Airfoil GAN: Encoding and Synthesizing Airfoils for Aerodynamic Shape Optimization

Yuyang Wang¹, Kenji Shimada¹ and Amir Barati Farimani¹

¹Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA
*Corresponding author. E-mail: barati@cmu.edu

Abstract

The current design of aerodynamic shapes, like airfoils, involves computationally intensive simulations to explore the possible design space. Usually, such design relies on the prior definition of design parameters and places restrictions on synthesizing novel shapes. In this work, we propose a data-driven shape encoding and generating method, which automatically learns representations from existing airfoils and uses the learned representations to generate new airfoils. The representations are then used in the optimization of synthesized airfoil shapes based on their aerodynamic performance. Our model is built upon VAEGAN, a neural network that combines Variational Autoencoder with Generative Adversarial Network and is trained by the gradient-based technique. Our model can (1) encode the existing airfoil into a latent vector and reconstruct the airfoil from that, (2) generate novel airfoils by randomly sampling the latent vectors and mapping the vectors to the airfoil coordinate domain, and (3) synthesize airfoils with desired aerodynamic properties by optimizing learned features via a genetic algorithm. Our experiments show that the learned features encode shape information thoroughly and comprehensively without predefined design parameters. By interpolating/extrapolating feature vectors or sampling from Gaussian noises, the model can automatically synthesize novel airfoil shapes, some of which possess competitive or even better aerodynamic properties comparing to airfoils used for model training purposes. By optimizing shapes on the learned latent domain via a genetic algorithm, synthesized airfoils can evolve to target aerodynamic properties. This demonstrates an efficient learning-based airfoil design framework, which encodes and optimizes the airfoil on the latent domain and synthesizes promising airfoil candidates for required aerodynamic performance.

Keywords: Deep Learning; Generative Adversarial Network; Airfoil Design; Design Optimization

1. Introduction

Geometry parameterization plays an important role in shape design (Chang and Chen, 2011; Masters et al., 2017; Salunke et al., 2014; Samareh, 2001), and geometry heavily influences the performance, especially in the design of aerodynamic products like airfoils. A practical and effective airfoil design must meet certain aerodynamic requirements, like lift, drag, pitching moment, and critical-speed characteristics (Abbott et al., 1945). Due to such multimodality of the airfoil design space, gradient-free methods is commonly applied to optimize the airfoil design (Rajnarayan et al., 2008). For feasible and efficient optimization, various airfoil shape parameterization or dimension-reduction methods are implemented to define a lower-dimensional design space. The selection of the design space greatly influences the performance of the airfoil design optimization. Traditional parameterization or dimension-reduction techniques rely on manually selected design parameters like control points of Bézier curves (Sederberg and Farouki, 1992) or B-splines (Derksen and Rogalsky, 2010), which places restrictions on the generalization to various shapes

Received: a; Revised: b; Accepted: c
as well as synthesizing novel geometries.

Recent years have witnessed the success of deep learning (Le-Cun et al., 2015) in many fields like computer vision (Voulodimos et al., 2019), natural language processing (Cambria and White, 2014) and robotics (Carrio et al., 2017; Pierson and Gashler, 2017). Such data-driven methods can automatically learn compact and comprehensive representations from samples. However, most of the prevalent deep learning models are based on supervised learning, meaning the samples are paired with manually tagged labels. Such supervision makes the model hard to generalize since both the amount of labeled data and the information contained in the label are limited. Hence, self-supervised learning is proposed to learn the features directly from data. Autoencoder (AE) (Kramer, 1991) proposes to learn latent features through an encoder-decoder architecture, where the encoder down-samples high-dimensional input into a latent feature domain while the decoder reconstructs the sample from learned low-dimensional feature. By minimizing the difference between the reconstructed sample and the original one, AE automatically learns features without any labels. However, AE performs poorly in generating novel samples from the latent domain. To this end, Variational Autoencoder (VAE) (Kingma and Welling, 2013; Rezende et al., 2014) introduces the KL divergence regularization on the latent feature domain with respect to a prior standard normal distribution while follows the insight of the encoder-decoder. Therefore, VAE can generate novel samples by decoding the latent vectors sampled from the prior distribution. Generative Adversarial Network (GAN) (Goodfellow et al., 2014) pushes learning from self-supervision even further via a min-max game between a generator and a discriminator. The discriminator works as a classifier to determine real samples from synthesized fake ones. Meanwhile, the generator, which synthesizes samples from random noise, is intended to cheat the discriminator. By jointly training the two components, GAN can generate super high-quality realistic samples (Karras et al., 2019), which VAE fails to achieve. Through learning to reconstruct or synthesizing samples, the self-supervised models automatically encode high-dimensional input into informative features, which can be generalized to different tasks without restrictions from human labels. Implementation of self-supervised deep learning methods on shape parameterization, like VAE and GAN, can help overcome the limitations of traditional techniques and synthesize shapes with great novelty, which can provide insights for future geometric design.

In this work, we intend to use deep generative models for the following challenges in shape designs: (1) learning of expressive representations that model the design, (2) generation of realistic designs, and (3) optimization of generated designs toward desired targets. In particular, our work focuses on the airfoil design. To this end, we adapt the insight of VAEGAN (Larsen et al., 2015) and curate the architecture and losses. VAEGAN takes advantage of both VAE and GAN. With the encoder-decoder architecture from VAE, the model learns to explicitly encode an existing airfoil shape into a low-dimensional feature domain and reconstruct the shape with little error. Also, the KL divergence regularizes the latent features to follow the prior standard normal distribution, so that latent vectors can be sampled from the prior distribution. With the discriminator from GAN, our decoder automatically learns to generate a large number of high-quality novel airfoils from the prior distribution by adversarially competing with the discriminator. Moreover, we introduce the genetic algorithm to VAEGAN generated airfoil designs to optimize the aerodynamic properties (i.e., lift and drag coefficients). We further apply the genetic algorithm to optimize the generated airfoils from VAEGAN toward desired aerodynamic properties.

Our experiments show that generated airfoils are smooth even without any smoothing post-process. K-means clustering in the learned feature domain demonstrates that feature vectors encode essential shape information in a way that each cluster represents various shape patterns. Further test on learned latent features illustrates that different geometry information is encoded in each dimension of the representation. The performance of the model in synthesizing novel airfoils is examined as well. By either interpolation or extrapolation of feature vectors, a synthesized airfoil inherits features of parent samples, while generated airfoils from sampled Gaussian noise show great novelty in a way that it is not a simple combination of two existing airfoils. A further experiment on the aerodynamic properties of synthesized airfoils, either by interpolating, extrapolating, or sampling, indicates the synthesized airfoils can possess competitive aerodynamic properties, and some even surpass the existing ones. With a genetic algorithm (Mitchell, 1998), airfoil geometries can be optimized on the feature domain and evolve to possess specific aerodynamic properties. Our model proves its ability to parameterize the existing airfoil shape as well as generating novel and practical airfoils; both lead to designing the new generation of airfoils more intelligently and efficiently without intensely relying on experimental experience or manually selected design parameters. Overall, experiments have demonstrated that the deep generative model presented in our work learns meaningful shape representations and generates realistic airfoil designs. Besides, the generated designs can be optimized toward target properties when directly combined with optimization methods.

