



Evolutionary generative design of supercritical airfoils: an automated approach driven by small data

Kebin Sun¹ · Weituo Wang¹ · Ran Cheng¹ · Yu Liang² · Hairun Xie³ · Jing Wang³ · Miao Zhang³

Received: 26 May 2023 / Accepted: 4 August 2023 / Published online: 23 August 2023
© The Author(s) 2023

Abstract

Supercritical airfoils are critical components in the design of commercial wide-body aircraft wings due to their ability to enhance aerodynamic performance in transonic flow regimes. However, traditional design methods for supercritical airfoils can be time-consuming and require significant manual effort, not to mention the high cost associated with computational fluid dynamics analysis. To address these challenges, this paper introduces a highly automated approach for supercritical airfoil design, called Evolutionary Generative Design (EvoGD). The EvoGD approach is based on the framework of Evolutionary Computation and employs a series of sophisticated data-driven generative models incorporated with physical information to iteratively refine initial airfoil shapes, resulting in improved aerodynamic performances and reduced constraint violations. Moreover, to speed up the evaluation of the generated airfoils, a series of accurate and efficient data-driven predictors are utilized. The efficacy of the EvoGD approach was demonstrated through experiments on a dataset of 501 supercritical airfoils, including one baseline design and 500 randomly perturbed airfoils. On average, the generated airfoils showed improved performance in terms of buffet lift coefficient, cruise lift-to-drag ratio, and thickness by 5%, 4%, and 1%, respectively. The best generated airfoil outperformed the baseline design in terms of critical buffet lift coefficient and cruise lift-to-drag ratio by 7.1% and 6.4%, respectively. The entire design process was completed in less than an hour on a personal computer, highlighting the high efficiency and scalability of the EvoGD approach.

Keywords Evolutionary computation · Artificial neural network · Generative learning · Supercritical airfoil design

Abbreviations

AI	Artificial intelligence	CVAE-GAN	Conditional variational autoencoder with generative adversarial network
ANN	Artificial neural network	Cycle GAN	Cycle generative adversarial network
BPO	Bernstein polynomial order	DE	Differential evolution
CFD	Computational fluid dynamics	DNN	Deep neural network
CGAN	Conditional generative adversarial network	EC	Evolutionary computation
CNN	Convolutional neural network	EvoGD	Evolutionary generative design
CST	Class shape transformation	GAN	Generative adversarial network
		GP	Gaussian process
		IWO	Invasive weed optimization
		MLP	Multi-layer perceptron
		MSE	Mean squared error
		NURBS	Non-uniform rational B-spline
		PARSEC	Parametric section
		PDF	Probability density function
		POD	Proper orthogonal decomposition
		PSO	Particle swarm optimization
		RANS	Reynolds average Navier–Stokes
		SST	Shear stress transport
		SVD	Singular value decomposition

Kebin Sun and Weituo Wang have contributed equally to this work and should be considered co-first authors.

✉ Ran Cheng
ranchengcn@gmail.com

¹ Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China

² Institute of Unmanned System, Beihang University, Beijing 100191, China

³ Shanghai Aircraft Design and Research Institute, Shanghai 200436, China

Introduction

Modern commercial wide-body aircrafts often operate at relatively high speeds, typically exceeding 0.8 Mach or 980 km/h. The presence of high-speed transonic air flows in certain regions around the aircraft, particularly above the wings, requires mitigation. Supercritical airfoils are utilized as cross sections in the design of commercial wide-body aircraft wings to decrease drag, improve lift, and increase stability and comfort. Given the growing importance of environmental and economic efficiency in commercial wide-body aircraft design, well-designed supercritical airfoils are crucial for success.

However, the design of supercritical airfoils presents a number of challenges, particularly the requirement for precise determination of shape parameters as aerodynamic performance is extremely sensitive to small geometric variations. This design process often necessitates the involvement of experienced professionals and multiple rounds of design iterations, simulations, and real-world testing.

In recent years, the rapid advancement of Artificial Intelligence (AI) has led to an increase in attention towards AI-assisted positive [1–6] and inverse [7, 8] supercritical airfoil design. Despite the notable progress made in the field, its development is still in its infancy and several critical open challenges remain to be addressed.

❶ **Costly datasets.** Existing AI-assisted supercritical airfoil design methods often require a significant number of samples for effective functioning. However, the collection of large-scale datasets through computationally costly and sometimes labor-intensive Computational Fluid Dynamics (CFD) analyses can be prohibitively expensive.

❷ **Complicated physical constraints.** Existing AI-assisted supercritical airfoil design approaches commonly use supercritical airfoil parametrization methods such as sampled coordinates or plain images. However, these approaches do not effectively encode the rigid and complex physical constraints of supercritical airfoils, resulting in potentially unstable design outcomes.

❸ **Inefficient design generation.** Existing AI-assisted supercritical airfoil design methods frequently rely on random or stochastic sampling for the generation of new airfoils. However, due to the highly sensitive nature of aerodynamic performance to design variables, these conventional AI methods can be inefficient in producing high-quality designs.

❹ **Imbalanced diversity and reliability.** Existing AI-assisted supercritical airfoil design methods either directly optimize a limited number of designs or do not incorporate local search operations. However, given the complex nature of supercritical airfoil design, these AI approaches may only

produce airfoils that are either diverse or reliable, but not both.

The Evolutionary Computation (EC) paradigm has been widely recognized for its power in solving optimization problems [9]. In recent years, the application of data-driven EC has shown promise in optimizing real-world problems [10]. However, despite efforts to apply data-driven EC to airfoil design [11–13], the precision of the designs remains subpar, particularly in the case of supercritical airfoils, which require higher precision as their aerodynamic performance is more sensitive than that of subcritical airfoils.

The limitations in precision of data-driven EC can be attributed mainly to the fact that previous studies have primarily focused on surrogate modeling of evaluators, leaving the EC framework underutilized in terms of fully exploiting the potential of data for solution generation. Although recent research has investigated the use of generative learning in EC for efficient solution generation [14], its low precision remains a challenge to its practical application in tasks such as airfoil design.

Therefore, to address the challenges faced by existing AI-assisted supercritical airfoil design methods by fully leveraging the potential of EC, this work proposes an Evolutionary Generative Design (EvoGD) approach. On the one hand, the framework of EC provides an optimal platform for the generation of diverse and innovative supercritical airfoils. On the other hand, filled by carefully designed data-driven evaluators and generators, the proposed EvoGD approach is able to generate and identify high-performing supercritical airfoils with a high degree of accuracy and efficiency. The main contributions of this work are:

- The proposed approach in this work offers a departure from the traditional supercritical airfoil design process, which often involves the expertise of experienced professionals and multiple rounds of design iterations, simulations, and real-world testing. In contrast, the proposed approach is purely data-driven, offering the potential to significantly reduce the human effort and computational costs involved in supercritical airfoil design.
- To address challenge ❶, the proposed approach in this work leverages the EC framework and integrates it with a unique type of generative models for accommodating unpaired inputs, thus allowing for effective operation even on small datasets.
- To address challenge ❷, the proposed approach in this work conducts an empirical comparison of various parameterization methods for supercritical airfoils. Based on the comparison, a suitable parameterization method is selected that is capable of effectively capturing most of the physical constraints of supercritical airfoils.
- To address challenge ❸, the proposed approach incorporates an AI generative model that is capable of directly

generating high-performing supercritical airfoils with improved aerodynamic and geometric characteristics. Additionally, the approach minimizes the number of designs that violate physical constraints, ensuring a robust design solution.

- To address challenge ④, the proposed approach in this work employs a multi-objective EC framework integrated with fast and accurate data-driven supercritical airfoil generators and performance predictors. This allows for the inherent consideration of both design diversity and design reliability in the proposed approach.
- To accommodate the proposed approach with small data, this work introduces a new generative model training method. First, it incorporates physics information into the training process to boost the accuracy and efficiency. Second, it employs a combination of multiple generative models rather than a single one, to enhance robustness at the system level.

The remainder of this paper is organized as follows. In “[Related work](#)” section, we provide a comprehensive overview of previous studies on the application of AI to airfoil design. In “[Problem formulation](#)” section, we justify our choice of parameterization method and formulate the optimization problem for supercritical airfoil design. In “[Method](#)” section, we elaborate our proposed EvoGD approach and provide a thorough description of its implementation. In “[Experimental study](#)” section, we demonstrate the comprehensive experimental results. In “[Conclusion](#)” section, we conclude this paper with a summary of our findings and future directions for research.

Related work

In recent years, the application of AI to airfoil design, particularly supercritical airfoils, has become a topic of increasing interest due to the prohibitive computational expense of traditional CFD solvers. Early work by Yilmaz and German [15] utilized Convolutional Neural Networks (CNNs) to predict subcritical airfoil performance, which was later modified by Hui et al. [16] to include wall pressure coefficients. In addition to the use of data-driven surrogate models as performance predictors, there has also been a growing body of research on data-driven airfoil shape generation models. For instance, Sekar et al. [17] used CNNs for inverse airfoil design, where a subcritical airfoil was generated from a given wall pressure distribution. On the other hand, direct airfoil generation models have been explored, such as the use of Generative Adversarial Networks (GANs) by Chen et al. [18] and the use of Conditional Generative Adversarial Networks (CGANs) by Yilmaz [19] for the generation of subcritical airfoils with desired aerodynamic performance.

Despite the ease of generalization to different airfoil design problems offered by these methods, which often adopt plain images as representations for airfoil shapes and prediction results, they may not be directly applicable to supercritical airfoil design due to the highly sensitive nature of aerodynamic performance to small changes in supercritical airfoil shape.

The application of AI-assisted approaches towards supercritical airfoil design has received growing attention in recent years. A number of studies have proposed utilizing AI optimization methods and/or AI predictors to address the challenges in this field. For example, Wang et al. [1] used a modified Particle Swarm Optimization (PSO) method and a simple back-propagation neural network as the performance predictor. Liu et al. [2] adopted a hybrid of Differential Evolution (DE) and Invasive Weed Optimization (IWO) methods combined with a Gaussian Process (GP) as the predictor. The rapid advancements in AI techniques, particularly deep learning, have also shown promising potential in this field. For example, Bouhleb et al. [4] employed a CFD-enhanced Artificial Neural Network (ANN) to achieve a prediction error of 0.48%. Wu et al. [20] combined CNN and (GAN) to establish a mapping between supercritical airfoils and transonic flow fields. Wang et al. [8] employed a Conditional Variational Autoencoder with Generative Adversarial Network (CVAE-GAN) to generate supercritical airfoils’ wall Mach number distributions and a Deep Neural Network (DNN) to map them back to supercritical airfoils. These works have demonstrated the potential of utilizing AI techniques in supercritical airfoil design.

