Journal of Basic & Applied Sciences, 2025, *21*, 53-65

Published by SET Publisher

Journal of Basic & Applied Sciences

ISSN (online): 1927-5129

Optimal Design of a Biconvex Airfoil for a Supersonic Aircraft Using the Basin-Hopping and Exhaustive Search Methods

Zhenxue Han[1], Owen Luo[2] and Cheng Luo[1,*]

[1] *Department of Mechanical and Aerospace Engineering, University of Texas, Arlington, TX, USA 76019, USA*

[2] *Winston Churchill High School, 11300 Gainsborough Road Potomac, MD 20854, USA*

Article Info:

Keywords:
Supersonic airfoil,
Exhaustive search,
Global optimization,
Lift-to-wave drag ratio,
Basin-hopping Algorithm.

Timeline:
Received: December 28, 2024
Accepted: January 20, 2025
Published: February 14, 2025

Citation: Han Z, Luo O, Luo C. Optimal Design of a Biconvex Airfoil for a Supersonic Aircraft Using the Basin-Hopping and Exhaustive Search Methods. J Basic Appl Sci 2025; 21: 53-65.

Abstract:

In this study, based on target design conditions, an airfoil is designed for a supersonic aircraft to achieve the maximum lift-to-wave drag ratio, with constraints on the lift coefficient, pitching moment, and maximum thickness. The coefficients of lift and wave drag are calculated numerically using shock/expansion wave theory. To solve the corresponding optimization problem, the Basin-Hopping algorithm—a method commonly used in computational chemical physics for determining minimum energy structures of molecules—is employed. To enhance the search for local extrema, the Sequential Least Squares Programming (SLSQP) method, known for handling constrained optimization problems, is integrated with the Basin-Hopping algorithm. For comparison and validation, the exhaustive search method, a simple technique that evaluates various combinations of design variables to find the optimal solution, is also applied. The results show that while the exhaustive search identifies the optimal design, the Basin-Hopping algorithm yields a slightly better design and requires only about 1/60 of the computation time. This work outlines the design process and demonstrates how advanced optimization algorithms can efficiently address engineering design challenges.

DOI: https://doi.org/10.29169/1927-5129.2025.21.07

---

*Corresponding Author
E-mail: chengluo@uta.edu

1. INTRODUCTION

It is well known that airfoils significantly influence the aerodynamic performance of aircraft [1-4]. In addition to aircraft, they also play important roles in aerodynamic bodies, such as helicopters [5,6], micro-air vehicles [7,8], propellers [9,10], and wind turbines [11,12]. This study focuses specifically on the design of an airfoil for a supersonic aircraft, which exceeds the speed of sound and typically features a thin airfoil section [13]. This section is characterized by sharp leading and trailing edges, as well as relatively flat upper and lower surfaces, all designed to minimize wave drag and maximize lift production. In the supersonic aerodynamic regimes, due to the high speed of the flow, the shock wave and wave drag emerge, and the flow is compressible [14]. In contrast, these effects are less significant for subsonic aircraft, which fly at speeds below the speed of sound. As a result, compared to supersonic airfoils, subsonic airfoils typically feature thicker sections with rounded leading edges, optimizing lift generation at lower speeds [15].

To achieve the maximum lift-to-wave drag ratio, the specific profile of a supersonic airfoil, which includes its thickness and camber, must be carefully designed based on target conditions and supersonic aerodynamic principles. The aerodynamic performance of a given airfoil shape can be evaluated using the shock/expansion wave theory [14]. Since different shapes yield varying aerodynamic efficiencies and numerous potential configurations exist, the design of a supersonic airfoil becomes an optimization problem. The lift-to-wave drag ratio may exhibit multiple extrema in different regions of the design space, requiring a search that spans the entire variable space to identify the global extremum. Several techniques are available for finding the global extremum, and two methods are employed here: exhaustive search [16] and Basin-Hopping [17]. The exhaustive search method, also known as brute-force search [16], is straightforward to implement. It involves meshing the design variable space and calculating the lift coefficient, wave drag coefficient and pitching moment at each mesh point. If the constraints are satisfied at a given point, the corresponding design is considered feasible. The optimal design is the feasible solution that yields the maximum lift-to-wave drag, which represents the global maximum or its approximation. Naturally, the likelihood of finding the global maximum increases with a finer mesh, but this also increases the computation time.

Basin-hopping is an efficient technique for locating global extrema. It is an iterative method that was originally developed in 1997 to find the minimum energy structure of molecules [17]. Since then, it has been widely used in computational chemical physics to solve complex optimization problems [18-22]. In [23], Basin-hopping was compared with four established metaheuristics for global optimization: Evolution Strategies (ES) [24], Differential Evolution (DE) [25], Particle Swarm Optimization (PSO) [26], and Simulated Annealing (SA) [27]. The study found that Basin-hopping outperformed these algorithms in solving six real-world problems (three Lennard-Jones and three Morse problems), both in terms of speed and solution quality.

In [11, 12], the non-dominated sorting genetic algorithm II (NSGA-II) [28], a type of evolutionary algorithm, was employed to optimize the designs for wind turbines' airfoils. ES, DE, and PSO are also evolutionary algorithms. As Basin-hopping has demonstrated advantages in speed and efficiency over these methods [23], it has been selected over both these evolutionary algorithms (including NSGA-II) and SA, for finding the optimal design of a supersonic airfoil in this study.

Basin-hopping is a two-phase method that combines a global stepping algorithm with local minimization at each iteration. In this method, the search direction at each iteration is randomly chosen, and a local search is performed using the new iterate as the starting point. The random "hop" in the variable space allows the method to potentially find a global extremum. Basin-hopping has been implemented as a Python function, which can be accessed through an open-source Python library [29] and incorporated into custom code. As a result, it is adopted here to find the global maximum of the lift-to-wave drag ratio.

To effectively search for a local extremum, an optimization method must be integrated with Basin-hopping. Sequential Quadratic Programming (SQP)-based methods are commonly used to solve constrained optimization problems and locate local extrema [30,31]. In such methods, the objective function is first reformulated into a Lagrangian function, incorporating the original objective function, Lagrange multipliers, and constraints. The Newton-Raphson method [32,33], for example, is then applied to identify a point where the gradient of the Lagrangian function vanishes. This point is a critical point of the Lagrangian function, where the function may exhibit an extremum. The standard Newton-Raphson method solves for the critical point by iterating a vector equation involving the inverse of a matrix of function derivatives. However, this matrix is often singular and non-invertible, which can prevent the vector equation from being solved to determine the next search step. To overcome this challenge, the SQP method determines the search step by solving an equivalent optimization subproblem. This subproblem has a quadratic objective function with linear constraints for the search step. The quadratic term in the objective function contains a matrix of second-order partial derivatives of the Lagrangian function. The partial derivatives in this matrix, as well as those in the linear constraints, can be computed using the finite-difference method [31], based on the values of the function and constraints at each iterate.

Sequential Least Squares Programming (SLSQP) [34, 30] and Constrained Optimization By Quadratic Approximations (COBYQA) [35] are both SQP-based approaches. In the SLSQP method, the matrix of second-order partial derivatives is decomposed using LDL^T, L denotes a lower triangular matrix, D is a diagonal matrix, and T represents the transpose of a matrix. The reformulated subproblem is then solved using linear least squares software [36]. In contrast, the COBYQA method uses interpolation-based quadratic polynomials to approximate the original objective function and constraints based on their values. The corresponding subproblem is solved for the next search step using the Byrd-Omojokun approach [37, 38]. This approach assumes that the search step is the sum of the normal and tangential steps, which are determined by solving two related optimization subproblems.

Given our familiarity with the SLSQP method, we adopt it in the Basin-hopping algorithm to search for local extrema. Additionally, the SLSQP method has been implemented as a Python function [34], which can be directly called within the Python code of the Basin-hopping method.

