33RD CONGRESS
OF THE INTERNATIONAL COUNCIL
OF THE AERONAUTICAL SCIENCES
STOCKHOLM, SWEDEN, 4-9 SEPTEMBER, 2022

ICAS 2022 SWEDEN

DEEP LEARNING TECHNIQUES FOR HIGH-DIMENSIONAL SURROGATE-BASED AERODYNAMIC DESIGN

Muhammad Alfiyandy Hariansyah[1,2] and Koji Shimoyama[2]

[1]Department of Aerospace Engineering, Tohoku University, Japan
[2]Institute of Fluid Science, Tohoku University, Japan

Abstract

Utilizing surrogate-based optimization (SBO) to solve high-dimensional aerodynamic design is challenging. The difficult part is to train the surrogate model to achieve sufficient accuracy. Surrogate models with poor accuracy might degrade the optimization performance. Hence, this paper addresses this issue by incorporating a deep convolutional generative adversarial network (DCGAN)-based sampling and a convolutional neural network (CNN)-based geometric filtering in the SBO procedure. In the conventional SBO method, gradient-free population-based optimizers such as genetic algorithm are often used along with Latin hypercube sampling (LHS) in the design of experiment process. To demonstrate the advantages of our proposed methods, we compare their performance with the conventional SBO with LHS by solving a lift-constrained drag minimization of the Common Research Model (CRM) wing with 193 design variables. The results show that the proposed methods could achieve improvements over the conventional SBO method by producing more accurate models. The optimal designs by our methods have lower drags compared to the ones by the conventional method and the baseline CRM wing while maintaining the same lift and ensuring feasibility.

Keywords: Aerodynamic design, CRM wing, DCGAN, CNN, surrogate-based optimization

1. Introduction

Aerodynamic shape optimization (ASO) of the Common Research Model (CRM) wing is a benchmark case defined by the AIAA Aerodynamic Design Optimization Discussion Group (ADODG). Previous research has been performed to solve the problem e.g., [1-5]. The common ground between them is that they relied on local gradient-based optimizers combined with adjoint solvers for computing the gradients. In [2], Lyu et al. argued that the gradient-based optimization is the only hope for handling the large number of design variables required for aerodynamic shape optimization. Nonetheless, population-based methods such as genetic algorithm (GA) are still used since they are easy to implement, gradient-free, robust, and feature a global search that increases the likelihood of finding the global optimum. Oyama et al. [6], for example, performed a transonic wing optimization by directly coupling a CFD solver with an adaptive range GA. The optimization costed 4160 CFD evaluations with 87 design variables. Sasaki et al. [7] also used a similar method to perform an ASO of supersonic wings that costed 4800 CFD evaluations with 72 design variables. Each CFD evaluation in [6] took about 100 minutes of CPU time. The optimization would not have been possible without the presence of parallel computing since it would sequentially have taken more than 9 months to evaluate 4160 design candidates. Efforts have thus been made in the last two decades towards reducing the number of evaluations required by a population-based method. One such effort is to replace the CFD with analytical models and perform the search on them, referred to as a surrogate-based optimization (SBO) [8].

The use of SBO methods in wing design has also been performed previously. Zhang et al. [9] applied an SBO method based on a Kriging model [10] and Expected Improvement (EI) coupled with a combination of GA and local optimizers to the optimization of CRM wing with 39 design variables.

As they explained in the result section, it is still hard to construct the surrogate model with sufficient accuracy in high-dimensional problems. This is in line with the findings by Díaz-Manríquez et al. [11], suggesting that the Kriging model is a good approach to be used only in low dimensionality problems since the accuracy deteriorates as the dimensionality increases. Li et al. [12] performed the ASO of the CRM wing by coupling a Kriging model and a local optimizer called sequential least-squares quadratic programming (SLSQP) algorithm without requiring adjoint solvers. They also proposed a compact wing shape parameterization by deriving global wing mode shapes from the sample wings. Another example includes a low-boom supersonic wing planform optimization by Jim et al. [13] using a Kriging model coupled with a GA and a local optimizer. However, since they only optimized the wing planform, the design variables are only 6 and 11.

Based on the past research we identify that the issues of using global optimizer such as GA in a high-dimensional wing design fall into two primary issues: 1) the GA requires numerous evaluations and 2) the difficulty of constructing an accurate surrogate model in high-dimensional problems. The former can be alleviated by using SBO methods. Recent techniques must be introduced to solve the latter. In this paper, we integrate some deep learning techniques in the SBO method to solve the ASO of CRM wing with high dimensionality (>100 design variables).

First, we train a multilayer perceptron (MLP) as the surrogate model to replace high fidelity CFD analysis in the SBO method. MLP is more efficient than Kriging when it comes to high dimensionality and multiple expensive functions to evaluate. It is since MLP can map inputs to outputs in a single model, unlike Kriging that needs K models for K expensive functions to evaluate. Second, we train a deep convolutional generative adversarial network (DCGAN) [14], as opposed to the Latin hypercube sampling (LHS) to improve the initial sampling quality. Generative adversarial network (GAN) was proposed by Goodfellow et al. [15] to generate new data with the same statistics as the training data. DCGAN is a direct extension of GAN that only features convolutional neural network (CNN) in the architecture. By training the DCGAN using transonic airfoils from the UIUC database, it can produce a set of new airfoils from a set of random latent variables. These new airfoils can then be used as wing sections to produce new geometry of transonic wings. Third, we train a CNN-based geometric filter proposed by Li et al [16] to quickly detect the geometric abnormality applied in the infill sampling process of the SBO procedure.

The first technique is applied to reduce the number of CFD evaluations, while the last two techniques are used to increase the accuracy of the surrogate model by adding meaningful samples with better quality to train with. One can also use DCGAN to reduce the dimensionality of the problem, i.e., performing the search in the low-dimensional latent space instead of high-dimensional original space. By integrating our methods with a GA called NSGA-II [17] and applying it to the ASO of CRM wing, we aim to bridge the advancements in both the aerodynamic shape optimization and evolutionary computation societies.

2. Problem Formulation

In this section, the optimization approach in terms of geometry and formulation are presented. In this problem, we are minimizing the drag while imposing aerodynamic and geometric constraints to ensure the feasibility of the design. We start with the baseline geometry, propose a set of design variables with geometric constraints, and finally summarize with the full optimization problem formulation, which is a slightly modified version of the AIAA ADODG benchmark problem.

2.1 Baseline Geometry

The CRM was proposed by Vassberg et al. [18] with a wing/body/nacelle/pylon/horizontal-tail configuration and the size like that of a Boeing 777. The CRM transonic supercritical wing design was developed with aerodynamic characteristics that are well-behaved and of high performance for configurations with and without the nacelle/pylon group. In this paper, we only used the CRM wing-alone as the baseline without other groups. The CRM wing is already a good performing design to start with. Hence, we do not expect any huge aerodynamic improvement from it.

2.2 Design Variables and Geometric Constraints

The Free Form Deformation (FFD) method implemented in pyGeo [19] is used to parameterize the wing geometry. As the name suggests, the FFD parameterizes the geometry deformation rather than the geometry itself. The baseline geometry is embedded in the FFD volume. By perturbing the FFD volume, we have a great deal of control of the baseline geometry as the embedded object. The FFD volume is created by specifying 192 FFD points that embed the baseline, as shown in Fig. 1. The 192 FFD points are distributed in the 8 spanwise sections, with 24 points in each section. These points are allowed to move only in z-direction, following the direction by the ADODG.

We impose fixed trailing edge (TE) that can be achieved by setting $\Delta z_{TE,upper} = -\Delta z_{TE,lower}$. The changes in z-direction Δz is treated as the design variable. Consequently, the optimizer has control over Δz. The design boundary is [-0.35t, 0.35t], where t is the local thickness of the FFD volume. Fig. 2 shows the FFD boundaries of the third spanwise section from the wing root. We allow twist variations by allowing the leading edge (LE) to vary and fixing the trailing edge (TE) except for the wing root section. To produce practical wings, geometric constraints should be considered.

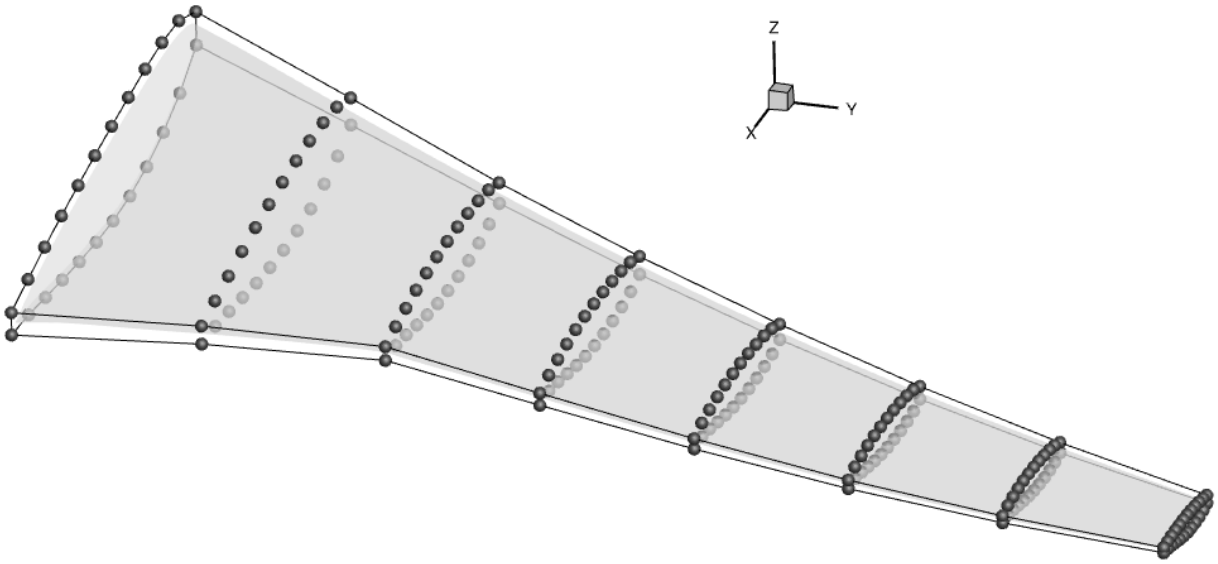

Figure 1 – FFD points embedding the baseline geometry.

