Analysis of Generative Adversarial Networks for Data Driven Inverse Airfoil Design

Priyam Gupta, Prince Tyagi and Raj Kumar Singh

July 1, 2021

Analysis of Generative Adversarial Networks for Data Driven Inverse Airfoil Design

Priyam Gupta, Prince Tyagi, and Raj Kumar Singh

Abstract. Data-Driven Methods have led to new approaches in the field of Aerodynamic Design and have recently found success in Inverse Design applications. The conventional Inverse Design methods are analytically complex and mathematically demanding to formulate. This research attempts to perform Inverse Airfoil Design using Generative Adversarial Networks(GANs) with the objective to generate airfoil shapes that produce desired Pressure Distribution at given flow conditions. The Convolutional Neural Network-based Generator extracts features from the pressure coefficient profiles and predicts the corresponding airfoil shape coordinates. These deep ConvNet structures eliminate the problems posed by shape parameterization in classical methods and extract patterns from the data with finer details. This work examines the performance of three advanced Generative Adversarial Network architectures to obtain a model which is stable, computationally efficient and has competitive prediction accuracy. The candidate GANs include Wasserstein GAN, Boundary Seeking GAN and Bidirectional GAN. The networks are trained on a database of airfoil shapes and pressure coefficient distribution. It is shown that Boundary Seeking Generative Adversarial Network produces highly accurate results and is computationally least expensive to train.

Keywords: Machine Learning · Generative Adversarial Networks · Inverse Design

1 Introduction

Inverse Aerodynamic Design is a category of Design Optimization techniques wherein the objective is to design aerodynamic shapes which produce desired pressure or velocity distribution at given flow conditions. There has been a significant advancement in computational power and computational fluid dynamics (CFD) algorithms, still, the heavy cost of high fidelity simulation demands more efficient design optimization techniques. The forward aerodynamic shape optimization approach involves evaluating the geometry for the desired parameter using CFD simulations followed by manipulating it until the desired parameter has been optimized[1]. This morphing of shape in every iteration makes the process computationally expensive. The Inverse design methods are an efficient alternative to the time consuming iterative process of analysing several airfoil shapes to obtain the one with desired characteristics.

The inverse aerodynamic design can be widely classified as surface flow design and flow-field design[2]. While the surface flow design needs the desired aerodynamic property specified on the surface, the Flow-field design incorporates global flow-field features for shape design constraints. There are a large number of inverse methods for surface flow design and most of them are considered analytically complex and mathematically demanding to formulate[3]. This demands a computationally efficient and tractable inverse design method. A good Inverse Aerodynamic Design Model should be able to generate smooth-contoured and manufacturable designs within the enforced constraints and should avoid exploiting the weakness of the problem formulation[4].

There has been a significant advancement in the application of machine learning and data-driven algorithms in recent years due to unprecedented volumes of data and advanced computing power. The field of fluid mechanics is also incorporating these advanced algorithms for flow field prediction, reduced-order and turbulence closure modelling[5]. Basu et al[6, 7] conducted a comprehensive study of machine learning algorithms for pressure reconstruction framework from PIV velocity data and observed a high degree of accuracy in the pressure distribution. Generative Adversarial Networks(GANs)[8] have achieved impressive results in the image to image translation and have shown to be effective in capturing the essential features in training data. GANs have also been used for a one to one mapping from a parameterized supercritical airfoil to its corresponding transonic flow field profile over the parametric space[9]. They have found effective use in supervised as well as unsupervised super-resolution reconstruction of Turbulent flows[10, 4]. There is a recent surge in interest to study deep learning models for aerodynamic design applications. Chen et al. [11] implemented Bezier GANs to learn low dimensional latent representation, encoding the variation in aerodynamic shapes. Du et al.[12] utilized a B-Spline based generative adversarial network parameterization method along with a multi neural network-based surrogate model for predicting aerodynamic coefficients which was demonstrated on an airfoil shape optimization problem. Yilmaz et al[4] implemented Conditional GANs and CNN based framework for surface flow inverse airfoil design conditioned on desired stall regions and downsampled desired drag polar data.

Motivated by the impressive performance of Generative Adversarial Networks for image to image translation and observing the need for a GAN architecture that is stable to train and requires less computation, the objective of this work is to investigate the performance of different GAN architectures for data-driven Inverse Airfoil Design. This study examines three GAN architectures and performs a side by side comparison. The candidate GAN architectures are Wasserstein GAN (WGAN)[13], Bidirectional GAN (BiGAN)[14] and Boundary seeking GAN (BGAN)[15]. These are selected on the basis of their promising performance and enhanced training stability over the vanilla GAN model in the past studies.