2. GEOMETRIC MODEL

A supersonic airfoil generally has a thin section formed of either angled panels or opposed arcs (called “double wedge airfoils” and “biconvex airfoils” respectively) with sharp leading and trailing edges. A biconvex airfoil is illustrated in Figure **1**. Here, the dimensions of the airfoil are nondimensionalized with respect to the chord length, which is normalized to 1. A rectangular coordinate system is established, with the x-axis aligned along the chord of the airfoil and the y-axis perpendicular to it. The origin is located at the leading edge of the airfoil. In this study, both upper and lower surfaces are mathematically described using third-order polynomial functions of the form:

$$y = ax^3 + bx^2 + cx + d. \tag{1}$$

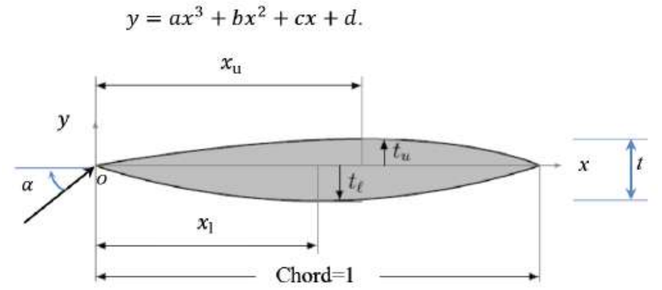

Figure 1: Sketch of a biconvex Airfoil.

Let y_u and y_l denote the y-coordinates of the points on the upper and lower surfaces, respectively. Set t_u and t_l to be the maximum thicknesses of the upper and lower surfaces, respectively. Use x_u and x_l to denote the positions along the chord where t_u and t_l are located. Let α denote the angle of attack, which is the angle between the chord of the airfoil and the incoming airflow. Use t to represent the maximum thickness of the airfoil. Then $t= t_u+t_l$.

The upper and lower surfaces are subject to the following boundary conditions:

$$y(0)= 0, \tag{2}$$

$$y(1)= 0, \tag{3}$$

$$y\,(x_u)= t_u \tag{4}$$

$$\frac{dy}{dx}\Big|_{x=x_u} = 0, \tag{5}$$

$$y(x_l)=-t_l, \tag{6}$$

$$\frac{dy}{dx}\Big|_{x=x_l} = 0. \tag{7}$$

Eqs. (2)-(5) are utilized to find the four coefficients a, b, c, and d in Eq. (1) for the upper surface. Similarly, Equations (2), (3), (6), and (7) are employed to determine the four coefficients for the lower surface. Accordingly, the functions for the upper and lower surfaces are given by

$$y_u = \frac{t_u x[(1-2x_u)x^2+(3x_u{}^2-1)x+x_u(2-3x_u)]}{x_u{}^2(1-x_u)^2}, \tag{8}$$

$$y_l = -\frac{t_l x[(1-2x_l)x^2+(3x_l{}^2-1)x+x_l(2-3x_l)]}{x_l{}^2(1-x_l)^2}. \tag{9}$$

As seen from Eq. (8) and (9), the profile of the airfoil depends on four parameters: t_u, t_l, x_u and x_l.

3. PROBLEM DEFINITION

The design Mach number is set to be 3.0, meaning the aircraft's speed is three times the speed of sound. To ensure efficient flight, the airfoil should have a high lift coefficient (c_l) and a low wave drag coefficient (c_d). Therefore, the objective function is c_l/c_d, and the goal is to maximize c_l/c_d.

A high value of c_l indicates that the aircraft generates more lift for given parameters, such as velocity. In our case, it is required to be at least 0.3. Let c_m denote the pitching moment about the leading edge of the biconvex airfoil. This moment causes the aircraft to rotate nose-up or nose-down. In our case, to ensure stable flight, its absolute value must not exceed 0.1. The most efficient shape for supersonic flight is a flat plate, as it theoretically does not incur any wave drag penalty due to thickness. Due to structural constraints, the value of *t* is set to 0.1. Additionally, to prevent the upper or lower surfaces from intersecting the chord (other than at the leading and trailing edges), x_u and x_l must be positioned between 1/3 and 2/3 of the chord length. In addition, the angle of attack is also constrained to vary between 0° and 10°.

According to the above considerations, there are six constraints for the objective function:

(i) $t = 0.1$;
(ii) $c_l \geq 0.3$;
(iii) $|c_m| \leq 0.1$
(iv) $1/3 \leq x_\mathrm{u} \leq 2/3$;
(v) $1/3 \leq x_\mathrm{l} \leq 2/3$; and
(vi) $0^\circ \leq \alpha \leq 10^\circ$.

These constraints are similar to the ones typically required for a supersonic airfoil [14]. There are four design variables: α, t_u (or t_l), x_u, and x_l. The goal is to find a set of these design variables that maximizes c_l/c_d, while ensuring that all six constraints are satisfied.

4. AERODYNAMIC PERFORMANCE ANALYSIS

To calculate c_l, c_d, and c_m, shock/expansion wave theory will be used to calculate the pressure coefficient [14], assuming that the flow is steady and in viscid, and that it is supersonic throughout with no interference. Let C_{p_u} and C_{p_l} denote the pressure coefficients on the upper and lower surfaces of the airfoil, respectively. Then,

$$c_l = c_n \cos\alpha - c_a \sin\alpha, \tag{10}$$

$$c_d = c_n \sin\alpha + c_a \cos\alpha, \tag{11}$$

$$c_m = \int_0^1 \left(C_{p_\mathrm{u}} - C_{p_\mathrm{l}}\right) x\,dx + \int_0^1 \left(C_{p_\mathrm{u}} \frac{dy_\mathrm{u}}{dx} y_\mathrm{u} - C_{p_\mathrm{l}} \frac{dy_\mathrm{l}}{dx} y_\mathrm{l}\right) dx. \tag{12}$$

In Eqs. (10) and (11), c_n and c_a denote the normal force coefficient and axial force coefficient, respectively. They have the following expressions:

$$c_n = \int_0^1 \left(C_{p_\mathrm{l}} - C_{p_\mathrm{u}}\right) dx, \tag{13}$$

$$c_a = \int_0^1 \left(C_{p_\mathrm{u}} \frac{dy_\mathrm{u}}{dx} - C_{p_\mathrm{l}} \frac{dy_\mathrm{l}}{dx}\right) dx. \tag{14}$$

To calculate the integrals in Eqs. (12)-(14), the upper and lower surfaces of the airfoil are divided into *M* and *N* segments, where *M* and *N* are positive integers and may differ. Each segment is approximated as a straight-line element, with its endpoints described by the coordinates ($x_{\mathrm{u}i}$, $y_{\mathrm{u}i}$) or ($x_{\mathrm{l}j}$, $y_{\mathrm{l}j}$), where i=1,2, ..., M+1 and j = 1,2, ..., N+1. Let $\varphi_{\mathrm{u}i}$ denote the inclination angle that the i^{th} element of the upper surface forms with the chord line. Set $\varphi_{\mathrm{l}j}$ to be the inclination angle that the j^{th} element of the lower surface forms with the chord line. Both $\varphi_{\mathrm{u}i}$ and $\varphi_{\mathrm{l}j}$ range from -90° to 90°, measured from the chord line toward the segment, with the counter-clockwise direction being positive. The *y*-coordinates of these elements can be found using Eqs. (8) and (9) according to their *x*-coordinates.

