Multi-Discriminator cGAN for Airfoil Design*

Tristan Brigham
Department of Computer Science
Yale University
tristan.brigham@yale.edu

Eugene Han
Department of Statistics
Yale University
e.han@yale.edu

Elder Veliz
Department of Statistics
Yale University
elder.veliz@yale.edu

1 Introduction

Airfoils play a crucial role in the design process of aerospace vehicles, including airplanes, helicopters, and even automobiles—where aerodynamic efficiency and performance are paramount. Yet, traditional airfoil design methods face limitations transitioning from conceptual designs to tangible applications, which result in extended design cycles and compromised performance in real-world implementations. Recent work in computational design, adopting optimization algorithms, density approaches, and Conditional Variational Autoencoders (cVAEs), while innovative, struggles to output airfoils that generalize well beyond human-provided designs and abide by physical constraints.

Our research proposes the use of Conditional Generative Adversarial Networks (cGANs) as an alternative framework for airfoil shape design. Unlike other models like cVAEs, which focus primarily on minimizing the reconstruction and latent losses, cGANs undergo adversarial training mechanisms that can directly align generated designs with physical and performance constraints through a discriminative feedback loop. Our research aims to evaluate the efficacy of cGANs in overcoming the generalization limitations faced by existing airfoil designs methods and optimizing key aerodynamic performance metrics (namely, lift, momentum, and drag coefficients) under simulated real-world constraints, thereby establishing an effective, physics-informed design process that may be adapted or further refined for more advanced manufacturing techniques.

2 Related Work

Computational methods have made great strides in addressing both geometric and performance-related challenges within the airfoil design process. This section reviews key developments in the field and identifies gaps that our research aims to fill.

Almasri et al. (2022) (1) introduced the DL-AM-TO model, a cGAN integrating deep learning with topological optimization for additive manufacturing. DL-AM-TO simultaneously handles mechanical and geometric constraints in its 2D design process, unlike previous methods which handle these constraints individually. The DL-AM-TO framework, from which we draw inspiration, uses a generator conditioned on specific mechanical (e.g., boundaries, loads, and volume) and geometric constraints (e.g., thickness, length, and overhang). Its multi-network discriminator consists of a traditional adversarial discriminator and four geometric discriminators. While Almasri et al. (2022) address general 2D designs, they do not specifically optimize for the aerodynamic requirements of airfoil designs and do not directly integrate aerodynamic performance into the generative modeling process—underscoring a gap for the targeted generation of airfoil designs that are both geometrically sound and optimal for specific aerodynamic metrics.

Yonekura and Suzuki (2021) (7) used cVAEs for airfoil design, conditioning the generation process on specific aerodynamic metrics like lift coefficients. Their model uses an encoder to compresses the input airfoil shapes (2D curves discretized as vectors of coordinates) and associated performance indices into a latent space and a decoder that generates new shapes from this latent representation, conditioned on desired aerodynamic properties. The model was trained using the NACA four-digit airfoil set with lift values calculated using XFOIL. Though efficient for optimizing airfoil

*Links to GitHub repository and video explanation.

shape generation, we hypothesized that a cGAN model could be a promising alternative. cVAE training minimizes reconstruction loss and the KL divergence *without* the adversarial component of cGANs, which suggests cVAEs may struggle to generalize beyond training set conditions to the level achievable by cGANs, as suggested by Almasri et al. (2022).

Shu et al. (2020) (6) use traditional GANs to automate 3D aircraft model designs by using physics-based validation to enhance the functionality of the model's generated designs. Their research uses an iterative generate-evaluate-update cycle where generated designs are assessed via a fluid dynamics simulation evaluating real-world aerodynamic performance. While Shu et al. focus on 3D aircraft models, their approach emphasizes the potential of combining GANs with performance validations to optimize the overall aircraft design process, which we extend to 2D airfoil shapes.

3 Approach

3.1 Data

Initially, we planned to use various datasets (namely, the Department of Energy and ShapeNet airfoil datasets); however, we found that these datasets had concerning inconsistencies in data handling and results evaluation. Thus, we focused on data from the UIUC Airfoil Coordinates Database (5), which consists of over one thousand point clouds of effective airfoil designs used in industry.

