AIAA SciTech Forum
6-10 January 2020, Orlando, FL
AIAA Scitech 2020 Forum

10.2514/6.2020-2128

A B-Spline-based Generative Adversarial Network Model for Fast Interactive Airfoil Aerodynamic Optimization

Xiaosong Du*, Ping He†, and Joaquim R. R. A. Martins.‡
University of Michigan, Ann Arbor, MI, 48109, USA

Airfoil aerodynamic optimization is of great importance in aircraft design; however, it relies on high-fidelity physics-based models that are computationally expensive to evaluate. In this work, we provide a methodology to reduce the computational cost for airfoil aerodynamic optimization. Firstly, we develop a B-spline based generative adversarial networks (BSplineGAN) parameterization method to automatically infer design space with sufficient shape variability. Secondly, we construct multi-layer neural network (MNN) surrogates for fast predictions on aerodynamic drag, lift, and pitching moment coefficients. The BSplineGAN has a relative error lower than 1% when fitting to UIUC database. Verification of MNN surrogates shows the root means square errors (RMSE) of all aerodynamic coefficients are within the range of 20%–40% standard deviation of testing points. Both normalized RMSE and relative errors are controlled within 1%. The proposed methodology is then demonstrated on an airfoil aerodynamic optimization. We also verified the baseline and optimized designs using a high-fidelity computational fluid dynamic solver. The proposed framework has the potential to enable web-based fast interactive airfoil aerodynamic optimization.

I. Introduction

Aerodynamic optimization plays a key role in aircraft design because it effectively reduces the design period [1, 2]. However, both gradient-free [3–5] and gradient-based [6, 7] optimization algorithms rely on high-fidelity computational fluid dynamics (CFD) simulations that are computationally expensive to run. To reduce the computational budget and obtain fast optimization convergence, researchers have focused on two main branches: dimensionality reduction [8, 9], and surrogate modeling [10, 11].

On one hand, dimensionality reduction methods, such as principal component analysis and partial least squares, reduce the number of design variables by obtaining representative principal components. Moreover, advanced parameterization methods [12] including singular value decomposition and non-uniform rational B-spline are introduced to represent geometries with as few design variables as possible. On the other hand, surrogate models [13, 14], such as radial basis function and Gaussian regression process, have been widely used in various engineering areas for fast response estimations. These methods manage to alleviate the computational costs, however, they still suffer from these drawbacks [15, 16]: (1) dimensionality reduction methods lose part of available information as a trade-off, (2) typical parameterization methods have to guess the design variable limits which are always much larger than sufficient shape variability, (3) traditional surrogate models can hardly deal with large data set.

Generative adversarial networks (GAN) model was invented by Goodfellow et al. [17, 18] to generate new data with the same statistics as the training data. Goodfellow *et al*. [17, 18] successfully demonstrated this new conception on a series of machine learning data sets. They claimed the viability of the modeling framework and pointed out straightforward extensions including semi-supervised learning and efficiency improvements. Chen *et al*. [19] proposed an information-theoretic extension of GAN (InfoGAN) to learn disentangled representations in a completely unsupervised manner by maximizing mutual information between latent variables and training data observations. Chen *et al*. [15, 16] improved the InfoGAN to BezierGAN model for smooth shape representation and applied this approach to airfoil shape parameterization of aerodynamic optimization. BezierGAN model reduces the high dimensionality of Bezier representation to low-dimensional latent variables for optimization. Besides, BezierGAN model reduces design space by automatically inferring the boundary and keeping sufficient shape variability in the meantime. Results show that BezierGAN model accelerates the optimization convergence and generates smoother shapes than InfoGAN.

*Post-Doctoral Fellow, Department of Aerospace Engineering.
†Assistant Research Scientist, Department of Aerospace Engineering.
‡Professor, Department of Aerospace Engineering, AIAA Associate Fellow.

Neural networks [20, 21] surrogate models capture intricate structure of training data and handle large data set via batch optimization strategy, motivating breakthroughs in high-dimensional regression tasks, and processing images, audios, and videos. LeCun *et al*. [22] showed detailed insights and predicted the future of deep neural network methods including multi-layer neural networks (MNN), convolutional neural networks and recurrent neural networks. Raissi *et al*. [23, 24] proposed the physics-informed neural networks (PINN) to take advantage of the neural networks gradient and incorporate useful physics information from governing equations. They managed to demonstrate the proposed PINN model on flow field predictions. Zhu *et al*. [25] developed a physics-constrained neural to address constrains of data implied by partial differential equations, and demonstrated the model on high-dimensional unlabeled data.

In our previous work, we generated data-driven surrogate models, namely, gradient-enhanced Kriging with partial least squares [21, 26, 27], and gradient-enhanced MNN [28]. Surrogate models are both verified with sufficient accuracy, and successfully applied to our Webfoil online airfoil tool.* Webfoil is a web-based tool for fast interactive airfoil analysis and design optimization using any modern computer or mobile device. The completed work, however, defined large design space and filtered out unreasonable airfoil shapes through complex procedures. In addition, separate surrogate models with different numbers of parameterization variables were generated for subsonic and transonic regimes.

Continuing with previous work, we propose a B-spline-based GAN (BSplineGAN) model for Webfoil parameterization. BSplineGAN is an extension to the state-of-the-art BezierGAN airfoil parameterization method. After training with the UIUC airfoil database, the BSplineGAN automatically generates reasonable airfoil shapes with sufficient variability. The advantages of B-spline curves [29, 30] over Bezier curves provide BSplineGAN with a better shape control with fewer control parameters. Moreover, we construct one generalized MNN model for both subsonic and transonic regimes.

The rest of this paper is organized as follows. Section II describes the methods including BSplineGAN and MNN surrogate model used in this work. The optimization framework is demonstrated on an aerodynamic optimization case shown in Section III. Then we conclude the paper in Section IV.

II. Methodology

This section describes the general workflow of BSplineGAN, then steps into its key elements including B-spline parameterization, GAN model, BSplineGAN and surrogate modeling.

A. General Workflow

The BSplineGAN-based fast interactive aerodynamic optimization framework is summarized as follows (Fig. 1):

1) Starting with the UIUC airfoil database, we feed the existing airfoil shapes as training data into BSplineGAN model, where reasonable airfoils with sufficient variability are obtained. We add the B-spline layer onto the BSplineGAN generator module to enhance the smoothness of generated airfoils.
2) Apply Latin hypercube sampling (LHS) [31] on BSplineGAN input parameters for random generated airfoil shapes, which are fed together with operating conditions into the CFD solver, ADflow [†] in this work.
3) Use the training data set to construct MNN surrogate models.
4) Verify the surrogate model accuracy using verification metrics against testing data set, and determine whether the surrogate model is of sufficient accuracy.
5) If the surrogate model is sufficiently accurate we can start surrogate-based aerodynamic analysis and optimization. Otherwise, we re-sample a larger training data set, and repeat the process above until surrogate model has sufficient accuracy.

