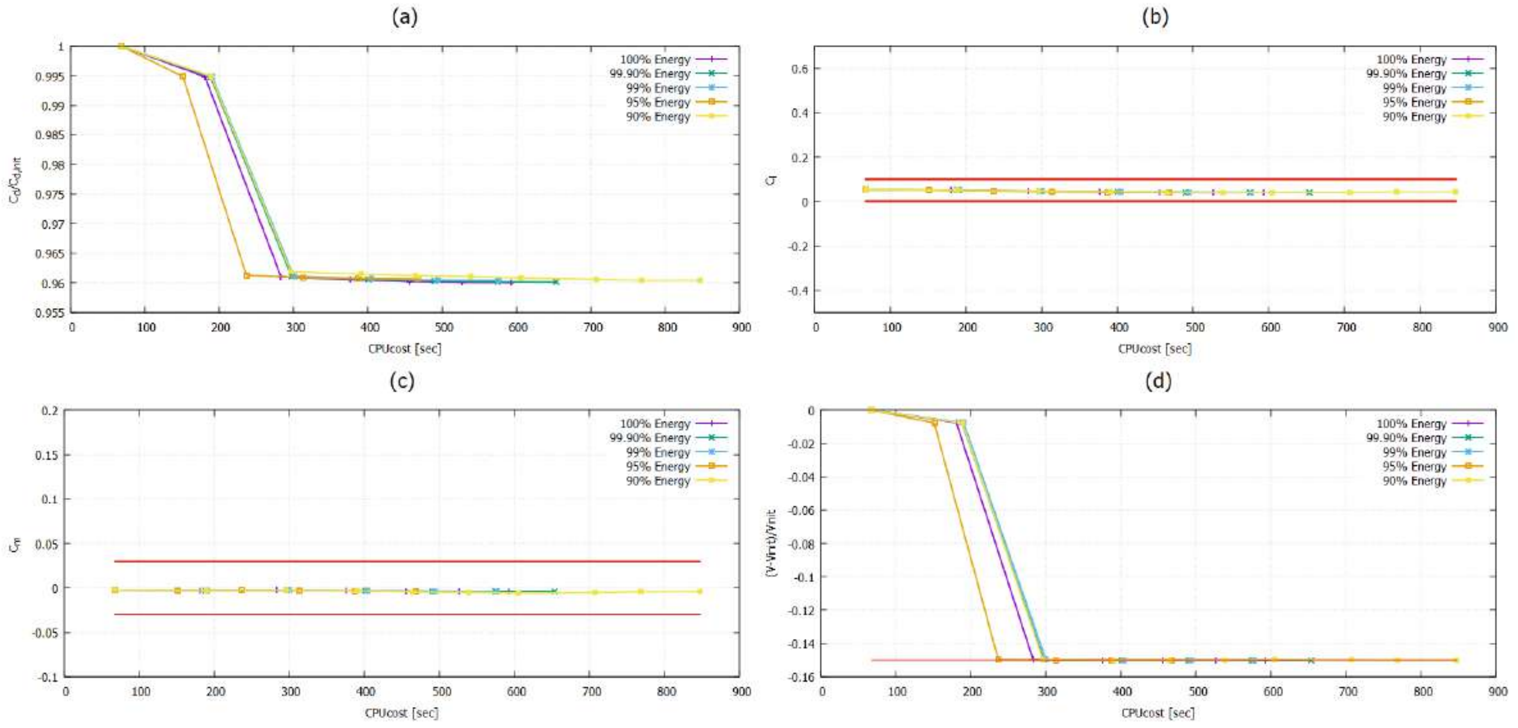


Figure A.5: *NACA 0012: The evolution of (a) normalized Drag Coefficient C_D, (b) Lift Coefficient C_L, (c) Moment Coefficient C_m and (d) total volume for different energy amounts. The red lines represent the constraints of each aerodynamic quantity.*

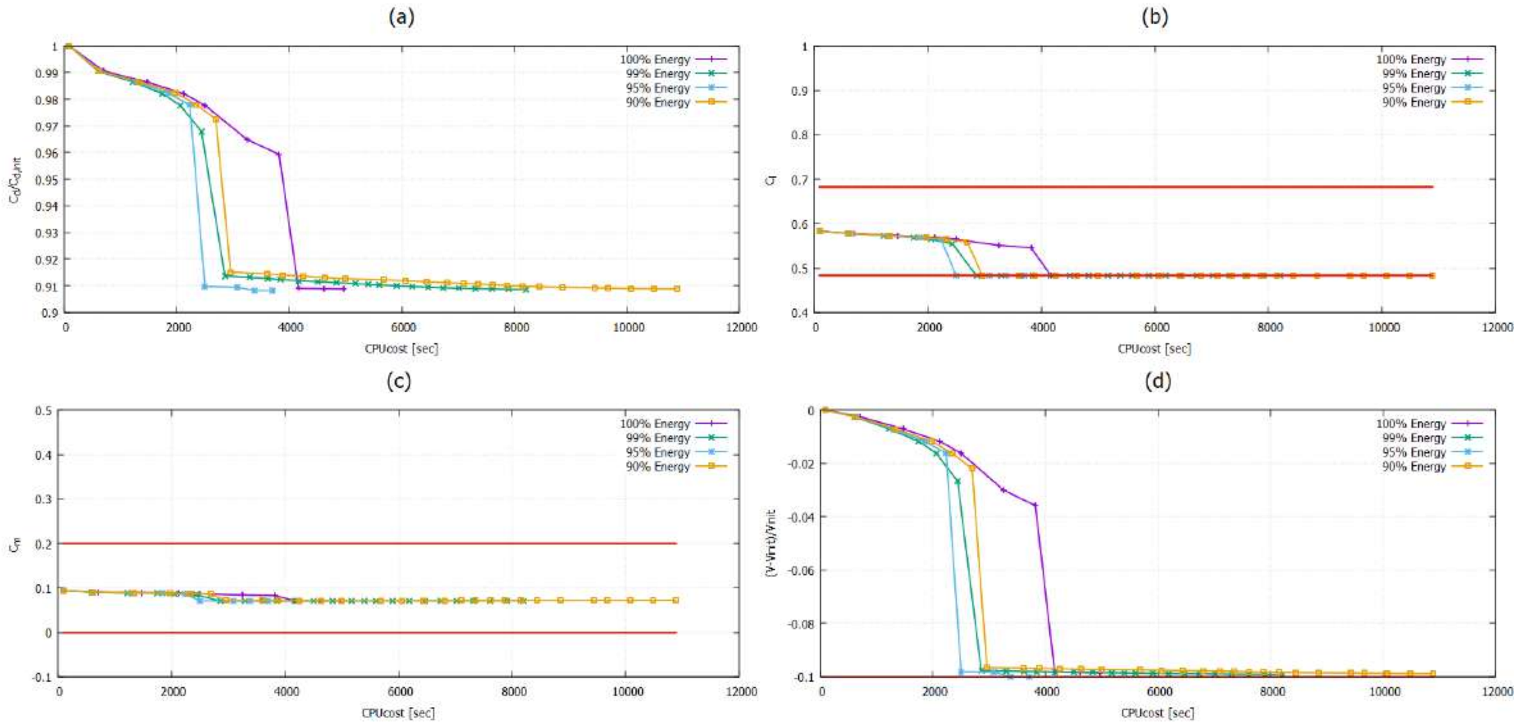


Figure A.6: *NACA 4412: The evolution of (a) normalized Drag Coefficient C_D, (b) Lift Coefficient C_L, (c) Moment Coefficient C_m and (d) total volume for different energy amounts. The red lines represent the constraints of each aerodynamic quantity.*

