Tenth International Conference on Computational Fluid Dynamics (ICCFD10), Barcelona, Spain, July 9-13, 2018

ICCFD10-223

Flow Structure Oriented Optimization Aided by Deep Neural Network

Kaiwen DENG*, Haixin CHEN* and Yufei ZHANG*
Corresponding author: chenhaixin@tsinghua.edu.cn

* School of Aerospace, Tsinghua University, China

Abstract: This article proposes a new way of aerodynamic design optimization by transferring current ubiquitous performance oriented optimization to flow field oriented optimization through incorporating individual improvement strategy with the help of state-of-the-art machine learning and deep learning techniques. The technological framework is presented and the optimization performance is shown and compared to traditional optimizer.

Keywords: Aerodynamic Optimization, Computational Fluid Dynamics, Machine Learning, Deep Learning.

1 Introduction

Recent development of computational fluid dynamics (CFD) and intelligent optimization algorithms has enabled researchers and engineers to efficiently sketch feasible aerodynamic configurations with satisfactory performance in engineering occasions. To maintain high-fidelity, costly numerical simulation scheme of higher accuracy is usually preferred and raises challenges upon optimization efficiency under limited computation resources. In recent decades, Evolutionary algorithms (EA)[1]–[8], surrogate models[9]–[15], adjoint method in conjunction with gradient-based search method[16], [17], [26], [18]–[25] and hybridization of part of above approaches are continuously proposed to tackle expensive aerodynamic optimization problem and brings considerable improvement in both optimization efficiency and performance. However, current optimizers are still faced with major defects while being utilized in engineering occasions, those defects can be sorted into two aspects:

- Unable to obtain solutions that fully suit designer's intension or satisfy realistic demands. Optimizer often relies on the designer to explicitly assign optimization objectives. Due to the inefficiency of Pareto dominance based multi-objective optimization, number of objectives is usually limited. Such focus on limited designated objectives cannot guarantee various off-design performances. Current optimizer also fails to well handle implicit objectives and constraints such as structural and geometrical constraints, dynamic performance characteristics, or inenarrable flow field expectation which are crucial but are often compromised for efficiency matter or ignored in early design stages for their fuzziness.
- Poor engineering applicability. Evolutionary approaches such as genetic algorithm often need massive objective evaluations to achieve convergence while gradient-based methods are prone to get stuck in local-optimum. Careful trade-off still needs to be made between global search and local search.

The key to above problems can be concluded as insufficiency of information utilization. Massive information exists behind the physics of the optimization problem but receives little attention. Current aerodynamic optimizer only focuses on the designated objectives and this is the so-called performance oriented optimization. However human experts focuses on the characteristics of flow field structures to improve current solutions' performances. Such preference significantly differs from ubiquitous

performance-oriented optimization and can be called as flow structure oriented optimization (FSOO). Fully automatic direct optimization approach was initially proposed to extricate the optimization loop of human interference, which is seriously constrained by human's limited memory, inexpertness of large scale trading-off and incapability of complex numerical analysis. While the most significant merits of human, the ability to inject physical comprehension to solutions that allows quick and precise analysis, is not well inherited by current optimizers, and thus leads to optimizers' strong reliance to experts, which seriously deviates from its purpose[27]. To effectively alleviate above problems, optimizers should look beyond variables and objectives and dig further into flow fields to gain more information feedback. Transient flow field structure and significant flow patterns inside usually implicate massive latent information that's necessary for the optimization.

With adequate identification of essential flow structure and right means to manipulate it, efficient and effective flow structure oriented optimization can be achieved. In short, human experts have merely done four things inside the optimization loop: (1). Identify and analyze the essential flow structures in the flow field; (2). Discover the mapping relationship between aerodynamic performance and flow structures; (3). Discover the mapping relationship between geometry (design variables) and flow structures; (4). Exert changes to the current solutions according to (2) and (3) to guide their improvement.

In this article technical framework is proposed to achieve FSOO through substituting above steps of human interference by machine learning and deep learning models to fully utilize and analyze the flow field data generated during optimization to achieve rational automated individual improvement. With improved utilization of information, further optimization speedup and performance boost is expected to be achieved.

2 Technical Framework

2.1 Definition of Aerodynamic Optimization

Assuming the dimensionality of design variables $\boldsymbol{x}$ and objectives $\boldsymbol{y}$ are N_x and N_y. Aerodynamic optimization can be briefed as finding the optimal design variables $\boldsymbol{x}^*$ such that equation (1) holds:

$$\boldsymbol{y}^* = \boldsymbol{f}(\boldsymbol{x}^*) = \underset{\boldsymbol{x}\in\mathbb{X}\cap\mathbb{R}}{\text{opt}}\,(\boldsymbol{f}(\boldsymbol{x})) \tag{1}$$

Where $\mathbb{X}$ is the pre-defined search space, usually hyperspace in R^{N_x}. $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$ serves as the implicit objective mapping, which usually refers to the non-analytical numerical evaluation process in aerodynamic optimization. $\mathbb{R}$ depicts the feasible region of $\boldsymbol{x}$ where N_g non-equality constraints $\boldsymbol{g}(\boldsymbol{x})$ and N_h equality constraints $\boldsymbol{h}(\boldsymbol{x})$ are satisfied according to equation (2) and (3).

$$g_i(\boldsymbol{x}) \le 0\big(i = 1,2\ldots, N_g, \forall x \in \mathbb{X}\big) \tag{2}$$

$$h_i(\boldsymbol{x}) = 0(i = 1,2\ldots, N_h, \forall x \in \mathbb{X}) \tag{3}$$

$\text{opt}(\boldsymbol{f})$ denotes the feasible optimal values of given expression $\boldsymbol{f}$. For single objective optimization $\text{opt}(\cdot)$ is equivalent to $\min(\cdot)$ or $\max(\cdot)$ while for multi-objective optimization, $\text{opt}(\boldsymbol{f})$ denotes the non-dominant solutions in the Pareto front of expression $\boldsymbol{f}$.