The pressure coefficient is assumed to be constant along each line segment, while its values may vary between different segments. Based on the division, an integral can be approximated as the sum of the integrals over the corresponding line segments. Consequently, Eqs. (12)-(14) can be re-written as

$$c_m = \sum_{i=1}^{M} \frac{C_{p_{\mathrm{u}i}}[{x_{\mathrm{u}(i+1)}}^2 - {x_{\mathrm{u}i}}^2]}{2} - \sum_{j=1}^{N} \frac{C_{p_{\mathrm{l}j}}[{x_{\mathrm{u}(j+1)}}^2 - {x_{\mathrm{u}j}}^2]}{2} + \sum_{i=1}^{M} \frac{C_{p_{\mathrm{u}i}} \tan\varphi_{\mathrm{u}i} [y_{\mathrm{u}(i+1)} + y_{\mathrm{u}i}][x_{\mathrm{u}(i+1)} - x_{\mathrm{u}i}]}{2} - \sum_{j=1}^{N} \frac{C_{p_{\mathrm{l}j}} \tan\varphi_{\mathrm{l}j} [y_{\mathrm{l}(j+1)} + y_{\mathrm{l}j}][x_{\mathrm{l}(j+1)} - x_{lj}]}{2}, \tag{15}$$

$$c_n = \sum_{j=1}^{N} C_{p_{\mathrm{l}j}} [x_{\mathrm{l}(j+1)} - x_{\mathrm{l}j}] - \sum_{i=1}^{M} C_{p_{\mathrm{u}i}} [x_{\mathrm{u}(i+1)} - x_{\mathrm{u}i}], \tag{16}$$

$$c_a = \sum_{i=1}^{M} C_{p_{\mathrm{u}i}} \tan\varphi_{\mathrm{u}i} [x_{\mathrm{u}(i+1)} - x_{\mathrm{u}i}] - \sum_{j=1}^{N} C_{p_{\mathrm{l}j}} \tan\varphi_{\mathrm{l}j}\, [x_{\mathrm{l}(j+1)} - x_{\mathrm{l}j}]. \tag{17}$$

The deflection angle refers to the change in the flow direction of a fluid, such as air, after it passes through a shock or expansion wave. Let $\theta_{\mathrm{u}i}$ denote the deflection angle for the i^{th} element of the upper surface. Set $\theta_{\mathrm{l}j}$ to be the deflection angle for the j^{th} element of the lower surface. These angles can be determined according to $\varphi_{\mathrm{u}i}$ and $\varphi_{\mathrm{l}j}$ as follows:

$$\theta_{\mathrm{u}i} = \varphi_{\mathrm{u}i} - \varphi_{\mathrm{u}(i-1)}, \quad i \geq 1, \tag{18}$$

$$\theta_{\mathrm{l}j} = -\varphi_{\mathrm{l}j} + \varphi_{\mathrm{l}(j-1)}. \quad j \geq 1, \tag{19}$$

where $\varphi_{\mathrm{u}0} = \alpha$ and $\varphi_{\mathrm{l}0} = \alpha$.

The value of $C_{p_{\mathrm{u}i}}$ or $C_{p_{\mathrm{l}j}}$ on a line segment is determined based on whether expansion or shock waves are formed on that segment. The following two criteria used to determine whether a shock or expansion wave forms on a line segment:

(i) If the deflection angle for a linear element is positive, the supersonic flow is turned into itself, and a shock wave is formed on the line segment. The theory of shock waves is then used to calculate the corresponding $C_{p_{\mathrm{u}i}}$ or $C_{p_{\mathrm{l}j}}$.

(ii) If the deflection angle for a linear element is negative, the supersonic flow is turned away from itself, and an expansion wave is formed on the line segment. In this case, the theory of expansion waves is applied to determine $C_{p_{\mathrm{u}i}}$ or $C_{p_{\mathrm{l}j}}$.

5. THEORIES OF SHOCK AND EXPANSION WAVES FOR CALCULATING C_p

The pressure coefficient (C_{pi}) for the i-th line element on either upper or lower surface is

$$C_{pi} = \frac{2\left(\prod_{m=1}^{i} \frac{p_{2m}}{p_{1m}} - 1\right)}{\gamma {M_\infty}^2}, \tag{20}$$

where p_{1m} and p_{2m} denote the upstream and downstream pressures of a shock wave or expansion waves, respectively, on the m-th element with m ranging from 1 to i, and p_{11} being the free-stream pressure. M_∞ is the free-stream Mach number, which is 3.0 in our case. For air, the specific heat ratio γ is 1.4. Eq. (20) has been used in both theories of expansion and shock waves to calculate the pressure coefficient on a line element, while they differ in the ways of calculating $\frac{p_{2m}}{p_{1m}}$.

5.1. Theory of Expansion Waves

If an expansion wave occurs on the line segment,

$$\frac{p_{2m}}{p_{1m}} = \left[\frac{1+\left(\frac{\gamma-1}{2}\right){M_1}^2}{1+\left(\frac{\gamma-1}{2}\right){M_2}^2}\right]^{\gamma/(\gamma-1)}, \tag{21}$$

where M_1 and M_2 denote the upstream and downstream Mach numbers on the m-th element. For the leading element on either upper or lower surface, M_1 is the free-stream Mach number, and its M_2 must be calculated. For each subsequent element, its M_1 is equal to M_2 of the previous element. For known M_1 and deflection angle, M_2 for the m-th element can be determined using the following two equations:

$$|\theta| = v(M_2) - v(M_1), \tag{22}$$

$$v(M) = \sqrt{\frac{\gamma+1}{\gamma-1}} \tan^{-1} \sqrt{\frac{\gamma-1}{\gamma+1}(M^2-1)} - \tan^{-1}\sqrt{M^2-1}, \tag{23}$$

where θ is the deflection angle of the m-th element. The right-hand side of Eq. (23) is known as the Prandtl-Meyer function, denoted by $v(M)$. The value of $v(M_1)$ can be determined using Eq. (23). This value is then substituted into Eq. (22) to find $v(M_2)$. Based on the value of $v(M_2)$, M_2 can be calculated by numerically solving Eq. (23).

5.2. Theory of Shock Waves

If a shock wave occurs on the m-th element,

$$\frac{p_{2m}}{p_{1m}} = 1 + \frac{2\gamma}{\gamma+1}\left(M_{n,1}^2 - 1\right), \tag{24}$$

where $M_{n,1}$ is the normal Mach number before the shock wave, and it is related to M_1 and oblique shock angle (β) by

$$M_{n,1} = M_1 \sin\beta. \tag{25}$$

The value of β can be determined by solving the following θ-β-M relation:

$$\tan\theta = 2\cot\beta \frac{{M_1}^2 \sin^2\beta - 1}{{M_1}^2(\gamma + \cos 2\beta) + 2}. \tag{26}$$

Subsequently, the normal Mach number ($M_{n,2}$) behind the shock wave can be found using the following relationship:

$$M_{n,2}^2 = \frac{1+\left(\frac{\gamma-1}{2}\right)M_{n,1}^2}{\gamma M_{n,1}^2 - \frac{\gamma-1}{2}}. \tag{27}$$

M_2 is related to $M_{n,2}$ by

$$M_2 = \frac{M_{n,2}}{\sin(\beta-\theta)}. \tag{28}$$

For a line segment on either upper or lower surface, for given M_1 and θ, its $M_{n,1}$ and M_2 can be determined using Eqs. (25) and (28), respectively. Subsequently, the corresponding value of $\frac{p_{2m}}{p_{1m}}$ can be found using Eq. (24).

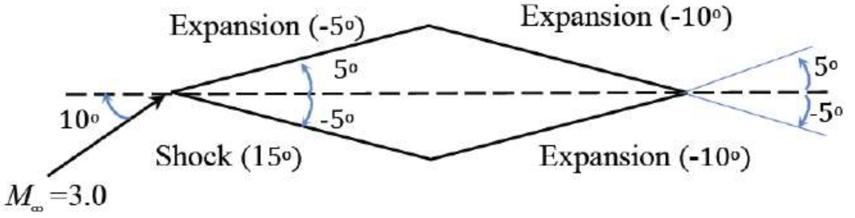

Figure 2: A simple example to illustrate the process to determine c_m, c_l, and c_d. The values in the paratheses are the deflection angles.

