Approximate Hessian for Accelerated Convergence of Aerodynamic Shape Optimization Problems in an Adjoint-Based Framework

Doug Shi-Dong

Masters of Engineering

Department of Mechanical Engineering

McGill University
Montreal, Quebec

July 2017

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Masters of Engineering

Abstract

An approximation of the Hessian matrix is developed and used to accelerate the convergence of an adjoint-based aerodynamic shape optimization framework. Exact analytical formulations of the direct-direct, adjoint-direct, adjoint-adjoint, and direct-adjoint Hessian approaches are presented and the equivalence between the adjoint-adjoint and direct-adjoint formulations is demonstrated. An approximation of the Hessian is obtained from the analytical formulation by partially solving first-order sensitivities to reduce computational time, while neglecting second-order sensitivities to ease implementation. Error bounds on the resulting approximation are presented for the first-order sensitivities through perturbation analysis and eigenvalue analysis. The proposed method is first assessed using an inverse design pressure problem for a quasi-one-dimensional Euler flow. Additionally, three-dimensional inviscid transonic test cases are used to demonstrate the effectiveness of the method. Eigenvalue analysis of the Hessian for various test cases will seek to explain the resulting acceleration.

Abrégé

Une approximation de la matrice Hessienne est développée et est utilisée afin d'accélérer la convergence d'un système d'optimisation de la forme aérodynamique basée sur la méthode adjointe. Les formulations analytiques exactes de la Hessienne sont présentées sous quatre différentes formes : directe-directe, adjointe-directe, adjointe-adjointe et directe-adjointe. L'équivalence entre la forme adjointe-adjointe et la forme directe-adjointe est démontrée. La Hessienne approximative est formée à partir des formulations analytiques en résolvant partiellement les sensibilités de premier ordre et en approximant certaines dérivées de second ordre. Les erreurs engendrées par l'estimation des dérivées de premier ordre sont examinées à travers la théorie des perturbations et une analyse des valeurs propres. La stratégie est d'abord testée à l'aide d'un problème inverse pour un écoulement quasi-uni-dimensionnel d'Euler. Additionellement, des cas types tridimensionels d'écoulements transsoniques non-visqueux sont utilisés pour démontrer l'efficacité de la méthode. L'analyse des valeurs propres de la Hessienne est performée pour les divers cas type afin d'expliquer l'origine de l'accélération.

Acknowledgements

I would like to express gratitude to my advisor, Prof. Siva Nadarajah, for his guidance and support throughout my academic development. His advice and encouragements have had an important impact on this work. It truly was a pleasure working under his supervision.

I must further thank him and Bombardier Aerospace for their financial support without which this research would not have been possible. Additionally, I am grateful to have received funds from the National Sciences and Engineering Research Council (NSERC), Hydro-Québec and the McGill Faculty of Engineering.

My time at the Computational Aerodynamics has been enjoyable thanks to the helpful discussions with my student colleagues. I am especially thankful for the advice from Farshad Navah, Yao Jiang, Chih-Hao Chen, Philip Zwanenburg, and David Brown who have answered many of my questions. A special thanks to Selim Belhaouane for his friendship and all the coding-related savviness he provided.

I would like to thank Susy Lacerte and my friends who have provided balance and diversion from work.

Finally, I am forever indebted to my mother, Shuojia Dong, who has raised and supported me. She passed down her passion for knowledge, which is now deeply ingrained in me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The use of computational fluid dynamics (CFD) in conjunction with numerical optimization has been a dominant design method in recent years, due to its lower cost than experimental trial and error. Flow solvers and optimization algorithms improvements have led to faster and more robust design tools which serve in the design of the next generation of environmentally and economically friendly aircraft. With the rise of massively parallel supercomputers and the growth of computational power, it is now possible to successively evaluate multiple CFD solutions in a matter of days or weeks. Consequently, the use of optimization algorithms with CFD has allowed engineers to numerically optimize aerodynamic shapes.

1.1 Numerical Optimization

Numerical optimization methods can be classified into two categories: derivative-free and gradient-based methods. The use of derivative-free methods entails that no gradient evaluations are required. Moreover, these methods tend to be more exploratory and may lead to multiple local minima. The approach is advantageous for problems where the functional is neither continuously differentiable nor its gradient can be estimated accurately. However, these methods are often associated with an increased number of objective function evaluations, which is not practical for high-fidelity flow evaluations. Although they may be well-suited for preliminary designs where low-fidelity models are often employed, they are less desirable at later design stages when high-fidelity models with computationally costly functional evaluations are required. Additionally, avoiding the gradient evaluations has become a much smaller benefit with the advent of the adjoint method developed by Pironneau [42] and extended by Jameson [23]. Hence, gradient-based methods have been the preferred optimization algorithms in the aerodynamic shape optimization (ASO) community.

The adjoint method provides the optimizer with the gradients of the objective function with respect to the design variables at a cost independent of the number of design variables, making it superior to the traditional finite-difference method or the direct method, which have a computational cost proportional to the number of design variables. Moreover, gradients, also known as sensitivities, are not only useful for the optimizer, but also invaluable for the engineer. The succession of gradient evaluations gives insights on the design space and how sensitive the system is. As a result, gradient-based optimization also develops knowledge in the system of interest. Gradient-based algorithms such as, steepest descent, quasi-Newton methods and Newton's method use the sensitivities to march towards a descent direction. While the steepest descent method only requires the gradients, Newton's method uses the second-order sensitivities to converge to a local minimum at a quadratic rate, whereas quasi-Newton methods approximate the Hessian to converge to a local minimum superlinearly.

Since the adjoint method was introduced to the ASO field, increasingly complex frameworks have been successfully developed. Wings in various flow regimes are designed with ease and are now simple enough to be considered as academic base cases [1, 6, 53, 20, 26]. Aircraft configurations may be studied through multidisciplinary design optimization that takes into account both the flow and the structure [45, 28, 24]. The adjoint method has also extended to rotorcraft [29, 12, 30] and turbomachinery [56, 34, 57] design optimization. The capability to perform high fidelity design optimization through the adjoint method will continue to grow as the community develops more efficient and robust flow solvers.

1.2 The Hessian

The Hessian, also known as second-order sensitivities, is the matrix containing the second derivatives of a multivariate function. It defines the rate of change of the gradients with respect to the design variables. Geometrically, the gradients represent the slopes of the function in the design space, while the Hessian matrix characterizes the curvature of the design space. In fact, the eigenvalue decomposition of the Hessian produces the magnitude of the principal curvatures and its principal directions. Furthermore, simply looking at the magnitude of the diagonal terms and off-diagonal terms gives a quick understanding of the relationship between the different design variables.

Multiple techniques have been devised to accelerate the optimization problem convergence, most of which involve a Hessian formulation. The steepest descent method can alternatively be interpreted as Newton's method with an identity matrix as its Hessian approximation. The most simplistic change to the gradient descent algorithm is to scale the design variables such that the search direction is not disproportionate. The resulting scaling can be represented as a Hessian with scaled diagonal entries that seek to imitate curvature information. Quasi-Newton methods approximate the Hessian through various algorithms such Broyden-Fletcher-Goldfarb-Shanno (BFGS) and symmetric rank-one (SR1) at each design cycle from the change in design variables and gradients. Analytical approximate Hessians that do not require second-order flow sensitivities have been formulated for inverse design problems due to their quadratic nature [22, 46]. The idea of gradient smoothing using Sobolev gradients [23] has extended to more complex approximate Hessians via shape calculus and Fourier analysis [2, 48]. With the advent of automatic differentiation (AD), exact second-order sensitivities [51, 38, 14] have been employed in ASO problems. However, the high cost of computing the exact Hessian has lead to truncated-Newton methods [35, 19] that use conjugate-gradient or Newton-Krylov methods to approximate the search direction with matrix-vector products of the Hessian. Due to their low computational and implementation cost, identity-initialized quasi-Newton methods are still the workhorse of most ASO frameworks such as SNOPT [16], NLPQLP [47] and IPOPT [55].

Ideally, Newton steps are taken from the initial design to the optimal design to converge quadratically. Unfortunately, the prohibitive cost of evaluating the Hessian and its instability in non-convex regions disfavor the use of Newton's method. The goal of the BFGS algorithm is to update the Hessian matrix at every design cycle, such that it properly estimates the design space curvature while maintaining positive-definiteness. In other words, it retrieves some second-order information at every step. During the final iterations of the optimization, the algorithm hopes to perform Newton steps such that a local minimum is found superlinearly. However, for a quadratic design space, the BFGS algorithm requires as many steps as the number of design variables to retrieve the exact Hessian [10]. A non-linear design space will require even more design cycles to retrieve a good approximation without any guarantee that it ever will. Practically, it is rare to have as many or more design cycles than design variables for ASO. Most practitioners would also be surprised by the inefficacy the BFGS algorithm in the ASO context and how similar it is to a scaled steepest descent.

The use of the adjoint method to calculate the Hessian matrix in an aerodynamic context was first seen in the work of Sherman et al. [51]. It has since been expanded to be used in different areas such as optimization, extrapolation, and uncertainty analysis. The work of Papadimitriou and Giannakoglou [38, 37, 36, 15, 39] is more closely tied to ASO and explores the use of an exactly-initialized BFGS algorithm to optimize two-dimensional aerodynamic shapes. Although the Hessian is only evaluated once in the exactly-initialized

BFGS algorithm, the initial cost is still too large to be effective. It is especially risky when the initial design lies in a non-convex region, where the Hessian must be modified. Furthermore, the initial time required to compute the Hessian is a large investment since the optimizer spends most of its computational time without stepping towards an optimum.

1.3 Contributions and Thesis Overview

The current work aims to drastically reduce the initial cost by evaluating an approximate Hessian. Furthermore, the proposed method is implemented in a three-dimensional framework which introduces additional implementation complexity such as computational scalability and memory usage. The same initialized-BFGS framework as Papadimitriou and Giannakoglou [38] is used, but with an approximate initial Hessian. The approximation is recovered by partially solving first-order flow sensitivities, which are the most computationally intensive terms. An error bound from the resulting approximation is derived for the Hessian entries, the search direction, and the Hessian eigenvalues. Further approximation of the Hessian by discarding second-order flow sensitivities is performed to alleviate the implementation complexity in an existing adjoint-based framework. The proposed method is tested in a quasi-one-dimensional and three-dimensional framework.

The flow solvers are first presented in Chapter 2, followed by the optimization framework in Chapter 3. Chapter 4 describes how the first-order and second-order sensitivities are derived and evaluated. Exact analytical formulations of the direct-direct, adjoint-direct, adjoint-adjoint, and direct-adjoint Hessian are presented based on previous works [51, 38]. The equivalence between the adjoint-adjoint and direct-adjoint formulations is demonstrated. Subsequently, the parts of the Hessian are approximated in order to alleviate computational and implementation cost. The methodology of the proposed methods and the bounds of the incurred errors are presented in Chapter 5. The proposed methods are tested using a quasi-one-dimensional Euler flow inverse pressure problem and three-dimensional inviscid transonic optimization problems, which are presented in Chapter 6.

Chapter 2

Flow Solver

The governing equations and numerical discretization are presented for the quasi-one-dimensional and three-dimensional Navier-Stokes flow solvers. The three-dimensional case uses the McGill University *Computational Aerodynamics Group* in-house finite-volume structured multi-block solver.

2.1 Quasi-one-dimensional Euler

2.1.1 Governing Equations

The conservative form of the quasi-one-dimensional Euler equations defined on a one-dimensional nozzle of area $S(x)$ are described by

$$\frac{\partial \mathbf{w}}{\partial t} + \frac{1}{S}\frac{\partial (\mathbf{f}S)}{\partial x} = \mathbf{q}, \tag{2.1}$$

where the conservative state vector $\mathbf{w}$, the inviscid flux vector $\mathbf{f}$ and the source term vector $\mathbf{q}$ are defined as

$$\mathbf{w} = \begin{bmatrix} \rho \\ \rho u \\ \rho E \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ (\rho E + p)u \end{bmatrix}, \quad \mathbf{q} = \begin{bmatrix} 0 \\ \frac{p}{S}\frac{dS}{dx} \\ 0 \end{bmatrix}. \tag{2.2}$$

The density, velocities and the total energy are respectively denoted as ρ, u, and E. The total energy is given by $E = e + \frac{1}{2}u^2$. The pressure p is determined by the equation of state

$$p = (\gamma - 1)\rho \left[E - \frac{1}{2}u^2\right], \tag{2.3}$$

where γ is the ratio of specific heats. Only steady state solutions are considered in this study.

2.1.2 Numerical Discretization

The governing equations (2.1) can be rewritten over the computational domain in semi-discrete form as,

$$\mathcal{V}\frac{\partial \mathbf{w}}{\partial t} + \mathbf{R}(\mathbf{w}) = \mathbf{0} \quad \text{in } \mathcal{D}, \tag{2.4}$$

where $\mathcal{V}$ is the volume, $\mathbf{R}(\mathbf{w})$ is the residual comprised of the convective and dissipative fluxes, as well as the source term, and $\mathcal{D}$ is the computational domain. Characteristic-based boundary conditions are imposed at the inlet and outlet of the nozzle. The solver used for this study uses a cell-centered finite-volume approach with scalar dissipation and Jameson's fourth-order Runge-Kutta with local time-stepping.

2.2 Three-Dimensional Navier-Stokes

2.2.1 Governing Equations

The conservative form of the three-dimensional compressible Reynolds-averaged Navier-Stokes equations (RANS) in Cartesian coordinates x_1, x_2, and x_3 are described, using Einstein notation, by

$$\frac{\partial \mathbf{w}}{\partial t} + \frac{\partial \mathbf{f_i}}{\partial x_i} - \frac{\partial \mathbf{f_{v,i}}}{\partial x_i} = \mathbf{0} \quad \text{in } V, \tag{2.5}$$

where the conservative state vector $\mathbf{w}$, inviscid flux vector $\mathbf{f_i}$, and viscous flux vector $\mathbf{f_{v,i}}$ are defined as,

$$\mathbf{w} = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho E \end{bmatrix}, \quad \mathbf{f_i} = \begin{bmatrix} \rho u_i \\ \rho u_i u_1 + p\delta_{i1} \\ \rho u_i u_2 + p\delta_{i2} \\ \rho u_i u_3 + p\delta_{i3} \\ \rho E u_i + p u_i \end{bmatrix}, \quad \mathbf{f_{v,i}} = \begin{bmatrix} 0 \\ \tau_{ij}\delta_{i1} \\ \tau_{ij}\delta_{i2} \\ \tau_{ij}\delta_{i3} \\ u_j \tau_{ij} + k\frac{\delta T}{\delta x_i} \end{bmatrix}. \tag{2.6}$$

The density, velocities, Kronecker delta function, the total energy, and viscous stresses are respectively denoted as ρ, u_i, δ_{ij}, E, and τ_{ij}. The total energy is given by $E = e + \frac{1}{2}(\|\mathbf{u}\|^2)$. The pressure p is determined by the equation of state

$$p = (\gamma - 1)\rho \left[E - \frac{\|\mathbf{u}\|^2}{2} \right], \tag{2.7}$$

where γ is the ratio of specific heats. Steady state solutions are considered in this study. For the inviscid solutions, the viscous fluxes are set to zero; the Euler equations are thus solved.

2.2.2 Numerical Discretization

Eq. (2.5) can be rewritten over the computational domain in semi-discrete form as,

$$\mathcal{V}\frac{\partial \mathbf{w}}{\partial t} + \mathbf{R}(\mathbf{w}) = \mathbf{0} \quad \text{in } \mathcal{D}, \tag{2.8}$$

where $\mathcal{V}$ is the volume, $\mathbf{R}(\mathbf{w})$ is the residual comprised of the convective and dissipative fluxes, and $\mathcal{D}$ is the computational domain. The solver used for this study uses a second-order cell-centered finite-volume numerical scheme with H-CUSP [54], five-stage modified Runge-Kutta [27] with local time-stepping. Lower-upper symmetric Gauss-Seidel (LU-SGS) [11] implicit smoother is used for more challenging test cases. No-penetration conditions are imposed at the wall. Residual averaging and multigrid techniques are used to accelerate the convergence.

Chapter 3

Optimization Framework

This chapter discusses the optimization framework used to minimize nonlinear aerodynamic objective functions. Unconstrained optimization is first presented to demonstrate the basic concepts and the role of the Hessian, followed by the sequential quadratic programming approach used in this work. Finally, we expand on the details of the mesh deformation scheme used to adapt the volume mesh and the different parametrization methods that will determine the design space.

3.1 Unconstrained Optimization

The unconstrained optimization problem seeks to minimize the cost function I with respect to the design variables $\boldsymbol{\alpha}$.

$$\begin{aligned} \text{minimize} \quad & I(\boldsymbol{\alpha}), \\ \text{w.r.t.} \quad & \boldsymbol{\alpha}. \end{aligned} \tag{3.1}$$

3.1.1 Gradient Descent

The gradient descent, also known as steepest descent, algorithm only requires the gradient vector $\mathbf{g} = \boldsymbol{\nabla} I$ to produce a search direction. In fact, the search direction $\mathbf{p}^k$ at iteration k is simply the negative of the gradient scaled by a factor γ

$$\boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k = \mathbf{p}^k = -\gamma \mathbf{g}^k. \tag{3.2}$$

The factor γ is the result of a line search to ensure a decrease of the cost function.

3.1.2 Newton's Method

Newton's method uses the Hessian $\mathbf{H} = \boldsymbol{\nabla}^2 I$ to scale and rotate the search direction. It can be derived from a Taylor series expansion of the function of interest around the point $\boldsymbol{\alpha}^k$

$$I(\boldsymbol{\alpha}^{k+1}) = I(\boldsymbol{\alpha}^k) + \boldsymbol{\nabla}^T I \mathbf{p}^k + \tfrac{1}{2}(\mathbf{p}^k)^T \mathbf{H} \mathbf{p}^k + \mathcal{O}((\mathbf{p}^k)^3). \tag{3.3}$$

The minimum is found when the gradient of the functional is zero, which gives the formula for a search direction.

$$\begin{aligned} \mathbf{0} &= \mathbf{g} + \mathbf{H}\mathbf{p}^k, \\ \mathbf{p}^k &= -\mathbf{H}^{-1}\mathbf{g}. \end{aligned} \tag{3.4}$$

For a more robust algorithm a scaling factor γ is added to the search direction formulation.

$$\mathbf{p}^k = -\gamma \mathbf{H}^{-1}\mathbf{g}. \tag{3.5}$$

As the solution gets closer to the optimum, the scaling factor should be equal to one to ensure quadratic convergence.

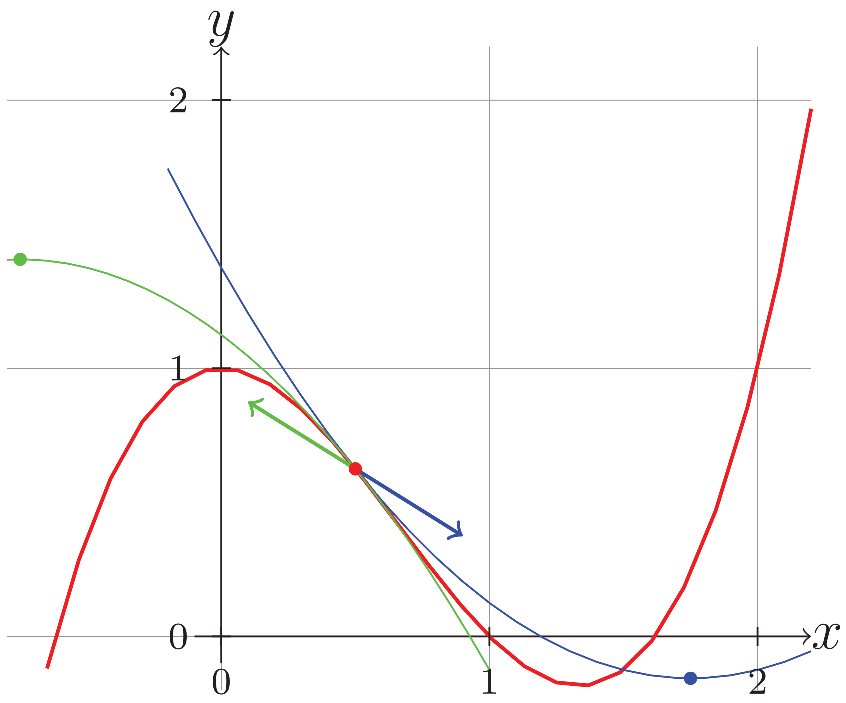


Figure 3.1 – Simple function $f(x) = x^3 - 2x^2 + 1$ (red), gradient descent (blue), and Newton's method (green)

Note that the Hessian matrix needs to be positive-definite in order to move in a direction of descent. If an eigenvalue of the Hessian is found to be negative, the point is in a region of non-convexity and the curvature information should be disregarded. This effect can easily be shown through the one-dimensional example represented in Fig. (3.1). Let us define the functional of interest, its gradient and its second-order sensitivity

$$y = f(x) = x^3 - 2x^2 + 1, \quad \nabla f = 3x^2 - 4x, \quad \nabla^2 f = 6x - 4, \tag{3.6}$$

which has a local minimum located at $x = 4/3$. A starting point $x = 1/2$ would have its gradient defined as $\nabla f(1/2) = -5/4$ and its Hessian defined as $\nabla^2 f(1/2) = -1$. The negative of the gradient is the correct search direction. However, if the Hessian is taken into account, the search direction does not tend towards the local minimum. This issue is discussed more thoroughly in section (5.4).

