Automated Parameterization for Aerodynamic Shape Optimization via Deep Geometric Learning

Zhen Wei *
EPFL, Lausanne, Vaud, 1015, Switzerland

Aobo Yang †
The Hong Kong University of Science and Technology (HKUST), Clear Water Bay, Hong Kong SAR, China

Jichao Li ‡
*Institute of High Performance Computing, A*STAR, 138632 Singapore, Republic of Singapore*

Michaël Bauerheim §
ISAE-SUPAERO, Toulouse, Occitania, 31055, France

Rhea P. Liem ¶
Imperial College London, London, SW7 2AZ, United Kindom

Pascal Fua ‖
EPFL, Lausanne, Vaud, 1015, Switzerland

Traditional aerodynamic shape optimization (ASO) is costly and time-intensive, requiring extensive manual tuning for parameterization to define the design space and manipulate geometries. Conventional parameterization methods limit flexibility and demand specialized knowledge, making efficient ASO challenging. We introduce the Deep Geometric Mapping (DeepGeo) model, a neural network-based approach that automates parameterization, accelerating ASO and reducing costs. By leveraging the expressive capacity of neural networks, DeepGeo handles large shape deformation while preserving global surface smoothness, allowing efficient optimization in high-dimensional design spaces. Eliminating the need for training datasets and manual hyperparameter tuning, and integrating Computational Fluid Dynamics (CFD) mesh deformation, DeepGeo significantly lowers ASO's implementation complexity and cost. Case studies including 2D circle, Common Research Model wing, and Blended-Wing-Body aircraft optimization, validate DeepGeo's effectiveness, achieving results comparable to state-of-the-art free-form deformation with minimal manual intervention. This work positions DeepGeo as

*Doctoral Student, Computer Vision Lab, School of Computer and Communication Sciences, Student Member AIAA
†Doctoral Student, Department of Mechanical and Aerospace Engineering, Student Member AIAA
‡Scientist, Fluid Dynamics Department, Member AIAA
§Associate Professor, Department of Aerodynamics, Energies and Propulsion
¶Reader in Sustainable Aviation, Department of Aeronautics (Imperial College London); also Adjunct Associate Professor, Department of Mechanical and Aerospace Engineering (HKUST). Senior Member AIAA
‖Professor, Computer Vision Lab, School of Computer and Communication Sciences (corresponding author, email: pascal.fua@epfl.ch)
This work is an extension of the AIAA's conference paper: Zhen Wei, Aobo Yang, Jichao Li, Michaël Bauerheim, Rhea P. Liem and Pascal Fua. "DeepGeo: Deep Geometric Mapping for Automated and Effective Parameterization in Aerodynamic Shape Optimization," AIAA 2024-3839. AIAA AVIATION FORUM AND ASCEND 2024. July 2024.

a novel and effective parameterization framework that streamlines the workflow and reduces implementation complexity in advanced ASO.

Nomenclature

α	=	the angle of attack
$C_D,\ C_L,\ C_M$	=	drag, lift and pitching moment coefficients
$\mathfrak{C}$	=	the subset of DeepGeo's input coordinate space that defines the CFD computational domain
$f_\Theta,\ \Theta$	=	the Deep Geometric Mapping model and its weights
g	=	the adjoint solver
L	=	the loss function and the soft constraint
M, V, E	=	the CFD mesh and its vertices, edges
O	=	the optimization objective
P	=	FFD control points
S	=	the initial object geometry for optimization
t, A, Vol	=	the geometry's thickness, area and volume
$V^S,\ V^F,\ V^V$	=	CFD mesh vertices on the objects surface, fixed patches and in the volumetric grids

I. Introduction

Aerodynamic Shape Optimization (ASO) is a process aimed at refining an object's shape design to enhance its aerodynamic performance, achieving objectives such as reducing fuel consumption, increasing operating efficiency or improving maneuverability, under specified design constraints. ASO holds significant importance across various fields where it facilitates designs for applications like aerospace, automotive, and marine engineering. As shown in Fig. 1(a), ASO typically involves iterative cycles of computational fluid dynamics (CFD) simulations, visualization and expert analysis, shape modeling and modification, and mesh deformation or remeshing. Traditional ASO can be both slow and costly, with shape parameterization and its associated geometric processing being especially labor-intensive.

Shape parameterization is central to ASO. It provides the mathematical framework through which object geometries are represented and manipulated. Its influence on ASO performance is both profound and multifaceted. The choice of parameterization approach defines the optimization boundaries, determining the maximum geometric complexity that can be handled, the allowable range of shape modifications, and the achievable level of surface smoothness. Then a key challenge lies in balancing the dimensionality of the parameter space with optimization effectiveness: a high-dimensional parameter space enables detailed geometric modeling but may hinder convergence, while a lower-dimensional space facilitates faster optimization but can restrict the search within the design space. Furthermore, the parameterization

method directly impacts ASO's implementation costs due to the level of human intervention required: from initial surface modeling and design variable configuration to the management of CFD mesh deformation. Consequently, selecting an effective parameterization approach is critical for achieving cost-effective and efficient ASO.

Given its critical importance, shape parameterization has been the focus of extensive research. Traditional approaches include Hicks-Henne bump functions [1], non-uniform rational B-spline (NURBS) curves, and class-shape function transformation (CST)[2] for two-dimensional applications, while free-form deformation (FFD) [3–5] is preferred for three-dimensional problems. Despite their success, these traditional methods face notable challenges. The performance and computational efficiency of ASO are highly sensitive to how the parameterization is implemented, including the choice of algorithm and hyperparameters [6–8], with no formal methodology to identify optimal configurations. This leads practitioners to rely on trial-and-error approaches, empirical knowledge, or adaptation of methods from similar cases, followed by extensive manual tuning. This reliance on manual processes makes traditional methods impractical for rapid prototyping and iterative design, where efficiency and adaptability are crucial. Moreover, designers must balance trade-offs between deformation freedom, optimization effectiveness, and global surface smoothness. While high-dimensional design variables provide greater design flexibility, they complicate optimization and increase the risk of poor convergence. Ensuring global surface smoothness is also challenging with large deformation freedom, where even minor surface inconsistencies can cause glitches in simulation, disrupting the optimization process. Furthermore, traditional parameterizations solely focus on the surface geometry, necessitating separate processing of volumetric CFD meshes through deformation or re-meshing, which adds substantial implementation complexity. While AI-based parameterization models have emerged as alternatives [9–11], their data-driven nature requires a large-scale training dataset and introduces new challenges on data engineering, particularly in aerospace applications where training data is scarce and computationally expensive to generate. The geometric priors learned by these models are inherently limited by the dataset available, making any newly generated shape an interpolation of existing data rather than a truly novel design. Consequently, AI-based models lack the innovation potential of free-form parameterization methods, which offer greater design space exploration.

To address these limitations, we aim at building an AI-assisted ASO pipeline and introduce the Deep Geometric Mapping (DeepGeo) model, a novel neural network-based approach that unifies shape representation and mesh deformation within a single framework, as shown in Fig. 1(b). DeepGeo uses a Multi-Layer Perceptron (MLP) to parameterize both the surface geometry and the surrounding CFD computational volumetric mesh, the latter being necessary for aerodynamic performance evaluation through CFD simulations. As illustrated in Fig.2(a), DeepGeo's initialization phase involves training the MLP to deform the surface vertices from a template shape to align with the target geometry, the starting configuration for optimization. The surrounding volumetric mesh is simultaneously deformed to ensure consistency with the surface geometry. By accurately reconstructing the target geometry, this initialization process encodes the geometric information within the network weights. Shape optimization is then performed by

applying an adjoint-based method [12] to minimize aerodynamic objective functions and enforce geometric constraints with respect to these weights, as shown in Fig.2(b). The updated weights enable DeepGeo to output a CFD mesh that defines a new geometry, thus manipulating the shape design iteratively. The optimization process continues until convergence is reached. This formulation establishes the network weights as the parameterization for both the surface shape and the volumetric mesh, enabling automated regeneration of the necessary CFD mesh as the geometry evolves with every optimization iteration. By unifying shape parameterization and mesh deformation within a single framework, DeepGeo significantly reduces traditional ASO implementation complexities and streamlines the optimization pipeline, by removing the need for a separate CFD mesh deformation or remeshing module and the associated time-consuming manual hyperparameter tuning. This simplification accelerates the ASO process.

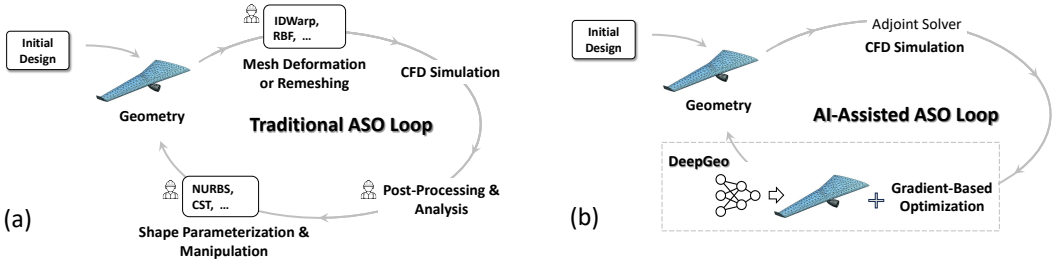


Fig. 1 The pipelines of (a) traditional ASO and (b) ASO that employs DeepGeo.

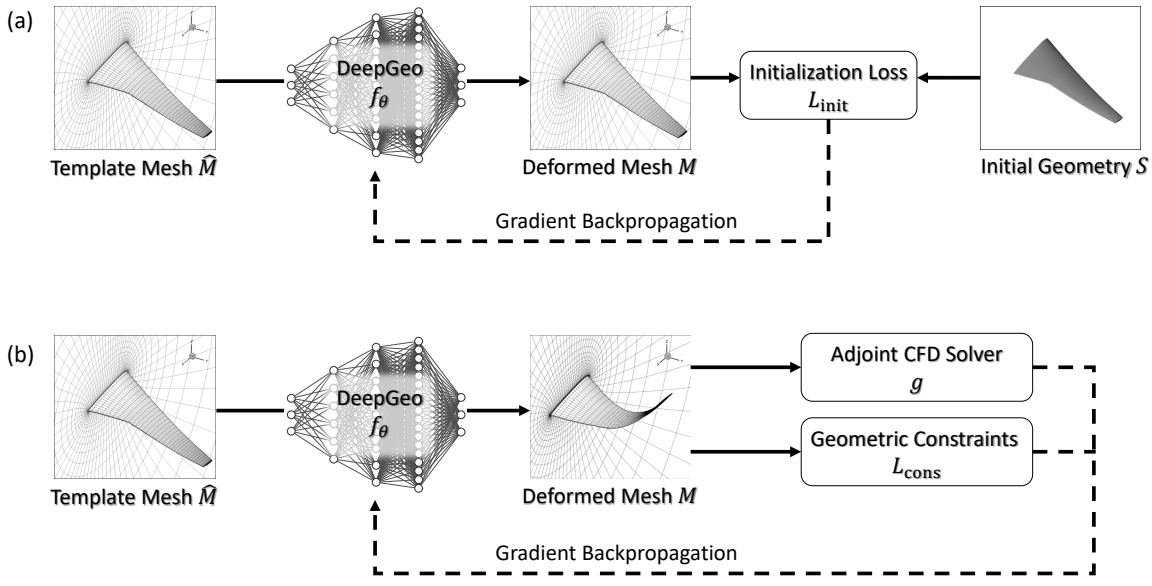


Fig. 2 The pipeline that employs DeepGeo in ASO, including (a) the initialization and (b) shape optimization stages.

DeepGeo leverages the expressive capacity of MLPs, which comes from their ability to represent complex nonlinear mappings as layered compositions of simple functions. Thus, DeepGeo can capture rich geometric transformations

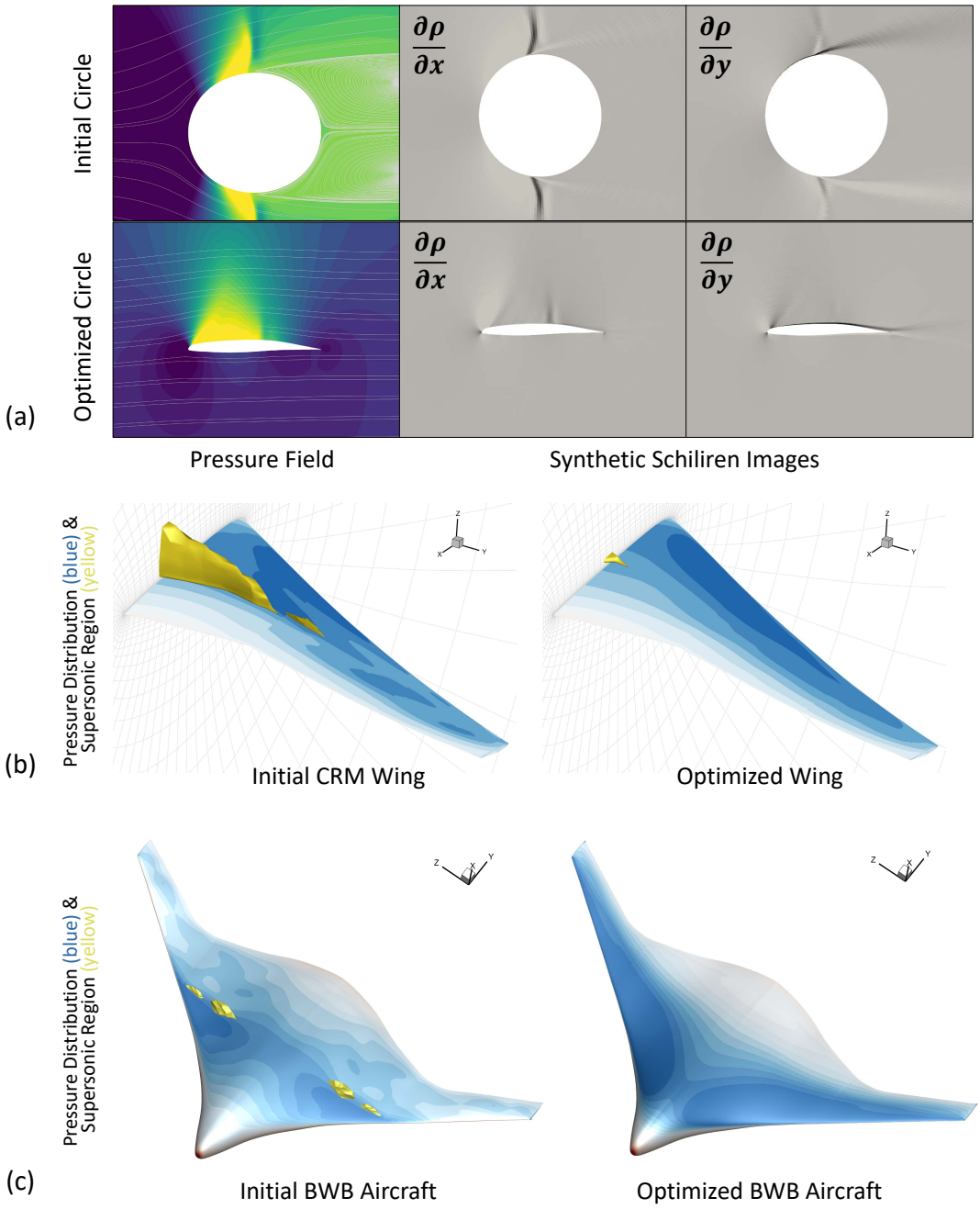


Fig. 3 The three ASO case studies implemented by DeepGeo, including optimization from (a) 2D circle, (b) CRM wing and (c) BWB aircraft.

and handle large deformations, while maintaining global surface smoothness. This expressiveness is key to achieving smooth, continuous transformations across complex geometries, a critical ability in high-fidelity shape optimization. DeepGeo's MLP backbone makes it possible to explore high-dimensional design spaces using a gradient-based approach by mitigating the curse of dimensionality or issues with over-parameterization [13, 14]. DeepGeo directly addresses the

dimensionality challenges that limit traditional optimization methods, such as CMAES [15], which become impractical beyond approximately 100 design parameters. Furthermore, DeepGeo's initialization requires only a target shape and a single template mesh to set its initial weights, in contrast to data-driven approaches that rely on extensive training datasets. This minimal data requirement significantly reduces implementation complexity, decreasing dependency on large datasets and making DeepGeo especially suitable for practical aerospace applications where geometric data is often scarce.