Despite the recent progress in AI-assisted supercritical airfoil design, the development of the field is still in its infancy, and there remain several technical details that require further investigation. For example, the precision of existing data-driven predictors can be limited when working with a small dataset, and the combination of high-performing predictors with simple, non-explorative AI optimization methods may not yield optimal results. Furthermore, the sophisticated AI generators currently in use do not guarantee the generation of supercritical airfoils with improved performance compared to existing designs. These limitations have underscored the ongoing need for further research and advancements in AI-assisted supercritical airfoil design.

Problem formulation

In this section, we will first introduce the NURBS as the method for parameterizing the supercritical airfoils. Then, the problem of supercritical airfoil design optimization will be mathematically formulated by defining its optimization objectives and constraints.

Table 1 Summarized features of several commonly adopted parametrization methods for supercritical airfoils

Properties of design variables (q)	Sampled coordinates	CST	PARSEC	POD	NURBS
Numbers to reach mean error of 10^{-5}	> 200	~ 18	Cannot reach 10^{-5}	~ 12	~ 16
Physical meanings	Direct Cartesian coordinate	Amplitude of functions	Angles and distances	Amplitude of modes	Cartesian control points
Robustness	Easily violating constraint	High	High	Low for too many modes	High
Effect of single variable modification	Local	Global	Regional	Global	Regional
$\max \{ q \}$	~ 100	~ 5	~ 5	$\sim 10^4$	~ 5
$\min \{ q \}$					

The sample coordinates method requires the largest number of design variables to reach the same error. Hence, it becomes imperative to utilize in excess of 200 variables for fitting an authentic (i.e., smoothly defined) supercritical airfoil using this method. The SVD method has the largest magnitude ratio of design variables ($\max \{ |q| \} / \min \{ |q| \}$). Generally, the relative importance of the design variables manifests a more pronounced variation when this value escalates. The design variables of the CST and SVD methods are not particularly physically meaningful as they have global effects when modifying a single variable. Overall, the NURBS is a much more balanced choice

Parametrization method

Since we are considering supercritical airfoils whose aerodynamic performances are highly sensitive to shape modifications and defects, the reasonable design variables shall be the parameterized ones that perfectly describe supercritical airfoil shapes. Apparently, the design variables of plain images are impractical due to the relatively low precision and too much redundant variables. As summarized in Table 1, although there are also other specified parameterization methods for supercritical airfoil design, not all of them are suitable in our case. For example, a commonly adopted method was proposed by Kulfan et al. [21], where class and shape functions are used to represent geometric Class Shape Transformations (CSTs) and fitted to many supercritical airfoils with an experienced Bernstein Polynomial Order (BPO) as shape functions. However, one defect of this method is that their parameters are not quite physically meaningful. Another commonly used method is the Parametric Section (PARSEC) method proposed by Sobieczky [22]. This method has all physically meaningful parameters that correspond only to smooth airfoils with high robustness, but it cannot precisely control supercritical airfoil shapes.

Out of comprehensive considerations, we have chosen the Non-Uniform Rational B Spline (NURBS) curve [23] as our parametrization method in this work. Below are detailed justifications.

First, sampled coordinates can be converted to a NURBS curve easily. The general form of a NURBS curve, $\vec{C}(u) \equiv [C_x(u), C_y(u)]$, is given by:

$$\vec{C}(u) = \sum_{i=1}^m \frac{N_{i,o}(u)w_i}{\sum_{j=1}^m N_{j,o}(u)w_j} \vec{P}_i, \quad (1)$$

where $u \in [0, 1]$ signifies the parameter of the function $\vec{C}(u)$, while $\vec{P}_i \equiv [P_{i,x}, P_{i,y}]$ denotes the i th control point. The array $\mathbf{P} = [\vec{P}_1, \dots, \vec{P}_m]^T$ represents the collective set of all control points. Further, w_i refers to the weight of \vec{P}_i , and $N_{i,o}(u)$ is the i th basis function of a Non-Uniform Rational B-Spline (NURBS) curve of order o . Notably, the vector notation corresponds to either a vector or a point in a two-dimensional Cartesian coordinate system. It can be shown that it is simple to fit a NURBS curve to a set of coordinates $\mathbf{X} = [\vec{x}_1, \dots, \vec{x}_n]^T$, which is equivalent to solving for \mathbf{P} in the least-squares solution of the system of equations:

$$\mathbf{N}\mathbf{P} = \mathbf{X}, \quad (2)$$

where

$$N_{i,j} = N_{j,o} \left(\frac{\sum_{l=1}^{i-1} \|\vec{x}_{l+1} - \vec{x}_l\|}{\sum_{l=1}^{n-1} \|\vec{x}_{l+1} - \vec{x}_l\|} \right) \quad (3)$$

is the basis matrix.

Second, NURBS can achieve high representation accuracy with a relatively small number of control points (i.e., design variables). For example, in our study, a NURBS curve can achieve a maximum relative error deviation of 10^{-5} with only 18 control points of order 3. By further removing redundant variables, the actual number of design variables can be even reduced to 16 real values, represented by the vector \mathbf{q} , i.e., all the y -coordinates except the first and last control points. The control points \mathbf{P} can then be reconstructed using a simple function $\mathbf{P}(\mathbf{q})$, where $\mathbf{P} : \mathcal{Q} \rightarrow \mathcal{P}$ and $\mathbf{q} \in \mathcal{Q}, \mathbf{P}(\mathbf{q}) \in \mathcal{P}$.

Third, the relative geometric flexibility of NURBS makes it desirable for use in our EvoGD approach. This feature provides us with an enhanced level of control over the details of supercritical airfoil shapes, making it possible to generate novel and diverse supercritical airfoils that are distinct from those in the dataset.

Additionally, the use of the Cartesian coordinate system for the design variables makes it easier for our AI generators and predictors to model and calculate the validity of designs. Furthermore, the actual supercritical airfoil shape depends almost linearly and locally on the NURBS control points, which helps to avoid the issue of generating designs with small non-smoothness or fluctuations that can occur in Proper Orthogonal Decomposition (POD) [24] and sampled coordinate methods.

Optimization objectives and constraints

For our proposed EvoGD approach to be practical and applicable in real-world scenarios, we consider several optimization objectives, including: (1) aerodynamic performance, (2) geometric performance, and (3) a series of aerodynamic and geometric constraints. In particular, we take the following geometric and aerodynamic performances of supercritical airfoils into consideration:

- h : Thickness. This is the maximum distance between the upper and lower surfaces of the supercritical airfoil, as illustrated in Fig. 1.
- $K(u)$: Convex metric. A smooth supercritical airfoil implies that its convex metric has one and only one zero point on the lower surface.
- α_0 : Cruise angle of attack. This is the angle at which the supercritical airfoil produces a lift coefficient of $C_{L,0}$, which is 0.8 in this work, as shown in Fig. 2.
- C_{LD} : Cruise lift-to-drag ratio. This is defined as $C_{L,0}/C_D(\alpha_0)$, where C_D is the drag coefficient; it is depicted in Fig. 2 as well.

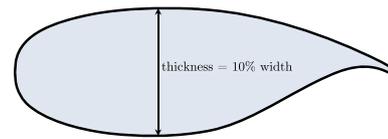


Fig. 1 The shape of NASA SC(2)-0410 supercritical airfoil. The digits “04” represent the theoretical optimal lift coefficient; the digits “10” represent that the maximum thickness is 10% of its total width or chord length c

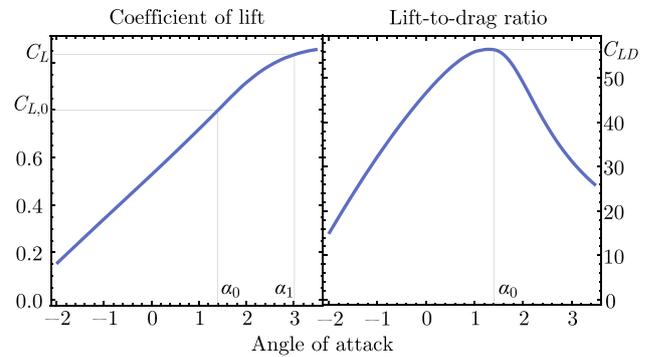


Fig. 2 The curves of lift coefficients and lift-to-drag ratios vs. the angles of attack for the baseline supercritical airfoil. The physical meanings of α_0 , C_{LD} and C_L are depicted: at the cruise angle of attack α_0 , the lift coefficient is exactly $C_{L,0}$; at the same angle, the lift-to-drag ratio is the C_{LD}

- C_L : Buffet lift coefficient. This is the lift coefficient at the buffet angle of attack, which is defined as

$$\alpha_l = (C'_L)^{-1} \left\{ C'_L \left[\operatorname{argmax}_{\alpha_l \leq \alpha \leq \alpha_h} |C''_L(\alpha)| \right] - k_{\text{diff}} \right\},$$

where $\alpha_l = -2$, $\alpha_h = 1$ and $k_{\text{diff}} = 0.1$ for this work. The buffet lift coefficient is also shown in Fig. 2.

- $C_P(\alpha_0)$: Wall pressure coefficients at cruise angle of attack. In practice, the suction peak point F_{sp} is the highest value of the wall pressure coefficients before maximum thickness, while the shock wave point F_{sw} is the wall pressure coefficient at the wall Mach number of 1. An acceptable wall pressure coefficient should have limited decrease from F_{sp} to F_{sw} and contain exactly one F_{sw} . This is plotted in Fig. 3.

In this work, we select C_L , C_{LD} and h as our optimization objectives, and α_0 , $C_P(\alpha_0)$, h and $K(u)$ as our constraints. These values are functions of the NURBS control points, represented by the design variables \mathbf{q} . For the rest of the paper, these values will be written as $C_L(\mathbf{q})$ and so on.