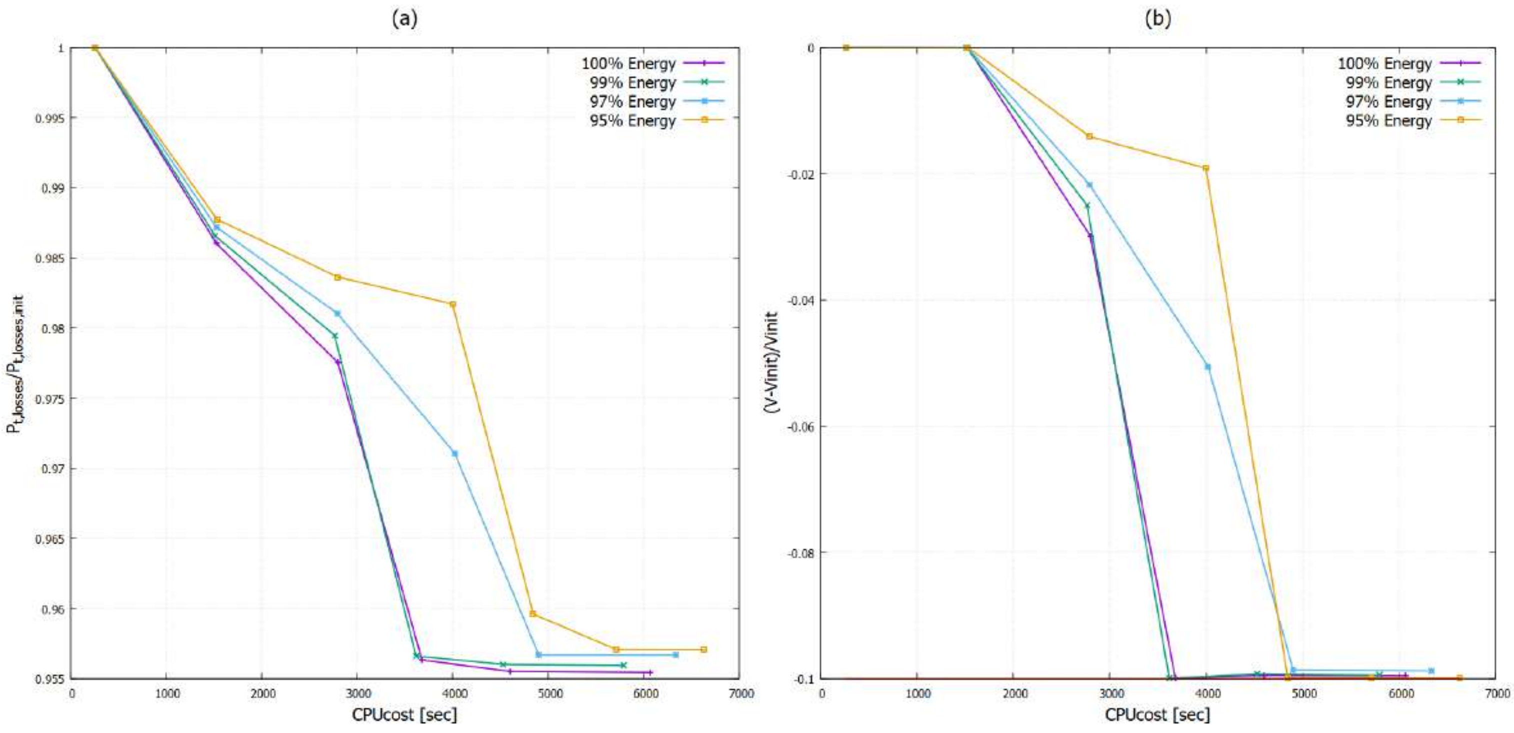


Figure A.7: *Stator Cascade: The evolution of (a) normalized Pressure Losses $P_{t,loss}$ and (b) total volume for different energy amounts. The red lines represent the constraints of each aerodynamic quantity.*

Bibliography

[1] K.C. Giannakoglou: *Optimization Methods in aerodynamics*, PcOpt/NTUA, 2006.

[2] P. Castonguay, S.K. Nadarajah: *Effect of Shape Parameterization on Aerodynamic Shape Optimization.*, Department of Mechanical Engineering, McGill University, 2007.

[3] N. Wu, C.A. Mader, and J.R.R.A. Martins: *Sensitivity-based Geometric Parameterization for Aerodynamic Shape Optimization*, AIAA journal, AIAA 2022-3931, 2022, https://doi.org/10.2514/6.2022-3931.

[4] S. Aly, M. Ogot, R. Pelz, F. Marconi and M. Siclari: *Stochastic optimization applied to CFD shape design*, 12th Computational Fluid Dynamics Conference 2012, https://doi.org/10.2514/6.1995-1647.

[5] Z. Li, X. Zheng: *Review of design optimization methods for turbomachinery aerodynamics*, Progress in Aerospace Sciences, Volume 93, August 2017, Pages 1-23.

[6] X. Wu, W. Zhang: *Robust aerodynamic shape design based on an adaptive stochastic optimization framework*, Structural and Multidisciplinary Optimization, volume 57, pages 639–651, 2018.

[7] A. Jameson: *Aerodynamic Shape Optimization Using the Adjoint Method*, Department of Aeronautics & Astronautics Stanford University, 2003.

[8] S. Aly, M. Ogot and R. Peltz: *Stochastic approach to optimal aerodynamic shape design*, Journal of Aircraft, Volume 33, Number 5, 1996.

[9] C. Sabater, S. Görtz: *Gradient-Based Aerodynamic Robust Optimization Using the Adjoint Method and Gaussian Processes*, Computational Methods in Applied Sciences, volume 55, pp 211–226, 2020.

[10] S.H. Berguin and D.N. Mavris: *Dimensionality Reduction In Aerodynamic Design Using Principal Component Analysis With Gradient Information*, 10th AIAA Multidisciplinary Design Optimization Conference 2014, 10.2514/6.2014-0112.

[11] J. Martins: *Aerodynamic design optimization: Challenges and perspectives*, Computer & fluids Computers & Fluids Volume 239, 105391, 2022.

[12] A. Jameson and S. Kim: *Reduction of the Adjoint Gradient Formula for Aerodynamic Shape Optimization Problems*, AIAA journal, Volume 41, Number 11, 2003.

[13] O. Polat, I.H. Tuncer: *Aerodynamic Shape Optimization of Wind Turbine Blades Using a Parallel Genetic Algorithm*, Procedia Engineering Volume 61, Pages 28-31, 2013.

[14] E.M. Papoutsis-Kiachagias, S. Porziani, C. Groth, M.E. Biancolini, E. Costa and K.C. Giannakoglou: *Aerodynamic Optimization of Car Shapes using the Continuous Adjoint Method and an RBF Morpher*, COMPUTMETHODS, volume 48, 2018.

[15] J. Martins: *Perspectives on aerodynamic design optimization*, AIAA Scitech 2020 Forum, https://doi.org/10.2514/6.2020-0043.

[16] Y. Yu, Z. Lyu, Z. Xu, J. Martins: *On the influence of optimization algorithm and initial design on wing aerodynamic shape optimization*, Aerospace Science and Technology, Volume 75, Pages 183-199, 2018.

[17] J. Samareh: *Aerodynamic Shape Optimization Based on Free-Form Deformation*, 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, AIAA 2004-4630, 2004.

[18] S. Jakobsson: *Mesh deformation using radial basis functions for gradient-based aerodynamic shape optimization*, Computers & Fluids, Volume 36, Issue 6,

Pages 1119-1136, 2007.

[19] Z. Lyu, Z. Xu and J. Martins: *Benchmarking Optimization Algorithms for Wing Aerodynamic Design Optimization*, International Conference on Computational Fluid Dynamics, ICCFD8-2014-0203, 2014.

[20] J. E.V. Peter , R.P. Dwight: *Numerical sensitivity analysis for aerodynamic optimization: A survey of approaches*, Computers & Fluids Volume 39, Issue 3, Pages 373-391, 2010.

[21] David W., Marian Nemec: *A comparative evaluation of genetic and gradient-based algorithms applied to aerodynamic optimization*, European Journal of Computaional Mechanics, Volume 17, 2008 - Issue 1-2.

[22] W. Zhou: *A modified BFGS type quasi-Newton method with line search for symmetric nonlinear equations problems*, Journal of Computational and Applied Mathematics, Volume 367, 112454, 2020.