In earlier work [16], we introduced the Direct Mapping Model, an initial version of DeepGeo developed to handle 2D airfoil optimization. In this paper, we extend the model to address more complex optimization tasks and 3D shapes, including 2D circle optimization, 3D wing optimization, and the optimization of a 3D Blended-Wing-Body aircraft. DeepGeo achieves optimization results comparable with the state-of-the-arts, without any case-specific adaptation and requiring training dataset, as shown in Fig. 3.

II. Related Work

DeepGeo spans across multiple research domains, including the shape parameterization algorithms in computer graphics and aerodynamic shape optimization studies, as well as self-prior models in deep learning research. Each part will be briefly reviewed, and the DeepGeo's technical prominence will be discussed afterward.

A. Shape Parameterization in Aerodynamic Shape Optimization

Shape parameterization in ASO refers to the mapping of explicit design variables to the object's surface mesh, a fundamental module for effectively solving ASO problems. Jameson [17] introduced a direct conformal mapping from a circle to optimize airfoils, which provided the maximal design freedom in design space but lacked sufficient smoothness for CFD solvers [18]. Therefore, shape parameterization is necessary to reduce the design space dimensionality and introduce smoothness control. It can be broadly categorized as non-data-driven and data-driven approaches.

1. Non-Data-Driven Approaches

Non-data-driven algorithms include both constructive and deformative methods [6]. Constructive methods, such as polynomial/spline based methods [2, 18, 19] and partial differential equation methods [20, 21], represent the airfoil surface based on specified parameters. Deformative methods deform an existing shape, including analytical methods [1, 22, 23] and the free-form deformation (FFD) [3–5]. Early analytical approaches, like the Hicks-Henne bump function [1] and others [22, 23], optimized the shape through linear combinations of basis functions. B-splines [18] and its variants, like the Bezier curve and nonuniform rational B-spline (NURBS) [19], use control points and piece-wise polynomials to define the geometry. Polynomial-based methods like parameterised sections (PARSEC) [24] and CST [2] approximate surfaces with weighted sums of polynomials with different orders. While effective, these methods are

suited for 2D curves. The FFD, originating from soft object animation in computer graphics [3, 4, 25], enables smooth continuous volume transformations based on control point movements. It is widely used in optimizing 3D objects like the CRM wing [26, 27] and the overall aircraft [28].

When applied to infinite-dimensional problems with finite-dimensional design variables, non-data-driven methods require compromising between the dimensionality for effective parameterization and the optimization complexity [29]. Additionally, the lack of self-adaptation leads to heavy reliance on expert knowledge and manual tuning for hyperparamters. As such, different settings can significantly impact optimization results. To address these limitations, DeepGeo bases on the neural network which provides universal approximation ability [13] and mitigates the curse of dimensionality [13, 14].

2. Data-Driven Approaches

Data-driven methods aim to reduce the dimension of design variables using existing geometry datasets, facilitating effective optimization for challenging problems or extracting patterns to reduce human intervention [30]. Linear dimension reduction methods use proper orthogonal decomposition (POD) to derive a set of orthogonal modes. This can be done by Gram-Schmitt orthogonalization [31] or applying a singular value decomposition (SVD) to find orthogonal shape modes from a dataset [32–34]. However, reconstructing the geometry from a latent vector remains challenging and requires dedicated manual designs for specific tasks. The active subspace model (ASM) [35–39] and active subspace identification (ASI) [40] reduce the dimensionality by analyzing the gradient from surrogate models for surrogate-based optimization or uncertainty quantification. Random sampling methods were used to estimate the real active subspace. However, generating valid samples requires handcrafted rules, and the approximation error is upper bounded by the Poincaré constant, which increases with dimensionality given a limited number of samples [41, 42]. Among the non-linear approaches, Viswanath et al. [43] proposed the generative topographic mapping (GTM) that projects a 30-dimensional design variable into a 2D latent vector. GTM's Bayesian generative model makes it challenging to integrate its latent representation into a gradient-based pipeline. More recently, generative models has been used for novel geometry generation [44–49] to improve the quality of dimension reduction [46, 47] or serve as a parameterization model directly [44, 45].

Data-driven methods typically demand large datasets for training [30, 33, 50]. DeepGeo stands out for its high data-efficiency as it requires only a single initial geometry for training, without requiring any additional user input comparing to the traditional ASO pipeline. This characteristic makes it a feasible solution for complex 3D geometries with limited data availability.

B. Shape Parameterization in Computer Graphics

Shape parameterization in computer graphics aims to create bijective mappings between two surfaces or volumes, with one domain represented as a mesh [51]. This long-standing and active research field has diverse applications,

like in mesh morphing, smoothing, remeshing, and so on. Creating one-to-one mappings between coordinate spaces inevitably introduces distortions that need to be minimized to retain desired mesh properties. Research has focused on surface mesh parameterization using cost functions that include distortion metrics, leading to the development of angle-preserving/conformal mappings, area-preserving/authalic mappings [51, 52], as-rigid-as-possible transformation [53, 54], least squares conformal mapping [55] and Dirichlet energy minimization methods [56–58]. DeepGeo extends the previous research focus to surface and triangulated meshes so that it works with complicated volumetric and unstructured CFD meshes. DeepGeo aligns more closely with the concept of a computer graphics parameterization model, as it establishes a mapping between volumetric CFD meshes instead of generating a design variable vector.

While early ASO shape parameterization methods were inspired by computer graphics, they are not directly applicable to volumetric CFD meshes. DeepGeo's key contribution lies in its regularization loss, which efficiently implements constrained volumetric mesh parameterization. Specifically, DeepGeo's regularization loss is derived from the continuous manifold learning theory, eliminating the need for mesh topology and allowing for flexible and highly sparse sampling for loss calculation, thus avoiding unaffordable time and memory consumption on large-scale CFD meshes. Then, DeepGeo uses a neural network, replacing finite difference and finite element approximations with analytical auto-differentiation. Additionally, it uses adjoint gradient descent optimization, which is more effective and computationally efficient than computer graphics parameterization models. Third, DeepGeo's approach is independent of mesh topology, which avoids complex mesh processing, especially in cases of extreme cell irregularity where the simple triangulation/tetrahedralization assumption do not hold for CFD meshes.

C. Deep Learning Model with Self-Prior

Self-prior is an emerging research topic in the field of deep learning, where deep neural networks are trained to acquire domain-specific prior knowledge from a single data sample. The deep image prior [59] and its follow-ups have shown strong capabilities in multiple low-level vision tasks with self-supervision, such as image super-resolution, denoising, inpainting, dehazing and deblur. In 3D domain, the deep geometric prior [60] introduced MLP models to reconstruct partial geometry of a point cloud, while Point2Mesh [61] proposed to reconstruct the entire surface mesh from a point cloud.

DeepGeo uses a similar learning technique, but it is the first model that demonstrates how the self-prior can be effectively harnessed and manipulated with the mesh representation under the guidance of an external adjoint CFD solver.

III. Method

A. Deep Geometric Mapping Model

Optimizing a shape for performance requires parameterizing it in terms of variables that facilitate effective optimization. The Deep Geometric Mapping (DeepGeo) model achieves this by using its neural network parameters directly for optimization. At the core of DeepGeo is a neural network f_Θ with weights Θ, which serve as the shape's parameterization. To initialize, DeepGeo learns to reconstruct the initial geometry S to be optimized by deforming a template CFD mesh $\hat{M}$, as shown in Fig. 2(a). By adjusting DeepGeo's weights, the model produces corresponding variations in both the shape and its surrounding CFD mesh, resulting in a deformed output mesh M. The accurate reconstruction of S with M means that the geometric information is embedded within DeepGeo's network weights.

During optimization, as shown in Fig. 2(b), the deformed mesh M is sent to an adjoint CFD solver (denoted as g), where the fluid field is simulated and the sensitivity of the aerodynamic objectives to the surface vertices is calculated. Gradients of any geometric constraints are computed simultaneously. DeepGeo then backpropagates these surface sensitivities to the network weights Θ via auto-differentiation, updating the weights using a gradient-based optimization algorithm, such as Adam [62]. With each update to the weights, the deformed surface adjusts accordingly, which is how the shape design is manipulated during optimization. This iterative process of simulation, gradient computation and weight updating continues until the geometry converges to an optimal design. Through this integrated pipeline, DeepGeo effectively automates shape parameterization and optimization.

1. Formalization

Let $\hat{M} = \{\hat{V}, E\}$ represent a template CFD mesh, where $\hat{V} = \{\hat{\mathbf{v}}_1, \hat{\mathbf{v}}_2, ..., \hat{\mathbf{v}}_N\}$ are the vertices and E denotes its edges. The vertices in $\hat{V}$ are grouped into three separate sets: $\hat{V}^S$, the vertices on the object surface; $\hat{V}^F$, the vertices from fixed patches if any; and $\hat{V}^V$, the remaining vertices defining the volumetric computational cells. Let $S = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_N\}$ be the vertices defining the surface of the initial geometry to be optimized, and S is matched with $\hat{V}^S$.

We implement DeepGeo as a feed-forward network f_Θ that acts as a continuous mapping between coordinates spaces, defined as:

$$f_\Theta : \mathbb{R}^3 \rightarrow \mathbb{R}^3 , \quad \delta\mathbf{v} = f_\Theta(\hat{\mathbf{v}}) , \tag{1}$$

where $\delta\mathbf{v}$ is the translation of the input vertex. For brevity, we denote $F_\Theta(\hat{V}) \in \mathbb{R}^{N\times 3}$ as the translation matrix of all N vertices. The resulting deformed mesh is represented as $M = \{\hat{V} + F_\Theta(\hat{V}), E\}$.

DeepGeo initializes and embeds the geometric information of the initial shape S into Θ by finding

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}}\, \mathcal{L}_{\text{init}}(\Theta, F_\Theta(\hat{V}^S), F_\Theta(\hat{V}^V), S) , \tag{2}$$

where $\mathcal{L}_{\text{init}}$ is a loss function that should be minimized when the surface vertices of the deformed template mesh $V^S = \hat{V}^S + F_\Theta(\hat{V}^S)$ accurately reconstruct the target geometry S, while preserving mesh quality and keeping fixed patches not moved. The initialization loss $\mathcal{L}_{\text{init}}$ is defined as:

$$\begin{aligned} \mathcal{L}_{\text{init}} &= \mathcal{L}_{\text{dst}}(\Theta) + \lambda_{\text{fix}}\mathcal{L}_{\text{fix}}(\Theta) + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}}(\Theta) \ , \\ \text{where } \mathcal{L}_{\text{dst}}(\Theta) &= \frac{1}{N}\|\hat{V}^S + F_\Theta(\hat{V}^S) - S\|_F^2 \ , \\ \mathcal{L}_{\text{fix}}(\Theta) &= \frac{1}{\left|\hat{V}^F\right|}\left\|F_\Theta(\hat{V}^F)\right\|_F^2 \ , \\ \mathcal{L}_{\text{reg}}(\Theta) &= \|\mathbf{H}\left(F_\Theta(\hat{V}^V)\right)\|_F^2 \ . \end{aligned} \tag{3}$$

$\mathcal{L}_{\text{dst}}$, $\mathcal{L}_{\text{fix}}$, and $\mathcal{L}_{\text{reg}}$ represent, the *Distance Loss*, *Fixed Loss*, and *Regularization Loss*, respectively. λ_{fix} and λ_{reg} are balancing weights that control their respective influence. $\mathbf{H}$ represents the Hessian operator of DeepGeo with respect to the input vertices.

The *Distance Loss* $\mathcal{L}_{\text{dst}}$ is the Euclidean distance between V^S and S. The *Fixed Loss* $\mathcal{L}_{\text{fix}}$ constrains movement of fixed patches, such as mesh boundaries or areas specified by the optimization problem. The *Regularization Loss* $\mathcal{L}_{\text{reg}}$ preserves CFD mesh quality for simulation by minimizing non-rigid volumetric deformations, maintaining cell properties like skewness, orthogonality, and aspect ratio.

The *Regularization Loss* $\mathcal{L}_{\text{reg}}$ is inspired by the Hessian-based Locally Linear Embedding (HLLE) model [63] and minimizes the Frobenius norm of the Hessian of deformation over a specified input volumetric space. This approach is based on the property that a vanishing Hessian guarantees locally isometric deformations, preserving the local geometric structure and effectively implementing the deformation as a rigid transformation, thus maintaining mesh quality. In practice, we treat the input template CFD mesh a discretization of an isotropic Euclidean coordinate space, where its tangent space is identical to the input coordinate space itself. DeepGeo's Hessian at an arbitrary position $\mathbf{v}$ can be calculated with the Cartesian coordinate system as:

$$\mathbf{H}\left(\mathbf{v}\right) = \begin{bmatrix} \frac{\partial^2 \mathbf{v}}{\partial x^2} & \frac{\partial^2 \mathbf{v}}{\partial x y} & \frac{\partial^2 \mathbf{v}}{\partial x z} \\ \frac{\partial^2 \mathbf{v}}{\partial x y} & \frac{\partial^2 \mathbf{v}}{\partial y^2} & \frac{\partial^2 \mathbf{v}}{\partial y z} \\ \frac{\partial^2 \mathbf{v}}{\partial x z} & \frac{\partial^2 \mathbf{v}}{\partial y z} & \frac{\partial^2 \mathbf{v}}{\partial z^2} \end{bmatrix} . \tag{4}$$

DeepGeo relies on a loss term that minimizes the quadratic form of Hessian as $\int_{\mathfrak{C}} \|\mathbf{H}(c))\|_F^2 \, dc$, where $\mathfrak{C}$ is the subset of input coordinate space bounded by $\hat{V}^V$ that defines the CFD computational domain. This quadratic term measures DeepGeo's average 'curviness' over $\mathfrak{C}$. To further speed up the computation and due to the inherent generalization ability of neural networks, $\mathcal{L}_{\text{reg}}$ can estimated by evaluating the Hessian only on a finite and small number of vertices sampled from $\hat{V}^V$, enabling efficient computation via auto-differentiation. Minimizing $\mathcal{L}_{\text{reg}}$ directly provides an adjoint

approach for optimization, and is considerably simpler than finding the Hessian's null space and corresponding basis vectors as required in the original HLLE.