2. Related Works

We propose to use a generative model to synthesize novel airfoils without any predefined design parameters. Through training via the gradient-based method, our deep learning model also automatically learns to parameterize airfoils into latent feature vectors. To better address the insight of our work, this section reviews previous work on shape parameterization, especially on aerodynamic geometries and the implementation of self-supervised deep learning on geometry design and synthesis.

2.1 Geometry Parameterization and Dimension Reduction

A lot of work has been done in parameterizing complex shapes and reducing geometric dimensions, including discrete approach, analytical approach, polynomial and spline approaches, free-form deformation (FFD) approach, etc. The discrete approach leverages the grid-point coordinates as design parameters and different shapes are generating by moving individual grid points (Campbell, 2003; Jameson et al., 1998). Though easy to implement, such method is likely to synthesize shapes that are not smooth, which can be a challenge for CFD solvers (Jameson, 2006). Therefore, it is commonly preferred to use more robust and efficient shape parameterization strategies rather than the discrete approach.

In analytical approach, a compact formulation for parameterization is obtained by adding analytic shape functions to the baseline shape (Hicks and Henne, 1978). The design variables, in this case, are the coefficients correlated with shape functions. Also, HAGER et al., 1992 follows the same formulation strategy but with different shape functions. The such method works well for several families of airfoils, but it can fail in representing radical new airfoil designs.

Polynomial and spline are also utilized in shape parameteri-
zation. With different orders of polynomials as the basis, the airfoil shape can be described as a linear combination of the basis (Elliott and Peraire, 1997; TAYLOR et al., 1991). However, high-order terms can overfit to high-frequency noise, especially when coefficients are of different magnitudes. Besides polynomial, Bézier curve (Sederberg and Farouki, 1992), which is built upon the Bernstein polynomials, is another mathematical formulation of curves. In detail, \( n + 1 \) control points of Bézier are needed to define an \( n \)-degree Bézier curve. Although Bézier and polynomial curves are mathematically equivalent, Bézier usually perform better in controlling a curve since control points are closely related to the curve position and shape. To mitigate the rounding error, De Casteljau (Boehm and Müller, 1999), a recursive algorithm, is introduced to compute the Bernstein polynomials numerically. The B-spline curve with B-spline basis functions is also utilized to describe the airfoil shape. Non-uniform rational B-spline (NURBS) is introduced (Farin, 2014) to represent both standard geometric objects like lines, circles, ellipses, and cones, as well as free-form geometries. The polynomial and spline approaches are suited for two dimensional shapes. However, when modeling complex shapes, a large number of control points are need, which can lead to irregular and wavy shapes in optimization.

FFD utilizes high-level shape deformation instead of lower-level geometric entities to represent a shape. Based on this insight, Sederberg and Parry, 1986 presents a technique which can apply deformation either locally or globally based on trivariate Bernstein polynomials. The deformation is manipulated by control points of trivariate Bézier volumes. In Coquillart, 1990, an extended free-form deformation (EFFD) method is presented, which enables arbitrarily shaped deformations by using non-parallellepiped lattices. Research presented in Yeh and Vance, 1998 incorporates FFD and sensitivity analysis where geometry changes and structural responses are correlated, and the shape satisfying deformation or stress constraints can be found easily.

Other physically intuitive approaches, like PARSEC (Sobieszczky, 1997), defines geometric parameters to express the airfoil shape, including upper and lower curvature, thickness, leading-edge radius, etc. Bézier-PARSEC (Derksen and Rogalsky, 2010), which combines Bézier and PARSEC, uses PARSEC parameters to define Bézier curves. Besides, camber and thickness mode shapes from the existing airfoils are utilized to parameterize airfoil shapes (J. Li et al., 2019). Also, the linear reduction method like the SVD is utilized to extract airfoil representations and optimize shape design (Poole et al., 2019). Parametric model embedding Serani and Diez, 2023 extends the formulation of Karhunen–Loève expansion Diez et al., 2015 via a generalized feature space that includes design variables vectors and a generalized inner product to select the latent design space. Active subspace Constantine et al., 2014 leverages gradient to detect the directions of the strongest variability and uses the directions as low-dimensional active subspace to parameterize surfaces. Ghorbani and Khameneifar, 2021 extends B-spline to reconstruct airfoil shapes from noisy 3D point clouds. The conventional dimension reduction or parameterization techniques have been implemented in different scenarios and successfully represented the existing airfoil shapes. However, these methods usually require pre-defining the design space as well as the boundary of design space, like the design parameters in PARSEC, shape functions, etc., which can degrade the synthesis of novel/new airfoils and optimization towards the desired design.

### 2.2 Deep Learning in Geometry Design

In recent years, deep learning has been successful at extracting informative features from data (LeCun et al., 2015). Supervised learning, which maps an input to an output based on training input-output pairs, has become general solutions in many fields, like image classification (He et al., 2015; LeCun et al., 1989; Szegedy et al., 2015), object detection (Redmon et al., 2016; Ren et al., 2015), and segmentation (He et al., 2017; Y. Li et al., 2017). However, supervised learning is limited to labeled data, and collection of a large number of labels can be expensive. Self-supervised learning proposes to utilize the data itself as labels and learn features by predicting the data, which requires no explicitly labeled data. Following the insight, variational Autoencoder (VAE) (Kingma and Welling, 2013), learns representative latent features by reconstructing the input. Also, with the introduction of GAN (Goodfellow et al., 2014), deep learning in a self-supervised manner has been widely used in synthesizing realistic samples, and some sophisticated GAN-based model can generate high-resolution images which are even hard to distinguish by humans (Karras et al., 2019). Due to the ability to extract representative features and synthesize realistic samples, deep learning has been widely used to make the geometric design more systematic and efficient. This section will introduce some work of deep learning in geometric design, especially in aerodynamic shape design.

In the work of Norgaard et al., 1997, a multi-layer perceptron (MLP), which takes the angle of attack and flap setting as input, is trained to predict multiple aerodynamic coefficients, including lift coefficient, drag coefficient, and moment of inertia. Further, Zhang et al., 2018 utilizes CNN with airfoil images as input to learn the lift coefficients of different airfoils in multiple flow Mach numbers, Reynolds numbers, and diverse angles of attack. Similarly, in Yilmaz and German, 2017, CNN is implemented to predict the pressure coefficient value at the test point. Moreover, Yilmaz and German, 2018 presents a CNN-based method to learn the correlation between airfoil geometry and pressure distribution. The model can also conduct an inverse airfoil design given the pressure.

On the other hand, deep generative models have been successfully implemented to generate realistic samples. VAE Kingma and Welling, 2013 is proposed to generate plausible samples via an encoder-decoder architecture while optimizing the variational lower bound of the log-likelihood of the training data. However, VAE suffers from generating samples that are blurry and averaged over the training samples Cai et al., 2019. Besides, GAN, with a discriminator and a generator, is designed to learn features and generate samples without manually tagged labels. The discriminator is a classifier telling the true input from the synthesized fake one, while the generator is intended to generate plausible samples to cheat the discriminator. Since first introduced (Goodfellow et al., 2014), many pieces of research have been dedicated to pushing the edge of GAN. In Wasserstein GAN (Arjovsky et al., 2017), by introducing Wasserstein distance, the quality of generated samples can be well measured during training. ConditionalGAN (Mirza and Osindero, 2014) and InfoGAN (X. Chen et al., 2016) extend GAN to generate a sample from various categories within one model. Also, DCGAN (Radford et al., 2015) and StyleGAN (Karras et al., 2019) can generate high-quality realistic samples with sophisticated architectures.