This paper has been arranged as follows: Section 2 delineates the fundamental concepts and mathematical formulation of Generative Adversarial Networks taken into consideration along with data generation methodology. Section 3 discusses the training of the candidate GAN models and compares their performance. Finally, concluding remarks are provided in Section 4.

2 Methodology

2.1 Generative Adversarial Network

Generative Adversarial Networks are a special type of neural network which consist of two elements: Discriminator and Generator. The Generator takes in a random noise from a distribution L_z and tries to generate samples G_s similar to the data samples D_s. These samples G_s are then fed to a Discriminator which predicts the probability of whether G_s is fake or not. The primary objective of the Generator is to generate samples that Discriminator is fooled to believe as identical to D_s while Discriminator‘s motive is to not get fooled by the Generator. During the training, the Generator learns to generate such real looking samples until the Discriminator can’t distinguish between real and fake samples. The architecture of a GAN is depicted in Fig. 1. The absolute expected error of the Discriminator when it is provided with ”actual” and ”generated” data is expressed by equation (1).

$$\begin{aligned} E(G,D) &= \frac{1}{2}E_{x\sim p_t}[1-D(x)] + \frac{1}{2}E_{z\sim p_z}[D(G(z))] \\ &= \frac{1}{2}(E_{x\sim p_t}[1-D(x)] + E_{x\sim p_g}[D(x)]) \end{aligned} \tag{1}$$

The training objective of the generator is to maximize the loss $E(G,D)$ while the Discriminator attempts to minimize it.

$$\max_G(\min_D E(G,D))$$

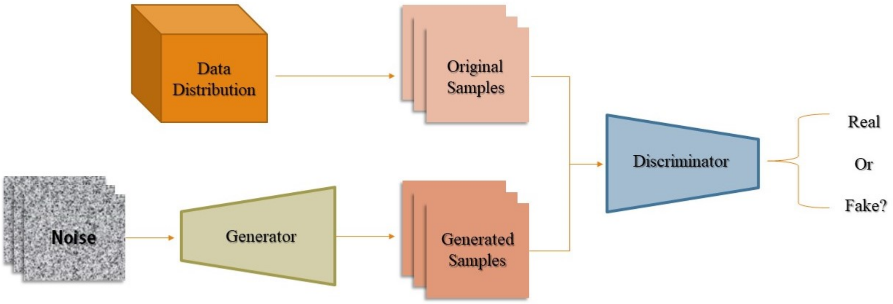


Fig. 1: Generic Architecture of a Generative Adversarial Network

1. ***Wasserstein Generative Adversarial Networks*** : The Wasserstein Generative Adversarial Network, or Wasserstein GAN, is an extended version of the generative adversarial network which enhances model‘s training stability and implements a loss function that correlates with the quality of generated samples. This substantially improves the approximation over the distribution of the data observed in the training set[13].
2. ***Boundary-Seeking Generative Adversarial Network*** : GANs face a major difficulty for training discrete data when outputs of the generator are completely differentiable with respect to the generator parameters, θ. The Boundary seeking GANs work well for discrete data and are equally efficient for multi-modal continuous distributions[15]. They utilize a boundary loss which ensures that the important weights have a strong connection to the decision boundary of the discriminator.
3. ***Bi-Directional Generative Adversarial Network*** : Bi-Directional Generative Adversarial Networks or Adversarial Feature Learning attempts to enhance the robustness of the generator by ensuring that the generated samples can be mapped back to input data [14]. Bi-GANs consist of an encoder in addition to the generator that is present in the standard GAN architecture which creates an inverse mapping between the generated samples and the input data. In order to ensure that the inverse mapping is done optimally, the BiGAN Discriminator distinguishes not only between the generated and actual samples but also between the inverse mapped samples and the corresponding actual data.

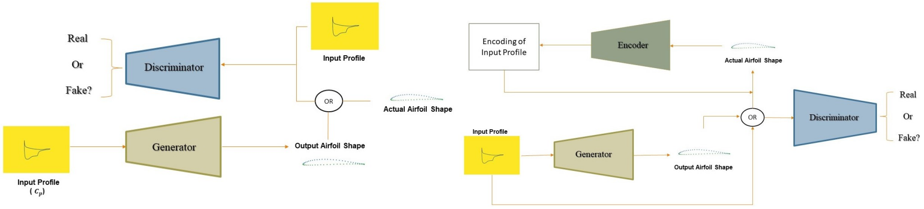
The schematic diagrams for all the algorithms are depicted in Fig. 2.