B. B-Spline Parameterization

B-spline curve is a generalization of Bezier curve [29, 30]. Moreover, B-spline curves provide more control flexibility and finer shape control because of the following reasons [29]:

1) The degree of B-spline curve is independent with the number of control points.
2) The strong convex hull property provides B-spline curves finer shape control.
3) Advanced techniques such as changing knots can be implemented for editing and designing shapes.

More details can be found in Piegl and Thiller [29].

*http://webfoil.engin.umich.edu

†https://github.com/mdolab/adflow

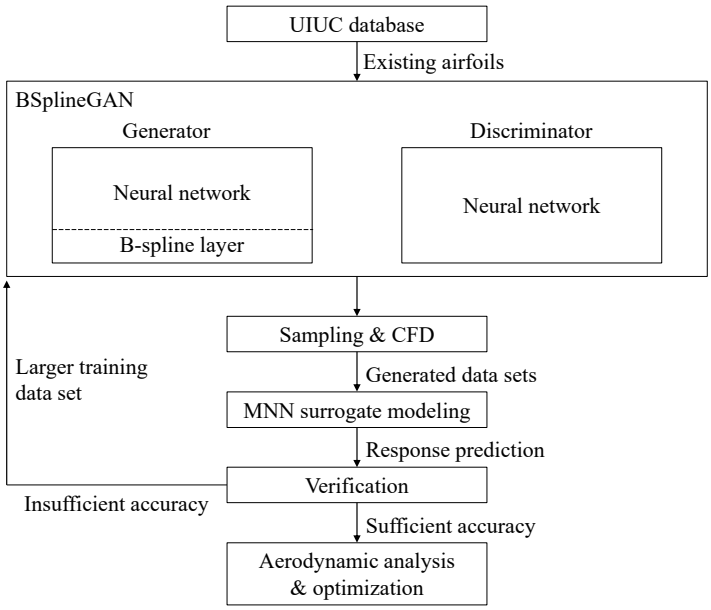


Fig. 1 BSplineGAN-based fast interactive aerodynamic optimization framework.

A B-spline curve is defined as

$$P(u) = \sum_{i=0}^{n} N_{i,k}(u) p_i, \tag{1}$$

where k is order of B-spline curve, u is knot within the range of [0, 1], p_i is the $(i+1)$ th control point, the total number of control points is $n+1$, $N_{i,k}$ is basis function and defined as

$$N_{i,1} = \begin{cases} 1 & u_i \le u \le u_{i+1}, \\ 0 & \text{otherwise}, \end{cases} \tag{2}$$

$$N_{i,k} = \frac{u - u_i}{u_{i+k-1} - u_i} N_{i,k-1}(u) + \frac{u_{i+k} - u}{u_{i+k} - u_{i+1}} N_{i+1,k-1}(u), \tag{3}$$

with the increasing knot vector $[u_0, ..., u_{n+k}]$ and $u_0 = 0, u_{n+k} = 1$ in this work.

B-spline curves are commonly used to represent airfoils [12]. We construct two distinct B-splines for upper and lower airfoil surfaces, separately. Each B-spline curve has two end control points fixed at leading edge (0, 0) and trailing edge (1, 0). The remaining control points of each surface are distributed on a half-cosine scale between (0, 1) along the chordwise direction and only allowed to vary in the vertical direction. The half-cosine scale is given as

$$p_{i,x} = \frac{1}{2}\left[1 - \cos\left(\frac{\pi(i-1)}{n+1}\right)\right]. \tag{4}$$

C. Generative Adversarial Networks and Key Variants

GAN model is a type of generative model, developed by Goodfellow *et al* [17]. to match the existing data statistics and patterns. As shown in Fig. 2, a GAN model consists of generator and discriminator neural networks. The former maps a set of input parameters with prior distributions, *i.e.* noise variables, into generated designs. The latter takes both existing data and generated designs as inputs, and output the probabilities of being real designs. The training process is typically seen as a competition between generator and discriminator. Specifically, discriminator is trained with existing data set to output **1** and with generated design to output **0**, while generator is trained to generate designs that are difficult for discriminator to judge. This process is mathematically formulated as a minimax problem

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim P_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim P_z}[\log(1 - D(G(z)))], \tag{5}$$

where x is sampled from existing data distribution P_{data}, z is sampled from the noise variable distribution P_{z}, and G and D are the generator and discriminator. In this way, a trained GAN model generates reasonable designs with sufficient shape variability within the prior noise variable distribution.

The noise variable z represents the design space, however, the relationship between the noise variable and generated shapes are entangled and disordered. The InfoGAN model (Fig. 3) was developed to solve this problem by decomposing design space into a set of semantically meaningful factors of variations. Specifically, InfoGAN model [19] uses two vectors of input variables: noise variable z representing the incompressible data information and latent variable c representing the salient structured semantic features of existing data set. Then we maximize a lower bound of the mutual information between c and generated designs. The mutual information lower bound is formulated as

$$L_1(G,Q) = \mathbb{E}_{x\sim P_G}\left[\mathbb{E}_{c'\sim P(c|x)}[\log Q(c'|x)]\right] + H(c), \tag{6}$$

where Q is the auxiliary distribution for approximating $P(c|x)$, $H(c)$ is the latent variable entropy which is viewed as a constant. Thus, the InfoGAN objective cost function is given as

$$\min_{G,Q}\max_{D} V(D,G) - \lambda L_1(G,Q), \tag{7}$$

where λ is a weighting factor.

BezierGAN model [15, 16] shares a similar structure as InfoGAN model except that a Bezier curve parameterization layer is added as output layer of generator neural networks. This Bezier layer synthesizes the control points, weighting factors, and parameter variables for a rational Beizer curve representation of airfoil shapes. Thus, the generator provides smooth airfoil shapes because of the Bezier layer, instead of simple discrete points provided by InfoGAN model. Besides these operations, BezierGAN objective cost function is regularized to avoid convergence to bad optima:

1) Regularize adjacent control points to keep them close via the corresponded average and maximum Euclidean distance

$$R_1(G) = \frac{1}{Nn}\sum_{j=1}^{N}\sum_{i=1}^{n+1}\|p_i^{(j)} - p_{i-1}^{(j)}\|_2, \tag{8}$$

$$R_2(G) = \frac{1}{N}\sum_{j=1}^{N}\max_i\|p_i^{(j)} - p_{i-1}^{(j)}\|_2, \tag{9}$$

where N is the sample size.

2) Regularize weighting factors w to eliminate unnecessary control points

$$R_3(G) = \frac{1}{N(n+1)}\sum_{j=1}^{N}\sum_{i=1}^{n+1}|w_i^{(j)}|, \tag{10}$$

3) Regularization to prevent highly non-uniform parameter variables

$$R_4(G) = \frac{1}{NM}\sum_{j=1}^{N}\sum_{i=0}^{M}\|a_i^{(j)} - 1\|_2 + \|b_i^{(j)} - 1\|_2, \tag{11}$$

where a and b are parameters of the Kumaraswamy distribution to obtain parameter variables, M is the number of Kumaraswamy cumulative distribution functions.