Deep generative models have been applied to geometry design. D’Agostino et al. D’Agostino et al., 2018 introduce autoencoder to learn a reduced design space and apply optimization on the learned latent space. With the ability to generate plausible samples, GAN provides a powerful architecture to learn representa-
tions that can help shape design Nie et al., 2021. Based on this insight, W. Chen et al., 2019; W. Chen et al., 2020 introduce Bézier-GAN, which uses a GAN to generate Bézier curve control points and then uses the control points to formulate the boundary of airfoils. Such a pipeline guarantees that generated airfoils are smooth, and further shape optimization can be conducted on the feature domain. BSplineGAN (Du et al., 2020) further extends the insights to B-splines parameterization and proposes a surrogate neural network model to predict lift and drag coefficients. Such methods can be data efficient as they leverage pre-defined formulations of the curves or surfaces. Grey et al., 2023 propose a data-driven approach that decouples affine and undulation deformations based on Grassmannian-based representation.

Furthermore, the adoption of deep learning-based methods presents significant advantages when dealing with high-dimensional designs. Unlike traditional methods that often struggle to effectively explore such complex design spaces, our approach efficiently navigates the vast parameter spaces associated with advanced and intricate airfoil designs. This ensures that our method can tackle challenging design problems, offering valuable insights and facilitating the creation of cutting-edge designs in a variety of engineering applications.

By not limiting ourselves to a specific design space, our VAEGAN method excels at directly capturing the underlying distribution of airfoil shapes from datasets. It achieves this by effectively learning intricate non-linear relationships between the latent features it acquires and the corresponding airfoil designs. An additional advantage is the ability of our method to generate novel airfoil shapes that do not exist in the training dataset. This facilitates the exploration of new design possibilities beyond the constraints of the chosen design space. Furthermore, deep learning-based methods offer efficient solutions for high-dimensional designs, whereas traditional methods often struggle to explore such design spaces effectively.

Conditional GAN has also been utilized to generate airfoil shapes given certain airfoil properties (Achour et al., 2020; Yilmaz and German, 2020). Further, GAN can be implemented to generate three-dimensional samples, as shown in Huang et al., 2015; Sinha et al., 2017; Wu et al., 2016. Wu et al., 2016 proposes to use a three-dimensional convolutional layer to generate volumetric objects. While Huang et al., 2015 trains a generative model of three-dimensional shape surfaces, which directly encodes surface geometry and shape structure, Kalogerakis et al., 2012 further proposes a model to represent probabilistic relationships between properties of shape components and relates them to learned underlying causes of structural variability within the domain.

In our work, we propose to use a VAEGAN-based model to extract features of airfoil shapes and synthesize new airfoil designs. It is demonstrated that such a self-supervised learning framework learns representative geometric features without predefined design parameters. The learned latent features can be utilized to synthesize new designs through interpolation or extrapolation. In comparison to conventional parameterization methods, our model can generate a wider variety of new airfoils, and some show promising aerodynamic properties. By applying the genetic algorithm, the VAEGAN synthesized airfoils can be optimized to desired aerodynamic properties. Unlike previous works, like Bézier GAN and B-Spline GAN, that rely on pre-defined design formulations like Bézier or B-Spline curves, we do not leverage prior formulation of the design and apply deep neural networks as universal function approximators to model a wide range of designs. By not imposing a specific design space, our proposed VAEGAN method is able to capture the underlying distribution of airfoil shapes directly from datasets more effectively by learning complex non-linear relationships between the learned latent feature and airfoil designs. Moreover, such a method is expected to generate novel airfoil shapes that are not present in the training dataset. This facilitates the exploration of new design possibilities beyond the constraints of the chosen design space. Besides, such deep learning-based methods can be extended to high-dimensional designs, where well-defined formulations are hard to comprehensively model the objects. Using the learned latent feature, optimization can navigate designs toward desired properties.

### 3. Method

#### 3.1 Data Pre-processing

The UIUC Coordinates Database (Selig, 1996), which contains more than 1,600 2-dimensional airfoils, is used to train the generative model. The x coordinates of all airfoils from the database have been scaled to \([0, 1]\). However, each airfoil in the database is represented by varying numbers of points with different x and y coordinates. Such variation restricts data to be fed directly into a neural network model, which requires a homogeneous input. To deal with it, the upper boundary and the lower boundary are interpolated by cubic splines, respectively. Within each boundary, \(N\) points are selected by the x coordinate given in Eq. 1.

\[
\theta_i = \frac{\pi(i - 1)}{N}, \quad x_i = 1 - \cos(\theta_i),
\]

where \(i\) denotes the index of each point, and \(N\) is set to 100 in our case. By this means, all the interpolated airfoils have 200 points in total and share the same \(x\) coordinates. Therefore, only the \(y\) coordinates of each airfoil are fed into the model, which reduces the dimensionality of the data. Finally, all \(y\) coordinates are scaled to \([-1, 1]\) by multiplying a normalization coefficient. As illustrated in Fig 1, the first row shows the original airfoils from the UIUC Coordinate Database, while the second row shows the corresponding pre-processed airfoils.

#### 3.2 VAEGAN

Our model is based upon VAEGAN (Larsen et al., 2015), which takes advantage of both VAE (Kingma and Welling, 2013; Rezende et al., 2014) and GAN (Goodfellow et al., 2014). VAE contains two components: an encoder and a decoder. The former encodes a high-dimensional sample, \(x\), into a low-dimensional latent representation, \(z\). While the decoder takes as input the latent vector, \(z\), and upsamples from the representation domain to the original data domain, \(\hat{x}\). The encoder and decoder are given as:

\[
z \sim \text{Enc}(x) = q(z|x), \quad \hat{x} \sim \text{Dec}(\hat{z}) = p(\hat{x}|z),
\]
where \( q(z|x) \) denotes the distribution of latent vector \( z \) given airfoil \( x \) and \( p(\tilde{x}|z) \) denotes the distribution of reconstructed airfoil \( \tilde{x} \) given \( z \). To regularize the encoder, VAE takes into consideration a prior distribution of the latent vector, \( p(z) \). Here it is assumed that \( z \sim \mathcal{N}(0, I) \), which follows an isotropic Gaussian distribution. The loss function for VAE to minimize is given by:

\[
L_{VAE} = L_{recon} + L_{prior},
\]

with

\[
L_{recon} = ||\tilde{x} - x||^2, \quad \text{and} \quad L_{prior} = D_{KL}(q(z|x)||p(z)),
\]

where \( L_{recon} \) measures how well the reconstructed data, \( \tilde{x} \), is comparing to the original \( x \) by the mean square error (MSE), and \( L_{prior} \) is the Kullback-Leibler divergence (KL divergence), which measures the difference between encoded representation vectors and Gaussian distribution.