Fig. 4 illustrates the effectiveness of the $\mathcal{L}_{\text{reg}}$ regularization. In this example, the template mesh $\hat{M}$ is a RAE-2822 airfoil while the target shape S is the same airfoil rotated by 45 degrees, imitating a large spanwise twist for 3D wing design. Without regularization, the deformed CFD mesh M exhibits severe defects, such as overlapping cells and high non-orthogonality, as shown in Figure 4(b). As the regularization weight λ_{reg} is progressively increased, these issues are resolved and the quality of M closely eventually matches that of $\hat{M}$, as shown in Figure 4(c-d).

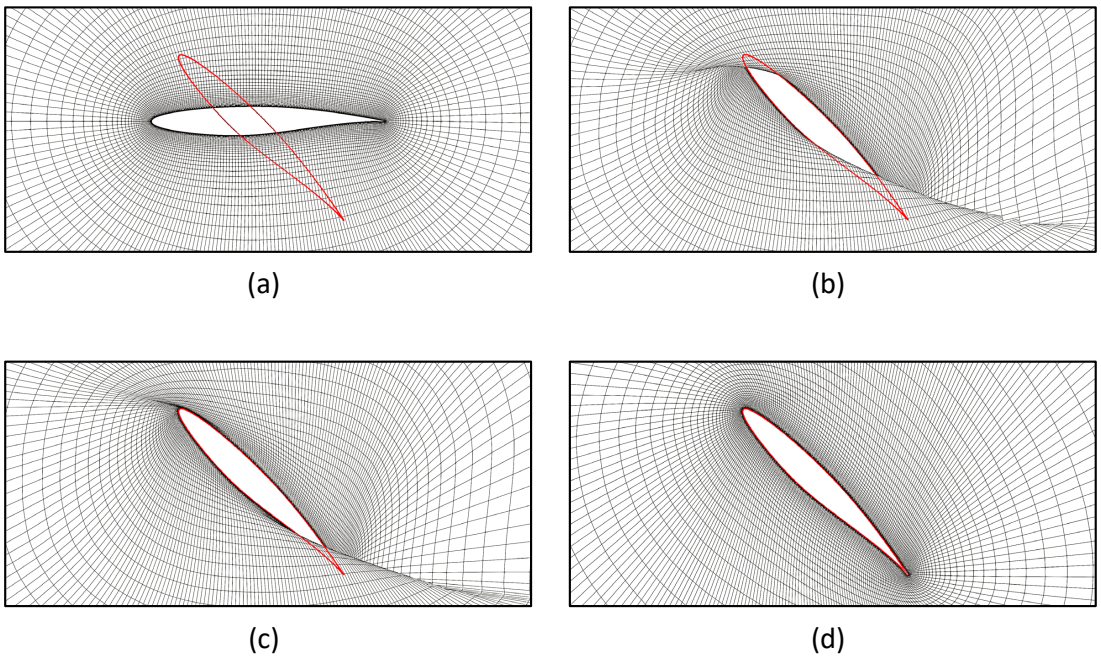


Fig. 4 Investigation of $\mathcal{L}_{\text{reg}}$ by deforming the mesh in black to fit the shape in red, as shown in (a), with (b) λ_{reg}=0, (c) λ_{reg}=10 and (d) λ_{reg}=100.

2. Implementation Details

DeepGeo is implemented as a multi-layer perceptron (MLP) model, with three-dimensional input and output layers. For 2D ASO cases, the z coordinate is simply set to zero. The default configuration of DeepGeo for all case studies includes three hidden layers with dimensions of 64, 256, and 512 from input to output, respectively. Each layer includes a bias terms and uses weight normalization [64], resulting in a total of 151, 686 parameters. While this amount of weights appears large, it does not directly reflect DeepGeo's actual model complexity, which is much smaller than what the raw number of weights might indicate. We provide a detailed discussion in Sec. V.A. The activation function is a sum of ReLU [65] and sine functions, which ensures fast convergence and allows for the computation of higher-order derivatives required by L_{reg}. During DeepGeo's initialization stage, $\lambda_{fix} = 1$ and $\lambda_{reg} = 100$ as in Eq. 4. DeepGeo

employs the same configuration across all cases discussed in Section IV, without any further tuning.

When a computational mesh associated with the target surface S is available, it can be used as the template CFD mesh $\hat{M}$. In this case, the minimization of Eq. 2 reduces to learning zero deformations, a process we name the self-initialization. A straightforward approach might be to initialize all network weights to zero, but this would prevent further optimization since all gradients would also be zero. Instead, we initialize the network randomly and train it to approximate zero deformations. Experience shows that this does not result in a trivial solution and allows further optimization.

B. Adjoint-Based Shape Optimization with Deep Geometric Mapping Model

For all ASO case studies discussed in this paper, the DeepGeo model is coupled with the finite-volume CFD solver ADflow [66] for Reynolds-Averaged Navier-Stokes (RANS) simulation using the Spalart–Allmaras (SA) turbulence model, based on the generated CFD mesh M from DeepGeo. Sensitivities of optimization objectives with respect to the object surface V^S are obtained with ADflow's discrete adjoint solver. Meanwhile, the deformed object surface is evaluated according to the geometric constraints, denoted as L_{cons}. The value of L_{cons} corresponds to the optimization infeasibility. These constraints are implemented as differentiable soft constraints, and their gradients with respect to V^S are calculated via auto-differentiation. The gradients of Θ are then computed by applying the chain rule that begins with the ADflow's surface sensitivities and the gradients of geometric constraints. The RAdam [67] optimizer is used to update Θ.

The overall objective function is formulated as:

$$O = O_{\text{CFD}} + \lambda_{\text{cons}} L_{\text{cons}} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}} + \lambda_{\text{fix}} \mathcal{L}_{\text{fix}} \ , \tag{5}$$

where $\lambda_{fix} = 1$ and $\lambda_{reg} = 0.1$. λ_{cons} is a balancing weight that depends on the number, dimension, and unit of different geometric constraints. The regularization loss $\mathcal{L}_{\text{reg}}$ and the fixed loss $\mathcal{L}_{\text{fix}}$ are as defined in Eq. 3, and the loss weights λ_{reg} and λ_{fix} remain constant. These losses help preserve CFD mesh quality and ensure fixed patches remain unchanged during optimization. O_{CFD} represents physical objectives, such as minimizing drag or maximizing lift. It is defined precisely, along with L_{cons}, in each case study below. ASO then involves finding the solution to:

$$\begin{aligned} \Theta^* &= \underset{\Theta}{\text{argmin}} \left(O\left(F_\Theta(\hat{V}), \Theta \right) \right) \\ &= \underset{\Theta}{\text{argmin}} \left(O_{CFD}\left(\left\{ \hat{V} + F_\Theta(\hat{V}), E \right\} \right) + w_{cons} L_{\text{cons}}\left(\Theta\right) + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}\left(\Theta\right) + \lambda_{\text{fix}} \mathcal{L}_{\text{fix}}\left(\Theta\right) \right) \ , \end{aligned} \tag{6}$$

where $\left\{ \hat{V} + F_\Theta(\hat{V}) \right\}$ denotes the set of vertices displaced by the translations predicted by f_Θ. This iterative process is terminated using an early stopping strategy, a common approach in gradient-descent-based neural network training [68],

which stops the process when O reaches a user-defined threshold or shows no significant improvement after a certain number of iterations.

DeepGeo's configuration remain fixed across different ASO applications, making it a tuning-free and fully automatic parameterization. In the following sections, the effectiveness of DeepGeo-based ASO framework is verified through three different case studies, including optimizing a 2D circle, a 3D business jet wing, and a 3D Blended-Wing-Body aircraft.

IV. Case Studies of Aerodynamic Shape Optimization

In this section, we demonstrate the versatility of our approach in three distinct cases: minimizing drag from a 2D circle, optimizing the NASA's Common Research Model Wing [69], and optimizing a Blended-Wing-Body aircraft [70]. Despite the substantial differences between these cases, we apply the same configuration for all 2D and 3D cases without any fine-tuning. This fixed configuration highlights the ease of using DeepGeo compared to the state-of-the-art FFD-based method, which we use as a baseline for comparison. Unlike FFD, our approach requires no careful configuration of control points across cases, eliminating the need to determine their number and positions, constrain the value ranges of design variables, or develop global design functions. Configuring FFD for different tasks and geometries is time-consuming, demands extensive expertise, and often relies on trial-and-error.

A. Optimization Starting from a Circle

We first study the case of a 2D circle , which we aim to optimize to minimize drag, with the expectation that it will naturally converge toward the shape of a supercritical airfoil. This case involves significant geometric deformations, testing DeepGeo's ability to maintain global shape smoothness while allowing large deformation freedom and ensuring robust mesh deformation under challenging conditions.

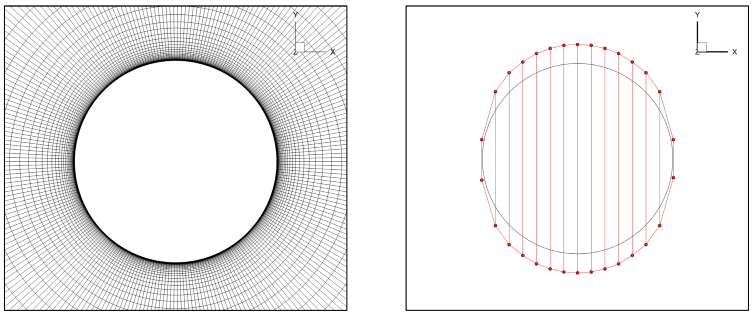
1. Problem Formulation

We begin with a 2D circle of diameter one and apply the same experimental setup as in the ADODG-2 transonic airfoil optimization benchmark [*]. The objective is to minimize the drag coefficient C_D while maintaining the lift coefficient C_L at 0.824 and constraining the pitching moment coefficient to $C_M \geq -0.092$. The optimization is performed in a transonic turbulent flow with a Mach number of 0.734, a Reynolds number of 5×10^6 and an initial angle of attack of 2.0°. We used pyHyp [28] to compute the CFD template mesh shown in Fig. 5(a), which comprises $20,331$ cells.

[*] https://sites.google.com/view/mcgill-computational-aerogroup/adodg

Table 1 Aerodynamic shape optimization task specifications for the Case Study I.

Objectives	Functions/Variables	Description	DeepGeo Quantity	FFD Quantity
Minimize	C_D	Drag coefficient		
With respect to	Θ	Weights of DeepGeo	151 585	
	P_z	Control points' z coordinates		30
	α	Angle of attack	1	1
Total design variables			151 586	31
Subject to	$C_L = 0.824$	Lift coefficient constraint	1	1
	$C_M \leq 0.092$	Moment coefficient constraint	1	1
	$t \geq 0.2\, t_{\text{RAE2822}}$	Minimum thickness constraints		30
	$A \geq 0.0654$	Minimum area constraints	1	1
	$\delta \mathbf{v}^{\text{TE}} = 0$	Trailing edge constraint	1	
	$\delta \mathbf{v}^{\text{LE}} = 0$	Leading edge constraint	1	
	$0.5 \leq \alpha \leq 4.0$	Angle of attack constraint	1	1
Total constraints			6	34
Need value range limits for each DV?			NO	YES



(a) (b)

Fig. 5 Geometric setup in the 2D airfoil case. (a) DeepGeo template mesh $\hat{M}$. (b) Control points for the FFD baseline.

2. Configuring DeepGeo

Following the same optimization strategy proposed in Li and Zhang [47], we perform a two-step optimization. First, the drag coefficient C_D is minimized. Then, terms are added to the objective function to address the lift and moment coefficients, C_L and C_M, and the optimization continues. For this purpose, two versions of the function O_{CFD} from Eq. 5 are defined to represent the physical objectives as:

$$
\begin{aligned}
O_{CFD,1} &= |C_D| \ , \\
O_{CFD,2} &= |C_D| + |C_L - 0.824| + \max\left(-0.092 - C_M, 0\right) \ ,
\end{aligned}
\tag{7}
$$

where the coefficients are obtained with the adjoint solver (denoted as g) as $(C_D, C_L, C_M) = g\left(\{\hat{V} + F_\Theta(\hat{V}), E\}\right)$. We use the Shoelace formula to compute deformed airfoil's area $A(V^S)$. The geometric loss $\mathcal{L}_{cons}$ includes a term that constrains the area from falling below the RAE 2822 profile area (0.0654), along with two additional terms that fix the

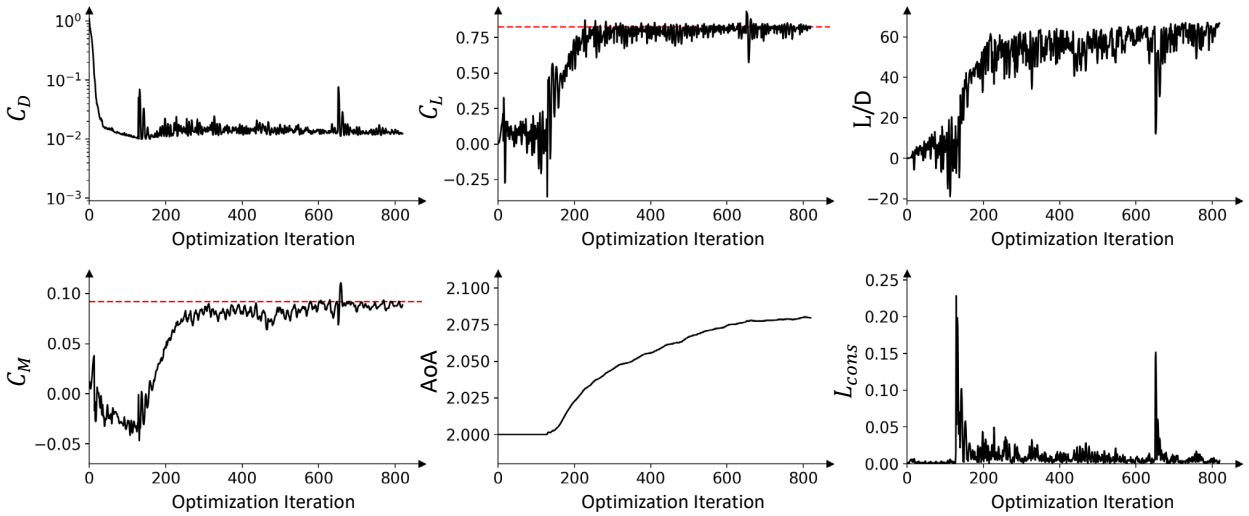


Fig. 6 Evolution of important quantities during optimization for the 2D airfoil case. The dashed lines in the C_L and C_M plots indicate the target values for these quantities, which are closely approached by the end of the optimization, demonstrating the precision and effectiveness of the DeepGeo model in reaching aerodynamic targets.