We standardized the point clouds to ensure each airfoil was represented by exactly 50 points on the upper half of the surface and 50 points on the lower half of the surface, and all airfoils were normalized to maintain a unit chord length in the x-direction. We excluded any airfoils with coefficient values exceeding an absolute value of 2 to remove outliers.

3.2 Model Architecture[2]

3.2.1 Generator

The cGAN's generator takes a 512-point vector of normally distributed noise concatenated with the target aerodynamic coefficients desired for the airfoil. It outputs a 100-point cloud representing an airfoil designed to match the specified input aerodynamic coefficients.

Since we trained the generator with order invariance, output airfoils were first reordered clockwise to match conventional configurations. To compensate for the generator's output of only 100 points—far fewer than the infinite needed for ideal smoothness—we employed Bezier curve smoothing to enhance the consistency of the airfoil's surface and sampled 50 evenly-spaced points from the top and bottom airfoil edges to represent the airfoil in XFOIL for analysis.

3.2.2 Discriminators

Our cGAN architecture includes four specialized discriminators:

The **Plausibility Discriminator**, a conventional cGAN discriminator that identifies whether a design is real (from the dataset) or fake (output by the generator), is *not* pre-trained and has the highest learning rate to aggressively learn to distinguish between real and generated designs.

The **Lift**, **Momentum**, and **Drag Discriminators** each evaluate the accuracy of their respective coefficient in generated airfoils against specified targets. They were pre-trained using a subset of our dataset to help with convergence which proved empirically advantageous for model convergence, likely because it reinforced the importance of accurate aerodynamic coefficients early on. These coefficients heavily influence the loss calculations, as they are over-weighted in the loss function (Eq. 2). Furthermore, achieving realistic aerodynamic coefficients, validated by XFOIL, requires a plausible airfoil design, inherently integrating the plausibility constraint into all discriminator functions.

Due to time constraints, we did not continuously use XFOIL during training; XFOIL takes 8 seconds per individual simulation. We freeze the pre-trained discriminators' weights to compensate for the lack of continuously accessible, differentiable ground truth and XFOIL's runtime. These discriminators proxied ground truth. Selectively fine-tuning with generated data evaluated by XFOIL between training epochs increased training time by orders of magnitude without improvements.

[2]A comparable model schematic is found in Analogous Model Schematic in the Appendix.

3.2.3 Loss Functions[3]

Our model uses specific adversarial loss functions to regulate the generator and discriminators:

$$L(G, D) = \min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x|c)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z|c)))] \tag{1}$$

The objective functions for the generator $\mathcal{L}_{\text{G}}$ and plausibility discriminator $\mathcal{L}_{\text{D}}$ are represented as:

$$\mathcal{L}_{\text{G}} = (\mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z|c)))]) + 0.1 \cdot \mathcal{L}_{\text{drag}} + 1.0 \cdot \mathcal{L}_{\text{lift}} + 0.7 \cdot \mathcal{L}_{\text{momentum}} \tag{2}$$

$$\mathcal{L}_{\text{D}} = -\mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x|c)] - \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z|c)))] \tag{3}$$

The model had a constrained domain with a relatively low 2.0° tilt on all airfoil simulations. Thus, we assigned the weights to the coefficient-specific losses in $\mathcal{L}_{\text{G}}$ based on the distribution of the training data set for each aerodynamic coefficient (Fig. 1). Since the distribution was greatest for lift and least for drag, weights of 1.0, 0.7, and 0.1 were respectively assigned to $\mathcal{L}_{\text{lift}}$, $\mathcal{L}_{\text{momentum}}$, and $\mathcal{L}_{\text{drag}}$. This prioritized minimizing the discrepancy between the generated and target lift coefficients, while proportionately emphasizing momentum and acknowledging the lesser importance of drag.

