A Manifold-Based Airfoil Geometric-Feature Extraction and Discrepant Data Fusion Learning Method

YU XIANG, Member, IEEE
LIWEIHU, Member, IEEE
GUANGBO ZHANG
JUN ZHANG
University of Electronic Science and Technology of China, Chengdu, China

The geometrical shape of airfoils and the corresponding flight conditions are crucial factors for aerodynamic coefficient prediction. The obtained geometric-features of airfoils in most existing approaches (e.g., geometrical parameter extraction, polynomial description, and deep learning) are in Euclidean space. State-of-the-art studies have shown that the curves or surfaces of an airfoil form a manifold in Riemannian space. Therefore, the features extracted by existing methods are not sufficient to reflect the geometric-features of airfoils. Meanwhile, flight conditions and geometric-features are greatly discrepant with different types. The discrepancy between these two factors must be considered and evaluated to improve the aerodynamic coefficient accuracy. Motivated by the advantages of manifold theory and multi-task learning (MTL), we propose a manifold-based airfoil geometric-feature extraction and discrepant data fusion learning method (MDF) to extract geometric-features of airfoils in Riemannian space (we call them manifold-features) and further fuse the manifold-features with flight conditions to predict aerodynamic coefficients. Experimental results show that our method can extract geometric-features of airfoils more accurately than existing methods, that the average mean square error (MSE) of airfoils rebuilt based on geometric-features is reduced by 41.30%, and that while keeping the same prediction accuracy level for the lift coefficient $C_L$, the MSE of the drag coefficient $C_D$ predicted by MDF is further reduced by 54.56%.

I. INTRODUCTION

The geometrical shape of an airfoil greatly affects the aerodynamic coefficient [1], [2], [3]. The most commonly used airfoil geometry description method is to define a set of airfoil geometry parameters, such as the chord length, maximum thickness, and leading edge radius, etc. Although these parameters are effective in representing the variations in airfoil geometric structures, the designers of airfoils need to manipulate them manually to obtain a smooth and continuous curve, which limits their applicability [4]. The use of polynomials is an alternative efficient mathematical approach for approximating airfoil curves (e.g., Bézier curves [5] and B-splines [6]) in which linear combinations of high-degree polynomials are usually employed to approximately describe the geometric variations of airfoil structures. Nevertheless, polynomial approaches can provide only approximate expressions for airfoils, and they are used to exploit the features of airfoils from Euclidean space. Therefore, it is difficult for polynomials to comprehensively describe some modern complex airfoils [7].

The curve of an airfoil has been shown to exist in Riemannian space [8]. Different from Euclidean space, there does not exist a single Cartesian coordinate system to which the Riemannian space consisting of curves of airfoils is homeomorphic [9]. In fact, the Riemannian space consisting of airfoil curves is homeomorphic to multiple different Euclidean spaces locally [10]. Therefore, it is more appropriate to analyze features of airfoils using manifold theory in Riemannian space than in Euclidean space. Manifold theory has been applied to the optimization of modern complex airfoil geometric structures with the manifold mapping method [11], [12]. Du et al. [13] applied manifold mapping to align a high-fidelity model and a low-fidelity model to obtain the optimal target based on the performance distribution (i.e., Mach number and pressure coefficient) in inverse design. Raul et al. [14] used manifold mapping within a surrogate-based optimization framework utilized for aerodynamic shape optimization to alleviate airfoil dynamic stalling. The above studies focused on the optimization problems of airfoils, and to the best of our knowledge, few studies have been conducted on airfoil geometric-feature extraction using manifold theory [11], [13].

In addition to the geometrical shape of airfoils, flight conditions are other factors affecting aerodynamic coefficient. Once the geometrical shape of an airfoil is fixed,
the aerodynamic coefficient is highly influenced by the flight conditions, and coefficient analysis becomes more complicated with variable geometrical shapes under different flight conditions. The influence of these two different types of data on aerodynamic coefficient is called the discrepancy of aerodynamic data. As a result, it is difficult to quantitatively evaluate the discrepant effects of geometrical shape and flight conditions on aerodynamic coefficient, which also leads to difficulty in fusing them. Recently, researchers have recognized the importance of aerodynamic data discrepancy [15] and tried to create new models that learn discrepant aerodynamic data in a distributed way according to the type of input data, for example, AeroCNN-I [16], physics-guided machine learning (PGML) [17], multilayer perceptron (MLP) [18], aerodynamic coefficient prediction model (ACPM) [19], and multitask learning (MTL) scheme [20], [21]. However, all the above studies use the coordinates or images of airfoils as representations of airfoil structures in Euclidean space.

In recent years, deep learning has achieved great success in feature extraction for modern complex airfoils and discrepant aerodynamic data fusion. For example, with radial basis function-based generative adversarial networks (RBF-GANs) [1], convolutional neural networks (CNNs) [16] and auto-Encoders [22], the feature maps of airfoils can be extracted. With MTL [20], the discrepant aerodynamic data can be fused effectively. Nevertheless, the problems with these approaches are as follows: 1) they rarely use manifold theory to extract latent geometric-features from Riemannian Space, and 2) the latent geometric-features of airfoils are not considered in the fusion method of discrepant aerodynamic data.

Motivated by manifold theory and MTL, we propose a manifold-based airfoil geometric-feature extraction and discrepant data fusion learning method (MDF). MDF is used to extract latent geometric-features in Riemannian space (called manifold-features) and to further fuse the manifold-features of airfoils with flight conditions to predict aerodynamic coefficient (see Fig. 1). Our proposed MDF consists of three modules: a manifold-based airfoil geometric-feature extraction module, flight condition input module, and multitask learning module. In the manifold-based airfoil geometric-feature extraction module, a set of self-intersection-free Bézier curves are employed to build a smooth segmented manifold from airfoil coordinates. Then, the Riemannian metric of this manifold is calculated as a manifold-feature. The smooth segmented manifold and the extracted Riemannian metric together form a smooth Riemannian manifold. In the flight condition input module, the flight conditions of airfoils are normalized. The MTL module, a discrepant data fusion learning method, is applied to fuse the extracted geometric-features of airfoils and flight conditions to further predict aerodynamic coefficient [20]. The output of MTL is the predicted aerodynamic coefficient parameters (e.g., the lift coefficient $C_L$ and drag coefficient $C_D$) of airfoils, which are used to evaluate whether the geometric-features and flight conditions can be fused to predict aerodynamic coefficient precisely.