To calculate the wing internal volume and the thickness at specific locations can be done in pyGeo. However, it takes around a second to convert a set of design variable (one design candidate) information into the wing surface coordinates and calculate the volume and thicknesses. This is not favorable for a population-based method since geometric constraints are often treated as analytical or fast calculation in millisecond. Instead, we use the FFD volume that is analytically calculated using the convex hull method; the smallest convex set that contains the shape. The thickness constraints are met by setting the design boundaries so that it will not produce unrealistically thin wing sections.

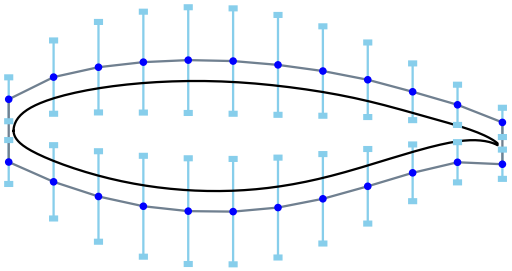

Figure 2 – FFD boundaries of the third spanwise section from the wing root.

2.3 Optimization Problem Formulation

The objective is to minimize the drag coefficient C_D at a fixed lift coefficient $C_L = 0.5$. Fixed lift is achieved by conducting a simple secant method search on the angle of attack α for a given design candidate. In other words, this is handled by the CFD solver, and the optimizer has no control over the design variable α. Another aerodynamic constraint includes the moment coefficient in y-direction C_{My} that must be greater than that of the baseline ($C_{My} \geq C_{My,base}$). Geometric constraint is imposed as explained previously. We impose fixed TE constraints for all wing sections and a fixed LE constraint only for the wing root, to maintain a constant incidence with the fuselage. We thus deal with 1 objective function, 193 design variables, and 12 constraints, summarized in Table 1.

Table 1 – The aerodynamic shape optimization problem formulation.

	Function/variable	Description	Quantity
Minimize	C_D	Drag coefficient	1
With respect to	α	Angle of attack	1
	Δz	FFD control point displacements	192
		Total design variables	193
Subject to	$C_L = 0.5$	Lift coefficient constraint	1
	$C_{My} \geq C_{My,base}$	Moment coefficient constraint	1
	$V \geq 0.8V_{base}$	Minimum FFD volume constraint	1
	$\Delta z_{TE,upper} = -\Delta z_{TE,lower}$	Fixed trailing edge constraints	8
	$\Delta z_{LE,upper} = -\Delta z_{LE,lower}$	Fixed leading edge constraint	1
		Total constraints	12

3. Methodology

In this section, the optimization methodology including the deep learning techniques are discussed. At its core, the SBO method is used in which the MLP-based cheap analytical models, as the surrogates, substitute the high-fidelity CFD, with the aim to reduce the computational burden. We start by explaining the deep learning techniques incorporated in the SBO. We then elaborate the SBO method along with the GA optimizer. Finally, the CFD method to provide the training data for the MLP-based surrogates is explained.

3.1 Deep Learning Techniques

As mentioned previously, the deep learning techniques in this paper are utilized with the aim to solve the issues associated with the curse of dimensionality, i.e., the amount of data that grows exponentially with the dimensionality and the difficulty to construct an accurate surrogate model. It is because the high-dimensional data are often sparse, and we thus need some techniques that efficiently explore the design domain to sample informative data.

3.1.1 Multilayer perceptron

Multilayer perceptron (MLP) is a class of neural network with fully connected layers. The MLP is trained by back-propagation techniques [20] to perform either regression or classification task. In this paper, it is used to perform a regression task: analytically mapping the design variables of the wing candidate to its aerodynamic performances. The training data are provided by using CFD. Hence, a model can be obtained and used for the fast prediction of the aerodynamic performances. The MLP then substitutes the CFD in the optimization process.

3.1.2 DCGAN-based sampling

Generative adversarial network (GAN) proposed by Goodfellow et al. [15] is a type of generative model trained via an adversarial process. Two models are trained simultaneously: a generative model G trained to produce a new sample with the same P_{data} distribution with the training samples and a discriminative model D trained to distinguish whether a sample came from the G model. This is a minimax problem, mathematically expressed as:

$$\min_G \max_D V(D,G) = \boldsymbol{E}_{\boldsymbol{x}\sim P_{data}}[\log D(\boldsymbol{x})] + \boldsymbol{E}_{\boldsymbol{z}\sim P_{\boldsymbol{z}}}[\log(1 - D(G(\boldsymbol{x})))], \quad (1)$$

where $\boldsymbol{x}$ and $\boldsymbol{z}$ are the training dataset and noisy inputs, respectively. By learning the underlying features of P_{data}, G manages to produce similar synthetic data just using noisy inputs. The deep convolutional GAN (DCGAN) is the direct extension of GAN proposed by Radford et al. [14] that utilizes purely convolutional layers in the architecture. The G model takes 100-dimensional noise and performs an upsampling process via deconvolutional layers to produce airfoil z-coordinates. On the other hand, the D model takes the airfoil z-coordinates and performs a downsampling process via convolutional layers to produce a scalar score value that estimates the probability of a sample being produced by G. The architecture is shown in Fig. 3 and the details are listed in Table 4.

In this paper, the idea is to train DCGAN using 77 transonic airfoil data from the UIUC airfoil database, following the normalization and transformation methods by Li et al. [16]. At the end of the training, we can obtain a G model that can produce airfoils that are expected to perform well in the transonic regime. The D model is no longer needed for later use. The airfoils produced by G are referred to as DCGAN airfoils and these are used as wing sections. The wing geometry is represented by 8 wing sections, as in Fig. 4 for the CRM wing. These 8 wing sections are used to perform an inverse FFD procedure described in Algorithm 1. This algorithm finds the corresponding FFD control point z-coordinates using NSGA-II. The resulting FFD points can subsequently be utilized as the design variables for the initial sampling.

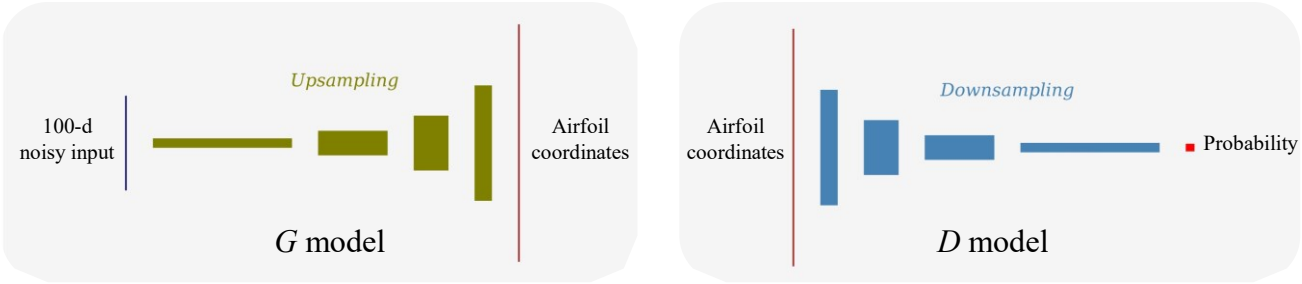

Figure 3 – DCGAN architecture.

Table 2 – Details of DCGAN architecture.

Layers	G model (deconvolutional)		D model (convolutional)	
	Number of kernels	Kernel size	Number of kernels	Kernel size
First layer	80	10	10	10
Second layer	40	10	20	10
Third layer	20	10	40	10
Fourth layer	10	10	80	10
Fifth layer	1	11	1	10

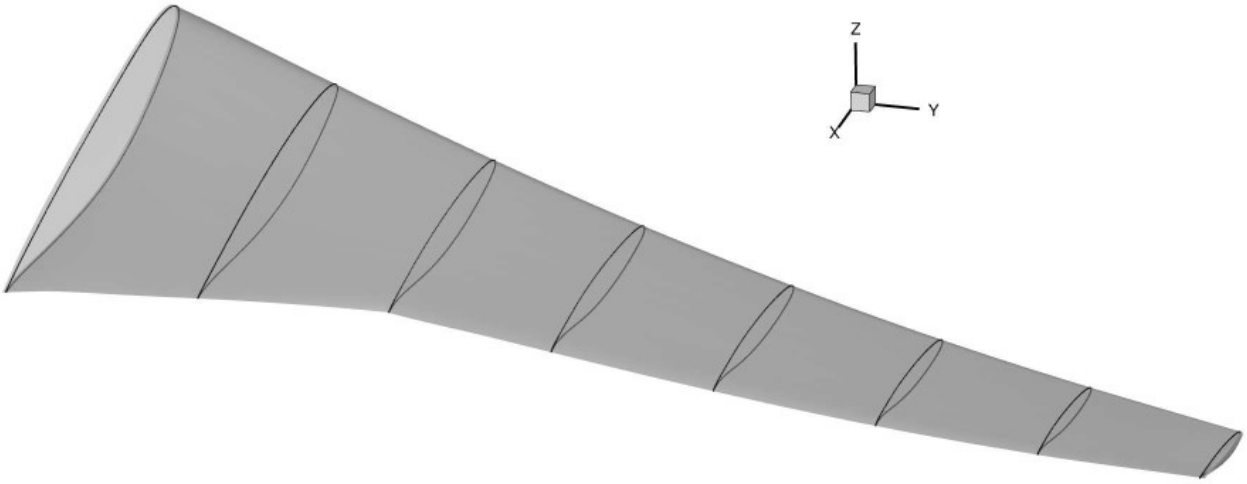

Figure 4 – The eight wing sections of the CRM wing.

Algorithm 1 – The search of the corresponding FFD control points given a set of 8 DCGAN airfoils.