[23] J. Pena: *Quasi-Newton Methods*, Convex Optimization 10-725/36-725, 2010.

[24] A. Jameson: *Gradient Based Optimization Methods*, Department of Aeronautics and Astronautics Stanford University, Stanford, CA 94305-4035.

[25] I. Ahmadianfar , O. Bozorg-Haddad: *Gradient-based optimizer: A new metaheuristic optimization algorithm*, Information Sciences, Volume 540, Pages 131-159, 2020.

[26] G. Carrier, D. Destarac, A. Dumont, M. Meheut: *Gradient-Based Aerodynamic Optimization with the elsA Software*, 52nd Aerospace Sciences Meeting - AIAA SciTech 2014, 10.2514/6.2014-0568.

[27] J.Nocedal and J. Wright: *Numerical Optimization*, Second Edition, Springer, 2006.

[28] J. A. Samareh: *A Survey Of Shape Paarametrization Techniques for High-Fidelity Multidisciplinary Shape Optimization*, AIAA Journal, Volume 39, Number 5, 2001.

[29] A. A. Montano, C. A. Coello Coello: *Evolutionary Algorithms Applied to Multi-Objective Aerodynamic Shape Optimization January*, Studies in Computational Intelligence, volume 356, pp 211–240.

[30] A. Bueno-Orovio, C. Castro, F. Palacios, E. Zuazua: *Continuous Adjoint Approach for the SpalartAllmaras Model in Aerodynamic Optimization* AIAA Journal, Volume 50, 2012, https://doi.org/10.2514/1.J051307.

[31] E. Arian, S. Ta'asan: *Analysis of the Hessian for aerodynamic optimization: inviscid flow*, computer & fluids, Volume 28, Issue 7, Pages 853-877, 1999.

[32] S. Tsaggaris: *Fluid Dynamics*, 2nd Edition, Tsiortas Publushing House, 2016.

[33] John D. Anderson Jr.: *Computatioanl Fluid Dynamics*, 1995

[34] Key Takeaways: *The Reynolds-Averaged Navier-Stokes (RANS) Equations and Models*

[35] K.C. Giannakoglou: *Viscous flows in Turbomachines*, PcOpt/NTUA, 2004.

[36] G. Mpergeles: *Computational fluid dynamics*, 2nd Edition, Simeon Publushing House, 2012.

[37] Steven R. Allmaras, Forrester T. Johnson∗ and Philippe R. Spalart: *Modications and Clarications for the Implementation of the Spalart-Allmaras Turbulence Model*, h International Conference on Computational Fluid Dynamics 2012.

[38] Kyriakos C. Giannakoglou: *Numerical Analysis for Engineers*, PcOpt/NTUA, 3rd Edition, 2003.

[39] https://en.wikipedia.org/wiki/Non-uniformRationalB-spline.

[40] Anargyros P. Felouris: *Linear Algebra and Analytical Geometry*, 2nd Edition, Tsiortas Publushing House, 2017.

[41] K.C. Giannakoglou, Dr. E.M. Papoutsis-Kiachagias, K.T. Gkaragkounis: *adjoint Optimization Foam manual*, 2020.

[42] R.P. Dwight and J. Brezillon: *Effect of Approximations of the Discrete Adjoint on Gradient-Based Optimization*, AIAA journal, Volume 44, Number 12, 2006.

[43] Golub, G. H. and Van Loan, C. F.: *Matrix Computations*, Johns Hopkins University Press.

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

Αιτιοκρατική Βελτιστοποίηση Μορφής στην Αεροδυναμική Υποβοηθούμενη από Μετασχηματισμό της Παραμετροποίησης –Μελέτες και Αξιολόγηση

Εκτενής Περίληψη Διπλωματικής Εργασίας

Σωτήριος Δαμιανός

Επιβλέποντες: Κυριάκος Χ. Γιαννάκογλου, Καθηγητής ΕΜΠ

Αθήνα, 2023

Εισαγωγή

Αυτή η διπλωματική εργασία κατατάσσεται στο πεδίο της αεροδυναμικής βελτιστοποίησης σχήματος η οποία εμπλέκει τη συνεχή συζυγή μέθοδο για υπολογισμό των παραγώγων ευαισθησίας της συνάρτησης στόχου. Τα σημερινά προβλήματα αιτιοκρατικής βελτιστοποίησης είναι μεγάλης πολυπλοκότητας και αποτελούνται από πλήθος μεταβλητών σχεδιασμού [10]. Έτσι η δυσκολία σύγκλισης και το υπολογιστικό κόστος μιας τέτοιας μεθόδου έχουν αυξηθεί σημαντικά, με αποτέλεσμα να βρίσκονται στο προσκήνιο της έρευνας [11]. Ένας από τους κύριους παράγοντες που επιδρούν στην πορεία βελτιστοποίησης είναι ο τρόπος παραμετροποίησης της γεωμετρίας, ο οποίος λόγω των πολλών μεταβλητών σχεδιασμού ενδέχεται να περιπλέκει ακόμα περισσότερο το πρόβλημα [2].

Σε αυτή τη διπλωματική εργασία ερευνώνται δύο διαφορετικές τακτικές, οι οποίες αποσκοπούν στη βελτίωση της πορείας βελτιστοποίησης. Και οι δυο τεχνικές το επιδιώκουν επιδρώντας αποκλειστικά στην παραμετροποίηση της γεωμετρίας και όχι στο φυσικό πρόβλημα. Πρακτικά, τροποποιούν την παραμετροποίηση της γεωμετρίας ώστε να δημιουργήσουν τις προυποθέσεις για να λυθεί το πρόβλημα πιο εύκολα.

Η πρώτη τακτική επικεντρώνεται στον Ιακωβιανό πίνακα της παραμετροποίησης. Αυτός εκφράζει τις παραγώγους των σημείων ελέγχου ως προς του κόμβους του επιφανειακού πλέγματος. Επειδή, οι συντεταγμένες των σημείων ελέγχου αποτελούν τις μεταβλητές σχεδιασμού, πρακτικά δείχνει πως επιδρά μια μικρή μεταβολή μιας μεταβλητής σχεδιασμού στην παραμετροποιημένη γεωμετρία. Ο πίνακας ορίζεται ως εξής:

$$\mathbf{J} = \begin{bmatrix} \frac{dX_1}{db_1} & \frac{dX_1}{db_2} & \cdots & \frac{dX_1}{db_N} \\ \frac{dY_1}{db_1} & \frac{dY_1}{db_2} & \cdots & \frac{dY_1}{db_N} \\ \frac{dX_2}{db_1} & \frac{dX_2}{db_2} & \cdots & \frac{dX_2}{db_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{dY_m}{db_1} & \frac{dY_m}{db_2} & \cdots & \frac{dY_m}{db_N} \end{bmatrix} \tag{A'.1}$$

όπου $\vec{X_i} = (X_i, Y_i), (i = 1, ..., m)$ είναι οι κόμβοι του επιφανειακού πλέγματος και $b_n (n = 1, ..., N)$ οι (ενεργές) μεταβλητές σχεδιασμού. Κάθε στήλη αναφέρεται σε μια συγκεκριμένη μεταβλητή σχεδιασμού. Έτσι, ορίζονται δύο παράμετροι για τον Ιακωβιανό πίνακα. Η πρώτη αφορά τη γωνία μεταξύ των στηλών του και οι δεύτερη το μέτρο κάθε στήλης. Επειδή, ο Ιακωβιανός πίνακας είναι πολύ σημαντικός για την πορεία βελτιστοποίησης [1], καθώς οι παράγωγοι ευαισθησίας δίνονται απο τη σχέση:

$$\frac{\partial F}{\partial \vec{b}} = \frac{\partial F}{\partial \vec{X}_s} \frac{\partial \vec{X}_s}{\partial \vec{b}} = \frac{\partial F}{\partial \vec{X}_s} \mathbf{J} \tag{A'.2}$$

διερευνάται το κατά πόσο αυτές οι παράμετροι επιδρούν στην πορεία βελτιστοποίησης. Άρα, η πρώτη τακτική μελετά αυτό το πρόβλημα και αξιολογεί σχετικούς τρόπους βελτίωσης.