In summary, the contributions of our work are as follows:

1) We prove that a set of self-intersection-free Bézier curves that are connected end to end form a smooth segmented Riemannian manifold that can describe the geometric shape of airfoils.
2) We propose a manifold-based airfoil geometric-feature extraction method using manifold metric calculated with the Riemannian manifold constructed above.
3) We propose a novel discrepant data fusion learning method MDF, to fuse the Riemannian manifold features of foils in Riemannian space and the flight conditions together to predict aerodynamic coefficient which outperforms existing methods.

The rest of this article is organized as follows. Section II introduces the research status of airfoil
feature extraction and discrepant data fusion in the field of airfoil-related modeling. In Section III, the prove of smooth segmented manifold and the details of MDF are described. In Section IV, a public UIUC airfoil dataset [23] is used to validate the effectiveness of MDF and the feasibility of using geometric-features to predict $C_L$ and $C_D$ of airfoils. Finally, Section V, concludes this article.

II. RELATED WORKS

In this section, we describe the current research status of airfoil feature extraction and discrepant aerodynamic data fusion.

A. Feature Extractions of Airfoils

Bézier curves, B-splines, and NURBSs are typical polynomials that have been used to derive equations for airfoil curves [5], [6], [24]. The use of these polynomials can be regarded as an effective approach for extracting features of airfoils. Among these polynomials, Bézier curves have the most basic and common expressions. An airfoil is usually represented by multiple control points $(x, y)$ of a polynomial function. Usually, an $n$-degree Bézier curve that connects $n + 1$ control points is chosen as the basis to form a smooth curve that is used to approximate a part of an airfoil function. Then, the airfoil curve function can be described as a linear combination of the basis. The polynomial expressions discussed above are flexible, and they can be combined with other parameterization methods to describe airfoil characteristics more accurately [6].

Polynomial-based class function/shape function transformation (CST) [25] is the mainstream method for extracting features in the field of airfoil parameterization. CST uses both the class function and the shape function to control the airfoil shape. The class function is applied to generate the basic shape of an airfoil, and the shape function is used to correct the basic shape to obtain an accurate airfoil shape. The coefficients of the class function and shape function are parameters to be determined in CST.

These mathematically interpretable polynomial approaches have been widely used to parameterize airfoils in Euclidean space. However, geometric shapes with curves or surfaces are more suitable for analysis in Riemannian space than in Euclidean space. Since the $n$-D Euclidean space $\mathbb{R}^n$ has linear properties. It is possible to construct a single Cartesian coordinate system $C$ in the whole Euclidean space $\mathbb{R}^n$ such that an arbitrary vector $v \in C$ corresponds to a point $p \in \mathbb{R}^n$ (i.e., $C$ is homeomorphic to $\mathbb{R}^n$). Nevertheless, there does not exist any open $\mathbb{R}^n$ to which the curves or surfaces are homeomorphic [9], which means it is impossible to construct a single Cartesian coordinate system in global curved space (i.e., Riemannian space) consisting of surfaces or curves. In fact, the curves and surfaces are homeomorphic to $\mathbb{R}^n$ locally [9], [10], which indicates that a small neighborhood $Q_i$, $i = 1, 2, \ldots, n$ in Riemannian space can be homeomorphic to $\mathbb{R}^n$. Riemannian space, as a generalization of Euclidean space, can combine the above $\mathbb{R}^n$. Therefore, geometric shapes with curves or surfaces should be analyzed in Riemannian space instead of Euclidean space. In addition, studies have shown that the curve of an airfoil exists in manifold space [8], [11], [13]. Hence, existing polynomial approaches can only capture geometric-features from Euclidean space, and some latent geometric-features (e.g., manifold-features from Riemannian space) are omitted. In contrast, manifold theory can extract geometric-features from the perspective of manifold space and further enrich airfoil features.

B. Discrepant Aerodynamic Data Fusion

The MTL scheme is an effective approach for fusing discrepant aerodynamic data [20], [26]. For neural networks without dropout, all neurons are activated for each input sample. This mechanism is difficult to adapt to discrepancies in aerodynamic data. In contrast, MTL is a partially activated neural network that activates different neurons according to different inputs.

The origin of MTL can be traced back to mixtures of experts (i.e., dedicated neural networks) in natural language processing [27]. The idea of mixtures of experts is that only one specific expert network is activated to analyze a word according to the part-of-speech tags of the input word. The final results are obtained by linearly weighting the results of all experts [28]. Experts can be different models, such as support vector machines (SVMs) [29], Gaussian processes [30], and neural networks [31], etc. After 2019, the mechanism of partial activation of neurons was developed in the form of MTL in the field of aerodynamics.

MTL is a novel neural network in which different tasks are assigned to different subnetworks. White et al. proposed ClusterNet (a precursor to MTL) in 2020, which is used to predict the velocity of air flows [26]. Zhang et al. adopted a ClusterNet-based physics-informed model for predicting future swarm trajectories by approximating the nonlinear dynamics of the swarm model [32]. Based on ClusterNet, Hu and Zhang et al. divided an aerodynamic dataset into different subtasks according to their discrepancy and proposed the MTL method to further fuse them to predict aerodynamic coefficient [20], [21].

By adopting the state-of-the-art MTL scheme, the above studies focused on the discrepancy of aerodynamic data. However, they used the coordinates or images of airfoils as representations of airfoil structures in Euclidean space and did not consider extracting latent geometric-features of airfoils from Riemannian space.

In conclusion, our proposed MDF is different from both the existing airfoil feature extraction methods and discrepant data fusion. The existing airfoil feature extraction methods output polynomials with certain coefficients, which are used to represent the structure of airfoils. However, a polynomial with certain coefficients is just an intermediate output of MDF. The final output of MDF is the predicted coefficient parameters. In addition, the input of the existing discrepant data fusion methods is the discrepant data from Euclidean space. However, MDF fuses the
manifold-features extracted from Riemannian space and the flight conditions from Euclidean space together to predict the aerodynamic coefficient of airfoils.

III. METHODOLOGY

A. Overview of MDF

Fig. 1 describes the proposed structure of MDF. MDF is mainly composed of three modules: a manifold-based airfoil geometric-feature extraction module, flight condition input module, and MTL module. In the manifold-based airfoil geometric-feature extraction module, first, a set of self-intersection-free Bézier curves that are connected end to end are used to form a smooth segmented manifold within the airfoil shape space. Then, the manifold metric calculation module calculates the Riemannian metric on the manifold that forms a Riemannian manifold. The Riemannian metric, calculated as the inner-product of two arbitrary vectors from the tangent space of the Riemannian manifold, forms the basis for other manifold-features of the Riemannian manifold. Through which the length, volume, connection, and other manifold-features of the Riemannian manifold can be calculated [33].