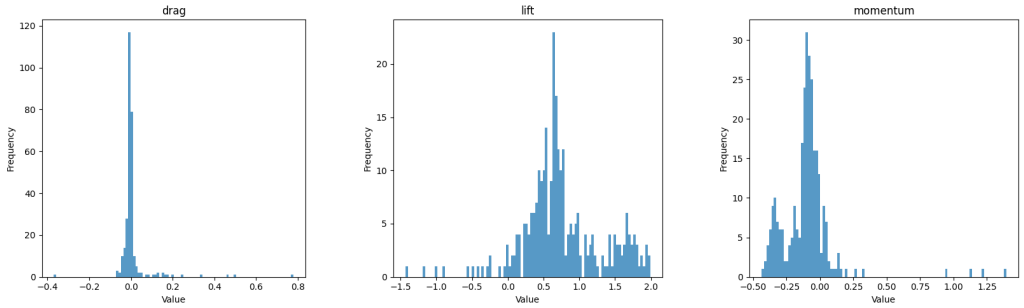

Figure 1: Distributions of aerodynamic coefficients in the training dataset.

3.3 Model Training

We used the Adam optimizer for the generator and plausibility discriminator parameters, adjusting learning rates based on empirical effectiveness across training cycles. Pre-trained discriminators used vanilla SGD optimizers to maintain lower learning rates and retain pre-training knowledge.

We trained the model with the generator learning rate of 0.001 (the highest in the model), plausibility discriminator at 0.0001, and pre-trained discriminators at 0.00001. Each training process lasted ~20 hours over 100,000 epochs. However, optimal generator performance typically occurred by epoch 18,000 (depending on the learning rate), with training stability deteriorating rapidly thereafter.

By epoch 80,000, the discriminator achieved 100% accuracy in distinguishing real from fake inputs, indicating potential overfitting or training imbalance. To mitigate training collapse, we implemented a conditional gradient update strategy—when the generator's loss exceeded 5.0, discriminator gradient updates stopped until the network loss values stabilized—for improved training consistency. We trained on NVIDIA GPUs with a batch size of 256, evaluating every 100 epochs for efficiency.

4 Results

4.1 Loss Graph Analysis

In Fig. 2, the relatively stable moving average loss graphs for the pre-trained discriminators suggest that our strategy to pre-train and maintain low learning rates effectively maintained their functionality without dominating the training dynamics.
In Fig. 3, the generator and plausibility discriminator exhibited a typical adversarial relationship—continuously trying to outperform each other. This pattern is important for improving the generator so it produces airfoils increasingly indistinguishable from the dataset. This back-and-forth indicates an effective training process, as both networks were forced to continually improve.

[3]Variables are defined in Loss Functions in the Appendix. This is inspired by Almasri et al. (2022) (1).

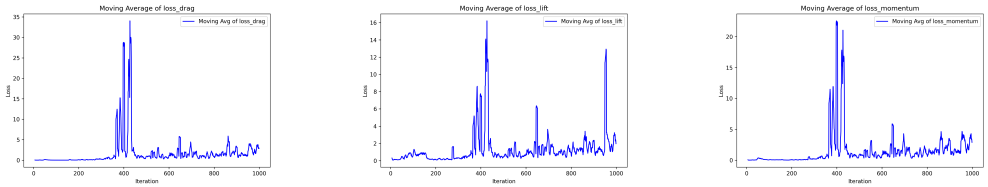

Figure 2: Progression of training loss for drag, lift, and momentum discriminators, respectively.

However, ~20% into training, the training collapsed as the plausibility discriminator became *too* effective, perfectly distinguishing between real and generated designs and leading to overfitting where it no longer provided useful gradients for the generator to learn from.

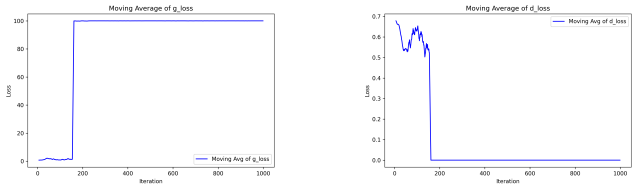

Figure 3: Progression of training loss for generator and plausibility discriminator, respectively.

4.2 Coherence and Performance Analysis

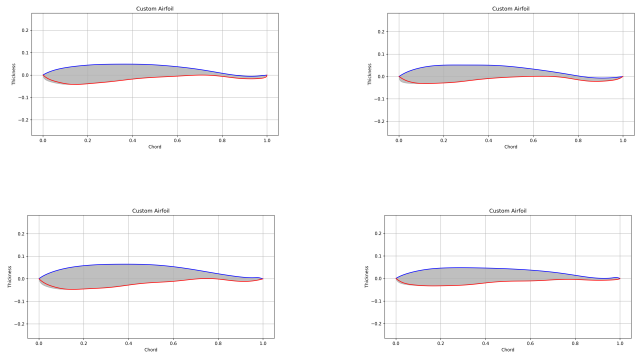

Figure 4: Examples of airfoils generated by the network after smoothing and normalization.