6. A SIMPLE EXAMPLE

For illustration, consider a simple diamond-wedge airfoil shown in Figure **2**, with α=10° and M_∞ =3.0. The airfoil consists of two line elements on both the upper and lower surfaces. As marked in Figure **2**, the two top elements have inclination angles of 5° and -5°, respectively, while the two bottom elements have inclination angles of -5° and 5°. Using Eqs. (18) and (19), the corresponding angles of deflections are calculated as -5°, -10°, 15°, and -10°, respectively. According to the two criteria outlined in Sec. 4, expansion waves are applied to both elements of the upper surface and the second element of the lower surface, while shock waves are applied to the first element of the lower surface. Subsequently, the theory of expansion waves is employed to determine the pressure ratios for the three elements under expansion. These pressure ratios are then substituted into Eq. (20) to calculate the pressure coefficient. Meanwhile, the theory of shock is applied to determine the pressure ratio for the first element of the lower surface, which is then substituted into Eq. (20) to find the corresponding pressure coefficient.

For the top two elements, the values of $\frac{p_2}{p_1}$ are 0.671 and 0.269, and $C_{p_{\mathrm{u}1}}$= −0.05222 and $C_{p_{\mathrm{u}2}}$= −0.11603. For the bottom two elements, the values of $\frac{p_2}{p_1}$ are 2.82 and 1.49, $C_{p_{\mathrm{l}1}}$= 0.28889, and $C_{p_{\mathrm{l}2}}$= 0.0746. Subsequently, by Eqs. (15)-(17),

$$c_m = \frac{(-0.05222)(0.5^2-0^2)}{2}+\frac{(-0.11603)(1^2-0.5^2)}{2}-\frac{0.28889\,(0.5^2-0^2)}{2}-\frac{0.0746(1^2-0.5^2)}{2}+\frac{(-0.05222)\tan 5^\mathrm{o}(0.5/\tan 5^\mathrm{o}+0)(0.5-0)}{2}+\frac{(-0.11603)\tan(-5^\mathrm{o})(0+0.5/\tan 5^\mathrm{o})(1.0-0.5)}{2}-\frac{0.28889\tan(-5^\mathrm{o})[0+0.5/\tan(-5^\mathrm{o})](0.5-0)}{2}-\frac{0.0746\tan 5^\mathrm{o}[0.5/\tan(-5^\mathrm{o})+0](1.0-0.5)}{2}=-0.1143, \tag{29}$$

$$c_n = 0.28889(0.5-0)+0.0746(1-0.5)-(-0.05222)(0.5-0)-(-0.11603)(1-0.5)=0.26578, \tag{30}$$

$$c_a = (-0.05222)\tan 5^\mathrm{o}(0.5^2-0^2)+(-0.11603)\tan(-5^\mathrm{o})(1^2-0.5^2)-0.28889\tan(-5^\mathrm{o})(0.5^2-0^2)-0.0746\tan 5^\mathrm{o}(1^2-0.5^2)=0.0122. \tag{31}$$

Accordingly, by Eqs. (10) and (11),

$$c_l = 0.26587\cos 10^\mathrm{o} - 0.0122\sin 10^\mathrm{o} = 0.2597, \tag{32}$$

$$c_d = 0.26587\sin 10^\mathrm{o} + 0.0122\cos 10^\mathrm{o} = 0.0582. \tag{33}$$

In terms of Eqs. (32) and (33),

$$\frac{c_l}{c_d} = 4.464. \tag{34}$$

Meanwhile, by Eq. (29), $|c_m| = 0.1143 > 0.1$. Therefore, it violates the design requirement regarding the pitch moment. As such, this design is not a feasible one.

7. OPTIMIZATION OF DESIGN

Three algorithms will be used to solve the optimal design problem. The first algorithm calculates c_l, c_d, and c_m for a given set of design parameters. The second and third algorithms are two optimization methods: exhaustive search and Basin-hopping. The first algorithm will be combined with each of the optimization methods to find the optimal set of design parameters that yields the global maximum value of $\frac{c_l}{c_d}$. The results from these two combinations will then be compared. The three algorithms are described below, one by one. Three Python codes are developed to implement these algorithms: the codes for the first two algorithms are developed by us, while the third one directly calls the Basin-Hopping function from an open-source Python library [29].

7.1. Algorithm to Find $\frac{c_l}{c_d}$

For a given set of design parameters, this algorithm is given below:

(i). Break the upper and lower surfaces of the airfoil into *M* and *N* line elements, respectively.

(ii). Compute the inclination angles of the line elements using Eqs. (8) and (9).

(iii). Find the deflection angles of the line elements using Eqs. (18) and (19).

(iv). Determine whether a shock or expansion wave applies on a line element according to the four criteria listed in Sec. 4.

(v). For elements under compression or expansion, calculate the pressure ratios using the corresponding theory, and then determine the pressure coefficients using Eq. (20).

(vi). Calculate c_n and c_d by Eqs. (16) and (17), respectively.

(vii). Finally, calculate c_l and c_d using Eqs. (10) and (11) and determine c_m using Eq. (15).

The example in Sec. 6 follows the above algorithm to calculate $\frac{c_l}{c_d}$ and c_m manually. A more complicated calculation will be performed here by Python programming using this algorithm for different sets of design parameters.

7.2. Exhaustive Search Algorithm

This algorithm is straightforward and simple to implement [16]:

(i) Divide each of design variables (α, t_u or t_l, x_u, and x_l) within their specified bounds into multiple discrete elements.

(ii) Use a nested loop to iterate through all possible combinations of these elements.

(iii) For each set of design variables, calculate the aerodynamic coefficients (c_l, c_d, and c_m) using the first algorithm, and then examine if all design constraints are met. A design that satisfies all constraints is considered a feasible design. Multiple feasible designs may be obtained after the nested loop is completed.

(iv) Finally, compare the feasible designs and select the one that produces the maximum value of $\frac{c_l}{c_d}$, which is the optimal design.

7.3. Basin-hopping Algorithm

Determining the global maximum of $\frac{c_l}{c_d}$ is equivalent to finding the global minimum of $-\frac{c_l}{c_d}$. The Basin-hopping algorithm is employed to identify the set of design parameters that leads to this global minimum while satisfying six design constraints. The value of $\frac{c_l}{c_d}$ for each tested set of design variables is also computed using the first algorithm. The Basin-hopping algorithm is illustrated below by finding the global minimum of a single-variable function $y(x)$. The original algorithm was proposed in [17], with the aim of finding the best conformation of a protein that corresponds to its lowest energy state.

(I) Take an initial guess x_{10}, which gives the initial value of the objective function (y_{10}), and conduct a local search for x_{11}, where the objective function has a local minimum value of y_{11}. Save y_{11} as the global minimum value.

(II) Take a random large step from x_{11} to x_{20}, and perform a local search to find x_{21}, where the objective function has a local minimum value of y_{21}.

(III) Perform an acceptance test, and consider two cases.

Case I: if the test passes, repeat the second and third steps. However, in the second step (Figures **3a** or **3b**), the random large step is taken from x_{21} (not x_{11}) to a new x value (say, x_{30}), which is followed by a local search for a new local minimum value of the objective function.

Case II: if the test fails, also repeat the second and third steps. In the second step (Figure **3c**), the random large step is still taken from x_{11}. The new x value is also labelled as x_{30} in Figure **3c**, followed by a local search for a new local minimum value of the objective function.

The acceptance test passes in two scenarios: (i) $y_{21} \leq y_{11}$, which is the case illustrated in Figure **3a**, and (ii) $y_{21} > y_{11}$ while it satisfies the Metropolis criterion [40,41] (Figure **3b**). In the first scenario, y_{21} will be saved as the new global minimum, while it is not in the second scenario. The Metropolis criterion used in the second scenario is developed according to standard Monte Carlo algorithms. It uses a prabability to decide whether x_{21} will be accepted as a new start point from the global search based on the value of y_{21}. If this criterion is sastified, the acceptance test passes, and x_{21} is accepted as the new start point (Figure **3b**). In case it is not, the acceptance test fails, and x_{11} is still the start point for the global search (Figure **3c**). SQRL method is adopted here for local search.