2.2 Differential Evolution Based Hybrid Optimizer

Differential evolution (DE) is a real-coded evolutionary algorithm famous for its robustness and population diversity preservation ability[28]. It's adopted as the basis of overall optimizer framework. In DE, the optimization process is carried out iteratively until pre-defined convergence criteria are satisfied. Here combine a solution's design variables and objectives as an individual $P = (\boldsymbol{x}, \boldsymbol{y})$, and gather a group of individuals as a population $\boldsymbol{P}^k = \{P_i^k\}_{i=1,2,\ldots,N_p}$ while subscript denotes current iteration or generation. For every iteration or generation, the optimization is carried out as:

(1).Mutation, for every individual P_i^k, select n individuals' design variables $\boldsymbol{x}_{r1}^k, \boldsymbol{x}_{r2}^k, \ldots, \boldsymbol{x}_{rn}^k$ from current population and generate the corresponding mutated individual of P_i^k as $V_i^k = (\boldsymbol{v}_i^k, \boldsymbol{f}(\boldsymbol{v}_i^k))$. where $\boldsymbol{f}(\boldsymbol{v}_i^k)$ is unknown and $\boldsymbol{v}_i^k$ is defined in different ways as formulated in equation (4) ~ (7) where subscript 'best' indicates the best individual in current population. In this article Rand/2 is adopted as the mutation scheme.

$$\text{Rand/1:}\boldsymbol{v}_i^k = \boldsymbol{x}_i^k + F \times (\boldsymbol{x}_{r2}^k - \boldsymbol{x}_{r3}^k) \tag{4}$$

$$\text{Rand}/2\boldsymbol{v}_i^k = \boldsymbol{x}_i^k + F \times (\boldsymbol{x}_{r2}^k - \boldsymbol{x}_{r3}^k + \boldsymbol{x}_{r4}^k - \boldsymbol{x}_{r5}^k) \quad (5)$$

$$\text{Best}/1\boldsymbol{v}_i^k = \boldsymbol{x}_{best}^k + F \times (\boldsymbol{x}_{r1}^k - \boldsymbol{x}_{r2}^k) \quad (6)$$

$$\text{Best}/2\boldsymbol{v}_i^k = \boldsymbol{x}_{best}^k + F \times (\boldsymbol{x}_{r1}^k - \boldsymbol{x}_{r2}^k + \boldsymbol{x}_{r3}^k - \boldsymbol{x}_{r4}^k) \quad (7)$$

(2).Crossover, combine the mutated individual V_i^k and corresponding parent individual P_i^k to generate the corresponding trial individual U_i^k=($\boldsymbol{u}_i^k, \boldsymbol{f}(\boldsymbol{u}_i^k)$). $\boldsymbol{u}_i^k$ is defined in equation (8) where r and L are random integers ranging from 1 to N_x that represent the starting index and length of the crossover operation. $<\cdot>_{N_x}$ denotes modular operation towards integer N_x.

$$u_{ij}^k = \begin{cases} v_{ij}^k, j = <r>_{N_x}, <r+1>_{N_x}, \dots, <r+L-1>_{N_x} \\ x_{ij}^k, j = <r+L>_{N_x}, <r+L+1>_{N_x}, \dots, <r>_{N_x} \end{cases} \quad (8)$$

(3).Accept external individuals and inject into the current population to take the place of worst ones. This is the interface set up for the need of individual improvement which serves as external interference as discussed above.

(4).Parallel computation of objective functions $\boldsymbol{f}(\boldsymbol{u}_i^k)$.

(5).Selection, execute selection operation between U_i^k and P_i^k solely according to Pareto-dominance based greedy principle to preserve the better individual into the offspring population as P_i^{k+1}.

Above procedures are concluded in Figure 1.

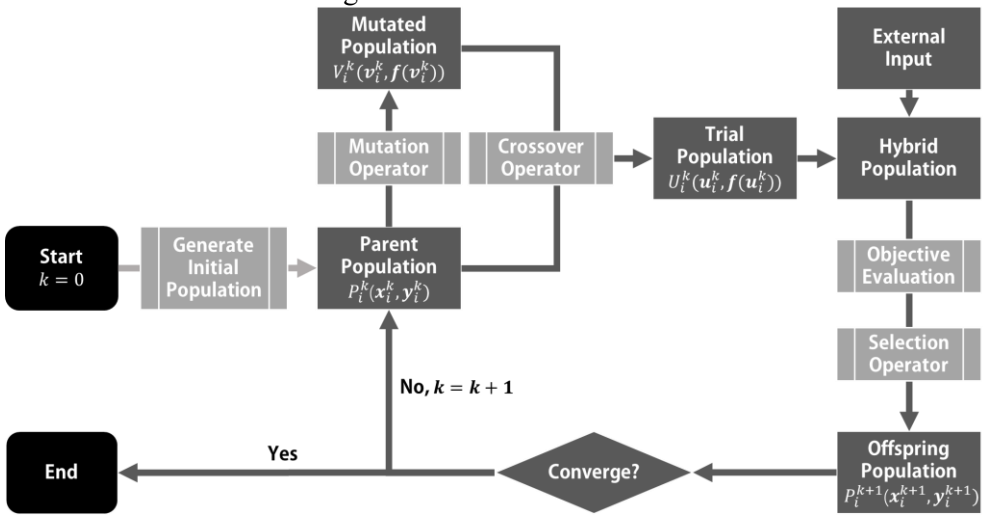


Figure 1: Optimization procedures of differential evolution

2.3 Deep Neural Network Aided Flow Field Feature Extraction

To achieve human-like individual improvement, it's significant to capture the essential flow patterns that influence the objectives we concern during optimization. The flow field data should be used as major data sources to be subtly analyzed and then passed to the optimizer. Such analysis usually appears as the form of dimensionality reduction or feature extraction of the flow field data to capture its essential flow patterns or 'features'. In previous studies, principal component analysis (PCA) was utilized as feature extractor to construct reduced order models (ROM)[29]–[35], carry out sensitivity analysis and variable filtering[36], [37]. Observation showed while PCA based dimensionality reduction was effective with prominent variance preservation and slight information loss, several severe shortcomings still existed while dealing with flow field data:

- Base vectors used to map the flow field into low-dimensional features are linear combinations of training samples. Such linearity symbols inability to capture complicated non-linear flow patterns even when kernel function is used.
- PCA treats different components of input data (features) equally, which erases both spatial and cross-physical-quantity relationship inside the flow field and also disables PCA to handle spatial invariance such as translation, rotation and scaling.

During last decades, machine learning and deep learning technology have experienced prominent development. Deep neural networks (DNNs) has been widely utilized in computer vision, natural

language processing and big data analysis. In the field of computational fluid dynamics, DNNs have been used for turbulence modeling[38] and reduced order model construction[39]–[41]. With the development of network architecture, the maximum trainable layer of neural network has been greatly extended, endowing the networks with stronger ability to efficiently capture source data's complicated high-level features. Convolutional neural network (CNN), which specializes in handling image-related data or high order tensor, is naturally amiable to structured flow field data generated by CFD solver. CNN is capable of handling spatial invariance and is extremely computationally efficient compared to traditional full-connected networks. AE is one type of feed-forward neural network architecture that accepts high dimensional source data as input and pass them to the decoder to gradually reduces data's dimensionality to a bottleneck, which is referred to as extracted features. Then the features are passed to the decoder that ascends data's dimensionality to the input size. Convolutional auto-encoder (CAE), which combines CNN's ability to handle image-like data and auto-encoder (AE)'s dimensionality reduction architecture, is an ideal tool for the flow field feature extraction.