Next, we assessed the network's ability to generate coherent airfoils. After applying Bezier smoothing and normalizing the airfoils to a unit chord, the initial results (Fig. 4) were promising.[4] To evaluate broader performance across varying conditions, we defined a testing method using percentile-based benchmarks from the training data. We extracted the 5^{th}, 50^{th}, and 95^{th} percentiles for each coefficient—lift, drag, and momentum.[5] For each test on a specific coefficient (e.g. lift), we varied its value from the 5^{th} percentile (0.0) to the 95^{th} percentile (1.8) in 0.1 increments, generating 5-10 airfoils at each step while discarding clear outliers and keeping the other two coefficients constant at their median values (drag at 0 and momentum at -0.1). The resulting airfoils were then analyzed using XFOIL to measure the actual aerodynamic coefficient and plotted against the target coefficients.

The low weight on $\mathcal{L}_{\text{drag}}$ (0.1) and the narrow range of drag coefficients in the training data (from -0.03 to 0.08) explain why the generator predominantly outputs values near zero for drag across all target values in the test benchmarking. The model appears unable to vary drag coefficient values beyond those observed in the training data, likely prioritizing other components of the loss function that are either weighted more heavily or exhibit greater variability in the training data.

Conversely, despite a high weight on $\mathcal{L}_{\text{lift}}$ (1.0) and a broad range of lift coefficients in the training data (from 0.00 to 1.76), the generator tends to produce outputs averaging near zero across targeted lift

[4]Further discussion is found in Effective Generalization Examples and Ineffective Generalization Example.
[5]These values are summarized in Training Data Summary.

values despite an average training value of 0.75. This suggests a possible overemphasis on minimizing errors associated with other loss terms at the expense of accurately generating lift coefficients. Thus, adjusting the weight on $\mathcal{L}_{\text{lift}}$ may help better align the generator's outputs with the target values.

The moderate weight on $\mathcal{L}_{\text{momentum}}$ (0.7) and the relatively narrow distribution of momentum coefficients in the training data (ranging from -0.36 to 0.06) reinforces a pattern in our test benchmarking. Despite the intention for the generator to match the target momentum values as indicated by the loss function, the generator's outputs often regress toward the average value observed in the training data, rather than adhering closely to the varied target values. This suggests that while the weight on $\mathcal{L}_{\text{momentum}}$ is moderate, it may not be sufficient to influence the generator to prioritize accuracy against the target values over reproducing common traits from the training dataset. This could indicate that the learning process is overly influenced by the central tendency of the training data, potentially overshadowing the objective of aligning with the specified target values in test scenarios.

We can extend this argument to the model generally. It seems the adversarial loss heavily influences training, likely at the expense of achieving aerodynamic targets, which may stem from our earlier observation of the inherent weight that plausibility plays in *all* the discriminators, as achieving target coefficients requires a "plausible" airfoil. The airfoils generated by the model, while visually plausible and in line with variations of modern airfoil shapes, often do not meet the target metrics and rather revert to average values rather than adapting to a broader range of specified targets.

This potential overfitting to common data traits at the expense of accuracy in meeting aerodynamic targets underscores the need for a recalibrated approach in the training process. The model is limited in achieving target aerodynamic metrics by its constrained training data and subprime loss weighting scheme. Thus, expanding the dataset and adjusting the weights within the loss function may help the model generate outputs that align with specified targets rather than merely replicating the average characteristics of the dataset.

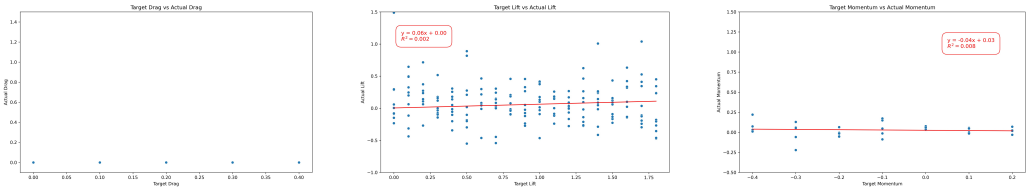

Figure 5: Target vs. generated values for drag, lift, and momentum coefficients, respectively.