Η δεύτερη τακτική, ασχολείται με τη μείωση των μεταβλητών σχεδιασμού, χωρίς όμως να χάνεται η δυνατότητα εύρεσης της βέλτιστης λύσης. Αυτό επιτυγχάνεται χρησιμοποιώντας μια μέθοδο, η οποία αντλώντας πληροφορία από τον Ιακωβιανό πίνακα, έχει τη δυνατότητα να μειώσει σημαντικά τις μεταβλητές που χειρίζεται το πρόβλημα βελτιστοποίησης χωρίς βλάβη στο αποτέλεσμα. Για αυτόν τον σκοπό, κρατά τις πιο σημαντικές μετασχηματισμένες μεταβλητές σχεδιασμού, δηλαδή αυτές με τη μεγαλύτερη ενέργεια, ώστε να διατηρεί ένα κατώτατο όριο συνολικής ενέργειας σε σχέση με τον αρχικό χώρο σχεδιασμού.

<u>Πρώτη Τεχνική</u>

Για να βρεθούν οι γωνίες μεταξύ δύο διανυσμάτων/στηλών i και j του Ιακωβιανού πίνακα, χρησιμοποιείται η παρακάτω σχέση

$$\theta_{ij} = arcos\Big(\frac{\vec{u}_i^T \vec{u}_j}{|\vec{u}_i| \cdot |\vec{u}_j|}\Big), \qquad i \neq j \tag{Α΄.3}$$

όπου, $\vec{u}_i$ και $\vec{u}_j$ είναι η $i-$οστή και $j-$οστή στήλη του Ιακωβιανού πίνακα.

Για να συγκριθούν τα μέτρα επίδρασης των μεταβλητών σχεδιασμού χρησιμοποιούνται οι ιδιάζουσες τιμές του Ιακωβιανού πίνακα, αντί να χρησιμοποιηθεί κατευθείαν το μέτρο της κάθε στήλης-διάνυσμα. Αυτό συμβαίνει, επειδή οι ιδιάζουσες τιμές είναι ικανές να δείξουν το σχετικό μέτρο της γεωμετρικής ευαισθησίας ανάμεσα στις μεταβλητές σχεδιασμού και, άρα, είναι ιδανικές για τέτοιου είδους συγκρίσεις [43].

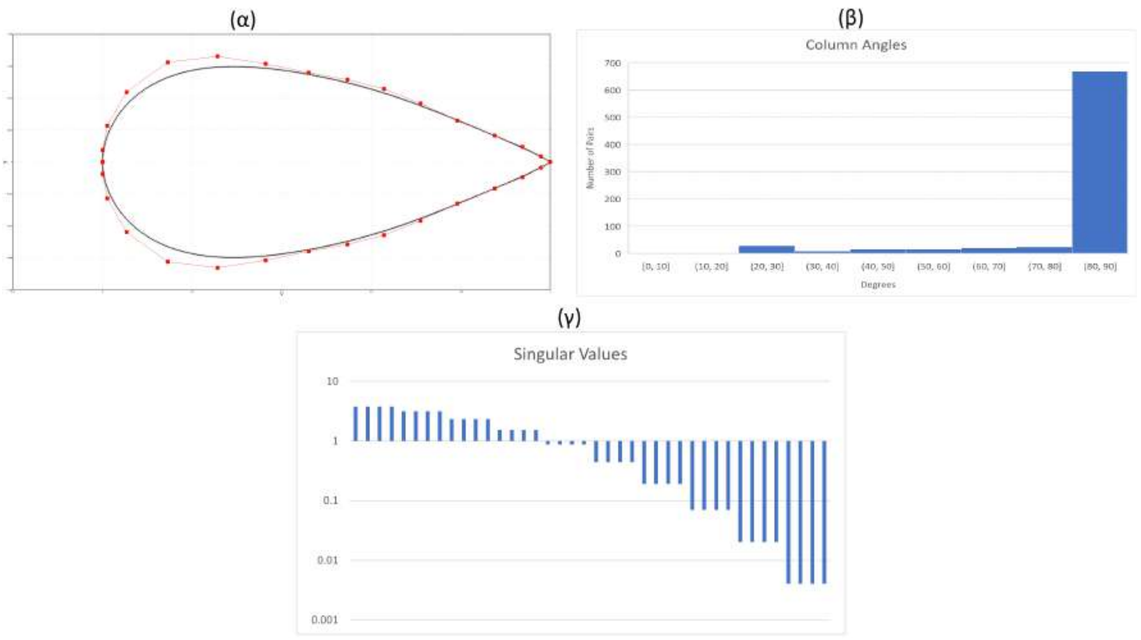
Έτσι, για την παραμετροποίηση της μορφής της αεροτομής NACA 0012 που φαίνεται στο Σχήμα Α΄.8.α, οι παράμετροι παίρνουν τιμές όπως φαίνονται στα Σχήματα Α΄.8.β και Α΄.8.γ αντίστοιχα. Είναι φανερό ότι υπάρχει μεγάλο εύρος τιμών και στα δύο μεγέθη το οποίο δυσκολεύει την πορεία της βελτιστοποίησης. Με άλλα λόγια, έντονη «μη ορθογωνιότητα» ανάμεσα στις στήλες του Ιακωβιανού πίνακα μπορεί να οδηγήσει σε κακώς ορισμένο μητρώο [40], ενώ αντίστοιχα το μεγάλο εύρος στο μέτρο γεωμετρικής ευαισθησίας μπορεί να οδηγεί σε ανακριβή υπολογισμό των παραγώγων ευαισθησίας της συνάρτησης στόχου ως προς κάθε μεταβλητή σχεδιασμού [3].

Για να μελετηθεί η επίδραση αυτών των παραμέτρων επιλέγεται να μετασχηματισθεί ο χώρος σχεδιασμού, σε νέο χώρο, ο οποίος έχει ίδιο αριθμό μεταβλητών σχεδιασμού αλλά οι στήλες του Ιακωβιανού είναι κάθετες μεταξύ τους και τα μέτρα των στηλών, γενικά, ομοιόμορφα. Συγκρίνοντας αυτόν τον χώρο σχεδιασμού με τον αρχικό γίνεται εμφανές το κατά πόσο αυτές οι παράμετροι επηρεάζουν την πορεία της βελτιστοποίησης.

Για να μετασχηματιστεί ο χώρος σχεδιασμού χρησιμοποιείται ο πίνακας $\mathbf{A}$ $[N \times N]$ και εκτελείται ο πολλαπλασιασμός

$$\hat{\vec{b}} = \mathbf{A}\vec{b} \tag{Α΄.4}$$

όπου $\hat{\vec{b}}$ είναι οι νέες/μετασχηματισμένες μεταβλητές σχεδιασμού. Ο πίνακας $\mathbf{A}$ είναι το γινόμενο δύο μητρώων, $\mathbf{A} = \mathbf{S\Phi}$. Ο πίνακας $\mathbf{\Phi}$, ή πίνακας στροφής, είναι ένας $N \times N$ πίνακας υπεύθυνος να επιβάλει ορθογώνιες στήλες ενώ ο $\mathbf{S}$, ή συντελεστής



Σχήμα Α′.8: *(α) η αεροτομή NACA 0012 χρησιμοποιώντας NURBS παραμετροποιήση με 32 σημεία ελέγχου, (β) το εύρος γωνιών μεταξύ των στηλών του Ιακωνιανού πίνακα, (γ) το εύρος των ιδιαζουσών τιμών του Ιακωβιανού πίνακα για κάθε μεταβλητή σχεδιασμού χρησιμοποιώντας λογαριθμική κλίμακα στον κατακόρυφο άξονα.*

κλιμάκωσης, είναι ένας $N \times N$ διαγώνιος πίνακας υπεύθυνος να μειώσει το εύρος των μέτρων της γεωμετρικής ευαισθησίας.