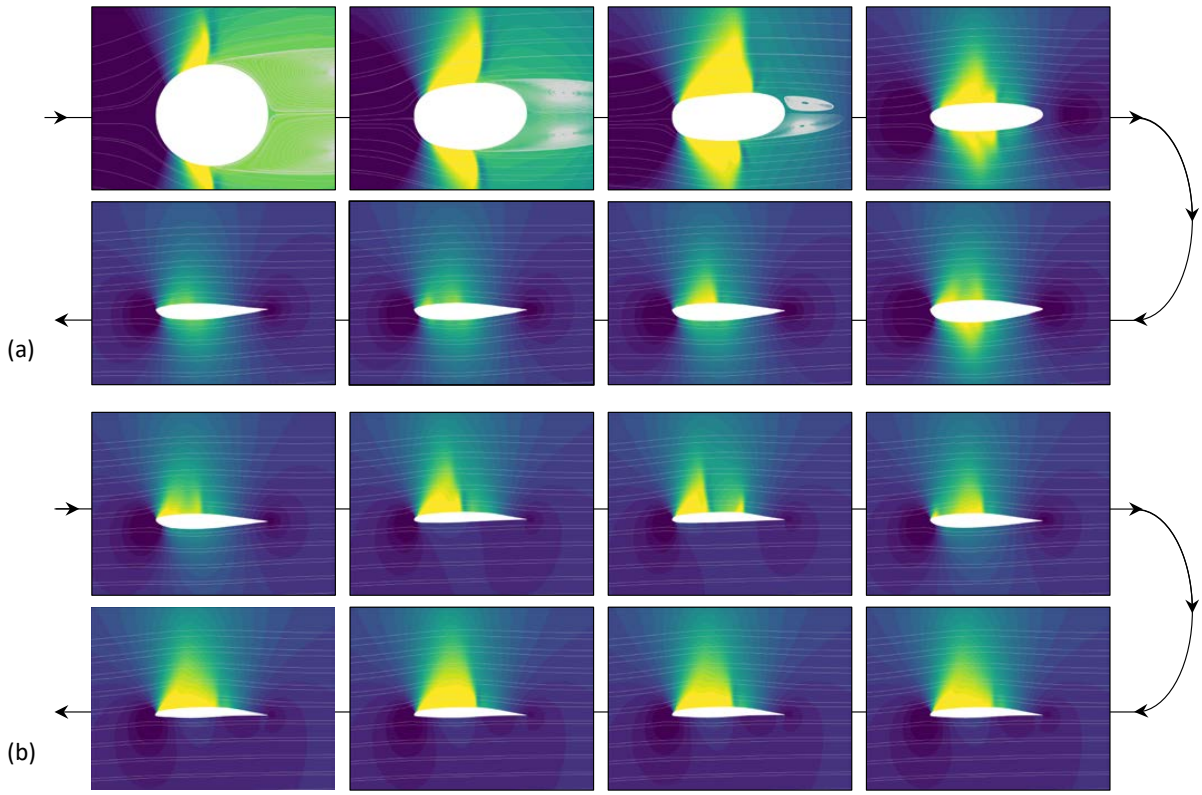


Fig. 7 Evolution of the pressure coefficient field in the 2D airfoil case. (a) Minimizing C_D alone. (b) Minimizing C_D while setting a target for C_L and C_M.

leading and trailing edge vertices (LE and TE), respectively. This is expressed as:

$$\mathcal{L}_{cons} = \underbrace{\max\left(0.0654 - A(V^S), 0\right)^2}_{\text{area constraint}} + \underbrace{\left\|\delta\mathbf{v}^{\text{LE}}\right\|^2}_{\text{LE constraint}} + \underbrace{\left\|\delta\mathbf{v}^{\text{TE}}\right\|^2}_{\text{TE constraint}} . \tag{8}$$

As discussed in Section III.A.2, we use the initial mesh as the template mesh $\hat{M}$. Tab. 1 summarizes our experimental setup.

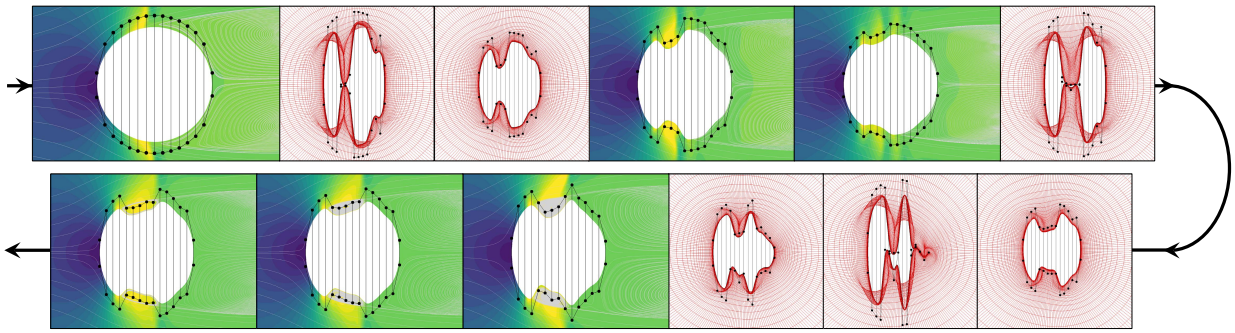


Fig. 8 Failing FFD-based optimization in the 2D airfoil case. The absence of a colored fluid field denotes a simulation that did not converge.

3. Configuring the Free-Form Deformation Baseline

For comparison purposes, we use the implementation of [47] with a 30-point configuration for FFD as shown in Fig. 5(b). Let the airfoil thickness be t, and let t_{RAE2822} represent the thickness of the RAE-2822 airfoil. A set of thickness constraints $t \geq 0.2, t_{\text{RAE2822}}$ is applied to prevent negative volume errors at the beginning of the optimization. The optimization is conducted using the SLSQP optimizer available in pyOptSparse[71].

4. Results and Analysis

The optimization process is illustrated in Fig. 6, while the fluid field changes are visualized in Fig. 7. When minimizing $O_{CFD,1}$, the drag coefficient C_D is reduced significantly by 99.1%. The shape gradually and smoothly becomes thinner, with the leading and trailing edges emerged and symmetry largely preserved. No singular shapes or simulation failures occur, demonstrating the robustness of DeepGeo to large mesh deformations. In the second stage, symmetry is broken to achieve a streamlined profile commonly observed in high-lift transonic airfoils, effectively generating more lift. A shock wave initially appears but is gradually diffused, and by the end of the optimization, it largely dissipates. The final airfoil achieves $C_D = 0.012\,195$ (i.e. 121.95 counts), and its lift-over-drag ratio (L/D) is significantly improved from 1×10^{-3} to 67.22, which makes it comparable to that of modern supercritical airfoils designed for transonic flights. This result validates DeepGeo's effectiveness in achieving high-performance designs.

In contrast, the FFD-based optimization fails after a small number of iterations due to singularities in the deformed shape, leading to severe meshing issues as shown in Fig. 8. In some iterations, simulation failure occurs, represented by the mesh in red. Although FFD ensures local smoothness within a control cage, the free movement of control points can lead to global singularities and geometric abnormalities. These issues persist despite adjustments to the hyperparameters. Such failures do not occur with DeepGeo, highlighting its stability and reliability for complex shape

optimizations that require high deformation freedom.

B. Optimizing the Common Research Model Wing

We now turn to NASA's Common Research Model Wing (CRM wing), which is extracted from the CRM wing-body configuration. As shown in Fig. 9(b), the initial geometry is a 3D wing with a blunt trailing edge. As in the previous case, results are compared to those obtained with the FFD baseline.

1. Problem Formulation

The objective is to minimize the drag coefficient C_D while maintaining a lift coefficient C_L of 0.5 and ensuring that the pitching moment coefficient C_M is at least -0.17. This single-point optimization problem is conducted under fully turbulent flow conditions, with a Mach number of 0.85, a Reynolds number of 5×10^6, and an initial angle of attack of 2.2°. Additional constraints require the wing's volume to remain at least as large as its initial value, and its thickness to be no less than 25% of the initial thickness. The trailing edge (TE) vertices and the leading edge (LE) vertex on the root section are fixed, while the remaining LE vertices are restricted to movement along the z direction, preserving the projected area. The optimization objectives and flow conditions follow the ADODG-4.1 single-point optimization benchmark [†].

The optimization is performed on a grid with 450, 560 cells, as shown in Fig. 5(a), which corresponds to the L2 grid defined in Lyu et al. [26]. In this setup, the DeepGeo weights affect only the wing's surface shape, while the angle of attack is treated as a standalone design variable. Tab. 2 summarizes the experimental setup.

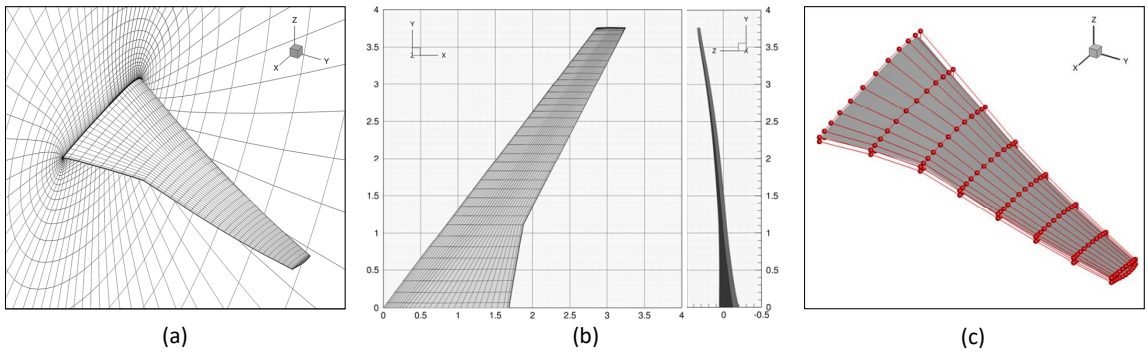


Fig. 9 The geometric setups for Case Study II, including: (a) the template mesh for DeepGeo, (b) the initial CRM wing geometry and (c) the 192-point FFD setting.

[†]https://sites.google.com/view/mcgill-computational-aerogroup/adodg

Table 2 Aerodynamic shape optimization task specifications for the Case Study II.

Objectives	Functions/Variables	Description	DeepGeo Quantity	FFD Quantity
Minimize	C_D	Drag coefficient		
With respect to	Θ	Weights of DeepGeo	151 585	
	P_z	Control points' z coordinates		720
		Twist function		7
	α	Angle of attack	1	1
Total design variables			151 586	728
Subject to	$C_L = 0.5$	Lift coefficient constraint	1	1
	$C_M \geq -0.17$	Moment coefficient constraint	1	1
	$t \geq 0.25\ t_{\text{original}}$	Minimum thickness constraints		750
	$Vol \geq Vol_{\text{original}}$	Minimum volume constraint	1	1
	$\Delta V^{\text{TE}} = 0$	Trailing edge constraint	1	
	$\Delta V_x^{\text{LE}} = 0,\ \Delta V_y^{\text{LE}} = 0$	Leading edge constraint	1	
	$\delta \mathbf{v}^{\text{LE,root}} = 0$	Fixed-wing root incidence constraint	1	
	$\Delta P_z^{\text{TE,upper}} = -\Delta P_z^{\text{TE,lower}}$	Fixed trailing-edge constraints		15
	$\Delta P_z^{\text{LE,upper,root}} = -\Delta P_z^{\text{LE,lower,root}}$	Fixed-wing root incidence constraint		1
	$2.0 \leq \alpha \leq 4.0$	Angle of attack constraint	1	1
Total constraints			7	770
Need value range limits for each DV?			NO	YES

2. Configuring DeepGeo

As in the previous case study, the L2 grid is used as the input template mesh $\hat{M}$. The objective function O_{CFD} and the loss function $\mathcal{L}_{cons}$ from Eq. 5 are defined as:

$$O_{\text{CFD}} = |C_D| + |C_L - 0.5| + \max\left(-0.17 - C_M, 0\right) \ , \tag{9}$$

$$\mathcal{L}_{cons} = \underbrace{\max\left(Vol(V^S) - Vol_{\text{original}}(S), 0\right)^2}_{\text{volume constraint}} + \underbrace{\left\|\Delta V^{\text{TE}}\right\|^2}_{\text{TE constraint}} + \underbrace{\left\|\Delta V_x^{\text{LE}}\right\|^2 + \left\|\Delta V_y^{\text{LE}}\right\|^2}_{\text{LE constraint}} + \underbrace{\left\|\delta \mathbf{v}^{\text{LE,root}}\right\|^2}_{\substack{\text{fixed-wing root} \\ \text{incidence constraint}}} \ .$$

Minimizing O_{CFD} aims to reduce C_D, bring C_L close to 0.5, and ensure C_M is greater than or equal to -0.17. The extended Shoelace formula [72] is applied to compute the volume $Vol(V^S)$ of the deformed wing, with an arbitrary reference point on the root section plane. $Vol_{\text{original}}(S)$ represents wing's initial volume. The term $\mathbf{v}^{\text{LE,root}}$ corresponds to the single leading-edge (LE) vertex on the root section, while V_x and V_y refer to the sets of x and y vertex coordinates, respectively. In practice, DeepGeo satisfies the thickness requirement without an explicit loss function term.