5 Conclusion

Given the limited compute and time we had for this project, there remains unexploited potential in our study. We propose potential extensions of this project below:

Increased Versatility: Our model used only three aerodynamic parameters: lift, drag, and momentum coefficients. This only scratches the surface of airfoil customizability. Expanding this model to include additional parameters—e.g., weight distribution, shape, and tensile strength—and perturbing environmental conditions like Mach and viscosity values would be an obvious next step for this project to improve airfoil design.

Integrate Live Evaluation: The overhead required by XFOIL for CFD calculations was one of our project's main constraints. Integrating it directly was impractical not only because it was non-differentiable but also because of its extensive runtime. Though our discriminators' pre-training provided a differentiable approximation aiding the generator to create more "plausible" designs, this can be replaced with a more robust CFD simulation method, which would allow for precise gradient feedback and encourage the generator to better align its generation with the target coefficients.

Expand to 3D Modelling: Given that real-world applications of arifoils extend beyond 2D models, adapting our project to generate three-dimensional shapes (as in Shu et al. (2020) (6)) could be integrated with additive manufacturing techniques, which could cut the time and costs associated with the operation and creation of planes, and potentially create safer planes while reducing the material costs, manufacturing complexity, and operating expenses.

References

[1] Almasri, W., Danglade, F., Bettebghor, D., Adjed, F., and Ababsa, F., "Deep Learning for Additive Manufacturing-driven Topology Optimization," *Procedia CIRP*, (2022). https://www.sciencedirect.com/science/article/pii/S2212827122007673

[2] Drela, M., "Xfoil," Version 6.99, *Massachusetts Institute of Technology*. https://web.mit.edu/drela/Public/web/xfoil/

[3] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., "Generative Adversarial Networks," *arXiv*, (2014). https://arxiv.org/abs/1406.2661

[4] Marzocca, P., "The NACA Airfoil Series," *Clarkson University*, (2016). http://people.clarkson.edu/~pmarzocc/AE429/The%20NACA%20airfoil%20series.pdf

[5] Selig, M., "UIUC Airfoil Coordinates Database," Department of Aerospace Engineering, *University of Illinois at Urbana-Champaign*. https://m-selig.ae.illinois.edu/ads/coord_database.html

[6] Shu, D., Cunningham, J., Stump, G., Miller, S. W., Yukish, M. A., Simpson, T. W., and Tucker, C. S. (November 28, 2019). "3D Design Using Generative Adversarial Networks and Physics-Based Validation.," ASME. *J. Mech*. Des., (2020). https://doi.org/10.1115/1.4045419

[7] Yonekura, K., Suzuki, K., "Data-driven design exploration method using conditional variational autoencoder for airfoil design." *Struct Multidisc Optim*, (2021). https://doi.org/10.1007/s00158-021-02851-0

Appendix

Analogous Model Schematic

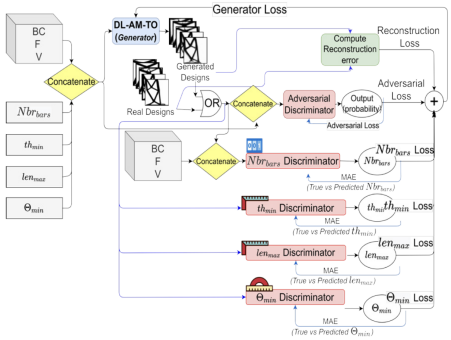

Figure 6: Example network from Almasri et. al. 2022 (1) that performs a similar experiment.

Loss Functions

Our model uses specific adversarial loss functions to regulate the generator and discriminators:

$$L(G, D) = \min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x|c)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z|c)))]$$

Here, $G((z|c), \theta_g)$ represents the conditional generator with parameters θ_g, and $D((x|c), \theta_d)$ is the conditional discriminator with parameters θ_d. The variables z, x, and c denote the latent vector, design variables vector, and vector of specific input conditions (aerodynamic metrics), respectively.