The cruise angle of attack $\alpha_0(\mathbf{q})$ must lie within a specified range, such as:

$$\alpha_{\min} \leq \alpha_0(\mathbf{q}) \leq \alpha_{\max}, \tag{4}$$

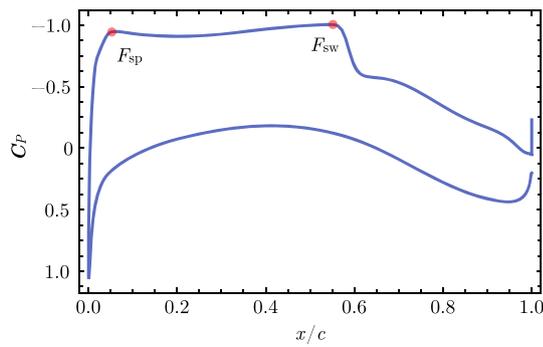


Fig. 3 The curve of wall pressure coefficients (C_p) vs. the horizontal coordinates (x/c). The chord length c is set to 1 throughout this work. The suction peak point F_{sp} and the shock wave point F_{sw} are the key features. The suction peak point F_{sp} is the highest value of the wall pressure coefficients before the maximum thickness, while the shock wave point F_{sw} is the wall pressure coefficient at the wall Mach number of 1

where $\alpha_{min} = 1^\circ$ and $\alpha_{max} = 2^\circ$ are commonly used. Additionally, the suction peak $F_{sp}(\mathbf{q}, \alpha_0(\mathbf{q}))$ should not exceed $F_{sp,0} \approx -1.2$ to avoid sharp leading edges. Similarly, the difference between the absolute values of $F_{sw}(\mathbf{q}, \alpha_0(\mathbf{q}))$ and $F_{sp}(\mathbf{q}, \alpha_0(\mathbf{q}))$ must lie in a certain range, typically 0.05:

$$\begin{cases} F_{sp}(\mathbf{q}, \alpha_0(\mathbf{q})) \geq F_{sp,0} \\ 0.05 \leq F_{sw}(\alpha_0) - F_{sp}(\alpha_0) \leq 0.05 \end{cases} \quad (5)$$

Moreover, the convex metric of the supercritical airfoil must satisfy the following conditions:

$$\begin{aligned} K(u) &\equiv x'(u)y''(u) - x''(u)y'(u) \\ &\begin{cases} \geq 0, & u \in [0, 1/2) \cup (u_c, 1] \\ \leq 0, & \text{otherwise,} \end{cases} \end{aligned} \quad (6)$$

where $[x(u), y(u)] \equiv \vec{N}_o^T(u)\mathbf{P}(\mathbf{q})$ is the parametric representation of the supercritical airfoil defined by the design variables \mathbf{q} in terms of the NURBS basis functions $\vec{N}_o^T(u)$ and the control points $\mathbf{P}(\mathbf{q})$, and $[x(u_c), y(u_c)]$ is the point on the lower surface of the supercritical airfoil where the curve changes from convex upward to concave upward.

Finally, to prevent extrapolation outside the range of our prediction models, we also consider the maximum values of $C_L(\mathbf{q})$ and $C_{LD}(\mathbf{q})$ among the given dataset \mathcal{D} :

$$\begin{cases} C_L(\mathbf{q}) < (1 + \varepsilon) \max_{\mathbf{q} \in \mathcal{D}} C_L(\mathbf{q}) \\ C_{LD}(\mathbf{q}) < (1 + \varepsilon) \max_{\mathbf{q} \in \mathcal{D}} C_{LD}(\mathbf{q}) \end{cases}, \quad (7)$$

where ε is a small number that has been validated to be safe for prediction extrapolation.

Based on the above, problem formulation of the supercritical airfoil design is summarized in Table 2. In general, the

Table 2 The optimization objectives and constraints of the supercritical airfoil design in this work. d is short for dimension

	Formulation	d
Design variable	\mathbf{q} from NURBS control points	16
Optimization objectives	$\max_{\mathbf{q}} C_L(\mathbf{q}), \max_{\mathbf{q}} C_{LD}(\mathbf{q}), \max_{\mathbf{q}} h(\mathbf{q})$	3
Constraints	$\alpha_{min} \leq \alpha_0(\mathbf{q}) \leq \alpha_{max},$ $F_{sp}(\mathbf{q}, \alpha_0(\mathbf{q})) \geq F_{sp,0},$ $ F_{sp}(\mathbf{q}, \alpha_0(\mathbf{q})) - F_{sw}(\mathbf{q}, \alpha_0(\mathbf{q})) \leq 0.05,$ $h(\mathbf{q}) \geq h_{min},$ $C_L(\mathbf{q}) < (1 + \varepsilon) \max_{\mathbf{q} \in \mathcal{D}} C_L(\mathbf{q}),$ $C_{LD}(\mathbf{q}) < (1 + \varepsilon) \max_{\mathbf{q} \in \mathcal{D}} C_{LD}(\mathbf{q}),$ $K(u) \begin{cases} \geq 0, & u \in [0, 1/2) \cup (u_c, 1] \\ \leq 0, & \text{otherwise} \end{cases}$	

optimization objectives are highly conflicting, particularly between h and C_L, C_{LD} . Hence, h is not only a constraint but also an optimization objective in this work. This modification may make the design process more challenging, but it can also result in improved outcomes for supercritical airfoil design.

Method

The proposed EvoGD approach for supercritical airfoil design consists of four components in its main iterative loop: Generator, Evaluator, Selector, and Interactor, as illustrated in Fig. 4. In each iteration, first, the Generator generates a new batch of supercritical airfoils based on the information obtained from the previous iteration, with the goal of improving the design diversity and performance of the existing supercritical airfoils; the Evaluator is then utilized to assess the aerodynamic performances and constraint violations of the new batch of airfoils; afterwards, the Selector selects high-quality airfoils by ranking with optimization objectives of C_L, C_{LD} and h , as well as constraints of $\alpha_0, C_L, C_{LD}, C_p(\alpha_0)$ and h ; finally, the designer has the option to interact with the process via Interactor before the next iteration begins.

In the case of supercritical airfoil design, the Selector is not data-driven since all optimization objectives and constraints are deterministic and well-defined, and thus it is not imminent to supply Selector with data. Hence, we adopt the selection method of [25] as our Selector. However, in other cases where selection process is nondeterministic or ill-defined, the proposed EvoGD can still be applied with an implementation of data-driven selector(s). Moreover, even though the Interactor in this work is merely used to dis-

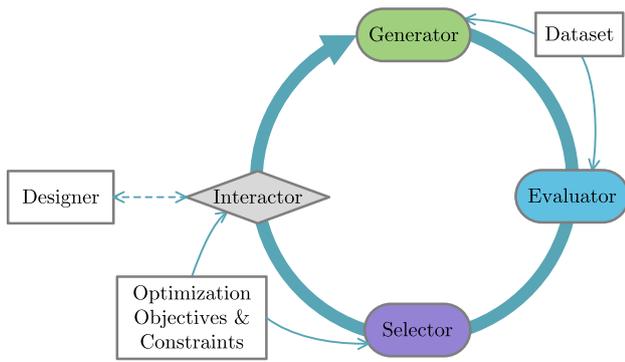


Fig. 4 Main loop of the proposed EvoGD approach. There are four main components—Generator, Evaluator, Selector, and Interactor. In the case of supercritical airfoil design, both Generator and Evaluator are purely data-driven ones, where Selector and Interactor are not

play the results to the designer, it can also show the selected supercritical airfoils to designers and asking for new information such as aerodynamic performances to update data-driven models on the fly. Specifically, the data-driven Generator and Evaluator are further elaborated in the following subsections.

Generator

Our Generator component further encompasses several data-driven generators, which offer a more efficient alternative to manual supercritical airfoil design. The generators previously adopted in AI-assisted supercritical airfoil design are instances of the conditional probability

$$\mathbb{P}(\mathbf{Q}|T = t), \tag{8}$$

where $T \equiv T(\alpha_0, C_L, C_{LD}, C_P, h)$ represents the joint distribution of target performances, t represents an arbitrary instance of T , and \mathbf{Q} represents the joint distribution of design variables \mathbf{q} . Various generators, such as Variational Autoencoders (VAEs) [26], Generative Adversarial Networks (GANs) [27] Conditional Generative Adversarial Networks (CGANs) [28], and SoloGAN [29], have been introduced in previous studies to model this Probability Density Function (PDF).

However, the target aerodynamic performance space spanned by T in the training dataset for this model is often limited and discrete, leading to an uneven distribution. This can result in generators’ failure in accurately modeling the target performance space and the generated supercritical airfoils may even be infeasible. To address the issue of uneven distribution of the target aerodynamic performance space, in this work, we adopt the following model:

$$q(q') = \mathbb{E}(q|q') = \int q_T \mathbb{P}(\mathbf{Q} = q_T | q_{T'}) dq_T, \tag{9}$$

where $q_{T'} \sim \mathbb{P}(\mathbf{Q}|T = t')$ and $q_T \sim \mathbb{P}(\mathbf{Q}|T = t)$ represent two conditional probabilities in a traditional generator, and $t' \preceq t$ indicates that the target performances of t' are generally worse than t 's.

The adopted model in this work can be considered as a redesign/refine model that generates a new supercritical airfoil based on learned dominance relations between the aerodynamic performances. Local and global search operators for optimization objectives such as C_L , C_{LD} , and h , as well as their combination, are examples of such generators. To implement these search operators, Artificial Neural Networks (ANNs) are commonly utilized. Specifically, in this work, we use the generators in the Cycle Generative Adversarial Networks (Cycle GANs) [30].

Notably, the model described in Eq. (9) can be viewed as a generative model that integrates physics information, resulting in the inherent consideration of physical performance and constraints during the training of our Cycle GANs. This integration of physics information enhances the stability of the training process, while the use of multiple generators for each physical performance improves the robustness of the entire Generator. Further details of the training process can be found in “Model training” section.

As illustrated in Fig. 5, the Cycle GAN is originally designed to find a pair of generators mapping between two domains, \mathcal{X} and \mathcal{Y} , using two generator-discriminator pairs: $G : \mathcal{X} \rightarrow \mathcal{Y}$ with D_G and $F : \mathcal{Y} \rightarrow \mathcal{X}$ with D_F . The discriminators, D_G and D_F , aim to minimize the classification error rates between real and generated fake inputs, while the generators, G and F , aim to maximize the discriminators’ classification error rates for fake inputs. Additionally, the Cycle GAN minimizes the cyclic consistency losses, $\|F(G(\mathbf{x})) - \mathbf{x}\|$ and $\|G(F(\mathbf{y})) - \mathbf{y}\|$, where $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$ are samples from the two domains.