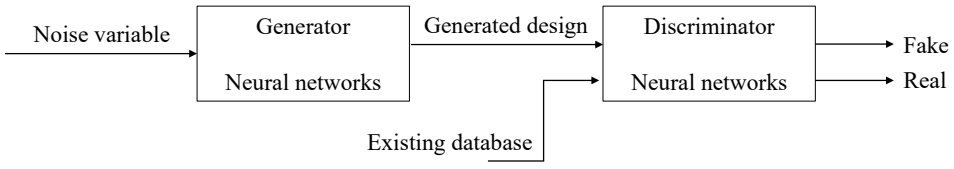


Fig. 2 GAN model architecture.

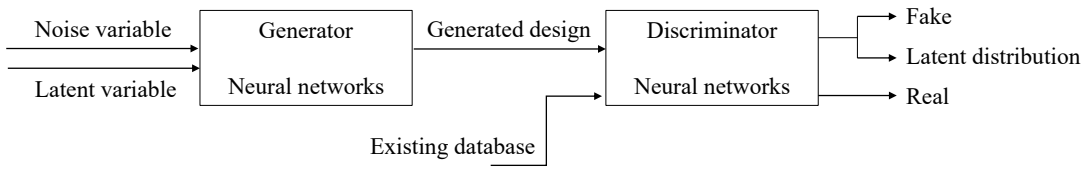


Fig. 3 InfoGAN model architecture.

Table 1 Neural network layers setup of generator and discriminator

Layers	Generator	Discriminator
L0	Fully connected layer, ReLU, batch normalization	Convolutional layer, ReLU, batch normalization, dropout=0.9
L1	Fully connected layer, ReLU, batch normalization	Convolutional layer, ReLU, batch normalization, dropout=0.9
L2	Deconvolutional layer, ReLU, batch normalization	Convolutional layer, ReLU, batch normalization, dropout=0.9
L3	Deconvolutional layer, ReLU, batch normalization	Fully connected layer, ReLU, batch normalization
L4	Deconvolutional layer, ReLU, batch normalization	Fully connected layer, no activation, no normalization
L5	Deconvolutional layer, Tanh, no normalization	
L6	B-spline parameterization layer	

Combining the above mentioned regularization terms, the objective function becomes

$$\min_{G,Q}\max_{D} V(D,G) - \lambda_0 L_1(G,Q) + \sum_{i=1}^{4} \lambda_i R_i(G). \tag{12}$$

D. B-Spline-Based Generative Adversarial Networks

The BSplineGAN model replaces the Bezier layer of BezierGAN model with a B-spline parameterization layer. As described in Section II.B, we use two separate B-spline curves sharing the x coordinates to represent the upper and lower airfoil surfaces. The B-spline layers takes control points generated by previous neural network layers to output smooth airfoil shapes. The neural network architectures of generator and discriminator are summarized in Table 1.

We add the following regularization terms to avoid bad converged optima

1) Regularize control points on each airfoil surface to keep them close by the average Euclidean distance between each adjacent control points

$$R_1(G) = \frac{1}{Nn}\sum_{j=1}^{N}\sum_{i=1}^{n+1} \|p_i^{(j)} - p_{i-1}^{(j)}\|_2, \tag{13}$$

2) Regularize the difference between upper and lower surface control points of the same x coordinates to avoid intersected airfoil shapes

$$R_2(G) = \frac{1}{Nn_s}\sum_{j=1}^{N}\sum_{i=1}^{n_s} \max(0, p_{l,i}^{(j)} - p_{u,i}^{(j)}), \tag{14}$$

where n_s is the number of control points on each surface. Thus, the objective cost function becomes

$$\min_{G,Q}\max_{D} V(D,G) - \lambda_0 L_1(G,Q) + \sum_{i=1}^{2} \lambda_i R_i(G). \tag{15}$$

We set λ_i as 1 in this work.

The advantages of BSplineGAN parameterization are summarized as follows

1) Share the properties of dimensionality reduction with sufficient shape variability as original GAN model.
2) Extract disentangled features of existing data for fast optimization convergence as described by Chen *et al*. [15, 16].

3) B-spline layer enables more control feasibility and finer shape control than Bezier layer.
4) Two separate B-spline curves force the leading and trailing edge of generated airfoil shapes going through points (0, 0) and (1, 0), respectively.

E. Multi-Layer Neural Networks Surrogate Modeling

In this work, the surrogate model input parameters are random input variables of BSplineGAN model and aerodynamic operating condition parameters, *i.e.*, Mach number (M), Reynolds number (Re), and angle of attack (α). We use the LHS scheme to sample the design space for training, validation, and testing data sets. We then obtain real model observations of all data sets using ADflow.

ADflow is a finite-volume structured CFD solver that is available under an open-source license. ADflow also has a discrete adjoint [32], overset mesh capability [33], and Newton-type solvers. The inviscid fluxes are discretized by using three different numerical schemes: the scalar Jameson–Schmidt–Turkel [34] (JST) artificial dissipation scheme, a matrix dissipation scheme based on the work of Turkel and Vatsa [35], and a monotone upstream-centered scheme for conservation laws (MUSCL) based on the work of van Leer [36] and Roe [37]. The viscous flux gradients are calculated by using the Green–Gauss approach. For turbulent RANS solutions, the Spalart–Allmaras [38] turbulence model is used to close the equations. To converge the residual equations, we use a Runge–Kutta (RK) algorithm, followed by an approximate Newton–Krylov (ANK) algorithm [39]. For all simulations we require the flow residuals to drop 14 order of magnitudes.

The quantities of interest in current work are drag coefficients (C_d), lift coefficient (C_l), and pitching moment coefficient C_m. We construct MNN surrogate models for C_d, C_l, C_m, separately. Each MNN model shares similar neural network architecture. The MNN construction process is shown in Fig. 4 and described as

1) Preprocess the input parameters with $MinMaxScaler$ within $SKlearn$ toolbox.
2) Build up multiple-hidden-layer neural networks, each layer of which ends with $ReLU$ activation function.
3) Set the cost function as the RMSE between training data observations and MNN predictions.
4) Train MNN model using Adam optimizer via batch optimization strategy.
5) Monitor the RMSE of training and validation data sets for the convergence of MNN model training.

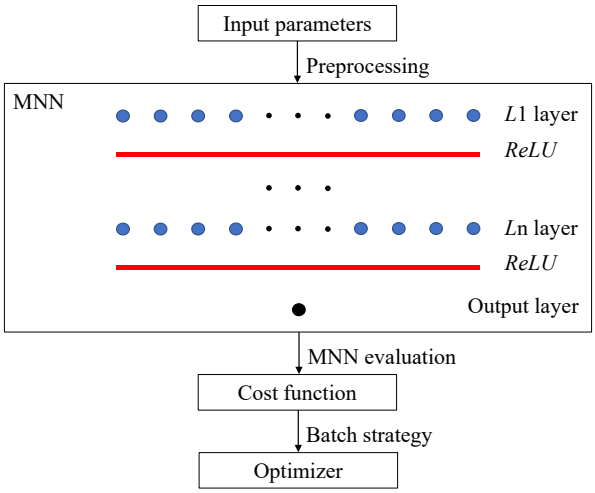


Fig. 4 Construction process of MNN surrogate model.