The purpose of CAE is to find the essential elements that can best describe the input data distribution. To achieve that, the overall training target of CAE is to recover the input data as much as possible, and the training loss of CAE can be formulated in equation (9):

$$L(\boldsymbol{T}, \tilde{\boldsymbol{T}}, \boldsymbol{w}) = \sum_i \sum_j \sum_k \left(T_{ijk} - \tilde{T}_{ijk}\right)^2 + R(\boldsymbol{w}) \tag{9}$$

$\boldsymbol{T}$ denotes input tensor, $\tilde{\boldsymbol{T}}$ denotes the output tensor, $\boldsymbol{w}$ symbols the network parameters. The first term on the right side denotes the recovery or reconstruction loss, and the second is the regularization term used to avoid over-fitting which is taken as L2-norm in most cases.

In our application the input $\boldsymbol{T}$ is the flow field data in structured mesh, and the encoded feature vector $\boldsymbol{c}$ is the flow field feature representation. The network structure of CAE can be depicted as Figure 2.

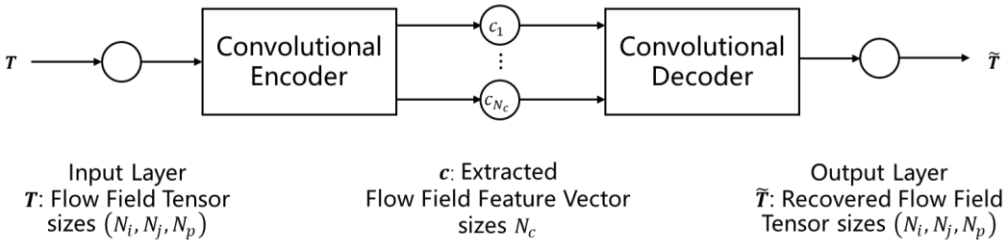


Figure 2: Convolutional autoencoder

Convolutional encoder is a stack of convolution layers and pooling layers, used for feature extraction while convolutional decoder is composed of convolution layers, deconvolution layers and pooling layers which specializes in recovering the input data using encoded features. The arithmetic of convolution, pooling and deconvolution layer are introduced in detail in tutorial[42].

2.4 MLP Aided Mapping Analysis

Multi-layer perceptron (MLP) is the most typical model of artificial neural network. Here MLPs are adopted for mapping relationship analysis between design variables $\boldsymbol{x}$, objectives $\boldsymbol{y}$ and flow field features $\boldsymbol{c}$. A typical MLP architecture with one hidden layer is presented in Figure 3.

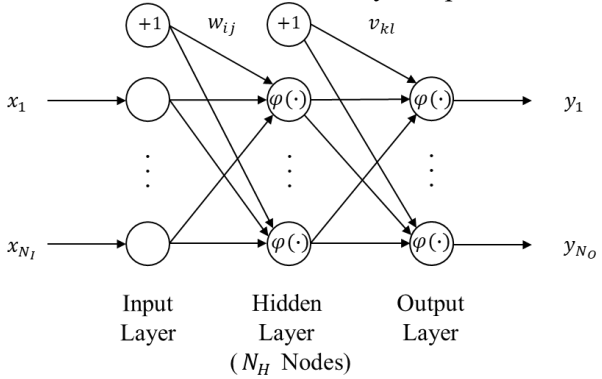


Figure 3: MLP Structure With One Hidden Layer

MLP is also called fully-connected network which symbols the nodes in adjacent layers are densely connected to each other. Output of each layer can be given by equation (10) and (11):

$$\text{Hidden Layer:}\quad p_i = \varphi_i\left(\sum_{j=1}^{N_x} w_{ji}\, x_j + w_{j0}\right) \tag{10}$$

$$\text{Output Layer:}\quad y_i = \varphi_i\left(\sum_{j=1}^{N_H} v_{ji}\, p_j + v_{j0}\right) \tag{11}$$

Training loss of MLP on one sample is given by equation (12):

$$L(\boldsymbol{y}, \boldsymbol{d}, \boldsymbol{w}) = \frac{1}{N_S}\sum_{i=1}^{N_S} \| \boldsymbol{y}_i - \boldsymbol{d}_i \|^2 + R(\boldsymbol{w}) \tag{12}$$

Similarly, $\boldsymbol{y}$ denotes the network output and $\boldsymbol{d}$ is the ground truth output. The first term on the right side denotes the regression loss contributed by this sample and the second term is regularization term.

2.5 Gradient Based Individual Improvement Strategy

Combining the models depicted in 2.3 and 2.4, the overall model architecture can be summarized in Figure 4. It's noted that the mapping relationship from flow feature to design variable is established not the other way around. The reason is that the information contained in $\boldsymbol{c}$ is sufficient to deduce corresponding $\boldsymbol{x}$ while the latter cannot fully infer the former. Training session for each model adopts gradient back-propagation incorporated with Adam optimizer.

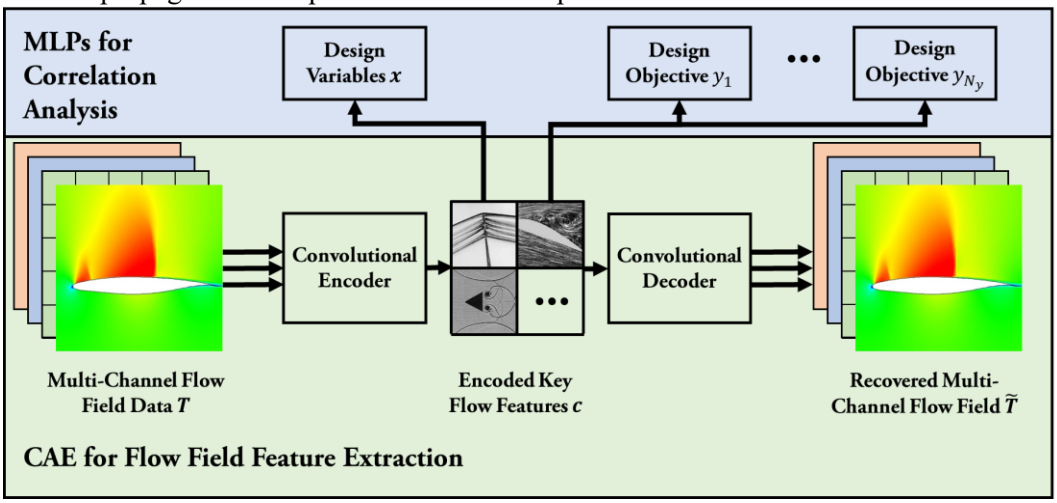


Figure 4: Overall model architecture

After gathering well-trained models, the individual improvement procedures can be defined with respect to the four major steps processed by human experts depicted above. Assuming a set of individuals $P_i = \{\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{T}_i\}$ are selected from current population as origin of improvement where $\boldsymbol{T}_i$ represents the structured flow field data. For every P_i, step (1) ~ (4) are carried out.