Specifically, there are four different loss functions in training the Cycle GAN:

$$\mathcal{L}_{GF}, \mathcal{L}_{FG}, \mathcal{L}_{D_G} \text{ and } \mathcal{L}_{D_F}. \tag{10}$$

They can be written as

$$\begin{aligned} \mathcal{L}_{GF} &= \|F(G(\mathbf{x})) - \mathbf{x}\|, \\ \mathcal{L}_{FG} &= \|G(F(\mathbf{y})) - \mathbf{y}\|, \\ \mathcal{L}_{D_G} &= \log D_G(\mathbf{y}) + \log [1 - D_G(G(\mathbf{x}))], \end{aligned}$$

and

$$\mathcal{L}_{D_F} = \log D_F(\mathbf{x}) + \log [1 - D_F(F(\mathbf{y}))].$$

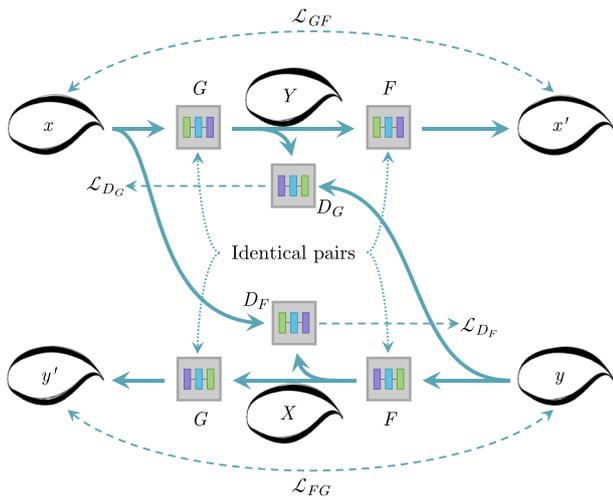


Fig. 5 Structure of a Cycle GAN. The input samples from domain \mathcal{X} are input as x and the samples from domain \mathcal{Y} are input as y . The generator G maps the samples from domain \mathcal{X} to domain \mathcal{Y} , generating the fake 2 output Y . Likewise, the generator F , maps the samples from domain \mathcal{Y} to domain \mathcal{X} , generating the fake 1 output X . The discriminators D_F , D_G receive x and X , y and Y , respectively, and output cross-entropy losses \mathcal{L}_{D_F} and \mathcal{L}_{D_G} , respectively, for classifying real and fake supercritical airfoils. The fake outputs X and Y are also sent to their corresponding generators G and F , respectively, to produce the reconstructed supercritical airfoils y' and x' . The cyclic consistency losses \mathcal{L}_{GF} and \mathcal{L}_{FG} of the Cycle GAN is the difference between x, x' and y, y' , respectively. The model minimizes the \mathcal{L}_{GF} and \mathcal{L}_{FG} with respect to both generators G and F . In the mean time, it also minimizes the \mathcal{L}_{D_G} and \mathcal{L}_{D_F} with respect to the D_G and D_F , respectively. Additionally, like in a traditional GAN, the classification errors for Y and X are inverted to train G and F , respectively

To train the discriminators D_G and D_F , the loss functions \mathcal{L}_{D_G} and \mathcal{L}_{D_F} are minimized with respect to the parameters of D_G and D_F , respectively. Since the data never flows between the discriminators, it is equivalent to minimize

$$\mathcal{L}_{\text{discriminator}} = \log D_G(y) + \log [1 - D_G(G(x))] + \log D_F(x) + \log [1 - D_F(F(y))] \tag{11}$$

with respect to the parameters of D_G and D_F combined.

To train the generators G and F , the loss function $\mathcal{L}_{\text{generator}}$ is minimized with respect to the parameters of G and F combined:

$$\mathcal{L}_{\text{generator}} = \lambda (\mathcal{L}_{GF} + \mathcal{L}_{FG}) - \log [1 - D_G(G(x))] - \log [1 - D_F(F(y))], \tag{12}$$

where λ is the coefficient that characterizes the importance of cyclic consistency losses in training.

When adopting Cycle GAN in the proposed EvoGD approach, the input datasets are carefully divided based on the aerodynamic and geometric performance objectives C_L , C_{LD} , and h . The original dataset consists of design variables of supercritical airfoils (NURBS control points) and their

respective performance objectives, defined as:

$$\mathcal{D} = \{(\mathbf{q}_1, C_{L,1}, C_{LD,1}, h_1), \dots, (\mathbf{q}_{501}, C_{L,501}, C_{LD,501}, h_{501})\}. \tag{13}$$

It is then divided into three pairs of input datasets:

$$\begin{aligned} \mathcal{Q}_{1,1} &\equiv \{\mathbf{q}_i \in \mathcal{D} | C_{L,i} > C_{L,\epsilon}\}, \\ \mathcal{Q}_{1,2} &\equiv \{\mathbf{q}_i \in \mathcal{D} | C_{L,i} < C_{L,\epsilon}\}, \\ \mathcal{Q}_{2,1} &\equiv \{\mathbf{q}_i \in \mathcal{D} | C_{LD,i} > C_{LD,\epsilon}\}, \\ \mathcal{Q}_{2,2} &\equiv \{\mathbf{q}_i \in \mathcal{D} | C_{LD,i} < C_{LD,\epsilon}\}, \\ \mathcal{Q}_{3,1} &\equiv \{\mathbf{q}_i \in \mathcal{D} | h_i > h_\epsilon\}, \\ \mathcal{Q}_{3,2} &\equiv \{\mathbf{q}_i \in \mathcal{D} | h_i < h_\epsilon\}, \end{aligned} \tag{14}$$

where $C_{L,\epsilon}$, $C_{LD,\epsilon}$ and h_ϵ are the tunable separation thresholds of three targets. Specifically, multiple thresholds are used for each target t_i to train different generators. To further enhance the performances of our generators, another pair of input dataset is also provided:

$$\mathcal{Q}_{4,1} \equiv \{\mathbf{q}_i \in \mathcal{Q}_D | t \gtrsim t_\epsilon\}, \mathcal{Q}_{4,2} \equiv \{\mathbf{q}_i \in \mathcal{Q}_D | t \lesssim t_\epsilon\}, \tag{15}$$

which are divided by Pareto dominance relations of all objectives.

As mentioned earlier, our generators can be represented as

$$G_i = \mathbb{E}[\mathbf{q}_i | \mathcal{Q} = \mathbf{q}_{t'_i}], \tag{16}$$

where $\mathbb{E}(t_i) > \mathbb{E}(t'_i)$. The generator G_i maps the supercritical airfoil design variable space with lower target performance t'_i to the one with higher target performance t_i . In our implementation, the generator G_i is the generator G of the i th Cycle GAN, trained by providing samples to y and x inputs with $\mathcal{Q}_{i,1}$ and $\mathcal{Q}_{i,2}$, respectively. Upon convergence, the generators will possess the ability to refine the given supercritical airfoils to produce new ones with better optimization objectives at high probabilities. In addition, the use of random divisions to separate median samples from the original dataset increases the diversity of generated supercritical airfoils.

Evaluator

Our Evaluator is also composed of data-driven predictors, which serve as computationally inexpensive surrogates for estimating the aerodynamic performances of supercritical airfoils, rather than relying on computationally intensive CFD simulations.

In general, a predictor is an instance of the discriminative/regression model—a model of conditional probability

$$\mathbb{P}(T | \mathbf{Q} = \mathbf{q}), \tag{17}$$

where \mathbf{q} is the given supercritical airfoil’s design variables (obtained from NURBS curve control points) and $T \equiv T(\alpha_0, C_L, C_{LD}, C_P)$ is the target aerodynamic performances’ joint distribution. To optimize both buffet and cruise performance of a supercritical airfoil, we need to model and predict its cruise angle of attack

$$\alpha_0(\mathbf{q}) = \mathbb{E}(\alpha_0 | \mathbf{q}) \equiv \int \alpha \mathbb{P}(A_0 = \alpha | \mathbf{Q} = \mathbf{q}) d\alpha, \tag{18}$$

its buffet lift coefficient

$$C_L(\mathbf{q}) = \mathbb{E}(C_L | \mathbf{q}) \equiv \int c \mathbb{P}(C_L = c | \mathbf{Q} = \mathbf{q}) dc, \tag{19}$$

its cruise lift-to-drag ratio

$$C_{LD}(\mathbf{q}) = \mathbb{E}(C_{LD} | \mathbf{q}) \equiv \int c \mathbb{P}(C_{LD} = c | \mathbf{Q} = \mathbf{q}) dc, \tag{20}$$

and its cruise surface pressure coefficients

$$\begin{aligned} C_P(\mathbf{q}) &= \mathbb{E}(C_P | \alpha_0; \mathbf{q}) \\ &\equiv \int c \mathbb{P}(C_P = c | A = \alpha_0(\mathbf{q}); \mathbf{Q} = \mathbf{q}) dc, \end{aligned} \tag{21}$$

where the vector notation of C_P indicates that it is a vector representation of curve at any given angle of attack and supercritical airfoil. It can be noticed that the angle of attack is not a condition variable for the first three predictors, but we directly target the aerodynamic performances at cruise and buffet angles of attack. This is crucial for these predictors to work properly on a dataset with an unprecedented small size of 501.

Specifically, each predictor is a Multi-Layer Perceptron (MLP) [31] composed of an input layer, two–three hidden full-connection layers, and an output layer. The adopted MLP is simple and efficient, with input layers consisting of the 16 design variables of the supercritical airfoil, output layers being the corresponding prediction values of size 1 or 30, and hidden layers of size 60–100. The hidden layer configurations vary between different predictors due to differences in purposes and complexities. All predictive models use Mean Squared Error (MSE) as the loss function \mathcal{L} :

$$\mathcal{L} = \|\mathbf{n}_{\text{last}} - \mathbf{n}_{\text{real}}\|^2, \tag{22}$$

where \mathbf{n}_{last} are the output values of the last layer of the MLP and \mathbf{n}_{real} are the actual values to predict.