Additionally, an eigendecomposition of the Hessian reveals that small eigenvalues

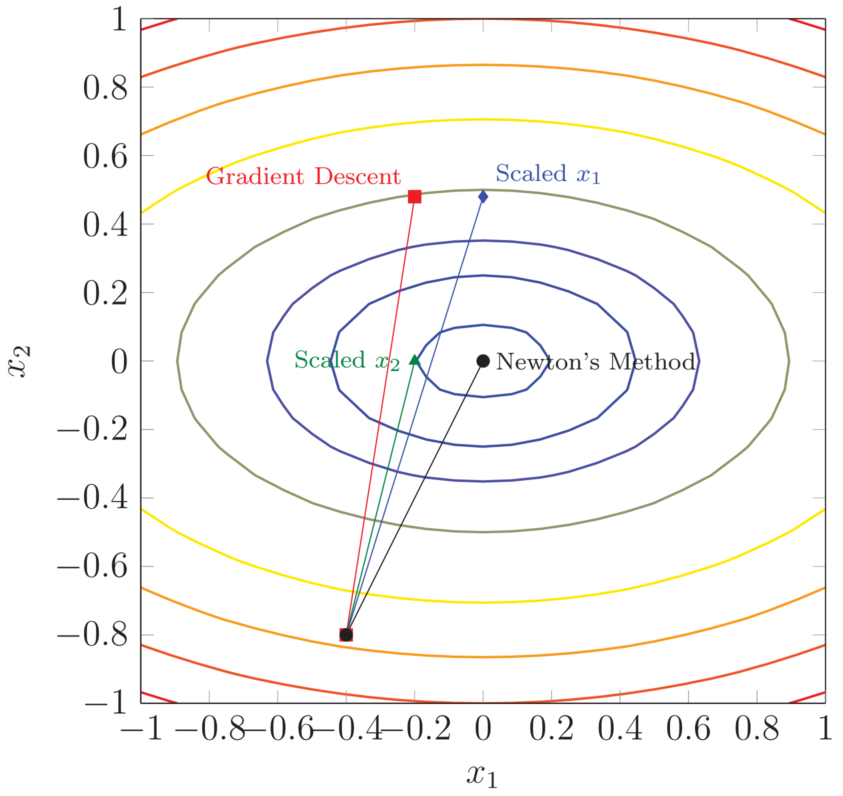


Figure 3.2 – Minimize $I(\mathbf{x}) = 1/4x_1^2 + 4/5x_2^2$, with gradient descent (red), scaled by an approximate Hessian $\mathbf{H} = \begin{bmatrix} 1/2 & 0 \\ 0 & 1 \end{bmatrix}$ (blue), an approximate Hessian $\mathbf{H} = \begin{bmatrix} 1 & 0 \\ 0 & 8/5 \end{bmatrix}$ (green), and the exact Hessian $\mathbf{H} = \begin{bmatrix} 1/2 & 0 \\ 0 & 8/5 \end{bmatrix}$ (black)

allow the optimizer to take larger steps in directions of low curvature, thus accelerating the convergence. On the other hand, large eigenvalues reduce the step length in directions of high curvature. In effect, the Hessian both accelerates and stabilizes the optimization process through proper scaling of the search direction, whereas the gradient descent algorithm requires tuning of the step size. Fig. (3.2) shows how gradient descent overshoots in the direction of high curvature and undershoots the direction of low curvature. Proper scaling results in the optimal search direction for the quadratic problem. Furthermore, eigenvectors of the Hessian are required to rotate the search direction such that scaling can occur in the proper frame of reference as shown in Fig. (3.3).

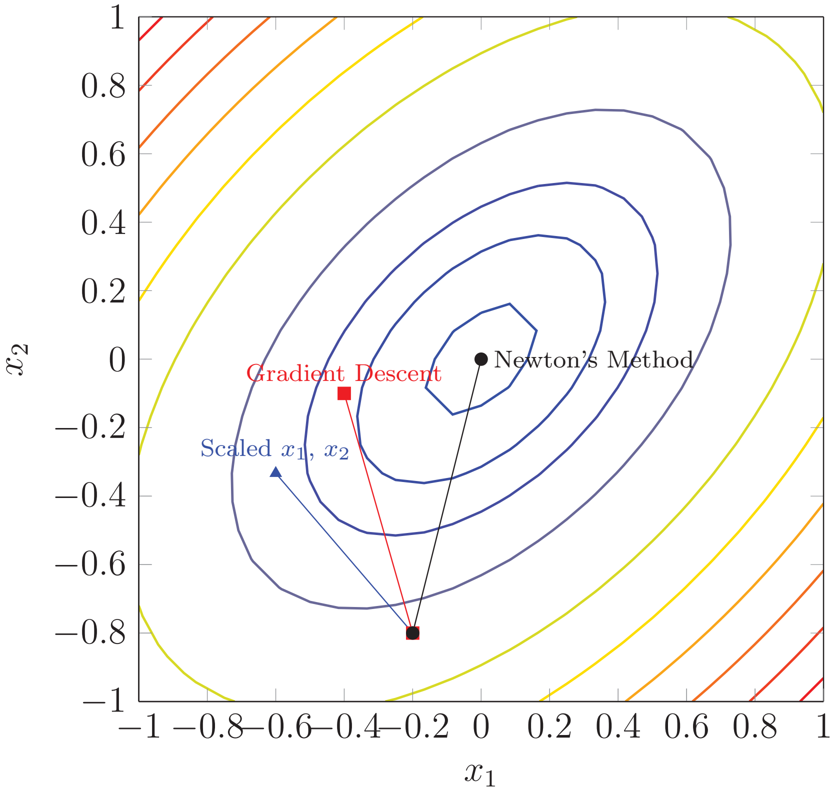


Figure 3.3 – Minimize $I(\mathbf{x}) = 1/2x_1^2 + 1/2x_2^2 - 1/2xy$, with gradient descent (red), scaled by an approximate Hessian $\mathbf{H} = \begin{bmatrix} 1/2 & 0 \\ 0 & 3/2 \end{bmatrix}$ (blue), and the exact Hessian $\mathbf{H} = \begin{bmatrix} 1 & -1/2 \\ -1/2 & 1 \end{bmatrix} = \begin{bmatrix} -\sqrt{1/2} & -\sqrt{1/2} \\ -\sqrt{1/2} & \sqrt{1/2} \end{bmatrix} \begin{bmatrix} 1/2 & 0 \\ 0 & 3/2 \end{bmatrix} \begin{bmatrix} -\sqrt{1/2} & -\sqrt{1/2} \\ -\sqrt{1/2} & \sqrt{1/2} \end{bmatrix}$ (black)

3.1.3 Quasi-Newton Method

Since the Hessian size is quadratically proportional to the number of design variables, it may not always be tractable to evaluate it. In fact, most ASO frameworks do not evaluate the Hessian and use a variation of the Broyden-Fletcher-Goldfarb-Shanno (BFGS) [9] [13] [17] [50] algorithm. The BFGS algorithm is a quasi-Newton method, which does not require the exact Hessian. However, it attempts to recover the second-order information from the change in gradients as the optimizer iterates through the design space. It is defined by the rank-two formula

$$\mathbf{H}^{k+1} = \mathbf{H}^k + \frac{\mathbf{y}\mathbf{y}^T}{\mathbf{y}^T\mathbf{p}} - \frac{(\mathbf{H}^k\mathbf{p})(\mathbf{H}^k\mathbf{p})^T}{\mathbf{p}^T\mathbf{H}^k\mathbf{p}}, \tag{3.7}$$

where $\mathbf{p} = \boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k$ and $\mathbf{y} = (\mathbf{g})^{k+1} - (\mathbf{g})^k$. The update guarantees positive definiteness if $\mathbf{H}^k$ is positive-definite and $\mathbf{y}^T\mathbf{p} > 0$.

3.2 Sequential Quadratic Programming (SQP)

The sequential quadratic programming framework is used to solve an optimization problem where the objective function or constraints are nonlinear. As the name entails, SQP methods successively solve quadratic problems (QPs) in the form of

$$\begin{aligned} \text{minimize} \quad & I(\boldsymbol{\alpha}) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{H}\boldsymbol{\alpha} + \mathbf{g}^T\boldsymbol{\alpha}, \\ \text{w.r.t.} \quad & \boldsymbol{\alpha}, \\ \text{subject to} \quad & c_j(\boldsymbol{\alpha}) = 0,\ j = 1, \ldots, m, \\ & \hat{c}_j(\boldsymbol{\alpha}) \geq 0,\ j = 1, \ldots, n. \end{aligned} \tag{3.8}$$

Convex QPs are known to be easily solvable through active-set methods or interior-point methods. Therefore, given the following equality constrained nonlinear optimization problem,

$$\begin{aligned} \text{minimize} \quad & I(\boldsymbol{\alpha}) = \tfrac{1}{2}\boldsymbol{\alpha}^T \mathbf{H}\boldsymbol{\alpha} + \mathbf{g}^T\boldsymbol{\alpha}, \\ \text{w.r.t.} \quad & \boldsymbol{\alpha}, \\ \text{subject to} \quad & c_j(\boldsymbol{\alpha}) = 0,\ j = 1, \dots, m, \end{aligned} \tag{3.9}$$

where $I(\boldsymbol{\alpha})$ is the quadratic approximation of the objective function, $\boldsymbol{\alpha}$ are the design variables, and $\mathbf{c}(\boldsymbol{\alpha})$ are the active equality constraints, the goal is to reformulate it into a sequence of convex QPs. The first step is the introduction of a dual variable $\boldsymbol{\lambda}$ which gives rise to the Lagrangian function

$$\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\lambda}) = I(\boldsymbol{\alpha}) - \boldsymbol{\lambda}^T \mathbf{c}(\boldsymbol{\alpha}). \tag{3.10}$$

Let us also define the Jacobian of the constraints as

$$\mathbf{A}(\boldsymbol{\alpha}) = \Big[\boldsymbol{\nabla} c_1(\boldsymbol{\alpha}), \boldsymbol{\nabla} c_2(\boldsymbol{\alpha}), \dots, \boldsymbol{\nabla} c_m(\boldsymbol{\alpha})\Big]^T. \tag{3.11}$$

The first-order Karush-Kuhn-Tucker (KKT) conditions of the presented problem can be expressed as

$$\mathbf{F}(\boldsymbol{\alpha}, \boldsymbol{\lambda}) = \begin{bmatrix} \mathbf{g}(\boldsymbol{\alpha}) - \mathbf{A}(\boldsymbol{\alpha})^T \boldsymbol{\lambda} \\ \mathbf{c}(\boldsymbol{\alpha}) \end{bmatrix} = 0. \tag{3.12}$$

In order to solve the above system, one may use Newton's method, where the Jacobian of $\mathbf{F}$ is defined by

$$\mathbf{F}'(\boldsymbol{\alpha}, \boldsymbol{\lambda}) = \begin{bmatrix} \boldsymbol{\nabla}^2_{\alpha\alpha} \mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\lambda}) & -\mathbf{A}(\boldsymbol{\alpha})^T \\ \mathbf{A}(\boldsymbol{\alpha}) & \mathbf{0} \end{bmatrix}. \tag{3.13}$$

The Newton step at an iterate $(\boldsymbol{\alpha}^k, \boldsymbol{\lambda}^k)$ is given by

$$\begin{bmatrix} \boldsymbol{\alpha}^{k+1} \\ \boldsymbol{\lambda}^{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha}^{n} \\ \boldsymbol{\lambda}^{n} \end{bmatrix} + \begin{bmatrix} \mathbf{p}^{k} \\ \mathbf{p}^{\lambda} \end{bmatrix}. \tag{3.14}$$

where $\mathbf{p} = \begin{bmatrix} \mathbf{p}^k \\ \mathbf{p}^\lambda \end{bmatrix}$ is found from the Newton-KKT system

$$\mathbf{F}'\mathbf{p} = \mathbf{F}, \tag{3.15}$$

$$\begin{bmatrix} \boldsymbol{\nabla}^2_{\boldsymbol{\alpha}\boldsymbol{\alpha}}\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\lambda}) & -\mathbf{A}(\boldsymbol{\alpha})^T \\ \mathbf{A}(\boldsymbol{\alpha}) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{p}^k \\ \mathbf{p}^\lambda \end{bmatrix} = \begin{bmatrix} -\mathbf{g}(\boldsymbol{\alpha}) + \mathbf{A}(\boldsymbol{\alpha})^T\boldsymbol{\lambda} \\ -\mathbf{c}(\boldsymbol{\alpha}) \end{bmatrix}. \tag{3.16}$$

The Hessian of the Lagrangian is required, which includes both the Hessian of the cost function $\boldsymbol{\nabla}^2 I$ and the Hessian of the nonlinear constraints $\boldsymbol{\nabla}^2\mathbf{c}(\boldsymbol{\alpha})$. An inequality constrained problem follows the same derivation. However, the QP becomes an iterative process to solve for the dual variables stemming from the inequalities.

Overall, the optimization process follows Fig. (3.4). A thorough review of gradient-based optimization techniques may be found in Nocedal and Wright [32]. The current work uses NLPQLP from Schittkowski [47] as the black-box optimizer since it provides the user access to the Hessian.

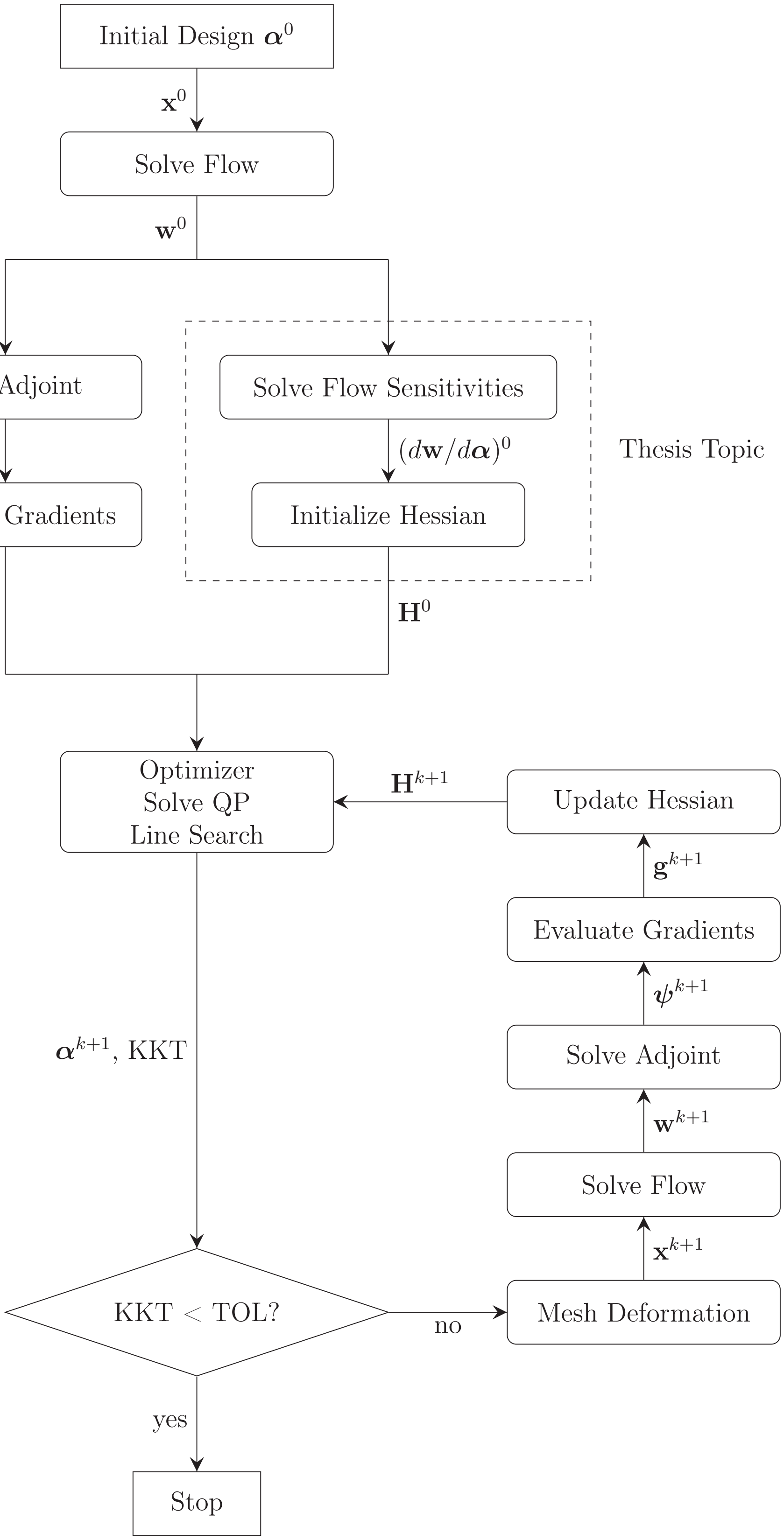


Figure 3.4 – Optimization framework flowchart

3.3 Mesh Deformation

Since the aerodynamic surface changes throughout the design cycles, it is important for the volume mesh to adapt accordingly. The mesh deformation scheme used in this work employs radial basis functions (RBF) to interpolate the volume mesh points. The volume mesh points are moved with respect to the RBF points displacements which are determined by the surface points. The surface points are parametrized through the design variables. Therefore, the volume mesh points are a function of the design variables.

$$\mathbf{x} = \mathbf{x_{vol}} = \mathbf{x_{vol}}(\mathbf{x_{rbf}}(\mathbf{x_{surf}}(\boldsymbol{\alpha}))) = \mathbf{x_{vol}}(\boldsymbol{\alpha}). \tag{3.17}$$

The RBF is defined as

$$f(x) = \sum_{i=1}^{n} \phi(\|x - x_i\|)\beta_i, \tag{3.18}$$

where ϕ is Wendland's C^2 function, $\|x - x_i\|$ is the Euclidean norm between two points, and β_i is the ith RBF coefficient. Given a set of RBF displacements, the goal is to solve for the RBF coefficients β_i to obtain an interpolating function. The interpolating function at the RBF points can be written as a system of linear equations

$$\Delta\mathbf{x_{rbf}} = \mathbf{M}\boldsymbol{\beta}. \tag{3.19}$$

The displacements of the volume mesh points use the same interpolating function

$$\Delta\mathbf{x_{vol}} = \mathbf{A}\boldsymbol{\beta} = \mathbf{A}\left[\mathbf{M}\right]^{-1}\Delta\mathbf{x_{rbf}}. \tag{3.20}$$

In order to avoid re-evaluating the factorization of the matrix $\mathbf{M}$, the mesh deformation always originates from the initial mesh. Therefore, at the nth design cycle, the new volume mesh points will be defined by

$$\mathbf{x}^n_{\mathbf{vol}} = \mathbf{x}^0_{\mathbf{vol}} + \mathbf{A}^0 \left[\mathbf{M}^0\right]^{-1} \Delta \mathbf{x}^n_{\mathbf{rbf}}. \tag{3.21}$$

Since every processor has all the necessary information to move its own volume points, the mesh deformation can be done fully in parallel. The reader may refer to deBoer [8] or Jakobsson [21] for further readings about mesh deformation using RBF.

Finally, it is possible to define the metric sensitivities

$$\frac{d\mathbf{x}_{\mathbf{vol}}}{d\mathbf{x}_{\mathbf{rbf}}} = \frac{\Delta \mathbf{x}_{\mathbf{vol}}}{\Delta \mathbf{x}_{\mathbf{rbf}}} = \mathbf{A} \left[\mathbf{M}\right]^{-1}, \tag{3.22}$$

$$\frac{d\mathbf{x}_{\mathbf{vol}}}{d\boldsymbol{\alpha}} = \frac{d\mathbf{x}_{\mathbf{vol}}}{d\mathbf{x}_{\mathbf{rbf}}} \frac{d\mathbf{x}_{\mathbf{rbf}}}{d\mathbf{x}_{\mathbf{surf}}} \frac{d\mathbf{x}_{\mathbf{surf}}}{d\boldsymbol{\alpha}} = \mathbf{A} \left[\mathbf{M}\right]^{-1} \frac{d\mathbf{x}_{\mathbf{rbf}}}{d\mathbf{x}_{\mathbf{surf}}} \frac{d\mathbf{x}_{\mathbf{surf}}}{d\boldsymbol{\alpha}}, \tag{3.23}$$

which will be used throughout the remainder of this work. Since the RBF points are placed on the surface points, the sensitivity of the RBF points with respect to the surface points $(dx_{\text{rbf}})_i/(dx_{\text{surf}})_j$ is 1 for the corresponding points and 0 otherwise.

3.4 Parametrization

The surface parametrization method is an important choice since it defines the optimization problem. Although using surface mesh points as the design variables themselves will result in the largest design space encompassing globally optimal shapes, it also creates multiple issues such as non-smooth surfaces, complex constraint definitions and an ill-conditioned optimization problem. In effect, using the surface parametrization control points as design variables is a way to condition the design space. The use of B-splines [41] and free-form deformation (FFD) [49] is used in this work. Details for both approaches are expanded in the following two subsections.

3.4.1 B-spline

A B-spline curve of order $p+1$ is defined by

$$\mathbf{C}(u) = \sum_{i=0}^{n} N_{i,p}(u)\mathbf{P}_i, \tag{3.24}$$

where $\mathbf{P}_i$ are the control points and $N_{i,p}$ are the pth-degree B-spline basis functions defined on the normalized knot vector of size m

$$U = \{\underbrace{0,\ldots,0}_{p+1}, u_{p+1},\ldots,u_{m-p-1},\underbrace{1,\ldots,1}_{p+1}\}. \tag{3.25}$$

The B-spline basis functions $N_{i,p}(u)$ are recursively defined as

$$\begin{aligned} N_{i,0}(u) &= \begin{cases} 1 & \text{if } u_i < u < u_{i+1} \\ 0 & \text{otherwise.} \end{cases} \\ N_{i,p}(u) &= \frac{u-u_i}{u_{i+p}-u_i}N_{i,p-1}(u) + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1}(u). \end{aligned} \tag{3.26}$$

The one-dimensional curve is used to define a continuous distribution of design variable, such as twist, thickness, or dihedral along the wing span. There are multiple advantages to parametrize the geometry based on B-spline curves First, a higher degree B-spline can achieve arbitrarily high smoothness, while a lower degree B-spline allows for local control of the geometry. Second, B-spline curves may be extended to non-uniform rational B-splines which are compatible with computer aided design software and would allow an exact representation of the parametrized surface. Finally, the analytical sensitivities of the control points are conveniently available

$$\frac{d\mathbf{C}}{d\mathbf{P}_i} = N_{i,p}(u), \tag{3.27}$$

and will be used in the remainder of this work. The reader may consult Poirier [43] or Bisson

[6] for the use of this parametrization method in an ASO framework.

3.4.2 Free-Form Deformation

The FFD parametrization introduced by Sederberg [49] encloses the design surface with a box defined by FFD points. The box uses a local coordinate system on a parallelepiped region with principal directions $\mathbf{S}$, $\mathbf{T}$, and $\mathbf{U}$. Any surface point $\mathbf{x_{surf}}$ has (s, t, u) coordinates in this system such that

$$\mathbf{x_{surf}} = \mathbf{X}_0 + s\mathbf{S} + t\mathbf{T} + u\mathbf{U}. \tag{3.28}$$

The (s, t, u) coordinates of the surface points can be recovered as follows

$$s = \frac{\mathbf{T} \times \mathbf{U}(\mathbf{x_{surf}} - \mathbf{X}_0)}{\mathbf{T} \times \mathbf{U} \cdot \mathbf{S}},\ t = \frac{\mathbf{S} \times \mathbf{U}(\mathbf{x_{surf}} - \mathbf{X}_0)}{\mathbf{S} \times \mathbf{U} \cdot \mathbf{T}},\ u = \frac{\mathbf{S} \times \mathbf{T}(\mathbf{x_{surf}} - \mathbf{X}_0)}{\mathbf{S} \times \mathbf{T} \cdot \mathbf{U}}. \tag{3.29}$$

The $l \times m \times n$ control points $P_{i,j,k}$ are distributed on a lattice and their locations are defined by

$$\mathbf{P}_{i,j,k} = \mathbf{X}_0 + \frac{i}{l}\mathbf{S} + \frac{j}{m}\mathbf{T} + \frac{k}{n}\mathbf{U}. \tag{3.30}$$

The design surface is deformed by prescribing displacements to the control points $\mathbf{P}_{i,j,k}$. For a surface point $\mathbf{x_{surf}}$, its local coordinate (s, t, u) is evaluated as shown in Eq. (3.29) and its new location $\mathbf{x}^*_{\mathbf{surf}}$ is obtained by evaluating the trivariate Bernstein polynomial

$$\mathbf{x}^*_{\mathbf{surf}} = \sum_{i=0}^{l} \begin{pmatrix} l \\ i \end{pmatrix} s^i (1-s)^{l-i} \left[\sum_{j=0}^{m} \begin{pmatrix} m \\ j \end{pmatrix} t^j (1-t)^{m-j} \left[\sum_{k=0}^{n} \begin{pmatrix} n \\ k \end{pmatrix} u^k (1-u)^{n-k} \mathbf{P}_{i,j,k} \right] \right]. \tag{3.31}$$

Like the B-spline curves, smoothness and local control is a desirable feature of the FFD parametrization. It also easily allows continuity to be maintained between different FFD boxes, such as the wing-body fairing. Furthermore, a simple formulation for the change in volume can be obtained, which is particularly useful in test cases where the wing volume must be maintained due to fuel capacity constraints. Finally, the analytical sensitivities of the control points are also readily available since the Bernstein polynomial is a linear combination of the Bernstein basis function.

$$\frac{d\mathbf{x}_{\mathbf{surf}}}{d\mathbf{P}_{i,j,k}} = \begin{pmatrix} l \\ i \end{pmatrix} s^i (1-s)^{l-i} \begin{pmatrix} m \\ j \end{pmatrix} t^j (1-t)^{m-j} \begin{pmatrix} n \\ k \end{pmatrix} u^k (1-u)^{n-k}. \tag{3.32}$$

Chapter 4

Sensitivity Analysis

4.1 First-Order Sensitivities

The first-order sensitivities of an objective function, also known as gradients, can be computed analytically from either direct differentiation or the adjoint method. Although the adjoint method is used to compute the gradient, the direct method will be required for the evaluation of the Hessian. The finite-difference approach is also introduced since it is a useful tool to verify the gradients retrieved through the analytical formulations.