(1).Pass $\boldsymbol{T}_i$ through trained CAE to obtain its corresponding flow field feature $\boldsymbol{c}_i$.

(2).Use the MLP modeling mapping relationship $\tilde{\boldsymbol{y}} = \boldsymbol{m}_1(\boldsymbol{c})$ from $\boldsymbol{c}$ to $\boldsymbol{y}$ to calculate the local gradient matrix from $\boldsymbol{y}_i$ to $\boldsymbol{c}_i$ as $\boldsymbol{O}_i$.

$$\boldsymbol{O}_i = \left[\boldsymbol{o}_1^i, \boldsymbol{o}_2^i, \ldots, \boldsymbol{o}_{N_y}^i\right]^T = \begin{bmatrix} \frac{\partial \tilde{y}_1}{\partial c_1}\big|_{c_1=c_{i1}} & \cdots & \frac{\partial \tilde{y}_1}{\partial c_{N_c}}\big|_{c_{N_c}=c_{iN_c}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \tilde{y}_{N_y}}{\partial c_1}\big|_{c_1=c_{i1}} & \cdots & \frac{\partial \tilde{y}_{N_y}}{\partial c_{N_c}}\big|_{c_{N_c}=c_{iN_c}} \end{bmatrix}$$

(3).Use the MLP modeling mapping relationship $\tilde{\boldsymbol{x}} = \boldsymbol{m}_2(\boldsymbol{c})$ from $\boldsymbol{c}$ to $\boldsymbol{x}$ to calculate the local gradient matrix from $\boldsymbol{y}_i$ to $\boldsymbol{x}_i$ as $\boldsymbol{V}_i$.

$$\boldsymbol{V}_i = \left[\boldsymbol{v}_1^i, \boldsymbol{v}_2^i, \ldots, \boldsymbol{v}_{N_x}^i\right]^T = \begin{bmatrix} \frac{\partial \tilde{x}_1}{\partial c_1}\big|_{c_1=c_{i1}} & \cdots & \frac{\partial \tilde{x}_1}{\partial c_{N_c}}\big|_{c_{N_c}=c_{iN_c}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \tilde{x}_{N_x}}{\partial c_1}\big|_{c_1=c_{i1}} & \cdots & \frac{\partial \tilde{x}_{N_x}}{\partial c_{N_c}}\big|_{c_{N_c}=c_{iN_c}} \end{bmatrix}$$

(4).Use the improvement direction Δ_j of design variables for every objective y_j of this individual with respect to equation (5). $m_j = -1$ if y_i desires to be minimized and $m_i = 1$ for maximization.

$$\Delta_{ij} = m_j \boldsymbol{V}_i^T \boldsymbol{o}_j^i / |\boldsymbol{V}_i^T \boldsymbol{o}_j^i| \tag{13}$$

Generate improved individuals $\boldsymbol{C}_i = \{C_{ij}\}_{j=1,2,\dots,N_y}$ of P_i according to certain optimization strategy. C_{ij}'s design variables $\boldsymbol{x}_{ij}$ is given by equation (14) where α is previously defined learning rate.

$$\boldsymbol{x}_{ij} = \boldsymbol{x}_i + \alpha \Delta_{ij} \tag{14}$$

2.6 Overall Flow Chart

The total technical framework above is called flow structure oriented optimization (FSOO). Overall flow chart is described in the figure below:

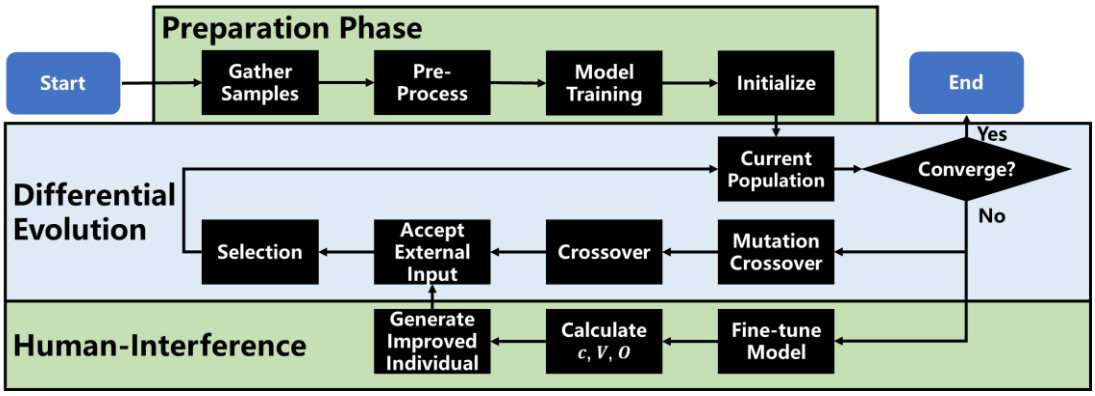


Figure 5: Overall Optimization Flow Chart

3 Test Case Validation

3.1 Introduction

Considering a multi-point drag reduction process for 2-D airfoils. Geometry is generated using class shape transformation (CST)[43] approach with 14 control points, among which 7 are used for upper surface and 7 for lower surface. The design variables, objectives and constraints are briefed in Table 1.

Table 1: Optimization Problem Settings

Design Variables				Objectives	Constraints
Upper Surface	Range	Lower Surface	Range		
U_1	[0.096,0.14]	L_1	[-0.20,-0.10]	C_{d1} (Minimize)	$C_{d1} < 0.010$
U_2	[0.04 , 0.14]	L_2	[-0.10, 0.00]		$C_{d2} < 0.014$
U_3	[0.12 , 0.20]	L_3	[-0.24,-0.14]		$C_{L1} > 0.849$
U_4	[0.02 , 0.08]	L_4	[-0.16,-0.09]		$C_{L1} < 0.851$
U_5	[0.20 , 0.26]	L_5	[-0.21,-0.11]	C_{d2} (Minimize)	$C_{L2} > 0.849$
U_6	[0.13 , 0.19]	L_6	[-0.11, 0.00]		$C_{L2} < 0.851$
U_7	[0.192,0.26]	L_7	[0.16, 0.22]		$R > 0.01$

C_{d1} and C_{d2} are the total drag coefficients at Mach 0.72 and 0.75 with fixed lift coefficient be set to 0.85. An in-house developed program NSAWET[44], [45] is used in this paper as CFD evaluation tool. Other CFD settings are listed in Table 2.