1:	**procedure** INVERSE_FFD($\boldsymbol{z\_t}$)	⇨ $\boldsymbol{z\_t}$ is the target DCGAN airfoil z-coordinates
2:	**for** $i = 1; i \leq 8; i++$ **do**	⇨ loop over the 8 airfoils
3:	$\boldsymbol{z_t} = \text{scale}(\boldsymbol{z_t})$	⇨ scale the airfoils to have the size like baseline
4:	$\boldsymbol{z_t} = \text{reposition}(\boldsymbol{z_t})$	⇨ translate+rotate so the TE position is like baseline
5:	**end for**	
6:	**start optimization**	⇨ perform an optimization using NSGA-II
7:	minimize $\lvert \boldsymbol{z} - \boldsymbol{z\_t} \rvert$	⇨ $\boldsymbol{z}$ is the candidate coordinates after perturbing $\boldsymbol{FFD}$
8:	with respect to $\boldsymbol{FFD}$	⇨ $\boldsymbol{FFD}$ is the candidate FFD control point coordinates
9:	subject to geometric constraint	⇨ the volume constraint
10:	**end optimization**	
11	**return** $\boldsymbol{FFD}$	⇨ return the FFD control point coordinates
12:	**end procedure**	

3.1.3 CNN-based geometric filtering

In the SBO method, the model is updated using infilling samples obtained by performing sub-optimization using GA on the model itself. Low model's accuracy results in the infilling samples with abnormal shapes that subsequently reduce the model's accuracy after the update, resulting in a vicious circle. Hence, there is a need for a geometric filter that quickly detects the shape abnormality without performing any CFD evaluation. In this paper, the geometric filtering method proposed by Li et al. [16] is used. In short, it is a CNN-based geometric filter that is trained using 4000 DCGAN airfoils and 4000 Latin-hypercube-sampling (LHS) airfoils. The LHS airfoils are the resulting wing sections obtained by directly perturbing the FFD control points. The former is given a score of 1, while the latter is given a score of 0. At the end of the training, the geometric filter that can estimate the score *S* of an airfoil is obtained and ready to be integrated with the SBO. The way we integrate it is to impose a score constraint, i.e., $S \geq 0.4$. The architecture of the CNN is identical to the *D* model in the DCGAN.

3.2 Surrogate-Based Optimization and Genetic Algorithm

The flow of the SBO method applied to the aerodynamic shape optimization of CRM wing is illustrated in Fig. 5 and elaborated as follows.

1. Geometric parameterization is done as the first step, in which the geometry is characterized by a set of design variables that determine the shape of a design, i.e., FFD control points.

2. Initial sampling in the design space is done, known as the design of experiment (DoE) step. Two methods are used in this paper: Latin-hypercube-sampling (LHS) [21] and the DCGAN-based sampling. The former is the conventional method that directly generates the perturbations of FFD points, while the latter generates them via Algorithm 1.
3. The objective functions and constraints are calculated using true evaluations, i.e., CFD and analytical functions. These data serve as the ground truth of the database.
4. The so-far obtained solutions are compiled in a design database.
5. MLP-based surrogate model is constructed for the expensive aerodynamic performances, i.e., C_D and C_{My}. This is done via back-propagation technique assisted by Adam optimizer [22].
6. A sub-optimization is performed by NSGA-II [17] implemented in Pymoo [23], in which the aerodynamic performances are evaluated using the obtained MLP-based surrogate model. If the CNN-based geometric filtering is to be used, it is integrated in this sub-optimization. Only one optimized solution is obtained since we only have one objective function.
7. If the believer maximum iteration is met, proceed to Step 9. If not, proceed to Step 8.
8. Append the predicted candidate data by the surrogate model. A believer model is to be trained afterwards. This model is called 'believer' because we append the data using predicted value by the surrogate model. Step 5 – 7 are repeated. In this way, we can efficiently obtain several infilling samples to be parallelly evaluated.
9. If the computational budget is still available, do Step 10. If not available, proceed to Step 11.
10. The new design points are to be evaluated by the true evaluations. Step 3 – 9 are repeated.
11. The best so-far (optimized) solution is obtained.

The sub-optimization in Step 6 is performed by NSGA-II, an elitist global optimizer that has been used widely by the evolutionary computation society. Like any GA, it relies on the bio-inspired operators: selection, crossover, and mutation. Refer to [17] for the details. The parameters for NSGA-II in this paper are listed in Table 3, that came from our best practice. Using this setting, 250 × 100 = 25000 design candidates are analytically evaluated in one sub-optimization.

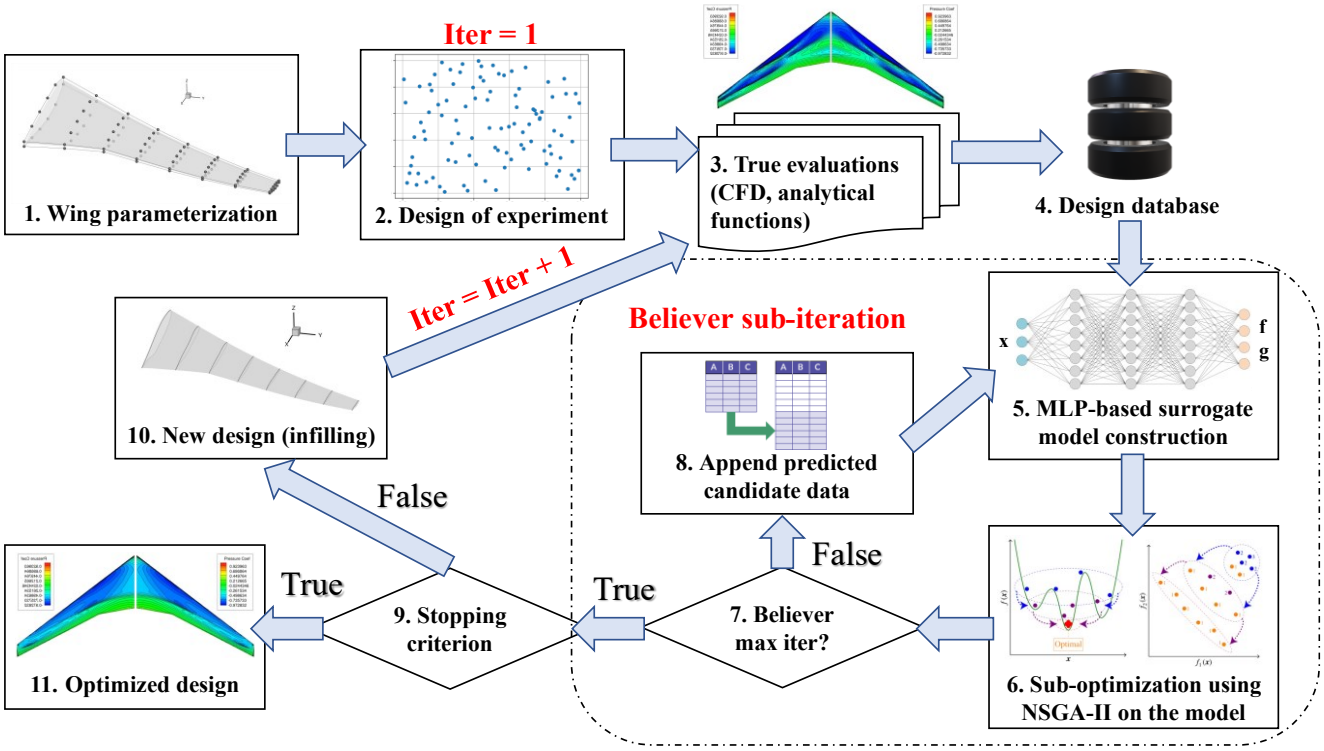

Figure 5 – The SBO method of the ASO of the CRM wing.

Table 3 – Parameters for the NSGA-II.

Population size	100
Max number of generations	250
Crossover	$\eta_c = 15$, rate = 0.9
Mutation	$\eta_m = 15$, rate = 0.01

3.3 Mesh Deformation and Computational Fluid Dynamics

Since the perturbation in the FFD control points modifies the baseline geometry, we must deform the baseline volume mesh for the CFD to solve for the modified geometry. This is more efficient rather than meshing every modified geometry from scratch. The mesh deformation is performed using an efficient analytic inverse-distance method implemented in IDWarp [24]. To provide the initial sampling and infilling data as the ground truth for the surrogates, a finite-volume CFD solver called ADflow is used [25]. ADflow solves the Reynolds-Averaged Navier Stokes (RANS) equations with the Spalart-Allmaras turbulence model iterated with the diagonally dominant alternating-direction implicit scheme (DDADI). For the startup, an approximate Newton-Krylov (ANK) solver [26] is used, combined with the Newton-Krylov (NK) solver for the final stages of convergence. The reference L0 baseline mesh was provided by Lyu et al. [1], and then we coarsened it in stages to obtain L1, L2, L3, and L4 meshes. We then performed a grid convergence study and decided to use the L2 mesh shown in Fig. 6, with approximately 450 thousand cells. The analysis took about 30 minutes on an Intel Xeon Gold 6148 2.4GHz with 40 processors. We only deal with a single-point ASO under the nominal flight condition (Mach 0.85, $Re = 5 \times 10^6$) with an initial angle of attack (AoA) guess of 2°. The CFD solver then performs a secant method to find the corresponding AoA that results in a C_L = 0.5. The allowed maximum iteration of the AoA search is 3.

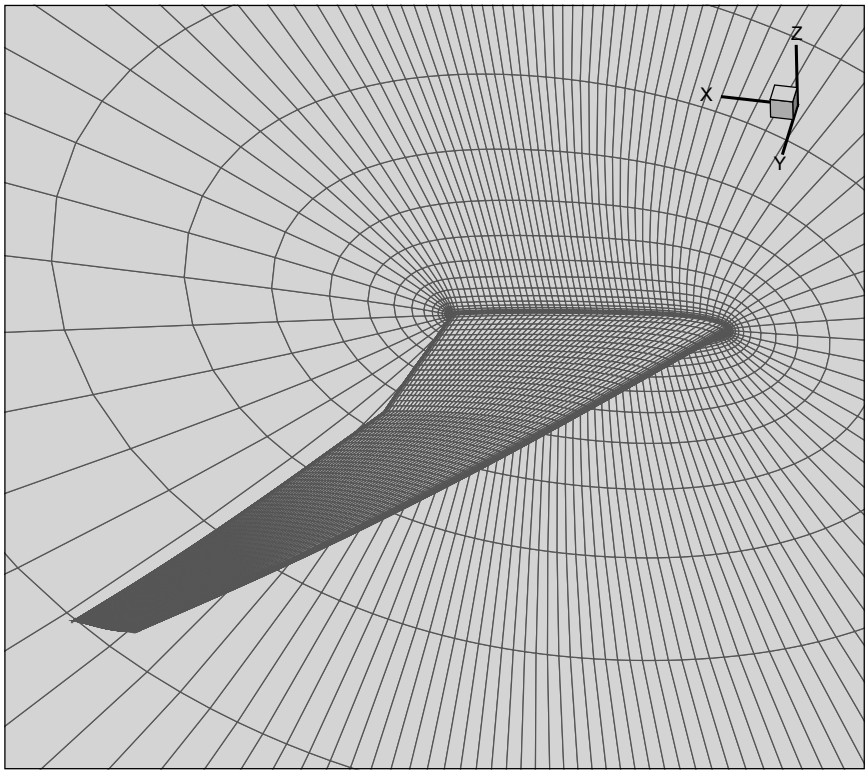

Figure 6 – The baseline mesh with approximately 450 thousand cells.