4.1.1 Flow Sensitivities

The objective function $I = I(\mathbf{w}, \mathbf{x})$ is a function of the flow state variables $\mathbf{w} = \mathbf{w}(\mathbf{x})$ and the geometry $\mathbf{x} = \mathbf{x}(\boldsymbol{\alpha})$, which in turn is parametrized through the control points $\boldsymbol{\alpha}$. The number of design variables $\boldsymbol{\alpha}$ is defined by $N_{\boldsymbol{\alpha}}$. The state variables are implicitly defined through the steady-state solution of the Navier-Stokes equations, where the residual of a converged solution is zero.

$$\mathbf{R} = \mathbf{R}(\mathbf{w}, \mathbf{x}) = \mathbf{0}. \tag{4.1}$$

The gradient of the cost function and the flow residual with respect to control points are defined through the chain rule.

$$\frac{dI(\mathbf{w},\mathbf{x})}{d\boldsymbol{\alpha}} = \frac{\partial I}{\partial \mathbf{w}}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \frac{\partial I}{\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}, \tag{4.2}$$

$$\frac{d\mathbf{R}(\mathbf{w},\mathbf{x})}{d\boldsymbol{\alpha}} = \frac{\partial \mathbf{R}}{\partial \mathbf{w}}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} = \mathbf{0}. \tag{4.3}$$

4.1.2 Direct Differentiation

The derivative of the state variables with respect to the design variables $d\mathbf{w}/d\boldsymbol{\alpha}$, also referred as first-order flow sensitivities, may be solved by using Eq. (4.3). The linear system is then solved for $N_{\boldsymbol{\alpha}}$ right-hand sides.

$$\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} = -\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}. \tag{4.4}$$

Unfortunately, the number of required solutions increases linearly with the number of design variables. As a result, the aerodynamic shape optimization community has been using the adjoint method to evaluate the derivatives of the functional. However, the first-order flow sensitivities will be necessary to retrieve the Hessian.

4.1.3 Adjoint Method

Since the residual is assumed to be zero, an augmented Lagrangian with the flow as a constraint can be formed by introducing the adjoint variable $\boldsymbol{\psi}$.

$$L(\mathbf{w},\mathbf{x},\boldsymbol{\psi}) = I(\mathbf{w},\mathbf{x}) + \boldsymbol{\psi}^T\mathbf{R}(\mathbf{w},\mathbf{x}). \tag{4.5}$$

The derivative of the augmented cost function can then be re-arranged to be expressed in terms of flow and metric contributions.

$$\begin{aligned}\frac{dL(\mathbf{w},\mathbf{x})}{d\boldsymbol{\alpha}} &= \frac{dI(\mathbf{w},\mathbf{x})}{d\boldsymbol{\alpha}} + \boldsymbol{\psi}^T\frac{d\mathbf{R}(\mathbf{w},\mathbf{x})}{d\boldsymbol{\alpha}} \\ &= \frac{\partial I}{\partial \mathbf{w}}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \frac{\partial I}{\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \boldsymbol{\psi}^T\left[\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right] \\ &= \underbrace{\left[\frac{\partial I}{\partial \mathbf{w}} + \boldsymbol{\psi}^T\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\right]\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}}_{\text{Flow Contributions}} + \underbrace{\left[\frac{\partial I}{\partial \mathbf{x}} + \boldsymbol{\psi}^T\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}}_{\text{Metric Contributions}}.\end{aligned} \tag{4.6}$$

As previously seen, the cost of computing the flow variable sensitivities is prohibitively large. On the other hand, the grid sensitivities $d\mathbf{x}/d\boldsymbol{\alpha}$ are readily available when using a linear mesh deformation scheme. Therefore, the adjoint variable is defined such that the flow contribution is zero.

$$\mathbf{R}^{\boldsymbol{\psi}} = \frac{\partial I}{\partial \mathbf{w}} + \boldsymbol{\psi}^T\frac{\partial \mathbf{R}}{\partial \mathbf{w}} = \mathbf{0}, \tag{4.7}$$

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\right]^T \boldsymbol{\psi} = -\left[\frac{\partial I}{\partial \mathbf{w}}\right]^T. \tag{4.8}$$

Note that solving for the dual variable $\boldsymbol{\psi}$ requires the solution of a linear system with only one right-hand vector as opposed to $N_{\boldsymbol{\alpha}}$ for the direct differentiation method. Although additional nonlinear constraints require more adjoint solutions, it is uncommon to have less design variables than nonlinear constraints.

After solving for the adjoint variable, the gradient of the augmented Lagrangian is simplified to

$$\frac{dL}{d\boldsymbol{\alpha}} = \left[\frac{\partial I}{\partial \mathbf{x}} + \boldsymbol{\psi}^T\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}, \tag{4.9}$$

such that the metric contributions encompasses the flow contributions through the adjoint variable.

4.1.4 Finite-Difference

The finite-difference method stems from the Taylor series expansion of the objective function. Given an objective function of multiple variables $I = I(\mathbf{x})$ and the standard basis vector $\mathbf{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0)^T$ where 1 is at the ith position, the small forward and backward perturbation h results in

$$I(\mathbf{x} + h\mathbf{e}_i) = \quad I(\mathbf{x}) + h\left.\frac{dI}{dx_i}\right|_{\mathbf{x}} + \frac{1}{2}h^2\left.\frac{d^2I}{dx_i^2}\right|_{\mathbf{x}} + \frac{1}{6}h^3\left.\frac{dI^3}{dx_i^3}\right|_{\mathbf{x}} + O(h^4). \tag{4.10}$$

$$I(\mathbf{x} - h\mathbf{e}_i) = \quad I(\mathbf{x}) - h\left.\frac{dI}{dx_i}\right|_{\mathbf{x}} + \frac{1}{2}h^2\left.\frac{d^2I}{dx_i^2}\right|_{\mathbf{x}} - \frac{1}{6}h^3\left.\frac{dI^3}{dx_i^3}\right|_{\mathbf{x}} + O(h^4). \tag{4.11}$$

By combining Eqs. (4.10 & 4.11) we obtain the second-order finite-difference formulation of the first derivative

$$\left.\frac{dI}{dx_i}\right|_{\mathbf{x}} = \frac{I(\mathbf{x} + h\mathbf{e}_i) - I(\mathbf{x} - h\mathbf{e}_i)}{2h} + O(h^2). \tag{4.12}$$

Additionally, the finite-difference method may also be used to evaluate first-order flow sensitivities.

$$\left.\frac{d\mathbf{w}}{dx_i}\right|_{\mathbf{x}} = \frac{\mathbf{w}(\mathbf{x} + h\mathbf{e}_i) - \mathbf{w}(\mathbf{x} - h\mathbf{e}_i)}{2h} + O(h^2). \tag{4.13}$$

The finite-difference approach is particularly useful when the linear system in Eq. (4.4) is stiff and the flow solver converges well. A framework that evaluates the flow and the adjoint explicitly will not need to set up the linear system in an implicit format. Furthermore, frameworks that solve the flow and the adjoint explicitly will not need to implement the linear system solver. Since the finite-difference perturbations are extremely small, the converged flow solution is a very good initialization for the slightly perturbed design and should converge quickly. As it will be demonstrated later, only an approximation of the first-order flow sensitivities will be required. Therefore, a first-order finite-difference approach may be sufficiently accurate.

4.1.5 Implementation

The current work consistently uses the adjoint approach to retrieve the gradients. All the partial derivatives are derived by hand for the quasi-one-dimensional case. For the three-dimensional case, automatic-differentiation [18] and finite-difference is used to evaluate the Jacobian $\partial \mathbf{R}/\partial \mathbf{w}$, the source term $\partial I/\partial \mathbf{w}$, and the metric sensitivities $\partial I/\partial \mathbf{x}$ and $\partial \mathbf{R}/\partial \mathbf{x}$. The analytical formulation of the the mesh sensitivities $d\mathbf{x}/d\boldsymbol{\alpha}$ is readily available through the employed radial basis function (RBF) mesh deformation scheme [44]. The systems of linear equations (4.4) & (4.8) are solved using PETSc's sparse linear solver GMRES [5, 4, 3].

Additionally, it is worth noting that when the linear system in Eq. (4.4) is solved repetitively for multiple right-hand sides, the solution is initialized with the one previously found. This should be advantageous when subsequent design variables affect the shape similarly.

4.2 Second-Order Sensitivities

This section demonstrates the four possible Hessian formulations based on previous works [51, 38]. The direct-direct, adjoint-adjoint, adjoint-direct, and direct-adjoint labelling refers to the first-order differentiation method in the prefix and second-order differentiation method in the suffix.

4.2.1 Direct-Direct

The direct-direct approach directly differentiates the first-order direct differentiation of the objective function shown in Eq. (4.2) to retrieve the Hessian.

$$\begin{aligned}\frac{d^2 I}{d\boldsymbol{\alpha}^2} = &\left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{w}\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{w}^2}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \frac{\partial I}{\partial \mathbf{w}}\frac{d^2\mathbf{w}}{d\boldsymbol{\alpha}^2} \\ &+ \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{x}\partial \mathbf{w}}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{x}^2}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \frac{\partial I}{\partial \mathbf{x}}\frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2}.\end{aligned} \tag{4.14}$$

As expected, the expensive first-order flow sensitivities $d\mathbf{w}/d\boldsymbol{\alpha}$ are still present, requiring $N_{\boldsymbol{\alpha}}$ linear solutions from Eq. (4.3). However, the appearance of the second-order flow sensitivities $d^2\mathbf{w}/d\boldsymbol{\alpha}^2$ aggravates the situation. This new term can be solved by directly differentiating the flow equations twice.

$$\begin{aligned}\frac{d^2 \mathbf{R}}{d\boldsymbol{\alpha}^2} = &\left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}^2}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \frac{\partial \mathbf{R}}{\partial \mathbf{w}}\frac{d^2\mathbf{w}}{d\boldsymbol{\alpha}^2} \\ &+ \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{x}\partial \mathbf{w}}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{x}^2}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2} = \mathbf{0},\end{aligned} \tag{4.15}$$

$$\begin{aligned}\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\frac{d^2\mathbf{w}}{d\boldsymbol{\alpha}^2} = -&\left(\left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}^2}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right. \\ &\left. + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{x}\partial \mathbf{w}}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{x}^2}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2}\right).\end{aligned} \tag{4.16}$$

Notice that Eq. (4.16) requires the solution of a large system with $(N_{\boldsymbol{\alpha}}(N_{\boldsymbol{\alpha}}+1)/2)$ right-hand sides. The direct-direct method requires a total of $(N_{\boldsymbol{\alpha}} + N_{\boldsymbol{\alpha}}(N_{\boldsymbol{\alpha}}+1)/2)$ large linear system solutions, which is quadratically proportional to the number of design variables.

4.2.2 Adjoint-Direct

The adjoint-direct method starts with the gradient of the augmented cost function derived from the adjoint method in Eq. (4.9) and is then directly differentiated.

$$\begin{aligned}\frac{d^2L}{d\boldsymbol{\alpha}^2} = &\left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{x}^2}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{x}\partial \mathbf{w}}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \frac{\partial I}{\partial \mathbf{x}}\frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2}\\ &+ \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{d\boldsymbol{\psi}^T}{d\boldsymbol{\alpha}} + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \left(\boldsymbol{\psi}^T\frac{\partial^2 \mathbf{R}}{\partial \mathbf{x}^2}\right)\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\\ &+ \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \left(\boldsymbol{\psi}^T\frac{\partial^2 \mathbf{R}}{\partial \mathbf{x}\partial \mathbf{w}}\right)\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \boldsymbol{\psi}^T\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2}.\end{aligned} \tag{4.17}$$

The first order flow sensitivities $d\mathbf{w}/d\boldsymbol{\alpha}$ are evaluated from Eq. (4.4) and require $N_{\boldsymbol{\alpha}}$ system evaluations. The adjoint variable is evaluated in Eq. (4.8) at the cost of one system evaluation.

That leaves $d\boldsymbol{\psi}^T/d\boldsymbol{\alpha}$ to be evaluated. The term can be found in the direct differentiation of the adjoint residual in Eq. (4.7)

$$\begin{aligned}\frac{d\mathbf{R}^{\psi}}{d\boldsymbol{\alpha}} =&\frac{\partial^2 I}{\partial \mathbf{w}\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \frac{\partial^2 I}{\partial \mathbf{w}^2}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\right]^T \frac{d\boldsymbol{\psi}^T}{d\boldsymbol{\alpha}}\\ &+ \boldsymbol{\psi}^T\frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \boldsymbol{\psi}^T\frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}^2}\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\\ =&\left[\frac{\partial^2 I}{\partial \mathbf{w}\partial \mathbf{x}} + \boldsymbol{\psi}^T\frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}\partial \mathbf{x}}\right]\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\\ &+ \left[\frac{\partial^2 I}{\partial \mathbf{w}^2} + \boldsymbol{\psi}^T\frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}^2}\right]\frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\right]^T \frac{d\boldsymbol{\psi}^T}{d\boldsymbol{\alpha}} = \mathbf{0}.\end{aligned} \tag{4.18}$$

Equation (4.18) can then be rearranged and the solution of the adjoint sensitivity with respect to the design variables is defined by

$$\begin{aligned}\left[\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\right]^T \frac{d\boldsymbol{\psi}^T}{d\boldsymbol{\alpha}} =& -\left[\frac{\partial^2 I}{\partial \mathbf{w}\partial \mathbf{x}} + \boldsymbol{\psi}^T\frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}\partial \mathbf{x}}\right]\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\\ &- \left[\frac{\partial^2 I}{\partial \mathbf{w}^2} + \boldsymbol{\psi}^T\frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}^2}\right]\frac{d\mathbf{w}}{d\boldsymbol{\alpha}},\end{aligned} \tag{4.19}$$

and requires the solution to $N_{\boldsymbol{\alpha}}$ right-hand sides. A total of $(2N_{\boldsymbol{\alpha}} + 1)$ system evaluations are required to compute the Hessian using the adjoint-direct approach. This approach is commonly used in conjunction with truncated-Newton methods [35, 19] since the matrix-vector product of the adjoint-direct Hessian with the search direction only requires two linear system evaluations.

4.2.3 Adjoint-Adjoint

The adjoint-adjoint method is first derived based on the work of Papadimitriou and Giannakoglou [38]. A simplification follows after determining the equivalence between the first-order flow sensitivities and one of the adjoint variables. Looking at the previous method, it is possible to add adjoint variables to eliminate some expensive terms by augmenting the derivative of the augmented cost function.

$$\frac{d^2 L^*}{d\boldsymbol{\alpha}^2} = \frac{d^2 L}{d\boldsymbol{\alpha}^2} + \boldsymbol{\beta}^T \frac{d\mathbf{R}}{d\boldsymbol{\alpha}} + \boldsymbol{\gamma}^T \frac{d\mathbf{R}^{\boldsymbol{\psi}}}{d\boldsymbol{\alpha}}. \tag{4.20}$$

The expression can be fully expanded by inserting equations (4.3), (4.17), and (4.18) into Eq. (4.20) giving the adjoint-adjoint form of the Hessian.

$$\begin{aligned}
\frac{d^2 L^*}{d\boldsymbol{\alpha}^2} =& \frac{\partial I}{\partial \mathbf{x}} \frac{d^2 \mathbf{x}}{d\boldsymbol{\alpha}^2} + \boldsymbol{\psi}^T \frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{d^2 \mathbf{x}}{d\boldsymbol{\alpha}^2} \\
& + \left\{ \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{x}^2} + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \left(\boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{x}^2}\right) + \boldsymbol{\beta}^T \frac{\partial \mathbf{R}}{\partial \mathbf{x}} \right. \\
& \qquad \left. + \boldsymbol{\gamma}^T \frac{\partial^2 I}{\partial \mathbf{w} \partial \mathbf{x}} + \boldsymbol{\gamma}^T \boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{w} \partial \mathbf{x}} \right\} \frac{d\mathbf{x}}{d\boldsymbol{\alpha}} \\
& + \left\{ \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{x} \partial \mathbf{w}} + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \left(\boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{x} \partial \mathbf{w}}\right) \right. \\
& \qquad \left. + \boldsymbol{\beta}^T \frac{\partial \mathbf{R}}{\partial \mathbf{w}} + \boldsymbol{\gamma}^T \frac{\partial^2 I}{\partial \mathbf{w}^2} + \boldsymbol{\gamma}^T \left(\boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}^2}\right) \right\} \frac{d\mathbf{w}}{d\boldsymbol{\alpha}} \\
& + \left\{ \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T + \boldsymbol{\gamma}^T \left[\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\right]^T \right\} \frac{d\boldsymbol{\psi}^T}{d\boldsymbol{\alpha}}.
\end{aligned} \tag{4.21}$$

The derivative of the adjoint can be eliminated by defining $\boldsymbol{\gamma}$ to zero-out the last parenthesis. Unfortunately, as seen in Eq. (4.22), the cost of eliminating $d\boldsymbol{\psi}^T/d\boldsymbol{\alpha}$ is the same as solving for it. In fact, $\boldsymbol{\gamma}$ exactly represents the first-order flow sensitivities when comparing the linear systems (4.4) and (4.22). Evaluating $\boldsymbol{\gamma}$ costs $N_{\boldsymbol{\alpha}}$ system evaluations.

$$\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\boldsymbol{\gamma} = -\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}. \tag{4.22}$$

The same unfortunate scenario happens for $d\mathbf{w}/d\boldsymbol{\alpha}$ where eliminating it costs $N_{\boldsymbol{\alpha}}$ system evaluations as seen in Eq. (4.23).

$$\frac{\partial \mathbf{R}}{\partial \mathbf{w}}^T \boldsymbol{\beta} = -\left\{\left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{x}\partial \mathbf{w}} + ...\right\}^T. \tag{4.23}$$

Finally, the adjoint-adjoint Hessian after solving for all the adjoint variables reduces to

$$\begin{aligned}\frac{d^2 L_a^*}{d\boldsymbol{\alpha}^2} =& \frac{\partial L}{\partial \mathbf{x}}\frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2} + \boldsymbol{\psi}^T\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2} \\ &+ \left\{\left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 L}{\partial \mathbf{x}^2} + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \left(\boldsymbol{\psi}^T\frac{\partial^2 \mathbf{R}}{\partial \mathbf{x}^2}\right) + \boldsymbol{\beta}^T\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right. \\ &\left. + \frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\boldsymbol{\gamma}^T\frac{\partial^2 L}{\partial \mathbf{w}\partial \mathbf{x}} + \boldsymbol{\gamma}^T\boldsymbol{\psi}^T\frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}\partial \mathbf{x}}\right\}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}.\end{aligned} \tag{4.24}$$

A total of $(2N_{\boldsymbol{\alpha}}+1)$ system evaluations are required to solve every adjoint. Note that the adjoint $\boldsymbol{\beta}$ was used to eliminate the first-order flow sensitivities when they were in fact computed as $\boldsymbol{\gamma}$. Therefore, augmenting the Hessian with the $\boldsymbol{\beta}$ adjoint in Eq. (4.20) was unnecessary and can be reformulated as the simpler augmented cost function

$$\frac{d^2 L^*}{d\boldsymbol{\alpha}^2} = \frac{d^2 L}{d\boldsymbol{\alpha}^2} + \boldsymbol{\gamma}^T\frac{d\mathbf{R}^{\psi}}{d\boldsymbol{\alpha}} = \frac{d^2 L}{d\boldsymbol{\alpha}^2} + \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T\frac{d\mathbf{R}^{\psi}}{d\boldsymbol{\alpha}}, \tag{4.25}$$

which leads to the following Hessian formulation

$$\begin{aligned}\frac{d^2 L^*}{d\boldsymbol{\alpha}^2} =& \frac{\partial I}{\partial \mathbf{x}}\frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2} + \boldsymbol{\psi}^T \frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2} \\ &+ \left\{ \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{x}^2} + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \left(\boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{x}^2}\right)\right. \\ &\quad \left. + \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{w}\partial \mathbf{x}} + \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}\partial \mathbf{x}} \right\} \frac{d\mathbf{x}}{d\boldsymbol{\alpha}} \\ &+ \left\{ \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{x}\partial \mathbf{w}} + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \left(\boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{x}\partial \mathbf{w}}\right)\right. \\ &\quad \left. + \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial \mathbf{w}^2} + \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \left(\boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial \mathbf{w}^2}\right) \right\} \frac{d\mathbf{w}}{d\boldsymbol{\alpha}} \\ &+ \left\{ \cancelto{0}{\left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T + \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \left[\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\right]^T} \right\} \frac{d\boldsymbol{\psi}^T}{d\boldsymbol{\alpha}}. \end{aligned} \tag{4.26}$$

where the sensitivity of the adjoint does not need to be evaluated since it is multiplied by Eq. (4.3), which equals zero. A total of $(N_{\boldsymbol{\alpha}} + 1)$ linear system solutions are required. This formulation turns out to be exactly the same as the direct-adjoint approach, presented in the next subsection.

