applied sciences

MDPI

Article

Interactive Airfoil Optimization Using Parsec Parametrization and Adjoint Method

Marek Belda * and Tomáš Hyhlík

Department of Fluid Dynamics and Thermodynamics, Faculty of Mechanical Engineering, Czech Technical University in Prague, Technická 1902/4 Prague 6–Dejvice, 160 00 Prague, Czech Republic
* Correspondence: marek.belda@fs.cvut.cz; Tel.: +420-602-663-423

Featured Application: The potential application of the work presented in this article lies in the preliminary airfoil and wing design and optimization. It is in this area that the strengths of this approach (negligible computation cost, robustness, ...) can be maximally exploited while the weaknesses that hinder it can be minimized. The code is ready to use for engineering practice.

Abstract: In the development of interactive aerodynamic optimization tools, the need to reduce the computational complexity of flow calculations has arisen. Computational complexity can be reduced by estimating the flow variables using machine learning, but that approach has a number of hindrances. Avoiding these hindrances through lowering the computational complexity by stating the assumptions of inviscid incompressible potential flow is the focus of this article. The assumptions used restrict the applicability of this approach to only specific cases, but in engineering practice, these cases are quite widespread. The assumptions allowed the coupling of the adjoint method with parsec parametrization and the panel method, yielding a highly computationally efficient and robust tool for optimizing an airfoil's lift coefficient (C_y). The optimization of the NREL S809 airfoil was carried out, and the results were verified using the Xfoil 6.99 software. The Xfoil verification showed that by making minimal changes to the airfoil's shape, the C_y and lift-to-drag ratios were significantly improved. The improvement magnitude was over 94% for a 0 deg angle of attack (AoA) and over 16% for 6.2 deg AoA. This indicates an improvement in performance that is similar to that of some genetic algorithms, but with computational costs that are many orders of magnitude lower.

Keywords: aerodynamic shape optimization; airfoil design; adjoint method; parsec parametrization; computational fluid dynamics; interactive optimization

Citation: Belda, M.; Hyhlík, T. Interactive Airfoil Optimization Using Parsec Parametrization and Adjoint Method. *Appl. Sci.* **2024**, *14*, 3495. https://doi.org/10.3390/app14083495

Academic Editor: Zhaosheng Yu

Received: 17 March 2024
Revised: 16 April 2024
Accepted: 17 April 2024
Published: 21 April 2024

1. Introduction

The process used in the aerodynamic optimization of an airfoil relies on two main features. The first is a computationally efficient optimization method that can effectively minimize (or maximize) the cost function. The second feature, which is perhaps even more important than the first one, is the ability to represent an airfoil in terms of design variables (to parametrize the airfoil). Currently, there are many parametrizations for airfoils and many optimization methods, yielding dozens of possible combinations. However, the ability to perform CFD-based optimization reliably and reasonably quickly relies heavily on the discovery of the adjoint method in the 1970s and 1980s and its development in the 1990s and early 2000s.

Currently, the objective of aerodynamic optimization research and development is shifting towards fast interactive design optimization [1], where the results of the optimization have to be known almost instantly. The CFD-based adjoint optimization method, which is now an extremely well thought-out and proven concept that is implemented in both commercial (like Ansys Fluent) and open source (such as OpenFOAM) big CFD codes, is not well suited to this task. This is due to the vast computational complexity of

modern-day CFD simulations, which makes the optimization too slow to be considered interactive, even with the use of the most efficient optimization methods [1]. It follows that the computational complexity of both the flow solution and the optimization algorithm must be substantially reduced to allow interactive design optimization [1]. This can be conducted in at least two ways. The first is presented in [1] and its references and involves the use of modern machine learning (ML) methods to estimate the flow variables (and sometimes even adjoint variables when coupled with the adjoint method) and force coefficients. The other way, which is presented in this article, is to stick to the adjoint method for optimization while making some assumptions about the flow, which allows the use of much simpler (and thus faster) methods for solving the flow variables, such as the panel method. This approach (when cautiously formulated) almost completely avoids three main hindrances of the ML approach: extrapolatory predictions, the curse of dimensionality, and the need for vast sets of good-quality training data [1]. However, in the adjoint-based assumption approach, great care has to be taken when placing the design point and simplifying the flow, as some important flow phenomena may be omitted in the simplification. This means that for some problems, especially where compressible and/or viscous flow must be assumed, it may be more feasible to use ML methods to deal with the challenges of interactive optimization.

The adjoint method is a computationally efficient optimization method, and thanks to its efficiency and accuracy, it is well suited for optimization in fluid mechanics, as stated in [2,3]. However, its complexity in both the mathematical apparatus and the programming implementation made initial progress after its proposal rather slow-paced [4]. Since its proposal, it has been successfully used in a variety of optimization problems in the field of fluid mechanics (both external and internal), such as those in [2–7]. The adjoint method comes from the theory of optimal control, as stated in [3,8], and was first proposed by Olivier Pironneau (*1945) and Anthony Jameson (*1934) [7]. Jameson developed and successfully used the first optimization code based on the adjoint approach [9]. The adjoint method, unlike the finite difference method, evaluates the gradient of the cost function indirectly using the adjoint variables [2,3,8]. Consequently, the computational effort tied to the computation of the gradient of the cost function is not coupled to the number of design variables and is always roughly equal to the computation of two flow solutions [2,3,8]. Despite the computational efficiency of the adjoint method, it is good practice to parametrize the optimized geometry with an appropriate number of design variables and to avoid the use of redundant ones, thereby preventing (or at least minimizing) problems related to the geometrical or physical infeasibility of the solution obtained. This is particularly applicable when the original problem has to be simplified for optimization.

To obtain the design variables required by the optimization method, there are, at least in the case of an airfoil, literally dozens of parametrization methods [10,11]. From those, parsec parametrization is only one of the many well-proven parametrizations available. Those parametrizations involve 4- and 5-digit NACA series and their modifications [12], class shape transformation (CST) [13], the analytic equation, as presented in [14], and so on. From those parametrizations, the 4- and 5-digit NACA series and analytic equation from [14] have less than or equal to six parameters. CST can have an arbitrary number of parameters, and parsec has 11 or 12, depending on its variant [13,15]. There are even some modified parsec methods [15]. The parsec method has been extensively used for airfoil optimization, such as in [13,16–18]. It has strong control over the airfoil's leading edge, crest locations, and curvatures at the crest locations [15]. On top of that, its parameters have a clear and straightforward geometrical meaning, giving better insight into the airfoil shape. Parsec is a very powerful parametrization for covering general airfoil shapes with a reasonable amount of design variables [10,19]. Its performance in geometrical accuracy is roughly comparable to that of the Bezier, RBF, and Hicks–Henne parametrizations, with a similar number of design variables [11]. In contrast, its performance in the accuracy of aerodynamical properties is comparable to all the other parametrization methods covered in [11], with a similar number of design variables. However, parsec lacks precise control of

the trailing edge shape [10], but this can be beneficial in this case, as the trailing edge shape has a very influential position in the panel method that may not be related to real physical phenomena.

For this article, the panel method proposed by Hess and Smith in 1966 [20] was used as a flow solver, allowing a very low computational cost. The simplicity of the panel method means that great care must be taken when selecting the parametrization. The ability to prescribe a nearly arbitrary trailing edge geometry (which is possible with many parametrizations) may hinder the practical usability of the optimization, either through manufacturing difficulties or through the flow phenomena the panel method does not account for. On top of that, too many design variables or excessively strong control over the trailing edge shape can lead to geometrically impossible (distorted) shapes arising during optimization. The parsec parametrization was used because a proven parametrization method was needed, and regarding the flow solver used, the assurance of the geometrical and physical feasibility of the resulting shape was required.

The next thing to consider is the uncertainty of the input data, together with the finite manufacturing precision. The dependence of the optimal shape on the input data uncertainties, together with the influence that the finite manufacturing precision has on the performance of both the original and optimized airfoils would be a very interesting research topic. Many methods for assessing the influence of uncertain operating conditions on the solution exist. Those methods and their possible usage can be found for example in [21,22]. This topic, however, is so broad that a full-length article would be needed to cover it in appropriate detail; thus, it will not be discussed here.

The main goal of this article is to study the possibility of creating an interactive airfoil optimization procedure that runs very quickly; this procedure is based on an adjoint method coupled with parsec parametrization. The second goal is to provide Xfoil data for verification of the optimization's efficiency. This article also focuses on comparing the efficiency of the created optimization code and its results with other articles that discuss 2D airfoil optimization. The optimization algorithm created in this research can be of great value in all fields that use airfoils. These include not only aircraft and wind turbine design and development but also vehicle dynamics and stability, as can be seen, for example, in [23].