4. Results and Discussion

In this section, we present and discuss our results: the obtained *G* model's capability to generate synthetic DCGAN airfoils, the CNN-based geometric filter's capability to detect the shape abnormality, the distribution of the initial samples with their CFD comparison, the optimization iteration history, the surrogate model's accuracy, CFD results comparison between the baseline and the best optimized design obtained, and the 3D shock wave visualization.

4.1 DCGAN Generative Model

The DCGAN is trained using 77 transonic airfoils from the UIUC airfoil database. Fig. 7 shows the training airfoils (left), the transformed 99 DCGAN airfoils generated by the *G* model using 99 × 100 noisy inputs (center), and the 99 LHS airfoils obtained by directly perturbing the FFD points. As observed from the training airfoils, they have different thickness and camber characteristics. Prior to training, these airfoils must be normalized first. The normalization technique was shown to be more effective in terms of producing a DCGAN model with better generative capability, refer to [16]. The airfoils from the *G* model must also be denormalized back to be ready as sampling airfoils. These denormalized airfoils are what we refer to as the transformed DCGAN airfoils below, simply referred as DCGAN airfoils. The DCGAN and LHS airfoils below are the examples of the wing sections of the 200 initial samples (before scaling).

It is observed that the DCGAN-based sampling produced smooth airfoils much better than the LHS airfoils that have irregular shapes. However, DCGAN airfoils are dependent with the training airfoils that makes it suffer from poor variability. On the other hand, LHS airfoils offer wide flexibility that covers huge design space.

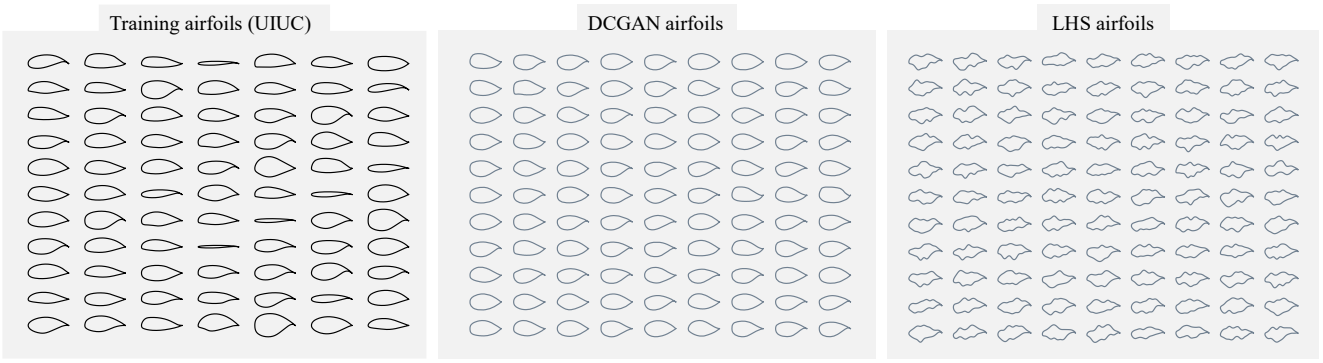

Figure 7 – The UIUC transonic airfoils (left), DCGAN airfoils (center), and LHS airfoils (right).

4.2 CNN-based Geometric Filter

The CNN-based geometric filter is trained using a separate set of 500 DCGAN and 500 LHS samples (not from the initial samples). We refer the former as the samples with realistic shapes and hence given a score of 1, while the latter as the samples with abnormal shapes with a score of 0. The filter maps the design variables (FFD points) to the geometric filter score, quickly detecting whether a given set of design variables will produce abnormal shapes. After training, we apply the filter to our LHS and DCGAN initial samples and the UIUC transonic wings (wings produced by using the UIUC transonic airfoils). The score density distribution is shown in Fig. 8.

In Fig. 8, the capability of the geometric filter to distinguish wings with abnormal sections from the realistic ones is showcased. The geometric filter score of around 0 means that the wing has abnormal airfoil shapes. On the other hand, the value near 1 means it has realistic shapes. To include this filter in the SBO method, we add a geometric filter constraint in the sub-optimization by NSGA-II, that is, $S \geq 0.4$. It is since all realistic shapes from the DCGAN initial samples and the UIUC wings have geometric filter score more than 0.4. This evaluation is analytical and does not consume much of the computational resource.

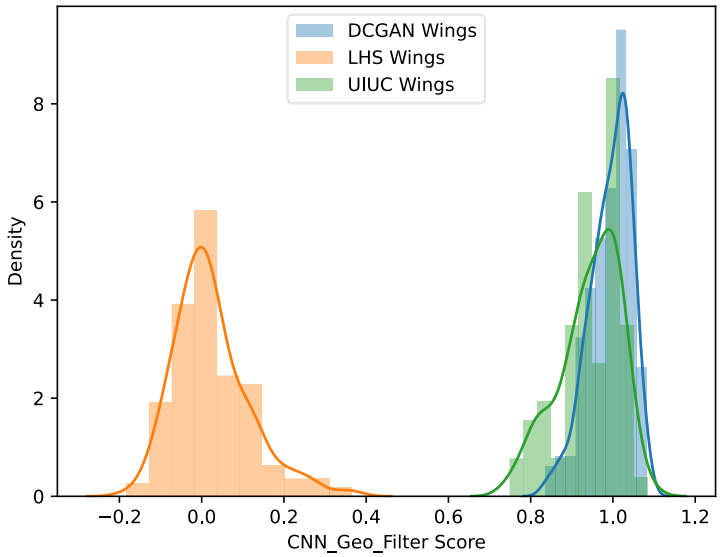

Figure 8 – The CNN-based geometric filter can separate abnormal airfoils from realistic ones.

4.3 DCGAN-based and LHS-based Initial Sampling Comparison

To build a surrogate model, a set of initial samples is needed. We used two types of methods to do it: LHS and DCGAN. Both methods search the initial samples on the design variable space. The initial sample size is 200. They are then evaluated using CFD and analytical functions to provide the ground truth information of objective functions and constraints. We plot the drag coefficient for every sample in Fig. 9. We count one sample as one CFD evaluation.

It can be observed from Fig. 9 that both methods have a distinct distribution. The DCGAN-based method produced samples that have drag coefficients between 260 – 470 counts. While the LHS method's samples have higher drag coefficients that are between 520 – 1250 counts. We will discuss shortly why it is the case from the point of view of the CFD results. Only 3 out of 200 samples are infeasible for the LHS-based method. It is since we employ a constraint handling strategy to deal with the cheap analytical constraints when performing LHS. The constraint handling strategy ensures the samples to meet the cheap analytical constraints, i.e., volume constraint, in the design of experiment (DoE) process. As for the DCGAN-based method, all samples are feasible. In other words, the feasibility rate of the initial samples by both methods is incredibly high.

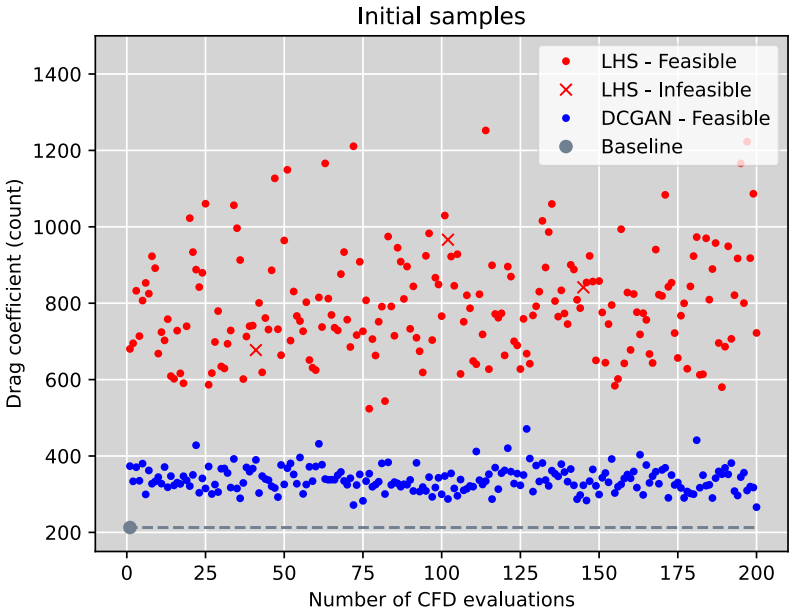

Figure 9 – DCGAN-based and LHS-based sampling have different distribution of initial samples.

The distribution difference comes directly from the way both methods produce their samples. The LHS-based method offers high flexibility by directly varying local shapes of the wing sections. This will introduce bumps and irregular shapes of the wing sections as we have discussed previously. The bumps in the LHS samples make the flow separate faster and result in more intense shock waves or higher drag. It is depicted from Fig. 10 that shows the CFD comparison between one of the samples from both methods. G1S177 of the LHS sample means it comes from the first generation (G1) and is the 177th sample (S177). While G1S105 of the DCGAN sample tells us that it is 105th sample (S105) from the first generation (G1). We randomly select these two samples for comparison.

From Fig. 10, it is observed that the DCGAN sample conforms better to the elliptical normalized lift distribution than that of the LHS sample. The DCGAN wing sections are also smoother compared to the LHS wing sections that results in lower drag. That is why the distribution of the initial samples of the DCGAN is in a lower drag region than that of the LHS initial samples.