The Riemannian metric can be calculated as a basic representation of manifold-features from the tangent space of our smooth segmented Riemannian manifold within airfoil shape space. In the flight condition input module, the flight conditions of airfoils are normalized. In the MTL module, the calculated Riemannian metric \( \mathbf{x}_1 \) of airfoils is taken as one input to function network_1, and the corresponding flight conditions \( \mathbf{x}_2 \) are taken as another input to function network_2. The inputs of the context network in the MTL module are the vectorized combinations of the Riemannian metric of airfoils and the flight conditions (i.e., \( [\mathbf{x}_1, \mathbf{x}_2] \)). The output of the MTL module is the predicted aerodynamic coefficient \( \mathbf{y} \). In this section, we mainly introduce the manifold-based airfoil geometric-feature extraction module and the MTL module in detail.

B. Manifold-Based Airfoil Geometric-Feature Extraction Module

1) Construction of the Smooth Segmented Manifold of Airfoils: In this section, first, we prove that a self-intersection-free Bézier curve forms a smooth manifold.

**THEOREM 1** A self-intersection-free Bézier curve forms a smooth manifold.

**PROOF** We consider a 2-D airfoil coordinate set \( D = \{P_i = (x_i, y_i)\}_{i=1}^M \), where \( M \) denotes the number of coordinate points. A Bézier curve can be built

\[
r(D; t) = \sum_{i=0}^{n} P_i B_{i,n}(t), \quad t \in [0, 1]
\]

where \( r(D; t) \) is the function of a Bézier curve, \( D \) denotes the sample space where the airfoil coordinates locates, \( t \) is the parameter of the Bézier curve, \( n \) is the degree of the Bézier curve, and \( B_{i,n}(t) \) denotes the coefficient, which satisfies

\[
B_{i,n}(t) = \binom{n}{i}(1-t)^{n-i} = \frac{n!}{i!(n-i)!} (1-t)^{n-i} [i = 0, 1, \ldots, n]
\]

An \( n \)-degree Bézier curve with \( n \geq 3 \) may have self-intersections [see Fig. 2(a)], in which case it cannot be used to construct a manifold [34]. There are two solutions to avoid self-intersections [see Fig. 2b and 2c]. In subgraph (b), the self-intersection \( A \) is deleted, and the remaining curves construct a manifold. In subgraph (c), the sequence (or positions) of four control points makes the Bézier curve have no self-intersections. A Bézier curve without self-intersections is the prerequisite of Theorem 1.

We introduce the following lemma [35, 36].

**LEMMA 1** A nonempty Hausdorff space \( \mathcal{M} \) is called an \( m \)-dimensional topological manifold if for every point \( p \in \mathcal{M} \), there exists an open neighborhood \( U' \) of the point \( p \) that satisfies \( U' \subset \mathcal{M} \) and a homeomorphism \( \phi : U' \rightarrow \mathbb{R}^m \) from \( U' \) to an open set of the \( m \)-dimensional Euclidean space \( \mathbb{R}^m \).

We assume that \( \mathcal{M}^* \) is the space where a Bézier curve \( r(D; t) \) is located and that \( \mathcal{M}^* \subset \mathcal{M} \). Then, \( \exists T \), such that \( (r, T) \) constitutes a topological space (i.e., \( \mathcal{M}^* \) is a nonempty Hausdorff space). This is the prerequisite of Lemma 1. To describe the proof easily, we let \( \mathcal{M}^* = \mathcal{M} \).

Let the Euclidean space where \( D = \{P_i = (x_i, y_i)\}_{i=1}^M \) locates be \( \mathbb{R}^2 \). Because of the smoothness of \( r(D; t) \), for \( \forall t_0, t \in r(D; t) \), there must exist an open neighborhood \( U' \) such that \( t_0 \in U' \). For \( \forall t_1, t_2 \in U' \) and \( t_1 \neq t_2 \), \( \exists P_1 \in \mathbb{R}^2 \) such that \( r : t_1 \rightarrow P_1 \), and \( \exists P_2 \in \mathbb{R}^2 \) such that \( r : t_2 \rightarrow P_2 \).

Suppose that \( P_1 = P_2 \), then, \( P_1 = r(D; t_1) \) and \( P_2 = r(D; t_2) \); therefore, \( r(D; t_1) = r(D; t_2) \), which contradicts the functional properties of \( r(D; t) \) (i.e., \( P_1 \neq P_2 \)). \( r(D; t) \) is a homeomorphism from \( U' \) to \( \mathbb{R}^2 \).

According to Lemma 1, there exists a homeomorphism \( r(D; t) : U' \rightarrow \mathbb{R}^2 \), such that the set of all elements in \( \mathcal{M}^* \) is a 2-D topological manifold. \( \square \)
To approximate a simple airfoil, it is sufficient to use a simple curve with one Bézier function. However, to approximate a complex airfoil, as the number of control points of the Bézier curve increases, the degree of the Bézier curve also increases, which could result in a larger error [7]. Instead, it is necessary to separate the airfoil curve into multiple segments and use a set of Bézier curves to approximate a complex airfoil. In this article, multiple Bézier curve segments are end-to-end connected, and each Bézier curve segment is determined by four control points (see Fig. 3). In this figure, control points A, B, C, and D determine segment 1; the control points D, E, F, and G determine segment 2; etc. Multiple smooth segments are connected end to end to form a segmented smooth curve of the airfoil.

Next, we prove that a set of Bézier curves are connected end to end to form a smooth segmented topological manifold.

**Theorem 2** Multiple smooth topological manifolds constructed by a set of Bézier curves are connected end to end to form a smooth segmented manifold.

**Proof** To prove Theorem 2, we begin with an arbitrary smooth function \( r_i(D; t) \) defined on \( \mathcal{M}_i \), where \( i = 1, 2, \ldots, N \), and \( N \) denotes the number of segments. We need to prove that \( r_i(D; t) \) can be extended to the whole manifold \( \mathcal{M} \), which satisfies \( \mathcal{M} = \bigcup_{i=1}^{N} \mathcal{M}_i \). Therefore, Theorem 2 can be rewritten in the following form.