VAE learns the representation of samples and can reconstruct them from \( z \) with the encoder-decoder architecture. However, it suffers from poor performance in generating novel samples, which have not been seen before (Goodfellow et al., 2014 Larsen et al., 2015). To this end, a generative adversarial network (GAN) (Goodfellow et al., 2014) is introduced, which contains a discriminator, \( D \), and a generator, \( G \), competing with each other in a self-supervised manner. \( G \) tries to generate plausible samples to fool \( D \), while \( D \) keeps sharpening its decision boundary to determine synthesized fake samples from real ones. In detail, the generator, \( G \), is fed with random noise \( \tilde{z} \sim p(\tilde{z}) \) and maps the noise to the data sample domain to generate fake \( \tilde{x} \). The discriminator takes both \( x \) and \( \tilde{x} \) to predict whether the input is from the real dataset or generated by \( G \). The cross-entropy loss function for the min-max game is given as:

\[
\min_G \max_D L_{GAN} = \log D(x) + \log(1 - D(G(\tilde{z}))).
\]

GAN generates samples from random noise and lacks the mechanism of learning latent representations of the input. Therefore, it is hard to directly learn compact features and optimize the generated airfoils in the latent feature domain using the original GAN. In practice, we not only care about generating new airfoil designs but also expect airfoils with optimized aerodynamic properties. Driven by this, we build our model upon VAEGAN, which combines VAE and GAN. Namely, the generator is replaced by an encoder-decoder structure from VAE, as shown in Figure 2. Notice that in VAEGAN, the model generates a reconstructed sample, \( \tilde{x} \), given a real sample, \( x \), and meanwhile generates a fake sample, \( \tilde{x} \), directly from noise, \( \tilde{z} \). Both \( \tilde{x} \) and \( \tilde{x} \) should be classified as fake by the discriminator, \( D \), and only \( x \) is recognized as the real sample. Hence GAN loss function \( L_{GAN} \) from Eq. 5 is modified to:

\[
L_{GAN} = \log(D(x)) + \log(1 - D(G(\tilde{z}))) + \log(1 - D(Dec(Enc(x)))).
\]

which takes into consideration the real sample, \( x \), reconstructed sample, \( \tilde{x} \), and fake sample, \( \tilde{x} \). Besides, to stabilize the training process and sharpen the decision boundary of \( D \), another loss function, \( L_{layer} \), is introduced when training the encoder and decoder. \( L_{layer} \), as given in Eq. 7, measures the \( l_1 \) distance between the values of the neurons in one particular layer of \( D \) when fake samples are fed and the values when real samples are fed. Such a layer consistency loss works as a supplementary to reconstruction loss which pushes the autoencoder to better reconstruct the real airfoil designs. In our case, we calculate \( L_{layer} \) using the second last hidden layer in the discriminator following Larsen et al., 2015.

\[
L_{layer} = ||D_l(x) - D_l(Dec(\tilde{z}))||_1.
\]

The complete loss function for VAEGAN is a weighted combination of all the loss terms given by:

\[
L = \lambda_0 L_{prior} + \lambda_1 L_{recon} + \lambda_2 L_{layer} + \lambda_3 L_{GAN}.
\]

The three components, encoder, decoder, and discriminator are trained jointly, and each term of the loss function is assigned with different weights \( \lambda \) when training each component.

### 3.3 Airfoil Synthesis

By training on the UIUC Database, the VAEGAN model automatically learns to encode airfoils into latent features and reconstruct airfoils from the feature domain. The learned latent features can be directly utilized for dimension reduction and shape parameterization. Moreover, the VAEGAN model is intended to synthesize novel airfoils which are different from samples in the training dataset. To this end, we propose three synthesis methods: interpolation, extrapolation, and sampling, all of which are conducted on the latent feature domain.

More specifically, in interpolation or extrapolation, two airfoils from the UIUC Database are first mapped to latent feature vectors, \( z_1 \) and \( z_2 \), via a well-trained encoder. A new feature vector \( \tilde{z} \), which is an affine combination of \( z_1 \) and \( z_2 \), is calculated as given in Eq. 9:

\[
\tilde{z} = \nu z_1 + (1 - \nu)z_2,
\]

where \( \nu \) is the coefficient controlling the weight between \( z_1 \) and \( z_2 \). When \( 0 \leq \nu \leq 1 \), \( \tilde{z} \) is an interpolated feature vector, else it is an extrapolation between \( z_1 \) and \( z_2 \). The interpolated/extrapolated feature vector, \( \tilde{z} \), is then fed into the decoder
to synthesize an airfoil. Also, such interpolation/extrapolation between two airfoils can be directly extended to a triplet case. Given \( z_1, z_2, \) and \( z_3 \) are three feature vectors mapped from three different airfoils via the encoder, the expression of triplet interpolation/extrapolation is shown in Eq. 10:

\[
\hat{z} = \alpha z_1 + \beta z_2 + \gamma z_3, \text{ where } \alpha + \beta + \gamma = 1. \tag{10}
\]

Similarly, when \( 0 < \alpha, \beta, \gamma < 0, \hat{z} \) is an interpolation of the three feature vectors, and an extrapolation otherwise.

Besides interpolation and extrapolation, sampling is another method to synthesize novel airfoils. Unlike interpolation or extrapolation, which relies on feature vectors from existing airfoils, sampling generates airfoils directly from random noise. A feature vector \( \hat{z} \) is randomly sampled from an isotropic Gaussian distribution, \( \mathcal{N}(0, I) \) and then mapped to an airfoil via the decoder. By this means, synthesized airfoils from sampling are less restricted since sampled latent vectors are not constrained by features extracted from airfoils in the UIUC Database and are more likely to introduce novelty to the synthesized shapes. Interpolation, extrapolation, and sampling are widely used methods to generate new samples with deep learning based models. Through interpolation and extrapolation, the model is evaluated whether it learns expressive representations to encode the airfoil designs. While sampling is more flexible to generate different and novel designs. To generate more practical airfoils, optimization can be combined with sampling to manipulate the generation as we introduce in the following section.

### 3.4 Aerodynamic-aware Shape Optimization

So far, how the VAEGAN model is built and used to generate novel airfoils has been introduced. However, the novelty in shape does not guarantee a better airfoil design. To design engineering effective airfoils, aerodynamic properties are supposed to be considered. To this end, we propose to use a genetic algorithm (GA) (Mitchell, 1998; Poon and Carter, 1995) to optimize airfoil shapes by controlling feature vectors learned from the VAEGAN model so that the airfoils can evolve to have the desired aerodynamic properties. Specifically, lift coefficient, \( C_l \), and drag coefficient, \( C_d \), which measure the aerodynamic force perpendicular and horizontal to the direction of motion, are considered to evaluate the aerodynamic performance of the synthesized airfoils.