2. Theory of Adjoint Optimization

2.1. General Basics of Optimization

The optimization methods vary in terms of their computational complexity, cost function requirements, type of search (local or global), etc. In fluid mechanics, great emphasis is placed on computational efficiency, as every solution of the flow field is very computationally expensive [1,3,9], and the computational costs of the optimization methods vary by many orders of magnitude. Global methods are, for this reason, rather sparsely used in fluid mechanics, and the far better choices are usually local methods, as they are far less computationally demanding [1]. Despite their locality, lower computational costs make them more suitable for most CFD optimization problems, and the locality of the search is not a problem when the cost function is carefully defined [3]. The computational complexity of various methods can be seen in Table 1, which underlines the computational efficiency of the adjoint method. The simplex method seems even more efficient, but Table 1 does not provide complete information about the computational demands of the simplex method. The data in Table 1 were obtained from [2] and as the results of a theoretical analysis of optimization algorithms. Those data are valid when the algorithms proceed normally. In the case of the simplex method, however, the startup of the method is computationally more expensive, and the computation cost of the first iteration is equal to $m + 1$ (simplex in m-dimensional space has $m + 1$ vertices). On top of that, something called "stalled convergence" can be expected to happen in the simplex method. Stalled convergence means that the algorithm gets stuck, possibly far from the desired extremum. Solving stalled convergence in the case of the simplex method means rescaling the simplex; thus,

the computational complexity of evading the stalled convergence is equal to m (one vertex from the original simplex can be kept). This means that the computational complexity from Table 1 is the absolute best-case scenario for the simplex method, and the real computational complexity may be much higher. The other algorithms do not particularly suffer from this problem. The symbols used in Table 1 are defined as follows:

m... number of parameters (dimensionality of design space);
k... number of individuals in each population (usually between 10 and 1000).

Table 1. Computation cost of 1 optimization iteration—compared to single flow solution [2].

Optimization Method	RSM	Genetic Algorithm	Finite Difference Method	Adjoint Method	Simplex Method
Computational complexity	m^2	k	$m+1$	≈ 2	≈ 1

2.2. Adjoint Method: Continuous and Discrete Approach

There are two main approaches to adjoint optimization: continuous and discrete [3,4]. They differ in the sequence of mathematical operations used during their derivation. Each of those two approaches can then be treated from either a Lagrange or a duality viewpoint as both yield the same results [7]. In this paper, the Lagrange viewpoint is used as it connects more naturally to the constrained optimization.

The continuous approach favored by Jameson seems more natural from the perspective of physics and continuum mechanics. In this approach, the cost function of a continuous problem is defined first. Then, the adjoint partial differential equations (PDEs) are derived from the continuous flow equations, together with their boundary conditions (BCs) and possible initial conditions (ICs). All the equations are then discretized separately and numerically solved. After solving those equations, it is possible to compute the gradient of the cost function with respect to the design variables. In this approach, great care has to be taken when deriving the adjoint equations, especially their BCs [3]. This makes this approach very complex and challenging in terms of math, although it can be computationally more efficient than the discrete version and can have lower memory requirements [4]. The continuous version is also critical for understanding the behavior of the adjoint equations and their physical significance and for assessing the possibility of ill-posed problems [3,4]. The continuous adjoint method workflow is shown in Figure 1.

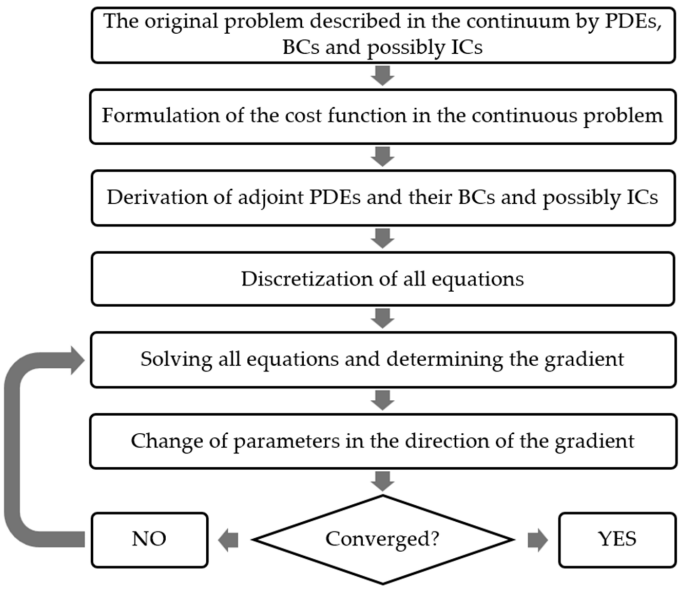

Figure 1. Flowchart of the continuous adjoint method.

The discrete adjoint method differs in terms of the order of operations. The beginning is the same as in the continuous approach, i.e., the definition of the cost function in the continuous problem. After that, the continuous problem is discretized as a whole. This means the discretization of the fluid flow PDEs with the corresponding BCs and ICs implemented and the formulation of the discrete analog to the continuous cost function as a function of many variables. Only then is the adjoint set of algebraic equations derived from the discretized flow PDEs. The advantages and disadvantages of this approach can be found in [4]. This approach avoids much of the complexity of the continuous approach, but great care must be taken when discretizing the problem. The discrete adjoint method workflow is shown in Figure 2.

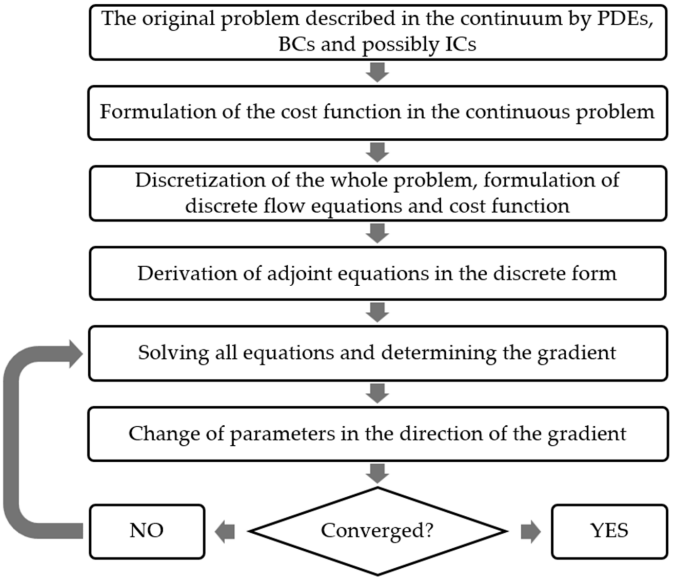

Figure 2. Flowchart of the discrete adjoint method.

2.3. Discrete Adjoint Method from the Lagrange Point of View: Governing Equations

In this article, the nomenclature defined by Jameson in [3] is used. Thus, $\boldsymbol{w}$ is a set of dependent variables, which are usually velocity components at the integration points, pressure at the integration points, and the like. In the panel method, the meaning of $\boldsymbol{w}$ is a bit more complex and is clarified later. $\boldsymbol{F}$ is the set of independent parameters (design variables) undergoing optimization; in this case, these are the parsec parameters describing the shape of an airfoil. Vectors $\boldsymbol{w}$ and $\boldsymbol{F}$ are implicitly tied via equations $\boldsymbol{R}$, which allows the computing of $\boldsymbol{w}$ from the known parameters $\boldsymbol{F}$. The fixed parameters (like viscosity, etc.) are not mentioned here as they do not change during optimization. With all the variables defined, it is possible to define the discrete version of the cost function I, where $\boldsymbol{w}$ and $\boldsymbol{F}$ are subject to constraints $\boldsymbol{R}$. This leads to the classical constrained extremum problem solved via the Lagrange multipliers $\boldsymbol{\lambda}$.

$$\boldsymbol{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}, \quad \boldsymbol{F} = \begin{bmatrix} F_1 \\ \vdots \\ F_m \end{bmatrix}, \quad \boldsymbol{R} = \begin{bmatrix} R_1 \\ \vdots \\ R_n \end{bmatrix} = \boldsymbol{0}, \quad I = I(\boldsymbol{w}, \boldsymbol{F}), \quad \boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{bmatrix} \tag{1}$$

$$\phi(\boldsymbol{w}, \boldsymbol{F}, \boldsymbol{\lambda}) = I(\boldsymbol{w}, \boldsymbol{F}) + \boldsymbol{\lambda}^T \cdot \boldsymbol{R}(\boldsymbol{w}, \boldsymbol{F}) \tag{2}$$

$$grad(\phi) = \left(\frac{\partial \phi}{\partial w_i}; \frac{\partial \phi}{\partial F_j}; \frac{\partial \phi}{\partial \lambda_k} \right) = \left(\frac{\partial I}{\partial w_i} + \lambda_l \cdot \frac{\partial R_l}{\partial w_i}; \frac{\partial I}{\partial F_j} + \lambda_l \cdot \frac{\partial R_l}{\partial F_j}; R_k \right) \tag{3}$$

From the gradient, the total differential of ϕ can be defined. However, as all variables in the function ϕ are treated as independent in the Lagrange approach, the gradient of ϕ

with respect to $\boldsymbol{F}$ can only be solved when the first and last terms in Equation (4) are equal to zero. From there, two systems of Equations (5) and (6) arise, containing 2n equations in total, which makes it possible to compute $\boldsymbol{w}$ and $\boldsymbol{\lambda}$.

$$d\phi = \left(\frac{\partial I}{\partial w_i} + \lambda_l \cdot \frac{\partial R_l}{\partial w_i}\right) dw_i + \left(\frac{\partial I}{\partial F_j} + \lambda_l \cdot \frac{\partial R_l}{\partial F_j}\right) dF_j + R_k d\lambda_k \tag{4}$$

$$\boldsymbol{R} = \boldsymbol{0} \tag{5}$$

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}}\right)^T \boldsymbol{\lambda} = -\left(\frac{\partial I}{\partial \boldsymbol{w}}\right)^T \tag{6}$$