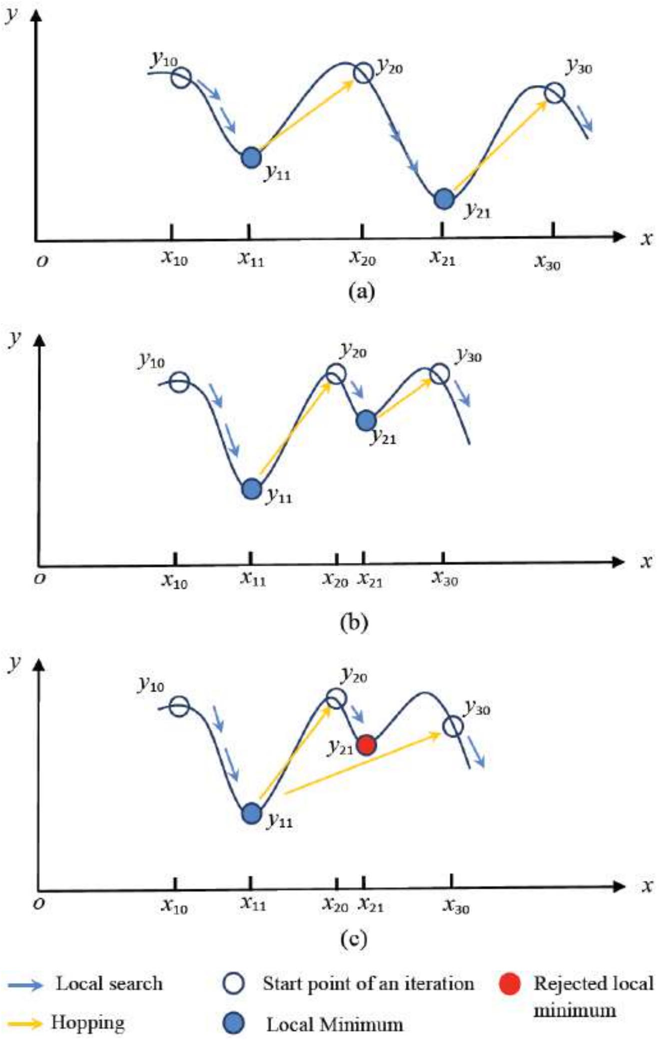

Figure 3: Illustration of the basin-hopping algorithm via the search for the global minimum of a function *y*(x).

8. NUMERICAL RESULTS AND DISCUSSIONS

8.1. Numerical Results for the Simple Example

Figure **4** gives the C_p distribution and airfoil shape calculated for the simple example, using the first algorithm. The values of C_p on the two top elements are -0.05280 and -0.11615, while for the two bottom elements, they are 0.28900 and 0.073206. These values closely match those obtained manually (see Sec. 6). The values of C_l and C_d are found to be 0.2594 and 0.0581, respectively. They are also in close agreement with the manually calculated values. The value of C_l /C_d is 4.463. It is nearly identical to the manually calculated value of 4.464. The simulated airfoil also has a diamond-shaped profile (Figure **4**).

8.2. Optimal Designs

When both methods were used to determine the optimal designs, either the upper or lower surface was divided into 20 line elements. In the exhaustive search method, the design variables α and t_u had 20 divisions within their respective bounds, while x_u and x_l were divided into 30 intervals. In contrast, the Basin-hopping did not require meshing of the design space. As such, no divisions were made. Meanwhile, for the Basin-hopping, the start point was chosen as the set of design variables (5^{o}, 0.05, 0.5, 0.5), with each variable being approximately the mean value of its respective interval. In the case of the exhaustive method, the start point was set by default to be the lowest value in the design space, which in our case was (0^{o}, 0.0, 1/3, 1/3), with each variable taking the minimum value of its interval. The Python codes of both methods were executed on the Google Colab platform [42], using a Microsoft surface laptop 6 equipped with an Inte® Ultra 5 Processor and 16GB of random-access memory.

The exhaustive search method first identified all feasible designs and then compared them to find the optimal design. The computation took 56 minutes and 31 seconds. In total, 3697 feasible designs were found. They were ranked according to their $\frac{c_l}{c_d}$ values, from lowest to highest. 10 of them were chosen and listed in

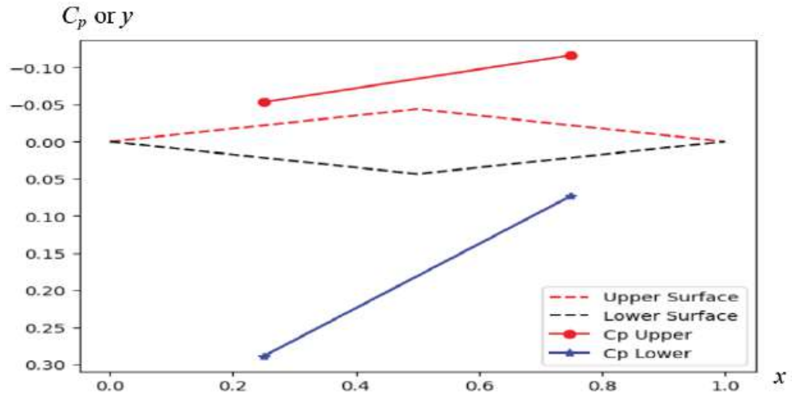

Figure 4: Numerically calculated C_p distribution and airfoil shape for the simple example.

Table 1: Feasible Designs and Optimal Designs

Designs Nos. 1-10, and 12 are the results of the exhaustive search, while Nos. 11 and 13 are those of the Basin-hopping. The bold-faced values represent the maximum $\frac{c_l}{c_d}$ found by each approach, with the full set of constraints or with fewer constraints.

No.	α (°)	t_u	x_u	x_l	c_l	c_d	c_m	$\frac{c_l}{c_d}$
1	8.42	0	0.333	0.425	0.325	0.123	-0.088	2.649
2	8.42	0	0.345	0.425	0.325	0.123	-0.088	2.649
3	9.47	0.005	0.333	0.460	0.335	0.120	-0.099	2.804
4	9.47	0.005	0.345	0.460	0.335	0.120	-0.099	2.805
5	8.42	0.011	0.586	0.414	0.305	0.105	-0.089	2.913
6	8.42	0.011	0.598	0.414	0.305	0.105	-0.089	2.913
7	8.95	0.016	0.667	0.414	0.308	0.102	-0.094	3.017
8	9.47	0.016	0.437	0.437	0.317	0.105	-0.100	3.018
9	9.47	0.021	0.586	0.460	0.302	0.094	-0.098	3.201
10	9.47	0.021	0.598	0.460	0.302	0.094	-0.098	**3.201**
11	9.82	0.024	0.667	0.500	0.300	0.091	-0.100	**3.312**
12 (Fewer constraints via exhaustive search)	6.32	0.063	0.632	0.621	0.149	0.032	-0.07	**4.711**
13 (Fewer constraints via Basin-hopping)	6.63	0.064	0.636	0.617	0.157	0.033	-0.073	**4.717**

Table **1** as Nos. 1 to 10. They included the top two on the ranking list, two designs near the 25% position, two near the 50% position, two near the 75% position, and the bottom two. As observed from Table **1**, for the feasible designs, the values of α range from to 8.42° to 9.47°, t_u from to 0 to 0.021, x_u from to 0.333 to 0.598, x_l from to 0.425 to 0.460, and $\frac{c_l}{c_d}$ from to 2.649 to 3.201. According to the result of exhaustive search, Design No. 10 has the highest value of $\frac{c_l}{c_d}$, which is 3.201, with design variables α=9.47°, t_u=0.021, x_u=0.598, and x_l =0.460.

In contrast, the Basin-hopping method took only 58 seconds, about 60 times faster than the exhaustive search method. The optimal design found by Basin-hopping was listed as No. 11 in Table **1**, with a $\frac{c_l}{c_d}$ value of 3.312, a 3.4% improvement over the exhaustive search result. The optimal set of design variables were α=9.82°, t_u=0.024, x_u=0.667, and x_l =0.500.