Let \( U \) be an open set in a smooth manifold \( \mathcal{M} \), and \( r(D; t) \in C^1(U) \), for \( \forall P_i \in U \), there must exist a neighborhood \( W \) that satisfies \( P_i \in W \subset \mathcal{M} \subset U \) and function \( \tilde{r}_i(D; t) \in C^1(W) \), such that \( \tilde{r}_i(D; t)|_{W} = r(D; t)|_{W} \).

To prove \( r(D; t)|_{W} = r(D; t)|_{W} \), we introduce a lemma of cutoff function [35, 36].

**Lemma 2** Let \( V', V \) be two open subsets in a smooth topological manifold \( \mathcal{M} \), and \( \tilde{V} \) is a compact subset that satisfies \( V' \subset V \). Then, \( \exists f \in C^\infty(M) \), such that \( 0 \leq f \leq 1 \) and \( f|_{V'} = 1 \), \( f|_{\mathcal{M}\setminus V} = 0 \).

According to Lemma 2, an arbitrary function defined on \( \mathcal{M} \) can be smoothly truncated by multiplying by \( f \):

\[
\begin{align*}
(f \cdot r(D; t))|_{V'} &\equiv r(D; t)|_{V'} \\
(f \cdot r(D; t))|_{\mathcal{M}\setminus V} &\equiv 0. \\
\end{align*}
\]

For an arbitrary \( P_j \in U \), we consider two open neighborhoods \( W \) and \( V \) such that \( W \) and \( V \) are compact and \( W \subset V \subset V \subset U \). According to Lemma 2 and (3), \( \exists f \in C^1(\mathcal{M}) \), such that \( f|_{W} = 1 \), \( f|_{\mathcal{M}\setminus V} = 0 \).

For \( \forall P_j = r(D; t_j) \in \mathcal{M} \), we let

\[
\tilde{r}_i(D; t_j) = \begin{cases} 
 r(D; t_j)f(P_j), & P_j \in U \\
 0, & P_j \notin U. 
\end{cases}
\]

Because \( r(\cdot)f(\cdot) \) is smooth in subset \( U \) and \( r(D; t_j)f(P_j) \equiv 0 \) in \( U \cap (\mathcal{M} \setminus V) \), according to (4), the function \( \tilde{r}_i(D; t) \) is smooth in subset \( U \) and satisfies \( \tilde{r}_i(D; t_j) \equiv 0 \) in \( U \cap (\mathcal{M} \setminus V) \). Because \( \mathcal{M} = U \cup (\mathcal{M} \setminus V) \), \( \tilde{r}_i(D; t) \) is a function defined in \( \mathcal{M} \).

According to Theorem 1, a Bézier curve forms a smooth manifold, and according to Theorem 2, a set of Bézier curves are connected end to end to form a smooth segmented manifold. Combining these two theorems together, we learn that it is feasible to separate a modern complex airfoil curve into segments and that a set of low-degree Bézier curves can be applied to approximate the segments.

2) Calculation of the Riemannian Metric: Manifolds from which a set of Riemannian metrics can be calculated are called Riemannian manifolds [35, 37, 38]. The construction of a Riemannian manifold builds a bridge between 2-D airfoil Euclidean space \( \mathbb{R}^2 = \{x_i \in \mathbb{R}, y_i \in \mathbb{R}\} \) and Riemannian space \( \mathcal{M} \) with the parameter \( t \). The Riemannian metric at an arbitrary point \( t \) can be calculated as [39]

\[
g_{vw}(t) = g_{vv}(t) = (\partial_v r(D; t)) \partial_v r(D; t) = g_{vv}(t).
\]

where \( g_{vw}(t) \) is the Riemannian metric of an airfoil at point \( t \) and \( \partial_v = \frac{\partial}{\partial v} \) denotes the direction of the partial derivative.

Since \( r(D; t) \) is a 1-D manifold, \( v = w \):

\[
g_{vw}(t) = g_{vv}(t) = (\partial_v r(D; t))^2 = g_{vv}(t).
\]

According to (5), the Riemannian metric \( g_{vw}(t) \) is the result of the inner-product of two arbitrary vectors in the tangent space \( \frac{\partial}{\partial v} \) of \( r(D; t) \), and the two vectors that make up this inner-product can be extended to the entire tangent space of the built Riemannian manifold. In other words, the Riemannian metric represents a sort of geometric-basis from the tangent space of airfoils, through which many manifold-features of the Riemannian manifold can be further measured and calculated. As a result, \( g_{vw}(t) \) can be chosen as a manifold-feature that represents the geometrical characteristics of the airfoil curves.

C. Multitask Learning (MTL) Module

In this section, we introduce the MTL module from two aspects: the structure and the training method.

1) Structure: As shown in Fig. 1, the MTL module consists of two function networks and a context network. Each function network learns one of the discrepant data (i.e., function network 1 learns the Riemannian metric \( \bar{x}_1 \), and function network 2 learns the flight conditions \( \bar{x}_2 \)). The context network learns the strategy (i.e., the fusion weights) for fusing these two discrepant data (i.e., \( [\bar{x}_1, \bar{x}_2] \)).

The Riemannian metric \( \bar{x}_1 \) and flight conditions \( \bar{x}_2 \) can be expressed as

\[
\begin{align*}
\bar{x}_1 &= g_{vw}(t)|_{t \in [0,1]} \\
\bar{x}_2 &= [Ma, \alpha, \Phi, \ldots] 
\end{align*}
\]

Xiang ET AL.: MANIFOLD-BASED AIRFOIL GEOMETRIC-FEATURE EXTRACTION
where $Ma$ denotes the incoming Mach number, $\alpha$ denotes the angle of attack, and $\Phi$ denotes the roll angle of the flow.

The outputs of the MTL module are the aerodynamic coefficient results $\tilde{y}$:

$$\tilde{y} = \{y_i | i = 1, 2, \ldots, K\}$$

$$= \sum_{m=1}^{K} f_{1,m}(\tilde{x}_1) * c_m((\tilde{x}_1, \tilde{x}_2)) + \sum_{n=1}^{M} f_{2,n}(\tilde{x}_2) * c_{K+n}((\tilde{x}_1, \tilde{x}_2))$$

(8)

where $\tilde{y}$ denotes the output vector, $f_{1,m}(\tilde{x}_1)$ denotes the $m$th component of the output vector from function network_1, $f_{2,n}(\tilde{x}_2)$ denotes the $n$th component of the output vector from function network_2, $K$ denotes the number of output nodes in the function networks, $c_m((\tilde{x}_1, \tilde{x}_2))$ denotes the $m$th component of the output vector from the context network, and $[\tilde{x}_1, \tilde{x}_2]$ denotes the concatenation of $\tilde{x}_1$ and $\tilde{x}_2$.