Table 2: CFD Evaluation Settings

Grid Size	Reynolds number	Discretization scheme	Reconstruction scheme	Turbulence model
257×97	6,100,000	Roe	3rd order MUSCL	$k-\omega$ SST

Table 3 gives the parameters of the optimizer. In this problem we compare FSOO to basic differential evolution optimizer. The parameters listed applies to both optimizers.

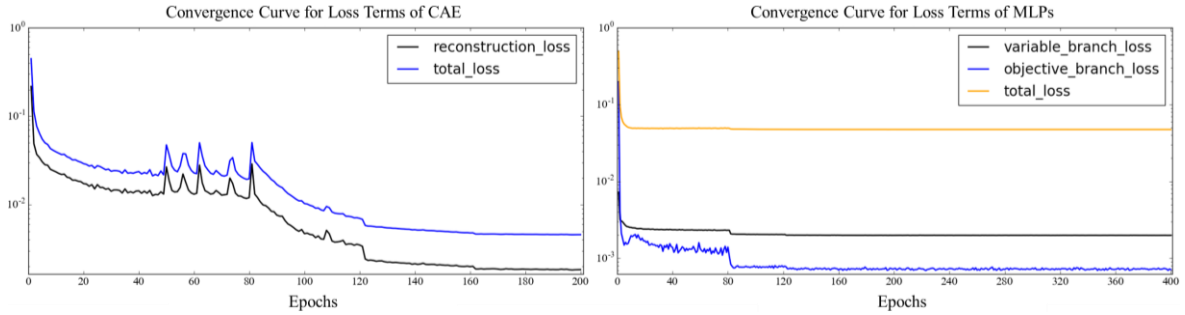
Table 3: Optimizer Settings

Total Generation N_g	Population Size N_p	N_I	F	CR
200	28	2	0.5	0.2~0.7
Physical Quantities Used for Analysis	Flow Feature Dimension N_c	Learning Rate α	Regularization Term	Regularization Weight
(x, y, u, v, p, T)	40	0.001	L2	0.01

The neural network architecture used by convolutional encoder and decoder is ResNet-50[46] and the total number of parameters in CAE is nearly 6,000,000. The MLPs used for relationship analysis has 3 hidden layers and nodes of hidden layer is set to be 400.

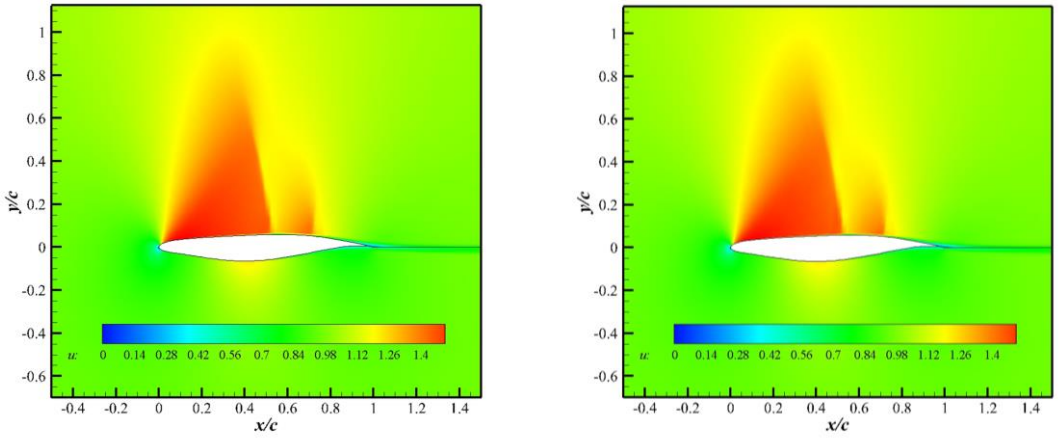
3.2 Model Training

Around 12,000 previously collected solutions are used to train the models. For CAE, the concerned flow field quantities are coordinates of grid vertexes x and y, velocity u and v, static pressure p and temperature T, as shown in Table 3. Figure 6 gives the convergence curve for CAE and MLPs used in this optimization. Figure 7 gives performance illustration for the well-trained CAE on a random test flow field. The final loss of both CAE and MLPs is at 10^{-3} level.



(a).Loss Curve for CAE (b).Loss Curve for MLPs

Figure 6: Convergence Curves



(a).Source Mach Contour (b).Recovered Mach Contour

Figure 7: CAE Feature Extraction Performance

3.3 Optimization Process and Results

Figure 8 shows the convergence curves of FSOO and basic DE. The convergence criterion is defined as the generation distance which refers to the average Euclidean distance from current population to the combined Pareto front obtained by optimizers. Figure 9 shows the Pareto front obtained by FSOO and basic DE. Figure 10 gives typical comparisons of surficial pressure distribution of 3 pairs (marked as A, B and C) of similar optimal solutions obtained by competing optimizers.