In the case of linear constraints $\boldsymbol{R}$, Equation (5) can be written in matrix form, and the Jacobi matrix on the left side of Equation (6) can be easily computed. The system of equations then forms two systems of linear algebraic equations (LAEs): (8) and (9).

$$\boldsymbol{R} = \boldsymbol{A}\boldsymbol{w} - \boldsymbol{b} = \boldsymbol{0} \quad \rightarrow \quad \boldsymbol{A}\boldsymbol{w} = \boldsymbol{b}, \quad \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}} \equiv \boldsymbol{A} \tag{7}$$

$$\boldsymbol{A}\boldsymbol{w} = \boldsymbol{b} \tag{8}$$

$$\boldsymbol{A}^T \boldsymbol{\lambda} = -\left(\frac{\partial I}{\partial \boldsymbol{w}}\right)^T \tag{9}$$

After solving for $\boldsymbol{w}$ and $\boldsymbol{\lambda}$, it is easy to obtain the gradient of ϕ with respect to $\boldsymbol{F}$. It can be proven that this gradient is identically equal to the gradient of I, as the constraints $\boldsymbol{R}$ must always be satisfied (5). Then, the simple algorithm of the steepest descent is applied to obtain the new approximation of the optimal parameters (11).

$$G_j = \frac{\partial \phi}{\partial F_j} = \frac{\partial I}{\partial F_j} + \lambda_l \cdot \frac{\partial R_l}{\partial F_j} \tag{10}$$

$$\boldsymbol{F}_{(n+1)} = \boldsymbol{F}_{(n)} + \zeta \boldsymbol{G} \tag{11}$$

3. Parsec Parametrization

Parsec parametrization was chosen for the optimization as it provides good accuracy for general airfoil shapes in terms of geometry and fluid flow characteristics [10,11]. Its parameters also have clear and straightforward geometrical meaning, allowing easy fixation of certain geometric quantities and better engineering insight into the airfoil shape. In the basic parsec parametrization, there are 11 parameters [10]. For the optimization, a slightly modified version of parsec, presented in [13], was used, allowing better control of the airfoil's shape near the leading edge [13]. The parameters are very similar to the ones used by the basic parsec; the only difference is that the leading edge radius is prescribed for the upper and lower surfaces separately. The geometrical meaning of those parameters is shown in Table 2 and Figure 3. The shape of the airfoil is then described by Equations (12) and (13) as a linear combination of base functions [10]. The coefficients for this linear combination are computed using Equations (14) and (15), as stated in [13].

$$y = \sum_{i=1}^{6} a_i \cdot x^{i-\frac{1}{2}} \tag{12}$$

$$y = \sum_{i=1}^{6} b_i \cdot x^{i-\frac{1}{2}} \tag{13}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ x_{lo}^{1/2} & x_{lo}^{3/2} & x_{lo}^{5/2} & x_{lo}^{7/2} & x_{lo}^{9/2} & x_{lo}^{11/2} \\ 1/2 & 3/2 & 5/2 & 7/2 & 9/2 & 11/2 \\ \frac{1}{2}x_{lo}^{-1/2} & \frac{3}{2}x_{lo}^{1/2} & \frac{5}{2}x_{lo}^{3/2} & \frac{7}{2}x_{lo}^{5/2} & \frac{9}{2}x_{lo}^{7/2} & \frac{11}{2}x_{lo}^{9/2} \\ -\frac{1}{4}x_{lo}^{-3/2} & \frac{3}{4}x_{lo}^{-1/2} & \frac{15}{4}x_{lo}^{1/2} & \frac{35}{4}x_{lo}^{3/2} & \frac{63}{4}x_{lo}^{5/2} & \frac{99}{4}x_{lo}^{7/2} \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \end{bmatrix} = \begin{bmatrix} y_{te} \\ y_{lo} \\ tg\left(\alpha_{te} + \frac{\beta_{te}}{2}\right) \\ 0 \\ y_{xxlo} \\ -\sqrt{2r_{lo}} \end{bmatrix} \quad (14)$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ x_{up}^{1/2} & x_{up}^{3/2} & x_{up}^{5/2} & x_{up}^{7/2} & x_{up}^{9/2} & x_{up}^{11/2} \\ 1/2 & 3/2 & 5/2 & 7/2 & 9/2 & 11/2 \\ \frac{1}{2}x_{up}^{-1/2} & \frac{3}{2}x_{up}^{1/2} & \frac{5}{2}x_{up}^{3/2} & \frac{7}{2}x_{up}^{5/2} & \frac{9}{2}x_{up}^{7/2} & \frac{11}{2}x_{up}^{9/2} \\ -\frac{1}{4}x_{up}^{-3/2} & \frac{3}{4}x_{up}^{-1/2} & \frac{15}{4}x_{up}^{1/2} & \frac{35}{4}x_{up}^{3/2} & \frac{63}{4}x_{up}^{5/2} & \frac{99}{4}x_{up}^{7/2} \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} = \begin{bmatrix} y_{te} \\ y_{up} \\ tg\left(\alpha_{te} - \frac{\beta_{te}}{2}\right) \\ 0 \\ y_{xxup} \\ \sqrt{2r_{up}} \end{bmatrix} \quad (15)$$

Table 2. Parsec parameters and their meaning.

Basic Parsec Parametrization [10]		Parsec Variant Used for Optimization [13]	
Parameter	**Meaning**	**Parameter**	**Meaning**
---	---	r_{lo} [1]	Lower surface leading edge radius
x_{lo} [1]	Lower surface crest location—horizontal position	x_{lo} [1]	Lower surface crest location—horizontal position
y_{lo} [1]	Lower surface crest location—vertical position	y_{lo} [1]	Lower surface crest location—vertical position
y_{xxlo} [1]	Lower surface crest curvature	y_{xxlo} [1]	Lower surface crest curvature
r_{le} [1]	Leading edge radius	r_{up} [1]	Upper surface leading edge radius
x_{up} [1]	Upper surface crest location—horizontal position	x_{up} [1]	Upper surface crest location—horizontal position
y_{up} [1]	Upper surface crest location—vertical position	y_{up} [1]	Upper surface crest location—vertical position
y_{xxup} [1]	Upper surface crest curvature	y_{xxup} [1]	Upper surface crest curvature
α_{te}[°]	Trailing edge direction	α_{te}[°]	Trailing edge direction
β_{te}[°]	Trailing edge wedge angle	β_{te}[°]	Trailing edge wedge angle
y_{te} [1]	Trailing edge location—vertical position	y_{te} [1]	Trailing edge location—vertical position
Δy_{te} [1]	Trailing edge thickness	Δy_{te} [1]	Trailing edge thickness

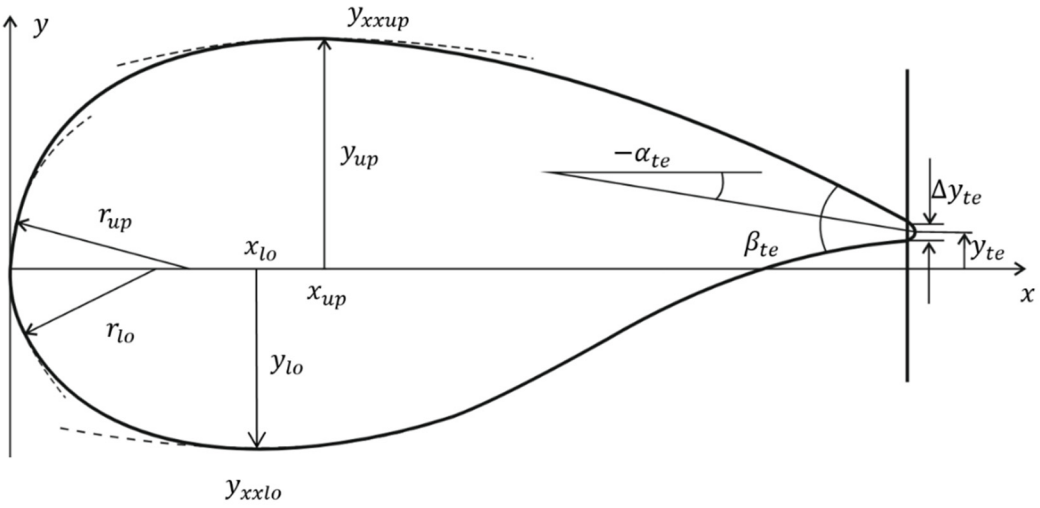

Figure 3. Parsec parameters of an airfoil and their geometrical meaning.