For each predictor for cruise angle of attack ($\alpha_0(\mathbf{q})$), buffet lift coefficient ($C_L(\mathbf{q})$), and cruise lift-to-drag ratio ($C_{LD}(\mathbf{q})$), a MLP with two hidden layers and hyperbolic tangent activation function is employed. To ensure proper performance, the α_0 , C_L , and C_{LD} values in the dataset are first normalized and any anomalies are removed. After this preprocessing, the three MLPs are trained, one for each value to be predicted.

For the predictor of cruise surface pressure coefficients $C_P(\mathbf{q})$, the Proper Orthogonal Decomposition (POD) method is utilized. The POD method transforms the original data points of C_P and extracts the first few modes for prediction. In this case, only 30 modes are required for the decomposition for achieve a high prediction accuracy. To further improve the overall performance of the predictive model of $C_P(\mathbf{q})$, the $C_P(\alpha; \mathbf{q})$ of all provided α values are included in the dataset, using the regression model:

$$\begin{aligned} C_P(\alpha; \mathbf{q}) &= \mathbb{E}(C_P | \alpha; \mathbf{q}) \\ &\equiv \int c \mathbb{P}(C_P = c | A = \alpha; \mathbf{Q} = \mathbf{q}) dc, \end{aligned} \tag{23}$$

where c is the POD modes’ singular values. The MLP structure for this predictor is compact, consisting of only three hidden layers of size 100–200, with a hyperbolic tangent activation function.

Selector

As articulated above, in the context of supercritical airfoil design, all optimization objectives and constraints are deterministic. Consequently, our Selector does not rely on a data-driven approach. The general process of the Selector can be summarized as follows:

1. Compute all objectives and violations of constraints for the current population;
2. Determine the Pareto rank r utilizing the fast non-dominant sort method as proposed in NSGA-II [25];
3. Compute the shared fitness values by

$$f_i = \frac{1}{r_i \cdot \sum_{j=1}^N \max\{1 - [\|F_i - F_j\|], 0\}},$$

where F_i is the rescaled objectives of individual i and N is the number of individuals in the population;

4. Adjust their shared fitness f_i by subtracting their constraint violations;
5. The subsequent generation is selected randomly by weights $w_i = e^{f_i} / \sum_{j=1}^N e^{f_j}$.

Experimental study

In this section, we present the details of the experimental study conducted to validate the proposed EvoGD approach. First, we introduce the dataset used in the experiment and present the design results. Then, we delve deeper into the key components of EvoGD: Evaluator and Generator, respectively.

Dataset

The dataset is an essential component for data-driven models to achieve better training and testing performance. Therefore, an adaptive sampling strategy was applied to generate a diverse range of supercritical airfoil shapes, ensuring the diversity of the dataset. To ensure its practicality, constraints such as a leading-edge radius of less than 0.007 and a thickness ranging from 0.113 to 0.123 were imposed. In this paper, the dataset consists of 501 supercritical airfoils in total, including one baseline plus 500 randomly perturbed variants. The aerodynamic performance of each airfoil was calculated using high-fidelity CFD simulations.

For the CFD simulations, a structured O-grid with a radius of $30c$ was adopted, where $c = 1$ is the chord length of all the supercritical airfoils. The grid sizes along the normal direction to the surface were kept relatively small to take the thickness of the boundary layer into account. After grid independence verification, the adopted grid consisted of 385 points along the circumference of the supercritical airfoil and 193 points along the normal direction. The Reynolds Average Navier–Stokes (RANS) equations with the Shear Stress Transport (SST) turbulence model were used, and the solutions were calculated using the open-source code CFL3D.¹ The MUSCL scheme was used for state-variable interpolation at the cell interfaces, the Roe scheme was used for spatial discretization, and the lower-upper symmetric Gaussian–Seidel method was used for time advancement.

As an overview, the shapes and performances of the supercritical airfoils in the dataset are depicted in Fig. 6.

General design results

With the above dataset, we ran the proposed EvoGD for 100 iterations and obtained a new set of 240 airfoils, where 95% meet all design constraints and 75% (i.e., 180 elite airfoils) successfully achieved the basic design goals as shown in Table 3. Notably, it only took less than a hour on a PC with i7 CPU to complete the entire design process, including the training of the Generator and Evaluator.

As summarized in in Fig. 7, generally, our EvoGD managed to improve the buffet lift coefficient C_L by at most 9%,

¹ <https://github.com/nasa/CFL3D>.

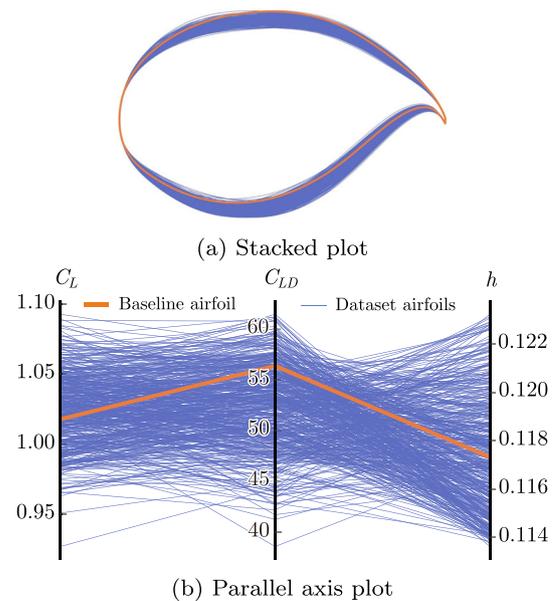


Fig. 6 The stacked plot of the baseline and dataset supercritical airfoils (a), and the parallel axis plot of the aerodynamic and geometric performances (C_L , C_{LD} and h) of the baseline and dataset supercritical airfoils (b). In the parallel axis plot, each vertical axis represents a performance of supercritical airfoils. The orange and blue lines in each plot represent the baseline supercritical airfoil and its 500 randomly perturbed variants respectively

Table 3 The basic design goals of the supercritical airfoil optimization design problem

	C_L	C_{LD}	h
Improvement goals	2%	1%	-2%

The buffet lift coefficient C_L , the cruise lift-to-drag ratio C_{LD} shall be improved by at least 2% and 1%, respectively; and the thickness h shall not be reduced more than 2%

the cruise lift-to-drag ratio C_{LD} by at most 9%, and the thickness h by at most 4%, as shown in Fig. 7. Out of the 180 elite airfoils as obtained, there are 120 that fully dominate the baseline airfoil in all optimization objectives. It can also be seen from the Fig. 7 that the distributions of improved C_L , C_{LD} , and h all have mean values that are significantly greater than 0. Furthermore, on average, C_L and C_{LD} were significantly improved by 5–6%, much more than the design goals of them. From the distribution of improved h in Fig. 7, it can be implied that our approach did not compromise geometric performance for better aerodynamics. Besides, from the parallel axis plot in Fig. 8b, one can also notice such implication. Moreover, the diversity of the generated supercritical airfoils was well preserved, as demonstrated by the stacked plot in Fig. 8a. By contrast, none of the 500 randomly perturbed airfoils in the dataset can achieve the results stated above.

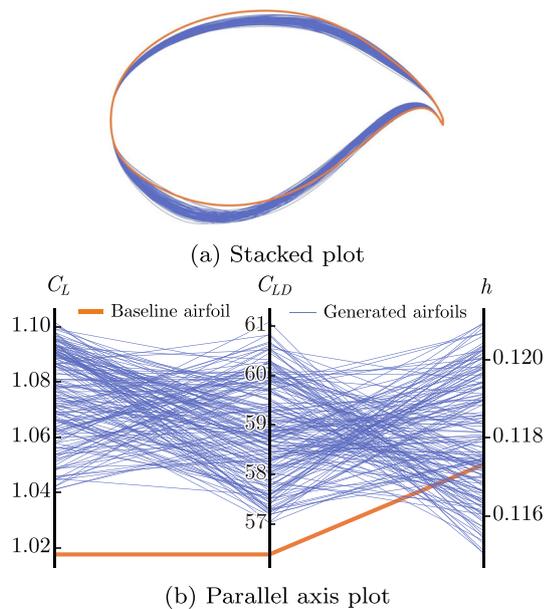


Fig. 8 The stacked plot of the baseline and the generated elite supercritical airfoils (a) and the parallel axis plot of the aerodynamic and geometric performances (C_L , C_{LD} and h) of the baseline and the generated elite supercritical airfoils (b). Each vertical axis in the parallel axis plot represents a supercritical airfoil’s performance. The orange and blue lines represent the baseline supercritical airfoil and the ones generated by EvoGD, respectively

Assessment of generator

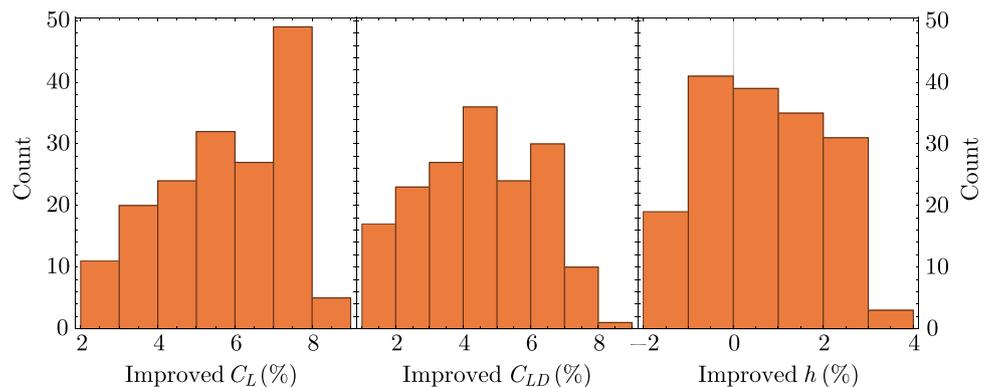
In this subsection, we will conduct the experiment for assessing the performance of Generator in the proposed EvoGD approach, including model training and model testing.