Εφαρμόζοντας ανάλυση πίνακα σε ιδιάζουσες τιμές για τον αρχικό Ιακωβιανό πίνακα, ισχύει:

$$\frac{d\vec{X}_s}{d\vec{b}} = \mathbf{U\Sigma V^T} \tag{Α′.5}$$

όπου $\mathbf{U}$ είναι ένας $m \times N$ ορθομοναδιαίος πίνακας, ο $\mathbf{\Sigma}$ ένας $N \times N$ διαγώνιος πίνακας απαρτιζόμενος από τις ιδιάζουσες τιμές του $\mathbf{J}$ και $\mathbf{V^T}$ ένας $N \times N$ ορθομοναδιαίος πίνακας. Επίσης ο Ιακωβιανός του νέου χώρου σχεδιασμού γράφεται ως:

$$\frac{d\vec{X}_s}{d\vec{\hat{b}}} = \frac{d\vec{X}_s}{d\vec{b}} \cdot \frac{d\vec{b}}{d\vec{\hat{b}}} \tag{Α′.6}$$

και επιλέγοντας $\mathbf{\Phi} = \mathbf{V^T}$, ο Ιακωβιανός πίνακας του μετασχηματισμένου χώρου είναι:

$$\frac{d\vec{X}_s}{d\vec{\hat{b}}} = \mathbf{U\Sigma S}^{-1} \tag{Α′.7}$$

Αυτός έχει ορθογώνιες στήλες καθώς το γινόμενο των $\mathbf{S}$, $\mathbf{\Sigma}$ και $\mathbf{U}$ είναι ένας μη-μοναδιαίος ορθογώνιος πίνακας. Επίσης, επειδή ο $\mathbf{V}$ είναι ορθοκανονικός πίνακας ουσιαστικά αυτός ο μετασχηματισμός οδηγεί σε περιστροφή του χώρου σχεδιασμού.

Από τη σχέση Α΄.7, είναι φανερό ότι οι τιμές του μέτρου της γεωμετρικής ευαισθησίας του νέου χώρου σχεδιασμού εξαρτώνται αποκλειστικά από το γινόμενο των $\mathbf{\Sigma}$ και $\mathbf{S}$. Για αυτόν τον λόγο, επιλέγεται το $\mathbf{S}$ να είναι συνάρτηση του $\mathbf{\Sigma}$,

$$\mathbf{S} = \mathbf{S}\left(\mathbf{\Sigma}\right) \tag{Α΄.8}$$

Δοκιμάζονται 4 διαφορετικές σχέσεις

1. $S_i = 1$ [no scaling]
   Άρα, ο νέος χώρος σχεδιασμού ισοδυναμεί μόνο με περιστροφή του πρωτότυπου.
2. $S_i = \Sigma_i$ [linear]
   Άρα, όλες οι μεταβλητές σχεδιασμού έχουν το ίδιο μέτρο γεωμετρικής ευαισθησίας.
3. $S_i = \sqrt{\Sigma_i}$ [sqrt]
   Αποσκοπεί στην κλιμάκωση των εισόδων του εσσιανού μητρώου με τον πίνακα των ιδιαζουσών τιμών με σκοπό τη διαμόρφωση του με τιμές ίδιας τάξης μεγέθους το οποίο μπορεί να βελτιώσει την πορεία βελτιστοποίησης [3].
4. $S_i = 1/\Sigma_i$ [reciprocal]
   Ο συγκεκριμένος συντελεστής κλιμάκωσης χρησιμοποιείται, πρακτικά ως αντιπαράδειγμα για τα προηγούμενα δυο.

Όπως είναι φανερό η μέθοδος αυτή είναι καθαρά γεωμετρική. Με άλλα λόγια, η σχέση Α΄.2 χωρίζεται σε δύο μέρη, το πρώτο δείχνει τη φυσική συσχέτιση μεταξύ συνάρτησης στόχου και γεωμετρίας, ενώ το δεύτερο δίνει τη γεωμετρική συσχέτιση μεταξύ παραμετροποίησης και γεωμετρίας. Αυτή η μέθοδος επιδρά μόνο στο δεύτερο, με σκοπό αυτό το μέρος να μην επηρεάζει το πρόβλημα και οι παράγωγοι ευαισθησίας της συνάρτησης στόχου να εξαρτώνται αποκλειστικά από τη φυσική συσχέτιση (βλέπε κεφ. 3.3.2).

Αξιολόγηση Πρώτης Τεχνικής

Χρησιμοποιούνται εφαρμογές αεροδυναμικής βελτιστοποίησης για σύγκριση και αξιολόγηση των δύο χώρων σχεδιασμού. Τα γενικά χαρακτηριστικά των πιο σημαντικών εξ αυτών φαίνονται στον Πίνακα Α΄.1 ενώ στον Πίνακα Α΄.2 ορίζονται οι περιορισμοί αυτών των προβλημάτων (όλα τα προβλήματα βρίσκονται στο κεφάλαιο 4 του πλήρους κειμένου στην Αγγλική). Η ροή θεωρείται πάντα ασυμπίεστη και χρησιμοποιείται η προσεγγιστική μέθοδος Newton για ανανέωση των μεταβλητών σχεδιασμού. Επίσης η γωνία πρόσπτωσης για την NACA 0012 είναι $\alpha = 1.92^o$ ενώ για την NACA 4412, $\alpha = 6.95^o$.

Εφαρμογές	Περιγραφή	Παραμετροποήηση	Μετ. Σχ.	Ροή	Αριθμός Reynold's
NACA 0012	ελαχιστοποίηση οπισθέλκουσας	NURBS	40	Στρωτή	1000
NACA 4412	ελαχιστοποίηση οπισθέλκουσας	NURBS	44	Τυρβώδης	$1.5 \cdot 10^6$
S-Type Tube	ελαχιστοποίηση απωλειών ολικής πίεσης	NURBS	20	Στρωτή	1482

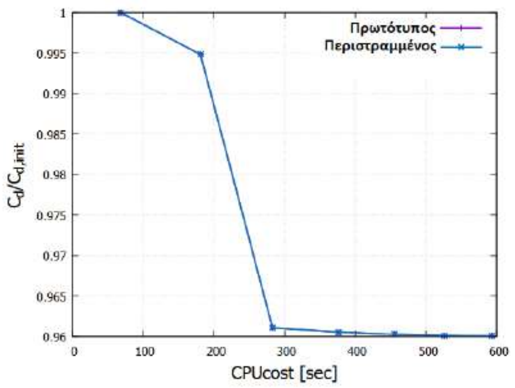
Πίνακας Α΄.1: *Περιγραφή των εφαρμογών που θα χρησιμοποιηθούν για αξιολόγηση της μεθόδου.*

Εφαρμογές	C_L		C_m		C_V	Θέση Σημ. Ελεγχ.
	MIN	MAX	MIN	MAX	MIN	Ναι ή Οχι
NACA 0012	0	0.1	-0.03	0.03	-0.15	Ναι
NACA 4412	1.035	1.235	0	0.15	-0.1	Ναι
S-Type Tube	-	-	-	-	-0.1	Ναι

Πίνακας Α΄.2: *Περιορισμοί κάθε εφαρμογής*

Επίδραση Ορθογωνιότητας

Για την κατανόηση της επίδρασης της γωνίας μεταξύ των στηλών του Ιακωβιανού μητρώου στην πορεία βελτιστοποίησης, συγκρίνεται ο πρωτότυπος χώρος σχεδιασμού με αυτόν που προκύπτει για $\mathbf{S} = \mathbf{I}$ και τα αποτελέσματα φαίνονται στο Σχήμα Α΄.9. Παρατηρείται ότι αυτή η παράμετρος δεν προκαλεί καμία απολύτως επίδραση στην πορεία βελτιστοποίησης, πράγμα το ποίο αποδεικνύεται και μαθηματικά στο κεφ. 4.1 του πλήρους κειμένου στην Αγγλική. Παρόλα αυτά, η περιστροφή του χώρου σχεδιασμού είναι αναγκαία για τον καθορισμό των συντελεστών κλιμάκωσης και, για αυτό, η εφαρμογή της δεν περιορίζεται εδώ.