3. Configuring the Free-Form Deformation Baseline

For comparison, we use the FFD-based baseline with 192-control-point configuration of Li et al. [33], Hwang et al. [73], as depicted in Fig. 9(c). Let the control points be $P = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_{192}\}$, where each $\mathbf{p} = \{p_x, p_y, p_z\}$ and their displacements denoted as ΔP. The positions of control points are optimized, and the wing's surface is

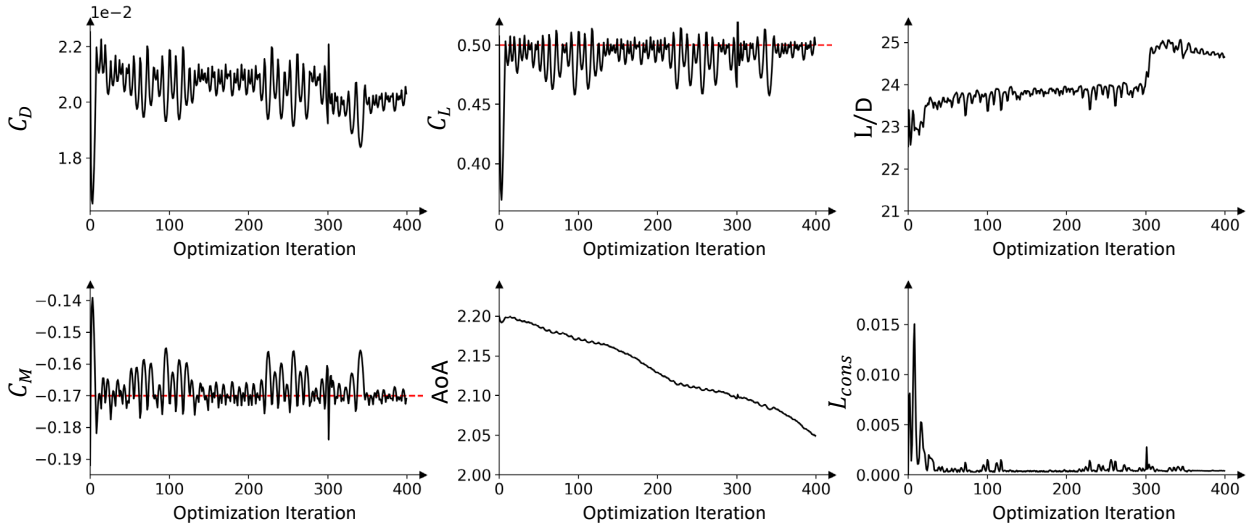


Fig. 10 The optimization history of Case Study II. The dashed line in C_L and C_M figures demonstrates the optimization objectives.

interpolated within each control cages to enable localized shape changes. Additionally, a global function is implemented to introduce spanwise twisting, with seven rotation angles treated as additional design variables. The 192 control points are grouped into seven sections along the wingspan and each section is rotated independently. For this optimization, we use the MACH-Aero framework, where pyGeo [5] implements the FFD parameterization, IDWarp [28] deforms the computational mesh and pyOptSparse [71] for optimization.

4. Results and Analysis

Comparative quantitative results are presented in Tab.3. The FFD results were obtained after extensive tuning, while DeepGeo achieved its results without requiring any parameter adjustments. The difference in the optimized drag coefficient C_D is less than 0.5%. As noted by Lyu et al. [26], even slight variations in FFD configuration can lead to increases of up to 2.4 counts in the optimized drag coefficient. Fig. 10 shows the optimization history. The comparison with the history of the FFD-based optimization is provided in Appendix A.B.

Initially, DeepGeo parameterizes the CRM wing through self-initialization, similar to the approach used in the first case study on the 2D circle optimization. During the first 300 optimization iterations, the wing's lift-to-drag ratio (L/D) increases from 22.5 to 23.8. Once the optimization converges, the partially optimized wing is reparameterized

Table 3 CRM wing optimization results of the Case Study II. One count for C_D equals to 10^{-4}.

Parameterization	**Final C_D** (counts)	**Final C_L**	**Final C_M**	**Final α**
DeepGeo	200.05	0.5000	-0.1717	2.08
FFD	199.03	0.5000	-0.1700	2.14

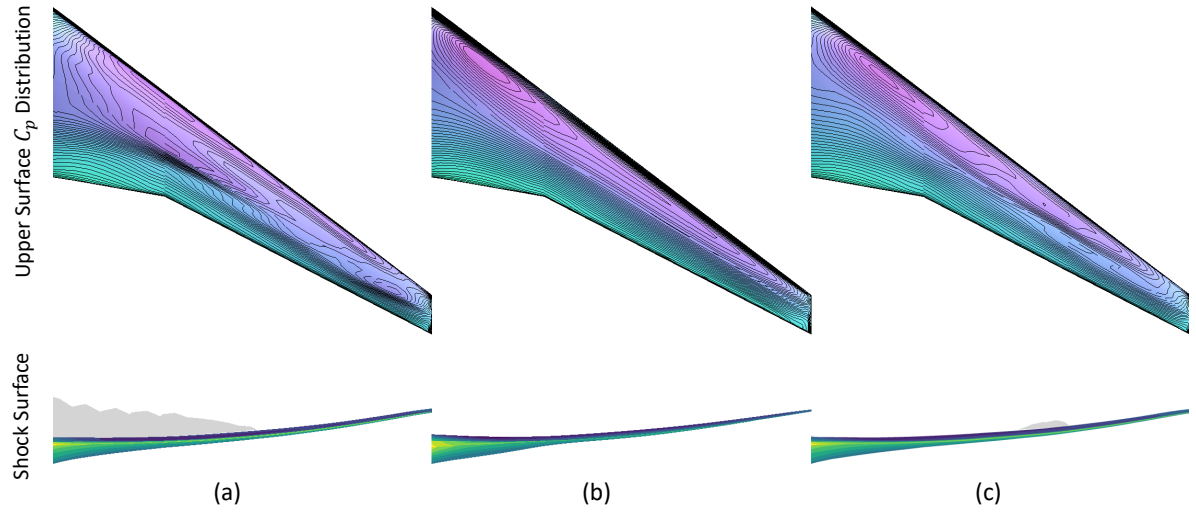


Fig. 11 A comparison of Case Study II on the pressure coefficient distributions and shock shapes of (a) the initial CRM wing, (b) the wing optimized with 192-point FFD, and (c) the wing optimized with DeepGeo.

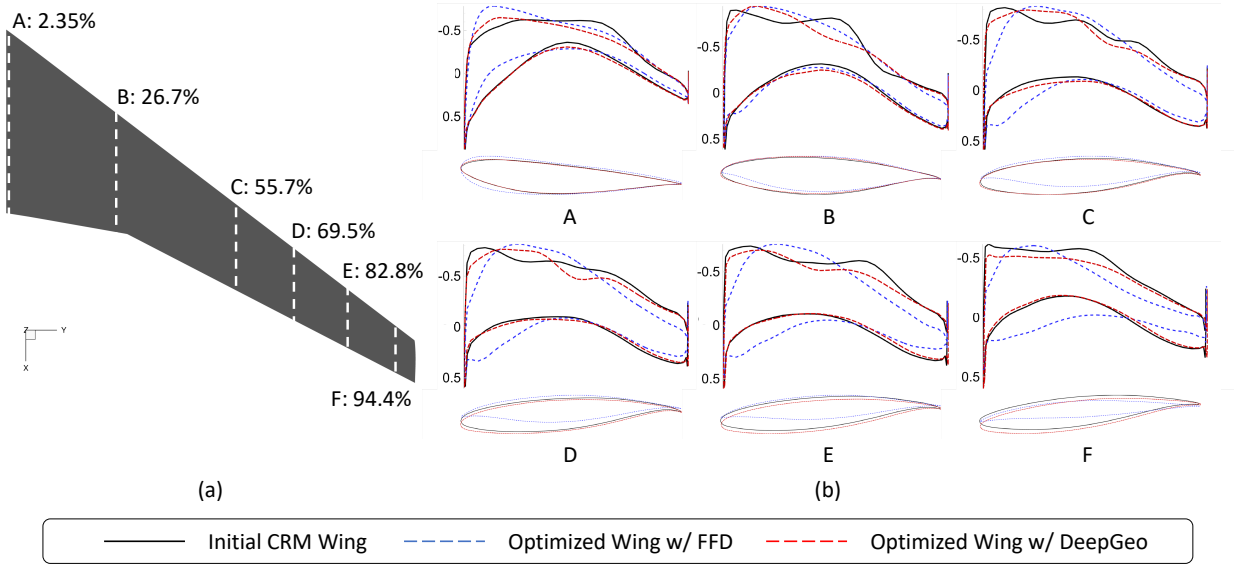


Fig. 12 A comparison of Case Study II on sliced geometries, including (a) the positions of slices, and (b) the shape variation and pressure coefficient distribution of each slice.

by self-initializing a new DeepGeo model. The optimization continues after that. The L/D ratio reaches a maximum of 25.0 before gradually decreasing, with the optimal result observed at the 320th iteration. The drag coefficient C_D has been reduced by 10.9%.

In Fig. 11, we compare the pressure distributions and shock wave patterns for both parameterization approaches. The shock waves are substantially reduced in both optimized results. The parallel alignment of pressure contour lines in both cases indicates improved aerodynamic performance, though they exhibit evident differences. Fig. 12 compares 2D

slices of the optimized wings at various locations along the wingspan. In terms of the constraint preservation, all six sectional slices across the span of DeepGeo's optimization result satisfy the 25% original thickness constraint. FFD and DeepGeo yield distinct shapes: FFD results display very sharp leading edges and significant thickness variations along the span, while DeepGeo's results show smaller deviations from the initial shape. The leading edges are less sharp, the trailing edges better adhere to geometric constraints, and the wing tips remain sufficiently thick so avoid extreme thinning.

The final design obtained with DeepGeo lies in a narrow design space close to the baseline geometry, which may be due to local convergence within the high-dimensional gradient-based optimization process under restrictive geometric constraints. The optimized shape tends to generate relatively smaller modifications of the initial design. In contrast, the FFD-based optimization aggressively reduces the leading edge thickness to minimize on-design drag, potentially compromising low-speed aerodynamic performance [33] critical for landing and takeoff. Thus, DeepGeo's approach is beneficial when starting from a good conceptual design, enabling a more controlled ASO pipeline with minimal adjustments.

In summary, DeepGeo achieves comparable aerodynamic performance to the best manually tuned FFD-based method with a simpler, less labor-intensive setup. Its ability to reduce the need for extensive tuning makes it an effective and practical choice for industrial aerodynamic optimization.

C. Optimizing the Blended-Wing-Body Aircraft

We now demonstrate DeepGeo's capability to handle complex 3D geometries without tuning. To this end, we focus on optimizing the first-generation Boeing Blended-Wing-Body (BWB) design as shown in Fig. 13. The BWB aircraft configuration integrates the wing and fuselage into a single, seamless structure to enhance aerodynamic efficiency, reduce drag, and lower fuel consumption. The BWB used in this case study has a span of 280 feet, a total length of 144 feet and is designed to carry 800 passengers [70]. A notable challenge in this design is that the supersonic airflow creates significant shock waves, introducing strong drag forces. The key to optimization is refining the BWB shape to mitigate these drag-inducing effects.

1. Problem Formulation

The optimization objective is to minimize the drag coefficient C_D while constraining the lift coefficient C_L to 0.20056 and ensuring a pitching moment coefficient C_M of 0. This single-point optimization problem is performed under fully turbulent flow conditions at a Mach number of 0.85, Reynolds number of 5×10^6 and an initial angle of attack of $0.58°$. The optimization is performed on both the mesh that has $1,070,080$ cells, corresponding to the same setting as the L2 grid configuration in Lyu and Martins [74]. Shape modifications are allowed only along the z axis to maintain a constant projected area, and the aircraft volume is constrained to remain above its initial value, preventing

shrinkage that could reduce drag but compromise capacity. Compared to the optimization task of the CRM wing, the task for BWB aircraft is less constrained. Tab. 4 summarizes the experimental setup.

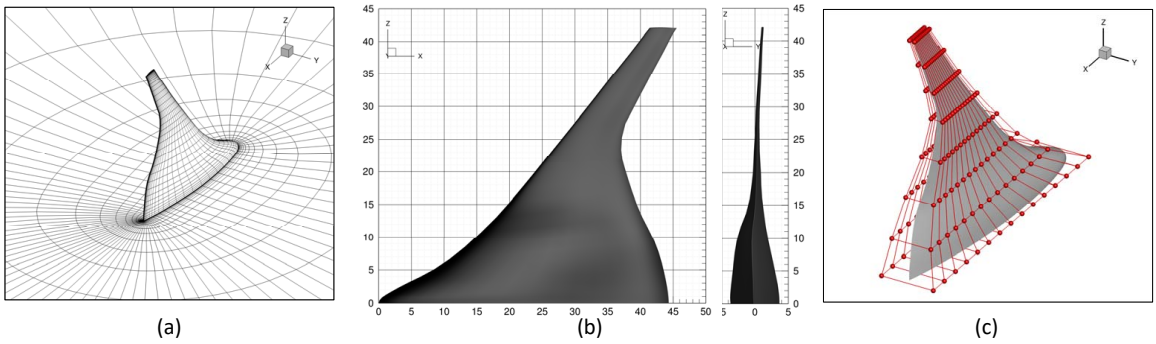


Fig. 13 Geometric setup for the BWB Case Study. (a) DeepGeo template mesh. (b) the initial BWB aircraft geometry and (c) the 240-point FFD setting.

2. Configuring DeepGeo

DeepGeo uses the L2 grid as template mesh. Similarly, DeepGeo parameterizes the BWB aircraft geometry through self-initialization, learning to generate all-zero deformation. Shape variations generated by DeepGeo are limited to the z axis, in line with the allowed modifications specified by the optimization problem. This avoids the need for additional geometric constraints on the leading edge (LE) and trailing edge (TE). The aerodynamic objective function O_{CFD} and the volume constraint $\mathcal{L}_{cons}$ from Eq. 5 are defined as

$$O_{CFD} = |C_D| + |C_L - 0.5| + |C_M| \ , \tag{10}$$

$$\mathcal{L}_{cons} = \max\left(Vol(V^S) - Vol_{\text{original}}(S), 0\right)^2 . \tag{11}$$

Table 4 Aerodynamic shape optimization task specifications for the Case Study III.

Objectives	Functions/Variables	Description	DeepGeo Quantity	FFD Quantity
Minimize	C_D	Drag coefficient		
With respect to	Θ	Weights of DeepGeo	151 585	
	P_z	Control points' z coordinates		240
	α	Angle of attack	1	1
Total design variables			151 586	241
Subject to	$C_L = 0.20056$	Lift coefficient constraint	1	1
	$C_M = 0$	Moment coefficient constraint	1	1
	$t \geq 0.01\ t_{\text{original}}$	Minimum thickness constraints		750
	$Vol \geq Vol_{\text{original}}$	Minimum volume constraint	1	1
	$0 \leq \alpha \leq 2.5$	Angle of attack constraint	1	1
Total constraints			7	754
Need value range limits for each DV?			NO	YES

Table 5 BWB aircraft optimization results of the Case Study III.