Table 1: Generator Architectures with their Hyperparameters. (The format for convolutional layer size is [Convolutional filter size, Number of convolutional filters]. In the fully connected layer, there are 100 neurons in each layer.)

Layer Type	BGAN	Bi-GAN	WGAN	Activation Function
Input	144 x 144 x 1	144 x 144 x 1	144 x 144 x 1	relu
1st convolution	4 x 4, 32	4 x 4, 32	4 x 4, 32	relu
2nd convolution	4 x 4, 64	4 x 4, 64	4 x 4, 64	relu
3rd convolution	4 x 4, 128	4 x 4, 128	4 x 4, 128	relu
4th convolution	4 x 4, 128	4 x 4,128	4 x 4, 128	relu
Flatten	-	-	-	-
Fully connected	100	100	100	tanh
Fully connected	100	100	100	tanh
Output layer	70	70	70	-

2.2 Data Generation

Data Generation is an essential component of the proposed framework. The input data consists of the Coefficient of Pressure (C_p) distribution over the surface



(a) WGAN and BGAN Architecture (b) BiGAN Architecturel

Fig. 2: Schematic of Generative Adversarial Network Architectures implemented for Inverse Airfoil Design

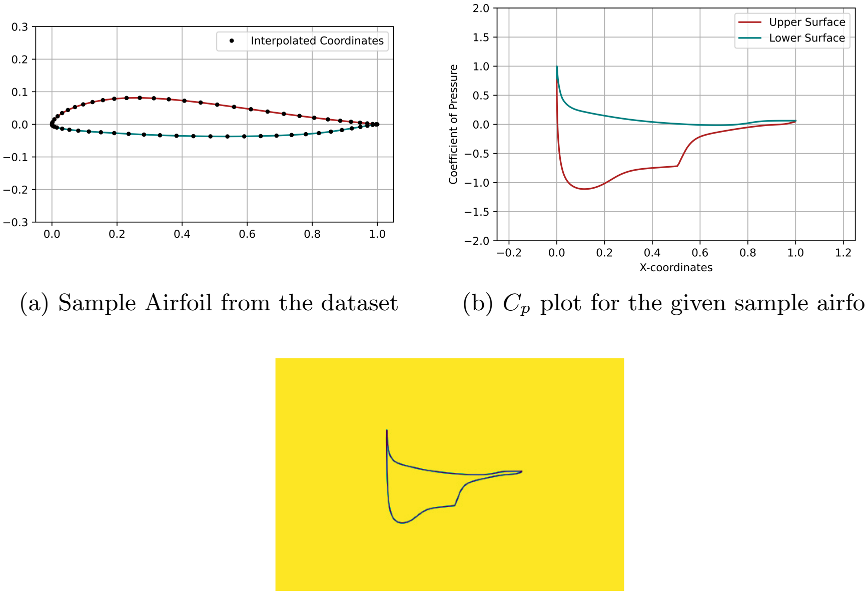
of the airfoil for an angle of attack of 3° and Reynolds Number of 100,000. The corresponding airfoil coordinates make up the output data. The C_p distribution over the airfoil is evaluated using the XFOIL simulation tool. XFOIL is considered over solving the Navier Stokes equation using CFD models because of the low computational cost involved which facilitated the generation of a large and diverse dataset enabling the demonstration of the maximum potential of the considered models. The dataset consists of 1400 airfoil profiles obtained from the UIUC airfoil database[16]. The C_p profile of each airfoil is plotted onto an image of size 144 X 144 X 1 which is then normalized to a range of [-1,1]. Figure 3 shows a sample of the upper and lower surface C_p plots used as the input data. The output data consists of the ordinates of the airfoil profile interpolated to a predefined set of x coordinates. This assisted in reducing the number of output parameters thereby enhancing the accuracy of the model.

3 Results

3.1 Training

The Dataset is split into a training and a test dataset with an 80% and 20% distribution. The network learns the weight using a *content loss* and an *adversarial loss* function. The content loss follows a simple approach wherein the generated and actual points are compared using mean squared error while the adversarial loss is subjected to the probability of the generator fooling the discriminator.