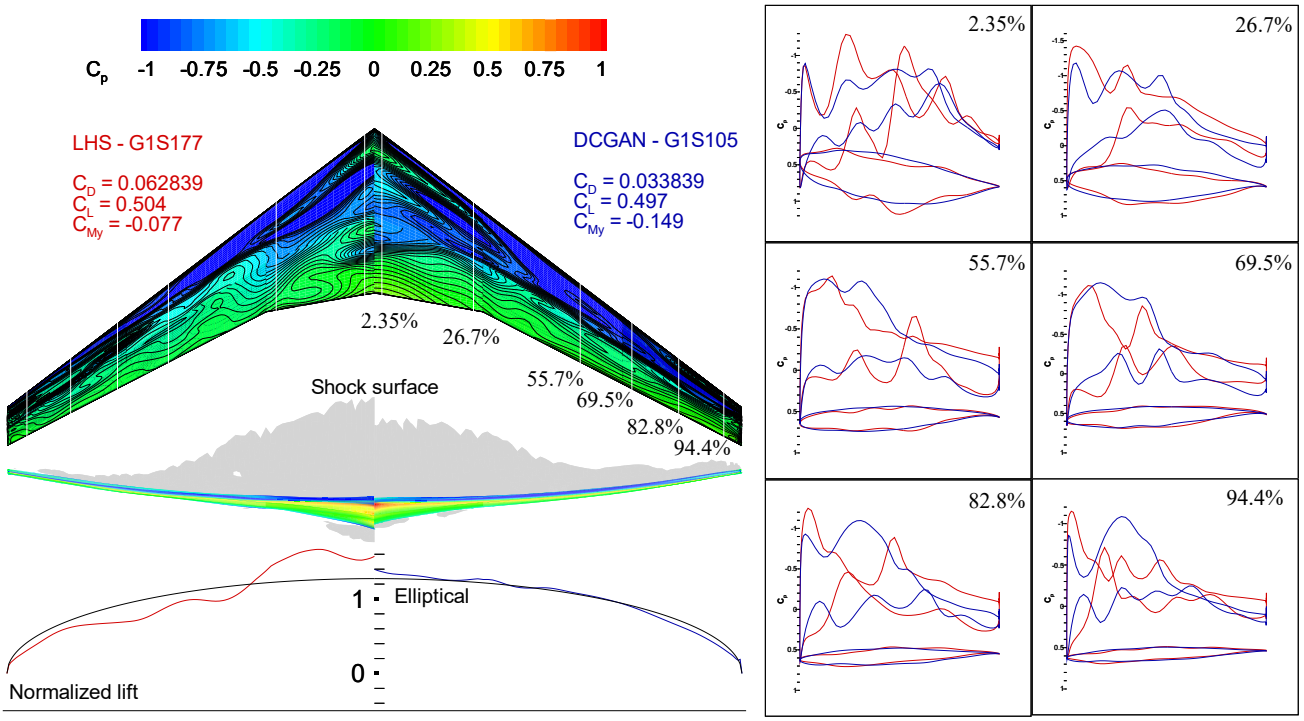

Figure 10 – The CFD comparison between one of the samples of the LHS and DCGAN methods.

4.4 Optimization Iteration History

We performed three SBO methods to study the efficacy of the proposed deep learning techniques. The first method is to build the initial surrogate model using the LHS initial samples and perform the ordinary SBO procedure, we simply call this the LHS method. We do the same thing in the second method but with the DCGAN initial samples to begin with. We call this the DCGAN method. For the third method, we combine both the DGCAN initial samples to build the initial surrogate model and the geometric filter to search for the infilling samples (DCGAN+GF method).

We started with 200 initial samples for each method and added 5 infilling samples per iteration. In other words, since one believer iteration produced one candidate, we conducted 5 believer sub-iterations. For plotting convenience, we define a penalized objective function as follows.

$$\text{Penalized objective (count)} = 10{,}000\times C_D + 1000\times|0.5\text{-}C_L| + 1000\times\left|\min\left(C_M\text{-}C_{M,base},0.0\right)\right| \quad (2)$$

The above equation considers the constraint violation, so it becomes easy to visualize all samples in one figure. We plotted the penalized objective functions for all samples in Fig. 11. It demonstrates the behaviors of each method in performing the optimization.

For the LHS method, it successfully found infilling samples that are better than the initial samples. As for the DCGAN method, it often found solutions that are worse than its initial samples. The reason behind this is that the surrogate model underestimates the objective function (the surrogate model predicts that a sample has a minimum drag, but the CFD proves otherwise). This might be because the training samples (DCGAN initial samples) suffer from poor diversity. This argument is supported by observing the infilling samples of the DCGAN+GF method. At the early optimization iterations, it failed to find solutions that are better than its initial samples. However, the presence of the geometric filter manages to speed up convergence.

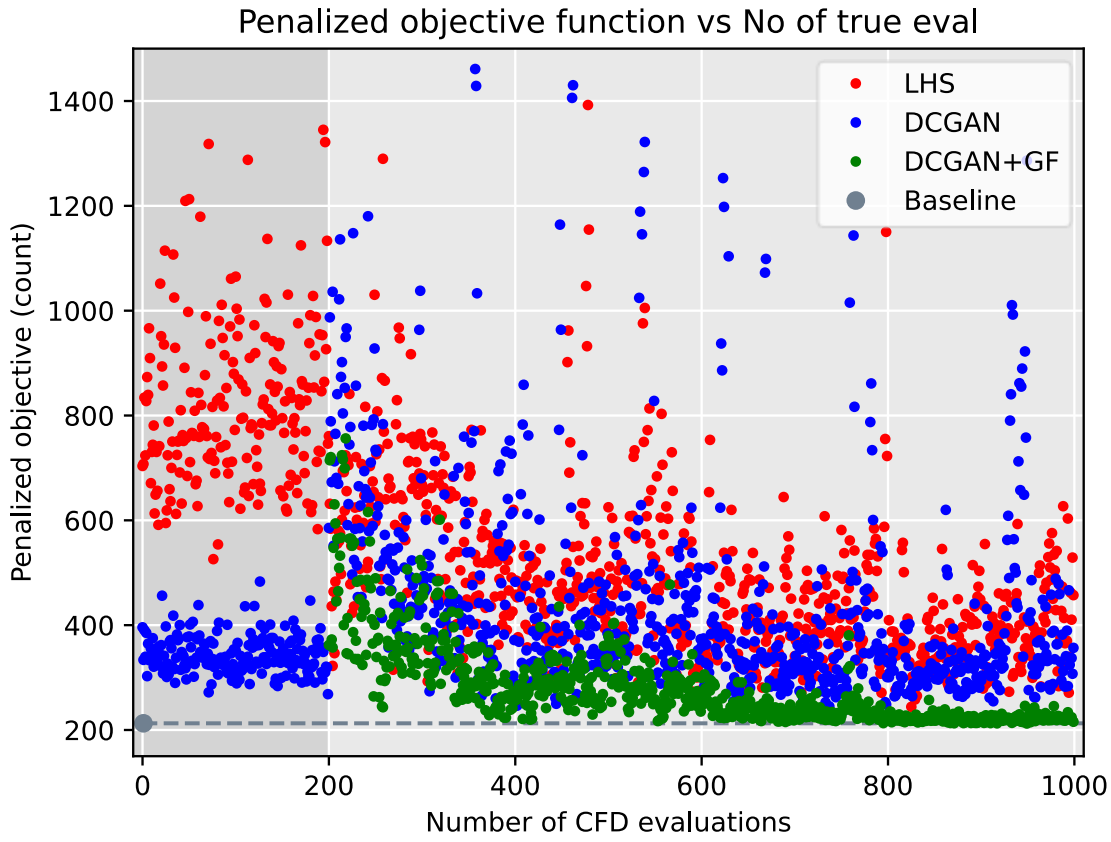

Figure 11 – The penalized objective functions for all samples show different characteristics of the optimization methods. The DCGAN+GF is the best among others.

To further compare the performance of the three methods, we plotted the optimization iteration history, excluding the initial samples, in Fig. 12. In that figure, we traced the history of the minimum feasible drag coefficient as a function of the number of CFD evaluations. At the second iteration of the LHS method, it could find a solution with an extreme improvement from its initial samples. Our best explanation to that is because we included the baseline data to the initial samples of the GA sub-optimization. However, no huge improvement is obtained afterwards. For the DCGAN method, it steadily found better solutions as the iteration increased. It started to find a better solution than the LHS method at the 310th CFD evaluation. The DCGAN+GF method converged faster with better solutions among the three methods. It converged near the baseline, or slightly better with only 1 drag count improvement. This demonstrates the difficulty level of the ASO of the CRM wing. Lyu et al. [2], found a solution with a drag coefficient that is 16 counts better than the baseline via an adjoint-based optimization with 720 FFD points and 800 iterations. Li et al. [12] could only find a solution with a drag coefficient less than 2 counts better using both adjoint-based and adjoint-free methods + gradient-based optimizer with 192 FFD points and 1000 iterations.

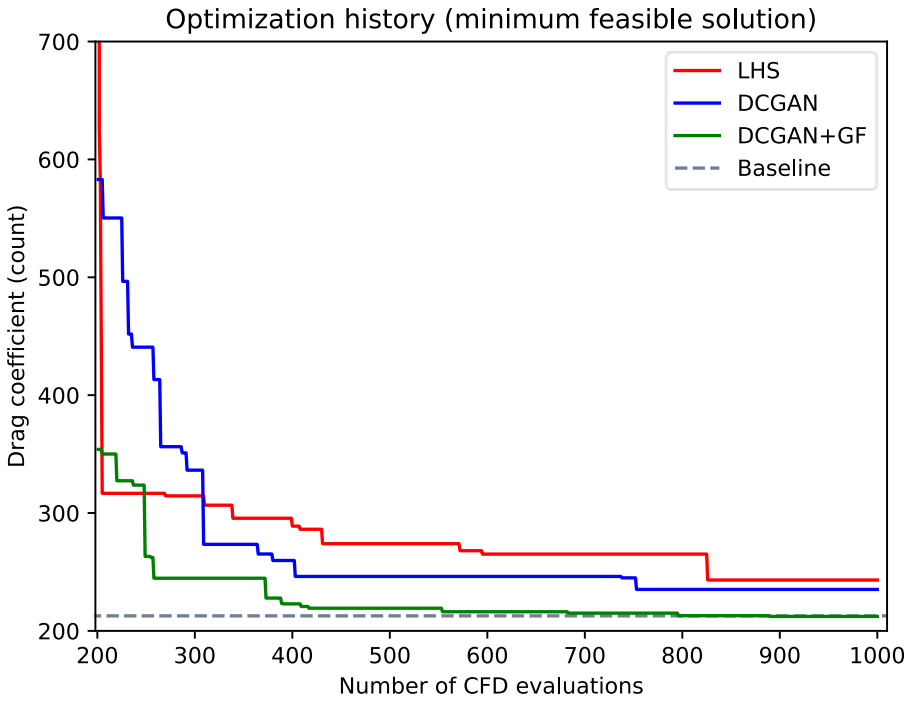

Figure 12 – The minimum feasible drag coefficient history.