As a non-gradient optimization technique, GA is inspired by natural selection and is intended to force individuals to gradually evolve to the optimal. Assume the GA has \( N \) generations in total and \( M \) individuals in each generation. In our case, individuals are feature vectors. We use \( z_i \) to represent all individuals in the \( i^{th} \) generation, and \( z_{ij} \) for the \( j^{th} \) individual in the \( i^{th} \) generation; also the airfoil decoded from \( z_{ij} \) is annotated as \( a_{ij} \). Similarly, \( C^l_{ij} \) and \( C^d_{ij} \) represents lift and drag coefficients of \( a_{ij} \) respectively. The fitness score, \( s_{ij} \), is used to measure the aerodynamic performance of the individual, \( z_{ij} \), as shown in Eq. 11:

\[
s_{ij} = \frac{(C^l_{ij} - C^l_t)^2 - (C^d_{ij} - C^d_t)^2}{C^l_t - C^d_t}, \tag{11}
\]

where the square of the difference between the target and current aerodynamic coefficients is calculated and normalized by the squared target \( C^l_t \) and \( C^d_t \). The fitness score is supposed to approach zero as individuals evolve on each generation. As shown in Algorithm 1, the initial generation, \( z_0 \), is randomly sampled from an isotropic Gaussian distribution, \( \mathcal{N}(0, I) \). The GA starts with the selection from the initial generation by randomly picking two individuals and comparing their fitness scores. The one with a higher fitness score wins the tournament and becomes one of the parents. \( p^I_1 \) and \( p^I_2 \) denote all the parents 1 and parents 2 in the \( i^{th} \) generation respectively. Single-point crossover is then implemented to generate offspring from parents 1 and 2. Namely, a crossover point on the parent vector is randomly selected, and all elements after that point are swapped between the two parents. Mutation in the natural selection process is also imitated with additive Gaussian noises.

#### Algorithm 1 Aerodynamic Shape Optimization via GA

```plaintext
1: procedure GA(N, M, C^l_t, C^d_t, p)
2: Initialize i := 0
3: Sample first generation \( z_0 \sim \mathcal{N}(0, I) \), for \( 0 < j < M \)
4: while \( i < N \) do
5:   Synthesize airfoils \( a_{ij} \) from \( z_{ij} \) via decoder
6:   Compute \( C^l_{ij}, C^d_{ij} \) and fitness score \( s_{ij} = \frac{(C^l_{ij} - C^l_t)^2 - (C^d_{ij} - C^d_t)^2}{C^l_t - C^d_t} \)
7:   Select \( M \) parent 1, \( p^I_1 \), and \( M \) parent 2, \( p^I_2 \), from \( z_i \) by tournament
8:   Generate next generation, \( z_{i+1,j} \), through single-point crossover
9:   With probability \( p \), \( z_{i+1,j} \) will add a Gaussian noise \( \mathcal{N}(0, I) \)
10:   \( i := i + 1 \)
11: end while
12: return The individual with the highest score from \( z_i \)
```

### 4. Results & Discussion

Our VAEGAN model consists of 3 components: an encoder, a decoder, and a discriminator, which are all built on multi-layer perceptron (MLP). We implement the VAEGAN model based on PyTorch framework (Paszke et al., 2019). As illustrated in Fig. 2, the encoder encodes 200-dimensional airfoil coordinates into a 32-dimensional feature domain while the decoder maps the feature back to the airfoil. We select the latent feature dimension, namely the number of design variables, to be 32 as a trade-off between the computational efficiency and the accuracy. Since the reconstruction error of the airfoils barely decreases even with larger latent space dimension. The discriminator is a classifier examining whether the input is a real airfoil from the UIUC Database, or a fake one reconstructed from the decoder, or synthesized from random noises. In detail, the encoder is modeled by a 3-layer MLP with the number of neurons [256, 128, 32] in each layer, and LeakyReLU (Maas et al., 2013) is implemented as the activation function in each layer. The decoder is also a 3-layer MLP with the number of neurons [128, 256, 200] in each layer. A hyperbolic tangent (Tanh) function works as the activation function in the output layer to scale all outputs into \([-1, 1]\). Similarly, the discriminator contains three layers with the number of neurons [256, 128, 1], and outputs the probability of whether the input is real or fake through a Sigmoid activation function.

To automatically learn the latent features and synthesize airfoils, the VAEGAN model is trained on the UIUC Database for 5,000 epochs, and each epoch goes through all the samples in
where $L_{\text{Enc}}$, $L_{\text{Dec}}$, and $L_D$ represent loss functions for the encoder, decoder, and discriminator, respectively. A large coefficient was assigned to the reconstruction loss $L_{\text{recon}}$ to ensure the feature contains the information of airfoil geometries. Prior loss $L_{\text{prior}}$ and $L_{\text{layer}}$ are assigned with weight 0.1 since it can lead to trivial solutions otherwise. The generator considers the adversarial GAN loss $L_{\text{GAN}}$, but instead of minimizing the loss, the generator tried to fool the discriminator by maximizing the term. $L_{\text{GAN}}$ also played an important role in VAEGAN to synthesize realistic samples. Therefore, a large coefficient is assigned to $L_{\text{GAN}}$.

To better investigate our VAEGAN-based model, we compare the performance with two other parameterization methods, principle component analysis (PCA) and variational autoencoder (VAE) (Kingma and Welling, 2013). PCA conducts a linear transformation from the pre-processed airfoil point coordinates into prioritized latent variables. In our case, the top 32 dimensions are kept as the feature. The VAE follows the same encoder-decoder architecture as the VAEGAN, but lacks the discriminator. The latent feature dimension is also set to 32, and the loss function is given in Eq. 15:

$$L_{\text{Enc}} = 0.1L_{\text{prior}} + 10L_{\text{recon}}.$$ (15)

### 4.1 Airfoil Reconstruction via Encoder-Decoder

The VAEGAN model can automatically learn feature vectors, namely mapping the high-dimension airfoils into low-dimension representations. To estimate whether or not the feature vector fully encodes the geometric information of the original airfoil, we first feed airfoils from UIUC Coordinate Database into the encoder to obtain the encoded feature vectors. The decoder then takes the vectors as input and outputs the reconstructed airfoils. Also, the Savitzky-Golay filter (Schafer, 2011), a moving polynomial fitting, is implemented to smoothen the boundary of reconstructed airfoils. In our case, the second-order polynomial is used in the Savitzky-Golay filter, and the length of the moving window is set to 7. In Fig 3, the first row illustrates samples from the UIUC Database, and the second row shows reconstructed airfoils from corresponding feature vectors, with an MSE, $3.65345 \times 10^{-4}$, between the reconstructed and original airfoils. This small error indicates the learned features well represent the shape of airfoils. The third row shows reconstructed airfoils with the Savitzky-Golay filter with an MSE, $3.65054 \times 10^{-4}$, comparing to the original airfoils. These results further demonstrate that the encoder-decoder can reconstruct airfoils that are smooth and realistic without smoothing filters. Also, we compare the reconstruction MSE on test set of our VAEGAN-based model with PCA and VAE as shown in Fig. 4a. Such reconstruction error evaluates how models perform in learning expressive representations of the designs W. Chen et al., 2020; Poole et al., 2015. Our VAEGAN-based model is compared with PCA and VAE as shown in Fig. 4a. PCA performs the best in terms of reaching the lowest MSE since it is calculated from a close form solution, whereas VAE and VAEGAN are optimized numerically via the stochastic gradient-based method. With the discriminator and adversarial loss from GAN, VAEGAN model performs slightly better than VAE in reconstruction. It should be pointed out that all three parameterization methods have small reconstruction MSEs of magnitude $10^{-4}$, meaning all the features extracted well encodes the airfoil shapes from the UIUC database. It should also be noted that though similar reconstruction errors are observed across different models, VAEGAN demonstrates preferable performance in generating new airfoil designs as shown in the following sections. We further investigate the effect of latent space dimension on the reconstruction performance as shown in Fig. 4b. As the latent dimension increases, the reconstruction MSE decreases. However, when the latent dimension exceeds 32, the improvement of reconstruction quality is trivial. Therefore, we set the latent dimension to 32 as a tradeoff between performance and computational cost. Besides, the curvatures of airfoils reconstructed using VAEGAN are compared with original airfoils as shown in Fig. 4c. It is indicated that VAEGAN can generate smooth airfoil designs similar to the ground truth.
Also, experiments are conducted to investigate what geometric features are encoded in each dimension of the learned representation. A series of manually designed feature vectors are fed into the decoder, where all the elements are set to zero except for one specific dimension. That particular element is changed gradually from –10 to 10, and the designed feature vectors are mapped to the airfoil coordinate domain by the decoder. Changes of generated airfoils are illustrated in Fig 6, and the 2D embedding of feature vectors using parametric t-SNE is shown in Fig 8. Here, only four dimensions are chosen for analysis purposes.