It has been assumed that the trailing edge thickness (Δy_{te}) is always equal to zero (sharp trailing edge) and is therefore not a design variable. This may limit the applicability in some cases, but the benefits in terms of the robustness of the algorithm were deemed to outweigh this drawback. The sharp trailing edge makes it easier to properly enforce the Kutta condition and prevents the optimization from making illegal changes to the trailing edge geometry. The area around the trailing edge is where the viscous effects are usually the most profound since the wake tends to start forming there. Because the optimization cannot account for those viscous effects (and especially the wake formation), it was decided to restrain it in this area as much as possible. On top of that, the trailing edge can be tweaked in many ways, most of them far exceeding the capabilities of the parsec parametrization, as the lack of trailing edge control is deemed to be one of its main weaknesses [10]. For the trailing edge fine-tuning, methods capable of properly capturing viscous phenomena must be used. Vector $\boldsymbol{F}$ of the 11 design variables based on parsec parametrization is given as follows.

$$\boldsymbol{F} = \left[r_{lo}, x_{lo}, y_{lo}, y_{xxlo}, r_{up}, x_{up}, y_{up}, y_{xxup}, \alpha_{te}, \beta_{te}, y_{te}\right]^T \tag{16}$$

4. Airfoil Optimization: Governing Equations and Implementation

4.1. Potential Flow and Panel Method

For modeling the flow around an airfoil, the inviscid incompressible potential flow assumption was used. This assumption may be rather restrictive but allows the use of potential flow theory and the panel method described in [24]. The incompressibility condition may be weakened in this case by the fact that the optimization presented in Section 4 holds, as long as the Prandtl–Glauert compressibility correction rule is valid. According to [25], this can be stated for flows with $Ma_\infty \leq 0.5$, although great caution should be taken when dealing with flows with $Ma_\infty > 0.4$ in this manner. To obtain the potential flow solution, the Hess–Smith panel method [20] was used. An explanation of the governing equations of potential flow theory, together with the theory and governing equations of the panel method, can be found in [20,24]. After employing the equations from [20,24], the lift coefficient (C_y) can be defined and rewritten in dimensionless variables (17); the definition of the dimensionless variables can be found in (18).

$$C_y = -\oint_l \frac{1}{L}\left(\gamma - \hat{u}^2 - \hat{v}^2\right)\cdot dx \tag{17}$$

$$\hat{u} = \frac{u}{c_\infty}, \qquad \hat{v} = \frac{v}{c_\infty}, \qquad \hat{p}_\infty = \frac{2\cdot p_\infty}{\rho\cdot c_\infty^2}, \qquad \gamma = 1 + \hat{p}_\infty \tag{18}$$

For the easier derivation of the equations in the following sections, the chord length L can be factored out from the integral in (17) and transferred to the left side of the equation, yielding the relationship for the continuous cost function (19).

$$I = L\cdot C_y = -\oint_l \left(\gamma - \hat{u}^2 - \hat{v}^2\right)\cdot dx \tag{19}$$

Now, it is possible to discretize the whole problem, including the governing harmonic equation for velocity potential, as presented in [24] and the cost function (19). The discrete version of the cost function is obtained simply by discretizing the integral in (19). To obtain the discretized flow equations, a Hess–Smith panel method [20] is used. The shape of the whole airfoil is discretized into many line segments (panels), leading to the panel method equations, as derived by [20]. The dependent variables $\boldsymbol{w}$ in the optimization are the source on each panel and the circulation around the airfoil. The governing equations of the panel

method are presented below, together with the discrete version of the cost function. The subscript i is the index of the panel. The panel numbering is shown in Figure 4.

$$I = L \cdot C_y = -\sum_{i=1}^{N} \left(\gamma - \hat{c}_{ti}^2\right) \cdot \Delta x_i \tag{20}$$

$$\Delta x_i = x_{i+1} - x_i, \qquad x_{N+1} \equiv x_1, \qquad \hat{c}_{ti}^2 = \hat{u}_i^2 + \hat{v}_i^2 \tag{21}$$

$$\boldsymbol{A}\boldsymbol{w} = \boldsymbol{b}, \qquad \boldsymbol{w} = [q_1, \ldots, q_N, \Gamma]^T \tag{22}$$

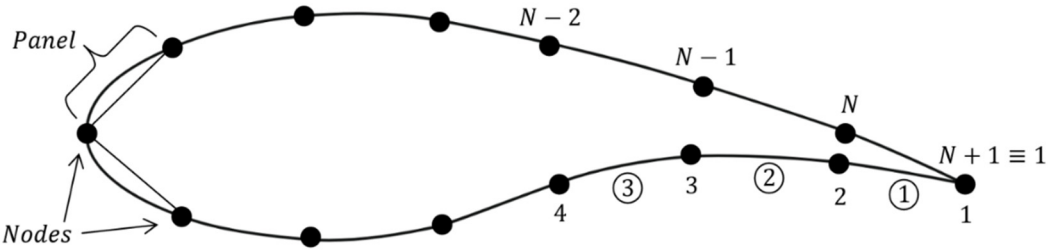

Figure 4. Node and panel numbering in the panel method.

The exact definition of the elements of matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$ can be found in [20]. From that definition, it can be seen that the value c_∞ can be factored out of each term in the vector $\boldsymbol{b}$, and the whole system of LAEs (22) can be divided by it, yielding Equation (23).

$$\boldsymbol{A}\hat{\boldsymbol{w}} = \hat{\boldsymbol{b}}, \qquad \hat{\boldsymbol{w}} = \frac{\boldsymbol{w}}{c_\infty}, \qquad \hat{\boldsymbol{b}} = \frac{\boldsymbol{b}}{c_\infty} \tag{23}$$

The tangent velocity on each panel can be extracted directly using the panel method [20]. That relationship can then be divided by c_∞, yielding Equation (24), which can be substituted into the cost function (20). This last modification completes the arrangement of equations needed to derive the adjoint optimization.

$$\begin{aligned} \hat{c}_{ti} = \tfrac{c_{ti}}{c_\infty} = \cos(\theta_i - \alpha) + \sum_{j=1}^{N} \tfrac{\hat{w}_j}{2\pi} \left(\sin(\theta_i - \theta_j)\beta_{ij} - \cos(\theta_i - \theta_j) \ln \tfrac{r_{ij+1}}{r_{ij}}\right) \\ + \tfrac{\hat{w}_{N+1}}{2\pi} \sum_{j=1}^{N} \left(\sin(\theta_i - \theta_j) \ln \tfrac{r_{ij+1}}{r_{ij}} + \cos(\theta_i - \theta_j)\beta_{ij}\right) \end{aligned} \tag{24}$$

4.2. Adjoint-Based Optimization with Panel Method and Parsec Parametrization

To obtain the equations for the adjoint optimization, it is necessary to start with Equations (9) and (23). Matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$ are known from [20]. The only part that must be specified in Equation (9) is its right-hand side, which contains, in this context, the gradient of I with respect to $\hat{\boldsymbol{w}}$.

$$\left(\frac{\partial I}{\partial \hat{\boldsymbol{w}}}\right) = 2(\boldsymbol{\Delta x}\hat{\boldsymbol{c}}_{\boldsymbol{t}})^T \boldsymbol{C}, \qquad (\boldsymbol{\Delta x}\hat{\boldsymbol{c}}_{\boldsymbol{t}}) = \begin{bmatrix} \Delta x_1 \hat{c}_{t1} \\ \vdots \\ \Delta x_N \hat{c}_{tN} \end{bmatrix} \tag{25}$$

In (25), $\boldsymbol{C}$ is defined as follows:

$$C_{ik} = \frac{\partial \hat{c}_{ti}}{\partial \hat{w}_k} = \frac{1}{2\pi}\left(\sin(\theta_i - \theta_k)\beta_{ik} - \cos(\theta_i - \theta_k) \ln \frac{r_{ik+1}}{r_{ik}}\right) \qquad for\ k = 1 \div N \tag{26}$$

$$C_{i,N+1} = \frac{\partial \hat{c}_{ti}}{\partial \hat{w}_{N+1}} = \frac{1}{2\pi} \sum_{j=1}^{N} \left(\sin(\theta_i - \theta_j) \ln \frac{r_{ij+1}}{r_{ij}} + \cos(\theta_i - \theta_j)\beta_{ij}\right) \tag{27}$$

The definition of the gradient comes directly from (10), and its constituent derivatives are specified in (28) and (29).

$$\frac{\partial I}{\partial F_k} = 2\sum_{i=1}^{N} \Delta x_i \hat{c}_{ti} \frac{\partial \hat{c}_{ti}}{\partial F_k} \tag{28}$$

$$\frac{\partial R_l}{\partial F_k} = \frac{\partial}{\partial F_k}\left(A_{lm}\hat{w}_m - \hat{b}_l\right) = \frac{\partial A_{lm}}{\partial F_k}\hat{w}_m - \frac{\partial \hat{b}_l}{\partial F_k} \tag{29}$$

Now, it is possible to continue processing the individual terms in Equations (28) and (29). The derivatives of the vectors $\hat{\boldsymbol{c}}_{\boldsymbol{t}}$ and $\hat{\boldsymbol{b}}$, and matrix $\boldsymbol{A}$ can be obtained semi-analytically, greatly reducing the number of numerical derivatives computed, which saves a lot of computational costs. It can be shown that only the following derivatives must be computed numerically:

$$\frac{\partial \theta_i}{\partial F_k}, \qquad \frac{\partial \beta_{ij}}{\partial F_k}, \qquad \frac{\partial r_{ij}}{\partial F_k} \tag{30}$$

To evaluate these derivatives, the first forward difference is used. This approximation is first-order accurate but computationally very cheap. Because the code runs in MATLAB R2023b, which uses double precision as standard, a rather small step when evaluating the finite difference can be used without losing precision to rounding errors. It has been found that the value $\delta = 10^{-8}$ gives accurate results and a stable optimization algorithm. The value $\delta = 10^{-9}$ has been tried out without any noticeable improvement in precision or stability. All the numerical derivatives are computed as presented in (31).

$$\frac{\partial r_{ij}}{\partial F_k} \cong \frac{r_{ij}(F_1, \ldots, F_{k-1}, F_k + \delta, F_{k+1}, \ldots, F_m) - r_{ij}(F_1, \ldots, F_m)}{\delta} \tag{31}$$