Parameterization	Final C_D (counts)	Final C_L	Final C_M	Final α
DeepGeo	86.42	0.200 72	9.8807×10^{-4}	0.60
FFD	87.05	0.200 56	2.4675×10^{-9}	2.47

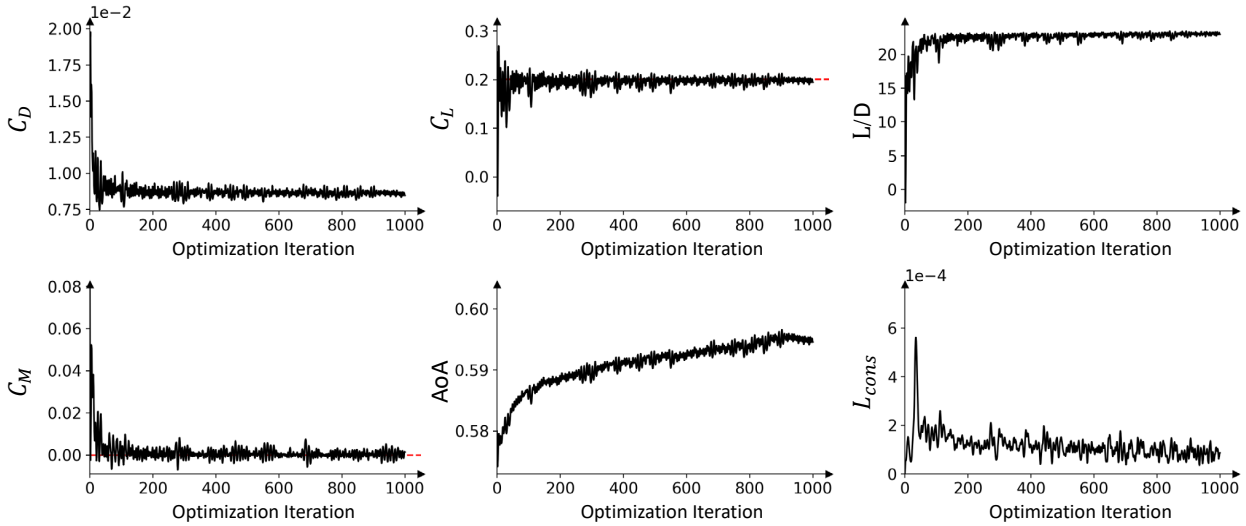


Fig. 14 Evolution of important quantities during optimization for the BWB case. The dashed line in the C_L and C_M plots represent the optimization objectives.

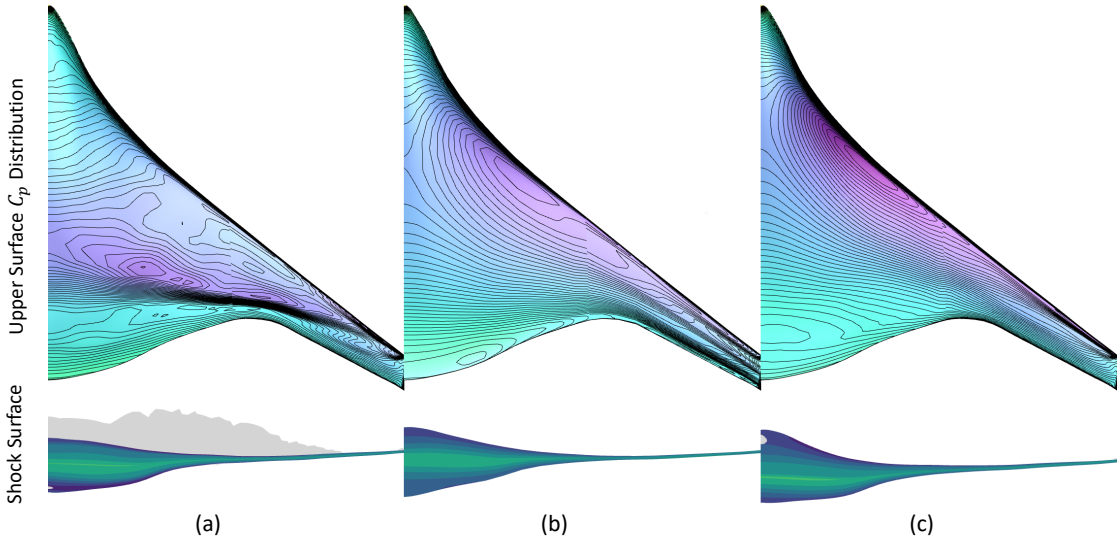


Fig. 15 A comparison of Case Study III on the pressure coefficient distributions and shock shapes of (a) the initial CRM wing, (b) the wing optimized with 240-point FFD, and (c) the wing optimized with DeepGeo.

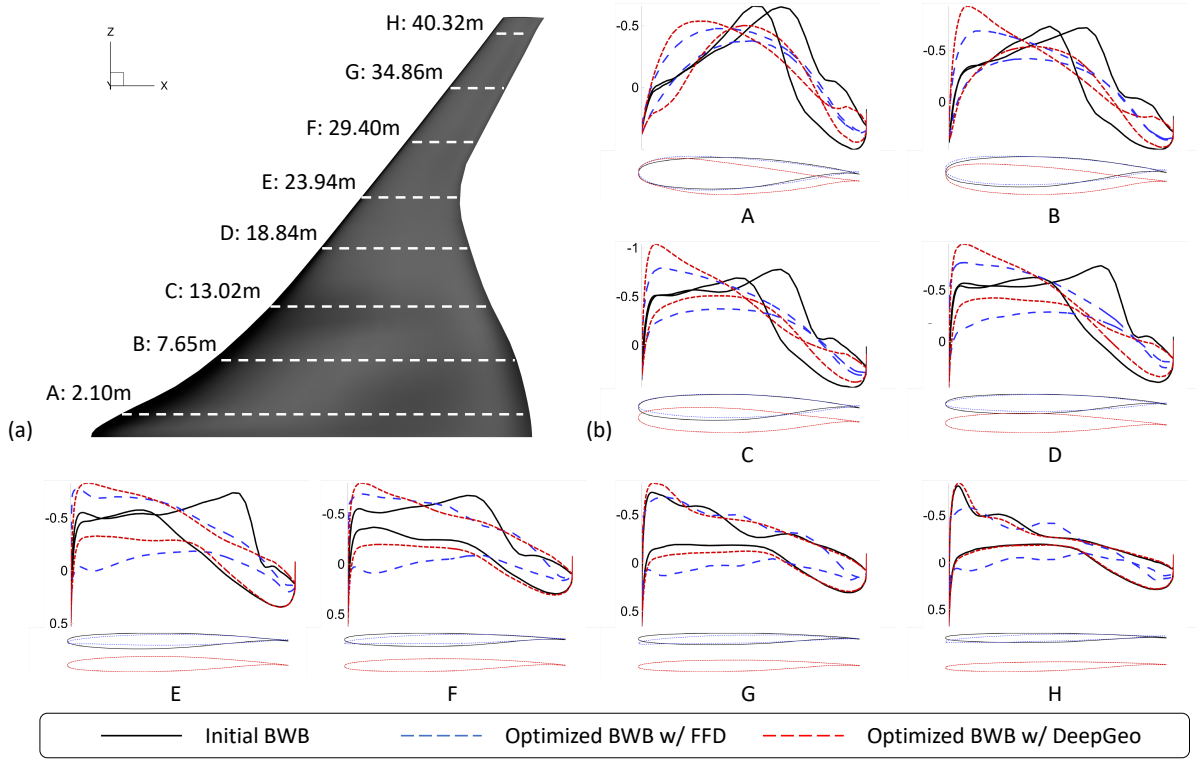


Fig. 16 A comparison of Case Study III on sliced geometries, including (a) the positions of slices, and (b) the shape variation and pressure coefficient distribution of each slice.

3. Configuring the Free-Form Deformation Baseline

The FFD configuration follows the well-established settings proposed by Lyu and Martins [74]. This configuration uses 240 control points, as shown in Fig. 13(c). A minimal thickness constraint is added to prevent crashes caused by the negative volume error during optimization. The FFD baseline case uses IPOPT implemented in pyOptSparse [71] for optimization.

4. Results and Analysis

Fig. 14 shows the evolution of key aerodynamic quantities throughout the optimization process. The comparison with the FFD-based optimization history is provided in Appendix A.B. The optimization constraints on C_L and C_M are quickly fulfilled. The DeepGeo-based optimization achieves a shock-free design by the 149th iteration, and the number of total iterations remains within the same order of magnitude as the FFD-based optimization. Quantitative results comparing the automatic DeepGeo with the best manually configured FFD baseline are presented in Tab. 5. In terms of aerodynamic performance, DeepGeo achieves 47.8% drag reduction in total and the drag performance is 0.63 counts lower than FFD's result in C_D. Furthermore, as can be seen in Fig. 15, the optimized FFD configuration exhibits a double shock near the wing tip, which is an undesirable feature in practical applications. In contrast, the

DeepGeo-optimized design is shock-wave-free near the tip.

Fig. 16, provides a sectional analysis of geometry variations along the wingspan. The DeepGeo-optimized result shows more drastic changes in the spanwise bending and is less tilted upwards, while the shape variation on each slice is less significant. Similar to the results for the CRM wing, the FFD-optimized design has much sharper leading edges, which may negatively impact low-speed aerodynamic performance.

In summary, DeepGeo provides greater deformation freedom and, consequently, a broader design space for exploration when given less restrictive geometric constraints. In contrast, achieving similar flexibility with FFD requires users to manually implement various global design functions and introduce additional design variables, such as sweep, twist and dihedral functions, while carefully constraining each new design variable to avoid optimization failures. This approach requires significant additional engineering effort, making DeepGeo a more efficient and flexible alternative.

V. Discussion

A. Justification of DeepGeo's Over-Parameterization and Complexity

Traditional parameterizations rely on low-dimensional design variables and suffer from several key limitations:

- They require extensive manual intervention. This includes selecting an appropriate method, configuring parameters to balance flexibility, smoothness and robustness, and carefully defining parameter constraints. Moreover, capturing essential inter-parameter correlations–such as the twisting or sweeping global functions used in wing optimization–must be explicitly hard-coded, which heavily depends on domain expertise.
- They rely on costly trial-and-error procedures. The quality of results often depends on iterative manual experimentation. For example, an inappropriate FFD configuration of a CRM wing can lead to a 42% reduction in drag improvement [26], making multiple optimization attempts necessary to achieve a near-optimal design.

In contrast, DeepGeo leverages over-parameterization to enable its capability, adaptability and robustness. The large number of parameters is not a drawback but a necessary design choice that allows DeepGeo to capture complex geometric variations and to implement the benefits described above through a fully automatic, data-independent process. During its initialization phase, as defined by Eq. 2, DeepGeo automatically learns a design parameterization by combining the selection, setting and integration of parameters into an adaptive framework. Consequently, DeepGeo's most significant advantages over traditional parameterizations are the reduction in manual intervention, minimized dependency on expert prior and the elimination of costly trial-and-error practices.

Using the number of weights for as a proxy for DeepGeo's actual complexity is misleading when comparing it to methods that do not rely on deep learning. This is because in DeepGeo, as in most deep-learning approaches with high-dimensional parameter spaces,the large parameter count does not necessarily translate into excessive complexity. Recent studies have revealed the *double descent* phenomenon [75, 76], as illustrated in Fig. 17, which shows that as

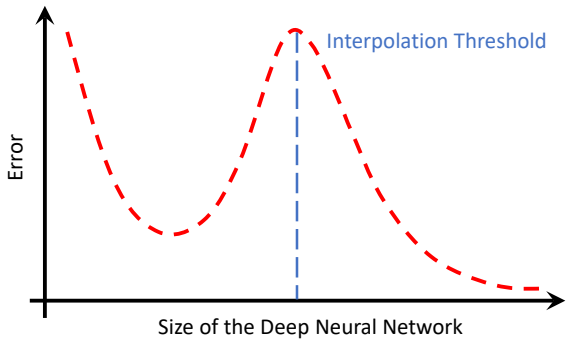


Fig. 17 The *double descent* phenomenon in deep learning models.

model size increases, the generalization error first decreases, then rises near the interpolation threshold, and finally decreases again with further over-parameterization. This second descent implies that highly over-parameterized models can be more effective despite their excessive parameter amount. This behavior is due to the weight correlations developed in neural networks [77] and the reduced number of independent degrees of freedom during training. Therefore, a more meaningful indicator of DeepGeo's complexity is its *effective dimensionality*, which quantifies the number of independent directions in parameter space that influence the model's behavior–akin to reduced dimensions in model order reduction. A well-trained network typically exhibits low *effective dimensionality*, meaning that its function lies in a relatively low-dimensional parameter subspace despite its large number of parameters, and it can effectively compress the information from data.

Table 6 Comparison of effective dimensionality for DeepGeo models. The values of random model are based on 20 repeats.

Model	**Layer 1** weight ratio: 0.1%	**Layer 2** weight ratio: 11.0%	**Layer 3** weight ratio: 87.9%	**Output Layer** weight ratio: 1.0%	**Overall**
Random	264.0 ± 44.9	114.4 ± 63.1	226.5 ± 33.5	332.7 ± 34.4	215.3 ± 37.1
2D Circle	2.6	2.9	2.6	3.9	2.6
CRM Wing	2.3	0.4	0.1	4.8	0.2
BWB Aircraft	134.8	51.5	24.6	−0.5	27.4

To empirically validate this, we compute DeepGeo's *effective dimensionality* using a metric based on the eigenvalues of the symmetric Hessian matrix of DeepGeo's weights [78] (as defined by Equation A.1). More details are included in Appendix A.A. As shown in Table 6, the initialized DeepGeo model has significantly lower effective dimensionalities compared to randomly initialized ones. This supports our claim that DeepGeo's actual complexity is far less than its parameter count may suggest.

B. Justification of DeepGeo's Smoothness, Expressiveness and the Role of L_{reg}

DeepGeo provides both global surface smoothness and sufficient deformation freedom to support high-fidelity design optimization. Several factors contribute to this. First, the smoothness arises naturally from the inherent properties of its MLP backbone, known in literature as the frequency principle [79, 80] or spectral bias [81]. As a result, the network tends to prioritize low-frequency features, while high-frequency components typically require more training iterations to emerge. Since ASO usually requires fewer iterations to converge than for typical deep learning training, undesired high-frequency patterns, such as bumps or discontinuities, are unlikely to appear. Second, DeepGeo learns to create a shape deformation field rather than a full geometry from scratch and such a field is inherently smoother. For instance, when deforming an airfoil, DeepGeo does not need to reconstruct sharp trailing edges or rounded leading edges from zero. Instead, it learns a smooth transformation that modifies these features, simplifying the learning and contributing to overall surface regularity. Third, the use of a gradient-descent-based optimizer leads to smooth transitions in the design trajectory. At each iteration, the newly deformed geometry is a slight variant of the previous one. This mostly avoids abrupt or unstable geometric transformations.