4.2.4 Direct-Adjoint

The direct-adjoint method is closely related to the direct-direct approach and gives the same final adjoint-adjoint formulation through simpler derivations. The Hessian of the cost function shown in Eq. (4.14) is augmented with the Hessian of the flow equation shown in Eq. (4.15) leading to the following formulation

$$\frac{d^2 L^{**}}{d\boldsymbol{\alpha}^2} = \frac{d^2 L}{d\boldsymbol{\alpha}^2} + \boldsymbol{\psi}^T \frac{d^2 \mathbf{R}}{d\boldsymbol{\alpha}^2}, \tag{4.27}$$

$$
\begin{aligned}
\frac{d^2L^{**}}{d\boldsymbol{\alpha}^2} = &\left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \left\{\frac{\partial^2 L}{\partial\mathbf{w}\partial\mathbf{x}} + \boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial\mathbf{w}\partial\mathbf{x}}\right\} \frac{d\mathbf{x}}{d\boldsymbol{\alpha}} \\
&+ \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]^T \left\{\frac{\partial^2 L}{\partial\mathbf{w}^2} + \boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial\mathbf{w}^2}\right\} \frac{d\mathbf{w}}{d\boldsymbol{\alpha}} + \left\{\frac{\partial L}{\partial\mathbf{w}} + \boldsymbol{\psi}^T \frac{\partial \mathbf{R}}{\partial\mathbf{w}}\right\} \frac{d^2\mathbf{w}}{d\boldsymbol{\alpha}^2} \\
&+ \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \left\{\frac{\partial^2 L}{\partial\mathbf{x}\partial\mathbf{w}} + \boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial\mathbf{x}\partial\mathbf{w}}\right\} \frac{d\mathbf{w}}{d\boldsymbol{\alpha}} \\
&+ \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \left\{\frac{\partial^2 L}{\partial\mathbf{x}^2} + \boldsymbol{\psi}^T \frac{\partial^2 \mathbf{R}}{\partial\mathbf{x}^2}\right\} \frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \left\{\frac{\partial L}{\partial\mathbf{x}} + \boldsymbol{\psi}^T \frac{\partial \mathbf{R}}{\partial\mathbf{x}}\right\} \frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2}.
\end{aligned}
\tag{4.28}
$$

The adjoint is used to eliminate the second-order flow sensitivities such that the third term of Eq. (4.28) vanishes. Coincidentally, the adjoint variable is solved through the same linear system as if we attempted to eliminate the first-order flow equations, requiring the solution to one linear system. The first-order flow sensitivities still have to be solved at the cost of $N_{\boldsymbol{\alpha}}$ from Eq. (4.4). Therefore, the total cost of $(N_{\boldsymbol{\alpha}} + 1)$ system evaluations are required, making the direct-adjoint the most inexpensive method.

4.2.5 Finite-Difference

Once again, the finite-difference method is a useful tool to validate results obtained analytically. Given the same objective function I and standard basis vector $\mathbf{e}_i$ as in section (4.1.4), the Taylor series expansion of the twice perturbed objective function is

$$
\begin{aligned}
I(\mathbf{x} + h\mathbf{e}_i + h\mathbf{e}_j) = \quad & I(\mathbf{x}) + h\left.\frac{dI}{dx_i}\right|_{\mathbf{x}} + h\left.\frac{dI}{dx_j}\right|_{\mathbf{x}} \\
& + \frac{1}{2}h^2\left.\frac{d^2I}{dx_i^2}\right|_{\mathbf{x}} + \frac{1}{2}h^2\left.\frac{d^2I}{dx_j^2}\right|_{\mathbf{x}} \\
& + \frac{1}{2}h^2\left.\frac{d^2I}{dx_i dx_j}\right|_{\mathbf{x}} + \frac{1}{2}h^2\left.\frac{d^2I}{dx_j dx_i}\right|_{\mathbf{x}} \\
& + \frac{1}{6}h^3\left.\frac{d^3I}{dx_i^3}\right|_{\mathbf{x}} + \frac{1}{6}h^3\left.\frac{d^3I}{dx_i^2 dx_j}\right|_{\mathbf{x}} + \frac{1}{6}h^3\left.\frac{d^3I}{dx_i^1 dx_j^2}\right|_{\mathbf{x}} \\
& + \frac{1}{6}h^3\left.\frac{d^3I}{dx_j^3}\right|_{\mathbf{x}} + \frac{1}{6}h^3\left.\frac{d^3I}{dx_j^2 dx_i}\right|_{\mathbf{x}} + \frac{1}{6}h^3\left.\frac{d^3I}{dx_j^1 dx_i^2}\right|_{\mathbf{x}} \\
& + O(h^4).
\end{aligned}
\tag{4.29}
$$

By combining Eq. (4.29) at $I(\mathbf{x}+h\mathbf{e}_i+h\mathbf{e}_j)$, $I(\mathbf{x}+h\mathbf{e}_i-h\mathbf{e}_j)$, $I(\mathbf{x}-h\mathbf{e}_i+h\mathbf{e}_j)$, and $I(\mathbf{x}-h\mathbf{e}_i-h\mathbf{e}_j)$, we obtain the central difference finite-difference formulation of the second-order derivatives

$$\left.\frac{d^2I}{dx_idx_j}\right|_{\mathbf{x}} = \frac{\begin{array}{c} I(\mathbf{x}+h\mathbf{e}_i+h\mathbf{e}_j) - I(\mathbf{x}+h\mathbf{e}_i-h\mathbf{e}_j) \\ - I(\mathbf{x}-h\mathbf{e}_i+h\mathbf{e}_j) + I(\mathbf{x}-h\mathbf{e}_i-h\mathbf{e}_j) \end{array}}{4h^2} + O(h^4). \tag{4.30}$$

A total of $4n^2/2+2n$ flow solutions are required to evaluate the Hessian.

Since the Hessian is the change in first-order derivatives, it is also possible to evaluate a full column at a time through finite-difference of the gradients. An off-diagonal term is evaluated as an average of itself and its symmetric counterpart to ensure a symmetric Hessian.

$$\left.\frac{d^2I}{dx_idx_j}\right|_{\mathbf{x}} = \frac{(\mathbf{g_i}(\mathbf{x}+h\mathbf{e}_j) - \mathbf{g_i}(\mathbf{x}-h\mathbf{e}_j)) + (\mathbf{g_j}(\mathbf{x}+h\mathbf{e}_i) - \mathbf{g_j}(\mathbf{x}-h\mathbf{e}_i))}{2(2h)} + O(h^2). \tag{4.31}$$

A total of $(2N_{\boldsymbol{\alpha}})$ gradient evaluations are required. When the gradient is evaluated through the adjoint, every perturbed gradient call will require a flow solution and an adjoint solution. Assuming that the adjoint requires the same computational effort as the flow, a total of $(4N_{\boldsymbol{\alpha}})$ function calls are required for the objective Hessian plus an additional $(2N_{\boldsymbol{\alpha}})$ adjoints per non-linear constraint Hessian.

Chapter 5

Approximate Hessian

The novel aspect of this work is the approximation of the Hessian through partial solutions of the first-order flow sensitivities $d\mathbf{w}/d\boldsymbol{\alpha}$. Additionally, the implementation complexity is reduced by neglecting second-order flow sensitivities $d^2\mathbf{w}/d\boldsymbol{\alpha}^2$.

5.1 Direct-Adjoint Approximation

Although the direct-adjoint analytical Hessian is the least expensive to compute, it is still costly due to the first-order flow sensitivities $d\mathbf{w}/d\boldsymbol{\alpha}$. The first-order flow sensitivities are solved iteratively using GMRES and their accuracy depends on the relative residual η_k

$$\mathbf{r}_k = \frac{\partial \mathbf{R}}{\partial \mathbf{w}} \left[\frac{d\mathbf{w}}{d\alpha_k}\right]_n + \frac{\partial \mathbf{R}}{\partial \mathbf{x}} \frac{d\mathbf{x}}{d\alpha_k}, \tag{5.1}$$

$$\eta_k = \frac{\|\mathbf{r}_k\|}{\left\| \dfrac{\partial \mathbf{R}}{\partial \mathbf{x}} \dfrac{d\mathbf{x}}{d\alpha_k} \right\|}, \tag{5.2}$$

where $[d\mathbf{w}/d\alpha_k]_n$ is the solution at iteration n.

The exact Hessian is retrieved when the relative residual reaches machine precision.

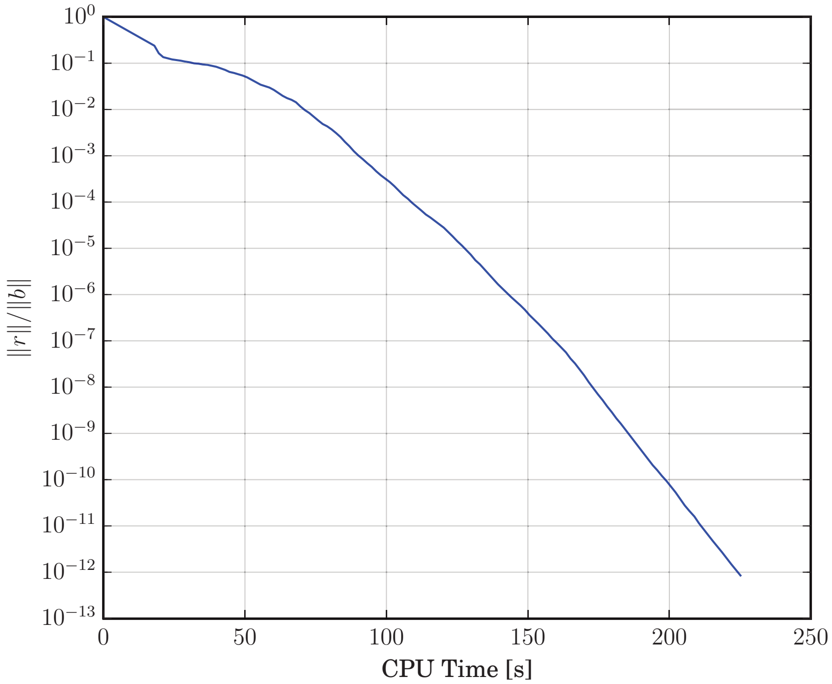


Figure 5.1 – First-order flow sensitivity convergence with GMRES

As seen in Fig. (5.1), computational time is proportional to the desired accuracy. The proposed method sets the iterative convergence criteria to a relative residual of $\eta_k = 1.0\text{E-}01$. Therefore, the cost of retrieving this approximate Hessian is only a fraction of recovering the exact one. From this approximation, the retrieved first-order sensitivities will only have a single digit of accuracy. The idea to approximate the Hessian stems from the fact that superlinear convergence does not require exact Hessian evaluations as it will be demonstrated. A linear relationship between the error incurred by the approximation has been observed and will be demonstrated in the following section.

Another cost of the direct-adjoint approach is the implementation time of the sparse tensors that appear in its formulation. More specifically, the partial second-order sensitivities of the residual ($\partial^2\mathbf{R}/\partial\cdot\partial\cdot$) need to be evaluated and stored efficiently. Papadimitriou and Giannakoglou [38] have already attempted to simplify the process by neglecting the second-order residual sensitivities $\partial^2\mathbf{R}/\partial\mathbf{w}^2$ and $\partial^2\mathbf{R}/\partial\mathbf{w}\partial\mathbf{x}$. However, the second-order residual sensitivities $\partial^2\mathbf{R}/\partial\mathbf{x}^2$ still need to be evaluated. The next section offers an alternative to ease the implementation.

5.2 Direct-Direct Approximation

In order to avoid the implementation complexity of tensors, we take another look at the most expensive method to compute the analytical Hessian: the direct-direct method. Once again, the first-order flow sensitivities are approximated through partial solves. Additionally, the second-order flow sensitivities $\partial^2\mathbf{w}/\partial\boldsymbol{\alpha}^2$ are neglected. The direct-direct method is reduced to

$$\begin{aligned}\frac{d^2I}{d\boldsymbol{\alpha}^2} = &\left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]_n^T \frac{\partial^2 I}{\partial\mathbf{w}\partial\mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]_n^T \frac{\partial^2 I}{\partial\mathbf{w}^2}\left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]_n \\ &+ \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial\mathbf{x}\partial\mathbf{w}}\left[\frac{d\mathbf{w}}{d\boldsymbol{\alpha}}\right]_n + \left[\frac{d\mathbf{x}}{d\boldsymbol{\alpha}}\right]^T \frac{\partial^2 I}{\partial\mathbf{x}^2}\frac{d\mathbf{x}}{d\boldsymbol{\alpha}} + \frac{\partial I}{\partial\mathbf{x}}\frac{d^2\mathbf{x}}{d\boldsymbol{\alpha}^2}.\end{aligned} \tag{5.3}$$

As a result, the approximate direct-direct method computational cost is a fraction of solving $N_{\boldsymbol{\alpha}}$ linear systems. Although it is similar in computational effort to the direct-adjoint approach, its implementation is more straightforward since sparse tensors, such as $\partial^2\mathbf{R}/\partial\mathbf{w}^2$, $\partial^2\mathbf{R}/\partial\mathbf{w}\partial\mathbf{x}$, and $\partial^2\mathbf{R}/\partial\mathbf{x}^2$ do not appear in the direct-direct formulation. Given that most ASO frameworks already support the adjoint method, which requires the Jacobian $\partial\mathbf{R}/\partial\mathbf{w}$, the first-order flow sensitivities can easily be computed by changing the right-hand side.

The only new terms are the second-order partials with respect to the objective function and the second-order mesh sensitivities. The partial derivatives of the objective function are usually readily available, whereas the second-order mesh sensitivities vanish for linear mesh deformation schemes. This approach is used for the three-dimensional case due to its ease of implementation.

The impact of neglecting $d^2\mathbf{w}/d\boldsymbol{\alpha}^2$ will depend on the problem and its contribution is expected to be greater for highly nonlinear design spaces. The test cases will seek to demonstrate that the contribution of this term is negligible.

5.3 Implementation

The first-order sensitivities are evaluated as shown in section (4.1). The second-order partial derivatives of the cost function and the nonlinear constraints are evaluated through AD [18] or finite-difference.

The size of the first-order flow sensitivities and first-order metric sensitivities can grow quite fast since they may be dense matrices. Although the second-order partial sensitivities of the cost function are sparse matrices, it is possible to avoid assembling them by multiplying each entry with the corresponding row and column of the adjacent matrices. This procedure is a sum of outer-products, which has to be performed as many times as there are entries in the functional sensitivity matrix. The result is a trade-off between computational time and memory usage. In fact, evaluating the residual sensitivity tensors would leave the user the choice between evaluating an intractable sum of outer-products, or storing a very large amount of data on memory. Overall, this second approximation is useful to reduce both the implementation cost and the computational requirements.

5.4 Hessian Positive-Definite Modification

Although the Hessian entries inform us about the design space, its eigendecomposition offers better insights about the curvature. The Hessian eigenvalues represent the principal magnitudes of the design space curvature, while the eigenvectors define the corresponding directions. At every design cycle, the search direction $\mathbf{p}$ is a function of the Hessian $\mathbf{H}$ and the gradient $\mathbf{g}$.

$$\begin{aligned} \mathbf{Hp} &= -\mathbf{g} \\ \mathbf{Q\Lambda Q^{-1}p} &= -\mathbf{g} \\ \mathbf{p} &= -\mathbf{Q\Lambda^{-1}Q^{-1}g}, \end{aligned} \tag{5.4}$$

where $\mathbf{\Lambda}$ is the diagonal matrix containing the eigenvalues of $\mathbf{H}$ and $\mathbf{Q}$ is the matrix of eigenvectors of $\mathbf{H}$. The matrix of eigenvectors $\mathbf{Q}$ is orthonormal and represents a unitary transformation that rotates the gradient vector $\mathbf{g}$ without scaling it. Therefore, the eigenvalues in $\mathbf{\Lambda}^{-1}$ dictate how to scale the gradient vector to find the optimal search direction. Since we know that the gradient vector represents a direction of descent, a negative eigenvalue will flip this direction in a direction of ascent. Additionally, small eigenvalues of $\mathbf{H}$ result in larger steps and large eigenvalues result in smaller steps.

A disadvantage of Newton's method is that the exact Newton step does not guarantee convergence. In fact, the search direction is a descent direction only if the Hessian is positive-definite, which represents convexity. Therefore, whether the exact Hessian or the approximate Hessian is evaluated, a mechanism must ensure positive-definiteness.

When the Hessian is evaluated, an eigenvalue decomposition is performed and the eigenvalues must be modified such that they are all positive. A common method to guarantee positive-definiteness is to shift the eigenvalues of $\mathbf{H}$ such that the smallest eigenvalue is greater than some positive value ϵ defined by the user [32].

$$\mathbf{H}^{+} = \mathbf{H} + \mathbf{I} \cdot \max(0, \epsilon - \lambda_{\min}(\mathbf{H})). \tag{5.5}$$

However, large negative eigenvalues would require a large modification, where the Hessian would lose the second-order information.

The utilized method evaluates the geometric mean of the positive eigenvalues, which translates to the average magnitude of the positive eigenvalues. The negative eigenvalues are then replaced by this average positive eigenvalue and the Hessian is reassembled, where it produces a search direction that is neither greater nor lower than all its positive counterparts. Given the cardinality n of the set of positive eigenvalues of the Hessian, the eigenvalues $\boldsymbol{\lambda}$ are modified such that

$$\lambda_k = \begin{cases} \lambda_k, & \text{if } \lambda_k > 0, \\ \sqrt[n]{\prod_{i:\lambda_i>0} \lambda_i}, & \text{otherwise.} \end{cases} \tag{5.6}$$

For added stability, the threshold to modify the eigenvalues may be changed to a multiple of the maximum eigenvalue, such that tiny eigenvalues do not disproportionally dictate the search direction.

$$\lambda_k = \begin{cases} \lambda_k, & \text{if } \lambda_k > \epsilon\lambda_{\text{max}}, \\ \sqrt[n]{\prod_{i:\lambda_i>0} \lambda_i} & \text{otherwise.} \end{cases} \tag{5.7}$$

Since, the cost of added stability results in a decreased convergence rate, it is suggested to modify this parameter only when the optimization process stalls.

5.5 Error Bounds

The error incurred by the first-order flow sensitivity approximation is bounded for the Hessian, the search direction and the Hessian eigenvalues. A shorthand notation is introduced to simplify the derivations. The direct-direct formulation of the entry (i, j) of the Hessian is shown in Eq. (5.8).

$$\begin{aligned} h_{i,j} = \frac{d^2 I}{d\alpha_i d\alpha_j} &= \left[\frac{d\mathbf{w}}{d\alpha_i}\right]^T \frac{\partial^2 I}{\partial \mathbf{w} \partial \mathbf{x}} \frac{d\mathbf{x}}{d\alpha_j} + \left[\frac{d\mathbf{w}}{d\alpha_i}\right]^T \frac{\partial^2 I}{\partial \mathbf{w}^2} \frac{d\mathbf{w}}{d\alpha_j} \\ &+ \left[\frac{d\mathbf{x}}{d\alpha_i}\right]^T \frac{\partial^2 I}{\partial \mathbf{x} \partial \mathbf{w}} \frac{d\mathbf{w}}{d\alpha_j} + \frac{\partial I}{\partial \mathbf{w}} \frac{d^2 \mathbf{w}}{d\alpha_i d\alpha_j} \\ &+ \left[\frac{d\mathbf{x}}{d\alpha_i}\right]^T \frac{\partial^2 I}{\partial \mathbf{x}^2} \frac{d\mathbf{x}}{d\alpha_j} + \frac{\partial I}{\partial \mathbf{x}} \frac{d^2 \mathbf{x}}{d\alpha_i d\alpha_j}. \end{aligned} \tag{5.8}$$

Let us define a simpler notation where,

$$\begin{aligned} \mathbf{z}_i &= \frac{d\mathbf{w}}{d\alpha_i}, \\ \hat{\mathbf{z}}_i &= \left[\frac{d\mathbf{w}}{d\alpha_i}\right]_n = \mathbf{z}_i(1+\eta), \\ \mathbf{m}_i &= \frac{\partial^2 I}{\partial \mathbf{w} \partial \mathbf{x}} \frac{d\mathbf{x}}{d\alpha_i}, \\ \mathbf{N} &= \frac{\partial^2 I}{\partial \mathbf{w}^2} = \mathbf{N}^T, \\ p_{i,j} &= \frac{\partial I}{\partial \mathbf{w}} \frac{d^2 \mathbf{w}}{d\alpha_i d\alpha_j} + \left[\frac{d\mathbf{x}}{d\alpha_i}\right]^T \frac{\partial^2 I}{\partial \mathbf{x}^2} \frac{d\mathbf{x}}{d\alpha_j} + \frac{\partial I}{\partial \mathbf{x}} \frac{d^2 \mathbf{x}}{d\alpha_i d\alpha_j}, \end{aligned}$$

simplifies the Hessian expression to

$$h_{i,j} = h_{i,j}(\mathbf{z}_i, \mathbf{z}_j) = \mathbf{z}_i^T \mathbf{m}_j + \mathbf{z}_i^T \mathbf{N} \mathbf{z}_j + \mathbf{m}_i^T \mathbf{z}_j + p_{i,j}. \tag{5.9}$$

5.5.1 Hessian Entry

A Taylor series of the Hessian entry perturbed by a relative error η is taken about the exact first-order flow sensitivities

$$\begin{aligned} \hat{h}_{i,j} = h_{i,j}(\hat{\mathbf{z}}_i, \hat{\mathbf{z}}_j) &= h_{i,j}(\mathbf{z}_i + \eta \mathbf{z}_i, \mathbf{z}_j + \eta \mathbf{z}_j) \\ &= h_{i,j}(\mathbf{z}_i, \mathbf{z}_j) + \eta \mathbf{z}_i^T \frac{\partial h_{i,j}}{\partial \mathbf{z}_i} + \eta \mathbf{z}_j^T \frac{\partial h_{i,j}}{\partial \mathbf{z}_j} + \mathcal{O}(\eta^2). \end{aligned} \tag{5.10}$$

Using the Matrix Cookbook as a reference [40] for derivatives of matrices and vectors, the derivatives are evaluated as follow:

$$\begin{aligned}\frac{\partial h_{i,j}}{\partial \mathbf{z}_i} &= \frac{\partial(\mathbf{z}_i^T\mathbf{m}_j)}{\partial \mathbf{z}_i} + \frac{\partial(\mathbf{z}_i^T\mathbf{N}\mathbf{z}_j)}{\partial \mathbf{z}_i} + \frac{\partial(\mathbf{m}_i^T\mathbf{z}_j)}{\partial \mathbf{z}_i} + \frac{\partial(p_{i,j})}{\partial \mathbf{z}_i} \\ &= \mathbf{m}_j + \mathbf{N}\mathbf{z}_j + \vec{\mathbf{0}} + \vec{\mathbf{0}},\end{aligned} \tag{5.11}$$

$$\begin{aligned}\frac{\partial h_{i,j}}{\partial \mathbf{z}_j} &= \frac{\partial(\mathbf{z}_i^T\mathbf{m}_j)}{\partial \mathbf{z}_j} + \frac{\partial(\mathbf{z}_i^T\mathbf{N}\mathbf{z}_j)}{\partial \mathbf{z}_j} + \frac{\partial(\mathbf{m}_i^T\mathbf{z}_j)}{\partial \mathbf{z}_j} + \frac{\partial(p_{i,j})}{\partial \mathbf{z}_j} \\ &= \vec{\mathbf{0}} + \mathbf{N}\mathbf{z}_i + \mathbf{m}_i + \vec{\mathbf{0}}.\end{aligned} \tag{5.12}$$

$$\hat{h}_{i,j} = h_{i,j}(\mathbf{z}_i, \mathbf{z}_j) + \eta(\mathbf{z}_i^T\mathbf{m}_j + \mathbf{z}_i^T\mathbf{N}\mathbf{z}_j + \mathbf{z}_j^T\mathbf{N}\mathbf{z}_i + \mathbf{z}_j^T\mathbf{m}_i) \tag{5.13}$$

Given a target maximum relative error t of the entry $h_{i,j}$, it is possible to choose a relative error η of the first-order flow sensitivities to satisfy

$$\left|\frac{\hat{h}_{i,j} - h_{i,j}}{h_{i,j}}\right| \leq t. \tag{5.14}$$

$$\eta \leq t\frac{|a_{i,j} + p_{i,j}|}{|a_{i,j} + \mathbf{z}_i^T\mathbf{N}\mathbf{z}_j)|}, \quad \text{where } a_{i,j} = \mathbf{z}_i^T\mathbf{m}_j + \mathbf{z}_i^T\mathbf{N}\mathbf{z}_j + \mathbf{z}_j^T\mathbf{m}_i. \tag{5.15}$$

Note that the convergence test would require the evaluation of $a_{i,j}$, $p_{i,j}$, and $\mathbf{z}_i^T N\mathbf{z}_j$ at every GMRES iteration. Instead, using

$$\eta \leq t, \tag{5.16}$$

as a simpler convergence test may be equivalent in multiple situations. First, if

$$\frac{|a_{i,j} + p_{i,j}|}{|a_{i,j} + \mathbf{z}_i^T\mathbf{N}\mathbf{z}_j)|} \geq 1, \tag{5.17}$$

the simpler convergence test always insures that the more complex one is satisfied. Second, if the magnitude of $a_{i,j}$ is much greater than $p_{i,j}$ and $\mathbf{z}_i^T\mathbf{N}\mathbf{z}_j$, the approximation

$$|a_{i,j} + p_{i,j}| \approx |a_{i,j} + \mathbf{z}_i^T\mathbf{N}\mathbf{z}_j|, \tag{5.18}$$

is valid and the simple convergence test holds true for

$$\eta(1-\epsilon) \leq t, \tag{5.19}$$

where ϵ is small. In practice, the simpler convergence test is used with a safety margin.