The input images are of size $144 \times 144 \times 1$ and are normalized down to a range of [-1,1]. The BiGAN and BGAN models are trained using the Adam optimizer with a learning rate of 2e−5 and a batch size of m = 32. Dropout layers were included after every convolution layer to avoid overfitting with a dropout probability of 0.4. The Wasserstein GAN model is trained using the RMSProp optimizer with a learning rate of 2e−5. The optimization algorithm for each model is chosen on the basis of the recommendations in their original works[14, 15, 13]. The hyperparameters for each model are tuned using the grid search method. The architecture of each GAN model is delineated in 1 and 2.



(a) Sample Airfoil from the dataset

(b) C_p plot for the given sample airfoil

(c) Input Sample of C_p profile

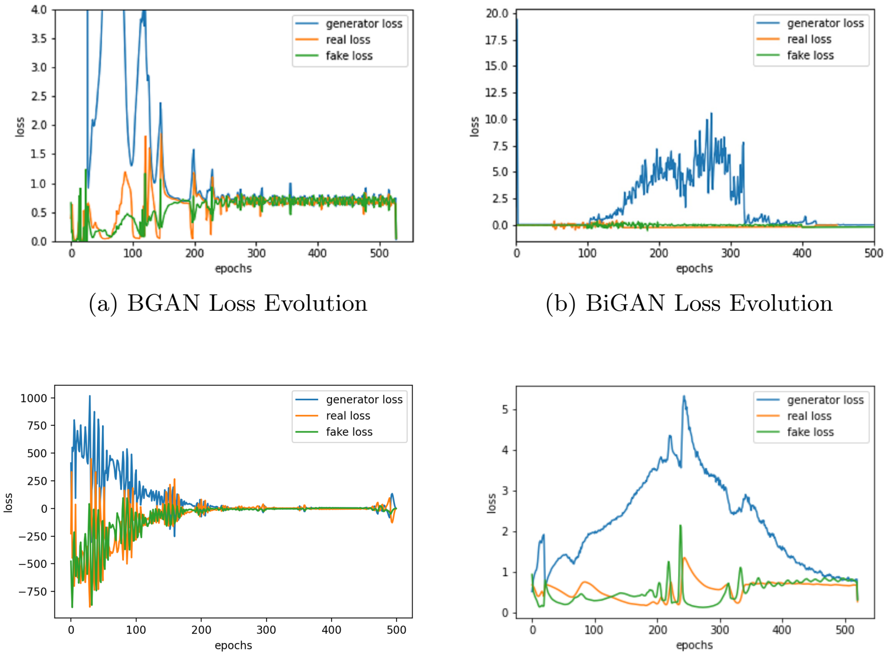
Fig. 3: Training Data Generated Using UIUC Airfoil Database and XFOIL

The models were written in TensorFlow and were trained on Tesla K80 GPUs. Detailed architecture of generator is depicted in Table 1 and of discriminator in Table 2.

3.2 Discussion

The training process of the GANs is analogous to a minimax game wherein two players compete with each other i.e. the Generator tries to generate real-looking samples that can't be differentiated further by the discriminator. Theoretically, GANs are said to be converged when the discriminator can no longer distinguish between fake and original data. The loss evolution of the generator and discriminator for each of the GAN model is depicted in Fig. 4. GAN loss function generally depicts vibrating and non-converging behaviour while training therefore a good practice to check that the model has converged is by visually observing whether the output produced is close to the training data distribution.

Initially, the comparison amongst the generated samples of the three proposed GANs and the vanilla GAN is done on the basis of the quality of the airfoils generated. This can be evaluated by observing the subset of the gen-



(a) BGAN Loss Evolution

(b) BiGAN Loss Evolution

(c) WGAN Loss Evolution

(d) Vanilla GAN Loss Evolution

Fig. 4: Loss Functions During The Training of the GAN Models

erated samples by each model shown in Fig. 5. Since the main objective of the proposed GAN frameworks is to generate airfoil geometries that produce desired aerodynamic properties at given flow conditions, a distance metric (d_{GAN}) averaged over all the generated samples is used to evaluate the proximity of the generated airfoils to the actual samples which is given by equation 2.