F. Verification

To check the accuracy of trained MNN surrogate model from various perspectives, we select root mean squared error (RMSE), normalized RMSE (NRMSE), and relative error as verification metrics, which are defined as follows

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N_t}(Y_{\text{pred}} - Y_{\text{real}})^2}{N_{\text{testing}}}}, \tag{16}$$

$$\text{NRMSE} = \frac{\text{RMSE}}{\max(Y_{\text{real}}) - \min(Y_{\text{real}})}, \tag{17}$$

$$\text{Rel. error} = \sum_{i=1}^{N_t}(Y_{\text{pred}} - Y_{\text{real}})^2 / \sum_{i=1}^{N_t}(Y_{\text{real}})^2, \tag{18}$$

where N_{testing} is the number of testing points, Y_{pred} is surrogate model prediction, Y_{real} is real model observation.

If RMSE is within one standard deviation of testing points, σ_{testing}, the surrogate model is relatively good. RMSE within $10\%\sigma_{\text{testing}}$ is a sign for a good surrogate model. NRMSE and Rel. error, as relative verification metrics, are expected to be within 1%.

III. Results and Discussion

In this section, we use the proposed approach to perform aerodynamic shape optimization. To this end, we generate CFD sample points and feed them into the MNN surrogate model to prediction aerodynamics. To ensure numerical accuracy, we conduct grid convergence studies, parametric studies about selecting the B-spline order, the number of control points, and the number of latent variables, and the MNN surrogate verification. Finally, we incorporate the MNN surrogate model into a gradient-based optimization framework and demonstrate a transonic airfoil aerodynamic optimization.

A. Grid convergence study

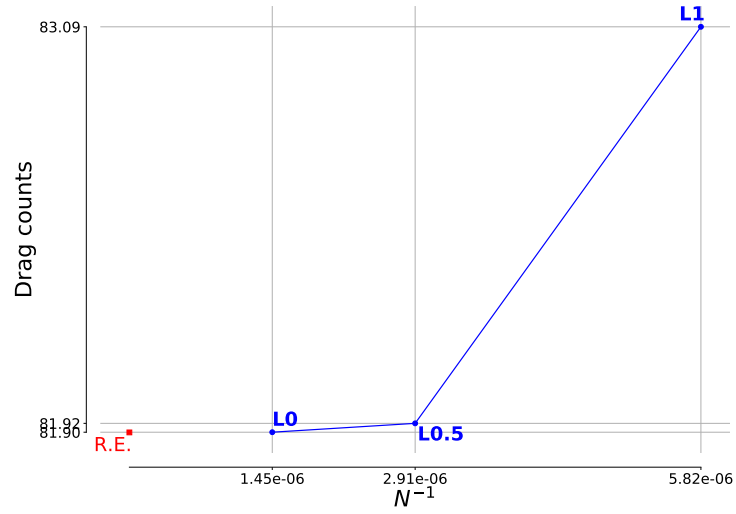
Since the MNN surrogate model is generated for both subsonic and transonic regimes, we run grid convergence studies for both types of cases, following [40]. We set up two set of grids for incompressible ($Ma < 0.3$) and compressible ($Ma \geq 0.3$) cases, and use a convergence threshold of 0.1 drag counts. Figure 5 shows the grid convergence study results on two typical aerodynamic optimization cases. In particular, Fig. 5(a) shows the NACA 0012 airfoil validation case, where Ma is 0.15, Re is 6×10^6, chord length of 1 m, and a C_l at target of 0.0. Table 2 shows the CFD results, showing a convergence between L0 and L0.5 grids. Figure 5(b) shows the RAE2822 case, where Ma is 0.725, Re is 6.5×10^6, chord length of 1 m, and a C_l at target of 0.824. Table 3 shows the CFD results, showing a convergence between L0 and L1 grids. There, we use L0.5 and L1 for subsonic and transonic cases, respectively.

Table 2 Grid convergence for the incompressible case. We use the L0.5 mesh for generated samples.

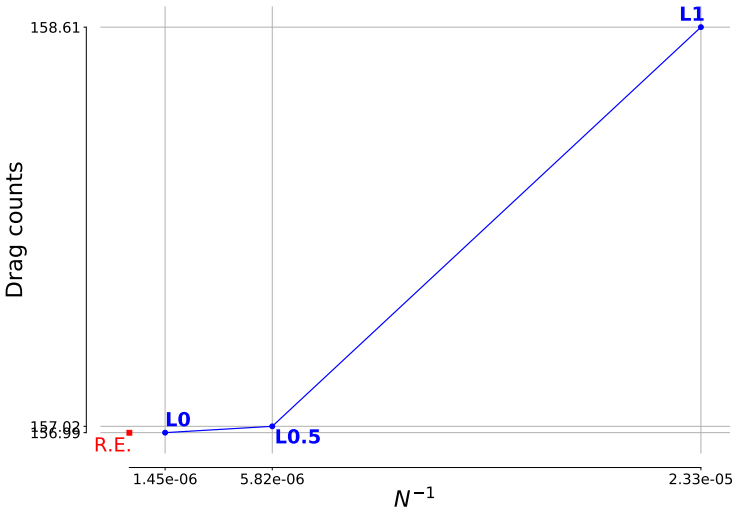
	Mesh size	α	C_l	C_d
L0	687,616	0.0	0.0	0.0081896
L0.5	343,808	0.0	0.0	0.0081922
L1	171,904	0.0	0.0	0.0083086

Table 3 Grid convergence for the compressible case. We use the L1 mesh for generated samples.

	Mesh size	α	C_l	C_d
L0	687,616	2.8825	0.823999	0.0156983
L1	171,904	2.8516	0.823999	0.0156985
L2	42,976	2.8197	0.823999	0.0158520



(a)



(b)

Fig. 5 Grid convergence study: (a) NACA 0012 case which has a convergence order of 10.97; (b) RAE2822 case which has a convergence order of 5.98.

B. Parametric study

Figure 6 shows the parametric study of mean relative L_1 norm [27] with respective to the order of B-spline curve and the number of control points for selected order. The mean relative L_1 norm is obtained by fitting B-spline curve to 1503 airfoils in UIUC database. Masters *et al*. [12] suggests a maximum B-spline order of 15, however, Fig. 6(a) shows a considerable accuracy increase using a order of 18 with maximum control points. Therefore, we set the order of both lower and upper airfoil surfaces as 18. Figure 6(b) shows the parametric study with respective to the total number of control points, and 32 control points have sufficient accuracy. Therefore, the B-spline layer of BSplineGAN generator has an order of 18 and 16 control points on each airfoil surface. We fix the number of BSplineGAN noise variables as 10. Figure 7 shows parametric study with respect to the number of latent variables. We use 16 latent

variables because of the fitting accuracy within 1%. Figure 8 shows a comparison between B-spline curve and the trained BSplineGAN parameterization methods. In particular, Fig. 8(a) shows randomly generated shapes using the B-spline layer of BSplineGAN, directly. The control points are set within the ranges of [-0.01, 0.10] and [-0.10, 0.01] for upper and lower airfoil shapes, respectively. Figure 8(b) has the randomly generated airfoils using the trained BSplineGAN model. Prior distributions of BSplineGAN variables are given in Table 4.