An additional issue arises in practice since the exact solution is not known. It is not possible to determine the exact relative error of the solution η. Therefore, the residual $\|A\mathbf{z}-\mathbf{b}\|/\|\mathbf{b}\|$ is used since it is proportional to the solution error and the condition number of A.

5.5.2 Hessian Matrix

Now that every entry $h_{i,j}$ is bounded by a target relative error t, it is possible to represent a perturbed Hessian by a matrix multiplication

$$\hat{\mathbf{H}} = (1+t)\mathbf{I}\mathbf{H} = (\mathbf{I}+\mathbf{E_r})\mathbf{H}, \quad \text{where } \mathbf{E_r} = t\mathbf{I}. \tag{5.20}$$

5.5.3 Search Direction

During a quadractic programming step, we solve

$$\mathbf{H}_k\mathbf{p}_k = -\mathbf{g}_k, \tag{5.21}$$

where $\mathbf{p}_k$ is the search direction and $\mathbf{g}_k$ is the gradient at step k. However, only the approximation of the Hessian is available. We now seek to bound the error on the search direction by using perturbation analysis. Since we are only interested in the perturbation due to partial solutions of the Hessian, we do not take into account the conditioning of the Hessian or numerical errors of the right-hand-side.

$$\begin{aligned}
\hat{\mathbf{H}}_k \mathbf{p}_k(\mathbf{E_r}) &= -\mathbf{g}_k, \quad \text{where} \quad \mathbf{p}_k(\mathbf{0}) = \mathbf{p}_k \\
(\mathbf{H}_k + \mathbf{E_r}\mathbf{H}_k)\mathbf{p}_k(\mathbf{E_r}) &= -\mathbf{g}_k \\
\mathbf{H}_k^{-1}(\mathbf{H}_k + \mathbf{E_r}\mathbf{H}_k)\mathbf{p}_k(\mathbf{E_r}) &= -\mathbf{H}_k^{-1}\mathbf{g}_k \\
(\mathbf{I} + \mathbf{E_r}\mathbf{I})\mathbf{p}_k(\mathbf{E_r}) &= \mathbf{p}_k.
\end{aligned} \tag{5.22}$$

$$(\mathbf{I} + \mathbf{E_r})^{-1} = \mathbf{I} - \mathbf{E_r} + \mathbf{E_r}^2 - ..., \quad \text{through Taylor series expansion.} \tag{5.23}$$

$$\begin{aligned}
\mathbf{p}_k(\mathbf{E_r}) &= (\mathbf{I} - \mathbf{E_r} + \mathbf{E_r}^2 - ...)\mathbf{p}_k \\
&= (1-t)\mathbf{I}\mathbf{p}_k + \mathcal{O}(t^2)
\end{aligned} \tag{5.24}$$

Finally, the norm of the relative error of the search direction due to the approximation of the Hessian is bounded as a function of the relative error of the Hessian, which in turn is a function of the flow sensitivities.

$$\begin{aligned}
\|\mathbf{p}_k(\mathbf{E_r}) - \mathbf{p}_k\| &= \| - t\mathbf{I}\mathbf{p}_k + \mathcal{O}(t^2)\| \\
&\le t\|\mathbf{p}_k\| + \|\mathcal{O}(t^2)\| \\
&\le t\|\mathbf{p}_k\|.
\end{aligned} \tag{5.25}$$

$$\frac{\|\mathbf{p}_k(\mathbf{E_r}) - \mathbf{p}_k\|}{\|\mathbf{p}_k\|} \le t. \tag{5.26}$$

5.5.4 Eigenvalues

Another interesting error bound to find is the error of the Hessian eigenvalues from the inexact first-order flow sensitivities. Li [25] reformulates Ostrowski's relative perturbation results [33] to show that for two $n \times n$ Hermitian matrices $\mathbf{A}$ and $\hat{\mathbf{A}} = \mathbf{D}^\star \mathbf{A} \mathbf{D}$, where $\mathbf{D}$ is nonsingular, with eigenvalues

$$\boldsymbol{\lambda}(\mathbf{A}) = \{\lambda_1, \cdots, \lambda_n\} \quad \text{and} \quad \boldsymbol{\lambda}(\hat{\mathbf{A}}) = \{\hat{\lambda}_1, \cdots, \hat{\lambda}_n\}, \tag{5.27}$$

ordered so that

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \quad \text{and} \quad \hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \cdots \geq \hat{\lambda}_n, \tag{5.28}$$

we have

$$\sigma_{\min}(\mathbf{D})^2 \cdot \lambda_i \leq \lambda_i \leq \sigma_{\max}(\mathbf{D})^2 \cdot \lambda_i \quad \text{for } 1 \leq i \leq n, \tag{5.29}$$

which implies a relative perturbation bound

$$\max_{1 \leq i \leq n} \frac{|\lambda_i - \hat{\lambda}_i|}{|\lambda_i|} \leq \|\mathbf{I} - \mathbf{D}^* \mathbf{D}\|_2. \tag{5.30}$$

Since the relative error bound on H is known to be $\mathbf{E_r} = t\mathbf{I}$, we can define a perturbation matrix

$$\mathbf{D} = (1 + i\sqrt{t})\mathbf{I}, \tag{5.31}$$

such that

$$\hat{\mathbf{H}} = (1 + t)\mathbf{I}\mathbf{H} = (\mathbf{I} + \mathbf{E_r})\mathbf{H} = \mathbf{D}^\star \mathbf{H}\mathbf{D}. \tag{5.32}$$

Finally, by applying the above theorem to the perturbed Hessian matrix, we recover a bound on the relative error of the eigenvalues

$$\max_{1 \leq i \leq n} \frac{|\lambda_i - \hat{\lambda}_i|}{|\lambda_i|} \leq \|\mathbf{I} - \mathbf{D}^* \mathbf{D}\|_2 = t. \tag{5.33}$$

The theoretical error bound implies that the maximum relative error of the Hessian eigenvalues is less than the relative error of the first-order flow sensitivities. A numerical study from the quasi-one-dimensional test case is shown in Fig. (5.2).

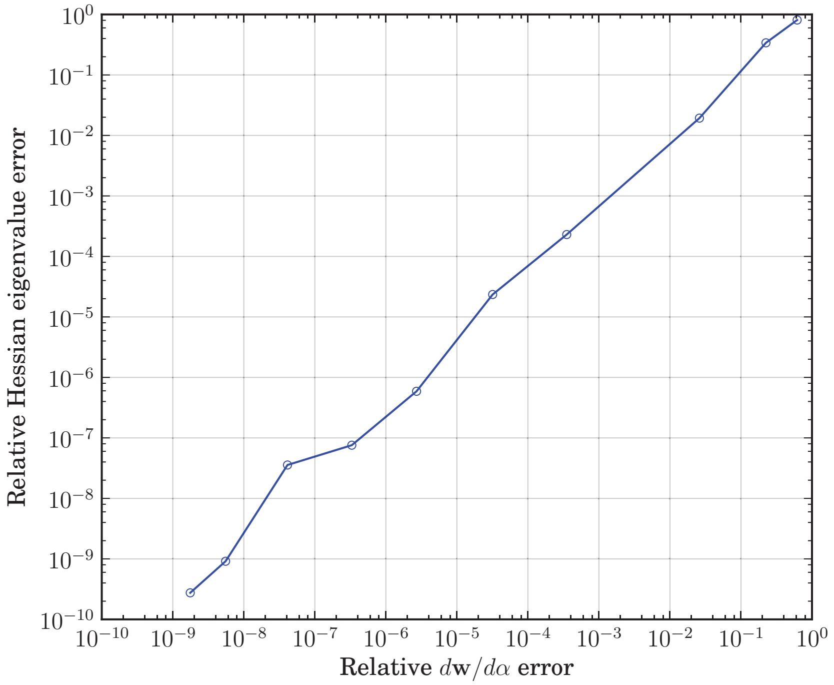


Figure 5.2 – Eigenvalue relative error is proportional to first-order flow sensitivities relative error

Chapter 6

Optimization Results

6.1 Quasi-One-Dimensional Nozzle

The quasi-one dimensional test case is an inverse design optimization problem where the optimization problem is defined as

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|p - p_t\|_2, \\ \text{w.r.t.} \quad & \boldsymbol{\alpha}, \end{aligned}$$

where $\boldsymbol{\alpha}$ are the control points of the B-spline defining the nozzle shape. A target pressure p_t is defined for a nozzle with a target shape S defined as

$$S(x) = 1 - h\left(\sin\left(\pi x^{t_1}\right)\right)^{t_2}, \tag{6.1}$$

with $h = 0.05$, $t_1 = 1.00$, $t_2 = 3.0$. The initial shape is recovered with $h = 0.1$, $t_1 = 0.8$, $t_2 = 6.0$ and the B-spline with control points $\boldsymbol{\alpha}$ is fitted through this initial shape.

Six different optimization methods are explored and compared: Newton's method

Design point #	Finite-Difference	Direct Method	Adjoint Method
1	1.445052**884714980**E-03	1.4450527128061**37**E-03	1.445052712806124E-03
2	3.27920**6296519615**E-04	3.279208105919**162**E-04	3.279208105919257E-04
3	3.35128**9469822513**E-05	3.35129675950**2973**E-05	3.351296759503905E-05
4	2.165**527440082186**E-06	2.1653420178**62913**E-06	2.165342017870861E-06
5	1.228**593748550424**E-06	1.228643267248**969**E-06	1.228643267248725E-06
6	1.43457**8767454679**E-06	1.4345713396**92973**E-06	1.434571339687410E-06
7	5.91**6587891457459**E-07	5.917751860**259154**E-07	5.917751860177707E-07
8	-2.199**928986749219**E-06	-2.19985510201**5695**E-06	-2.199855102017955E-06
9	-7.952**161399393077**E-06	-7.95201221308**6853**E-06	-7.952012213084091E-06
10	-1.7723**20493235569**E-05	-1.772314360287**281**E-05	-1.772314360287690E-05

Table 6.1 – Gradient validation of quasi-one-dimensional nozzle

(NM), BFGS with a scaled identity matrix as its initial Hessian (IBFGS), BFGS with an exactly-initialized Hessian (EBFGS), BFGS with an initial Hessian by partial convergence of $d\mathbf{w}/d\boldsymbol{\alpha}$ (ABFGS1), BFGS with an initial Hessian by neglecting $d^2\mathbf{w}/d\boldsymbol{\alpha}^2$ (ABFGS2), and BFGS with an initial Hessian by partial convergence of $d\mathbf{w}/d\boldsymbol{\alpha}$ and neglecting $d^2\mathbf{w}/d\boldsymbol{\alpha}^2$ (ABFGS3)

Note that ABFGS1 evaluates the Hessian with the direct-adjoint approach since $d^2\mathbf{w}/d\boldsymbol{\alpha}^2$ is too computationally expensive, whereas ABFGS2 and ABFGS3 uses the direct-direct approach. ABFGS2 is used to demonstrate the effects of neglecting the second-order flow sensitivities. The proposed novel methods are ABFGS1 and ABFGS3.

6.1.1 Sensitivity Analysis

First-order

The gradient evaluated through the adjoint method is verified against the finite-difference method and the direct method. A comparison of the first ten design points are seen in Table (6.1), while all fifty design points appear on Fig. (6.1).

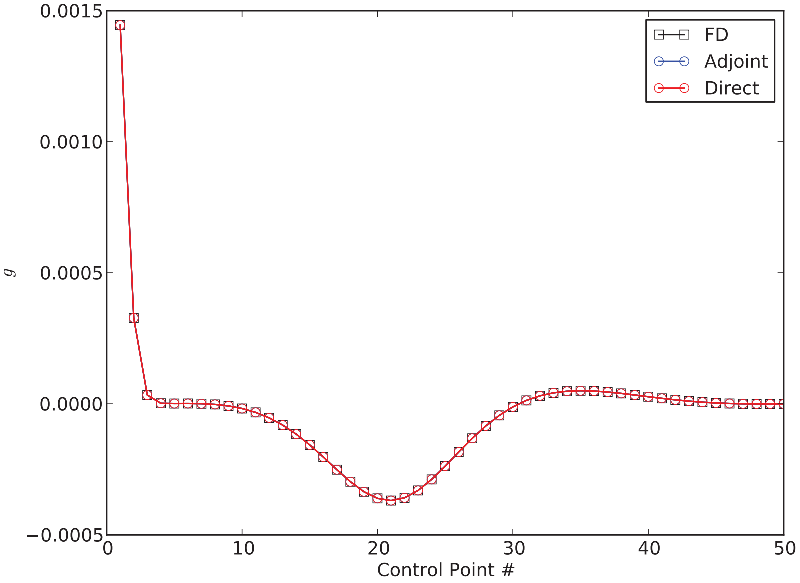


Figure 6.1 – Gradient comparison of quasi-one dimensional inverse design problem

Second-order

The Hessian evaluated through the direct-adjoint method is verified against the finite-difference of the gradient described in subsection (4.2.5). The entry-wise relative error is shown in Fig. (6.2), where $\log_{10}$ of the error is color-coded.

6.1.2 Convergence

The convergence plot shown in Fig. (6.3a) uses lBFGS to demonstrate the increase in the problem complexity with the number of design variables. For larger numbers of design variables, the gradient stagnates for a while before achieving a superlinear convergence rate. This plateauing behavior and decreased convergence rate has also been observed in the work of Skajaa [52] for nonsmooth problems. Using the 50 design variables parametrization as the basis of comparison, all the other methods outperform lBFGS as seen in Fig (6.3b), especially NM that converges quadratically in four design cycles. However, the number of design cycles can be misleading since the cost of each cycle is not taken into account.

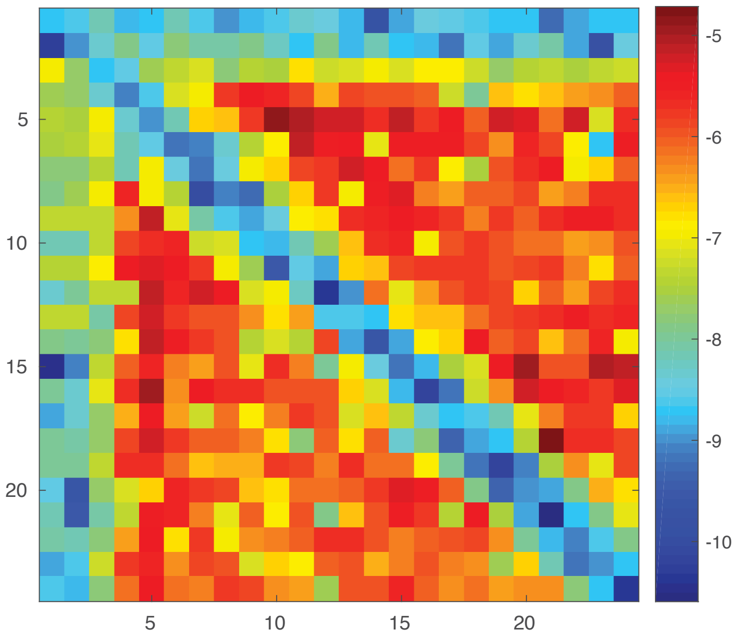
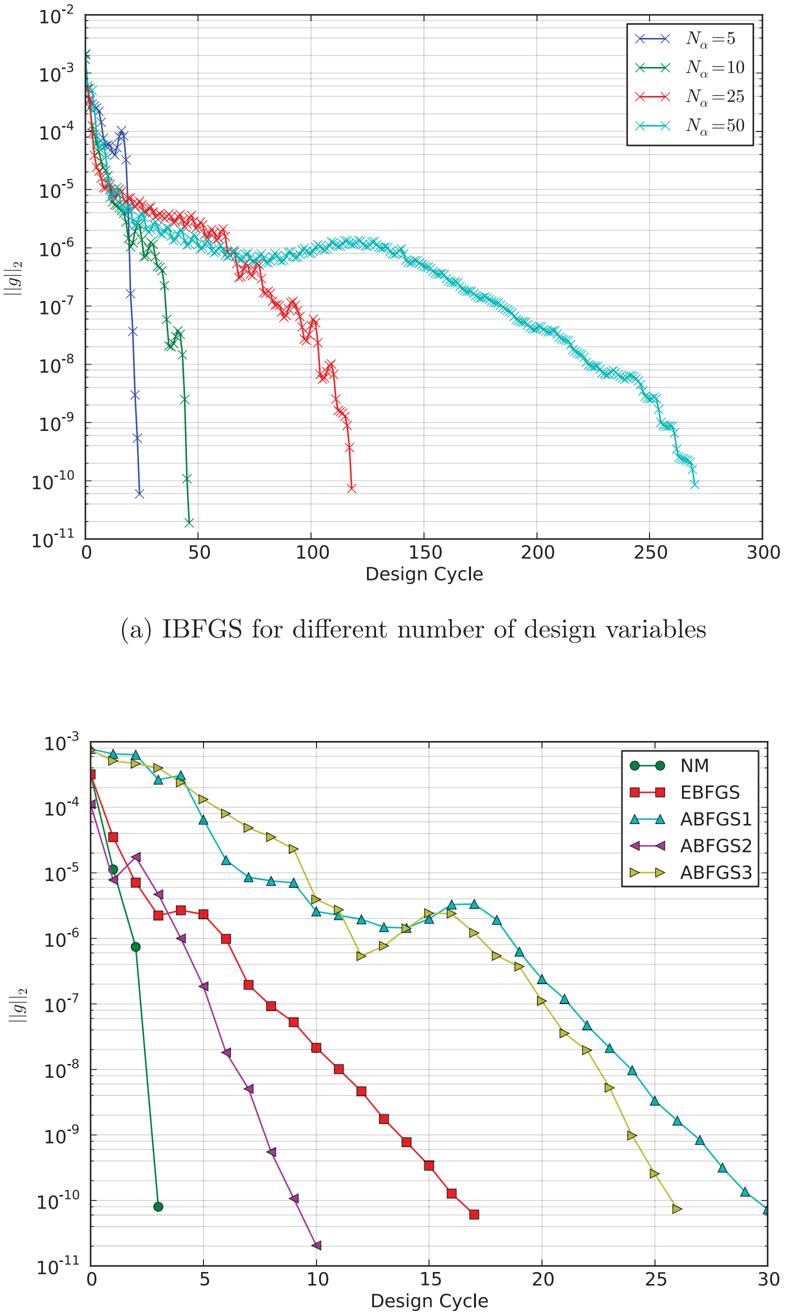


Figure 6.2 – Entry-wise relative error on an exponential scale of the direct-adjoint method and the finite-difference Hessian

Instead, the convergence with respect to the number of system evaluations is shown in Fig. (6.4a). NM requires the full Hessian to be evaluated at every design cycle, for a total of $(N_{\boldsymbol{\alpha}} + 2)$ system evaluations per design cycle. EBFGS and ABFGS has an initial cost of $(N_{\boldsymbol{\alpha}} + 2)$ system evaluations on the first design cycle, but only solves for the flow and the adjoint in the subsequent cycles. Although EBFGS and ABFGS require more design cycles, they require less system evaluations to fully converge. The initial cost is represented by the offset of the first design cycle in Fig. (6.4a). EBFGS and ABFGS2 seem to outperform the other methods, however, they require the first-order flow sensitivities $d\mathbf{w}/d\boldsymbol{\alpha}$ to be solved exactly.