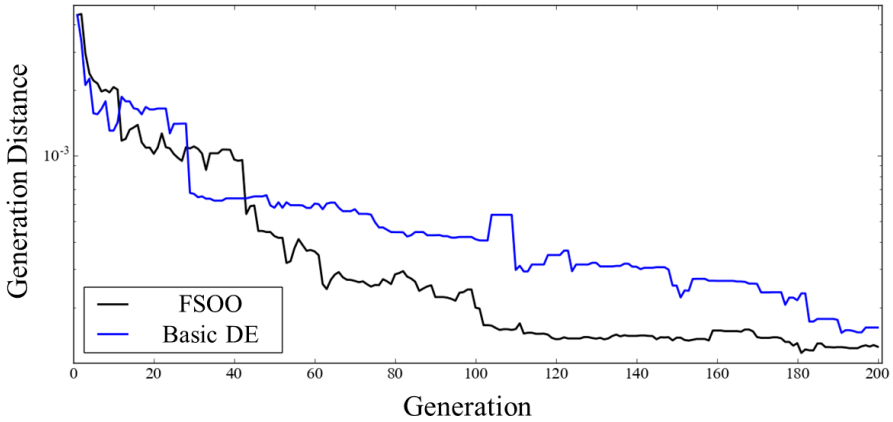


Figure 8: Convergence Curve for Competing Optimizers

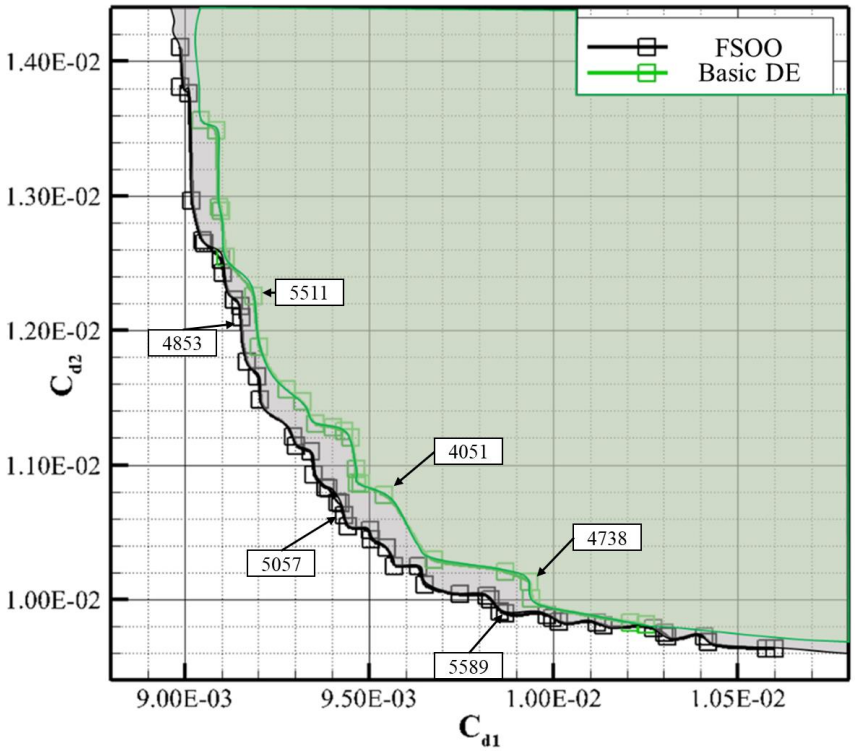
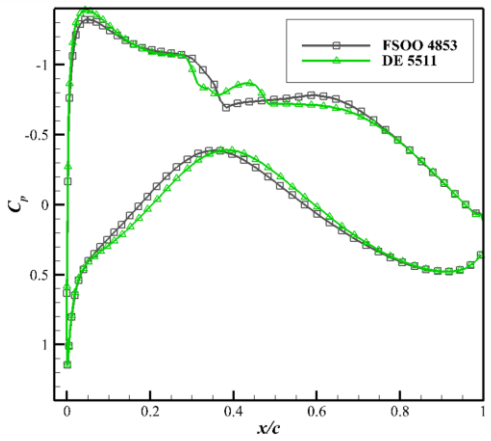
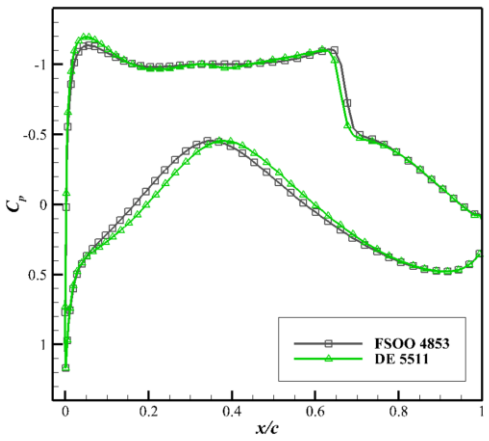


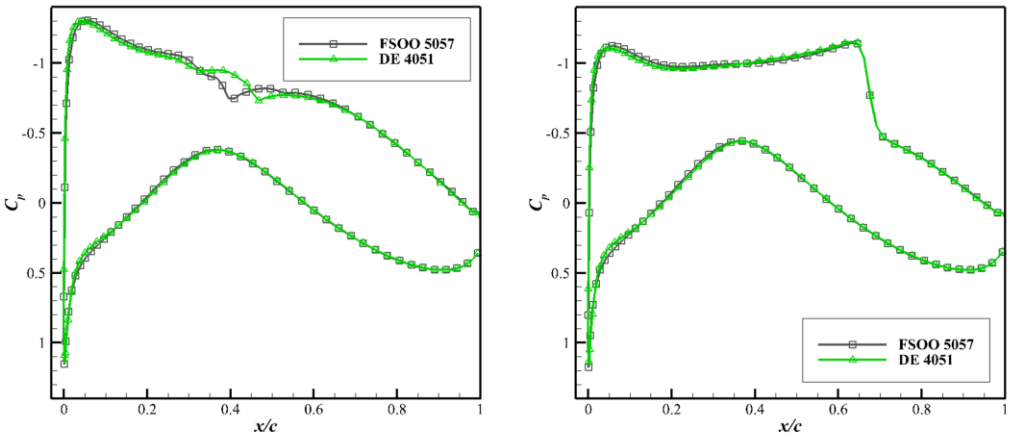
Figure 9: Pareto Front Comparison



(a). Pair A, Ma=0.72

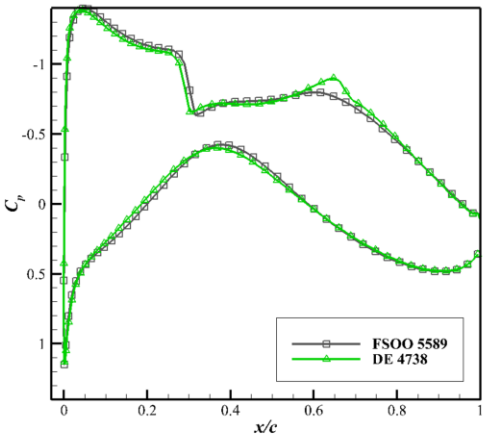


(b). Pair A, Ma=0.75

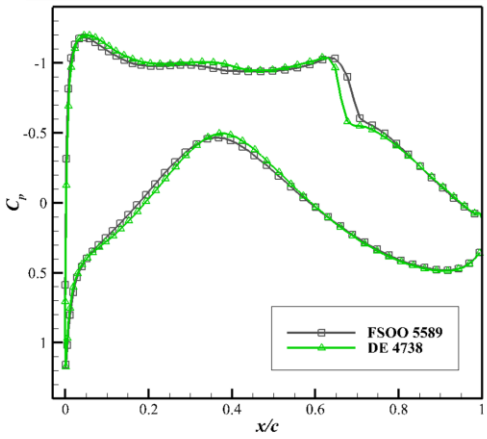


(c). Pair B, Ma=0.72

(d). Pair B, Ma=0.75



(e). Pair C, Ma=0.72



(f). Pair C, Ma=0.75

Figure 10: Surfacial Pressure Comparison of Solutions of Compting Optimizers

3.4 Discussion

The convergence curve shows that FSOO apparently converges faster than basic DE, about 40% of total iterated generations can be reduced to reach the same degree of convergence. Such speedup is within expectation because even if the individual improvement strategy is ineffective due to incorrect direction calculation by models, the improvement process degenerates to a random walk in the neighborhood of origin individuals which serves as a local search process and cannot cause serious optimization efficiency loss.