Σχήμα Α΄.9: *Αεροτομή NACA 0012: Σύγκριση της εξέλιξης της συνάρτησης στόχου του περιστραμένου χώρου σχεδιασμού με τον πρωτότυπο.*

Επίδραση κλιμάκωσης

Η επίδραση του χώρου κλιμάκωσης σε καθένα από τα προβλήματα που παρουσιάστηκαν στον Πίνακα Α΄.1, φαίνονται στο Σχήμα Α΄.10. Είναι φανερό ότι η κλιμάκωση του χώρου σχεδιασμού παίζει σημαντικό ρόλο στην πορεία βελτιστοποίησης και είναι ικανή να την επιταχύνει. Πιο συγκεκριμένα, λαμβάνοντας υπόψη και τα παραδείγματα που παρουσιάζονται στο κεφ. 4, κλιμακώνοντας $S_i = \Sigma_i$, ή με $S_i = \sqrt{\Sigma_i}$ είναι δυνατό να μειωθεί σημαντικά το υπολογιστικός και συγχρόνως να βελτιωθεί η λύση. Επιπλέον, βελτιώνεται η σταθερότητα του αλγόριθμου αφού καταφέρνει να συγκλίνει σε εφαρμογές όπου η κλασική προσέγγιση δεν τα καταφέρνει [αεροτομή NACA 4412]. Επιπρόσθετα, καθώς πλησιάζει τη βελτιστοποποιημένη λύση η ταχύτητα σύγκλισης αυξάνεται, με αποτέλεσμα να μειώνεται σημαντικά το υπολογιστικό κόστος για αυστηρά κριτήρια σύγκλισης. Μάλιστα είναι σημαντικό να σημειωθεί, ότι η θετική επίδραση της κλιμάκωσης παραμένει ανεξάρτητα από το πλήθος των μεταβλητών σχεδιασμού και του τρόπου παραμετροποίησης. Τέλος, ανάμεσα στους δύο συντελεστές κλιμάκωσης, η κλιμάκωση με $S_i = \Sigma_i$ έχει πλεονέκτημα ως προς την εύρεση της βέλτιστης λύσης, ενώ η κλιμάκωση με $S_i = \sqrt{\Sigma_i}$ έχει πλεονέκτημα ως προς το υπολογιστικό κόστος. Παρόλα αυτά επειδή γενικά οι βέλτιστες λύσεις που παράγουν έχουν συνήθως αμελητέα διαφορά, η κλιμάκωση με τη ρίζα των ιδιαζουσών τιμών, τελικά υπερτερεί.

Δεύτερη Τεχνική

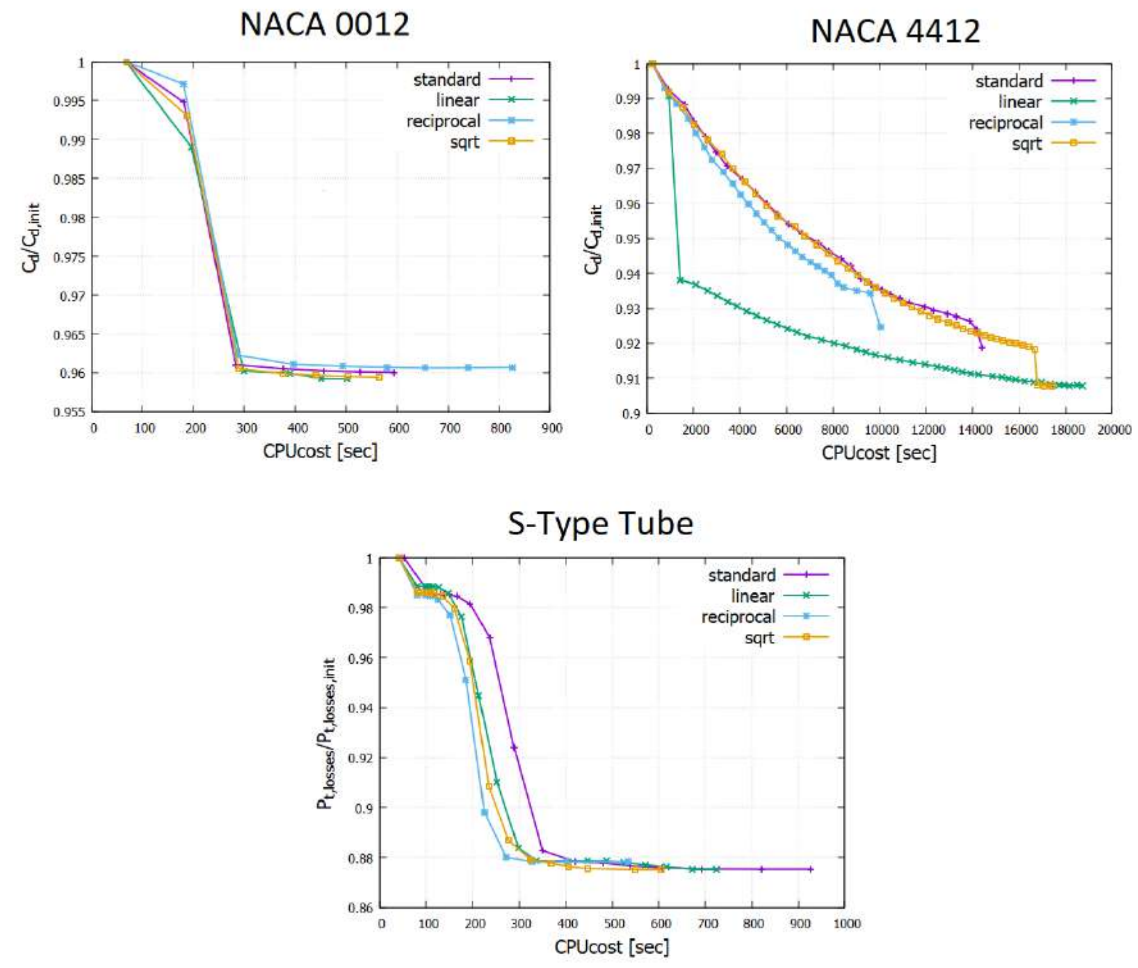
Για την κατανόηση της δεύτερης τεχνικής απαιτείται ο ορισμός της ενέργειας του Ιακωβιανού πίνακα. Αυτή ορίζεται ως η πληροφορία που περιέχει ο Ιακωβιανός πίνακας και υπολογίζεται μέσω των ιδιαζουσών τιμών της κάθε μεταβλητής σχεδιασμού. Με άλλα λόγια, η ενέργεια κάθε μεταβλητής σχεδιασμού δείχνει πόσο επιδρά στη γεωμετρία [27]. Επειδή όμως οι ιδιάζουσες τιμές περιγράφουν τον μετασχηματισμένο χώρο σχεδιασμού, τελικά η μέθοδος καταφέρνει τη μείωση των μεταβλητών σχεδιασμού, διαλέγοντας το καλύτερο υποσύνολο του χώρου σχεδιασμού που προκύπτει μέσω του μετασχηματισμού του αρχικού χώρου.

Η συνολική ενέργεια του πίνακα ορίζεται ως $E_{total} = \sum_{i=1}^{N} \Sigma_i^2$. Στην συνέχεια, ορίζεται η ενέργεια κάθε μεταβλητής σχεδιασμού ως προς την συνολική ενέργεια $E_i = \frac{\Sigma_i^2}{E_{total}}$, καθορίζεται η επιθυμητή τιμή ενέργειας του νέου χώρου σχεδιασμού ως ποσοστό της ολικής ενέργειας (AE), και επιλέγεται ο ελάχιστος αριθμός r μεταβλητών για να ικανοποιείται η ανισότητα

$$E_r = \frac{1}{E_{total}} \sum_{j=1}^{r} {\Sigma_j}^2 \geq AE \qquad (\text{Α΄}.9)$$

Ύστερα, το μητρώο $\mathbf{\Sigma}$ αποκόπτεται ανάλογα με το πλήθος r των μεταβλητών σχεδιασμού και τα μητρώα $\mathbf{U}$, $\mathbf{V}$ μετασχηματίζονται αντίστοιχα. Έτσι, τα νέα μητρώα που προκύπτουν έχουν τις διαστάσεις που παρουσιάζονται στον Πίνακα Α΄.3.