Model training

With the fully dataset consisting of 501 supercritical airfoils

$$\mathcal{Q} = \{q^1, \dots, q^{501}\}, \tag{24}$$

Fig. 7 The histograms of the improvements on aerodynamic and geometric performances (C_L , C_{LD} and h) of the generated elite supercritical airfoils compared to the baseline supercritical airfoil



we adopt distinct separation criteria with different thresholds to separate the original dataset into 20 pairs of training datasets to train four different types of generators. Specifically, as shown in Eq. (14), the tunable separation thresholds $C_{L,\epsilon}$, $C_{LD,\epsilon}$, and h_ϵ are randomly selected to ensure diversity. Each separation thresholds are five samples from distribution $\mathcal{N}(m_t, \sigma_t/10)$, where t represents each optimization objective and m_t and σ_t are the median and standard deviation of t in the original dataset, respectively. As for the separation of all objectives C_L , C_{LD} , and h , it is performed using Pareto dominance sorting. The first Pareto optimal set of the original supercritical airfoils is defined as

$$\mathcal{Q}^1 = \{q^i, q^j \mid \nexists (C_L^i > C_L^j \wedge C_{LD}^i > C_{LD}^j \wedge h^i > h^j)\}. \tag{25}$$

Then, the set \mathcal{Q}^1 is removed from \mathcal{Q} to find the next Pareto optimal set \mathcal{Q}^2 . This process is repeated until $|\mathcal{Q}^1 \cup \dots \cup \mathcal{Q}^r| > 501/2$. The median group \mathcal{Q}^r is then divided into 5 different datasets by random choices to form the datasets for the last type of generators.

To increase diversity in generating supercritical airfoil search directions and steps, our proposed Generator includes 20 generators (from 20 Cycle GANs), each trained on a different pair of datasets. During the training for each generator, the λ in Eq. (12) is set to 10, the batch size is fixed to 64 and it is stopped when training time exceeds 100s. Moreover, the sum of all losses in Eq. (10),

$$\mathcal{L} = \lambda(\mathcal{L}_{GF} + \mathcal{L}_{FG}) + \mathcal{L}_{DG} + \mathcal{L}_{DF}, \tag{26}$$

is used for model preservation: \mathcal{L} is calculated for each epoch and the epoch with lowest \mathcal{L} is selected as the final generator.

Model testing

After applying all the generators to the design variables of the original supercritical airfoils, their modifications (i.e., search steps) can be partially characterized by their marginal

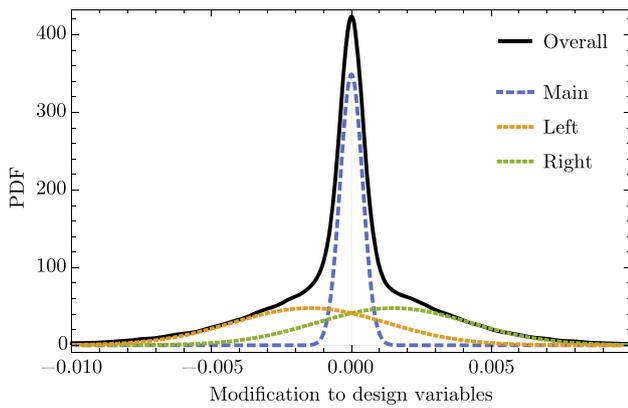


Fig. 9 The probability density function (PDF) of all generators’ modifications to the original supercritical airfoil design variables. The solid line represents the overall PDF, while the horizontal axis represents the search steps of the generators and the vertical axis represents the probability densities of the corresponding search steps. The dashed lines are the Gaussian mixture model fittings of three Gaussians with different means and variances. It is notable that the areas of these components are equal, implying that the generators have modeled both small and large search steps in a balanced manner

probability density function (PDF), as shown in Fig. 9. The decomposition of this PDF into Gaussian mixture models is also depicted. It is noticeable that our generators provide both large and small search steps: since the range of the original supercritical airfoil design variables is only 0.02, the mean values of the left or right Gaussian mixtures (around 0.002) correspond to an average search step size of around 10%, while the main Gaussian mixture corresponds to an average step size of around 1.5%.

Since the PDFs shown in Fig. 9 are only marginal PDFs that flattens all the modifications of different supercritical airfoil design variables, it is necessary to view the covariance matrices of such modifications to verify that our generators do models the performance improvement mapping Eq. (9). For example, the covariance matrices of the C_{LD} generators’ modifications to original supercritical airfoils are shown in Fig. 10. Their relatively stable mean values μ and covariance matrices Σ imply that the generators have learned certain modification directions from the given datasets. However, it does not imply that our generators are randomly perturbing the design variables (i.e., \mathbf{p}) by $\mathcal{N}(\mu, \Sigma)$. For example, Fig. 11 depicts the side-by-side comparison of the objective improvements between randomly perturbed \mathbf{p} with covariance matrix the same as Fig. 10 and modified \mathbf{p} via C_{LD} generators. It shows that our generators are inherently different from high-dimensional Gaussian distributions.

Moreover, the autocorrelation function (ACF) of our generators’ modifications is distinct from that obtained from randomly perturbing design variables, as shown in Fig. 12. Specifically, the ACF we adopt is calculated as follows:

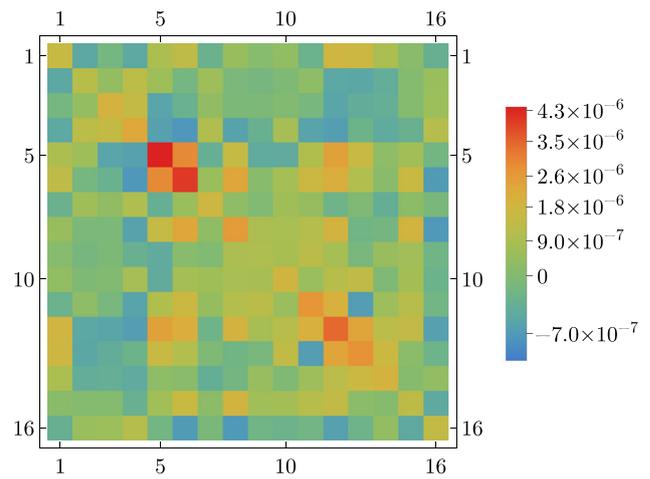
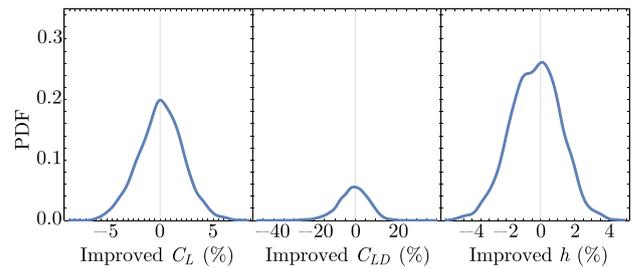
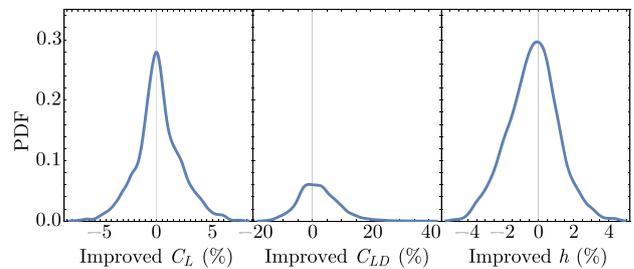


Fig. 10 The plot of the covariance matrix of the C_{LD} generators’ modifications to original supercritical airfoils. The C_{LD} generators are the ones trained with $Q_{2,1}$ and $Q_{2,2}$ in Eq. (14). Hence, it is aimed to improve the C_{LD} objective of the given supercritical airfoils. Since the differences between the covariance matrices of different generators are marginal, they are not shown and the mean value of the matrices are plotted instead



(a) The PDFs of the objective improvements of randomly perturbations



(b) The PDFs of the objective improvements of C_{LD} generators

Fig. 11 A comparison of the probability distribution functions (PDFs) of the objective improvements between randomly perturbed \mathbf{p} following $\mathcal{N}(\mu, \Sigma)$ with the same Σ as in Fig. 10a and modified \mathbf{p} via the C_{LD} generators (b). The comparison reveals that random perturbations are more likely to alter C_L than the generators, which is undesirable since our goal is to improve C_{LD} . Additionally, the random perturbations tend to reduce the thickness more significantly than the generators. Furthermore, the random perturbations’ C_{LD} improvements are evenly distributed around 0, indicating that they are not as effective in improving C_{LD} as the generators

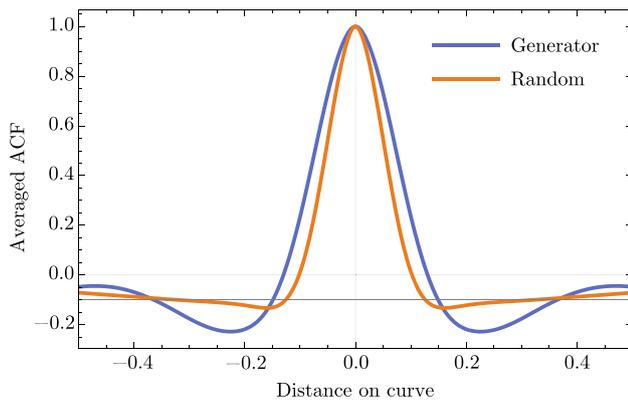


Fig. 12 The average autocorrelation function (ACF) of modifications to all original supercritical airfoils. The blue line represents the average ACF of modifications made by all generators, while the orange line represents the average ACF of random perturbations. The horizontal axis (distance on curve) indicates the distances between the pairs of points on the parametric modification curves being considered. The vertical axis indicates the average correlation between the point pairs of the corresponding distance. It is observed that the ACF of the random perturbations rapidly decreases to near 0, while the ACF of our generators has much longer tails. This difference implies that our generators tend to globally modify the shape of the supercritical airfoils with dedicated correlations

$$ACF[C_y, u_l] = \frac{(C_y(u) - \mathbb{E}[C_y]) * (C_y(u + u_l) - \mathbb{E}[C_y])}{\mathbb{V}[C_y]}, \tag{27}$$

where C_y represents the y coordinate of the modification curve, u is its parameter, and $C_y(u + 1) = C_y(u)$. The operator $*$ represents the convolution with respect to u , $\mathbb{E}[C_y]$ is the expectation of C_y , and $\mathbb{V}[C_y]$ is the variance of C_y . Empirically, the ACF at u_l indicates the average relationship between all point pairs separated by a distance of u_l on C_y : values close to 1 imply that most point pairs are positively related; values close to -1 imply that most are negatively related; while values close to 0 imply that they are on average not related.