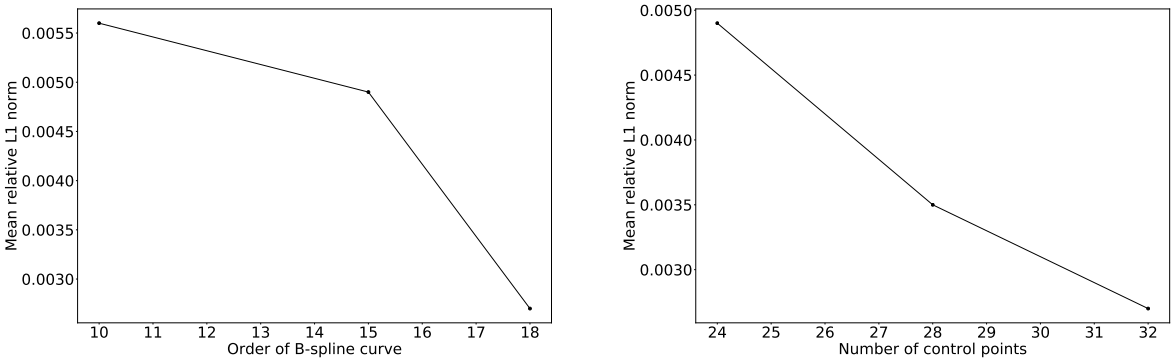


Fig. 6 B-spline parametric study of mean L1 norm w.r.t.: (a) B-spline order, where we use the 18-th order; (b) the number of control points, where we select the maximum number of control points for two separate 18-th order B-spline curves.

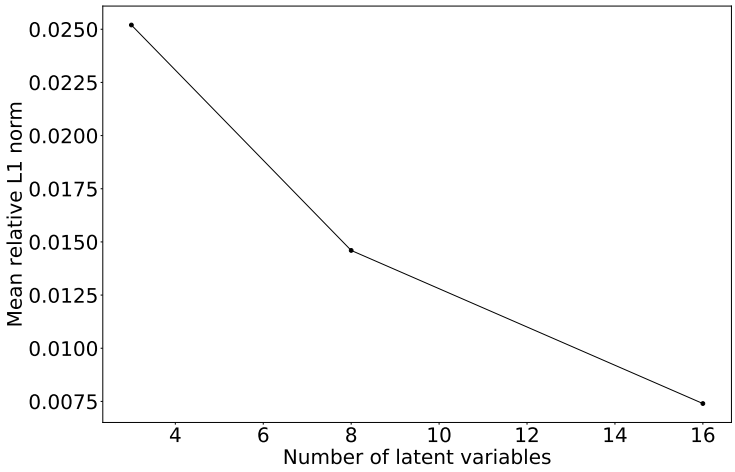
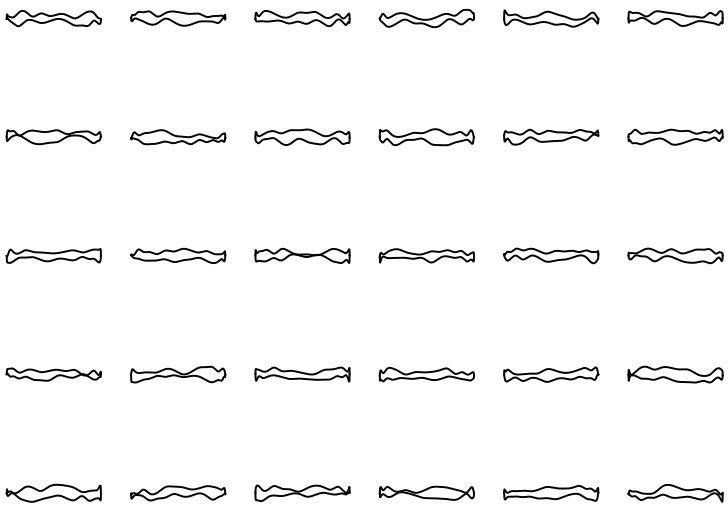
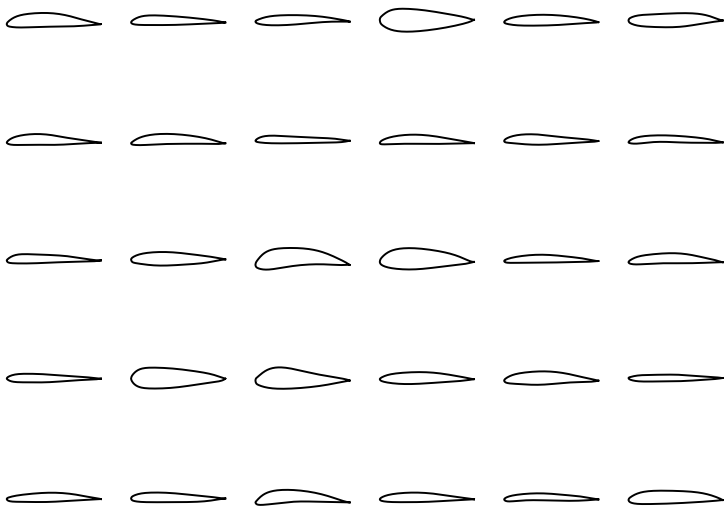


Fig. 7 Parametric study of BSplineGAN latent variables. We select 16 latent variables which reduces the mean relative L1 norm within 1%.



(a)



(b)

Fig. 8 Comparison between randomly generated shapes using: (a) B-spline curve; (b) BSplineGAN parameterization.

C. Accuracy verification

Having decided the mesh density, B-spline order and the number of control points and latent variables, we generate the CFD samples using ADflow. The distributions of input parameters are given in Table 4. We generate 8000 LHS points as training data set, 100 as validation set, and 1000 as testing set. MNN surrogate models of C_d, C_l, and C_m have an architecture of four, four, and three layers, respectively.

Key verification metrics are shown in Tables 5 to 7. NRMSE results of all three aerodynamic coefficients are within 5%, and all relative errors are well controlled within 1%. RMSE values vary between 20% to 40% meaning good global surrogate models [11]. Figures 9 to 11 show visual comparisons between MNN surrogate models and testing data sets. The mean absolute errors of C_d, C_l, and C_m are 37.863 counts, 476.79 counts, 258.515 counts, respectively. They have not reached the accuracy level of our previous work [26]. We speculate this is because we have only 8000 training points for 29 input parameters. We will generate more samples to improve the accuracy.

Table 4 Input parameter setup.