2) Training Method: In MTL, the function network and the context network are trained alternately. The training method is described in detail as follows.

Step 1: Forward propagation. The loss function of MTL is

$$E = \frac{1}{N} \sum_{i=1}^{N} \left( \frac{1}{K} \sum_{i=1}^{K} (y_i - \tilde{y}_i)^2 \right)$$

(9)

where $N$ denotes the number of data points in the dataset, $y_i$ denotes the real value in the training set, and $\tilde{y}_i$ denotes the predicted value.

Step 2: Function network updates. The parameters of the two function networks are updated by

$$\theta_f = \theta_f - \eta \frac{\partial E}{\partial \theta_f}, \eta > 0$$

(10)

where $\eta$ denotes the learning rate of MTL and $\theta_f$ denotes the parameters of the function networks.

Step 3: Context network updates. The parameters of the context network are updated by

$$\theta_c = \theta_c - \eta \frac{\partial E}{\partial \theta_c}, \eta > 0$$

(11)

where $\theta_c$ denotes the parameters of the context network.

Step 4: Step 1 to Step 3 are repeated until MTL converges.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

To validate the feasibility of our proposed method, two categories of experiments based on the UIUC\(^1\) airfoil dataset are conducted. Experiment I is designed to compare the results of airfoil geometric-feature extraction with various methods. Experiment II is designed to compare the aerodynamic coefficient prediction errors when discrepant data are fused by different methods.

---

\(^{1}\)[Online]. Available: https://m-selig.ae.illinois.edu/ads/coord_database.html

A. Dataset and Preprocessing

The UIUC dataset provides multiple real airfoils, each of which is discretized by 2-D coordinates. The preprocessing of the UIUC airfoil data are as follows: First, we remove airfoils with only upper or lower surfaces to guarantee that the remaining airfoils in the dataset are complete. Second, we sort the remaining airfoil coordinates in the order of the trailing edge, upper surface, leading edge, and lower surface to avoid self-intersections. Finally, we smooth the airfoil coordinates through a set of third-degree Bézier curves and then resample 279 coordinates for each airfoil uniformly. Because the number of airfoils in the UIUC dataset after preprocessing is limited, a random noise $\mathcal{N}$ is added to each airfoil to generate 5264 different airfoils. $\mathcal{N}$ is calculated by

$$\mathcal{N} = N(\mu, \sigma^2)$$

$$\mu = \frac{1}{6} \times \text{AVG}(T_i)$$

$$\sigma = \frac{1}{10} \times \text{AVG}(T_i)$$

(12)

where $T_i$ denotes the thickness of the $i$th airfoil, $\text{AVG}()$ denotes the average function. After calculation, $\text{AVG}(T_i) = 6.0 \times 10^{-3}$, $\mu = 1.0 \times 10^{-3}$, and $\sigma = 6.0 \times 10^{-4}$.

Since the geometrical parameters, coordinates, and manifold-features of airfoils, as various inputs, are compared experimentally, we describe their calculation in this paragraph. For the calculation of the geometrical parameters of airfoils, we choose seven different variables as the geometrical parameters: the chord length, maximum camber, position of maximum camber, maximum thickness, position of maximum thickness, leading edge radius, and trailing edge thickness. We use Profili to calculate the above geometrical parameters [40]. For the manifold-features of airfoils, we calculate the Riemannian metrics of airfoils at different points $t$ according to (5). The flight conditions (i.e., $Ma$ and $\alpha$) and the coefficients (i.e., $C_L$ and $C_D$) of the corresponding airfoils are obtained using the webfoil platform [41]. The numbers of inputs in the UIUC dataset after preprocessing are summarized in Table I.

B. Experiment I: Geometric-Features Comparative Experiments

In the field of feature extraction of airfoils, autoencoders are commonly used to rebuild airfoils because of their ability to extract features from the geometrical shape of airfoils and represent the airfoils as latent feature vectors [42], [43]. In addition, autoencoders reflect the accuracy of extracted features by calculating the similarities between rebuilt airfoils and real airfoils [44], [45]. Therefore, airfoils rebuilt using autoencoders with coordinates or manifold-features taken as inputs are compared.

1) Model Settings: The hyperparameters used in these experiments are shown in Table II. All the deep learning models compared in this article are implemented based on the PyTorch framework, and they are trained for 2000 epochs on four Tesla K80 GPUs [51]. To conduct tenfold cross validation [52], the whole dataset is randomly divided into ten subsets. Of the ten subsets, one subset is selected...
TABLE I
Numbers of Inputs in the UIUC Dataset After Preprocessing

| data type       | geometrical parameters | manifold-features | airfoil coordinates | flight conditions |
|-----------------|------------------------|-------------------|---------------------|-------------------|
| number of variables | 7                      | 558               | 279 × 2             | 2                 |

TABLE II
Hyperparameters of the Deep Learning Models in This Article

| parameter type         | parameter name          | value             |
|------------------------|-------------------------|-------------------|
| hardware platform      |                         |                   |
| software platform      |                         |                   |
|                        | programming language    | Python 3.9.0      |
|                        | deep learning framework | PyTorch 1.11.0    |
| optimization algorithm |                         |                   |
| data normalization     | Adam(β=0.9 and β=0.999)[46], [47], [48], [49] |
| activation function    | max-min normalization   |                   |
| batch normalization    |                         |                   |
| batch size             |                         | 128               |
| learning rate          |                         | 0.001             |
| epochs                 |                         | 2000              |
| validation             |                         | 10-fold cross validation |
| training set: validation set: testing set | | 8: 1: 1 |