For comparison, designs without Constraints 2 and 3 were also considered. These constraints are critical for maintaining flight efficiency and stability. Therefore, removing them rendered the designs unfeasible. This case was explored solely for comparison purposes. The optimal designs found in this scenario were listed as Nos. 12 and 13 in Table **1**. Without these two constraints, the Basin-hopping method completed the computation in just 7 seconds, while the exhaustive search took 49 minutes and 28 seconds. Both methods found nearly identical maximum values for $\frac{c_l}{c_d}$. They are 4.711 for exhaustive search and 4.717 for Basin-hopping. The corresponding design variables were close. They were (6.32°, 0.063, 0.632, 0.621) and (6.63°, 0.064, 0.636, 0.617), respectively. However, neither design satisfied Constraint 2. It is noteworthy that the maximum values of $\frac{c_l}{c_d}$ obtained in this case were higher than those achieved when all constraints were applied. For example, in the case with all constraints, the maximum value of $\frac{c_l}{c_d}$ was 3.312, while in the case without Constraints 2 and 3, it increased to 4.717. Additionally, the design variables changed significantly. For instance, the value of α decreased from 9.82° to 6.63°, demonstrating the impact of removing these constraints on the optimal design.

8.3. Profiles of Airfoils

Figure **5** gives the profiles of airfoils derived from different possible designs. Figure **5a** compares four feasible designs: Designs 1, 5, 10, and 11 listed in Table **1**. The four designs all have the same total thickness of 0.1, which is one of the constraints that should be met. However, the thicknesses of their upper surfaces decrease as $\frac{c_l}{c_d}$ increases, progressing from Design 1 to Design 5, then Design 10, and finally

Design 11. Furthermore, the location of the maximum thickness on the upper surface shifts towards the trailing edge in the same order: Designs 1, 5, 10, and 11. This suggests that a thicker upper surface with the largest thickness positioned further downstream may enhance the aircraft's flight efficiency. Conversely, the thicknesses of their lower surfaces follow an opposite trend. Figure **5b** compares the optimal designs that satisfy all or fewer constraints. Designs 10 and 11 are the optimal solutions that satisfy all constraints, identified using exhaustive search and Basin-hopping, respectively. They result in similar airfoil profiles. Designs 12 and 13, which do not satisfy Constraint 2, are found through the same methods. These two airfoils have almost identical shapes. However, compared to Designs 10 and 11, Designs 12 and 13 feature more convex upper surfaces and less convex lower surfaces. These geometric differences result in a higher lift-to-wave drag ratio for Designs 12 and 13. Nevertheless, their lift coefficients (c_l) do not meet the required minimum of 0.3, as specified by Constraint 2.

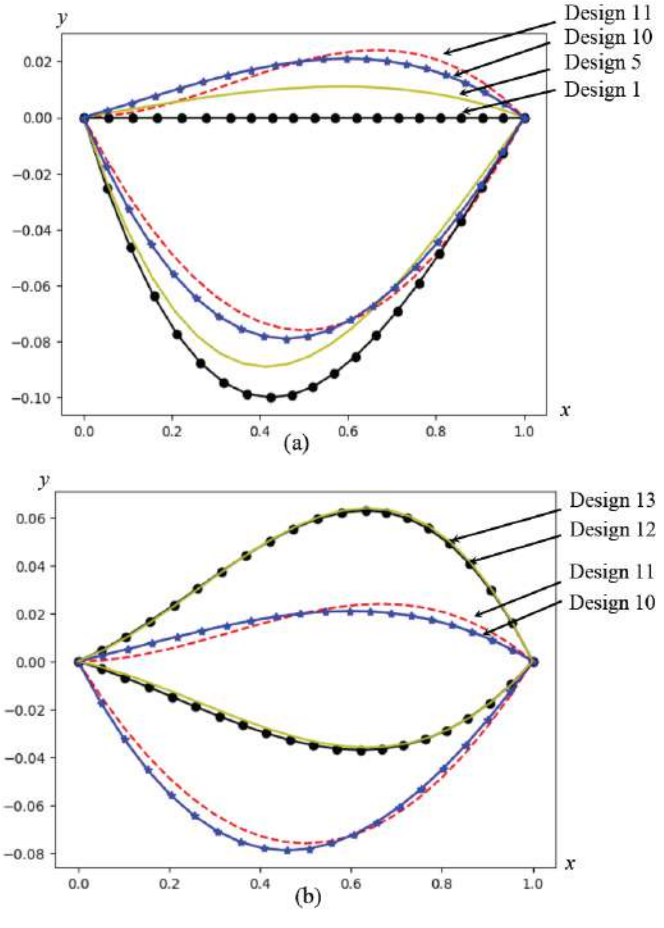

Figure 5: Profiles of airfoils according to (**a**) Designs 1, 5, 10 and 11, and (**b**) Designs 10-13 in Table 1 (not to scale). Designs 10 and 11 are the optimal designs satisfying all constraints and Designs 12 and 13 are the ones satisfying fewer constraints.

8.4. Aerodynamic Performance of the Optimal Design

Next, we specifically consider the aerodynamic performance of the optimal design that has been found, which is Design 11. Figure **6** gives the airfoil profile which is plotted to the scale in the sense that both *x*- and *y*-axes use the same scales, while the ones in Figure **5** are not to the scale. It is clear from Figure **6** that the optimal design leads to a thin biconvex profile, with the upper surface relatively flat and lower surface more convex. This figure also gives the distribution of the pressure coefficients along both the upper and lower surfaces. The pressure coefficient is approximately -0.1 along the upper surface, indicating that the pressure along the upper surface is 10% lower than the pressure of the oncoming air stream.

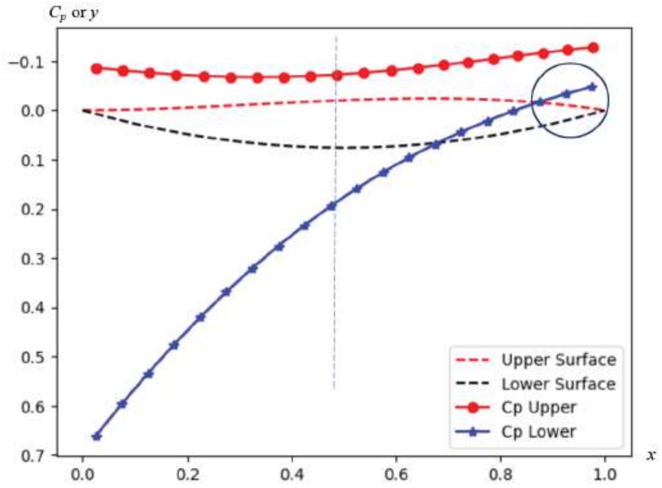

Figure 6: C_p distribution and scaled airfoil profile along the chord in the optimal design (Design 11).

Along the lower surface, the pressure coefficient decreases from 0.68 on the element near the leading edge to -0.03 on the element near the trailing edge. It is positive on the first 17 elements. Accordingly, the pressures on these elements are higher than that of the free air stream. The pressure coefficient is negative on the last three elements, which are circled in the figure, indicating that the pressures on those elements are lower than that of the oncoming air stream on each element. As the pressure coefficient on each element is larger than that of its counterpart of the upper surface, the pressure on the entire lower surface is larger than that on the entire upper surface. As such, the airfoil is lifted. The pressure coefficients are larger than 0.2 on the elements located on the left half of the lower surfaces. They are also larger than their counterparts on the right half of the lower surfaces. Therefore, the lift is mainly induced by the pressures applied to the left half of the lower surface. This seems reasonable, as this portion of the lower surface directly faces the incoming air flow and is thus pushed to move upwards.