16 latent variables	10 noise variables	Ma	Re	α
Uniform(0, 1)	Normal(0, 0.5^2)	Uniform(0, 0.9)	Uniform(1E4, 1E10)	Uniform(0, 3) deg

Table 5 Key verification metric about C_d

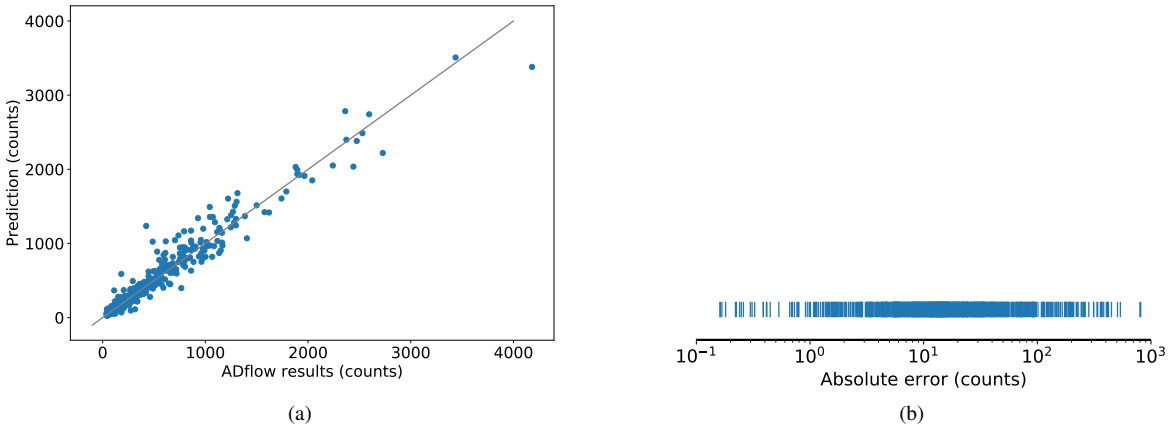
RMSE	σ_{testing}	NRMSE	Rel. Error
0.008314	0.039618	2.0%	0.53%

Table 6 Key verification metric about C_l

RMSE	σ_{testing}	NRMSE	Rel. Error
0.074643	0.276308	3.43%	0.47%

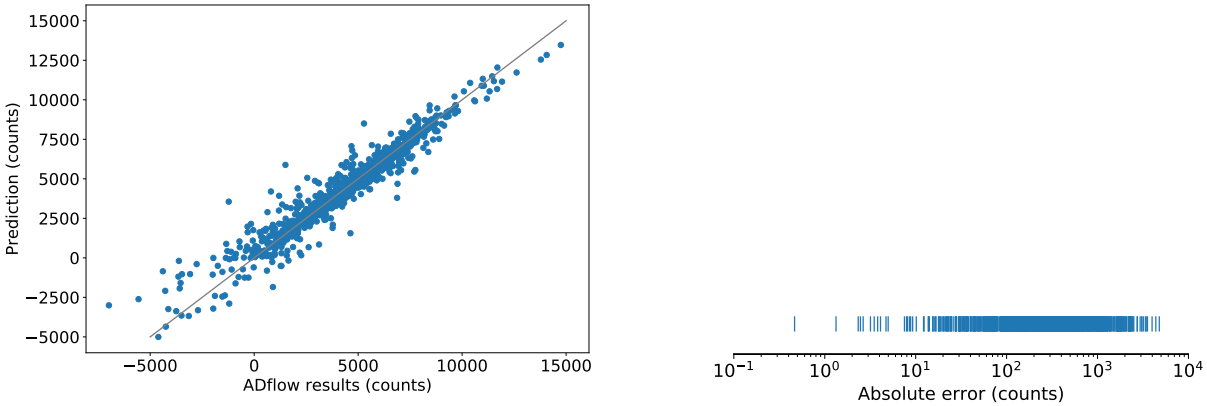
Table 7 Key verification metric about C_m

RMSE	σ_{testing}	NRMSE	Rel. Error
0.045355	0.123205	4.26%	0.70%



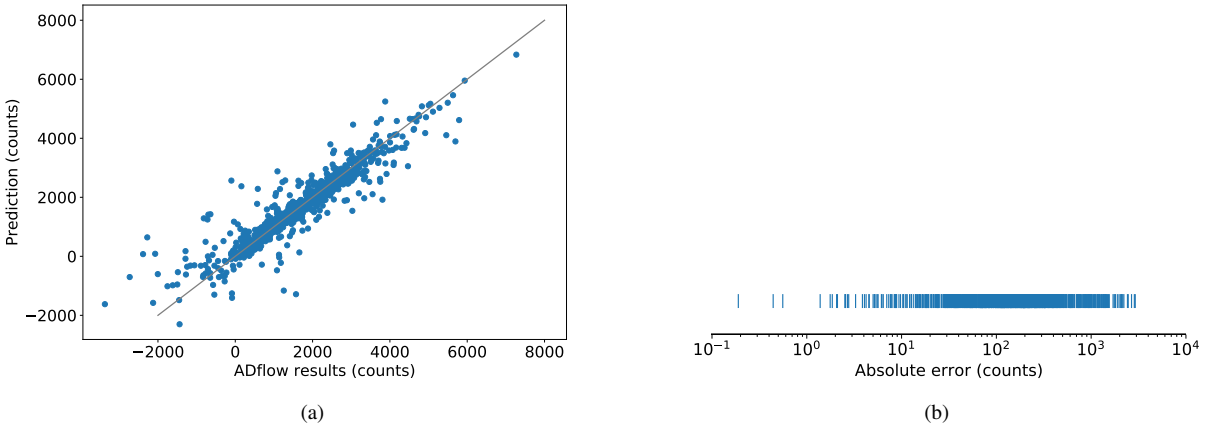
(a) (b)

Fig. 9 Validation of C_d: (a) prediction *vs.* ADflow results; (b) absolute error.



(a) (b)

Fig. 10 Validation of C_l: (a) prediction *vs.* ADflow results; (b) absolute error.



(a) (b)

Fig. 11 Validation of C_m: (a) prediction *vs.* ADflow results; (b) absolute error.

D. Aerodynamic shape optimization

We use the trained MNN surrogate model to perform a constrained aerodynamic shape optimization. The optimization configuration is summarized in Table 8. The baseline airfoil is NACA 0012. The objective function is C_d. The design variables are the 26 B-Spline control points that morph the airfoil shape, along with the angle of attack. We constrain the lift coefficient to be equal to 0.5. In addition, we constrain the area of the airfoil to be equal to or larger than 80% of its baseline value. The flow condition is at $Ma = 0.734$ and $Re = 6.5 \times 10^6$.

We use an open-source Python package pyOptSparse[‡] to setup the optimization problem. The SNOPT [41] optimizer is used, which adopts the sequential quadratic programming (SQP) algorithm for optimization. C_d and C_l are predicted by using the MNN surrogate model, and their derivatives are computed by using the finite-difference method.

The optimization results are summarized in Table 9. We obtain a 67.3% drag reduction in C_d. To confirm the drag reduction, we run high-fidelity CFD simulations for the baseline and optimized designs using ADflow. The drag reduction predicted by ADflow is 64.2%, 3.1% lower than that predicted by MNN. The optimized C_d value predicted by ADflow is 2.9 count higher than that predicted by MNN. However, for the baseline design, the C_d value predicted by MNN is 16.9 count higher than ADflow. We speculate the relatively large error is primarily due to the limited sample size (8000 sample points) used in this study. In the future, we will increase the number of sample points to improve the accuracy, as mentioned before. In addition, we will implement an analytical approach to compute derivatives, as opposed to the finite-difference method, for better speed and accuracy.