TABLE III
Test MSEs of Five Typical Airfoils Rebuilt by Four Autoencoders and Their Average Training Time (s)

| airfoils | Auto-Encoder-I [45] | Auto-Encoder-II | Auto-Encoder-III [53] | Auto-Encoder-IV | \( \eta \) |
|----------|---------------------|----------------|------------------------|----------------|---------|
| airfoil_1 | 5.87 × 10⁻⁶ | 3.90 × 10⁻² | 1.69 × 10⁻³ | 1.13 × 10⁻³ | 34.67% |
| airfoil_2 | 6.56 × 10⁻⁶ | 2.22 × 10⁻⁴ | 1.49 × 10⁻⁴ | 1.00 × 10⁻⁴ | 66.16% |
| airfoil_3 | 1.04 × 10⁻³ | 1.76 × 10⁻⁶ | 2.14 × 10⁻⁶ | 9.98 × 10⁻⁴ | 82.88% |
| airfoil_4 | 1.82 × 10⁻³ | 1.34 × 10⁻⁶ | 2.20 × 10⁻⁶ | 1.31 × 10⁻³ | 37.36% |
| airfoil_5 | 1.24 × 10⁻³ | 7.07 × 10⁻⁶ | 1.48 × 10⁻⁶ | 1.18 × 10⁻³ | 47.24% |
| average MSEs of 527 airfoils in testset | 9.08 × 10⁻⁶ | 5.33 × 10⁻⁴ | 1.78 × 10⁻⁵ | 1.12 × 10⁻⁵ | 41.30% |
| average training time (s) | 18639.90 | 8743.32 | 14559.80 | 8460.99 | ——— |

as the testing set, and the remaining nine subsets are selected for the training process. Among the nine subsets, we randomly select one subset as the validation set, and the remaining eight subsets are used as the training set, i.e., the training set, validation set, and testing set are randomly divided at a ratio of 8:1:1. There is no intersection among the subsets, and every subset has the same probability of being chosen as the testing set. The statistical errors in this article are the average errors across tenfolds.

We design four different autoencoders. To compare the effects of different structures on experimental results, the optional numbers of convolution and fully connection layers in an encoder are 2, 3, 4, and 5, respectively. Therefore, every autoencoder produces four different structures. The optimal structure of the four autoencoders is shown in Fig. 4. Autoencoder-I [45] and Autoencoder-III [53] take airfoil coordinates as inputs, Autoencoder-II and Autoencoder-IV take Riemannian metrics as inputs. The outputs of all autoencoders are airfoil coordinates.

The mean square error (MSE) is chosen as the loss function of the above autoencoders.

\[
\mathcal{L}_{AE-i} = \frac{1}{N} \sum_{z=1}^{N} (AE_i (P_z) - P_z)^2
\]

\[
\mathcal{L}_{AE-II} = \frac{1}{N} \sum_{z=1}^{N} (AE_2 (g_z) - P_z)^2
\]

\[
\mathcal{L}_{AE-III} = \frac{1}{N} \sum_{z=1}^{N} (AE_3 (P_z) - P_z)^2
\]

\[
\mathcal{L}_{AE-IV} = \frac{1}{N} \sum_{z=1}^{N} (AE_4 (g_z) - P_z)^2
\]

where \( \mathcal{L}_{AE-i} \) denotes the loss function of autoencoder-i, \( AE_i(\cdot) \) denotes the function map of autoencoder-i, \( P_z = \bigcup_{i=1}^{M} P_i \) is a matrix with dimensions \( M \times 2 \) that represents the \( z \)th airfoil coordinates, and \( g_z = \{ g_{vw}(t) \}_{t \in [0,1]} \) denotes the vector of manifold metrics of the \( z \)th airfoil.

2) Results and Analyses: The test MSEs and average training time of the four autoencoders on five typical airfoils are shown in Table III. In terms of test MSEs, comparing Autoencoder-II with Autoencoder-I, we see that the MSEs of the five rebuilt airfoils generated by Autoencoder-II are always smaller than those generated by Autoencoder-I. Comparing Autoencoder-IV with Autoencoder-III, the MSEs of the five rebuilt airfoils generated by Autoencoder-IV are always smaller than those generated by Autoencoder-III. To compare the above autoencoders objectively, we calculate the average MSEs of 527 rebuilt airfoils in testing set. We see that the Autoencoder-II has the smallest average MSEs, followed by Autoencoder-I. In addition, the MSE reduction \( \eta \) of Autoencoder-II compared with Autoencoder-I is calculated as

\[
\eta = \frac{\| \sigma_I - \sigma_{II} \|}{\| \sigma_I \|} \times 100\%
\]

where \( \sigma_I \) denotes the test MSE of Autoencoder-I and \( \sigma_{II} \) denotes the test MSE of Autoencoder-II. The calculation result shows that the average MSE of Autoencoder-II is 41.30% lower than that of Autoencoder-I.

In terms of average training time, Autoencoder-II and Autoencoder-IV have shorter training time than other Autoencoders. There are two reasons: 1) theoretically, the Riemannian metric is a manifold-feature, there is no necessary to adopt complex layers for further feature extractions; 2) the Autoencoder-II and Autoencoder-IV are constructed...
Fig. 4. Structures of Autoencoder-I, Autoencoder-II, Autoencoder-III, and Autoencoder-IV. In Autoencoder-I, the first convolution layer consists of sixteen $2 \times 3$ filters of stride 2, and the padding of the convolution layer is “SAME.” The remaining convolution layers are similar.

Based on FCN such that their forward and back propagations do not have to face complex operations such as convolutions.

Fig. 5 depicts the above five airfoils rebuilt by the four autoencoders. Although we use a $10^\circ$ Bézier curve to smooth all rebuilt airfoils, all the airfoils rebuilt by Autoencoder-I, Autoencoder-III, and Autoencoder-IV are still not smooth, which means that they cannot be applied in the field of airfoil design. In contrast, the airfoils rebuilt by Autoencoder-II are closer to the real airfoils and smoother than those rebuilt by other Autoencoders. For example, in subgraph (c) and (d), the red airfoil curves (airfoils rebuilt by Autoencoder-II) are smoother than other rebuilt airfoil curves. In addition, the red airfoil curves almost coincide with the black airfoil curves (the real airfoils). From another perspective, the rebuilt airfoil curves generated by Autoencoder-II and Autoencoder-IV (both take the Riemannian metrics as inputs) are more accurate and smoother than those generated by Autoencoder-I and Autoencoder-III (both take airfoil coordinates as inputs).

Experiments I demonstrates that Riemannian metrics, as a sort of manifold-feature, can better reflect the geometrical properties of airfoils than coordinates.

C. Experiment II: Aerodynamic Performance Prediction

Experiments

In this experiment, we compare multiple methods in terms of discrepant data fusion and aerodynamic coefficient prediction. The methods are MTL\(_g\), MTL\(_c\) [20], RBF-GAN [1], PGML [17], MLP [18], ACPM [19], and MDF.