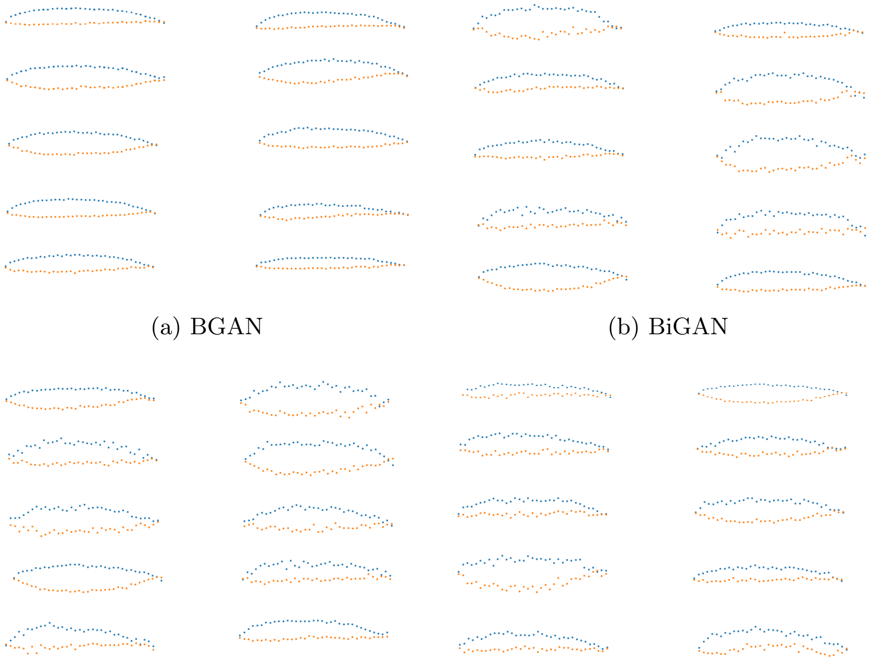
$$d_{GAN} = \sqrt{\frac{\sum_{i=1}^{N}(y_{p_i} - y_{a_i})^2}{N}} \tag{2}$$

where y_{p_i} is the vector containing 70 y-coordinates of each predicted sample, y_{a_i} is the vector containing the actual points and N is the total number of samples. The d_{GAN} metric for each model is shown in Fig. 7, with BGAN performing the best and vanilla GAN performing the worst.

From Fig. 5, it can be observed that the BGAN produces smooth-contoured aerodynamic shapes while WGAN fails to do so. Though it is able to capture the features of the airfoil, it suffers from overtraining. BiGAN is able to capture

Table 2: Discriminator Architectures with their hyperparameters. (In the 1st layer , 512 indicates 512 neurons in each layer.). All Layers are fully connected.

Layer Type	BGAN	Bi-GAN	WGAN	Activation Function
Input	70 x 1	70 x 1, 144 x 144 x 1	70 x 1	Leaky Relu
1st Layer	512	Flatten+512	512	Leaky Relu
2nd Layer	256	256	256	Leaky Relu
3rd Layer	128	128	128	Leaky Relu
4th Layer	64	64	64	Leaky Relu
Output	1	1	1	sigmoid(not in WGAN)



(a) BGAN

(b) BiGAN

(c) WGAN

(d) Vanilla GAN

Fig. 5: Generated Airfoil Samples of GAN Models

the structure of the airfoil but it has quite a high distance metric (d_{GAN}) as compared to BGAN. Vanilla GANs also showed the potential to capture the features of the airfoil, but they had the worst d_{GAN} which means that they were not able to generalize over all the airfoil shapes. Despite the repeated efforts to fine-tune the models, WGAN and BiGAN suffered from overtraining and mode collapse i.e they produced a restricted variety of output samples while BGAN consistently produced samples close to the actual data distribution. It should be noted that all the GAN model architectures have been modified to

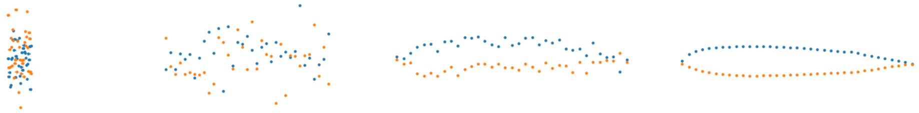


Fig. 6: Shape Evolution Over Training Epochs For BGAN

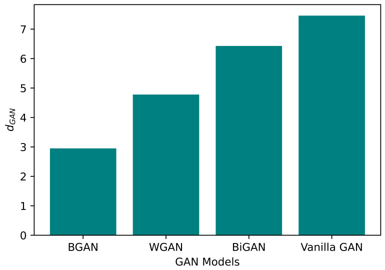


Fig. 7: Distance Metric over test samples for the GAN Models

suit a low computational resource requirement in order to optimize the training time. Considering that the samples generated by BGAN had the lowest distance metric, its high training stability, its ability to learn the aerodynamic features of the airfoils and generalize over the airfoil data distribution, BGAN shows the best performance amongst the studied GAN architectures. Figure 6 shows how airfoils are generated over the epochs during the training of BGAN and its ability to extract aerodynamic features from the data distribution.