The 1st dimension encodes the height of the upper boundary, as illustrated in Fig 6a. As the 1st dimension increases from –10 to 10, the height of the front half airfoil increases while the tail becomes thinner. The 8th dimension encodes the camber of both the upper boundary and the lower boundary. It is shown that by tuning the 8th dimension, the upper boundary of the airfoil changes from a concave to a horizontal straight line, while the lower boundary evolves from a concave to a convex curve. Interestingly, the 22nd dimension encodes quite similar representations as the 1st dimension while in the opposite direction. In other words, generated airfoils from feature vectors whose 22nd dimension change from –10 to 10 are like those with the 1st dimension change from 10 to –10 as illustrated in Fig 8a and Fig 8c. Besides, Fig 6d shows how the last dimension is connected to the camber of the lower boundary. In detail, the curvature of the lower boundary decreases as the 32nd dimension increases.

In comparison to the features learned by our VAEGAN-based model, Fig. 7 shows the VAE-synthesized airfoils when changing only one feature dimension. As shown in Fig 7a, the first dimension of VAE features fails to encode any shape representations. Even in the dimensions where representations are learned as dimension 3 and 31 shown in Fig 7b and Fig 7c, the feature does not change continuously as we observe in the VAEGAN results. Our VAEGAN-based model learns more representative and thorough features than the VAE model. Also, the representations in each dimension are entangled, like the 3rd dimension encodes both the upper bound and lower bound.

These results indicate that, without manually designed parameters, our VAEGAN-based model learns geometrically meaningful features, and each dimension of the learned feature domain encodes informative and different geometry features. Also, each learned latent dimension can be leveraged as a design parameter when synthesizing new airfoils. The geometric information in each dimension provides new guidance for design parameter selection in comparison to conventional design methods like NACA.

### 4.4 Synthesizing Novel Airfoils

To make use of the encoder-decoder architecture to synthesize novel airfoils, we conduct experiments on interpolation and extrapolation of feature vectors obtained from the UIUC Database airfoils as well as sampling from random Gaussian noises following the method introduced in Section 3.3.

As given in Eq. 9, the affine combination of two feature vec-
our implementation, we set the number of principal components as 32 in consistence with the latent dimension of the deep generative models. And each component of value $\sigma$ is sampled from a Gaussian distribution $\mathcal{N}(\nu, 0.2|\sigma|)$. PCA, as a linear project technique, generates similar airfoil designs through sampling, which greatly restricts the ability to guide novel designs. On the other hand, both deep-learning-based generative models, VAE and VAEGAN, can synthesize different smooth airfoil shapes. To quantitatively measure the synthesized airfoils from VAE and VAEGAN, we here introduce Fréchet Inception Distance (FID) (Heusel et al., 2017), which is used to evaluate the quality of samples from deep learning-based generative models. FID is calculated by computing the Fréchet distance between two feature representations. Generally, lower FID indicates higher generative sample quality. In our case, we feed the synthesized airfoils and the UIUC airfoils into the well-trained discriminator from VAEGAN and extract the second hidden layer as the representation. The FID for VAE and VAEGAN are 1.38788 and 0.65366, respectively, meaning VAEGAN synthesizes more realistic airfoils. Also, airfoils synthesized via VAEGAN possess more novelty while maintaining the general geometric pattern of airfoils. For instance, in Fig. 11c, the first airfoils in the fourth row and the third one in the second row are different from existing samples in the UIUC database. Though such novelty does not guarantee better aerodynamic properties, some airfoils are likely to have negative lift coefficients, which are infeasible in practice. The VAEGAN-based model can synthesize a wide variety of airfoils that serve as candidates for further optimization through CFD simulation as we will investigate in Section 4.6.

### 4.5 Aerodynamic Properties of Synthesized Airfoil

Aerodynamic properties of airfoils are essential in airfoil design. Synthesized airfoils are expected to meet certain aerodynamic properties to guarantee the designs are feasible and effective. To evaluate the aerodynamics, Xfoil is utilized to compute the lift coefficient, $C_l$, and the drag coefficient, $C_d$. The experiments on Xfoil are set for a low-speed condition: Reynolds number $Re = 2 \times 10^6$, Mach number $Ma = 0.02$, and attack angle $\alpha = 0^\circ$. As illustrated in Fig. 12a, $C_l$ and $C_d$ are tested on three airfoils: NACA1412, NACA2424 and NACA4415 from the UIUC database. Following Eq. 10, triplex interpolation/extrapolation is conducted with feature vectors, $z_1$ and $z_2$, encoded from the three NACA airfoils. The interpolated airfoils, marked by green dots, possess features from different clusters. Some airfoils synthesized by extrapolation have high $C_l$ with low $C_d$ located at the upper part of Fig. 12a. This demonstrates that by interpolation/extrapolation in the feature domain, novel airfoils with promising aerodynamic properties can be synthesized. Also, aerodynamic properties of synthesized airfoils by sampling are tested in comparison with some airfoils from the UIUC database.

---

1 https://web.mit.edu/drela/Public/web/xfoil/
as shown in Fig 12b. Though the generated airfoils from Gaussian noises are not guaranteed to have good aerodynamic properties, like red dots lying on the bottom left. Some promising airfoils can be synthesized, as shown on the top right, with a high lift coefficient and a low drag coefficient. By sampling, the VAEGAN-based model can synthesize airfoils with a wide variety of aerodynamic properties. Such a variety provides abundant candidates to explore in design space for airfoil shape optimization.

4.6 Shape Optimization on Aerodynamic Properties via Genetic Algorithm

The VAEGAN model has been proven to be able to parameterize existing airfoils to latent feature vectors and synthesize novel airfoils automatically. However, whether or not the learned features and synthesized airfoils can be optimized to possess desired aerodynamic properties remains untested. To this end, this section demonstrates that with the VAEGAN model, airfoil shapes can be optimized to target lift and drag coefficients, \( C_L \) and \( C_D \), through a genetic algorithm (GA).