4.5 MLP-based Surrogate Model's Accuracy

The performance of the SBO method relies on the accuracy of the surrogate model. Hence, it is important to keep track of it as the iteration increases. We calculate the root mean square error (RMSE) for the modeled function: drag and moment coefficient. The RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2} \tag{3}$$

where y_i is the ground truth by CFD, $\hat{y}_i$ is the predicted value by the model, and *N* is the number of samples. We calculate the RMSE at each main iteration after we evaluate the new design candidates using CFD. *N* is five, since we add five infilling samples per iteration. The RMSE measures the distance between the predicted value by the model and the ground truth by the CFD. In other words, it measures the model's accuracy. We plot the RMSE for the drag and moment coefficients as the iteration increases for the three methods in Fig. 13.

From Fig. 13, the history of the model's accuracy is demonstrated. By analyzing the model's accuracy, we could comprehend each method's optimization performance. For the LHS method, there is no clear trend of the RMSE that is reflected in its optimization performance that shows little improvement after the second iteration (the 202nd CFD evaluation). For the DCGAN method, it could find lower RMSE compared to the LHS method. However, it has several spikes at certain iterations, demonstrating low model's accuracy. At these iterations, the model underestimates the data, that is why we could observe data that are much worse than its DCGAN initial samples. It indicates that the model has little information of the design candidates, caused by poor diversity of the DCGAN initial samples. However, it managed to find a better solution than that of the LHS method. The drawback of the DCGAN method is alleviated by the geometric filter (GF), shown in the RMSE history of the DCGAN+GF method. The GF successfully filtered out irregular infilling samples with low performance. This could increase the model's accuracy (low RMSE) and speed up the convergence. The selection of the GF score criterion might affect the performance of the DCGAN+GF method since the GF imposes constraints directly in the design space and might hinder the findings of new designs. This should be carefully studied in the future research.

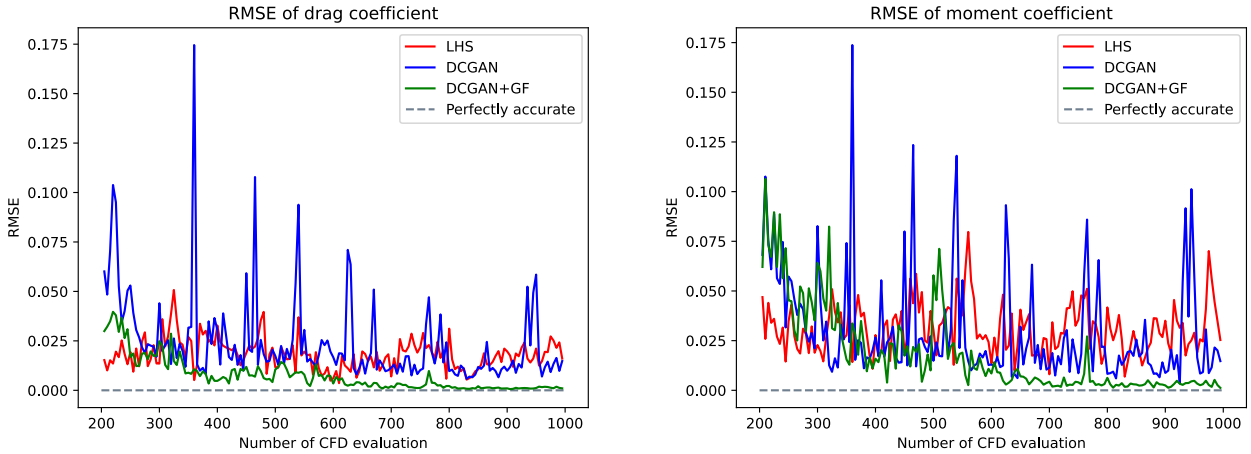

Figure 13 – The RMSE of the drag and moment coefficient.

4.6 The Optimized Designs

The true evaluations for the best designs found by the three methods along with the baseline are summarized in Table 4. With the same CFD budget, the DCGAN method could find a solution that has a drag coefficient 8 counts less than the one found by the LHS method. The integration of the geometric filter (DCGAN+GF) could give 23 counts improvement from the one found by the DCGAN (only) method with 0.5 count improvement (G139S5) from the baseline. We presented another design alternative (DCGAN+GF G136S2) that has 1 drag count improvement from the baseline. However, this design violates the moment constraint by 0.001. From the results, it is sufficient to demonstrate the advantage of using DCGAN and geometric filtering techniques in the SBO method over the standard LHS method.

All the optimized designs (except the last one) meet the constraints. The CFD solver could satisfy the lift constraint by only 3 iterations with 0.001 tolerance. It is noted that we used different definitions of the moment and volume constraints defined by the ADODG. Our moment constraint is less strict ($C_{My} \geq C_{My,base}$), compared to the ADODG formulation ($C_{My} \geq 0.170$). Thus, the solutions by the DCGAN and the DCGAN+GF meet our constraints criteria but not the ADODG's. As for the volume constraint, we used the FFD volume criterion ($V_{FFD} \geq 0.8\ V_{FFD,base}$), as opposed to wing internal volume since it is more expensive to evaluate the latter than the former. However, we found that there is a high correlation between the FFD volume and the wing internal volume. In fact, our best designs only differ by a small amount of wing internal volume compared to the baseline.

To improve the solutions, one might increase the computational budget (more iterations) or reformulate the problem by using more FFD points or greater design space. Although the latter might introduce more complexities as well as difficulties in training the surrogate models.

	Baseline	LHS	DCGAN	DCGAN+GF	DCGAN+GF (lowest drag)
Design	-	G127S1	G112S3	G139S5	G136S2
C_D (counts)	212.745	243.160	235.179	212.261	211.788
C_L	0.500	0.499	0.500	0.500	0.500
C_{My}	-0.181	-0.168	-0.174	-0.176	-0.182
AoA	2.212°	2.286°	2.267°	2.207°	2.217°
FFD Volume	0.689	0.722	0.709	0.700	0.687
Wing Volume	0.231	0.232	0.232	0.230	0.230

Table 4 – Best designs true evaluations comparison.

The CFD results for the baseline and the ones by the DCGAN+GF method are presented in Fig. 14 and 15. Both solutions have quite similar wing sections. The shock surface of the DCGAN+GF is less intense in the area between 0% – 45%, but not in the area between 45% – 90% of the wing. Some pressure oscillations are still spotted for the DCGAN+GF solution, indicating that there are still tiny bumps on its surface. They all also show quite similar lift distribution profile. However, both solutions by the DCGAN+GF have lower drag compared to the baseline by 0.5 – 1 drag count. To explain this, the 3D shock regions will be carefully observed in the section 4.7.

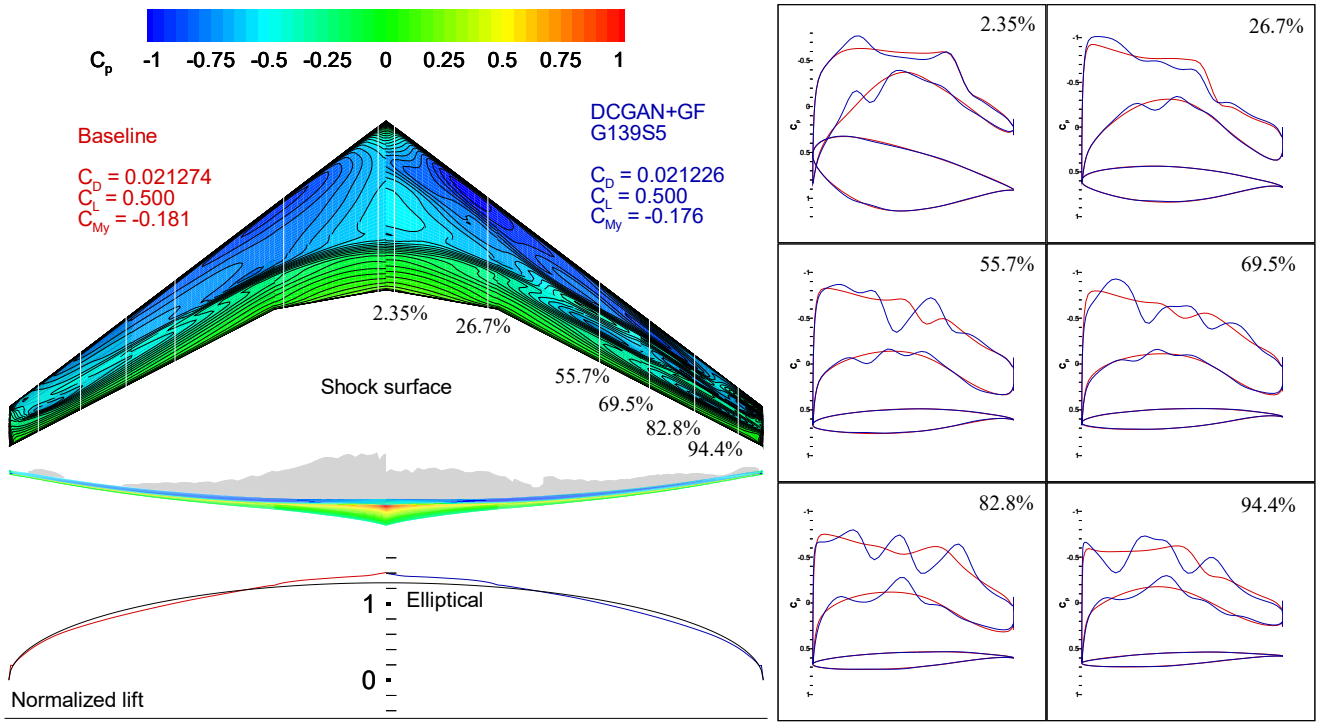

Figure 14 – CFD comparison between the baseline and DCGAN+GF G139S5.