4 Conclusion

In this work, we analyzed the performance of Generative Adversarial Networks for aerodynamic design applications based on its ability of the image to image translation and concluded upon a framework with reasonably tamed training stability and appreciable quality of generated airfoil design. We pointed out the vulnerability of GANs and worked towards a model that could minimize them.

Three modified GAN structures were tested for inverse airfoil design and were compared with the vanilla GAN based on quality of airfoils generated, proximity to the corresponding airfoil for given flow conditions and the training stability. When analyzing the application of the models, all the models were able to learn aerodynamic features of the airfoil but only BGAN produced appreciably smooth contoured surfaces. It generated aerodynamic curves with the least distance metric which was confirmed by the comparison between the actual and generated airfoils, making it the most suitable candidate for inverse airfoil design.

Future work may include testing these models on variable flow conditions and developing more accurate datasets using Reynolds Averaged Navier Stokes based computations. The results of this work are quite motivating to test the larger-scale performance of these models with higher computational power. Overall GANs have proved to have substantial potential in the application for aerodynamic design and can be employed in different applications across this field.

References

1. Kumar, S., Gupta, P., Singh, R.K.: A natural evolution based numerical optimisation framework to develop and enhance airfoil-slat arrangement. In: ASMEInternational Mechanical Engineering Congress and Exposition. vol. 59445, p.V007T08A056. American Society of Mechanical Engineers (2019)
2. Dulikravich, G., Baker, D.: Aerodynamic shape inverse design using a fourier series-method. In: 37th Aerospace Sciences Meeting and Exhibit. p. 185 (1999)
3. Dulikravich, G.S.: Aerodynamic shape design and optimization-status and trends.Journal of aircraft29(6), 1020–1026 (1992)
4. Yilmaz, E., German, B.: Conditional generative adversarial network framework forairfoil inverse design. In: AIAA AVIATION 2020 FORUM. p. 3185 (2020)
5. Brunton, S.L., Noack, B.R., Koumoutsakos, P.: Machine learning for fluid mechanics. Annual Review of Fluid Mechanics52, 477–508 (2020)
6. Basu, M., Kumar, S., Gupta, P., Singh, R.K.: A Quantitative Analysis of Machine Learning based Regressors for Pressure Reconstruction in PIV Applications, Fluids Engineering DivisionSummer Meeting. In: ASME Fluids Engineering Division Summer Meeting (2020)
7. Tyagi, P., Gupta, P., Singh, R.K.: Extraction of flow features for predicting pressuredistribution using convolutional neural networks. Tech. rep. (2021)
8. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair,S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Ghahramani, Z.,Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.) Advances inNeural Information Processing Systems 27, pp. 2672–2680. Curran Associates, Inc.(2014), http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf
9. Wu, H., Liu, X., An, W., Chen, S., Lyu, H.: A deep learning approach for efficientlyand accurately evaluating the flow field of supercritical airfoils. Computers Fluids198, 104393 (2020)
10. Kim, H., Kim, J., Won, S., Lee, C.: Unsupervised deep learning for super-resolutionreconstruction of turbulence. arXiv preprint arXiv:2007.15324 (2020)
11. Chen, W., Chiu, K., Fuge, M.: Aerodynamic design optimization andshapeexplorationusinggenerativeadversarialnetworks(012019). https://doi.org/10.2514/6.2019-2351
12. Du, X., He, P., Martins, J.: A b-spline-based generative adversarial net-work model for fast interactive airfoil aerodynamic optimization (01 2020).https://doi.org/10.2514/6.2020-2128
13. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein gan. arXiv preprint-arXiv:1701.07875 (2017)
14. Donahue, J., Kr ahenb uhl, P., Darrell, T.: Adversarial feature learning. arXivpreprint arXiv:1605.09782 (2016)
15. Hjelm, R.D., Jacob, A.P., Che, T., Trischler, A., Cho, K., Bengio, Y.: Boundary-seeking generative adversarial networks. arXiv preprint arXiv:1702.08431 (2017)
16. Selig, M.: UIUC airfoil data site. Department of Aeronautical and AstronauticalEngineering University of Illinois at Urbana-Champaign (1996)