Για να προκύψει ο νέος περιορισμένος χώρος σχεδιασμού, το διάνυσμα των μεταβλητών



Σχήμα Α′.10: *Επίδραση του χώρου κλιμάκωσης στην πορεία βελτιστοποιήσης για τρεις διαφορετικές εφαρμογές.*

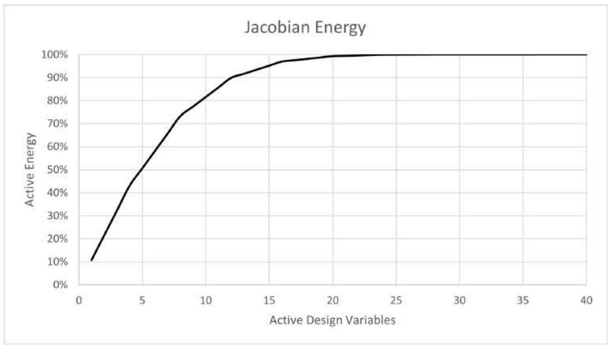
σχεδιασμού πολλαπλασιάζεται με τον πίνακα **A**, όπως φαίνεται στη σχέση Α′.4. Λόγω όμως των νέων διαστάσεων των μητρώων, το $\hat{\vec{b}}$ προκύπτει να έχει διάσταση r αντί για N. Επίσης αφού χρησιμοποιείται η Σχέση Α′.4, είναι προφανές ότι για να μειωθεί ο χώρος σχεδιασμού πρέπει πρώτα να μετασχηματισθεί. Τέλος, σημειώνεται ότι εδώ $\mathbf{S} = \mathbf{I}$, άρα δεν χρησιμοποιείται συντελεστής κλιμάκωσης και αφού η περιστροφή δεν επιδρά στην πορεία βελτιστοποίησης η λύση εξαρτάται αποκλειστικά από τη μείωση του χώρου σχεδιασμού.

Πίνακας	Διαστάσεις
U	$N \times r$
Σ	$r \times r$
V^T	$r \times N$

Πίνακας Α΄.3: *Νεές διαστάσεις μητρώων μετά την αποκοπή του χώρου σχεδιασμού.*

Καταλληλότητα της Μεθόδου

Από το Σχήμα Α΄.11 είναι φανερό, ότι διατηρώντας την συνολική ενέργεια άνω του 99% είναι δυνατή η μείωση των μεταβλητών σχεδιασμού κατά 50% ενώ, ακόμα και για τιμές υψηλότερες του 99.9% της συνολικής ενέργειας, επιτυγχάνεται μείωση 14 μεταβλητών σχεδιασμού. Αυτό υποδηλώνει ότι για μικρή μείωση ενεργείας, άρα για μικρή έλλειψη πληροφορίας πετυχαίνεται μια σημαντική απλοποίηση του προβλήματος.

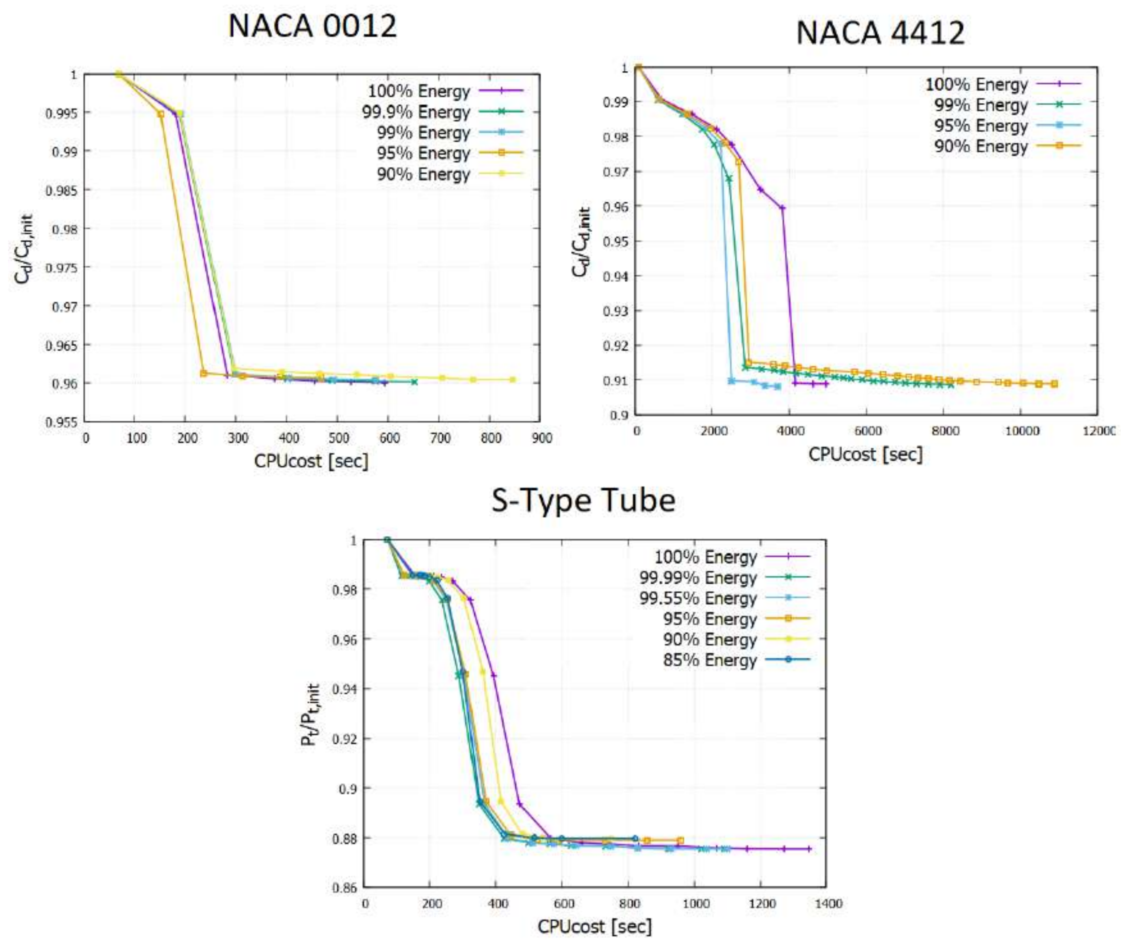


Σχήμα Α΄.11: *Σχέση της ενέργειας του Ιακωβιανού με το πλήθος των μεταβλητών σχεδιασμού, της αεροτομής NACA 0012.*

Αξιολόγηση Δεύτερης Τεχνικής

Οι εφαρμογές είναι παρόμοιες με προηγουμένως. Η μόνη διαφορά έγκειται στο πλήθος των μεταβλητών σχεδιασμού που χρησιμοποιούνται στην εφαρμογή του αγωγού τύπου S και στη γωνία πρόσπτωσης της NACA 4412 η οποία τώρα ισούται με $\alpha = 1.95^o$. Είναι σημαντικό να αναφερθεί ότι διαφορετικές τιμές ενέργειας χρησιμοποιούνται σε κάθε εφαρμογή, ανάλογα με την επίδραση που έχουν στις μεταβλητές σχεδιασμού.