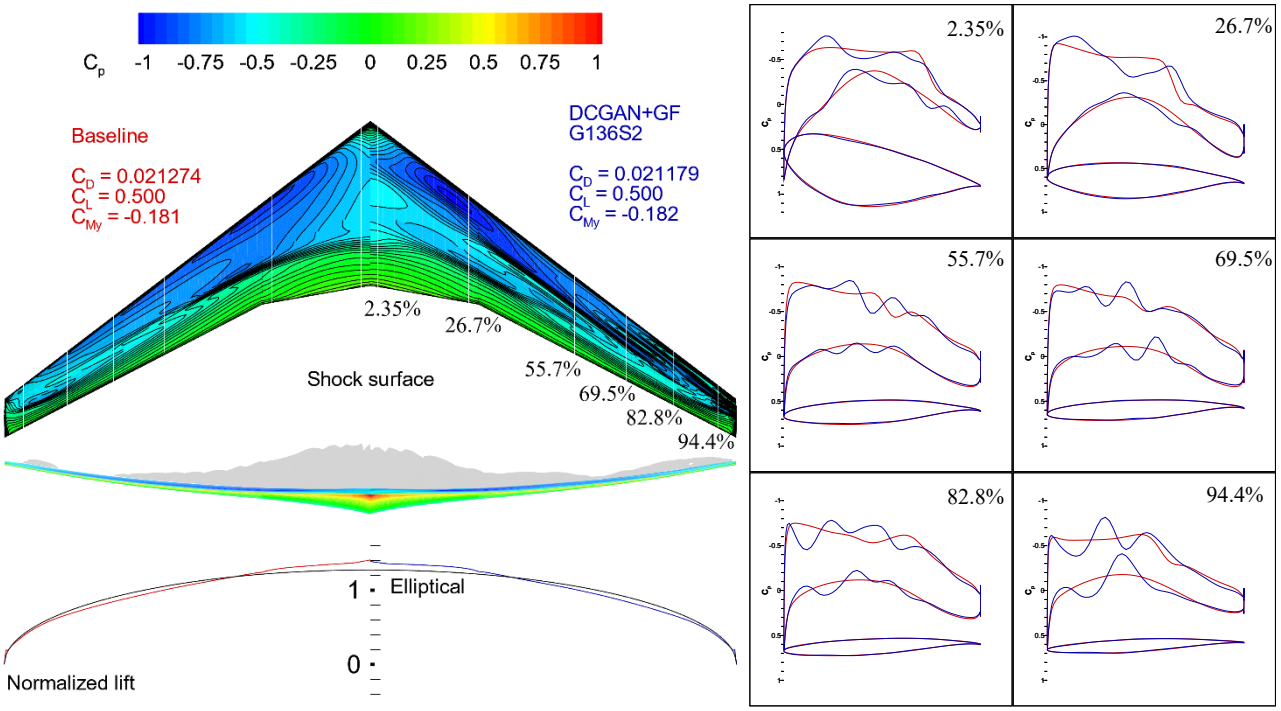

Figure 15 – CFD comparison between the baseline and DCGAN+GF G136S2.

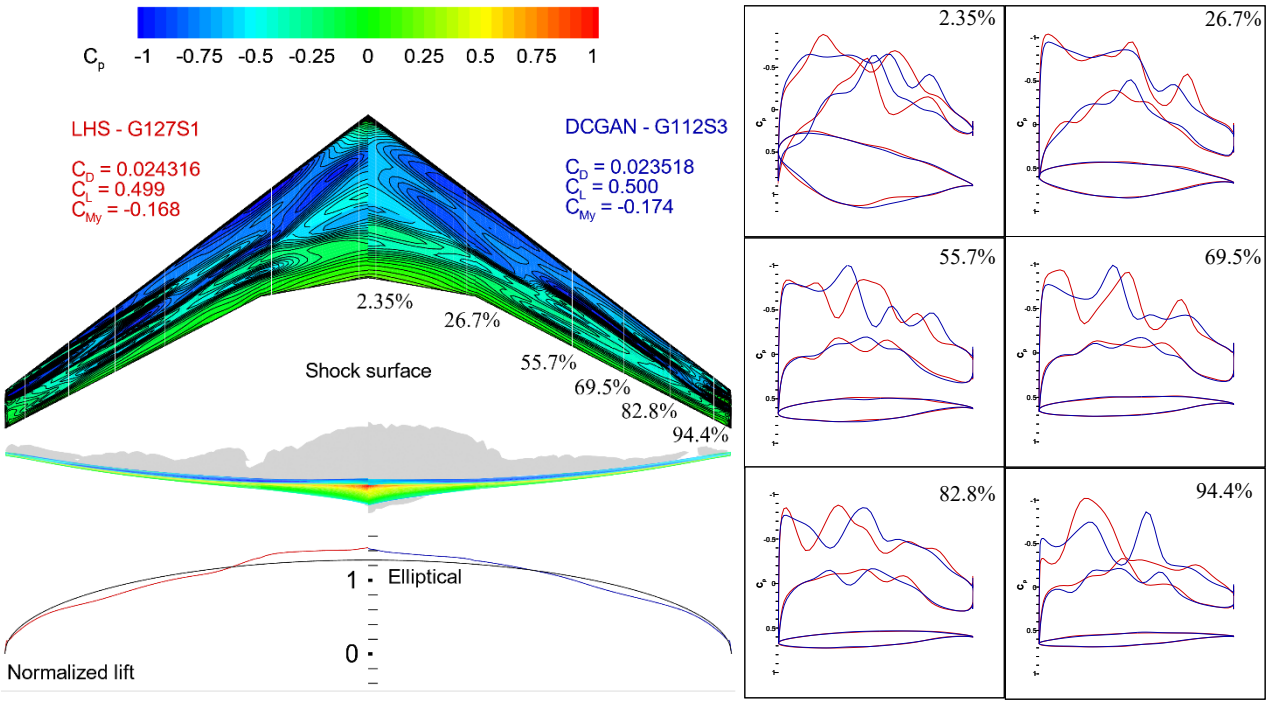

Figure 16 – CFD comparison between the solutions by the LHS and DCGAN methods

The CFD results for the solutions found by the LHS and DCGAN methods are given in Fig. 16. Pressure oscillations and bumps are easily observed for both solutions, especially the one by the LHS. These result in more severe shock and higher drag. By comparing the optimized designs by each method with its initial samples in Fig. 10, it is clear that improvements in drag coefficient (less severe shocks) could be achieved by smoothing the surface of wing sections and conforming to the elliptical normalized lift distribution. In this problem, we did not directly vary wing twists. Instead, we allowed wing twists by fixing the TE and freeing the LE, except for the wing root section (both are fixed). In this way, variations in wing twists could be achieved by vertical shearing of the wing sections. The results show little difference in wing twists compared to the baseline. Independent wing twists should be considered to increase the chance of finding new designs with better performance.

4.7 3D Shock Visualization

In Fig. 17, the 3D shock regions are plotted for the baseline and the obtained optimized designs. The shock regions are visualized as the transparent grey regions on the suction side of the wing. As opposed to the baseline that has a single shock region, the optimized designs clearly show bifurcated shock regions. The DCGAN+GF G139S5 and G136S2 seem to have a shock region with Lambda-configuration [27]. In other words, rather than having a single shock region like the baseline, they exhibit two shock regions at the wing root. The two shock regions seem to coalesce near the wing tip, forming a λ-structure. It is noted that the obtained optimized designs exhibit a completely different shock configuration, yet they show a competitive drag compared to the baseline.

The formation of the double shock regions on the optimized designs is attributed to a small curvature of their suction side profiles at $0.2 < x < 0.5$, see Fig. 18. The small curvature produces a flow deceleration and creates a shock to terminate the supersonic region at the LE. At the same time, a larger curvature at $0.5 < x < 0.8$ causes a flow acceleration and formation of another supersonic region that will later be terminated by the second shock. The curvature $\kappa(x)$ is as follows,

$$\kappa(x) = \frac{d^2z}{dx^2} \cdot \left(1 + \left(\frac{dz}{dx}\right)^2\right)^{-1.5}, \tag{4}$$

where z is the airfoil *z*-coordinates, and x is the normalized *x*-coordinates.

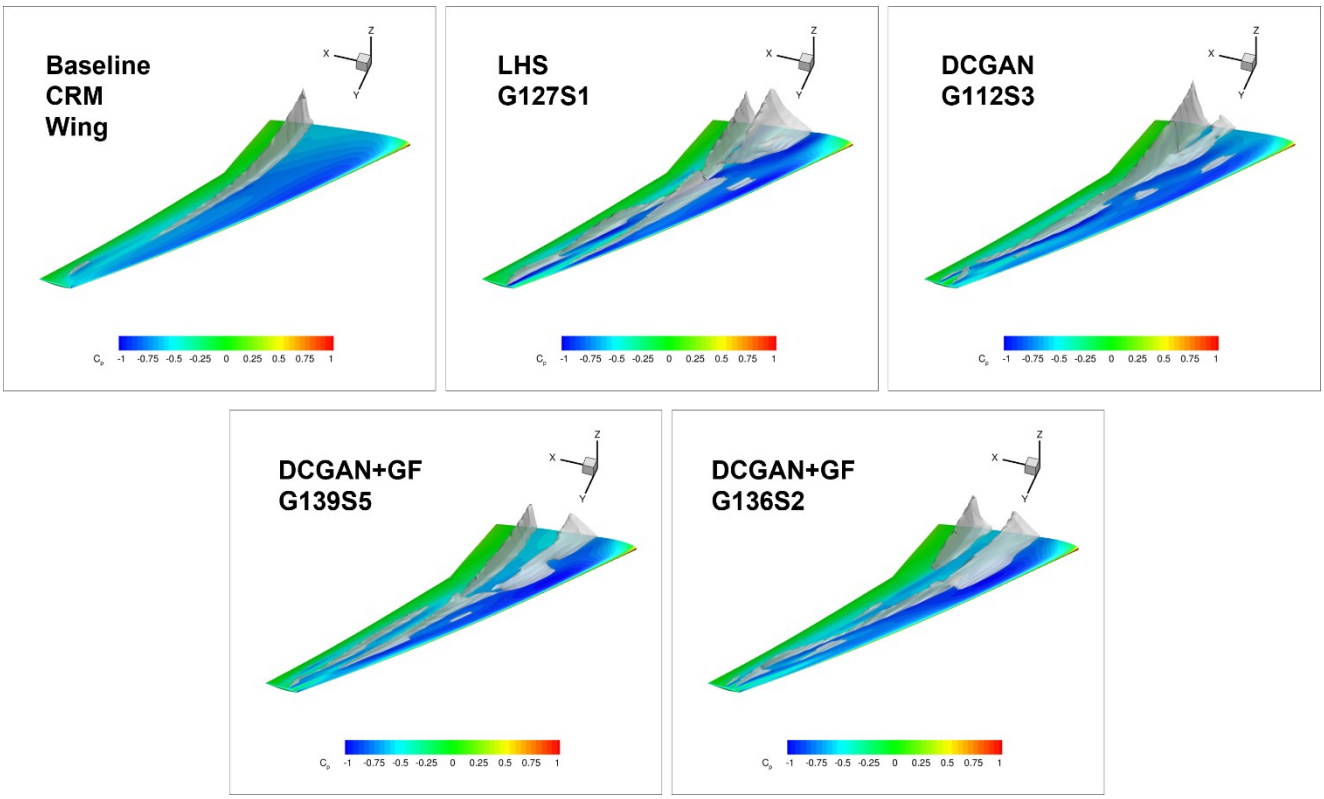

Figure 17 – 3D shock region of the baseline and the obtained optimized designs.