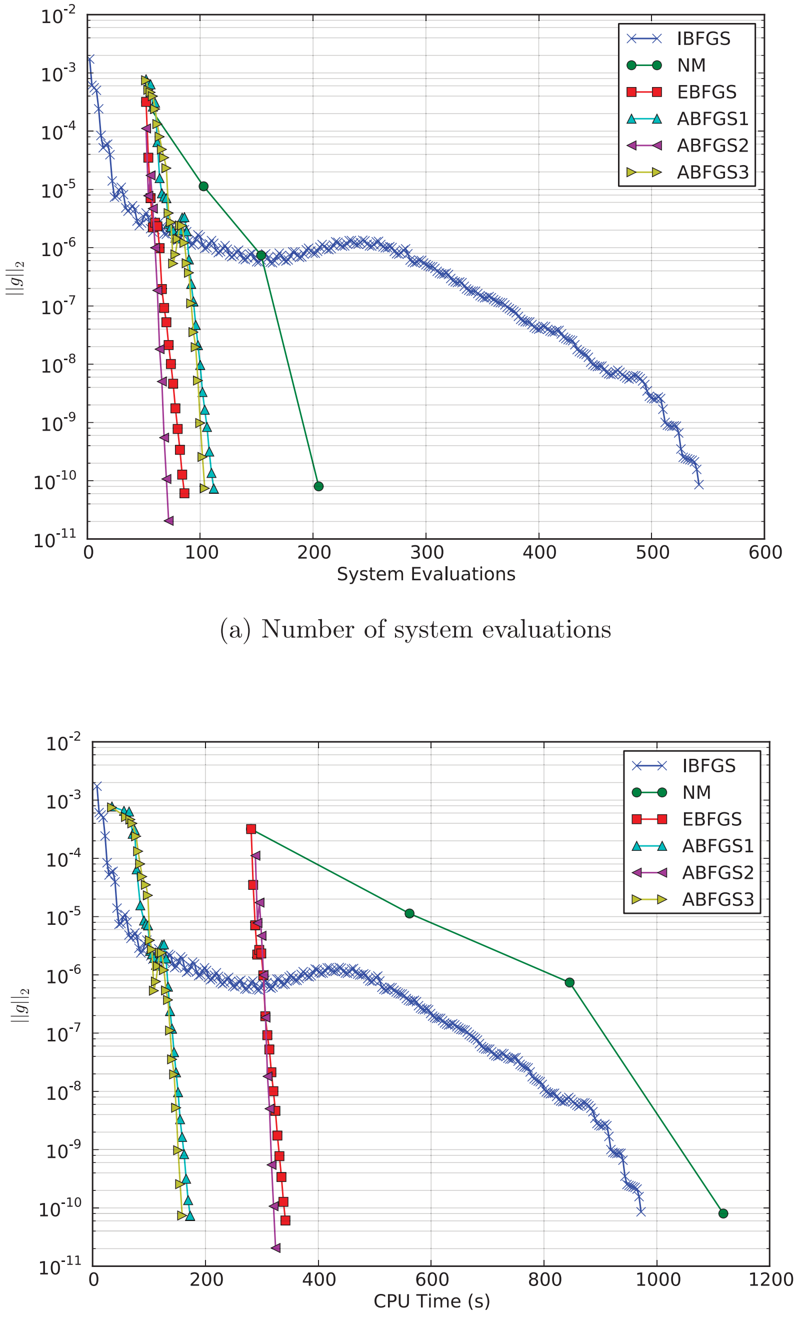
Therefore, the CPU time reflects more accurately the difference in initial cost from approximating first-order flow sensitivities as seen in Fig. (6.4b). The proposed methods ABFGS1 and ABFGS3 greatly outperform the other existing methods with this evaluation criterion because they avoid evaluating the exact first-order flow sensitivities. Additionally,



(a) IBFGS for different number of design variables

(b) Different optimization methods for 50 design variables

Figure 6.3 – Gradient convergence of quasi-one-dimensional inverse design problem with respect to design cycles



(a) Number of system evaluations

(b) CPU time

Figure 6.4 – Gradient convergence of quasi-one-dimensional inverse design problem

we can observe that IBFGS requires more system evaluations but converges more quickly in CPU time than NM. This is explained by the overhead costs of using PETSc's sparse solvers and the rapidity of solving quasi-one-dimensional flows. It is expected that flow equations, adjoint equations, and first-order sensitivities have a similar cost on more complex problems.

Moreover, omitting second-order flow sensitivities seems to slightly improve the convergence. This behavior is not expected to hold for more complex nonlinear problems, so its effects will be case dependent.

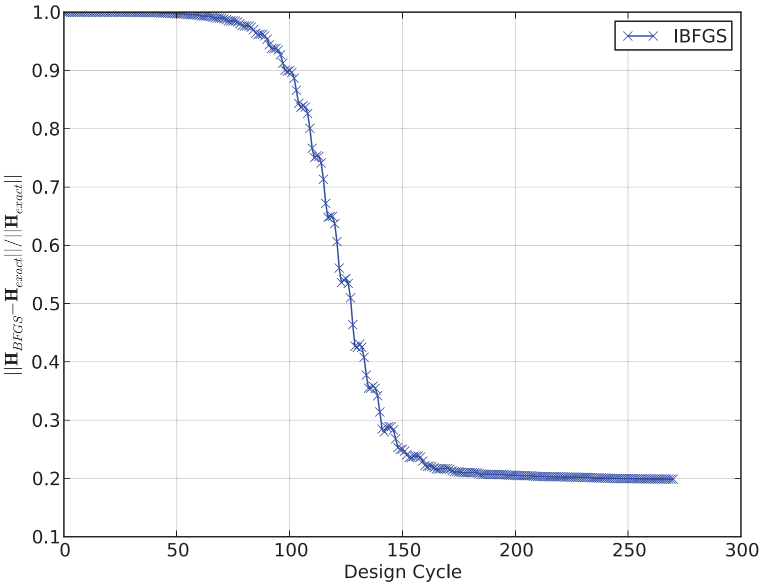
6.1.3 Hessian Accuracy

In order to better understand the importance of the Hessian, the required accuracy for BFGS to retrieve superlinear convergence is studied. The relative error of the Hessian matrix is defined as

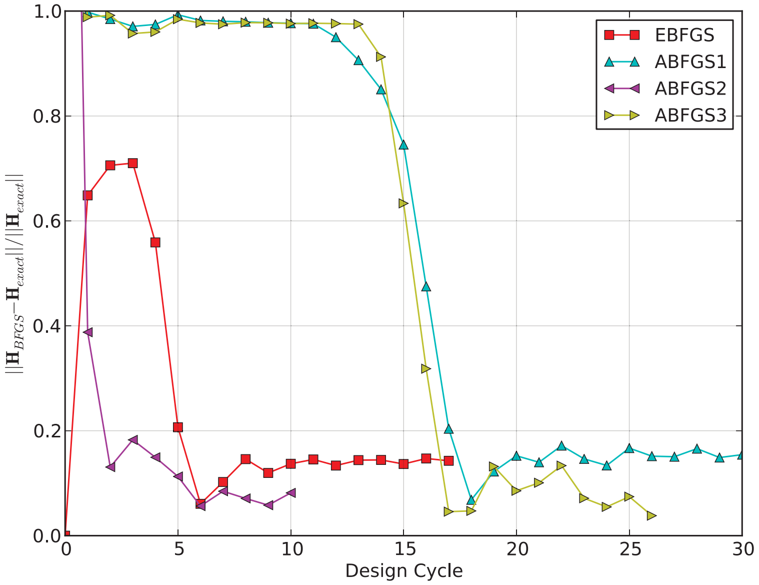
$$\mathbf{E_r} = \frac{\|\mathbf{H}_{exact} - \mathbf{H}_{BFGS}\|}{\|\mathbf{H}_{exact}\|}. \tag{6.2}$$

Fig. (6.5a) demonstrates that IBFGS never recovers the exact Hessian. However, it still converges at a higher rate once its relative error is approximately 0.2 as demonstrated by Fig. (6.4). The same behavior is observed for EBFGS and ABFGS on (6.5b). To compute the error, the exact Hessian is evaluated at every design cycle, making this analysis possible only for simple problems.

Because the first-order flow sensitivities are solved within a specific tolerance, it is unexpected to see high initial errors for ABFGS. The rapid convergence hints that the approximated Hessians should be more accurate than what is shown in Fig. (6.5).



(a) Hessian error for IBFGS



(b) Hessian error for EBFGS and ABFGS

Figure 6.5 – Hessian error from BFGS over the design cycles

6.1.4 Eigenvalue Analysis

The goal of partially solving the first-order flow sensitivities such that the correct magnitudes are retrieved is to approximate the eigenvalues magnitudes and eigenvectors directions. As shown in Fig. (6.6), IBFGS requires an incredible amount of design cycles to approximate the eigenvalues correctly, whereas EBFGS and ABFGS already capture the curvature information of the design space. An in-depth analysis of BFGS's effect on eigenvalues is discussed by Nocedal [31].

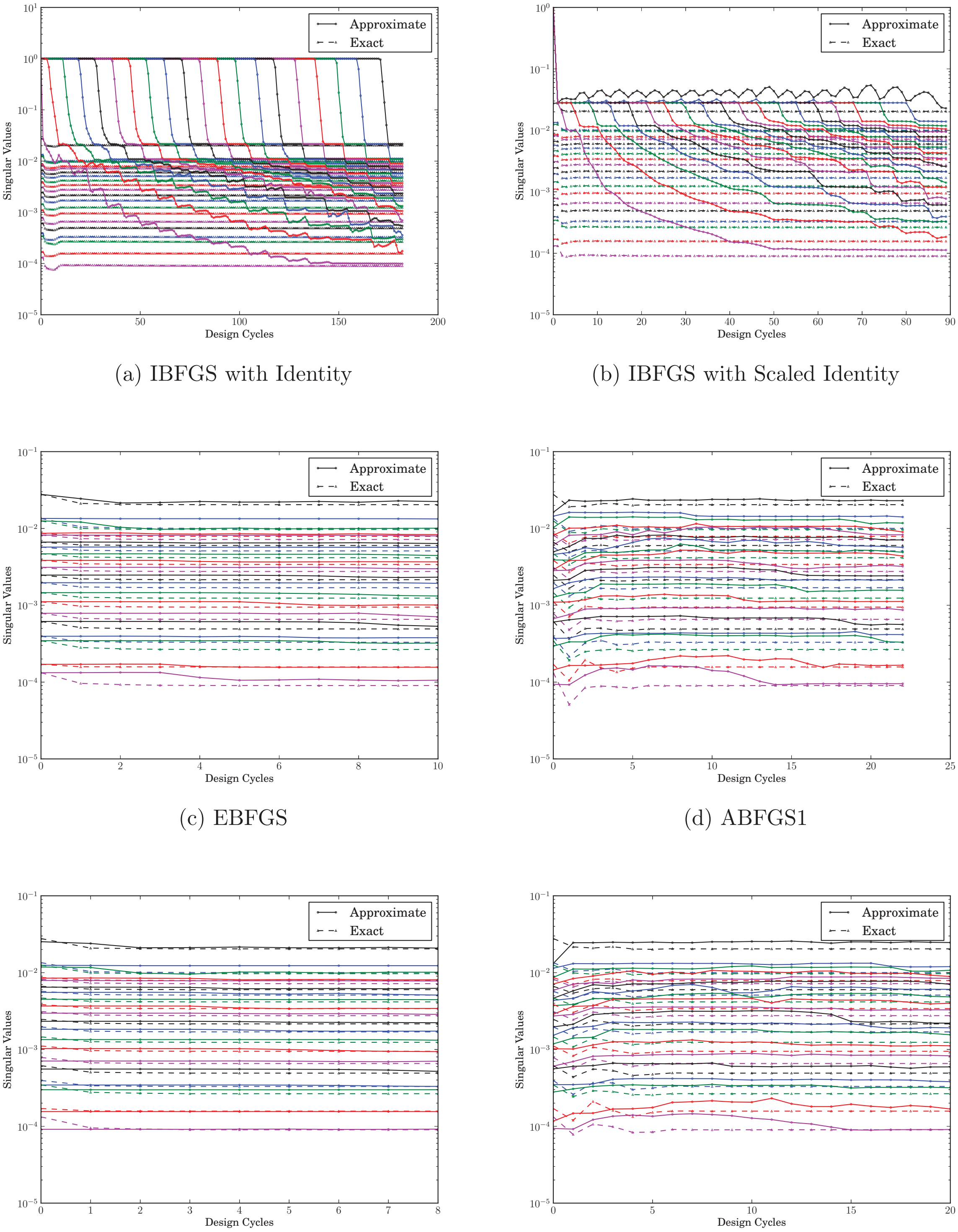
Essentially, second-order information is necessary to compute a proper search direction. Without a Hessian, the quadratic sub-problem reduces to a linear sub-problem leading to the deprecated steepest descent algorithm for most of the early iterations.

6.2 Inviscid Transonic NACA0012 Airfoil

The NACA0012 airfoil problem is the first case of the ADODG [1] problem set. A drag minimization optimization is performed on the airfoil in transonic flow at zero angle of attack. Thickness is constrained such that it may not decrease. The optimization problem is posed as follows

$$\begin{aligned} \text{minimize} \quad & C_D, \\ \text{w.r.t.} \quad & \boldsymbol{\alpha}, \\ \text{subject to:} \quad & |y - y_{\text{baseline}}| \geq 0, \end{aligned}$$

where $\boldsymbol{\alpha}$ are the y-displacements of the control points of a FFD box as shown in Fig. (6.8). The Mach number is fixed at 0.85. The original 2D grid of size 768×128 was extruded in order to use the three-dimensional implementation of the Hessian.



(a) IBFGS with Identity

(b) IBFGS with Scaled Identity

(c) EBFGS

(d) ABFGS1

(e) ABFGS2

(f) ABFGS3

Figure 6.6 – Comparison of BFGS eigenvalues to exact Hessian eigenvalues with 20 design variables

6.2.1 Sensitivity Analysis

The drag gradient is evaluated via the adjoint method throughout the optimization process. A comparison of the finite-difference approach against the adjoint method is shown in Table (6.2) and Fig. (6.7). The relatively high error in the relative gradient error may be due to the inaccuracy of the flow solution since it was only able to converge with a residual of 1.0E-04. As a result, the adjoint, which assumes a zero residual, will be inaccurate and the finite-difference approach, which relies on accurate function evaluations, will also degrade in accuracy. Nevertheless, the optimizer was still able to reduce the KKT optimality by several orders of magnitude as seen in the following section.

Function	L_2 relative error norm
C_D	8.24E-03

Table 6.2 – L_2 relative error norm between finite-difference and adjoint method of inviscid NACA0012 drag gradient

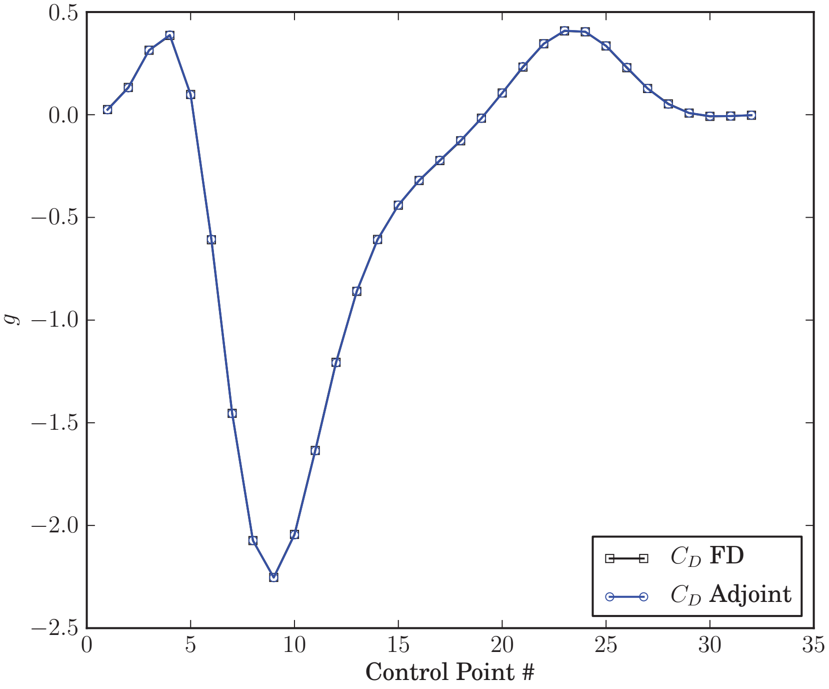


Figure 6.7 – Gradient validation of the NACA0012

6.2.2 Results

The optimization is performed for 32, 64, 96, and 128 FFD points chord-wise. The Mach contours and pressure distributions of the initial and optimal shapes are shown in Fig. (6.8). In [7], the authors parametrized the airfoil with B-splines and achieved a drag coefficient of 37.5 counts with the same 768 × 128 grid utilized here. A super fine grid of 3072 × 512 yielded 25 counts of drag. We believe the difference in the final results may be due to the choice of parametrization. However, the convergence of the Karush-Kuhn-Tucker (KKT) optimality in Fig. (6.9) shows that the optimizer has found an optimum for the given parametrization and validates the contribution presented in this work.

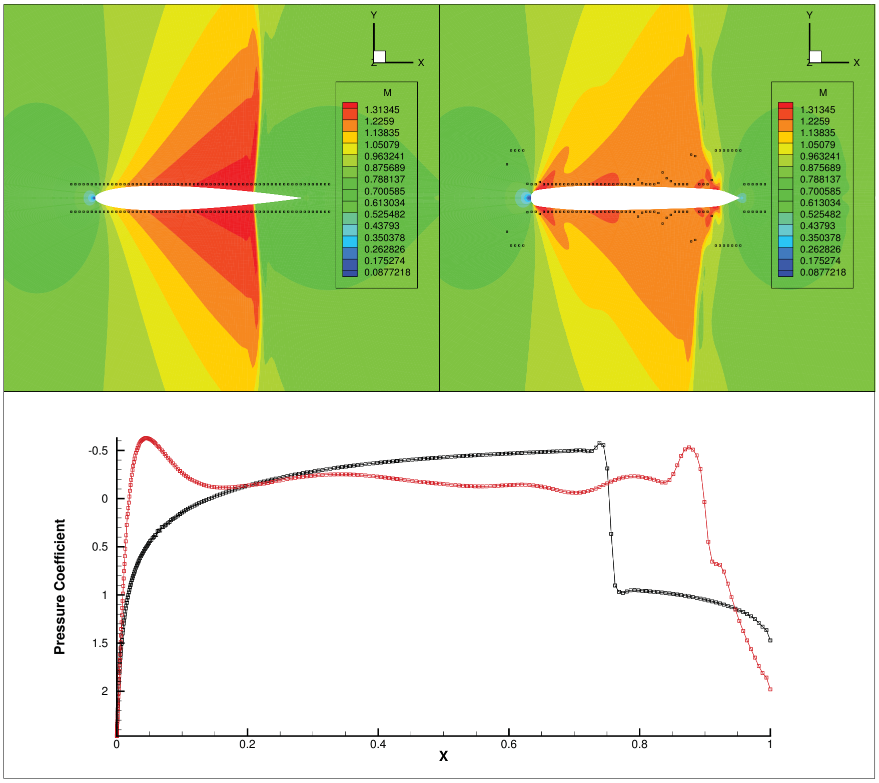


Figure 6.8 – NACA0012 airfoil drag minimization under transonic conditions with 64 design variables

The resulting convergences of the 32 and 64 design variables parametrization in Fig. (6.9a & 6.9b) showcase the acceleration resulting from an approximate Hessian where the optima are found in a lower amount of time. Moreover, the KKT optimality of the conventional approach demonstrates a sublinear convergence rate, characteristic of a gradient descent algorithm, whereas the proposed approach indicates a superlinear rate.

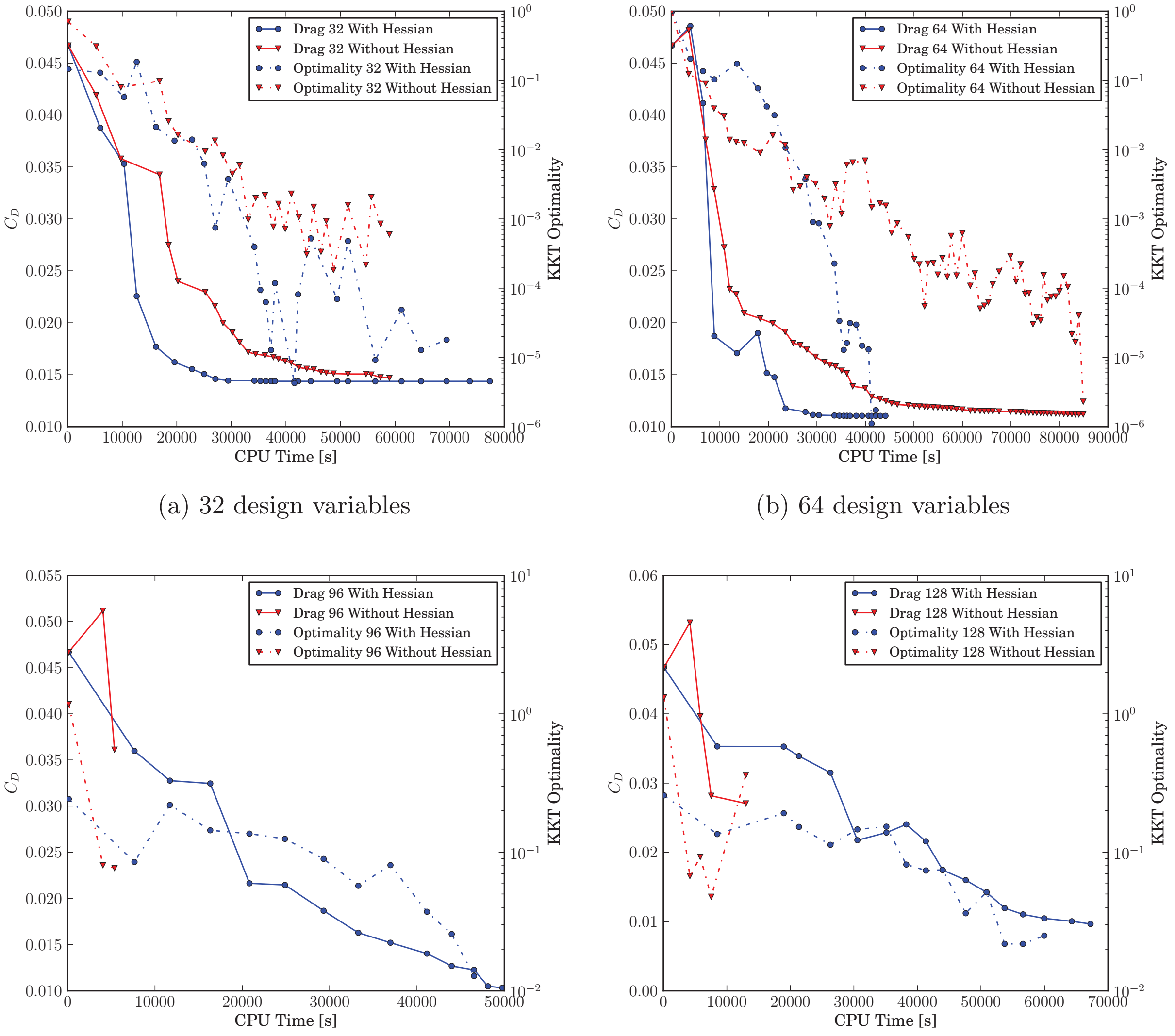
The 96 and 128 design variables parametrizations were initially unable to converge for both the conventional and Hessian approach. A modification as shown in section (5.4), with $\epsilon = 1.0\text{E-}07$, adds stability to the Hessian approach, thus allowing it to take more design steps before stalling. As a result, the increase in design variables allows for a lower optimal drag to be recovered.

Inspecting the eigenvalues of the Hessian in Fig. (6.10), the conventional approach modified less than half of the eigenvalues magnitudes. As a result, the optimizer locates an optimum before BFGS reaches the superlinear convergence rate. This superlinear convergence is attained in the early design cycles of the Hessian-based approach due to second-order information.