1) Model Settings: All methods in this experiment are verified by tenfold cross validation (i.e., each method is repeated ten times). All the statistic errors in this section are the average errors across tenfolds. The hyperparameters in experiments II are the same as those in experiments I (see Table II).

Most methods in this experiment consist of two different components: convolution layers and fully connected layers. To compare the effects of different structures on experimental results, the optional layer numbers of convolution and fully connection are 3, 4, and 5, respectively. Meanwhile, the optional kernel sizes of convolution layers are $2 \times 2$ and $3 \times 3$, the optional numbers of channels can be 2, 8, 16, 32, or 64. Therefore, MTL\(_g\), MTL\(_c\), RBF-GAN, PGML, MLP, ACPM, and MDF produce 27, 90, 3, 3, 3, 90, and 27 different structures, respectively. The optimal structures of the above
TABLE IV
Detailed Results on the Methods Compared in Experiment-II

| method       | $x_1$               | $x_2$               | structure of $f_1$ | structure of $f_2$ | structure of $c$ |
|--------------|---------------------|---------------------|--------------------|--------------------|-----------------|
| MTL_g        | geometrical parameters (1×7) | flight conditions $(M_a$ and $\alpha)$ | FCN: 32×3         | FCN: 8×3           | FCN: 32×3       |
| MTL_c        | coordinates (279×2)  | flight conditions $(M_a$ and $\alpha)$ | conv_1:kernel=2×2×16, stride=2 | FCN: 16×3         | FCN: 512×3      |
| RBF-GAN      | coordinates (279×2)  | flight conditions $(M_a$ and $\alpha)$ | generator FCN: 1024×3 | discriminator RBF: 512×1 | —                |
| PGML         | coordinates (279×2)  | flight conditions $(M_a$ and $\alpha)$ | FCN: 1024×3       | FCN: 1024×1        | —                |
| MLP          | coordinates (279×2)  | flight conditions $(M_a$ and $\alpha)$ | FCN: 1024×3       | —                  | —                |
| ACPM         | coordinates (279×2)  | flight conditions $(M_a$ and $\alpha)$ | conv_1:kernel=3×3×2, stride=2 | —                  | —                |
| MDF (our method) | manifold-features (1×558) | flight conditions $(M_a$ and $\alpha)$ | FCN: 1024×3       | FCN: 16×3           | FCN: 512×3      |

TABLE V
Average MSEs, MAEs and Training Time (s) of MTL_g, MTL_c, RBF-GAN, PGML, MLP, and MDF

| model          | MSE of $C_{12}$ | MAE of $C_{12}$ | MSE of $C_{12}$ | MAE of $C_{12}$ | average training time (s) |
|----------------|-----------------|-----------------|-----------------|-----------------|--------------------------|
| MTL_g          | 7.41 × 10^{-8}  | 1.57 × 10^{-4}  | 6.27 × 10^{-4}  | 1.03 × 10^{-2}  | 1970.96                  |
| MTL_c [20]     | 6.77 × 10^{-8}  | 1.56 × 10^{-4}  | 1.13 × 10^{-4}  | 8.15 × 10^{-3}  | 2017.72                  |
| RBF-GAN [1]    | 1.30 × 10^{-5}  | 1.96 × 10^{-3}  | 3.42 × 10^{-2}  | 9.34 × 10^{-2}  | 2930.31                  |
| PGML [17]      | 1.29 × 10^{-6}  | 2.78 × 10^{-4}  | 1.67 × 10^{-3}  | 1.35 × 10^{-2}  | 1905.75                  |
| MLP [18]       | 7.29 × 10^{-8}  | 1.38 × 10^{-4}  | 1.07 × 10^{-4}  | 7.90 × 10^{-3}  | 1983.28                  |
| ACPM [19]      | 7.35 × 10^{-8}  | 1.25 × 10^{-4}  | 7.53 × 10^{-5}  | 6.66 × 10^{-4}  | 2283.61                  |
| MDF (our method) | 3.34 × 10^{-8}  | 1.07 × 10^{-4}  | 8.95 × 10^{-5}  | 7.16 × 10^{-3}  | 2231.26                  |
| error reduction (MDF vs. ACPM) | 54.56% | 14.40% | −15.87% | −6.98% | — |

Methods are shown in Table IV. MTL_g takes geometrical parameters and flight conditions $(M_a$ and $\alpha)$ as inputs. Considering that the geometrical parameters constitute a vector with dimensions $1 \times 7$, the structure of function network_1 in MTL_g is set to “FCN: 32×3,” which indicates that the FCN-based function network_1 has 3 layers, each of which has 32 neurons. The structure of the context network in MTL_g is set to “FCN: 32×3.”

MTL_c takes coordinates as inputs. Because the coordinates of an airfoil are organized as a matrix with dimensions 281 × 2, the function network_1 in MTL_c is designed as a CNN. CNN-based function network_1 consists of three convolutional layers and one max pooling layer. The convolution kernels of the three convolutional layers have dimensions $2 \times 2 \times 16$, $2 \times 2 \times 32$, and $2 \times 2 \times 64$. The kernel dimensions of the max pooling layer are set to $2 \times 2 \times 64$. The stride of all kernels is 2. In addition, the structure of function network_2 in MTL_c is set to “FCN: 16×3,” and the structure of the context network in MTL_g is set to “FCN: 32×3.”

RBF-GAN, whose RBF-based discriminator has been proven to be the optimal approximation of a discriminator, can capture more accurate details of flow fields and further predict the variations in flow fields [1]. Therefore, RBF-GAN is chosen as a comparative method that takes both geometric-features and flight conditions as inputs to predict aerodynamic coefficients. Considering that the discriminator of RBF-GAN is an RBFNN, the coordinate matrix with dimensions $279 \times 2$ is expanded into a vector with dimensions $1 \times 558$. The input of the RBF-based discriminator is a combination of the expanded coordinate vector and flight condition vector. The structure of the RBF-based discriminator is set to “RBF: 512×1.” In addition, according to [1], the structure of the corresponding generator is set to “FCN: 1024×3.”

PGML is used to fuse the coordinates of airfoils and flight conditions in [17]. PGML consists of two FCN-based subnetworks. The first subnetwork takes airfoil coordinates as inputs, and the structure is set to “FCN: 1024×3.” The second subnetwork takes the flight conditions as inputs, and the structure is set to “FCN: 1024×1.” The last layer of each of the two subnetworks is concatenated by an FCN to enable the fusion of the coordinates of airfoils and flight conditions.