Now, it is possible to define the following auxiliary variables. These variables arise when obtaining the analytical derivatives of individual terms in (28) and (29). They are not essential for the math, but they make programming implementation easier and far cleaner.

$$dIdFk1_i = -\sin(\theta_i - \alpha)\frac{\partial \theta_i}{\partial F_k} \tag{32}$$

$$dIdFk2_{ij} = \cos(\theta_i - \theta_j)\left(\frac{\partial \theta_i}{\partial F_k} - \frac{\partial \theta_j}{\partial F_k}\right)\beta_{ij} + \sin(\theta_i - \theta_j)\frac{\partial \beta_{ij}}{\partial F_k} \tag{33}$$

$$dIdFk3_{ij} = -\sin(\theta_i - \theta_j)\left(\frac{\partial \theta_i}{\partial F_k} - \frac{\partial \theta_j}{\partial F_k}\right)\ln\frac{r_{ij+1}}{r_{ij}} + \cos(\theta_i - \theta_j)\frac{\left(\frac{\partial r_{ij+1}}{\partial F_k}r_{ij} - r_{ij+1}\frac{\partial r_{ij}}{\partial F_k}\right)}{r_{ij}r_{ij+1}} \tag{34}$$

$$dIdFk4_{ij} = \cos(\theta_i - \theta_j)\left(\frac{\partial \theta_i}{\partial F_k} - \frac{\partial \theta_j}{\partial F_k}\right)\ln\frac{r_{ij+1}}{r_{ij}} + \sin(\theta_i - \theta_j)\frac{\left(\frac{\partial r_{ij+1}}{\partial F_k}r_{ij} - r_{ij+1}\frac{\partial r_{ij}}{\partial F_k}\right)}{r_{ij}r_{ij+1}} \tag{35}$$

$$dIdFk5_{ij} = -\sin(\theta_i - \theta_j)\left(\frac{\partial \theta_i}{\partial F_k} - \frac{\partial \theta_j}{\partial F_k}\right)\beta_{ij} + \cos(\theta_i - \theta_j)\frac{\partial \beta_{ij}}{\partial F_k} \tag{36}$$

With the use of the variables defined in (32)–(36), it is now possible to rewrite all the terms in Equations (28) and (29) in the following manner:

$$\frac{\partial \hat{c}_{ti}}{\partial F_k} = dIdFk1_i + \sum_{j=1}^{N}\left[\frac{\hat{w}_j}{2\pi}\left(dIdFk2_{ij} - dIdFk3_{ij}\right) + \frac{\hat{w}_{N+1}}{2\pi}\left(dIdFk4_{ij} + dIdFk5_{ij}\right)\right] \tag{37}$$

$$\frac{\partial A_{lm}}{\partial F_k} = \frac{1}{2\pi}(dIdFk4_{lm} + dIdFk5_{lm}) \qquad for\ l, m = 1 \div N \tag{38}$$

$$\frac{\partial A_{lN+1}}{\partial F_k} = \frac{1}{2\pi}\sum_{j=1}^{N}\left(dIdFk3_{lj} - dIdFk2_{lj}\right) \qquad for\ l = 1 \div N \tag{39}$$

$$\frac{\partial A_{N+1,m}}{\partial F_k} = \frac{1}{2\pi}\sum_{i=1,N}(dIdFk2_{im} - dIdFk3_{im}) \qquad for\ m = 1 \div N \tag{40}$$

$$\frac{\partial A_{N+1,N+1}}{\partial F_k} = \frac{1}{2\pi}\sum_{i=1,N}\sum_{j=1}^{N}(dIdFk4_{ij} + dIdFk5_{ij}) \tag{41}$$

$$\frac{\partial \hat{b}_l}{\partial F_k} = \cos(\theta_l - \alpha)\frac{\partial \theta_l}{\partial F_k} \tag{42}$$

$$\frac{\partial \hat{b}_{N+1}}{\partial F_k} = -(dIdFk1_1 + dIdFk1_N) \tag{43}$$

Then, the gradient of the cost function with respect to $\boldsymbol{F}$ can be constructed, and the new approximation of the optimal design variables $\boldsymbol{F}$ can be computed. In the gradient $\boldsymbol{G}$, $\hat{\boldsymbol{w}}$ is the solution of (23), and $\boldsymbol{\lambda}$ is the solution of (9), where the gradient term on the right-hand side is given in (25).

$$G_k = 2\sum_{i=1}^{N}\Delta x_i \hat{c}_{ti}\frac{\partial \hat{c}_{ti}}{\partial F_k} + \lambda_l \cdot \left(\frac{\partial A_{lm}}{\partial F_k}\hat{w}_m - \frac{\partial \hat{b}_l}{\partial F_k}\right) \tag{44}$$

$$\boldsymbol{F}_{(n+1)} = \boldsymbol{F}_{(n)} + \zeta \boldsymbol{G} \tag{45}$$

$$\zeta = \frac{\zeta_0}{||\boldsymbol{G}||} \tag{46}$$

The value ζ is the iteration step. For stability reasons, it proved beneficial to define it in the manner shown in (46). This definition is equal to using a unit gradient, which eliminates the stability problems tied to the strong nonlinearity of the cost function in some regions of the design space. The value ζ_0 is the nominal iteration step and is user-defined.

4.3. Resulting Shape Feasibility and Convergence of the Optimization

The programming implementation of the procedure presented in Section 4.1. and Section 4.2. is rather simple but requires the addressing of a few tasks. The first task is to prevent the occurrence of distorted shapes during optimization. In the solution of this problem, the clear geometrical meaning of parsec parameters is very handy. With that meaning, it is possible to simply prescribe the conditions that assure the geometrical feasibility of those shapes. This is carried out by monitoring the values of the parameters (like r_{up}, r_{lo}, x_{up}, x_{lo}, and β_{te}); if they get outside the allowed range, the optimization is immediately stopped. The allowed range can be completely controlled by the user, enabling custom restrictions to be made. On top of that, the airfoil paneling is constructed in such a way that the panel nodes in the top and bottom halves of the airfoil share x coordinates. Because of that, it is easy to compare the y coordinates of the corresponding nodes and to make sure that the y coordinate of the upper-surface node is always greater than that of the lower-surface one. If any self-intersection of the airfoil's surface is detected in this way, the optimization is immediately stopped, providing another layer of protection besides the allowed parameter values.

The next task is convergence monitoring. In the case of this optimization, monotone convergence is required. In the case of the worsening of the value of the cost function between subsequent iterations, a warning is displayed, but the optimization process is not terminated. This allows the procedure the possibility to regain monotone convergence once again.

The final task is to define the convergence criteria. As the inviscid incompressible potential flow is assumed, the C_y may become unbounded, as there is no flow separation allowed other than that at the trailing edge. This means that reaching the true maximum of C_y in the incompressible inviscid scenario may be, and probably is, impossible. This renders the classical convergence criteria for reaching the extremum unusable and also means that geometrically feasible but nonsensical and obviously wrong shapes are possible

if the optimization is allowed to continue for too many iterations (more than a few tens). To stop this from happening, other convergence criteria were implemented. The optimization has three stopping criteria in total. The first states that convergence is reached when a defined improvement, whether absolute or relative, has been made. The second one stops the iteration when a defined change in shape, measured as the root mean square difference from the original shape, has been made. The final one limits the maximum number of iterations allowed.

The proposed criteria, however, cause the optimal result to be influenced by the threshold values chosen. The shape change criterion is the most objective of the three, as it can be interpreted as defining the region of design space for the algorithm to search in. It can also be tackled from the engineering viewpoint, as the baseline shape is usually chosen for a reason, and one may not want to change it much during the optimization. For more objective criteria to be defined, it would be possible to add a simple boundary layer solver based on the von Karman momentum integral equation and Pohlhausen's method to determine whether the flow separation was occurring on the airfoil. The stopping criterion could then be the occurrence of the separation, for example (or, as a variation of this criterion, the reaching of a certain distance from the trailing edge). This modification, however, would bring with it some computation costs, which in this context would be rather large.

The script conducting the optimization proposed in this paper was written in MATLAB R2023b. The script operates according to the flowchart shown in Figure 5.