It can also be observed that the convergence curve sometimes bumps up, that is due to the selection mechanism of DE that forces the individual in parent population and offspring population to carry out one on one selection. Although such mechanism cannot guarantee to eliminate the individual with greater distance to the Pareto front, the population diversity can be well preserved.

Figure 10 shows the comparison of solutions obtained by FSOO and basic DE. (a) and (b) show that the solution 4853 obtained by FSOO has a single shock wave with middle intensity at Mach 0.72 while solution 5511 obtained by basic DE has two weak shock waves and shows apparent second acceleration in the upper surface, and solution 4853 has slightly weaker shock wave at Mach 0.75. (c) and (d) show that at Mach 0.72, solution 5057 obtained by FSOO has one weaker shock wave than solution 4051 obtained by basic DE, which leads to the superiority upon drag coefficient while their pressure distribution is nearly identical at Mach 0.75. (e) and (f) show that solution 5589 obtained by FSOO successfully avoids the apparent pressure bump at Mach 0.72 comparing to solution 5738 obtained by DE at Mach 0.72, and at Mach 0.75 the former solution has weaker shock wave intensity. Those comparisons show that FSOO is able to obtain solutions with more reasonable flow structures and better performance under the same optimization iterations.

4 Conclusions and Future Work

In this article a technical framework to achieve flow structure oriented optimization (FSOO) is proposed by utilizing convolutional autoencoder as flow field feature extractor and multi-layer perceptrons to analyze mapping relationships. Validation cases have shown that CAE is able to capture the essential flow pattern inside the flow field with profound information compression rate (from nearly 150,000 to 40, about 0.027%) without significant information loss. Utilization of FSOO in test case validates that with proper incorporation of deep learning based individual improvement, the performance and efficiency of current optimizers can be significantly boosted.

While incorporating deep learning as analysis tool within aerodynamic optimization is preliminarily proven effective in this article, there are still apparent obstacles to utilize deep learning and deep neural networks in engineering occasions. The biggest challenge here is the collection of sufficient samples for training deep neural network. The case used in this article is a 2-D optimization problem whose computation cost is acceptable and large scale of samples can be affordably collected before or during optimization. While for more complicated 3-D optimization problems and unsteady optimization problems, since the computation is extremely expensive, available samples may not be sufficient for the training process thus extra auxiliary tools need to be incorporated to tackle the insufficiency of training samples at early stages of optimization.

Aside from autoencoder, there are other network architectures that can be effectively incorporated in the optimization process, literature[47] utilizes generative adversarial network (GAN) to learn to generate flow fields that satisfies the partial differential equations (PDE) that govern the flow physics of a 2-D steady incompressible cavity according to arbitrarily defined boundary conditions, this work is enlightening since utilization of those generative models enables direct generation of the feasible ideal optimal flow field, which can be used to directly guide the optimization process in flow structure oriented optimization problems.

Acknowledgement

This work is supported by Tsinghua University Initiative Scientific Research Program (2015Z22003).

References

[1] P. C. Mosetti G, “Aerodynamic Shape Optimization by Means of a Genetic Algorithm,” in *5th International Symposium on Computational Fluid Dynamics*, 1993, pp. 279–284.

[2] S. Obavashi and S. Takanashi, “Genetic Algorithm for Aerodynamic Inverse Optimization Problems,” in *First International Conference on IET*, 1995, pp. 7–12.

[3] S. Obayashi and T. Tsukahara, “Comparison of optimization algorithms for aerodynamic shape design,” *AIAA J.*, vol. 35, no. 8, pp. 1413–1415, 1997.

[4] A. Vicini and D. Quagliarella, “Multipoint Transonic Airfoil Design by Means of a Multiobjective Genetic Algorithm,” in *35th Aerospace Sciences Meeting and Exhibit*, 1997, p. 82.

[5] A. Vicini and D. Quagliarella, “Inverse and Direct Airfoil Design Using a Multiobjective Genetic Algorithm,” *AIAA Journal*, vol. 35, no. 9. pp. 1499–1505, 1997.

[6] B. R. Jones, W. a. Crossley, and A. S. Lyrintzis, “Aerodynamic and Aeroacoustic Optimization of Rotorcraft Airfoils via a Parallel Genetic Algorithm,” *Journal of Aircraft*, vol. 37, no. 6. pp. 1088–1096, 2000.

[7] A. Oyama, S. Obayashi, and K. Nakahashi, “Real-coded Adaptive Range Genetic Algorithm Applied to Transonic Wing Optimization,” *Appl. Soft Comput.*, vol. 1, pp. 179–187, 2001.

[8] D. W. Zingg, M. Nemec, and T. H. Pulliam, “A Comparative Evaluation of Genetic and Gradient-based Algorithms Applied to Aerodynamic Optimization,” *Eur. J. Comput. Mech. Eur. Mécanique Numérique*, vol. 17, no. 1–2, pp. 103–126, 2008.