Figure 12 shows the comparison of pressure and airfoil profiles between the baseline and optimized designs. The optimized design uses a relatively flat upper surface to reduce the intensity of shock, which eventually reduces the drag. This can be further confirmed by comparing the pressure contours between the baseline and optimized designs, as shown in Fig. 13.

Table 8 Aerodynamic optimization setup for the NACA 0012 airfoil, which has 27 design variables and 2 constraints.

	Function or variable	Description	Quantity
minimize	C_D	Drag coefficient	
with respect to	y	Coordinates of B-Spline control points	26
	α	Angle of attack	1
		Total design variables	**27**
subject to	C_L=0.5	Lift-coefficient constraint	1
	$A \geq 0.8A_{\text{baseline}}$	Minimum-area constraint	1
		Total constraints	**2**

Table 9 Comparison of baseline and optimized C_d and C_l computed by MNN and ADflow. The drag reduction predicted by MNN is qualitatively verified by ADflow.

	C_d	C_l
Baseline (MNN)	0.02830	0.5000
Optimized (MNN)	0.00924	0.5000
Baseline (ADflow)	0.02661	0.5000
Optimized (ADflow)	0.00953	0.5000

[‡]https://github.com/mdolab/pyoptsparse

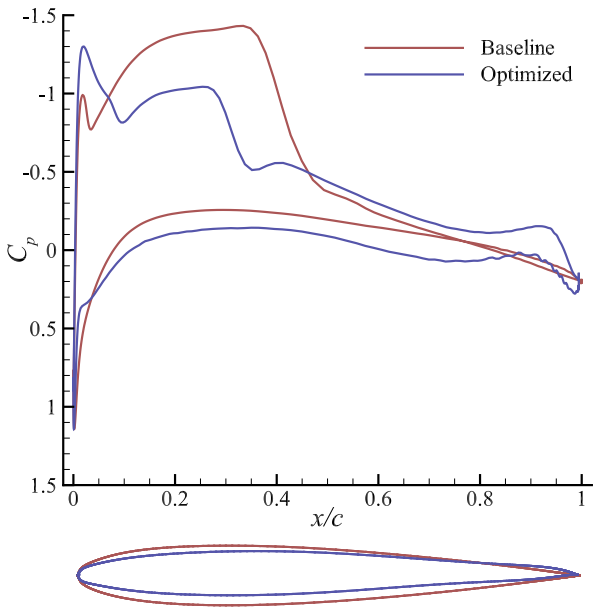


Fig. 12 Comparison of pressure and airfoil profiles between the baseline and optimized designs. The optimized design reduces the shock intensity for drag reduction.

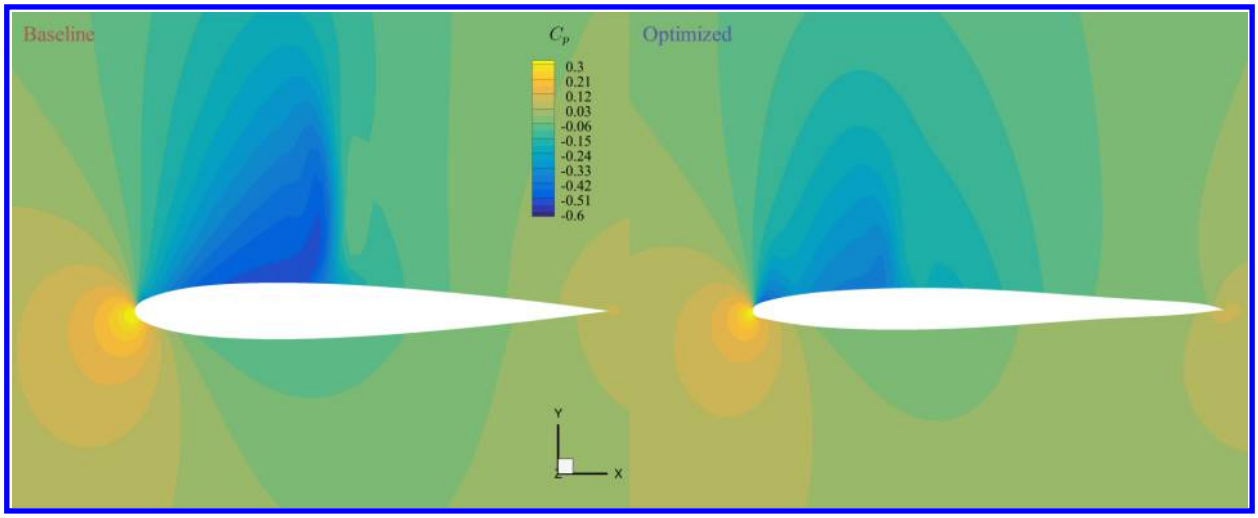


Fig. 13 Comparison of pressure contour between the baseline and optimized designs. The optimized design reduces the shock intensity for drag reduction.

IV. Conclusion

In this work, we proposed a fast-response aerodynamic optimization methodology. We developed the BSplineGAN parameterization approach based on the stat-of-the-art BezierGAN method. The BSplineGAN parameterization provides more control feasibility and finer shape control. Besides, BSplineGAN automatically infers a reduced design space with sufficient shape variability. Multi-layer neural networks surrogate models were constructed for fast prediction of aerodynamic coefficients. Optimization results showed the potentiality of this conception. We are currently running a larger data set to further improve the accuracy of completed work. The proposed methodology has the potential to improve the current Webfoil toolbox on fast interactive airfoil aerodynamic optimization.

References

[1] Jameson, A., and Vassberg, J., "Computational fluid dynamics for aerodynamic design—Its current and future impact," *39th Aerospace Sciences Meeting and Exhibit*, American Institute of Aeronautics and Astronautics, Reno, NV, 2001. https://doi.org/10.2514/6.2001-538.

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

[3] Mitchell, M., *An introduction to genetic algorithms*, MIT press, 1998.

[4] Kennedy, J., "Particle swarm optimization," *Encyclopedia of machine learning*, 2010, pp. 760–766.

[5] Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, Vol. 13, No. 4, 1998, pp. 455–492.

[6] Boggs, P. T., and Tolle, J. W., "Sequential quadratic programming," *Acta numerica*, Vol. 4, 1995, pp. 1–51.

[7] Nocedal, J., and Wright, S. J., *Numerical Optimization*, 2nd ed., Springer-Verlag, 2006.

[8] Chen, X., Diez, M., Kandasamy, M., Zhang, Z., Campana, E. F., and Stern, F., "High-fidelity global optimization of shape design by dimensionality reduction, metamodels and deterministic particle swarm," *Engineering Optimization*, Vol. 47, No. 4, 2015, pp. 473–494.

[9] Diez, M., Campana, E. F., and Stern, F., "Design-space dimensionality reduction in shape optimization by Karhunen–Loève expansion," *Computer Methods in Applied Mechanics and Engineering*, Vol. 283, 2015, pp. 1525–1544.