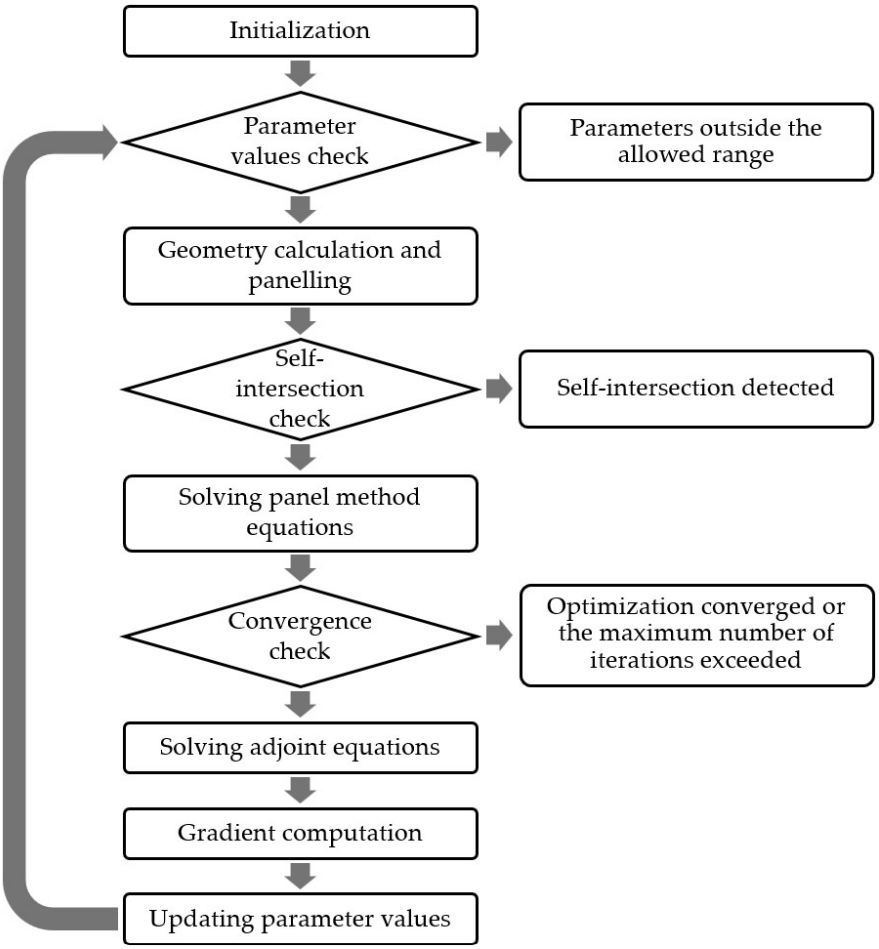

Figure 5. Flowchart of the optimization script.

5. Results and Verification

5.1. Adjoint Optimization Results: Incompressible and Inviscid Flow

The most important results of the optimization, using the example of the NREL S809 airfoil, are presented below in Table 3 and Figure 6. Different airfoils were tried, yielding very similar results. These results show that massive improvement in C_y can be obtained by minimal change in the design variables, which implies minimal changes in the geometry. These minimal changes to the shape allow the assumption that the drag coefficient will not be affected. This was later proven true by the Xfoil 6.99 computation results. The changes in the design variables and C_y from the optimization solver for the nominal angle of attack (AoA) of 0 and 10 deg are presented in Table 3. Optimization settings with 50 iterations and a nominal iteration step $\zeta_0 = 0.0002$ (a very conservative step choice) were used throughout this study. The results have been rounded to four significant digits in the case of the design variables and four decimal places in the case of C_y.

Table 3. Parsec parameters of NREL S809 airfoil before and after the optimization.

Parameter	Original		Optimized	
α [°]	0	10	0	10
r_{lo} [1]	0.0100	0.0100	0.01115	0.01168
x_{lo} [1]	0.3633	0.3633	0.3630	0.3629
y_{lo} [1]	−0.1081	−0.1081	−0.1038	−0.1038
y_{xxlo} [1]	1.526	1.526	1.526	1.526
r_{up} [1]	0.02160	0.02160	0.02123	0.02167
x_{up} [1]	0.3826	0.3826	0.3829	0.3829
y_{up} [1]	0.1018	0.1018	0.1057	0.1057
y_{xxup} [1]	−1.201	−1.201	−1.201	−1.201
α_{te}[°]	−8.500	−8.500	−8.558	−8.558
β_{te}[°]	8.500	8.500	8.499	8.495
y_{te} [1]	0	0	−0.007687	−0.007616
C_y [1]	0.2178	1.4256	0.3507	1.5529
ΔC_y^{ABS} [1]	---	---	0.1329	0.1273
ΔC_y^{REL}[%]	---	---	61.02	8.93

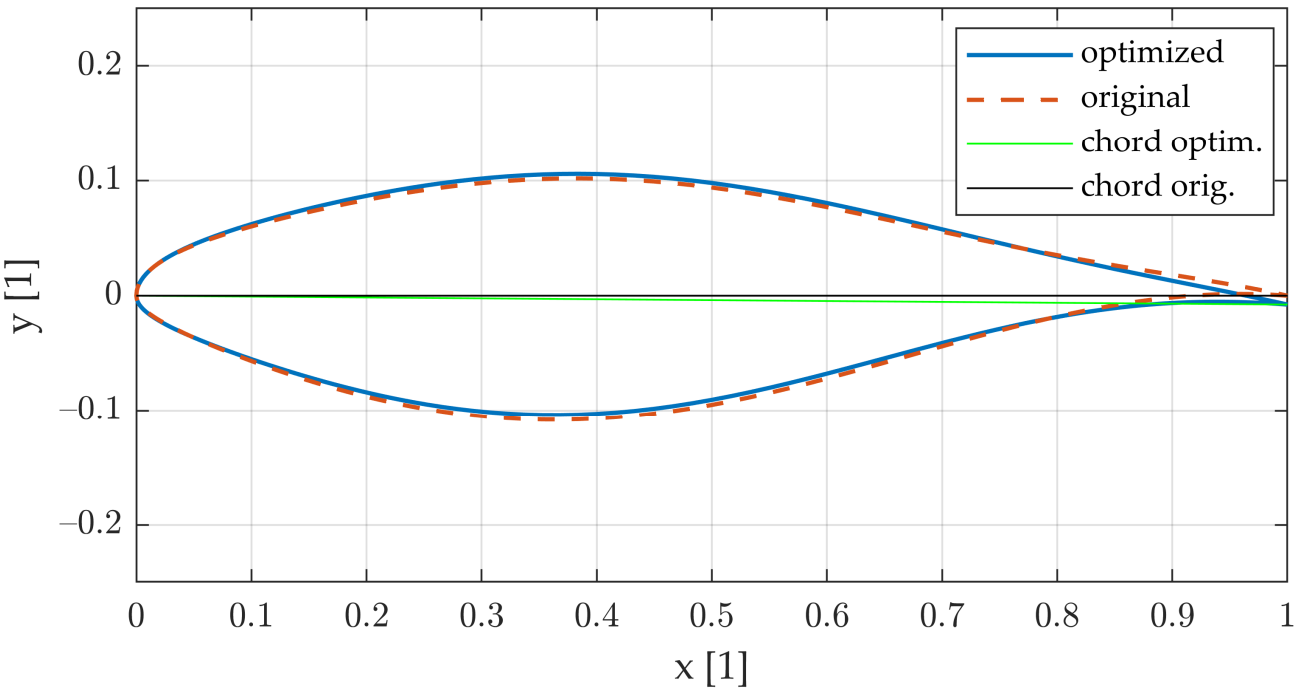

Figure 6. Shape of NREL S809, nominal AoA 0 deg.

5.2. Xfoil Verification Computations: Compressible and Viscous Flow

The results presented in this section were obtained using the well-established Xfoil 6.99 software. The Reynolds and Mach numbers were set to Re = 750,000 and Ma = 0.02, allowing a direct comparison between the obtained results and those published in [13]. The nominal AoA for the adjoint optimization was zero; the resulting values of the design variables can be found in Table 3, Col. 4, and the original and optimized shapes can be seen in Figure 6. The dependency of the optimal shape on AoA is, at least in this case, very weak, as can be seen in Table 3. This means that the value of AoA is not of great importance in this case and that it has negligible effects on the optimization result. The result's lack of dependence on AoA, however, cannot be generalized and has to be assessed on each optimized airfoil separately.

The adjoint optimization results presented in Table 4 were compared to those acquired by Akram and Kim [13], as shown in Table 5. For details regarding the acquisition of the data in Table 5 please see the original paper by Akram and Kim [13]. The CFD computation model, grid, BCs, and other necessary information used for generating the CFD data in Table 5 are described in great detail in [13]. A comparison of the different computation methods used during the research presented in this paper can be seen in Table 6 which directly compares those simulation methods to the available experimental data. For the viscous Xfoil 6.99 simulation, Re = 750,000 and Ma = 0.02 were used to match the experimental data. From this comparison, it can be concluded that the inviscid computation methods consistently overestimate the lift coefficient, which is an expected aspect of their behavior. Also, the accuracy of the inviscid methods can be considered sufficient only when there is no large flow separation occurring since they are unable to predict and account for this phenomenon. If a large separation occurs, their results can qualitatively differ from reality. Those methods cannot be used for computing the drag coefficient; thus, they cannot compute the L/D ratio either. However, as their results are consistent, the MATLAB (R2023b) version can be used as the highly efficient core for adjoint optimization, as long as no large flow separation is expected to occur. It can generally be said that a significant separation starts occurring when the lift coefficient curve starts deviating from the linear progression seen near zero AoA (roughly 6–8 deg AoA for NREL S809 airfoil at the presented values of Re and Ma). As for the compressible and viscous simulation, it can be concluded that this Xfoil 6.99 computation is accurate enough to serve as a verification computation for the optimization in this case. Thus, the benefit of the optimization can be assessed by the compressible and viscous simulation in Xfoil 6.99. When assessed in this way, the benefits should be maintained in the real world.

Table 4. Xfoil results for NREL S809 airfoil—adjoint optimization.