6.3 Inviscid Onera M6 Drag Minimization

The ONERA M6 test case is a drag minimization problem where lift and volume are constrained to be greater or equal to their initial values. The minimum thickness must be within 25% of the initial thickness to avoid unrealistically thin wing tips. The optimization problem is posed as follows

$$\begin{aligned}
\text{minimize} \quad & C_D, \\
\text{w.r.t.} \quad & \boldsymbol{\alpha}, \\
\text{subject to:} \quad & C_L - 0.0875 \geq 0, \\
& V - 0.0633 \geq 0,
\end{aligned}$$



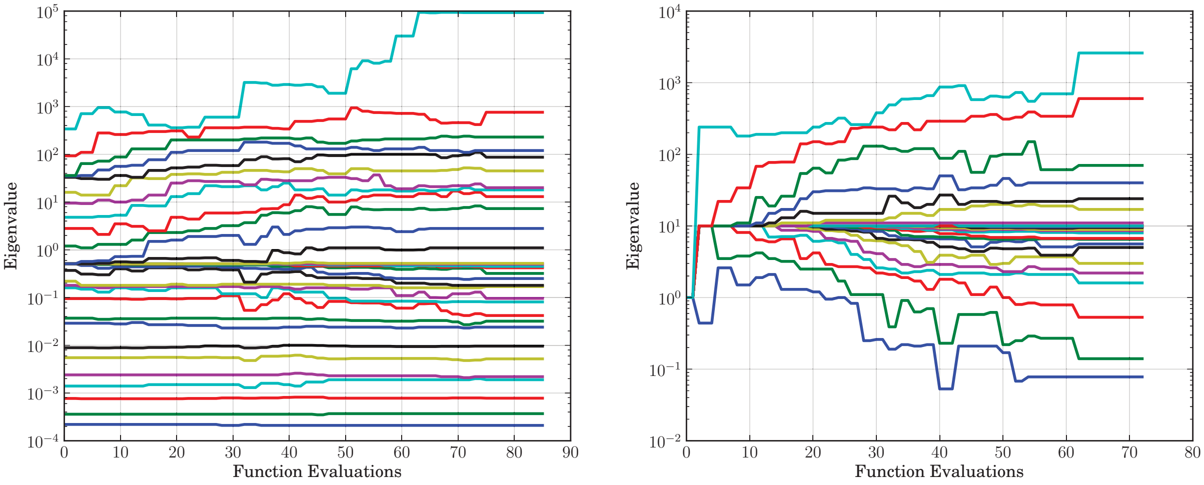
(a) 32 design variables

(b) 64 design variables

(c) 96 design variables

(d) 128 design variables

Figure 6.9 – Drag and KKT optimality convergence of NACA0012 airfoil



(a) With Hessian

(b) Without Hessian

Figure 6.10 – Hessian eigenvalues of NACA0012 airfoil with 32 design variables

where $\boldsymbol{\alpha}$ are the y-displacements of the control points of a FFD box as shown in Fig. (6.11). The leading edge, trailing edge, root and tip are constrained. The Mach number is fixed at 0.84.

6.3.1 Sensitivity Analysis

The lift and drag gradients are evaluated via the adjoint method throughout the optimization process. A comparison of the finite-difference approach against the adjoint method is shown in Table (6.3) and Fig. (6.12).

Function	L_2 relative error norm
C_D	2.01E-04
C_L	2.48E-04

Table 6.3 – L_2 relative error norm between finite-difference and adjoint method of inviscid ONERA M6 lift and drag gradients

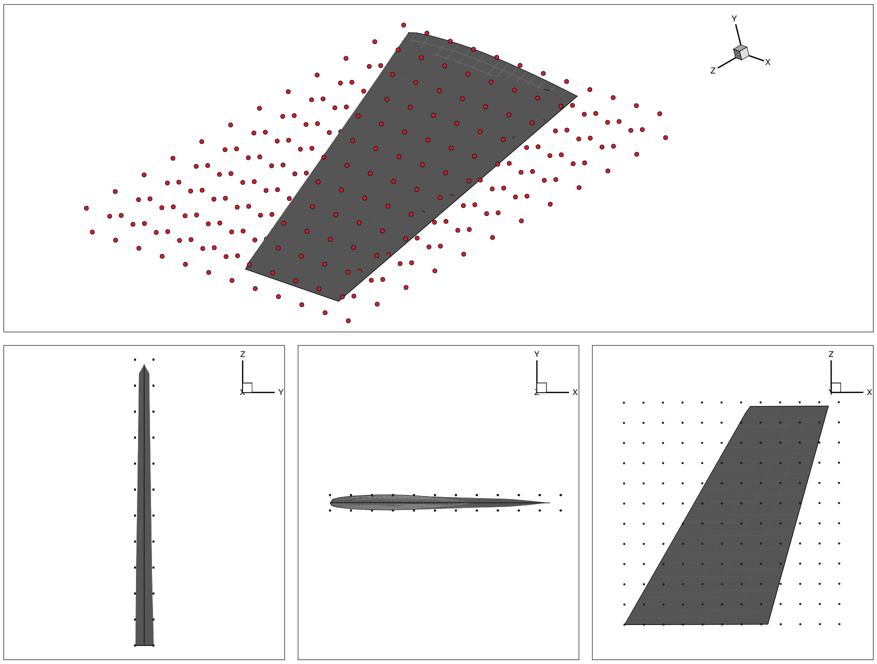


Figure 6.11 – ONERA M6 wing enclosed in FFD box with 12 control points span-wise and chord-wise

6.3.2 Results

The optimization is performed for four different number of design variables, where the span-wise and chord-wise distribution is refined. The convergence of drag and KKT optimality can be found in Fig. (6.13). Drag is minimized by reducing the strength of the shock, which is achieved by decreasing the thickness of the tip while increasing the root thickness to maintain volume.

As expected, the optimization framework infused with second-order sensitivities require fewer steps to reach an optimum than the identity-based BFGS method. Furthermore, the CPU time required to reach the optimum shows that the Hessian is also beneficial in terms of computational cost due to its higher convergence rate. As seen in Fig. (6.14), only a few eigenvalues were modified with the conventional approach. The identity-initialized BFGS method is therefore extremely similar to gradient descent, whereas the proposed approach would scale the directions according to the design space curvature.

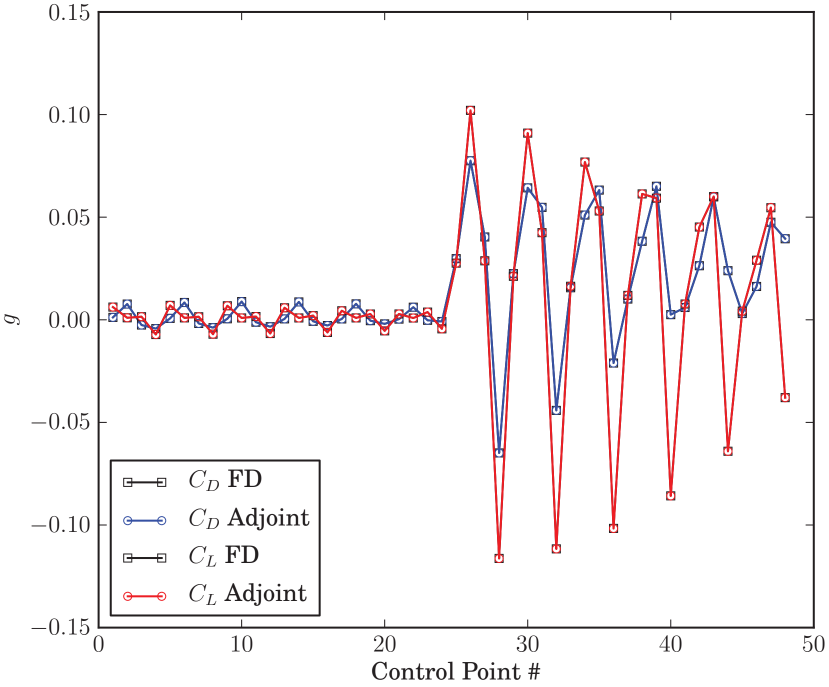
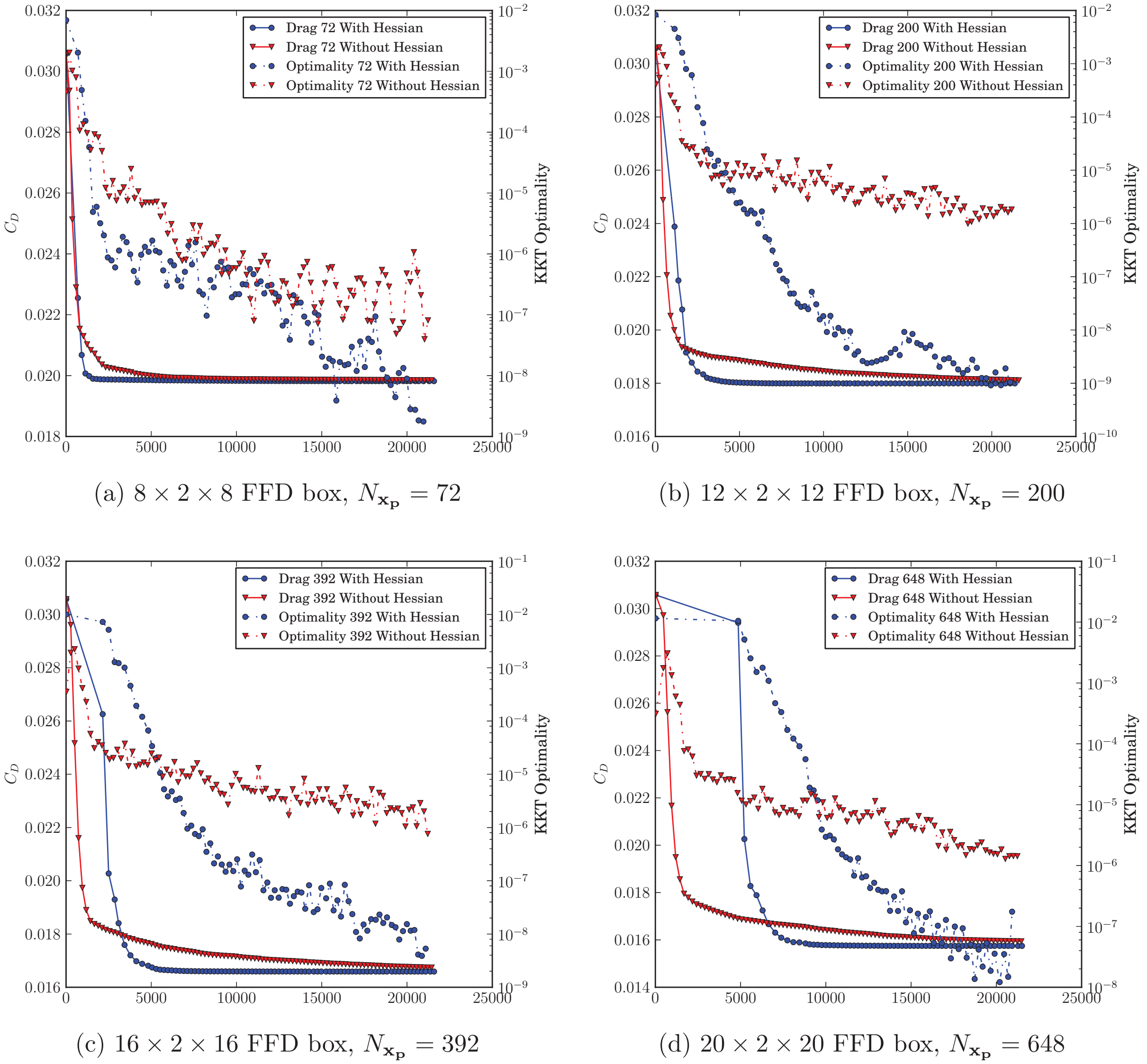


Figure 6.12 – Gradient validation of the ONERA M6

Unfortunately, both methods suffer from a sublinear convergence of the KKT optimality. The sublinear behaviour of the Hessian-based approach may be caused by the following two reasons. First, the initial non-convexity of the problem leads to a positive-definite modification of the Hessian, which estimates the curvature magnitude in the descent direction.

In the 200 design variables case, 72 out of 200 eigenvalues were initially modified to the positive geometric mean eigenvalue. The set of modified eigenvalues can be visualized in Fig. (6.14a), where they are initialized with the value 2.3E-03.

Second, as the optimizer moves throughout the design space, the curvature also changes for optimization problems that are not quadratic in nature and BFGS may not update the Hessian as fast as the curvature is changing. Therefore, it is not surprising to witness a decrease in convergence rate as we move away from the point where the approximate Hessian was initially evaluated. Re-evaluating an approximate Hessian later in the



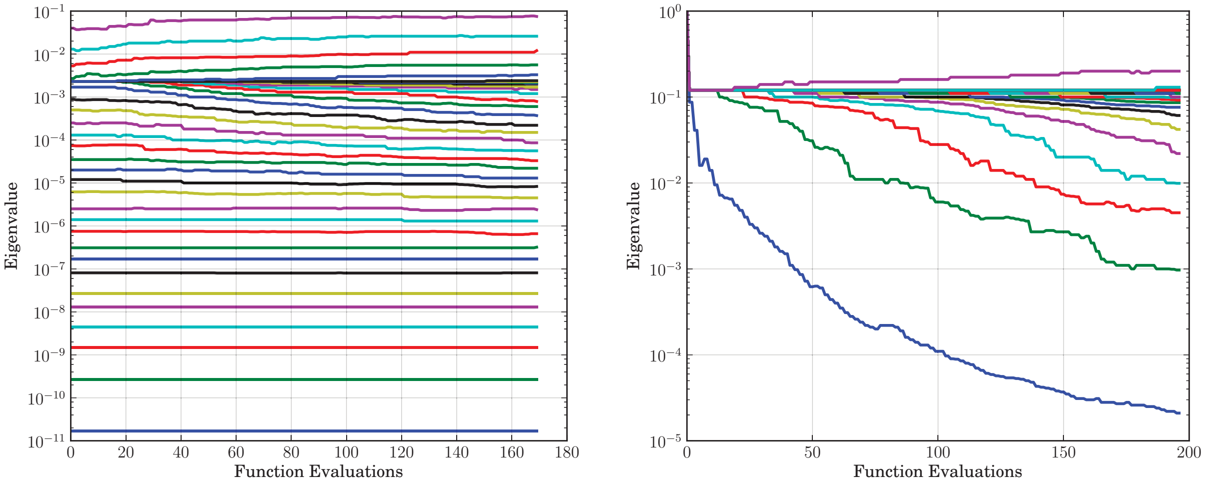
(a) $8 \times 2 \times 8$ FFD box, $N_{\mathbf{x_p}} = 72$

(b) $12 \times 2 \times 12$ FFD box, $N_{\mathbf{x_p}} = 200$

(c) $16 \times 2 \times 16$ FFD box, $N_{\mathbf{x_p}} = 392$

(d) $20 \times 2 \times 20$ FFD box, $N_{\mathbf{x_p}} = 648$

Figure 6.13 – Drag and KKT optimality convergence of ONERA M6 Wing



(a) With Hessian

(b) Without Hessian

Figure 6.14 – 40 out of 200 Hessian eigenvalues of ONERA M6 wing

design process is subject for further investigation to determine cost effectiveness.

Furthermore, as the number of design variables increases, the optimization problem does not seem to increase in complexity as we had initially expected. We believe this behaviour is due to the nature of the problem, where shock removal only requires local changes. Since the optimum is found quickly, the acceleration induced by the Hessian may not outweigh the initial cost for a high number of design variables in this case.

Finally, an interesting feature appearing in the convergence plots is the larger oscillations of the KKT optimality. The large change in optimality from one design cycle to another indicates that the optimizer is taking larger steps in the design space which should increase the convergence rate. However, it would also favor violation of the linearized constraints, depending on the nonlinearity of the problem.

Function	L_2 relative error norm
C_D	3.18E-05
C_L	3.33E-05
C_M	3.02E-05

Table 6.4 – L_2 relative error norm between finite-difference and adjoint method of inviscid CRM lift, drag, and pitching moment gradients

6.4 Inviscid CRM Drag Minimization

The inviscid CRM test case is a drag minimization problem where lift and volume are constrained to be greater or equal to their initial values. It is similar to the fourth ADODG [1] test case, except that it is inviscid and the pitching moment constraint has been removed. The optimization problem is posed as follows

$$\begin{aligned} \text{minimize} \quad & C_D, \\ \text{w.r.t.} \quad & \boldsymbol{\alpha}, \\ \text{subject to:} \quad & C_L - 0.5 \geq 0, \\ & V - 0.2615 \geq 0, \end{aligned}$$

where $\boldsymbol{\alpha}$ are the y-displacements of the control points of a mapped FFD box as shown in Fig. (6.15). The leading edge, trailing edge, root and tip are constrained. The minimum thickness must be within 25% of the initial thickness to avoid unrealistically thin wing tips. The Mach number is fixed at 0.85.

6.4.1 Sensitivity Analysis

The drag, lift and pitching moment gradients are evaluated via the adjoint method throughout the optimization process. A comparison of the finite-difference approach against the adjoint method is shown in Table (6.4) and Fig. (6.16).

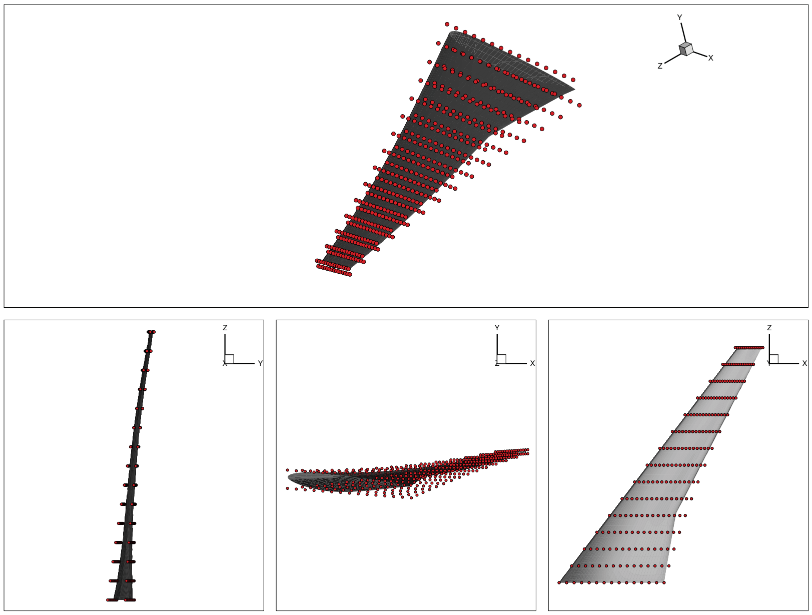


Figure 6.15 – CRM wing enclosed in FFD box with 15 control points span-wise and chord-wise

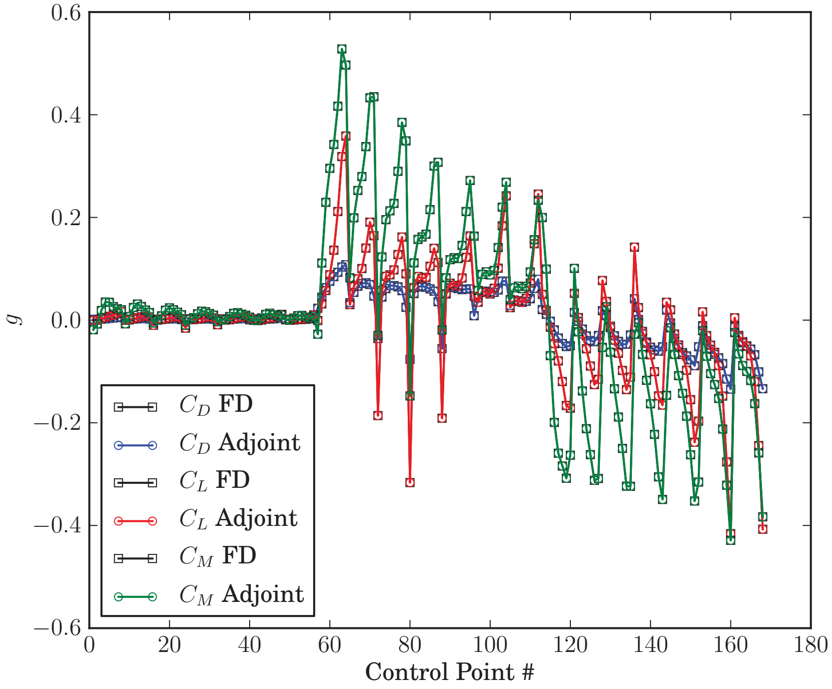


Figure 6.16 – Gradient validation of the CRM

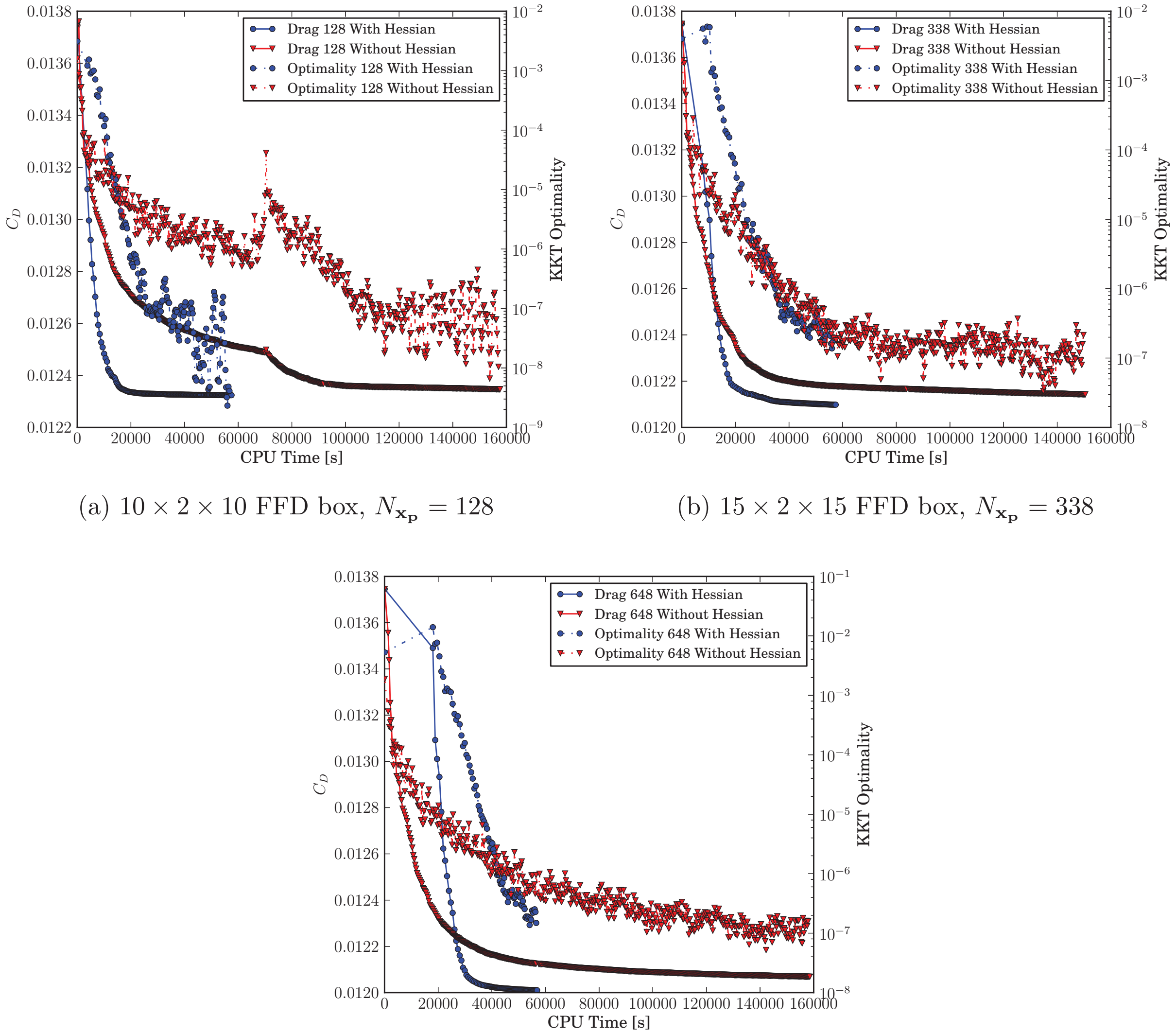
6.4.2 Results

The optimization is performed for three different number of design variables, where the span-wise and chord-wise distribution is refined. The convergence of drag and KKT optimality can be found in Fig. (6.17). Drag is minimized by reducing the strength of the shock the same way it was achieved for the ONERA M6.