DeepGeo's expressiveness is supported by its over-parameterized MLP structure and, according to the universal approximation theory [13, 14], this expressiveness is up-bounded by the model size. However, empirical results in Section V.A show that the current model size has enabled the over-parameterization effect for all ASO tasks presented in this work. If DeepGeo were to be applied to significantly more complex tasks, increasing model size might be necessary.

The Hessian-based regularization loss L_{reg}, though introduced to stabilize volumetric mesh deformation, directly penalizes the magnitude of the network's Hessian, and could potentially constrain DeepGeo's frequency behavior. While a full understanding of theoretical effects of this regularization on the model's spectral properties would require further investigation, our empirical results do not indicate a significant loss in DeepGeo's expressiveness when L_{reg} is properly weighted. As shown in Figure 4, even changing the weight of L_{reg} by several orders of magnitude produces nearly identical deformed shapes. Furthermore, we analyze the gradient magnitudes of O_{CFD} and L_{reg} with respect to DeepGeo's parameters at the start of design optimization. As shown in Table 7, the norm of ∇L_{reg} is multiple orders of magnitude smaller than that of ∇O_{CFD} across all layers. This demonstrates that the physical objective driven by the adjoint solver is the dominant force that shapes DeepGeo's behavior.

Additionally, DeepGeo is different from traditional spectral parameterization methods used in ASO, such as Li et al. [82], Li and Zhang [83]. Those methods rely on explicit mode truncation for dimensionality reduction, which limits the design space in advance. In contrast, DeepGeo does not manually cut off its spectrum, it instead adapts its behavior according to different geometries and design tasks through its optimization.

Recent advances in computer vision and graphics have introduced techniques such as positional embeddings [84] and feature grids [85] to inject high-frequency detail into MLPs. These techniques offer promising directions for future work aimed at implementing DeepGeo's controllable smoothness and expanding the model's expressiveness when

Table 7 The comparison of mean vector norms of ∇O_{CFD} and ∇L_{reg} with respect to all DeepGeo's layers

ASO Case	Mean Vector Norm of ∇O_{CFD}			
	Layer 1	Layer 2	Layer 3	Output Layer
2D Circle	0.1825	0.4186	0.5105	6.3915
CRM Wing	0.1254	0.2814	0.2411	6.4747
BWB Aircraft	0.1137	0.2166	0.1495	1.3316
ASO Case	Mean Vector Norm of ∇L_{reg}			
	Layer 1	Layer 2	Layer 3	Output Layer
2D Circle	0.0003	0.0003	0.0009	0.0235
CRM Wing	9.4266×10^{-5}	9.9453×10^{-5}	3.8320×10^{-5}	0.0016
BWB Aircraft	0.0009	0.0008	0.0003	0.0034

needed.

VI. Conclusion

This research introduces a novel framework that leverages deep geometric learning to implement a fully automated shape parameterization method for Aerodynamic Shape Optimization (ASO). By automating the parameterization process, DeepGeo significantly enhances efficiency and robustness over traditional methods, minimizing the need for manual intervention while maintaining optimization performance that is either superior or at least comparable to the best manually tuned techniques. Specifically, in comparison to the state-of-the-art Free-Form Deformation (FFD) model, DeepGeo fully automates the handling of diverse geometries in both 2D and 3D. In contrast, FFD requires users to design a custom configuration for each task to achieve optimal performance, which demands extensive tuning and strong domain prior knowledge.

Additionally, DeepGeo integrates Computational Fluid Dynamics (CFD) mesh deformation directly into the parameterization model within the adjoint-based ASO pipeline, whereas traditional methods rely on a separate module to address meshing issues as pre-processing to physical simulations. This integration simplifies the ASO pipeline and removes the need for additional mesh management. Furthermore, DeepGeo offers significant freedom in shape deformation while maintaining global surface smoothness. For example, in the case study that optimizes a 2D circle, DeepGeo successfully converges to a solution without complex adjustments, whereas FFD encounters substantial challenges and fails to achieve convergence. The necessary remedy for FFD were complex and non-trivial [86]. Notably, DeepGeo works without requiring any training data, while it still leverages the strengths of deep learning to manage high-dimensional optimization problems. This data independence eliminates the needs extensive data engineering. The DeepGeo-based ASO framework requires no additional user input compared to the conventional ones.

By automating shape parameterization and simplifying the geometric processing in ASO, DeepGeo significantly reduces development complexity, making this technology more accessible to a broader range of users. This decentraliza-

tion enables more engineers and designers to leverage advanced computer-aided engineering capabilities with minimal technical barriers. In an industrial context, DeepGeo's automated framework also has the potential to substantially lower ASO development time and costs, facilitating faster prototyping and iterative design cycles without the need for extensive manual adjustments.

In future work, we plan to explore more advanced ASO applications, such as optimizing biologically inspired shapes or designs derived from user sketches, by utilizing DeepGeo's simplified geometric processing capabilities and its independence from expert prior. Meanwhile, we will explore the coupling of data-driven inverse design model and DeepGeo, where DeepGeo could benefit from better design initializations for more effective design optimization. Additionally, the automation provided by DeepGeo can potentially benefit other engineering domains that require high-dimensional optimization and shape manipulation. We aim to investigate how DeepGeo could serve as a foundational geometric model in a broader range of computer-aided engineering applications, potentially driving innovation across multiple disciplines.

Appendix

A. Computation of DeepGeo's *effective dimensionality*

To quantify the *effective dimensionality* of DeepGeo, we use the approach described in Maddox et al. [78]. In this framework, *effective dimensionality* is computed using the eigenvalues $\{\lambda_1, \lambda_2, ..., \lambda_K\}$ of a symmetry matrix–the Hessian of DeepGeo's weights $\mathbf{H}_W$, defined as

$$N_{eff}(\mathbf{H}_W) = \sum_{i=1}^{K} \frac{\lambda_i}{\lambda_i + z} , \tag{A.1}$$

where $z = 10^{-3}$ is a regularization constant.

Instead of calculating a single large Hessian for all parameters, we compute the Hessian separately for each fully connected layer (layers 1–3 and the output layer). This layer-wise computation not only reduces the computational burden but also provides clearer insight into how each layer contributes to the overall *effective dimensionality*. For each layer, the Hessian matrix is obtained using Pytorch's auto-differentiation. The overall value is computed as a weighted sum of each layer's *effective dimensionality*, with weights corresponding to the parameter ratio of that layer in DeepGeo. For random DeepGeo models, the *effective dimensionality* is computed with 20 repeated random initializations. This methodology provides a more in-depth view of how DeepGeo utilizes over-parameterization through its initialization.

B. Comparison with FFD-Based Optimization Convergence History

Figure A.1 shows the convergence histories for optimizations with both FFD and DeepGeo. The CRM wing case uses 192-point FFD configuration, while the BWB aircraft case uses 240 control points. Both FFD-based optimizations

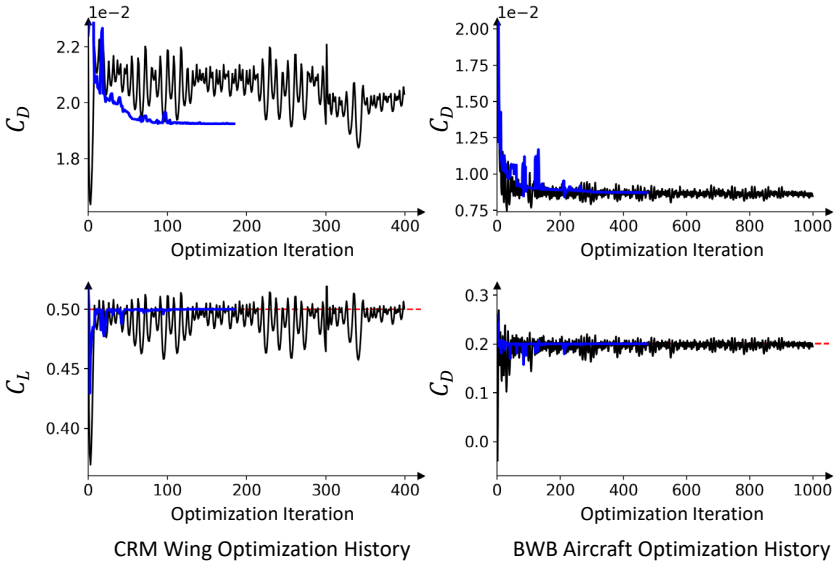


Fig. A.1 The comparison of DeepGeo-based (black) and FFD-based (blue) optimization convergence histories.

use the IPOPT optimizer.

The FFD-based optimizations, as shown in blue lines, converge faster than the DeepGeo-based ones, though the number of iterations remains within the same order of magnitude. DeepGeo can achieve significant improvement at early stage, but requires more iterations to reach better optima. For example, in the BWB aircraft case, the DeepGeo-based optimization achieves a shock-free design by the 149th iteration, but continued to improve and reached higher performance after 998 iterations. DeepGeo's convergence path can be less stable due to the inefficient first-order gradient descent optimizer. We believe that further investigation into more advanced optimization strategies could improve DeepGeo's convergence behavior.

Acknowledgement

Zhen Wei is supported in part by the Swiss National Science Foundation (SNSF 200020_204231) and the French "Programme d'Investissements d'avenir" EUR-TSAE. Michaël Bauerheim is supported by the French Direction Générale de l'Armement through the Agence de l'Innovation de Défense in the project DECAP (Design et Contrôle par Apprentissage Profond) project. Rhea P. Liem and Aobo Yang are supported by the Hong Kong Research Grant Council General Research Fund (Project No. 16210621).

References

[1] Hicks, R. M., and Henne, P. A., "Wing Design by Numerical Optimization," Journal of Aircraft, Vol. 15, No. 7, 1978, pp. 407–412. https://doi.org/10.2514/3.58379.

[2] Kulfan, B. M., "Universal Parametric Geometry Representation Method," Journal of Aircraft, Vol. 45, No. 1, 2008, pp. 142–158. https://doi.org/10.2514/1.29958.

[3] Sederberg, T., and Parry, S., "Free-Form Deformation of Solid Geometric Models," ACM SIGGRAPH, Vol. 20, No. 4, 1986.

[4] Lamousin, H. J., and Jr., W. N. W., "NURBS-based free-form deformations," Computer Graphics and Applications, Vol. 14, No. 6, 1994, pp. 59–65. https://doi.org/10.1109/38.329096.

[5] Kenway, G., Kennedy, G., and Martins, J. R. R. A., "A CAD-Free Approach to High-Fidelity Aerostructural Optimization," 13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference, Fort Worth, Texas, USA, September 13-15, 2010. https://doi.org/10.2514/6.2010-9231.

[6] Masters, D. A., Taylor, N. J., Rendall, T. C. S., Allen, C. B., and Poole, D. J., "Geometric Comparison of Aerofoil Shape Parameterization Methods," American Institute of Aeronautics and Astronautics Journal, Vol. 55, No. 5, 2017, pp. 1575–1589.

[7] Vuruskan, A., and Hosder, S., "Impact of Turbulence Models and Shape Parameterization on Robust Aerodynamic Shape Optimization," Journal of Aircraft, Vol. 56, No. 3, 2019, pp. 1099–1115. https://doi.org/10.2514/1.C035039.

[8] Song, W., and Keane, A., "A Study of Shape Parameterisation Methods for Airfoil Optimisation," 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, 2004. https://doi.org/10.2514/6.2004-4482.

[9] Bamford, T., Keane, A., and Toal, D., "SDF-GAN: Aerofoil Shape Parameterisation via an Adversarial Auto-Encoder," AIAA AVIATION FORUM AND ASCEND 2024, 2024. https://doi.org/10.2514/6.2024-3665.

[10] Swannet, K., Varriale, C., and Doan, N. A. K., "Towards Universal Parameterization: Using Variational Autoencoders to Parameterize Airfoils," AIAA SCITECH 2024 Forum, 2024. https://doi.org/10.2514/6.2024-0686.

[11] Secchi, P. d., and Sêcco, N. R., "A Fully Data-Driven Generative Design Routine for Subsonic Airfoils Based on Adversarial Autoencoders," AIAA AVIATION FORUM AND ASCEND 2024, 2024. https://doi.org/10.2514/6.2024-3667.

[12] Jameson, A., "Optimum aerodynamic design using CFD and control theory," Computational Fluid Dynamics Conference, 1995, p. 1729.

[13] Barron, A., "Universal approximation bounds for superpositions of a sigmoidal function," IEEE Transactions on Information Theory, Vol. 39, No. 3, 1993, pp. 930–945. https://doi.org/10.1109/18.256500.

[14] Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., and Liao, Q., "Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review," International Journal of Automation and Computing, Vol. 14, 2017, pp. 503–519. https://doi.org/10.1007/s11633-017-1054-2.

[15] Hansen, N., "The CMA Evolution Strategy: A Tutorial," arXiv Preprint, 2016.

[16] Wei, Z., Fua, P., and Bauerheim, M., "Automatic Parameterization for Aerodynamic Shape Optimization via Deep Geometric Learning," AIAA AVIATION Forum, 2023, p. 3471.

[17] Jameson, A., "Aerodynamic Design via Control Theory," Journal of Scientific Computing, Vol. 3, 1988, pp. 233–260. https://doi.org/10.1007/BF01061285.

[18] Braibant, V., and Fleury, C., "Shape optimal design using B-splines," Computer Methods in Applied Mechanics and Engineering, Vol. 44, No. 3, 1984, pp. 247–267. https://doi.org/10.1016/0045-7825(84)90132-4.

[19] Farin, G. E., NURB curves and surfaces: from projective geometry to practical use, A. K. Peters, Ltd., 63 South Avenue Natick, MA, United States, 1995.

[20] Bloor, M. I. G., and Wilson, M. J., "Efficient parametrization of generic aircraft geometry," Journal of Aircraft, Vol. 32, No. 6, 1995, pp. 1269–1275. https://doi.org/10.2514/3.46874.

[21] Smith, R., Bloor, M., Wilson, M., and Thomas, A., "Rapid airplane parametric input design (RAPID)," 12th Computational Fluid Dynamics Conference, 1995. https://doi.org/10.2514/6.1995-1687.

[22] Pickett, R. M., Rubinstein, M., and Nelson, R., "Automated Structural Synthesis Using a Reduced Number of Design Coordinates," AIAA Journal, Vol. 11, No. 4, 1973, pp. 489–494. https://doi.org/10.2514/3.50489.

[23] Hager, J., Eyi, S., and Lee, K., "A multi-point optimization for transonic airfoil design," 4th Symposium on Multidisciplinary Analysis and Optimization, 1992. https://doi.org/10.2514/6.1992-4681.

[24] Sobieczky, H., "Parametric Airfoils and Wings," Recent Development of Aerodynamic Design Methodologies: inverse Design and Optimization, 1999, pp. 71–87.

[25] Barr, A. H., "Local and Global Deformations of Solid Primitives," ACM SIGGRAPH, Vol. 18, No. 3, 1984, pp. 21–30.