Compared to random perturbation of the original supercritical airfoils’ design variables, our generators generate or redesign supercritical airfoils in a more global manner in their geometries, as demonstrated in Fig. 12. In addition, Fig. 13 shows that our generators generally produce supercritical airfoils with better aerodynamic performance compared to the input ones. 43% of the generated supercritical airfoils show dominant performance over the input ones, while 40.5% exhibit trade-off between the two aerodynamic performances. Only 16.5% of the generated supercritical airfoils are dominated by the input ones, with the majority clustered around the origin. This further supports that our generators correctly model the performance improvement mapping as described in Eq. (9).

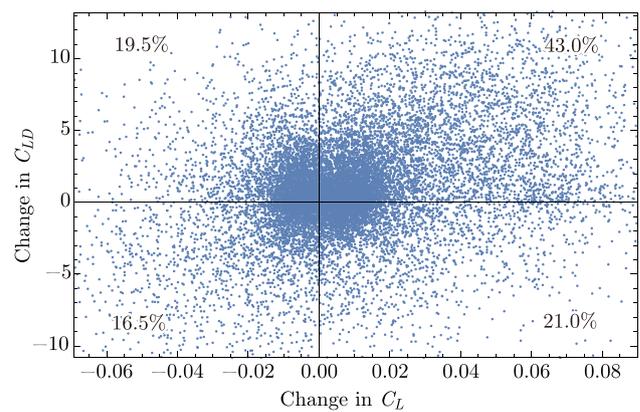


Fig. 13 The improvements in aerodynamic performance of all the generators applied to all supercritical airfoils in the dataset. The four numbers in the figure indicate the percentages of supercritical airfoils in each region. It is evident that our generators can generate supercritical airfoils with superior performance compared to the input ones

In conclusion, our proposed Generator has been shown to be an effective approximation of both local and global search operators in the supercritical airfoil design space. Through training on a dataset, the generators have acquired the ability to learn the natural constraints and generate reasonable modifications to supercritical airfoils by refining them globally with both small and large steps, while maintaining their geometric constraints.

Assessment of evaluator

In this subsection, we will conduct the experiment for assessing the performance of Evaluator in the proposed EvoGD approach, including model training and model testing.

Model training

With the full dataset consisting of 501 supercritical airfoils, the following datasets are used for training the C_L , C_{LD} , α_0 , and C_P predictors, respectively:

$$\begin{aligned} \mathcal{D} &= \left\{ \left(\mathbf{q}^1, C_L^1 \right), \dots, \left(\mathbf{q}^{501}, C_L^{501} \right) \right\}, \\ \mathcal{D} &= \left\{ \left(\mathbf{q}^1, C_{LD}^1 \right), \dots, \left(\mathbf{q}^{501}, C_{LD}^{501} \right) \right\}, \\ \mathcal{D} &= \left\{ \left(\mathbf{q}^1, \alpha_0^1 \right), \dots, \left(\mathbf{q}^{501}, \alpha_0^{501} \right) \right\} \text{ and} \\ \mathcal{D} &= \left\{ \left(\mathbf{q}^1, \alpha^{1,1}, C_P^{1,1} \right), \dots, \left(\mathbf{q}^{501}, \alpha^{501,57}, C_P^{501,57} \right) \right\}. \end{aligned} \tag{28}$$

The superscripts of \mathbf{q} indicate the index of the supercritical airfoil, while the other superscripts indicate the sample index.

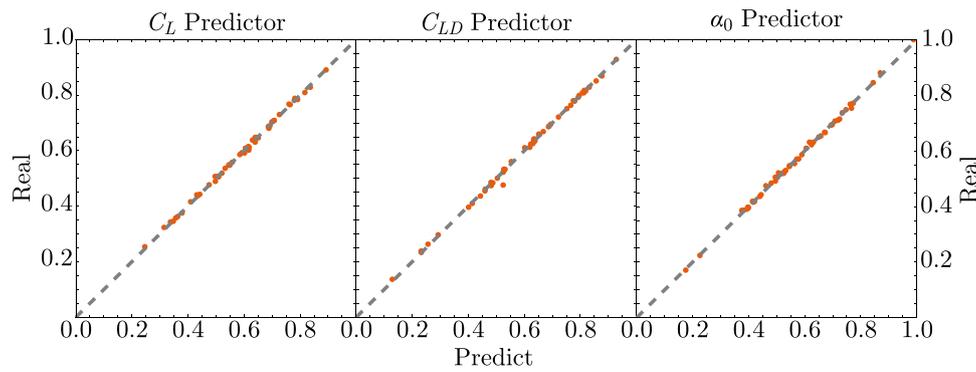


Fig. 14 The actual vs. predicted plots for C_L , C_{LD} , and α_0 predictors in the testing dataset. The orange dots represent the predicted normalized performances and the actual normalized performances, respectively,

For the predictors of C_L , C_{LD} , and α_0 , the aerodynamic performances are first normalized:

$$\tilde{t}^i = \frac{t^i - \min\{t\}}{\max\{t\} - \min\{t\}}, \quad (29)$$

where t is the aerodynamic performance. This normalization ensures that the error $\varepsilon^i = \tilde{t}_{\text{predict}}^i - \tilde{t}_{\text{real}}^i$ is inherently normalized. The datasets are then randomly divided into training sets of size 450 and testing sets of size 51.

For the C_P predictor, the angle of attacks are first normalized with same extreme values of α_0 as above:

$$\tilde{\alpha}^{k,i} = \frac{\alpha^{k,i} - \min\{\alpha_0\}}{\max\{\alpha_0\} - \min\{\alpha_0\}}, \quad (30)$$

which ensures that the predictor models no additional normalization. Then, proper orthogonal decomposition (POD) is applied to $C_P^{k,i}$ through the following procedure: first, all $C_P^{k,i}$ are combined into a matrix

$$\mathbf{M}_{C_P} := \left[C_P^{1,1}, \dots, C_P^{1,i_{\max}^{501}}, C_P^{2,1}, \dots, C_P^{501,i_{\max}^{501}} \right], \quad (31)$$

where $\mathbf{M}_{C_P} \in \mathbb{R}^{353 \times 27,514}$ for our dataset; second, Singular Value Decomposition (SVD) is applied to \mathbf{M}_{C_P} to obtain $\mathbf{M}_{C_P} \rightarrow \mathbf{U}\mathbf{S}\mathbf{V}^T$, where $\mathbf{S} = \text{diag}\{s_1, \dots, s_{353}\}$; finally, the first 30 columns of the left singular vectors \mathbf{U} and the entries of the mode amplitudes \mathbf{S} are retained to achieve the desired decomposition accuracy as shown in Fig. 16. In this way, the dataset is transformed into

$$\begin{aligned} \tilde{\mathcal{D}} = & \left\{ \left\{ \mathbf{q}^1, \tilde{\alpha}^{1,1} \right\} \rightarrow \left\{ s_1^{1,1}, \dots, s_{30}^{1,1} \right\}, \right. \\ & \left\{ \mathbf{q}^1, \tilde{\alpha}^{1,2} \right\} \rightarrow \left\{ s_1^{1,2}, \dots, s_{30}^{1,2} \right\}, \dots, \\ & \left. \left\{ \mathbf{q}^{501}, \tilde{\alpha}^{501,i_{\max}^{501}} \right\} \rightarrow \left\{ s_1^{501,i_{\max}^{501}}, \dots, s_{30}^{501,i_{\max}^{501}} \right\} \right\}. \end{aligned} \quad (32)$$

with the horizontal positions indicating the predicted values and the vertical positions indicating the actual values. The dashed lines represent the ideal predictions

The training and testing sets for the C_P predictor are randomly divided into 90% and 10%, respectively, prior to training the model.

In the training process of each predictor, the batch size is set to the maximum possible value (24,762 for the C_P predictor and 450 for the others) and training is stopped if it exceeds 200 s. The loss function in Eq. (22) is used to evaluate the performance of each predictor, and the epoch with the lowest loss on the testing set is selected as the final predictor.

Model testing

For the C_L , C_{LD} and α_0 predictors, the actual vs. predicted plots of the testing set are shown in Fig. 14. In addition, the prediction errors on the whole datasets are depicted in Fig. 15.

Overall, our predictive models for C_L , C_{LD} , and α_0 show high accuracy, with normalized mean absolute errors of 0.44%, 0.45%, and 0.47%, respectively. The probability density functions of the prediction errors are all close to each other, and the probabilities of predicting absolute normalized errors beyond 1% are small. Null hypothesis tests of $H_0 : \mathbb{P}(\varepsilon) \approx \mathcal{N}(\mu, \sigma)$ can all be rejected at a confidence level of 95%, indicating that our predictive models are unbiased in this confidence level (Fig. 15).

The C_P predictor achieved a prediction error of only 0.33% for mode amplitudes s on the testing dataset. The PDF of the prediction error is not significantly degraded compared to the POD result, as shown in Fig. 16

In conclusion, our proposed Predictor has demonstrated its ability to efficiently and accurately evaluate aerodynamic performance for both optimization objectives and constraints, even with a limited dataset of 501 supercritical airfoils. The results show great potential for improvement, and these models could be further enhanced with a larger dataset.