Μέσω του Σχήματος Α΄.12 και των Πινάκων Α΄.4, Α΄.5, Α΄.6 φαίνεται η επίδραση κάθε ποσού ενέργειας στην πορεία βελτιστοποίησης και στις μεταβλητές σχεδιασμού. Λαμβάνοντας υπόψη και τα επιπλέον παραδείγματα που παρουσιάζονται στο κεφάλαιο 5 είναι φανερό ότι η ενέργεια παίζει καθοριστικό ρόλο στην πορεία βελτιστοποίησης. Πιο συγκεκριμένα, για μείωση ενέργειας λιγότερο από 5% επιταχύνεται σημαντικά η πορεία βελτιστοποίησης ή τουλάχιστον, η εύρεση μιας προσεγγιστικής λύσης η οποία απέχει ελάχιστα από τη βέλτιστη, όπως στην αεροτομή NACA 4412. Επίσης, γενικά μειώνοντας την ενέργεια λιγότερο από 5%, οι βελτιστοποιημένες τιμές της συνάρτησης στόχου έχουν αμελητέα διαφορά από εκείνην που προκύπτει για το 100% της ενέργειας και, άρα, επιφέρει βελτίωση της πορείας βελτιστοποίησης. Τέλος, το μειονέκτημα της μεθόδου είναι ότι γενικά διαφορετικές τιμές ενέργειας συμπεριφέρονται με διαφορετικό τρόπο από εφαρμογή σε εφαρμογή, παρόλα αυτά είναι ασφαλές να θεωρηθεί ότι για προβλήματα με λίγες μεταβλητές σχεδιασμού η πτώση δεν πρέπει να ξεπερνά το 1%, και για προβλήματα με πολλές δεν πρέπει να ξεπερνά το 5%. Παρόλα αυτά, αυτή η πρόταση δεν μπορεί να γενικευτεί παντού.



Σχήμα Α΄.12: *Επίδραση της συνολικής ενέργειας στην πορεία βελτιστοποίησης σε τρεις διαφορετικές εφαρμογές.*

Σύγκριση/Συνδιασμός Μεθόδων

Τέλος, επιχειρείται ένας συνδυασμός των δύο μεθόδων. Αυτό επιτυγχάνεται χρησιμοποιώντας την ενεργειακή μέθοδο αποκοπής με μη-μοναδιαίες τιμές για τον πίνακα **S**. Για την αξιολόγηση αυτού του συνδυασμού χρησιμοποιείται η περίπτωση βελτιστοποίησης αγωγού τύπου S. Τα αποτελέσματα φαίνονται στο Σχήμα Α΄.13.

Είναι φανερό ότι χρησιμοποιώντας το 90% της συνολικής ενέργειας για $S_i = \Sigma_i$ δίνει χειρότερα αποτελέσματα από το να χρησιμοποιείται ο ίδιο συντελεστής κλιμάκωσης με το 100% της ενέργειας του Ιακωβιανού πίνακα. Αυτό συμβαίνει επειδή η αποκοπή, προηγείται της κλιμάκωσης και, άρα, ο τελικός χώρος σχεδιασμού δεν θα ικανοποιεί

Ποσό Ενέργειας	Ενεργές Μεταβλ. Σχεδιαμού	Υπολογιστικό Κόστος [sec]	βέλτιστο $\frac{C_d}{C_{d,init}}$
100%	40	592	0.9600
99.9%	26	653	0.9601
99%	20	576	0.9604
95%	15	468	0.9606
90%	13	846	0.9604

Πίνακας Α΄.4: *Αεροτομή NACA 0012: Επίδραση της συνολικής ενέργειας στις μεταβλητές σχεδιασμού και στη βελτιστοποιημένη λύση.*

Ποσό Ενέργειας	Ενεργές Μεταβλ. Σχεδιαμού	Υπολογιστικό Κόστος [sec]	βέλτιστο $\frac{C_d}{C_{d,init}}$
100%	44	4965	0.9088
99%	37	8205	0.9086
95%	27	3705	0.9085
90%	23	10880	0.9088

Πίνακας Α΄.5: *Αεροτομή NACA 4412: Επίδραση της συνολικής ενέργειας στις μεταβλητές σχεδιασμού και στη βελτιστοποιημένη λύση.*

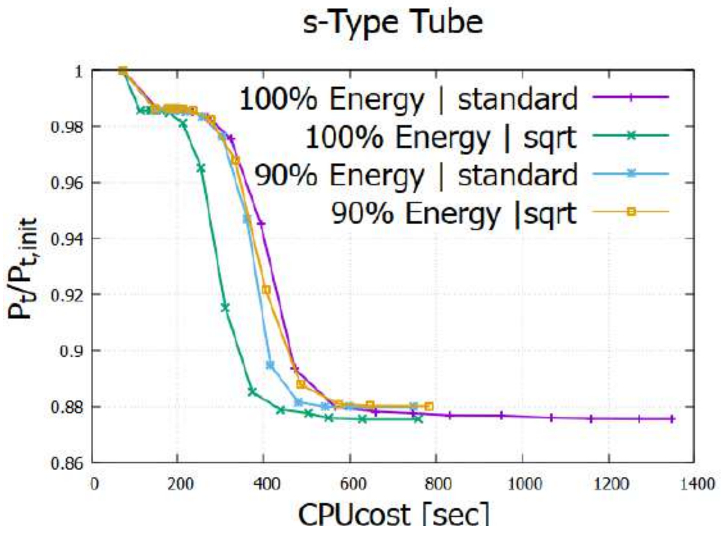
Ποσό Ενέργειας	Ενεργές Μεταβλ. Σχεδιαμού	Υπολογιστικό Κόστος [sec]	βέλτιστο $\frac{C_d}{C_{d,init}}$
100%	36	1346	0.8754
99.99%	27	1091	0.8755
99.55%	20	1099	0.8754
95%	13	959	0.8790
90%	11	746	0.8797
85%	10	820	0.8797

Πίνακας Α΄.6: *Αγωγός Μορφής S: Επίδραση της συνολικής ενέργειας στις μεταβλητές σχεδιασμού και στη βελτιστοποιημένη λύση.*

το απαιτούμενο ποσό ενέργειας, αφού η κλιμάκωση θα το αλλάξει. Επίσης είναι φανερό ότι, χρησιμοποιώντας κλιμάκωση, το υπολογιστικό κόστος μειώνεται παρόμοια με αυτό που προκαλεί η χρήση του 90% της ενέργειας χωρίς κλιμάκωση, το οποίο ήταν και το καλύτερο ποσοστό ενέργειας. Παρόλα αυτά η μέθοδος της κλιμάκωσης εντοπίζει και ελαφρώς καλύτερη λύση με αποτέλεσμα να της δίνει το πλεονέκτημα μεταξύ των δύο μεθόδων.

Συμπεράσματα

Αρχικά φάνηκε ότι οι γωνίες μεταξύ των στηλών του Ιακωβιανού πίνακα δεν παίζουν κανένα ρόλο στην πορεία βελτιστοποίησης. Στη συνέχεια, αποδείχθηκε μέσω διαφόρων εφαρμογών, ότι κλιμακώνοντας τον χώρο σχεδιασμού με σκοπό την επίδραση



Σχήμα Α′.13: *Αγωγός τύπου S: Σύγκριση πορείας βελτιστοποίησης για συνδιασμό των δύο μεθόδων.*

του μέτρου γεωμετρικής ευαισθησίας με τις ιδιάζουσες τιμές ή τη ρίζα αυτών, οδηγεί σε σημαντική βελτίωση της πορείας βελτιστοποίησης. Πιο συγκεκριμένα, βελτιώνει και τα τρία σημαντικά κριτήρια, δηλαδή το υπολογιστικό κόστος, τη βέλτιστη λύση και τη σταθερότητα του αλγόριθμου. Όσα αφορά την αποκοπή του χώρου σχεδιασμού, δείχθηκε ότι και αυτή είναι ικανή να μειώσει το υπολογιστικό κόστος χωρίς να αποκλίνει από τη βέλτιστη λύση. Παρόλα αυτά, το πόσο ενέργειας που θα δημιουργήσει τη μέγιστη βελτίωση δεν είναι προκαθορισμένο. Τέλος, ο συνδυασμός των δύο μεθόδων δεν απέφερε θετικά αποτελέσματα και απαιτούνται περαιτέρω μελέτες πριν ενσωματωθεί σε ένα λογισμικό βελτιστοποίησης.