As illustrated in Fig. 13, the target lift coefficient is \( C_L = 0.6 \), and the target drag coefficient is \( C_D = 0.06 \). In our case, the total number of generations, \( N \), is set to be 60, and the number of populations on each generation, \( M \), to be 25. Fig. 13a depicts the evolution of average score by generation when different generative methods are applied. PCA fails to generate optimized airfoils with target aerodynamic properties. On the other hand, airfoils generated from deep generative models (i.e., VAE and VAEGAN) can be optimized toward target \( C_L \) and \( C_D \). However, under the same number of generations and populations, VAEGAN demonstrates better performance in generating optimized airfoil designs with higher fitness scores. Fig. 13b show how \( C_L \) and \( C_D \) change by generation using VAEGAN. As Fig. 13c illustrates, the airfoil shape gradually evolves to the target \( C_L \) and \( C_D \). Also, we compare the performance of airfoil optimization using different featurization techniques, PCA, and VAE. The genetic algorithm with the same objective function and settings are conducted. Moreover, Table 1 lists the performance of different featurization methods when different target lift and drag coefficients \( C_L \) and \( C_D \) are applied. The mean and standard deviation of lift coefficients, drag coefficients, and fitness scores of the last generation are reported. For example, for \( C_L = 0.6 \) and \( C_D = 0.006 \), the VAEGAN-synthesized airfoils reach an average lift coefficient of 0.5857 and an average drag coefficient of 0.0061. The coefficients are close to the desired aerodynamic properties. Whereas PCA and VAE fail to synthesize desired airfoils within the same number of generations and population size. Also, it is shown that with other target aerodynamic properties, VAEGAN outperforms the other models in generating optimized airfoil designs. This is because our VAEGAN-based model generates a wider variety of airfoils that serves as potential candidates in design optimization. Such experiments prove that a simple optimization technique like GA and the well-trained VAEGAN model can synthesize airfoils with desired aerodynamic properties, which can guide the design of effective and efficient aerodynamic products.

5. Conclusion

In this work, a data-driven method is proposed to achieve three goals: (1) automatically featuring airfoil geometries from the UIUC Database without manually designed parameters, (2) synthesizing novel airfoils by either interpolating or extrapolating the encoded features, as well as generating from random noise, and (3) optimizing the features to synthesize airfoils with desired
aerodynamic properties. Our model is built upon VAEGAN, which combines the encoder-decoder architecture from VAE and the discriminator from GAN. With the encoder-decoder structure, our model learns explicit mappings from airfoil coordinates to latent feature domain as well as from feature vectors to airfoils, while with the discriminator, the model can automatically synthesize realistic samples. Also, our model is trained in a self-supervised manner. Namely, the model learns compact and informative features directly from airfoil shapes without manually tagged labels or designed parameters. Optimized on the learned feature domain via GA, the synthesized airfoils can evolve to have desired aerodynamic properties.

Experiments show that our model learns compact and comprehensive features encoding shape information of airfoils and can automatically generate novel airfoils. First, airfoils can be reconstructed via decoding the learned features with minor errors compared to the origin coordinates. Second, K-Means clustering on the feature domain of the UIUC Coordinate Database further demonstrates the learned representations are meaningful in a way that the centroid of each cluster represents different shapes. It is also investigated what is encoded in each dimension of the feature domain by gradually changing the feature vector on one specific dimension with all other dimensions fixed. Without human prior, each dimension encodes different geometric information like height, camber, symmetry, and even coupled features. Moreover, novel airfoils are synthesized by interpolating and extrapolating learned features from different airfoils as well as directly generated from random noise. By interpolating or extrapolating, the synthesized airfoil inherits and blends features from existing airfoils, which provides insights for designing new airfoils. On the other hand, airfoils generated from Gaussian noise are more aggressive in a way that they follow a less geometrical format of existing airfoils, and more novelty is introduced to the airfoil design. Finally, the synthesized airfoils can be optimized via GA to possess competitive or even better aerodynamic properties in comparison to existing ones, indicating the synthesized geometries are not only plausible in shape but also practical in aerodynamic performance.

Compared to the other featurization methods, like PCA and VAE, our model performs better in synthesizing realistic airfoil shapes and encoder more representative geometric features. Besides, our method synthesizes airfoils without pre-defined polynomials or splines formula and demonstrates deep generative models as an efficient tool for airfoil design. In the scope of this paper, we only focus on the design of 2D airfoils. Extending the deep generative models to 3D airplane wing design is of great interest to investigate. Besides, airfoil design optimization on the learned latent feature can be further investigated. For example, optimizing the airfoils under varying aerodynamic conditions. Deep reinforcement learning, which learns an optimal trajectory towards the objective, can also be implemented in optimizing the aerodynamic shape design on the latent feature domain. In UIUC database, all airfoils share the same length of 1. Including the length as a design parameter to optimize the performance of airfoils can be a valuable extension. Other optimization methods, like Bayesian optimization (Snoek et al., 2012), may be applied to optimize the generated airfoil designs more efficiently and reduce the computational costs of simulations to evaluate the aerodynamic performance. In future works, the proposed deep generative method could be further extended to airfoil designs under transonic or high-speed flow.

**Acknowledgements**

The authors thank the start-up fund provided by the Department of Mechanical Engineering at Carnegie Mellon University.

**Conflict of interest statement**

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

**References**

Abbott, I. H., Von Doenhoff, A. E., & Stivers Jr, L. (1945). Summary of airfoil data.

Achour, G., Sung, W. J., Pinon-Fischer, O. J., & Mavris, D. N. (2020). Development of a conditional generative adversarial network for airfoil shape optimization. AIAA Scitech 2020 Forum, 2261. https://doi.org/10.2514/6.2020-2261

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein generative adversarial networks. In D. Precup & Y. W. Teh (Eds.), Proceedings of the 34th international conference on machine learning (pp. 214–223). PMLR.

Boehm, W., & Müller, A. (1999). On de casteljau’s algorithm. Computer Aided Geometric Design, 16(7), 587–605. https://doi.org/10.1016/S0167-8396(99)00023-0

Cai, L., Gao, H., & Ji, S. (2019). Multi-stage variational auto-encoders for coarse-to-fine image generation. Proceedings of the 2019 SIAM International Conference on Data Mining, 630–638.

Cambria, E., & White, B. (2014). Jumping nlp curves: A review of natural language processing research. IEEE Computational Intelligence magazine, 9(2), 48–57. https://doi.org/10.1109/MCI.2014.2307227

Campbell, R. L. (2003). An approach to constrained aerodynamic design with application to airfoils. NASA Langley Technical Report Server. https://doi.org/10.5555/887353

---

**Table 1: Airfoil design optimization results:** $C_l$, $C_d$, and fitness score, through different featurization methods for different target $C_l$ and $C_d$.