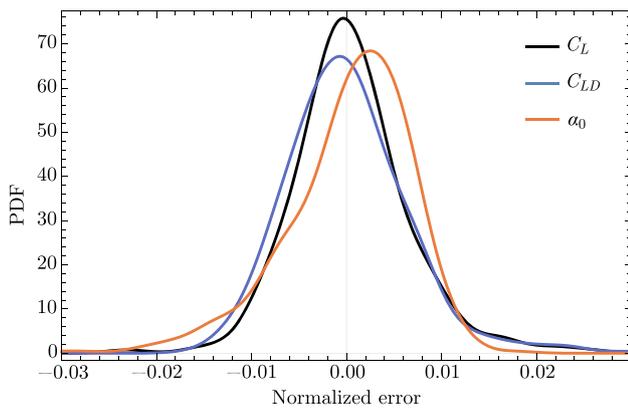


Fig. 15 The normalized error probability density functions (PDFs) for the C_L , C_{LD} and α_0 predictors on combined dataset (training and testing). Due to the limited amount of original supercritical airfoils, no meaningful figure can be made if we only plot the PDF of normalized error on testing dataset. Generally, the predictors are of high accuracy, with maximum relative errors less than 3%. The errors are evenly distributed around zero, indicating that the predictors are unbiased

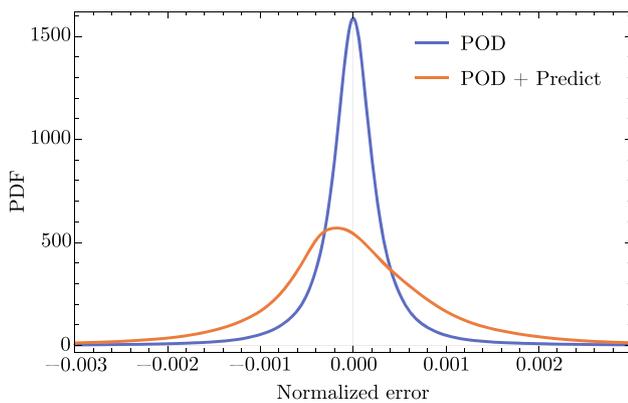


Fig. 16 The normalized error probability density functions (PDFs) of proper orthogonal decomposition (POD) alone and POD with C_P predictor on combined dataset (training and testing). Due to the limited amount of original supercritical airfoils, no meaningful figure can be made if we only plot the PDF of normalized error on testing dataset. It can be noticed that the POD accuracy is high: its maximum relative error is less than 0.3%. Moreover, the prediction error is quite similar to the POD error, proving that our C_P predictor is accurate enough

CFD assessment

As the evaluations previously discussed were all performed using the provided dataset and Multi-Layer Perceptron (MLP) evaluators, the Computational Fluid Dynamics (CFD) results for the initial batch of generated supercritical airfoils were calculated to confirm that we have attained the improvement goals as detailed in Table 3. These results are presented in Table 4. Due to the high computational burden of CFD, only the first 28 supercritical airfoils were processed.

Table 4 The CFD results for the initial 28 generated supercritical airfoils. The zeroth row represents the CFD outcome for the baseline supercritical airfoil. It can be seen that over half (16 out of 28) of the generated supercritical airfoils achieved the design goal, while none of the original dataset did

Airfoil #	C_{LD}	C_L	h	Meet the design goal
0	56.3778	1.01759	0.117488	N/A
1	58.1864	1.09286	0.115910	True
2	51.5491	1.07917	0.118859	False
3	55.9990	1.07422	0.119807	False
4	53.8398	1.06056	0.117586	False
5	58.8495	1.06240	0.116724	True
6	55.5598	1.08737	0.117906	False
7	59.9797	1.09027	0.117423	True
8	59.5910	1.05622	0.115402	True
9	51.8578	1.09073	0.118985	False
10	52.0998	1.09003	0.118844	False
11	58.6832	1.08712	0.118026	True
12	58.1554	1.08654	0.117480	True
13	56.0767	1.08007	0.117301	False
14	53.6372	1.08824	0.117469	False
15	58.6273	1.08447	0.118579	True
16	54.0305	1.09387	0.117754	False
17	50.4746	1.05661	0.119389	False
18	56.0766	1.08744	0.117307	False
19	61.8557	1.04844	0.115494	True
20	59.2281	1.08529	0.118956	True
21	60.5269	1.08306	0.116861	True
22	59.9101	1.08788	0.118540	True
23	46.5633	1.09144	0.117627	False
24	60.1678	1.05575	0.118106	True
25	60.4934	1.08304	0.116006	True
26	59.8208	1.04798	0.118078	True
27	58.5782	1.08883	0.116787	True
28	58.5473	1.07518	0.116605	True

Conclusion

In this work, we proposed a highly efficient and accurate data-driven evolutionary generative design (EvoGD) approach based on AI for supercritical airfoil design. The approach consisted of an evolutionary computational framework combined with data-driven generators and predictors. The results of this approach were highly impressive, producing 180 diverse supercritical airfoils within an hour on a personal computer while improving the design optimization objectives C_L , C_{LD} , and h by 2–9%, 1–9%, and up to 4% compared to the baseline, respectively.

The study on supercritical airfoil parametrization methods showed that the NURBS curve was a balanced choice and was especially suitable for the proposed AI-assisted design approach. The data-driven generators, consisting of a series of Cycle GANs incorporated with physics information, were able to refine the given airfoils towards better-performing ones while maintaining the geometric constraints. Moreover, the data-driven predictors were able to accurately estimate the aerodynamic performances of the airfoils, reducing the need for computationally expensive CFD simulations.

However, there are a few limitations associated with the proposed EvoGD approach. First, to bolster the Generator's performance, a suite of generative models is trained during the process, which inevitably contributes to a significant startup time for the algorithm. Second, according to the CFD result, the current C_{LD} MLP model does not extrapolate effectively, as it fails to accurately predict some generated C_{LD} values. Lastly, the proposed EvoGD approach has not been tested on other problems, and hence, further adjustments may be required when comprehensive experiments are conducted.

Nonetheless, the proposed EvoGD approach was purely data driven and could work with a small dataset, making it highly efficient and effective. This approach has great potential to be generalized to other industrial design processes, reducing costs and increasing the diversity and performance of the design outcomes.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Wang YY, Zhang BQ, Chen YC (2011) Robust airfoil optimization based on improved particle swarm optimization method. *Appl Math Mech (English Edn)* 32:1245–1254
2. Liu Z, Liu X, Cai X (2018) A new hybrid aerodynamic optimization framework based on differential evolution and invasive weed optimization. *Chin J Aeronaut* 31(7):1437–1448
3. Xu Z, Saleh JH, Yang V (2019) Optimization of supercritical airfoil design with buffet effect. *AIAA J* 57(10):4343–4353
4. Bouhleb MA, He S, Martins JRRA (2020) Scalable gradient-enhanced artificial neural networks for airfoil shape design in the subsonic and transonic regimes. *Struct Multidiscip Optim* 61:1363–1376
5. Li R, Zhang Y, Chen H (2021) Learning the aerodynamic design of supercritical airfoils through deep reinforcement learning. *AIAA J* 59:32
6. Liu X, Wei F, Zhang G (2022) Uncertainty optimization design of airfoil based on adaptive point adding strategy. *Aerosp Sci Technol* 130:107875
7. Lei R, Bai J, Wang H, Zhou B, Zhang M (2021) Deep learning based multistage method for inverse design of supercritical airfoil. *Aerosp Sci Technol* 119:107101
8. Wang J, Li R, He C, Chen H, Cheng R, Zhai C, Zhang M (2022) An inverse design method for supercritical airfoil based on conditional generative models. *Chin J Aeronaut* 35:62–74
9. Agoston EE, Smith JE (2015) Introduction to evolutionary computing. Springer, New York
10. Jin Y, Wang H, Chugh T, Guo D, Miettinen K (2019) Data-driven evolutionary optimization: an overview and case studies. *IEEE Trans Evol Comput* 23(3):442–458
11. Doherty JJ, Wang H, Jin Y (2018) Use of global drag rise boundaries to investigate ill-posed transonic airfoil optimization. In: 2018 Applied aerodynamics conference, p 3953
12. Chen W, Chiu K, Fuge MD (2020) Airfoil design parameterization and optimization using bézier generative adversarial networks. *AIAA J* 58(11):4723–4735
13. Phiboon T, Khankwa K, Petcharat N, Phoksombat N, Kanazaki M, Kishi Y, Bureerat S, Ariyarat A (2021) Experiment and computation multi-fidelity multi-objective airfoil design optimization of fixed-wing UAV. *J Mech Sci Technol* 35(9):4065–4072
14. He C, Huang S, Cheng R, Tan KC, Jin Y (2021) Evolutionary multi-objective optimization driven by generative adversarial networks (GANs). *IEEE Trans Cybern* 51(6):3129–3142
15. Yilmaz E, German B (2017) A convolutional neural network approach to training predictors for airfoil performance. In: 18th AIAA/ISSMO multidisciplinary analysis and optimization conference, p 3660
16. Hui X, Bai J, Wang H, Zhang Y (2020) Fast pressure distribution prediction of airfoils using deep learning. *Aerosp Sci Technol* 105:105949
17. Sekar V, Zhang M, Shu C, Khoo BC (2019) Inverse design of airfoil using a deep convolutional neural network. *AIAA J* 57(3):993–1003
18. Chen W, Chiu K, Fuge M (2019) Aerodynamic design optimization and shape exploration using generative adversarial networks. In: AIAA Scitech 2019 forum, p 2351
19. Yilmaz E, German B (2020) Conditional generative adversarial network framework for airfoil inverse design. In: AIAA aviation 2020 forum, p 3185
20. Wu H, Liu X, An W, Chen S, Lyu H (2020) A deep learning approach for efficiently and accurately evaluating the flow field of supercritical airfoils. *Comp Fluids* 198:104393
21. Kulfan B, Bussoletti J (2006) "Fundamental" parameteric geometry representations for aircraft component shapes. In: 11th AIAA/ISSMO multidisciplinary analysis and optimization conference, p 6948
22. Sobieczky H (1999) Parametric airfoils and wings. Recent development of aerodynamic design methodologies: inverse design and optimization. Vieweg Teubner Verlag, Wiesbaden, pp 71–78

23. Piegl L, Tiller W (1996) *The NURBS book*. Springer Science & Business Media, Berlin
24. Toal DJJ, Bressloff NW, Keane AJ, Holden CME (2010) Geometric filtration using proper orthogonal decomposition for aerodynamic design optimization. *AIAA J* 48(5):916–928
25. Deb K, Pratap A, Agarwal S, Meyarivan TAMT (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput* 6(2):182–197
26. Kingma DP, Welling M (2014) Auto-encoding variational Bayes. In: *Proceedings of second international conference on learning representations*
27. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. *Adv Neural Inf Process Syst* pages 2:2672–2680
28. Mirza M, Osindero S (2014) Conditional generative adversarial nets. *arXiv Preprint*
29. Huang S, He C, Cheng R (2022) Sologan: multi-domain multi-modal unpaired image-to-image translation via a single generative adversarial network. *IEEE Trans Artif Intell* 3(5):722–737
30. Zhu J-Y, Park T, Isola P, Efros AA (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. In: *Proceedings of the IEEE international conference on computer vision*, pp 2223–2232
31. Haykin S (1994) *Neural networks: a comprehensive foundation*. Prentice Hall PTR, Upper Saddle River

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.