The convergence rate is greatly increased both in terms of design cycles and CPU time in all three cases. The same pressure coefficients and airfoil shapes are recovered as seen in Fig. (6.19). Sublinear convergence is explained by the same arguments mentioned for the ONERA M6. Once again, the eigenvalues of the Hessian seen in Fig. (6.18) support the higher convergence rate of the proposed approach.

The 128 design variables parametrization convergence in Fig. (6.17a) demonstrates an interesting behaviour of the conventional approach. The KKT optimality initially stalls very quickly and rises before converging further as previously seen in the quasi-one-dimensional nozzle in Fig. (6.4a). We believe the optimizer slowdown was due to the presence of a saddle point where gradient vector is close to zero. Therefore, the optimizer requires multiple BFGS updates to properly scale and rotate the search directions. Once it retrieves an appropriate search direction, the gradient magnitudes are no longer zero and the cost function decreases at a higher rate until it finds the same critical point recovered by the Hessian approach. It is hypothesized that the curvature information from the Hessian approach allows the optimizer to jump over some noisy regions and saddle points of the design space, where gradient descent might stall.

A greater number of design variables does not seem to increase the optimization complexity for wave drag reduction problems. Hence, increasing the number of FFD control points does not necessarily increase the stiffness of the optimization problem as previously discussed in subsection (6.1).

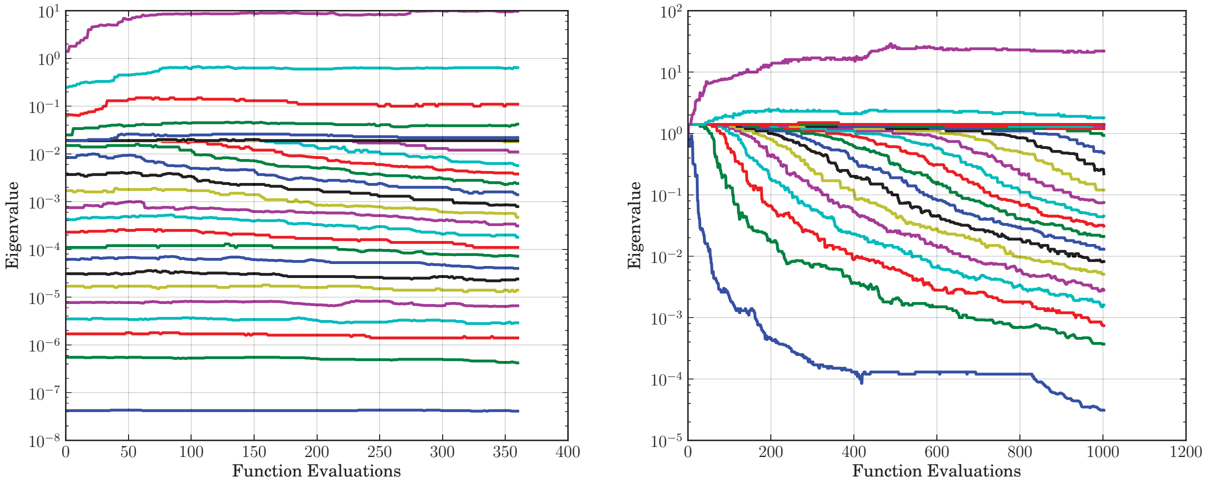


(a) $10 \times 2 \times 10$ FFD box, $N_{\mathbf{x_p}} = 128$

(b) $15 \times 2 \times 15$ FFD box, $N_{\mathbf{x_p}} = 338$

(c) $20 \times 2 \times 20$ FFD box, $N_{\mathbf{x_p}} = 648$

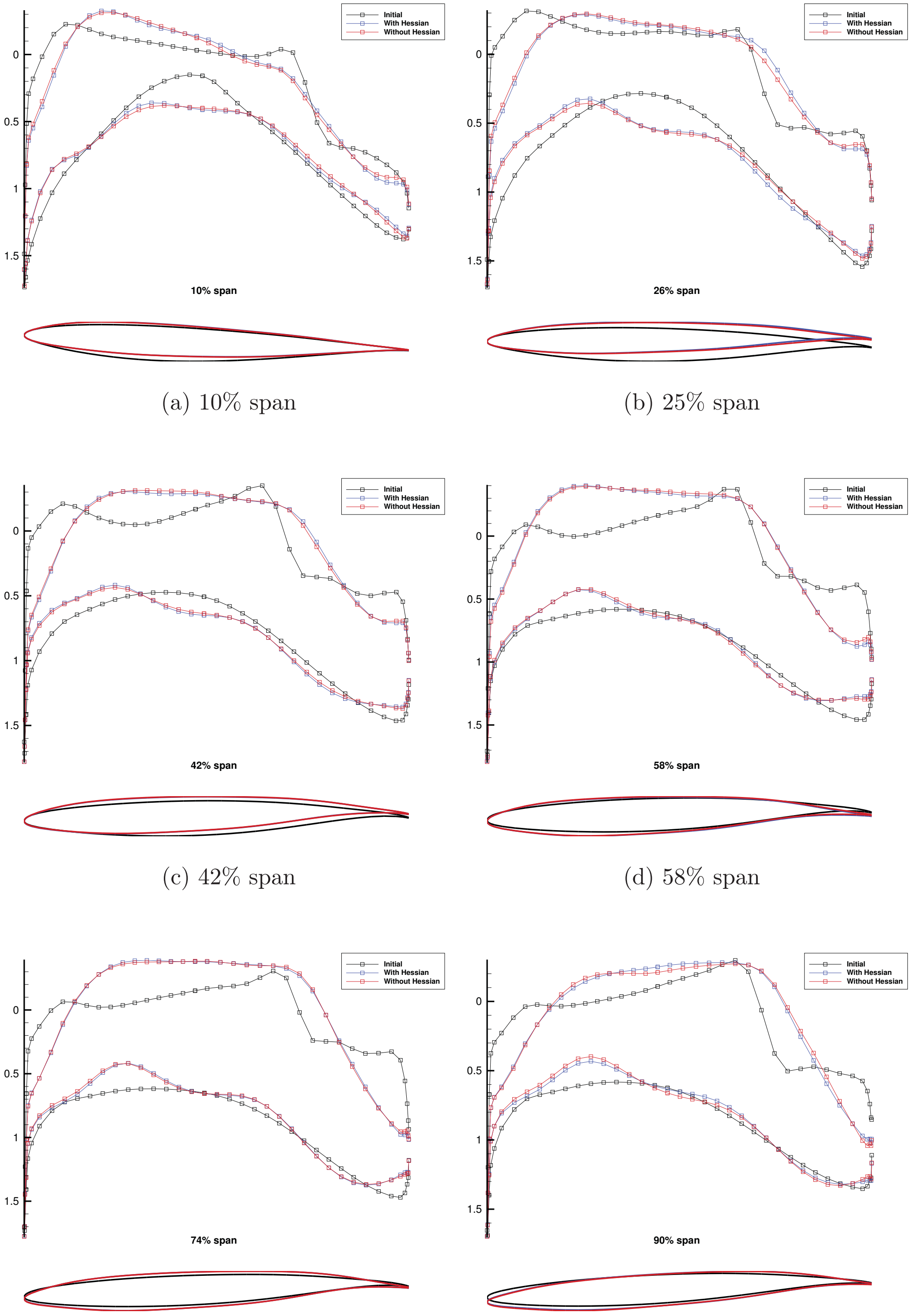
Figure 6.17 – Drag and KKT optimality convergence of CRM wing



(a) With Hessian

(b) Without Hessian

Figure 6.18 – 25 out of 128 Hessian eigenvalues of CRM



(a) 10% span

(b) 25% span

(c) 42% span

(d) 58% span

(e) 74% span

(f) 90% span

Figure 6.19 – Pressure coefficient plots of CRM wing at different span locations for 128 design variables

Chapter 7

Conclusion

The approximation of the initial Hessian by partial solutions of first-order flow sensitivities in a BFGS framework may greatly reduce the computational effort in ASO. The error incurred by this approximation is bounded for the Hessian entries, the search direction and the Hessian eigenvalues. Further approximation neglecting second-order flow sensitivities leads to a simpler implementation of the algorithm while decreasing computational requirements. An existing adjoint-based framework can easily evaluate the first-order flow sensitivities and only requires partial second-order derivatives of the cost function.

The proposed approach is superior to the conventional identity-initialized BFGS approach on every test case. The superlinear convergence is observed in the quasi-one-dimensional and two-dimensional test cases. Although the convergence rate becomes sublinear for the three-dimensional test cases, the Hessian-based approach still outperforms the conventional approach. The acceleration benefits may be limited when the optimization problem does not have a complex enough design space and a large number of design variables. More specifically, three-dimensional wave drag reduction does not exhibit the decreased convergence rate as the number of design variables increases.

An eigenvalue analysis for the presented test cases demonstrate the ability of the optimizer to capture second-order information. Most practitioners may find that their BFGS algorithm predominantly behaves as a gradient descent algorithm. The wide range of eigenvalues recovered by the approximate Hessian aid to scale the descent directions. In effect, small eigenvalues will accelerate the convergence, whereas large eigenvalues will stabilize it. It is also possible to perform further analysis of the eigenvectors effect on the rotation of the search direction.

The full benefits of the proposed algorithm are yet to be explored by applying it to more challenging test cases. Its tractability and benefits on a viscous test case is the next step to justify the high initial cost. Using the finite-difference method may be an attractive alternative to approximate the first-order flow sensitivities. Additionally, a mathematical bound on the error incurred by the second-order flow sensitivities or a better approximation would help justify the current approximation. In addition to an accelerated convergence rate of ASO, second-order information may be of use in one-shot methods and uncertainty quantification.

Bibliography

[1] *Aerodynamic Design 0ptimization Discussion Group*. American Institute of Aeronautics and Astronautics. 2015. URL: `https://info.aiaa.org/tac/ASG/APATC/AeroDesignOpt-DG/default.aspx`.

[2] E. Arian and S. Ta'asan. « Analysis of the Hessian for aerodynamic optimization: inviscid flow ». In: *Computers & Fluids* 28.7 (Sept. 1999), pp. 853–877.

[3] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. « Efficient management of parallelism in object oriented numerical software libraries ». In: *Modern Software Tools in Scientific Computing*. Ed. by E. Arge, A. M. Bruaset, and H. P. Langtangen. Birkhäuser Press, 1997, pp. 163–202.

[4] S. Balay et al. *PETSc users manual*. Tech. rep. ANL-95/11 - Revision 3.7. Argonne National Laboratory, Apr. 2016.

[5] S. Balay et al. *PETSc Web page*. 2016. URL: `http://www.mcs.anl.gov/petsc`.

[6] F. Bisson. « Aerodynamic design optimization using the drag decomposition method ». MA thesis. McGill University, Aug. 2015.

[7] F. Bisson and S. Nadarajah. « Adjoint-based aerodynamic optimization of benchmark problems ». In: *53rd AIAA Aerospace Sciences Meeting*. Paper No. AIAA 2015-1948. American Institute of Aeronautics and Astronautics, Jan. 2015.

[8] A. de Boer, M. van der Schoot, and H. Bijl. « Mesh deformation based on radial basis function interpolation ». In: *Computers & Structures* 85.11 (July 2007). Fourth MIT Conference on Computational Fluid and Solid Mechanics, pp. 784–795.

[9] C. G. Broyden. « A new double-rank minimization algorithm ». In: *Notices American Mathematical Society* 16 (1969), p. 670.

[10] C. G. Broyden. « The convergence of a class of double-rank minimization algorithms 1. general considerations ». In: *IMA Journal of Applied Mathematics* 6.1 (Mar. 1970), pp. 76–90.

[11] J. Cagnone, K. Sermeus, S. Nadarajah, and E. Laurendeau. « Implicit multigrid schemes for challenging aerodynamic simulations on block-structured grids ». In: *Computers & Fluids* 44.1 (May 2011), pp. 314–327.

[12] S. Choi, K. Lee, M. M. Potsdam, and J. J. Alonso. « Helicopter rotor design using a time-spectral and adjoint-based method ». In: *AIAA Journal of Aircraft* 51.2 (Feb. 2014), pp. 412–423.

[13] R. Fletcher. « A new approach to variable metric methods ». In: *The Computer Journal* 13.3 (1970), pp. 317–322.

[14] D. Ghate and M. Giles. « Efficient Hessian calculation using automatic differentiation ». In: *25th AIAA Applied Aerodynamics Conference*. Paper No. AIAA-2007-4059. June 2007.

[15] K. C. Giannakoglou and D. I. Papadimitriou. « Adjoint methods for shape optimization ». In: *Optimization and Computational Fluid Dynamics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. Chap. 10, pp. 79–108.

[16] P. E. Gill, W. Murray, and M. A. Saunders. « SNOPT: an SQP algorithm for large-scale constrained optimization ». In: *SIAM Review* 47.1 (Aug. 2005), pp. 99–131.

[17] D. Goldfarb. « A family of variable-metric methods derived by variational means ». In: *Mathematics of Computation* 24.109 (1970), pp. 23–26.

[18] L. Hascoët and V. Pascual. « The Tapenade automatic differentiation tool: principles, model, and specification ». In: *ACM Transactions On Mathematical Software* 39.3 (Apr. 2013). Article No. 20.

[19] J. E. Hicken. « Inexact Hessian-vector products in reduced-space differential-equation constrained optimization ». In: *Optimization and Engineering* 15.3 (Sept. 2014), pp. 575–608.

[20] J. E. Hicken and D. W. Zingg. « Induced-drag minimization of nonplanar geometries based on the Euler equations ». In: *AIAA Journal* 48.11 (Nov. 2010), pp. 2564–2575.

[21] S. Jakobsson and O. Amoignon. « Mesh deformation using radial basis functions for gradient-based aerodynamic shape optimization ». In: *Computers & Fluids* 36.6 (July 2007), pp. 1119–1136.

[22] A. Jameson, L. Martinelli, and N. Pierce. « Optimum aerodynamic design using the Navier-Stokes equations ». In: *Theoretical and Computational Fluid Dynamics* 10.1 (Jan. 1998), pp. 213–237.

[23] A. Jameson. « Aerodynamic design via control theory ». In: *Journal of Scientific Computing* 3.3 (Sept. 1988), pp. 233–260.

[24] G. K. Kenway and J. R. Martins. « Multipoint high-fidelity aerostructural optimization of a transport aircraft configuration ». In: *AIAA Journal of Aircraft* 51.1 (Jan. 2014), pp. 144–160.

[25] R.-C. Li. « Relative perturbation theory: I. eigenvalue and singular value variations ». In: *SIAM Journal on Matrix Analysis and Applications* 19.4 (Feb. 1998), pp. 956–982.

[26] Z. Lyu, G. K. Kenway, and J. R. Martins. « Aerodynamic shape optimization investigations of the common research model wing benchmark ». In: *AIAA Journal* 53.4 (Apr. 2014), pp. 968–985.

[27] L. Martinelli and A. Jameson. « Validation of a multigrid method for the Reynolds averaged equations ». In: *AIAA 26th Aerospace Sciences Meeting.* Paper No. AIAA-88-0414. Jan. 1988.

[28] J. R. Martins, J. J. Alonso, and J. J. Reuther. « A coupled-adjoint sensitivity analysis method for high-fidelity aero-structural design ». In: *Optimization and Engineering* 6.1 (Mar. 2005), pp. 33–62.

[29] S. Nadarajah and C. Tatossian. « Multi-objective aerodynamic shape optimization for unsteady viscous flows ». In: *Optimization and Engineering* 11.1 (Feb. 2010), pp. 67–106.

[30] E. J. Nielsen, E. M. Lee-Rausch, and W. T. Jones. « Adjoint-based design of rotors in a noninertial reference frame ». In: *AIAA Journal of Aircraft* 47.2 (Mar. 2010), pp. 638–646.

[31] J. Nocedal. « Theory of algorithms for unconstrained optimization ». In: *Acta Numerica* 1.5 (1992), pp. 199–242.

[32] J. Nocedal and S. J. Wright. *Numerical optimization*. 2nd ed. Springer, 2006.

[33] A. M. Ostrowski. « A quantitative formulation of Sylvester's law of inertia ». In: *Proceedings of the National Academy of Science* 45.5 (May 1959), pp. 740–744.

[34] D. I. Papadimitriou and K. C. Giannakoglou. « A continuous adjoint method with objective function derivatives based on boundary integrals, for inviscid and viscous flows ». In: *Computers & Fluids* 36.2 (Feb. 2007), pp. 325–341.

[35] D. I. Papadimitriou and K. C. Giannakoglou. « Aerodynamic design using the truncated Newton algorithm and the continuous adjoint approach ». In: *International Journal for Numerical Methods in Fluids* 68.6 (Jan. 2012), pp. 724–739.

[36] D. I. Papadimitriou and K. C. Giannakoglou. « Aerodynamic shape optimization using first and second order adjoint and direct approaches ». In: *Archives of Computational Methods in Engineering* 15.4 (Aug. 2008), pp. 447–488.

[37] D. I. Papadimitriou and K. C. Giannakoglou. « Computation of the Hessian matrix in aerodynamic inverse design using continuous adjoint formulations ». In: *Computers & Fluids* 37.8 (Sept. 2008), pp. 1029–1039.

[38] D. I. Papadimitriou and K. C. Giannakoglou. « Direct, adjoint and mixed approaches for the computation of Hessian in airfoil design problems ». In: *International Journal for Numerical Methods in Fluids* 56.10 (Sept. 2008), pp. 1929–1943.

[39] D. I. Papadimitriou and K. C. Giannakoglou. « The continuous direct-adjoint approach for second order sensitivities in viscous aerodynamic inverse design problems ». In: *Computers & Fluids* 38.8 (Sept. 2009), pp. 1539–1548.

[40] K. B. Petersen and M. S. Pedersen. *The Matrix Cookbook*. Version 20121115. Nov. 2012.

[41] L. Piegl and W. Tiller. *The NURBS Book*. 2nd ed. New York, NY, USA: Springer-Verlag New York, Inc., 1997.

[42] O. Pironneau. « Optimal shape design for elliptic systems ». In: *System Modeling and Optimization*. Vol. 38. Lecture Notes in Control and Information Sciences. Springer Berlin Heidelberg, Sept. 1982, pp. 42–66.

[43] V. Poirier. « An efficient radial basis function mesh deformation scheme within an adjoint-based aerodynamic optimization framework ». MA thesis. McGill University, Aug. 2011.

[44] V. Poirier and S. Nadarajah. « Efficient reduced-radial basis function-based mesh deformation within an adjoint-based aerodynamic optimization framework ». In: *AIAA Journal of Aircraft* 53.6 (July 2016), pp. 1905–1921.

[45] J. Reuther, J. J. Alonso, J. Martins, and S. C. Smith. « A coupled aero-structural optimization method for complete aircraft configurations ». In: *37th Aerospace Sciences Meeting and Exhibit*. Paper No. AIAA-99-0187. Jan. 1999.

[46] M. P. Rumpfkeil and D. J. Mavriplis. « Efficient Hessian calculations using automatic differentiation and the adjoint method with applications ». In: *AIAA Journal* 48.10 (Oct. 2010), pp. 2406–2417.

[47] K. Schittkowski. *NLPQLP: a Fortran implementation of a sequential quadratic programming algorithm with distributed and non-monotone line search - User's guide, Version 3.0*. Report. Department of Mathematics, University of Bayreuth, 2009.

[48] S. Schmidt. « Efficient large scale aerodynamic design based on shape calculus ». PhD thesis. University of Trier, Germany, Apr. 2010.

[49] T. W. Sederberg and S. R. Parry. « Free-form deformation of solid geometric models ». In: *SIGGRAPH Computer Graphics* 20.4 (Aug. 1986), pp. 151–160.

[50] D. F. Shanno. « Conditioning of quasi-Newton methods for function minimization ». In: *Mathematics of Computation* 24.111 (1970), pp. 647–657.

[51] L. L. Sherman, A. C. T. III, L. L. Green, P. A. Newman, G. W. Hou, and V. M. Korivi. « First- and second-order aerodynamic sensitivity derivatives via automatic differentiation with incremental iterative methods ». In: *Journal of Computational Physics* 129.2 (Dec. 1996), pp. 307–331.

[52] A. Skajaa. « Limited memory BFGS for nonsmooth optimization ». MA thesis. New York University, Jan. 2010.

[53] G. M. Streuber and D. W. Zingg. « Investigation of multimodality in aerodynamic shape optimization based on the Reynolds-averaged Navier-Stokes equations ». In: *35th AIAA Applied Aerodynamics Conference*. Paper No. AIAA 2017-3752. June 2017.

[54] S. Tatsumi, L. Martinelli, and A. Jameson. « A new high resolution scheme for compressible viscous flows with shocks ». In: *33rd AIAA Aerospace Sciences Meeting and Exhibit*. Paper No. AIAA 95-0466. Jan. 1995.

[55] A. Wächter and L. T. Biegler. « On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming ». In: *Mathematical Programming* 106.1 (Mar. 2006), pp. 25–57.

[56] B. Walther and S. Nadarajah. « Optimum shape design for multirow turbomachinery configurations using a discrete adjoint approach and an efficient radial basis function deformation scheme for complex multiblock grids ». In: *ASME Journal of Turbomachinery* 137.8 (Aug. 2015), p. 081006.

[57] D. X. Wang and L. He. « Adjoint aerodynamic design optimization for blades in multistage turbomachines—part I: methodology and verification ». In: *ASME Journal of Turbomachinery* 132.2 (Oct. 2010), p. 021011.