| $C_l$ | $C_d$ | Featurization | mean | std | mean | std | mean | std | Fitness score mean | std |
|-------|-------|---------------|------|-----|------|-----|------|-----|-------------------|-----|
| 0.6   | 0.006 | PCA           | 0.41953 | 0.012672 | 0.0069430 | 6.9231 | $10^{-4}$ | -0.12893 | 0.046250 |
|       |       | VAE           | 0.52746 | 0.0017073 | 0.0056662 | 1.4631 | $10^{-5}$ | -0.017819 | 0.0014068 |
|       |       | VAEGAN        | 0.58570 | 0.0028946 | 0.0061030 | 7.0711 | $10^{-6}$ | -8.5312 | 0.00089211 |
| 0.6   | 0.005 | PCA           | 0.40342 | 0.0119782 | 0.0066422 | 7.5323 | $10^{-4}$ | -0.23431 | 0.064320 |
|       |       | VAE           | 0.53245 | 0.0015981 | 0.0055432 | 1.1176 | $10^{-5}$ | -0.025632 | 0.014233 |
|       |       | VAEGAN        | 0.57230 | 0.002061 | 0.0053033 | 1.5985 | $10^{-5}$ | -0.0057176 | 0.0012436 |
| 0.5   | 0.005 | PCA           | 0.44243 | 0.024322 | 0.0060134 | 7.4312 | $10^{-4}$ | -0.056324 | 0.067883 |
|       |       | VAE           | 0.49186 | 0.001168 | 0.0053432 | 2.2679 | $10^{-5}$ | -0.0051764 | 0.0012432 |
|       |       | VAEGAN        | 0.50587 | 0.0012456 | 0.0051879 | 5.6709 | $10^{-6}$ | -0.0015501 | 0.00086793 |
Carrio, A., Sampedro, C., Rodriguez-Ramos, A., & Campoy, P. (2017). A review of deep learning methods and applications for unmanned aerial vehicles. Journal of Sensors, 2017. https://doi.org/10.1155/2017/2963874

Chang, K.-H., & Chen, C. (2011). 3d shape engineering and design parameterization. Computer-Aided Design and Applications, 8(5), 681–692. http://dx.doi.org/10.3722/cadaps.2011.681-692

Chen, W., Chiu, K., & Fuge, M. (2019). Airfoil design parameterization and optimization using bézier generative adversarial networks. AIAA Journal, 58(11), 4723–4735. https://doi.org/10.2514/1.J059317

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., & Abbeel, P. (2016). Infogan: Interpretable representation learning by information maximizing generative adversarial nets. Advances in neural information processing systems, 2172–2180. https://doi.org/10.5555/3157096.3157340

Constantine, P. G., Dow, E., & Wang, Q. (2014). Active subspace. SIAM Journal on Scientific Computing, 36(4), A1500–A1524. https://doi.org/10.1137/130916384

Coquillart, S. (1990). Extended free-form deformation: A sculpturing tool for 3d geometric modeling (Vol. 24). ACM. https://doi.org/10.1145/97879.97900

D’Agostino, D., Serani, A., Campana, E. F., & Diez, M. (2018). Deep autoencoder for off-line design-space dimensionality reduction in shape optimization. 2018 AIAA/ASCE/ASH/ASC Structures, Structural Dynamics, and Materials Conference, 1648.

Derksen, R., & Rogalsky, T. (2010). Bezier-parsec: An optimized aerofoil parameterization for design. Advances in engineering software, 41(7-8), 923–930. https://doi.org/10.1016/j.advengsoft.2010.05.002

Dietz, M., Campana, E. F., & Stern, F. (2015). Design-space dimensionality reduction in shape optimization by karhunen-loève expansion. Computer Methods in Applied Mechanics and Engineering, 283, 1525–1544. https://doi.org/10.1016/j.cma.2014.10.042

Du, X., He, P., & Martins, J. R. (2020). A b-spline-based generative adversarial network model for fast interactive airfoil aerodynamic optimization. AIAA Scitech 2020 Forum, 2128. https://doi.org/10.2514/6.2020-2128

Elliott, J., & Peraire, J. (1997). Practical three-dimensional aerodynamic design and optimization using unstructured meshes. AIAA Journal, 35(9), 1479–1485. https://doi.org/10.2514/2.271

Farin, G. (2014). Curves and surfaces for computer-aided geometric design: A practical guide. Elsevier. https://doi.org/10.1016/C2009-0-22251-8

Ghorbani, H., & Khameneifar, F. (2021). Airfoil profile reconstruction from unorganized noisy point cloud data. Journal of Computational Design and Engineering, 8(2), 740–755. https://doi.org/10.1093/jcde/qwab011

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 2672–2680. https://doi.org/10.5555/2969033.2969125

Grey, Z. J., Doronina, O. A., & Glaws, A. (2023). Separable shape tensors for aerodynamic design. Journal of Computational Design and Engineering, 10(1), 468–487. https://doi.org/10.1093/jcde/qwac140

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

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. Proceedings of the IEEE international conference on computer vision, 2961–2969. https://doi.org/10.1109/ICCV.2017.322

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. Proceedings of the IEEE International Conference on Computer Vision, 1026–1034. https://doi.org/10.1109/ICCV.2015.123

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems, 6626–6637. https://doi.org/10.5555/3295222.3295408

Hicks, R. M., & Henne, P. A. (1978). Wing design by numerical optimization. Journal of Aircraft, 15(7), 407–412. https://doi.org/10.2514/3.58379

Huang, H., Kalogerakis, E., & Marlin, B. (2015). Analysis and synthesis of 3d shape families via deep-learned generative models of surfaces. Computer Graphics Forum, 34(5), 25–38. https://doi.org/10.1111/cgf.12694

Jain, A. K. (2010). Data clustering: 50 years beyond k-means. Pattern recognition letters, 31(8), 651–666. https://doi.org/10.1016/j.patrec.2009.09.011

Jameson, A. (2006). Advances in aerodynamic shape optimization. Computational fluid dynamics 2004 (pp. 687–698). Springer. https://doi.org/10.1007/3-540-31801-1_100

Jameson, A., Martinelli, L., & Pierce, N. A. (1998). Optimum aerodynamic design using the navier-stokes equations. Theoretical and computational fluid dynamics, 10(1), 213–237. https://doi.org/10.1007/s001620050060

Kalogerakis, E., Chaudhuri, S., Koller, D., & Koltun, V. (2012). A probabilistic model for component-based shape synthesis. ACM Transactions on Graphics (TOG), 31(4), 55. https://doi.org/10.1145/2185520.2185551

Karras, T., Laine, S., & Aila, T. (2019). A style-based generator architecture for generative adversarial networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4401–4410. http://dx.doi.org/10.1109/CVPR.2019.00453

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. AIChE journal, 37(2), 233–243. https://doi.org/10.1002/aic.690370209

Larsen, A. B. L., Sønderby, S. K., Larochelle, H., & Winther, O. (2015). Autoencoding beyond pixels using a learned similarity metric. arXiv preprint arXiv:1512.09300.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436. https://doi.org/10.1038/nature14539

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. Neural computation, 1(4), 545–551. https://doi.org/10.1162/neco.1989.1.4.541

Li, J., Bouhlel, M. A., & Martins, J. R. (2015). Data-based approach to handwritten zip code recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 6401–6409. http://dx.doi.org/10.1109/CVPR.2015.7298648

Li, Y., Qi, H., Dai, J., Ji, X., & Wei, Y. (2017). Fully convolutional instance-aware semantic segmentation. Proceedings of the
Yilmaz, E., & German, B. (2020). Conditional generative adversarial network framework for airfoil inverse design. AIAA AVIATION 2020 Forum, 3185. https://doi.org/10.2514/6.2020-3185

Zhang, Y., Sung, W. J., & Mavris, D. N. (2018). Application of convolutional neural network to predict airfoil lift coefficient. 2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 1903. https://doi.org/10.2514/6.2018-1903