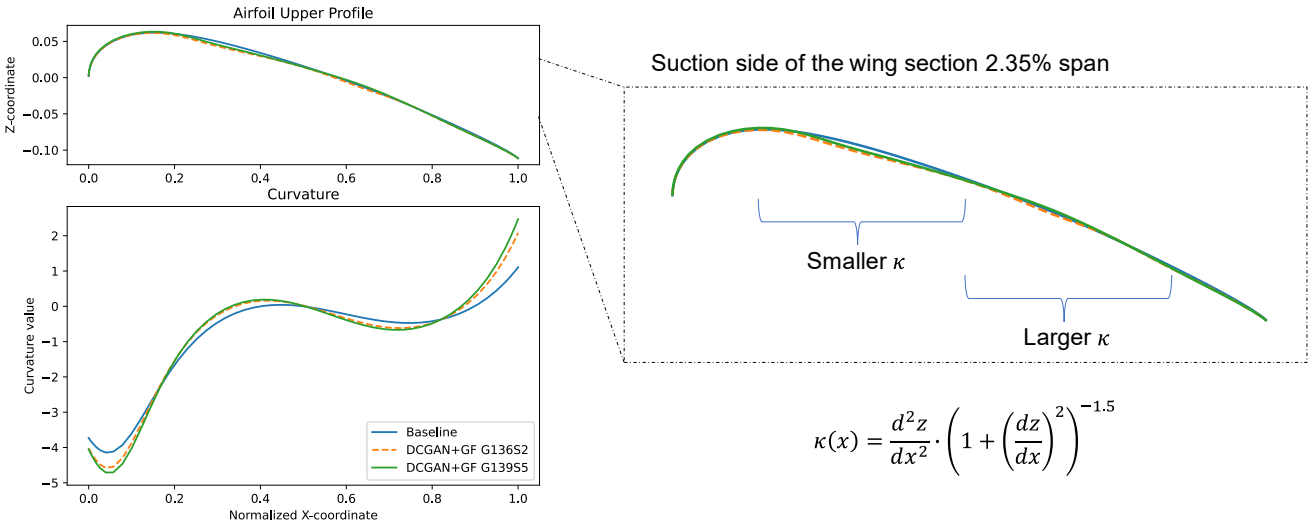

Figure 18 – The optimized designs and the baseline curvatures of the wing sections 2.35% span.

5. Concluding Remarks

- The SBO methods rely on the accuracy of the surrogate model. In high-dimensional problems, it is difficult to build an accurate surrogate model. Hence, we introduced two recent deep learning techniques: DCGAN-based sampling and CNN-based geometric filter.
- The DCGAN-based sampling is used to produce synthetic wing designs for the initial samples.
- The DCGAN method could find a solution with a drag coefficient that is 8 counts less than the one found by the conventional LHS method.

- The geometric filtering strategy could filter out irregular shapes that have poor performance. The GF combined with DCGAN-based sampling shows the best performance among the three methods with a drag coefficient that is 23 counts less than the DCGAN (only) method. This better performance came from a more accurate model demonstrated in the history of RMSE of predictions at every iteration.
- The optimized designs found by the DCGAN+GF method have a lower drag compared to the baseline by 0.5 – 1 count. They exhibit double bifurcated shock regions near the wing root, as opposed to the baseline that has a single shock region.
- Although no huge drag improvement from the baseline was obtained, it is still sufficient to conclude that the proposed deep learning methods could improve the performance of the conventional SBO method (with LHS).
- To improve the obtained optimized designs, one might increase the computational budget (more iterations) or reformulate the problem to include more FFD control points with greater design space, although this will introduce more complexities and difficulties in achieving sufficient accuracy of the surrogate models.

Acknowledgements

This work was supported by the IFS Graduate Student Overseas Presentation Award and partially supported by MEXT as “Program for Promoting Researches on the Supercomputer Fugaku” (Leading research on innovative aircraft design technologies to replace flight test). This work was also supported in part by the Grant-in-Aid for Scientific Research (A) No. 22H00184 administrated by the Japan Society for the Promotion of Science (JSPS).

Contact

If you have any questions, feel free to contact the author to discuss anything related to the contents at muhammad.alfiyandy.hariansyah.s8@dc.tohoku.ac.jp.

Copyright Statement

References

[1] Lyu Z, Xu Z and Martins J R R A. Aerodynamic shape optimization investigations of the Common Research Model wing benchmark. *AIAA Journal*, Vol. 53, No. 4, pp 968-985, 2015.

[2] Lyu Z, Xu Z and Martins J R R A. Benchmarking optimization algorithms for wing aerodynamic design optimization. *Proceedings of the 8th International Conference on CFD*, Chengdu, Sichuan, China, 2014.

[3] Telidetzki K, Osusky L, and Zingg D W. Application of Jetstream to a suite of aerodynamic shape optimization problems. *52nd Aerospace Sciences Meeting*, 2014.

[4] Lee C, Koo D, Telidetzki K, Buckley H, Gagnon H and Zingg D W. Aerodynamic shape optimization of benchmark problems using Jetstream. *53rd Aerospace Sciences Meeting*, 2015.

[5] Bisson F and Nadarajah S. Adjoint-based aerodynamic optimization of benchmark problems. *53rd Aerospace Sciences Meeting*, 2015.

[6] Oyama A, Obayashi S and Nakamura T. Real-coded adaptive range genetic algorithm applied to transonic wing optimization. *Parallel Problem Solving from Nature PPSN VI*, pp 712-721, 2000.

[7] Sasaki D, Morikawa M, Obayashi S and Nakahashi K., Aerodynamic shape optimization of supersonic wings by adaptive range multiobjective genetic algorithms. *Evolutionary Multi-Criterion Optimization*, pp 639-652, 2001.

[8] Queipo N V, Haftka R T, Shyy W, Goel T, Vaidyanathan R and Tucker P K. Surrogate-based analysis and optimization. *Progress in Aerospace Sciences*, Vol. 41, Issue 1, pp 1-28, 2005.

[9] Zhang Y, Han Z H and Leifsson L. Surrogate-based optimization applied to benchmark aerodynamic design problems. *AIAA Aviation Forum*, Denver, CO, 2017.

[10] Krige D G. A statistical approach to some basic mine valuation problems on the Witwatersrand. *Journal of the Chemical, Metallurgical and Mining Society of South Africa*, Vol. 52, No. 6, pp 119-139, 1951.

[11]Díaz-Manríquez A, Toscano-Pulido G and Gómez-Flores W. On the selection of surrogate models in evolutionary optimization algorithms. *IEEE Congress of Evolutionary Computation* (CEC), pp 2155-2162, 2011.

[12]Li J and Zhang M. Adjoint-free aerodynamic shape optimization of the common research model wing. *AIAA Journal*, Vol. 59, No. 6, 2021.Jim T M S, Faza G A, Palar P S and Shimoyama K. Bayesian optimization of a low-boom supersonic wing planform. *AIAA Journal*, Vol. 59, No. 11, 2021

[13]Radford A, Metz L and Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks. 2016.

[14]Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A and Bengio Y. Generative adversarial nets. *Advances in Neural Information Processing Systems*, Vol. 27, pp 2672-2680, 2014.

[15]Li J, Zhang M, Martins J R R A and Shu C. Efficient aerodynamic shape optimization with deep-learning-based geometric filtering. *AIAA Journal*, Vol. 58, No. 10, pp 4243-4259, 2020.

[16]Deb K, Pratap A, Agarwal S and Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, pp 182-197, 2002.

[17]Vassberg J C, DeHaan M A, Rivers S M and Wahls R A. Development of a common research model for applied CFD validation studies. *26th AIAA Applied Aerodynamics Conference*, 2008.

[18]Kenway G K, Kennedy G J and Martins J R R A. A CAD-free approach to high-fidelity aerostructural optimization. *Proceedings of the 13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Fort Worth, TX, 2010.

[19]Rumelhart D E, Hinton G E and Williams R J. Learning representations by back-propagating errors. *Nature*, Vol. 323, No. 9, pp 533-536, 1986.

[20]McKay M D, Beckman R J and Conover W J. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, Vol. 21, No. 2, pp. 239-245, 1979.

[21]Kingma D P and Ba J L. Adam: a method for stochastic optimization. *3rd International Conference for Learning Representations*, San Diego, 2015.

[22]Blank J and Deb K. Pymoo: multi-objective optimization in Python. *IEEE Access*, Vol. 8, pp 89497-89509, 2020.

[23]Secco N R, Kenway G K W, He P, Mader C A and Martins J R R A. Efficient mesh generation and deformation for aerodynamic shape optimization. *AIAA Journal*, Vol. 59, No. 4, 2021.

[24]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*, Vol. 17, No. 9, pp 508-527, 2020.

[25]Yildirim A, Kenway G K W, Mader C A and Martins J R R A. A Jacobian-free approximate Newton-Krylov startup strategy for RANS simulations. *Journal of Computational Physics*, Vol. 397, 2019.

[26]Kuzmin, A. On the Lambda-Shock Formation on ONERA M6 Wing. *International Journal of Applied Engineering Research*, Vol. 9, No. 20, pp. 7029 – 7038, 2014.