[9] A. Giunta, “Aircraft Multidisciplinary Design Optimization using Design of Experiments Theory and Response Surface Modeling Methods,” Virginia Polytechnic Inst. and State Univ., 1997.
[10] T. W. Simpson, T. M. Mauery, J. J. Korte, and F. Mistree, “Kriging models for global approximation in simulation-based multidisciplinary design optimization,” *AIAA J.*, vol. 39, no. 12, pp. 2233–2241, 2001.
[11] S. Jeong, M. Murayama, and K. Yamamoto, “Efficient optimization design method using kriging model,” *J. Aircr.*, vol. 42, no. 2, pp. 413–420, 2005.
[12] A. Forrester and A. Keane, “Recent advances in surrogate-based optimization,” *Prog. Aerosp. Sci.*, pp. 1–77, 2009.
[13] K. K. Saijal, R. Ganguli, and S. R. Viswamurthy, “Optimization of Helicopter Rotor Using Polynomial and Neural Network Metamodels,” *J. Aircr.*, vol. 48, no. 2, pp. 553–566, 2011.
[14] Z. Han and S. Görtz, “Hierarchical Kriging Model for Variable-Fidelity Surrogate Modeling,” *AIAA J.*, vol. 50, no. 9, pp. 1885–1896, 2012.
[15] E. Iuliano and E. A. Pérez, *Application of Surrogate-based Global Optimization to Aerodynamic Design*. Springer International Publishing, 2015.
[16] A. Jameson, “Optimum aerodynamic design using CFD and control theory,” *AIAA Pap.*, vol. 1729, pp. 124–131, 1995.
[17] J. Reuther, A. Jameson, J. Farmer, L. Martinelli, and D. Saunders, “Aerodynamic Shape Optimization of Complex Aircraft Configurations via an Adjoint Formulation,” in *34th Aerospace Sciences Meeting and Exhibit*, 1996, p. 94.
[18] W. K. Anderson and V. Venkatakrishnan, “Aerodynamic Design Optimization on Unstructured Grids with a Continuous Adjoint Formulation,” *Comput. Fluids*, vol. 28, no. 4, pp. 443–480, 1999.
[19] E. Arian and M. D. Salas, “Admitting the inadmissible: Adjoint formulation for incomplete cost functionals in aerodynamic optimization,” *AIAA J.*, vol. 37, no. 1, pp. 37–44, 1999.
[20] J. Reuther, J. J. Alonso, J. Martins, and S. C. Smith, “A coupled aero-structural optimization method for complete aircraft configurations,” *AIAA Pap.*, vol. 187, p. 1999, 1999.
[21] J. J. Reuther, A. Jameson, J. J. Alonso, M. J. Rimllnger, and D. Saunders, “Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, part 2,” *J. Aircr.*, vol. 36, no. 1, pp. 61–74, 1999.
[22] W. K. Anderson and D. L. Bonhaus, “Airfoil design on unstructured grids for turbulent flows,” *AIAA J.*, vol. 37, no. 2, pp. 185–191, 1999.
[23] S. Kim, J. J. Alonso, and A. Jameson, “A gradient accuracy study for the adjoint-based Navier-Stokes design method,” *AIAA Pap.*, vol. 299, 1999.
[24] J. C. Newman III, A. C. Taylor III, R. W. Barnwell, P. A. Newman, and G. J.-W. Hou, “Overview of sensitivity analysis and shape optimization for complex aerodynamic configurations,” *J. Aircr.*, vol. 36, no. 1, pp. 87–96, 1999.
[25] S. K. Kim, J. J. Alonso, and A. Jameson, “Two-dimensional high-lift aerodynamic optimization using the continuous adjoint method,” *AIAA Pap.*, vol. 4741, no. 8, 2000.
[26] S. Nadarajah and A. Jameson, “A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization,” *AIAA Pap.*, vol. 667, p. 2000, 2000.
[27] R. LI, Y. Zhang, and H. Chen, “Evolution and development of ‘man-in-loop’ in aerodynamic optimization design,” *ACTA Aerodyn. Sin.*, vol. 35, no. 4, pp. 529–543, 2017.
[28] R. Storn and K. Price, “Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces,” *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, 1997.
[29] J. P. Thomas, E. H. Dowell, and K. C. Hall, “Three-Dimensional Transonic Aeroelasticity Using Proper Orthogonal Decomposition Based Reduced Order Models,” *Proc. 42^{\rm nd} {AIAA/ASME/ASCE/AHS/ASC} Struct. Struct. Dyn. Mater. Conf. Exhib.*, vol. 40, no. 3, pp. 1–10, 2001.
[30] A. Agarwal and L. T. Biegler, “A trust-region framework for constrained optimization using reduced order modeling,” *Optim. Eng.*, vol. 14, no. 1, pp. 3–35, 2013.

[31] K. H. Park, S. O. Jun, S. M. Baek, M. H. Cho, K. J. Yee, and D. H. Lee, "Reduced-Order Model with an Artificial Neural Network for Aerostructural Design Optimization," *J. Aircr.*, vol. 50, no. 4, pp. 1106–1116, 2013.
[32] E. Iuliano and D. Quagliarella, "Aerodynamic design with physics-based surrogates," in *Springer Handbook of Computational Intelligence*, 2015, pp. 1185–1209.
[33] M. J. Zahr and C. Farhat, "Progressive construction of a parametric reduced-order model for PDE-constrained optimization," *Int. J. Numer. Methods Eng.*, vol. 102, no. 5, pp. 1111–1135, 2015.
[34] D. Amsallem, M. Zahr, Y. Choi, and C. Farhat, "Design optimization using hyper-reduced-order models," *Struct. Multidiscip. Optim.*, vol. 51, no. 4, pp. 919–940, 2015.
[35] W. Zhang, J. Kou, and Z. Wang, "Nonlinear Aerodynamic Reduced-Order Model for Limit-Cycle Oscillation and Flutter," *AIAA Pap.*, vol. 54, no. 10, pp. 3304–3311, 2016.
[36] D. J. J. Toal, N. W. Bressloff, and A. J. Keane, "Geometric Filtration Using POD for Aerodynamic Design Optimization," in *26th AIAA Applied Aerodynamics Conference*, 2008, p. 6584.
[37] E. Iuliano and D. Quagliarella, "Aerodynamic shape optimization via non-intrusive POD-based surrogate modelling," *2013 IEEE Congr. Evol. Comput. CEC 2013*, pp. 1467–1474, 2013.
[38] J. N. Kutz, "Deep learning in fluid dynamics," *J. Fluid Mech.*, vol. 814, pp. 1–4, 2017.
[39] Z. Wang, D. Xiao, F. Fang, R. Govindan, C. C. Pain, and Y. Guo, "Model identification of reduced order fluid dynamics systems using deep learning," *Int. J. Numer. Methods Fluids*, vol. 86, no. 4, pp. 255–268, 2018.
[40] O. Hennigh, "Lat-Net: Compressing Lattice Boltzmann Flow Simulations using Deep Neural Networks," 2017.
[41] J. N. Kani and A. H. Elsheikh, "DR-RNN: A deep residual recurrent neural network for model reduction," 2017.
[42] V. Dumoulin and F. Visin, "A Guide to Convolution Arithmetic for Deep Learning," *arXiv:1603.07285*, 2016. [Online]. Available: http://arxiv.org/abs/1603.07285.
[43] B. Kulfan and J. Bussoletti, "'Fundamental' Parameteric Geometry Representations for Aircraft Component Shapes," in *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2006.
[44] Y. Zhang, "Aerodynamic Optimization of Civil Aircraft Design Based on Advanced Computational Fluid Dynamics," Tsinghua University, 2010.
[45] Y. Zhang, H. Chen, and S. Fu, "Improvement to Patched Grid Technique with High-Order Conservative Remapping Method," *J. Aircr.*, vol. 48, no. 3, pp. 884–893, 2011.
[46] K. He, X. Zhang, S. Ren, and J. Sun, "ResNet," *arXiv Prepr. arXiv1512.03385v1*, vol. 7, no. 3, pp. 171–180, 2015.
[47] A. B. Farimani, J. Gomes, and V. S. Pande, "Deep Learning the Physics of Transport Phenomena," *arXiv:1709.02432 [physics]*, 2017.