A polar diagram illustrates the relationship between two aerodynamic parameters as the angle of attack varies, providing a visual representation of the airfoil's aerodynamic performance. In our case, the angle of attack is varied from 0° to 10°, and the airfoil is defined by the three geometric values from the optimal design: t_{u}=0.024, x_{u}=0.667, and x_{l}=0.500.

Figure **7** gives the polar diagram of the wave drag coefficient versus lift coefficient. As the lift coefficient increases, the drag coefficient also increases, with both drag and lift forces following a similar trend in their relationship.

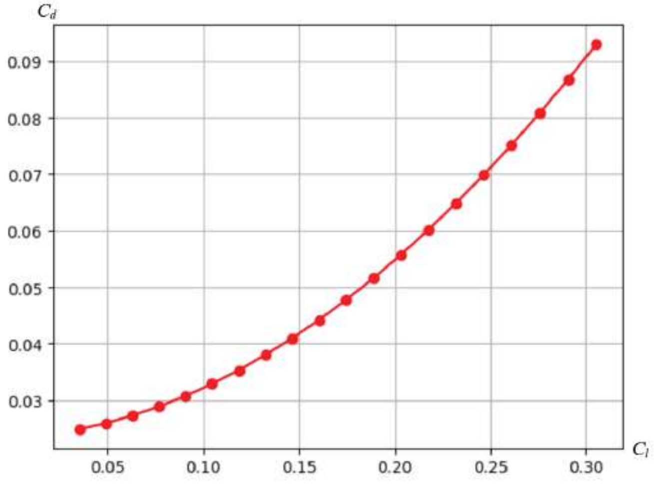

Figure 7: C_d versus C_l polar diagram for the optimally designed airfoil.

Nevertheless, the relationship between lift-to-wave drag ratio and the lift coefficient does not follow the same trend. As shown in Figure **8**, the lift-to-wave drag ratio initially increases as the lift coefficient rises from 0.03 to 0.177, and then decreases as the lift coefficient continues to increase. In our case, c_l is constrained to be no less than 0.30. As observed from Figure **8**, when c_l ≥ 0.3, the lift-to-wave drag ratio decreases with increasing c_l. Therefore, the maximum lift-to-wave drag ratio occurs at c_l =0.3, where it reaches 3.312. The corresponding angle of attack is 9.82°. This observation explains how the three geometric values in Design 11 resulted in c_l =0.3, $\frac{c_l}{c_d}$=3.312, and α=9.82°.

Figure **9** shows the polar diagram of pitching moment versus the lift coefficient. Since the pitching moment is negative, it rotates the airfoil counterclockwise about the leading edge, causing the aircraft's nose to pitch down. The pitching moment decreases approximately linearly as the lift coefficient increases. On the other hand, its magnitude increases with the increase in the lift coefficient, indicating that a large lift force causes a pitching moment with a large magnitude. When c_l reaches 3.0, the magnitude of the pitching moment reaches the maximum allowed value of 0.10. Accordingly, to satisfy the constraint that $|c_m|$ ≤ 0.1, c_l cannot exceed 0.3. Together with the constraint that c_l ≥ 0.3, we must have $c_l = 0.3$. The corresponding angle of attack is also 9.82°. This observation further explains how the three geometric values in Design 11 yielded that c_l =0.3 and α=9.82°, as well as that c_m=-0.100.

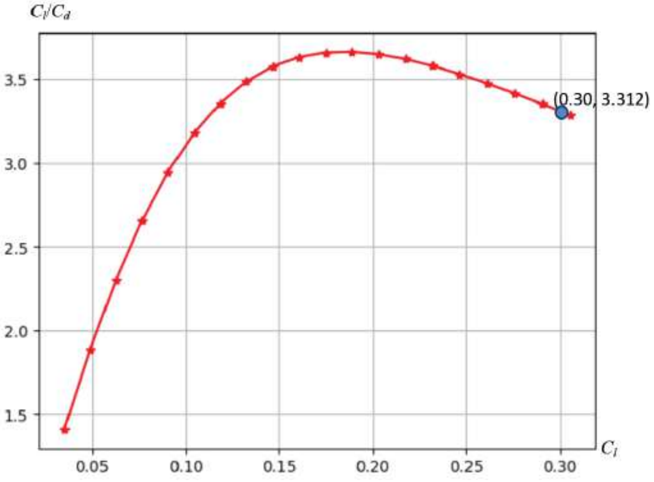

Figure 8: C_l/C_d versus C_l polar diagram for the optimally designed airfoil.

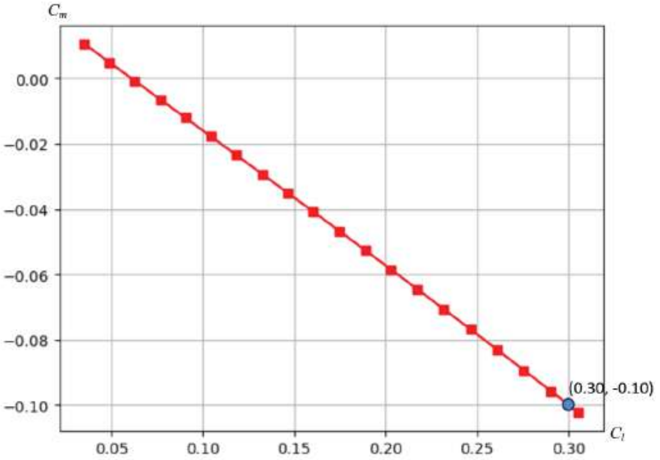

Figure 9: C_m versus C_l polar diagram for the optimally designed airfoil.

9. SUMMARY AND CONCLUSIONS

This work focused on determining the optimal design of a biconvex airfoil by solving an optimization problem that included six constraints, using three different algorithms. The first algorithm was used to numerically determine the aerodynamic performance resulting from a specific design. The other two algorithms, exhaustive search and Basin-hopping, were applied to find the global extrema. The exhaustive search method provided a list of feasible designs based on the results, taking nearly 1 hour to complete the search.

In comparison, the Basin-hopping algorithm conducted a local search within the region where a feasible design was found. This approach has the potential to identify a better design within this region. Additionally, Basin-hopping randomly "hops" to new start points for further local searches, which increases its chances of finding a global maximum in fewer search steps and with less computational time. In our case, the Basin-hopping algorithm completed the search in about 1 minute, approximately 60 times faster than the exhaustive search. The best design found by Basin-hopping was slightly superior to the one obtained by the exhaustive search.

However, a limitation of Basin-hopping is the risk of missing the global extremum if the algorithm does not "hop" into the region where it exists. To mitigate this risk, multiple start points could be used to initiate several different search paths, thereby increasing the likelihood of finding the global extremum.

An optimal design, identified as Design 11 in Table **1**, was found using Basin-hopping. This design features a thin airfoil with an approximately flat upper surface that is less curved than the lower surface. The lift generated by this airfoil primarily comes from the pressure applied to the forward half of the lower surface. The aerodynamic performance of this airfoil was further analyzed through polar diagrams showing the drag coefficient, lift-to-wave drag ratio, and pitching moment versus the lift coefficient. These results provided valuable insights into how the design met the requirements and achieved excellent aerodynamic performance.

This study presents an efficient approach to finding optimal designs, combining an algorithm for computing aerodynamic performance numerically with another for finding the global extremum. This methodology may also be applied to efficiently find optimal designs of similar airfoils under different constraints.

REFERENCES

[1] Gopalarathnam A, McAvoy C. Effect of airfoil characteristics and trim considerations on aircraft performance, 19th AIAA Applied Aerodynamics Conference (Anaheim, CA, U.S.A) 2001. https://doi.org/10.2514/6.2001-2409

[2] Gopalarathnam A, McAvoy CW. Effect of airfoil characteristics on aircraft performance. Journal of Aircraft 2002; 39(3): 427-433. https://doi.org/10.2514/2.2968

[3] Letko W, Brewer JD. Effect of Airfoil Profile of Symmetrical Sections on the Low-Speed Rolling Derivatives of 45° Sweptback-Wing Models of Aspect Ratio 2.61, National Advisory Committee for Aeronautics, Washington, D.C., 1949.