[26] Lyu, Z., Kenway, G. K. W., and Martins, J. R. R. A., "Aerodynamic Shape Optimization Investigations of the Common Research Model Wing Benchmark," American Institute of Aeronautics and Astronautics Journal, Vol. 53, No. 4, 2015, pp. 968–985. https://doi.org/10.2514/1.J053318.

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

[28] Secco, N. R., Kenway, G. K. W., He, P., Mader, C., and Martins, J. R. R. A., "Efficient Mesh Generation and Deformation for Aerodynamic Shape Optimization," AIAA Journal, Vol. 59, No. 4, 2021, pp. 1151–1168. https://doi.org/10.2514/1.J059491, URL https://doi.org/10.2514/1.J059491.

[29] Ceze, M., Hayashi, M., and Volpe, E., "A Study of the CST Parameterization Characteristics," 27th AIAA Applied Aerodynamics Conference, 2009. https://doi.org/10.2514/6.2009-3767.

[30] Li, J., Du, X., and Martins, J. R., "Machine Learning in Aerodynamic shape Optimization," Progress in Aerospace Sciences, Vol. 134, 2022, p. 100849.

[31] Robinson, G. M., and Keane, A. J., "Concise Orthogonal Representation of Supercritical Airfoils," Journal of Aircraft, Vol. 38, No. 3, 2001, pp. 580–583.

[32] Poole, D. J., Allen, C. B., and Rendall, T. C. S., "Metric-Based Mathematical Derivation of Efficient Airfoil Design Variables," American Institute of Aeronautics and Astronautics Journal, Vol. 53, No. 5, 2015, pp. 1349–1361.

[33] Li, J., Bouhlel, M. A., and Martins, J. R. R. A., "Data-Based Approach for Fast Airfoil Analysis and Optimization," American Institute of Aeronautics and Astronautics Journal, Vol. 57, No. 2, 2019, pp. 581–596.

[34] Kedward, L., Allen, C. B., and Rendall, T., "Towards Generic Modal Design Variables for Aerodynamic Shape Optimisation," AIAA Scitech Forum, 2020.

[35] Constantine, P. G., Dow, E., and Wang, Q., "Active Subspace Methods in Theory and Practice: Applications to Kriging Surfaces," SIAM Journal on Scientific Computing, Vol. 36, No. 4, 2014, pp. A1500–A1524.

[36] Li, J., Cai, J., and Qu, K., "Surrogate-Based Aerodynamic Shape Optimization with the Active Subspace Method," Structural and Multidisciplinary Optimization, Vol. 59, No. 2, 2019, pp. 403–419. https://doi.org/10.1007/s00158-018-2073-5.

[37] Lukaczyk, T. W., Constantine, P., Palacios, F., and Alonso, J. J., "Active Subspaces for Shape Optimization," Multidisciplinary Design Optimization Conference, 2014. https://doi.org/10.2514/6.2014-1171.

[38] Namura, N., Shimoyama, K., and Obayashi, S., "Kriging surrogate model with coordinate transformation based on likelihood and gradient," Journal of Global Optimization, Vol. 68, No. 3, 2017, pp. 827–849. https://doi.org/10.1007/s10898-017-0516-y.

[39] Grey, Z. J., and Constantine, P. G., "Active Subspaces of Airfoil Shape Parameterizations," American Institute of Aeronautics and Astronautics Journal, Vol. 56, No. 5, 2018, pp. 2003–2017. https://doi.org/10.2514/1.J056054.

[40] Bauerheim, M., Ndiaye, A., Constantine, P., Moreau, S., and Nicoud, F., "Symmetry breaking of azimuthal thermoacoustic modes: the UQ perspective," Journal of Fluid Mechanics, Vol. 789, 2016, p. 534–566. https://doi.org/10.1017/jfm.2015.730.

[41] Payne, L. E., and Weinberger, H. F., "An optimal Poincaré inequality for convex domains," Archive for Rational Mechanics and Analysis, Vol. 5, 1960, pp. 286–292. https://doi.org/10.1007/BF00252910.

[42] Beyer, K., Goldstein, J., Ramakrishnan, R., and Shaft, U., "When is "nearest neighbor" meaningful?" International Conference on Database Theory, 1999, pp. 217–235. https://doi.org/10.1007/3-540-49257-7_15.

[43] Viswanath, A., Forrester, A. I. J., and Keane, A. J., "Dimension Reduction for Aerodynamic Design Optimization," American Institute of Aeronautics and Astronautics Journal, Vol. 49, No. 6, 2011, pp. 1256–1266.

[44] Achour, G., Sung, W. J., Pinon-Fischer, O. J., and Mavris, D. N., "Development of a Conditional Generative Adversarial Network for Airfoil Shape Optimization," AIAA Scitech Forum, 2020.

[45] Chen, W., Chiu, K., and Fuge, M. D., "Airfoil Design Parameterization and Optimization Using Bézier Generative Adversarial Networks," American Institute of Aeronautics and Astronautics Journal, Vol. 58, No. 11, 2020, pp. 4723–4735. https://doi.org/10.2514/1.J059317.

[46] Li, J., Zhang, M., Martins, J. R. R. A., and Shu, C., "Efficient Aerodynamic Shape Optimization with Deep-Learning-Based Geometric Filtering," American Institute of Aeronautics and Astronautics Journal, Vol. 58, No. 10, 2020, pp. 4243–4259.

[47] Li, J., and Zhang, M., "On Deep-Learning-Based Geometric Filtering in Aerodynamic Shape Optimization," Aerospace Science and Technology, Vol. 112, 2021, p. 106603.

[48] Wei, Z., Dufour, E., Pelletier, C., Fua, P., and Bauerheim, M., "Diffairfoil: An efficient novel airfoil sampler based on latent space diffusion model for aerodynamic shape optimization," AIAA AVIATION FORUM AND ASCEND, 2024. https://doi.org/10.2514/6.2024-3755.

[49] Yang, A., Zhang, J., Li, J., and Liem, R., "Data-driven Aerodynamic Shape Optimization and Multi-fidelity Design Exploration using Conditional Diffusion-based Geometry Sampling Method," Proceedings of the 34th Congress of the International Council of the Aeronautical Sciences (ICAS 2024), Florence, Italy, 2024.

[50] Wei, Z., Guillard, B., Bauerheim, M., Chapin, V., and Fua, P., "Latent Representation of CFD Meshes and Application to 2D Airfoil Aerodynamics," American Institute of Aeronautics and Astronautics Journal, 2023. https://doi.org/10.2514/1.J062533.

[51] Sheffer, A., Praun, E., and Rose, K., "Mesh Parameterization Methods and Their Applications," Foundations and Trends® in Computer Graphics and Vision, Vol. 2, No. 2, 2006, pp. 105–171. https://doi.org/10.1561/0600000011.

[52] Floater, M. S., and Hormann, K., "Surface Parameterization: a Tutorial and Survey," Advances in Multiresolution for Geometric Modelling, 2005, pp. 157–186.

[53] Sumner, R., and Popovic, J., "Deformation Transfer for Triangle Meshes," ACM Transactions on Graphics, 2004, pp. 399–405.

[54] Sorkine, O., and Alexa, M., "As-Rigid-As-Possible Surface Modeling," Geometry Processing, 2007. https://doi.org/10.2312/SGP/SGP07/109-116.

[55] Lévy, B., Petitjean, S., Ray, N., and Maillot, J., "Least Squares Conformal Maps for Automatic Texture Atlas Generation," ACM Transactions on Graphics, Vol. 21, No. 3, 2002, pp. 362–371. https://doi.org/10.1145/566654.566590.

[56] Schreiner, J., Asirvatham, A., Praun, E., and Hoppe, H., "Inter-Surface Mapping," ACM Transactions on Graphics, Vol. 23, No. 3, 2004, pp. 870–877. https://doi.org/10.1145/1015706.1015812.

[57] Smith, J., and Schaefer, S., "Bijective Parameterization with Free Boundaries," ACM Transactions on Graphics, Vol. 34, No. 4, 2015. https://doi.org/10.1145/2766947.

[58] Terzopoulos, D., Witkin, A., and Kass, M., "Symmetry-Seeking Models and 3D Object Reconstruction," International Journal of Computer Vision, Vol. 1, 1987, pp. 211–221.

[59] Ulyanov, D., Vedaldi, A., and Lempitsky, V., "Deep Image Prior," Conference on Computer Vision and Pattern Recognition, 2018, pp. 9446–9454.

[60] Williams, F., Schneider, T., Silva, C. T., Zorin, D., Bruna, J., and Panozzo, D., "Deep Geometric Prior for Surface Reconstruction," Conference on Computer Vision and Pattern Recognition, 2019, pp. 10130–10139. https://doi.org/10.1109/CVPR.2019.01037.

[61] Hanocka, R., Metzer, G., Giryes, R., and Cohen-Or, D., "Point2Mesh: A Self-Prior for Deformable Meshes," ACM SIGGRAPH, Vol. 39, No. 4, 2020. https://doi.org/10.1145/3386569.3392415.

[62] Kingma, D. P., and Ba, J., "Adam: A Method for Stochastic Optimisation," International Conference on Learning Representations, 2015.

[63] Donoho, D. L., and Grimes, C., "Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data," Proceedings of the National Academy of Sciences USA, 2003.

[64] Salimans, T., and Kingma, D., "Weight Normalization - A Simple Reparameterization to Accelerate Training of Deep Neural Networks," Advances in Neural Information Processing Systems, 2016.

[65] Fukushima, K., "Visual Feature Extraction by a Multilayered Network of Analog Threshold Elements," IEEE Transactions on Systems Science and Cybernetics, Vol. 5, No. 4, 1969, pp. 322–333. https://doi.org/10.1109/TSSC.1969.300225.

[66] Mader, C. A., Kenway, G. K. W., Yildirim, A., and Martins, J. R. R. A., "ADflow—An open-source computational fluid dynamics solver for aerodynamic and multidisciplinary optimization," Journal of Aerospace Information Systems, 2020. https://doi.org/10.2514/1.I010796.

[67] Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J., "On the Variance of the Adaptive Learning Rate and Beyond," International Conference on Learning Representations, 2020.

[68] Morgan, N., and Bourlard, H., "Generalization and Parameter Estimation in Feedforward Netws: Some Experiments," Neural Information Processing Systems, 1989. URL https://api.semanticscholar.org/CorpusID:18821787.

[69] Vassberg, J., Dehaan, M., Rivers, M., and Wahls, R., Development of a Common Research Model for Applied CFD Validation Studies, 2008. https://doi.org/10.2514/6.2008-6919.

[70] Liebeck, R. H., "Design of the Blended Wing Body Subsonic Transport," Journal of Aircraft, Vol. 41, No. 1, 2004, pp. 10–25. https://doi.org/10.2514/1.9084.

[71] Wu, E., Kenway, G., Mader, C. A., Jasa, J., and Martins, J. R. R. A., "pyOptSparse: A Python framework for large-scale constrained nonlinear optimization of sparse systems," Journal of Open Source Software, Vol. 5, No. 54, 2020, p. 2564. https://doi.org/10.21105/joss.02564.

[72] Zhang, C., and Chen, T., "Efficient feature extraction for 2D/3D objects in mesh representation," International Conference on Image Processing, Vol. 3, 2001, pp. 935–938.

[73] Hwang, J. T., Jasa, J. P., and Martins, J. R. R. A., "High-Fidelity Design-Allocation Optimization of a Commercial Aircraft Maximizing Airline Profit," Journal of Aircraft, Vol. 56, No. 3, 2019, pp. 1164–1178.

[74] Lyu, Z., and Martins, J. R. R. A., "Aerodynamic Design Optimization Studies of a Blended-Wing-Body Aircraft," Journal of Aircraft, Vol. 51, No. 5, 2014, pp. 1604–1617. https://doi.org/10.2514/1.C032491.

[75] Belkin, M., Hsu, D., Ma, S., and Mandal, S., "Reconciling modern machine-learning practice and the classical bias–variance trade-off," Proceedings of the National Academy of Sciences, Vol. 116, No. 32, 2019, pp. 15849–15854. https://doi.org/10.1073/pnas.1903070116.

[76] Spigler, S., Geiger, M., d'Ascoli, S., Sagun, L., Biroli, G., and Wyart, M., "A Jamming Transition From Under- to Over-Parametrization Affects Generalization In Deep Learning," Journal of Physics A: Mathematical and Theoretical, Vol. 52, No. 47, 2019, p. 474001. https://doi.org/10.1088/1751-8121/ab4c8b.

[77] Jin, G., Yi, X., Zhang, L., Zhang, L., Schewe, S., and Huang, X., "How does Weight Correlation Affect Generalisation Ability of Deep Neural Networks?" Advances in Neural Information Processing Systems, Vol. 33, 2020, pp. 21346–21356.

[78] Maddox, W. J., Benton, G., and Wilson, A. G., "Rethinking Parameter Counting in Deep Models: Effective Dimensionality Revisited," arXiv Preprint, 2020.

[79] Xu, Z.-Q. J., Zhang, Y., and Xiao, Y., "Training Behavior of Deep Neural Network in Frequency Domain," Neural Information Processing, Vol. 11953, 2019, pp. 264–274. https://doi.org/10.1007/978-3-030-36708-4_22.

[80] Xu, Z.-Q. J., Zhang, Y., Luo, T., Xiao, Y., and Ma, Z., "Frequency Principle: Fourier Analysis Sheds Light on Deep Neural Networks," Communications in Computational Physics, Vol. 28, No. 5, 2020, pp. 1746–1767. https://doi.org/10.4208/cicp.OA-2020-0085.

[81] Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A., "On the Spectral Bias of Neural Networks," International Conference on Machine Learning, Vol. 97, 2019, pp. 5301–5310.

[82] Li, J., He, S., and Martins, J. R., "Data-driven constraint approach to ensure low-speed performance in transonic aerodynamic shape optimization," Aerospace Science and Technology, Vol. 92, 2019, pp. 536–550. https://doi.org/https://doi.org/10.1016/j.ast.2019.06.008.

[83] Li, J., and Zhang, M., "Adjoint-Free Aerodynamic Shape Optimization of the Common Research Model Wing," American Institute of Aeronautics and Astronautics Journal, Vol. 59, No. 6, 2021, pp. 1990–2000. https://doi.org/10.2514/1.J059921.

[84] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I., "Attention is All You Need," Advances in Neural Information Processing Systems, 2017.

[85] Müller, T., Evans, A., Schied, C., and Keller, A., "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding," ACM Transactions on Graphics, Vol. 41, No. 4, 2022, pp. 102:1–102:15. https://doi.org/10.1145/3528223.3530127.

[86] He, H., Yang, D., Wang, S., Wang, S., and Li, Y., "Road Extraction by Using Atrous Spatial Pyramid Pooling Integrated Encoder-Decoder Network and Structural Similarity Loss," Remote Sensing, Vol. 11, No. 9, 2019, p. 1015.