The objective functions for the generator $\mathcal{L}_{\text{G}}$ and plausibility discriminator $\mathcal{L}_{\text{D}}$ are represented as:

$$\mathcal{L}_{\text{G}} = (\mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z|c)))]) + 0.1 * \mathcal{L}_{\text{drag}} + 1.0 * \mathcal{L}_{\text{lift}} + 0.7 * \mathcal{L}_{\text{momentum}}$$

$$\mathcal{L}_{\text{D}} = -\mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x|c)] - \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z|c)))]$$

The adversarial loss term measures the generator's effectiveness at creating designs realistic enough to be classified as real by the discriminator. Each $\mathcal{L}_j = \|c_j - \hat{c}_j\|_2^2$ corresponds to losses associated with each of the three aerodynamic performance metrics (i.e., lift, momentum, and drag coefficients).

Effective Generalization Examples

Fig. 7 (left) showcases an airfoil generated by the model, where we specified a target lift coefficient of 0.5, with both drag and momentum coefficients targeted at 0. The model successfully produced an airfoil with a lift coefficient of 0.4858, while maintaining near-zero values for both drag and momentum.

Similarly, re-running the same target coefficients yields Fig. 7 (right), a generated airfoil with a lift coefficient of roughly 0.52—an acceptable result for a target lift coefficient of 0.5.

Repeated simulations showed that although some generated airfoils are infeasible, a substantial portion align well with our desired specifications. These experiments demonstrate the model's potential for achieving high accuracy with specified aerodynamic metrics, suggesting effective generalization.

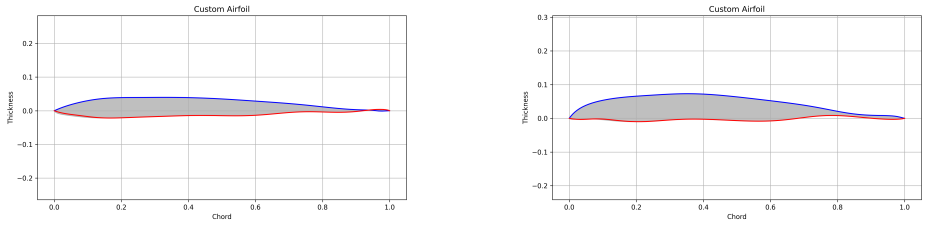

Figure 7: Airfoil generations close to target specifications.

Ineffective Generalization Example

Despite its strengths, the model has limitations. Currently, our model tends to generate airfoils with relatively low lift coefficients, typically concentrated around 0 from -0.25 to 0.25 (Fig. 5), which may be expected given the training data's clustering around the median (Fig. 1). Given that the maximum lift coefficient in our operational environment is clipped at 2.0, the model struggles to generalize effectively toward higher targets. This limitation is demonstrated in Fig. 8, where the model attempts to achieve a lift coefficient of 1.0, with zero drag and momentum. While it manages to approximate drag and momentum accurately (0.0000 and 0.2400), the lift coefficient is significantly off-target (-0.5871), even turning negative, underscoring the need for an expanded training dataset that includes higher lift examples.

Nevertheless, even though this outlying generated lift coefficient is on the extreme end (per Fig. 5), we qualitatively observe that the airfoil shape roughly mimics an expected airfoil shape, suggesting the model is certainly learning to produce plausible airfoil shapes, it just may not be optimal yet. This suggests, with further refinement in model architecture and extended computing resources, performance could improve significantly.

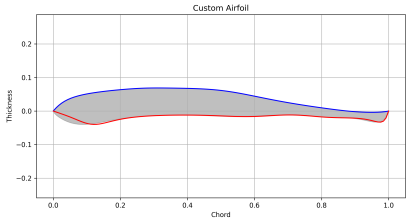

Figure 8: Airfoil generation far from target specifications.

Training Data Summary

Table 1: Training Data Summary Rounded Percentiles by Coefficient

Coefficient	5^{th} Percentile	50^{th} Percentile	95^{th} Percentile
Drag	-0.0369	0.0039	0.0772
Lift	0.0000	0.6609	1.7571
Momentum	-0.3562	-0.0969	0.0594