[10] Peherstorfer, B., Willcox, K., and Gunzburger, M., "Survey of Multifidelity Methods in Uncertainty Propagation, Inference, and Optimization," *SIAM Review*, Vol. 60, No. 3, 2018, pp. 550–591. https://doi.org/10.1137/16M1082469.

[11] Koziel, S., and Leifsson, L. (eds.), *Surrogate-Based Modeling and Optimization*, Springer New York, 2013. https://doi.org/10.1007/978-1-4614-7551-4, URL https://doi.org/10.1007/978-1-4614-7551-4.

[12] Masters, D. A., Taylor, N. J., Rendall, T., Allen, C. B., and Poole, D. J., "A Geometric Comparison of Aerofoil Shape Parameterisation Methods," *54th AIAA Aerospace Sciences Meeting*, American Institute of Aeronautics and Astronautics, 2016. https://doi.org/10.2514/6.2016-0558.

[13] Chaudhuri, A., Marques, A. N., Lam, R., and Willcox, K. E., "Reusing Information for Multifidelity Active Learning in Reliability-Based Design Optimization," *AIAA Scitech 2019 Forum, AIAA SciTech Forum, (AIAA 2019-1222)*, 2019. https://doi.org/doi:10.2514/6.2019-1222.

[14] Koziel, S., Ciaurri, D. E., and Leifsson, L., "Surrogate-Based Methods," *Computational Optimization, Methods and Algorithms*, Springer Berlin Heidelberg, 2011, pp. 33–59. https://doi.org/10.1007/978-3-642-20859-1_3, URL https://doi.org/10.1007/978-3-642-20859-1_3.

[15] Chen, W., and Fuge, M., "BezierGAN: Auomatic Generation of Smooth Curves from Interpretable Low-Dimensional Parameters," *arXiv:1808.08871*, 2018.

[16] Chen, W., Chiu, K., and Fuge, M., "Aerodynamic Design Optimization and Shape Exploration using Generative Adversarial Networks," *AIAA SciTech Forum*, AIAA, San Diego, USA, 2019.

[17] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., "Generative Adversarial Nets," *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[18] Goodfellow, I., "NIPS 2016 Tutorial: Generative Adversarial Networks," *arXiv preprint arXiv:1701.00160*, 2016.

[19] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P., "Infogan: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets," *arXiv:1606.03657*, 2016.

[20] Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning*, MIT Press, 2016. URL http://www.deeplearningbook.org.

[21] Li, J., He, S., and Martins, J. R. R. A., "Data-driven Constraint Approach to Ensure Low-speed Performance in Transonic Aerodynamic Shape Optimization," *Aerospace Science and Technology*, Vol. 92, 2019, pp. 536–550. https://doi.org/10.1016/j.ast.2019.06.008.

[22] LeCun, Y., Bengio, Y., and Hinton, G., "Deep Learning," *Nature*, Vol. 521, 2015, pp. 436–444.

[23] Raissi, M., Perdikaris, P., and Karniadakis, G., "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations," *Journal of Computational Physics*, Vol. 378, No. 1, 2018, pp. 686–707.

[24] Raissi, M., "Deep Hidden Physics Models: Deep Learning of Nonlinear Partial Differential Equations," *Journal of Machine Learning Research*, Vol. 19, No. 1, 2018, pp. 932–955.

[25] Zhu, Y., Zabaras, N., Koutsourelakis, P., and Perdikaris, P., "Physics-Constrained Deep Learning for High-dimensional Surrogate Modeling and Uncertainty Quantification without Labeled Data," *Preprint submitted to Journal of Computational Physics*, 2019.

[26] Li, J., Bouhlel, M. A., and Martins, J. R. R. A., "A data-based approach for fast airfoil analysis and optimization," *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Kissimmee, FL, 2018. https://doi.org/10.2514/6.2018-1383.

[27] Li, J., Bouhlel, M. A., and Martins, J. R. R. A., "Data-based Approach for Fast Airfoil Analysis and Optimization," *AIAA Journal*, Vol. 57, No. 2, 2019, pp. 581–596. https://doi.org/10.2514/1.J057129.

[28] Bouhlel, M., He, S., and Martins, J. R. R. A., "Scalable Gradient-Enhanced Artifical Neural Networks for Airfoil Shape Design in Subsonic and Transonic Regimes," *Structural and Multidisciplinary Optimization (in press)*, 2019.

[29] Piegl, L., and Tiller, W., *The NURBS book*, Springer Science & Business Media, 2012.

[30] Lamousin, H. J., and Waggenspack Jr, W. N., "NURBS-based free-form deformations," *Computer Graphics and Applications, IEEE*, Vol. 14, No. 6, 1994, pp. 59–65.

[31] McKay, M. D., Beckman, R. J., and Conover, W. J., "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, Vol. 21, No. 2, 1979, pp. 239–245.

[32] Kenway, G. K. W., Mader, C. A., He, P., and Martins, J. R. R. A., "Effective Adjoint Approaches for Computational Fluid Dynamics," *Progress in Aerospace Sciences*, Vol. 110, 2019, p. 100542. https://doi.org/10.1016/j.paerosci.2019.05.002.

[33] Kenway, G. K. W., Secco, N., Martins, J. R. R. A., Mishra, A., and Duraisamy, K., "An Efficient Parallel Overset Method for Aerodynamic Shape Optimization," *Proceedings of the 58th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, AIAA SciTech Forum*, Grapevine, TX, 2017. https://doi.org/10.2514/6.2017-0357.

[34] Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solution of the Euler Equations by Finite Volume Methods Using Runge–Kutta Time Stepping Schemes," *14th Fluid and Plasma Dynamics Conference*, 1981. https://doi.org/10.2514/6.1981-1259.

[35] Turkel, E., and Vatsa, V. N., "Effects of Artificial Viscosity on Three-Dimensional Flow Solutions," *AIAA Journal*, Vol. 32, 1994, pp. 39–45. https://doi.org/10.2514/3.11948.

[36] van Leer, B., "Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method," *Journal of Computational Physics*, Vol. 32, 1979, pp. 101–136. https://doi.org/10.1016/0021-9991(79)90145-1.

[37] Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, 1981, pp. 357–372. https://doi.org/10.1016/0021-9991(81)90128-5.

[38] Spalart, P., and Allmaras, S., "A One-Equation Turbulence Model for Aerodynamic Flows," *30th Aerospace Sciences Meeting and Exhibit*, 1992. https://doi.org/10.2514/6.1992-439.

[39] Yildirim, A., Kenway, G. K. W., Mader, C. A., and Martins, J. R. R. A., "A Jacobian-free approximate Newton–Krylov startup strategy for RANS simulations," *Journal of Computational Physics*, Vol. 397, 2019, p. 108741. https://doi.org/10.1016/j.jcp.2019.06.018.

[40] Jespersen, D. C., Pulliam, T. H., and Childs, M. L., "OVERFLOW Turbulence Modeling Resource Validation Results," Tech. Rep. NAS-2016-01, NASA Ames Research Center, 2016.

[41] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Journal of Optimization*, Vol. 12, No. 4, 2002, pp. 979–1006. https://doi.org/10.1137/S1052623499350013.