[4] Wood RM, Miller DS. Impact of airfoil profile on the supersonic aerodynamics of delta wings. Journal of Aircraft 1986; 23(9): 695-702. https://doi.org/10.2514/3.45364

[5] Aoki M, Nishimura H, Yamakawa E. Effect of Airfoil on rotational noise of helicopter rotor, Aircraft Symposium, 33 rd, Hiroshima, Japan 1995.

[6] McVeigh MA, McHugh FJ. Influence of tip shape, chord, blade number, and airfoil on advanced rotor performance. Journal of The American Helicopter Society 1984; 29(4): 55-62. https://doi.org/10.4050/JAHS.29.55

[7] Teja NNSK, Teja A, Harsha JS, Ganesh S, Ramana KV. Optimization of micro air vehicle airfoil. International Journal of Research in Engineering and Technology 2016; 5(3): 217-219. https://doi.org/10.15623/ijret.2016.0503043

[8] Zhao L, Yang S. Influence of thickness variation on the flapping performance of symmetric NACA airfoils in plunging motion. Mathematical Problems in Engineering 2010; p. 19. https://doi.org/10.1155/2010/675462

[9] Ismail KA, Rosolen CV. Effects of the airfoil section, the chord and pitch distributions on the aerodynamic performance of the propeller. Journal of the Brazilian Society of Mechanical Sciences and Engineering 2019; 41(3): 1-19. https://doi.org/10.1007/s40430-019-1618-x

[10] Zhang H, Hu Y, Wang G. The effect of aerofoil camber on cycloidal propellers. Aircraft Engineering and Aerospace Technology 2018; 90(8): 1156-1167. https://doi.org/10.1108/AEAT-08-2016-0128

[11] Zhang S, Li H, Jia W, Xi D. Multi-objective optimization design for airfoils with high lift-to-drag ratio based on geometric feature control. IOP Conf. Series: Earth and Environmental Science 2019; 227: 032014. https://doi.org/10.1088/1755-1315/227/3/032014

[12] Zhang R, Li D, Chang H, Wei X, Wang H, Multi-objective optimization on blade airfoil of vertical axis wind turbine Physics of Fluids 2024; 36: 085172. https://doi.org/10.1063/5.0220194

[13] Courant R, Friedrichs KO. Supersonic Flow and Shock Waves. Edwards Brothers, Inc., Ann Arbor, MI 1991.

[14] Anderson JD. Jr. Fundamentals of Aerodynamics (Sixth ed.). New York, NY 2016.

[15] Zucker R, Biblarz O. Fundamentals of Gas Dynamics, Second Edition. John Wiley & Sons, Inc. 2002.

[16] Bernstein DJ. Understanding brute force. Workshop Record of ECRYPT STVL Workshop in Symmetric Key Encryption, eSTREAM report 2005/036, 2005. https://cr.yp.to/snuffle/bruteforce-20050425.pdf

[17] Wales, DJ, Doye JPK. Global Optimization by Basin-Hopping and the Lowest Energy Structures of Lennard-Jones Clusters Containing up to 110 Atoms. The Journal of Physical Chemistry A 1997; 101(28): 5111-5116. https://doi.org/10.1021/jp970984n

[18] Wales DJ, Scheraga HA. Global optimization of clusters, crystals, and biomolecules. Science 1999; 285(5432): 1368-1372. https://doi.org/10.1126/science.285.5432.1368

[19] Doye JPK, Leary RH, Locatelli M, Schoen F. Global optimization of Morse clusters by potential energy transformations. INFORMS Journal on Computing 2004; 16(4): 371-379. https://doi.org/10.1287/ijoc.1040.0084

[20] Kucharik M, Hofacker IL, Stadler PF, Qin J. Basin hopping graph: a computational framework to characterize RNA folding landscapes. Bioinformatics 2014; 30(14): 2009-2017. https://doi.org/10.1093/bioinformatics/btu156

[21] Zhou C, Ieritano C, Hopkins WS. Augmenting basin-hopping with techniques from unsupervised machine learning: Applications in spectroscopy and ion mobility. Frontiers in Chemistry 2019; 7: 519. https://doi.org/10.3389/fchem.2019.00519

[22] Banerjee A, Jasrasaria D, Niblett SP, Wales DJ, Crystal structure prediction for benzene using basin-hopping global optimization. The Journal of Physical Chemistry A 2021; 125(17): 3776-3784. https://doi.org/10.1021/acs.jpca.1c00903

[23] Baioletti M, Santucci V, Tomassini M. A performance analysis of Basin hopping compared to established metaheuristics for global optimization. J Glob Optim 2024; 89(3): 803-832. https://doi.org/10.1007/s10898-024-01373-5

[24] Bäck T, Foussette C, Krause P. Contemporary Evolution Strategies. Springer, Berlin, Heidelberg 2013. https://doi.org/10.1007/978-3-642-40137-4

[25] Storn R, Price K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization 1997; 11(4): 341-359. https://doi.org/10.1023/A:1008202821328

[26] Kennedy J, Eberhart R. Particle swarm optimization. In Proceedings of ICNN'95-International Conference on Neural Networks. IEEE 1995; 4: 1942-1948. https://doi.org/10.1109/ICNN.1995.488968

[27] Kirkpatrick S, Daniel Gelatt C, Vecchi MP. Optimization by simulated annealing. Science 1983; 220(4598): 671-680. https://doi.org/10.1126/science.220.4598.671

[28] Deb K, Agrawal S, Pratap A, Meyarivan T. A Fast Elitist Multi-objective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 2002; 6(2): 182-197. https://doi.org/10.1109/4235.996017

[29] Website of the Basin-hopping in Scipy, an open-source Python library used for scientific and technical computing: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.basinhopping.html

[30] Kraft D. A software package for sequential quadratic programming. Tech. Rep. DFVLR-FB 88-28, DLR German Aerospace Center -- Institute for Flight Mechanics, Koln, Germany 1988.

[31] Nocedal J, Wright SJ. Numerical Optimization. Springer New York, 2006.

[32] Ypma TJ. Historical development of the Newton-Raphson method. SIAM Review 1995; 37(4): 531-551. https://doi.org/10.1137/1037125

[33] Dedieu JP. Newton-Raphson Method. In: Engquist, B. (eds) Encyclopedia of Applied and Computational Mathematics. Springer, Berlin, Heidelberg 2015. https://doi.org/10.1007/978-3-540-70529-1_374

[34] Wilson RB. A simplicial algorithm for concave programming. Doctoral dissertation, Harvard Business School 1963.

[35] Ragonneau TM. Model-Based Derivative-Free Optimization Methods and Software. PhD thesis, Department of Applied Mathematics, The Hong Kong Polytechnic University, Hong Kong, China 2022. URL: https://theses.lib.polyu.edu.hk/handle/200/12294

[36] Lawson CL, Hanson RJ, Solving Least Squares Problems. Prentice-Hall, Englewood Cliffs 1974.

[37] Byrd RH. Robust trust region methods for constrained optimization. In: The Third SIAM Conference on Optimization 1987.

[38] Omojokun EO. Trust Region Algorithms for Optimization with Nonlinear Equality and Inequality Constraints. PhD thesis. Boulder, CO, US: University of Colorado Boulder 1989.

[39] Website of the SLSQP in Scipy, an open-source Python library used for scientific and technical computing: https://docs.scipy.org/doc/scipy/reference/optimize.minimize-slsqp.html

[40] Kalos MH, Whitlock PA. Monte Carlo Methods Volume I: Basics. New York: Wiley 1986; pp. 78-88. https://doi.org/10.1002/9783527617395

[41] Tierney L. Markov chains for exploring posterior distributions, The Annals of Statistics 1994; 22(4): 1701-1762. https://doi.org/10.1214/aos/1176325750

[42] Google Colab Platform for running Python codes: https://colab.research.google.com/