Type of Xfoil Simulation	Incompressible and Inviscid				Compressible and Viscous			
	Original Airfoil	Optimized Airfoil	Absolute Difference	Relative Difference	Original Airfoil	Optimized Airfoil	Absolute Difference	Relative Difference
α [°]	0	0	---	---	0	0	---	---
C_y[1]	0.2137	0.3502	0.1365	+63.9%	0.1480	0.2881	0.1401	+94.7%
C_x[1]	---	---	---	---	0.00883	0.00885	0.00002	+0.2%
C_y/C_x[1]	---	---	---	---	16.76	32.55	15.79	+94.2%
C_M[1]	−0.0578	−0.0778	−0.0200	−34.6%	−0.0425	−0.0638	−0.0213	−50.1%
α [°]	6.2	6.2	---	---	6.2	6.2	---	---
C_y[1]	0.9869	1.1215	0.1346	+13.6 %	0.8108	0.9415	0.1307	+16.1%
C_x[1]	---	---	---	---	0.01347	0.01337	−0.00010	−0.7%
C_y/C_x[1]	---	---	---	---	60.20	70.40	10.20	+16.9%
C_M[1]	−0.0778	−0.0974	−0.0196	−25.2%	−0.0449	−0.0641	−0.0192	−42.8%

Table 5. Results for NREL S809—GA optimization by Akram and Kim [13].

	Exp. Data	Xfoil Computation Data				CFD Computation Data	
	Original Airfoil (OSU)	Optimized Airfoil (CST)	Optimized Airfoil (PARSEC)	CST vs. Experiment	PARSEC vs. Experiment	Optimized Airfoil (CST)	CST vs. Experiment
α [°]	6.2	6.2	6.2	---	---	6.2	---
C_y[1]	0.79	0.883	0.87	+11.8%	+10.1%	0.985	+24.6%
C_x[1]	0.0131	0.0134	0.0148	+2.2%	+12.1%	0.0147	+12.2%
C_y/C_x[1]	60.3	65.9	58.8	+9.6%	−2.0%	67	+12.4%

Table 6. Comparison of different computation methods with experiments using the original airfoil.

	Original Airfoil						
	MATLAB Inviscid	Xfoil Inviscid	Xfoil Viscous	Experiment [13,26]	MATLAB Inviscid vs. Experiment	Xfoil Inviscid vs. Experiment	Xfoil Viscous vs. Experiment
α [°]	6.2	6.2	6.2	6.2	---	---	---
C_y[1]	0.9777	0.9869	0.8108	0.79	+23.8%	+24.9%	+2.6%
C_x[1]	---	---	0.01347	0.0131	---	---	+2.8%
C_y/C_x[1]	---	---	60.20	60.30	---	---	−0.2%

From the comparison of the adjoint optimization results with Akram and Kim's results [13], it is possible to conclude that the adjoint optimization in this paper yielded very similar or even better results; however, it obtained them with a computational cost that was more than three orders of magnitude lower. The computation time of the adjoint optimization is 11 seconds, compared to the 272 minutes needed by the GA optimization published in [13]. The computation time could be further reduced to less than 5 seconds by increasing the iteration step and lowering the number of iterations accordingly. This can be easily carried out with no risk of losing stability, as the initially chosen iteration step is very small. The number of iterations in the adjoint optimization should be somewhere between 10 and 50, as stated in [3]; therefore, after doubling the current iteration step and halving the current number of iterations, the optimization would still obey those recommendations. Another way of lowering the computational complexity of the optimization is by coarsening the airfoil paneling. As matrix A is a full matrix with no special properties, the cost of solving the equations is proportional to $(N + 1)^3$. Thus, by lowering the number of panels from 300 (a very fine resolution used throughout this paper) to 200 (which is still an acceptable resolution), it is possible to lower the optimization cost by more than a factor of 3, achieving a computation time that is 4 orders of magnitude shorter when compared to [13]. After applying the suggestions above, by setting the iteration step to $\zeta_0 = 0.0005$, the number of iterations to 20, and the number of panels to 200, the computation time dropped to two seconds, while very similar optimization results were achieved ($\Delta C_y^{REL} = 58$ % for 0 deg AoA). All the computations were carried out using the Intel Core i7-12700H processor (Intel, Santa Clara, CA, USA). The panel method simulations in the optimization were performed in serial (due to the nature of the adjoint method making consecutive changes to the design), but the MATLAB R2023b solver used for solving systems of LAEs, which are by far the most computationally expensive part of the adjoint method, incorporated parallel computation. The abovementioned data and computation times prove that the adjoint method can be highly computationally effective and, despite the simplifications applied during the development of the optimization code, can vastly improve the efficiency of the initial design.

The following figures (Figures 7 and 8) further document the optimization results. The optimization settings were the same as those used for the results presented in Table 3 and Figure 6. The nominal AoA was set to zero. In the following charts, the benefit of the optimization across all the AoAs is visible. The Reynolds and Mach numbers were set to Re = 1,000,000 and Ma = 0 to make them consistent with the available experimental data,

such as those in [26,27]. The experimental data presented in Figures 7 and 8 were taken from [26]. The computations were carried out using the compressible and viscous Xfoil 6.99 simulation.

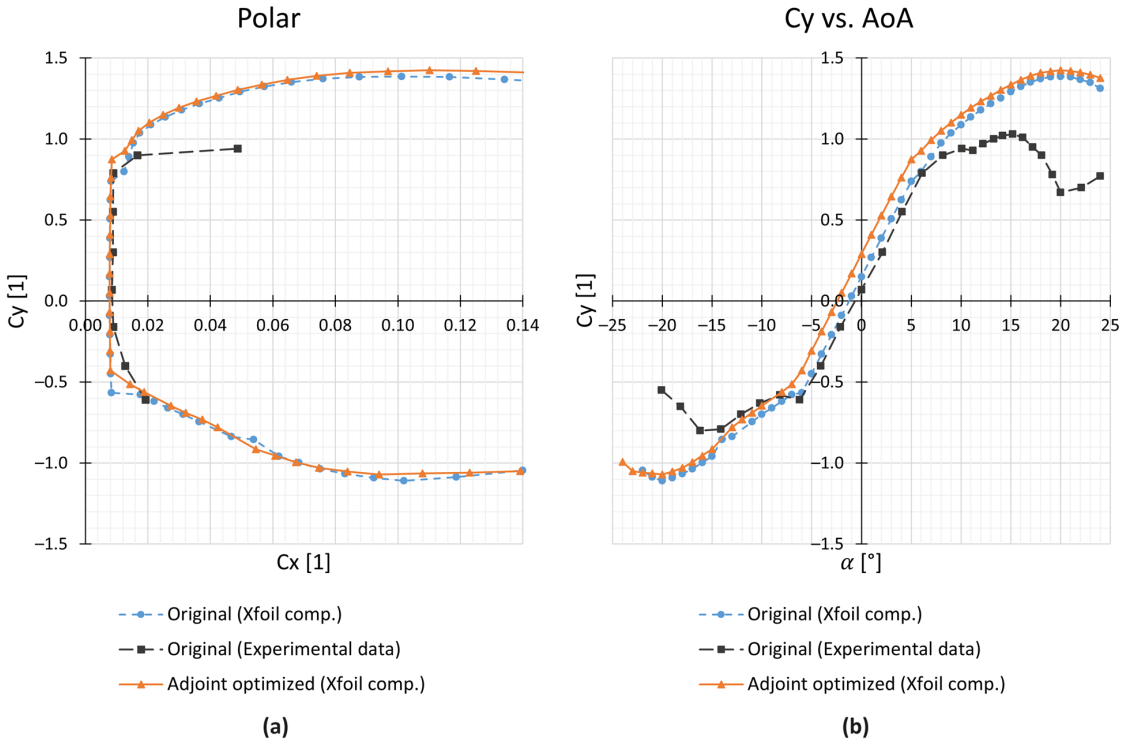

Figure 7. Characteristics of the original and optimized NREL S809 airfoil. Xfoil 6.99 computation for Re = 1,000,000, Ma = 0. Exp. data from [26]. (**a**) Polar of the airfoil, (**b**) dependence of C_y on AoA.

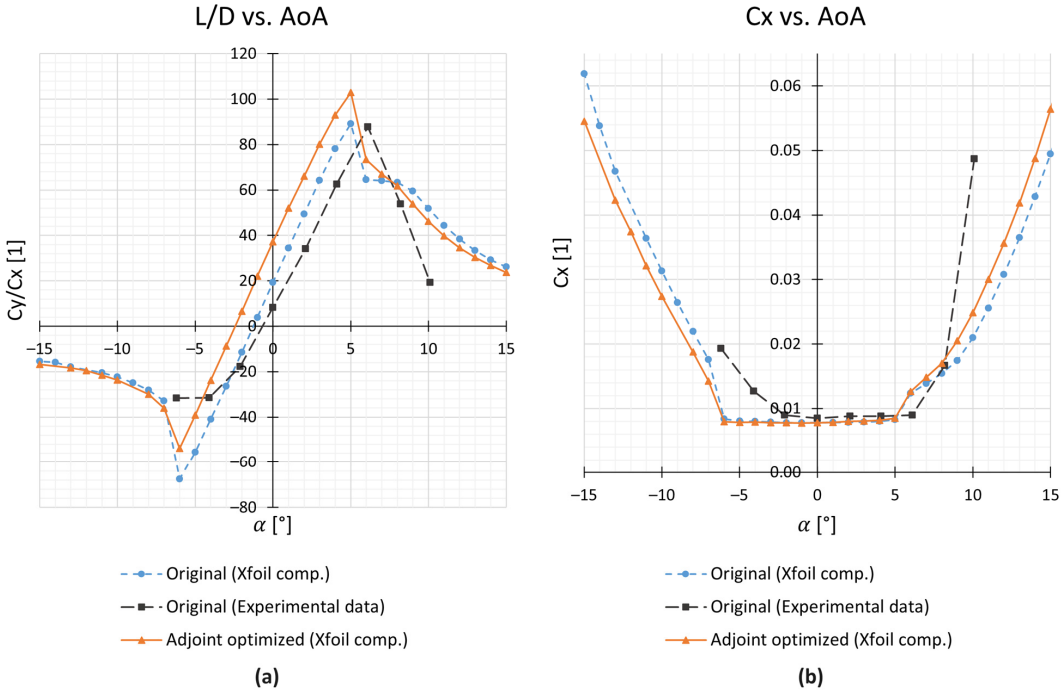

Figure 8. Characteristics of the original and optimized NREL S809 airfoil. Xfoil 6.99 computation for Re = 1,000,000, Ma = 0. Exp. data from [26]. Results beyond (−15; 15) deg AoA omitted. (**a**) Dependence of Lift-to-drag ratio on AoA, (**b**) dependence of drag coefficient on AoA.

6. Conclusions

This article focused on the fast interactive airfoil shape optimization achieved by assuming an incompressible inviscid potential flow. The objective of the optimization was to improve the lift coefficient of the airfoil. The optimization was carried out on the well-known NREL S809 airfoil, which is used on wind turbines. From the results obtained, it can be concluded that coupling the adjoint method with parsec parametrization and the panel method can yield feasible results and can run at a speed that is more than fast enough for it to be considered interactive. The panel method and the adjoint optimization code were written in MATLAB R2023b, and the verification computations were carried out in Xfoil 6.99. From the results obtained, the following statements can be derived:

1. The optimization showed a significant improvement to the objective function C_y. For 0 deg AoA, the improvement was 94.7%, and for 6.2 deg AoA, it was 16.1%.
2. A big improvement was also observed in the L/D ratio, which matched or even exceeded the improvement to C_y, as the optimization did not noticeably affect the C_d.
3. For different AoAs (0 and 10 deg), the resulting shape was almost the same, meaning that the shape optimized for one operating point should work well in a wide range of AoAs, adding to the practical usability of the optimized airfoil.
4. The runtime of the adjoint optimization was, for the same airfoil with the same cost function, orders of magnitude shorter than the runtime of the GA coupled with Xfoil while yielding similar results.

Overall, this coupling seems very promising, but it is not without its difficulties. For example, further work would be needed to incorporate additional constraints into the optimization, most likely via the penalty formulation. However, the approach presented in this paper appears to suit those engineering problems where the inviscid incompressible flow can be assumed to be better than the ML approach because it avoids most of the main hindrances of ML while yielding convincing results.

Author Contributions: Conceptualization, M.B. and T.H.; methodology, M.B.; software, M.B. and T.H.; supervision, T.H.; validation, M.B.; writing—original draft, M.B.; writing—review and editing, M.B. and T.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Grant Agency of the Czech Technical University in Prague, grant No. SGS23/105/OHK2/2T/12.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Li, J.; Du, X.; Martins, J.R.R.A. Machine learning in aerodynamic shape optimization. *Prog. Aerosp. Sci.* **2022**, *134*, 100849. [CrossRef]
2. Luo, J.; Xiong, J.; Liu, F. Aerodynamic design optimization by using a continuous adjoint method. *Sci. China Phys. Mech. Astron.* **2014**, *57*, 1363–1375. [CrossRef]
3. Jameson, A. *Aerodynamic Shape Optimization Using the Adjoint Method*; The von Kármán Institute: Belgium, 2003. Available online: http://aero-comlab.stanford.edu/Papers/jameson.vki03.pdf (accessed on 6 October 2023).
4. Giles, M.B.; Pierce, N.A. An Introduction to the Adjoint Approach to Design. *Flow Turbul. Combust.* **2000**, *65*, 393–415. [CrossRef]
5. Monfaredi, M.; Asouti, V.; Trompoukis, X.; Tsiakas, K.; Giannakoglou, K. Aeroacoustic and Aerodynamic Adjoint-Based Shape Optimization of an Axisymmetric Aero-Engine Intake. *Aerospace* **2023**, *10*, 743. [CrossRef]
6. Kim, H.J.; Sasaki, D.; Obayashi, S.; Nakahashi, K. Aerodynamic Optimization of Supersonic Transport Wing Using Unstructured Adjoint Method. *AIAA J.* **2001**, *39*, 1011–1020. [CrossRef]
7. Li, W.; Tian, Y.; Yi, W.; Ji, L.; Shao, W.; Xiao, Y. Study on adjoint-based optimization method for multi-stage turbomachinery. *J. Therm. Sci.* **2011**, *20*, 398–405. [CrossRef]
8. Jameson, A. Aerodynamic Design via Control Theory. *J. Sci. Comput.* **1988**, *3*, 233–245. [CrossRef]

9. Jameson, A.; Martinelli, L.; Pierce, N.A. Optimum Aerodynamic Design Using the Navier–Stokes Equations. *Theor. Comput. Fluid Dyn.* **1998**, *10*, 213–237. [CrossRef]
10. Salunke, N.P.; Ahamad, R.A.J.; Channiwala, S.A. Airfoil Parameterization Techniques: A Review. *Am. J. Mech. Eng.* **2014**, *2*, 99–102. [CrossRef]
11. Masters, D.A.; Taylor, N.J.; Rendall, T.C.S.; Allen, C.B.; Poole, D.J. Geometric Comparison of Aerofoil Shape Parameterization Methods. *AIAA J.* **2017**, *55*, 1575–1589. [CrossRef]
12. Cantwell, B.J. The NACA Airfoil Series. *STANFORD UNIVERSITY*, Index of /~cantwell/AA200_Course_Material. 2013. Available online: https://web.stanford.edu/~cantwell/AA200_Course_Material/The%20NACA%20airfoil%20series.pdf (accessed on 10 October 2023).
13. Akram, M.T.; Kim, M.H. Aerodynamic Shape Optimization of NREL S809 Airfoil for Wind Turbine Blades Using Reynolds-Averaged Navier Stokes Model—Part II. *Appl. Sci.* **2021**, *11*, 2211. [CrossRef]
14. Ziemkiewicz, D. Simple analytic equation for airfoil shape description. *arXiv* **2016**, arXiv:1701.00817.
15. Sobieczky, H. Parametric Airfoils and Wings. *Notes Numer. Fluid Mech.* **1998**, *68*, 71–88.
16. Khurana, M.; Winarto, H.; Sinha, A. Airfoil Optimisation by Swarm Algorithm with Mutation and Artificial Neural Networks. In Proceedings of the 47th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Orlando, FL, USA, 5–8 January 2009; ISBN 978-1-60086-973-0. [CrossRef]
17. Della Vecchia, P.; Daniele, E.; d'Amato, E. An airfoil shape optimization technique coupling PARSEC parameterization and evolutionary algorithm. *Aerosp. Sci. Technol.* **2014**, *32*, 103–110. [CrossRef]
18. Suprayitno, Y.; Aminnudin, J.C.; Wulandari, R. Airfoil aerodynamics optimization under uncertain operating conditions. *J. Phys. Conf. Ser.* **2020**, *1446*, 012014. [CrossRef]
19. Arias-Montaño, A.; Coello Coello, C.A.; Mezura-Montes, E. Evolutionary Algorithms Applied to Multi-Objective Aerodynamic Shape Optimization. In *Computational Optimization, Methods and Algorithms*; Koziel, S., Yang, X.S., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; Volume 356, pp. 211–240; ISBN 978-3-642-20858-4. [CrossRef]
20. Alonso, J.J. Hess-Smith Panel Method: AA200b—Applied Aerodynamics II—Lecture 3. 2005. Available online: http://aero-comlab.stanford.edu/aa200b/lect_notes/lect3-4.pdf (accessed on 8 September 2023).
21. Xia, L.; Zou, Z.; Wang, Z.; Zou, L.; Gao, H. Surrogate model based uncertainty quantification of CFD simulations of the viscous flow around a ship advancing in shallow water. *Ocean Eng.* **2021**, *234*, 109206. [CrossRef]
22. Cravero, C.; De Domenico, D.; Marsano, D. The Use of Uncertainty Quantification and Numerical Optimization to Support the Design and Operation Management of Air-Staging Gas Recirculation Strategies in Glass Furnaces. *Fluids* **2023**, *8*, 65. [CrossRef]
23. Broniszewski, J.; Piechna, J.R. Fluid-Structure Interaction Analysis of a Competitive Car during Brake-in-Turn Manoeuvre. *Energies* **2022**, *15*, 2917. [CrossRef]
24. Collicott, S.H.; Valentine, D.T.; Houghton, E.L.; Carpenter, P.W. Potential flow. In *Aerodynamics for Engineering Students*, 6th ed.; Butterworth-Heinemann: Waltham, MA, USA, 2013; pp. 149–207; ISBN 9780080966328.
25. Caldwell, A.L. Solutions of the Two-Dimensional, Subsonic Flow About an Airfoil. Master's Thesis, Oregon State College, Corvallis, OR, USA, 1949.
26. Ramsay, R.R.; Hoffmann, M.J.; Gregorek, G.M. *Effects of Grit Roughness and Pitch Oscillations on the S809 Airfoil*; The Ohio State University: Columbus, OH, USA, 1995.
27. Sommers, D.M. *Design and Experimental Results for the S809 Airfoil*; National Renewable Energy Laboratory: Golden, CO, USA, 1997.