MLP is a complete FCN whose input is a combination of airfoil coordinates and flight conditions. MLP is the most intuitive fusion method that takes a combination of airfoil coordinates and flight conditions as inputs. The purpose of selecting MLP as a comparative method is to compare this intuitive fusion method with MDF. The structure of MLP is set to “FCN: 1024×3.”

ACPM takes airfoil coordinates and flight conditions as inputs. The flight conditions with dimensions $(1,2)$ are appended to the matrix of coordinates to form the inputs with dimensions $(280,2)$. ACPM consists of three convolution layers and two fully connected layers. The convolution kernels of the three convolution layers have dimensions $279 \times 2$ and $1 \times 558$. The input of the RBF-based discriminator is a combination of the expanded coordinate vector and flight condition vector. The structure of the RBF-based discriminator is set to “RBF: 512×1.” In addition, according to [1], the structure of the corresponding generator is set to “FCN: 1024×3.”

MDF is a complete FCN whose input is a combination of airfoil coordinates and flight conditions. MDF is the most intuitive fusion method that takes a combination of airfoil coordinates and flight conditions as inputs. The purpose of selecting MDF as a comparative method is to compare this intuitive fusion method with MDF. The structure of MDF is set to “FCN: 1024×3.”
Fig. 6. MSE and MAE distributions of \( C_D \) and \( C_L \) predicted by MTL\(_g\), MTL\(_c\), PGML, MLP, and MDF in tenfolds. (a) Distribution of test MSE of \( C_D \). (b) Distribution of test MAE of \( C_D \). (c) Distribution of test MSE of \( C_L \). (d) Distribution of test MAE of \( C_L \).

\[ \text{MSE} = \frac{1}{N} \sum_{z=1}^{N} \left( \sum_{i=1}^{K} \left( y_{zi} - \hat{y}_{zi} \right)^2 \right) \]
\[ \text{MAE} = \frac{1}{N} \sum_{z=1}^{N} \left( \sum_{i=1}^{K} |y_{zi} - \hat{y}_{zi}| \right) \]

where \( y_{zi} \) denotes the true coefficient of the \( z \)th input in the test set, and \( \hat{y}_{zi} \) denotes the predicted coefficient of the \( z \)th input.

2) Results and Analyses: In the field of aerodynamics, \( C_D \) of an aircraft is of a small order of magnitude than \( C_L \), which leads to difficulties in predicting \( C_D \) \([54,55,56,57]\). Hence, we give special attention to the prediction of \( C_D \) in this article.

Table \( \text{V} \) shows the average prediction errors of MTL\(_g\), MTL\(_c\), RBF-GAN, PGML, MLP, ACPM, and MDF in tenfolds and their average training time. In each column, the figures in bold are the minimum average test errors. In terms of \( C_D \), both the test MSE and MAE of MDF are smallest among those of all methods, while both the test MSE and MAE of RBF-GAN are the largest. In terms of \( C_L \), the test MSE and MAE of ACPM are the smallest, and both the test MSE and MAE of MDF are of the same order of magnitude as those of ACPM. By comparing MDF and ACPM, we see that

1) the average MSE of \( C_D \) predicted by MDF is reduced by 54.56%;
2) the average MAE of \( C_D \) predicted by MDF is reduced by 14.40%;
3) the average MSE of \( C_L \) predicted by MDF is increased by 15.87%; and
4) the average MAE of \( C_L \) predicted by MDF is increased by 6.98%.

In terms of average training time consumptions, RBF-GAN is the method with the longest average training time, and MTL\(_g\) is the method with the shortest average training time. But they did not achieve low test errors in predicting \( C_D \) and \( C_L \). In contrast, although ACPM and MDF require relatively long average training time, their test errors of both \( C_D \) and \( C_L \) are smaller than other methods. This table
indicates that the manifold-features extracted by manifold-based airfoil geometric-feature extraction module can be fused with flight conditions by MDF to reduce the prediction errors of $C_D$ while keeping the predicted $C_L$ on the same order of magnitude as $C_L$ predicted by ACPM.

Fig. 6 shows the distributions of the MSEs and MAEs of $C_D$ and $C_L$ predicted by MTL_g, MTL_c, PGML, MLP, ACPM, and MDF in ten folds. Because the prediction errors of RBF-GAN are too large, the distributions of the prediction errors of RBF-GAN are omitted from this figure. In each subgraph, the blue line at the top indicates the maximum value, the blue line in the middle represents the mean value, and the blue line at the bottom indicates the minimum value. In addition, the light blue shade indicates the error intensity. Subgraph (a) shows that the average MSE of $C_D$ predicted by MDF is the smallest among those of all methods. In subgraph (b), both the minimum and mean MAEs of $C_D$ predicted by MDF are smaller than those of $C_D$ predicted by the other methods. In subgraph (c), all the maximum, minimum, and mean MSEs of $C_L$ predicted by MTL_c, MLP, ACPM, and MDF are similar. In subgraph (d), the mean MAE of $C_L$ predicted by MDF is similar to those of $C_L$ predicted by MTL_c, MLP, and ACPM. In addition, the average MSE and MAE of $C_D$ predicted
by MDF are the smallest among those of all methods, and the average MSE and MAE of $C_L$ predicted by ACPM are the smallest, which is consistent with the statistic average error in Table V. These subgraphs indicate that MDF can reduce the prediction errors of $C_D$ while keeping the $C_L$ prediction errors similar to those of MTL_c, MLP, and ACPM.

Figs. 7 and 8 depict the training and validation losses of the above methods in two different folds. Subgraphs (a)–(f) in Fig. 7 show the loss variations of MTL_g, MTL_c, PGML, MLP, ACPM, and MDF, respectively, in the secondfold. Subgraphs (a)–(f) in Fig. 8 show the loss variations of MTL_g, MTL_c, PGML, MLP, ACPM, and MDF, respectively, in the ninthfold. The fluctuations of the validation loss of MTL_g, ACPM, and MDF are always smaller than those of MTL_c, PGML, and MLP during the entire training process. In the beginning of training process (epoch < 750), MTL_g and ACPM are more stable than MDF. After convergence (epoch ≥ 750), the fluctuations of both the training loss and the validation loss of MTL_g, ACPM, and MDF are similar. From the perspective of whole convergence process, MDF is more stable than MTL_c, PGML, and MLP. We deduce that MDF is stable for two reasons: 1) manifold-features of airfoils are latent features in Riemannian space that can be used to precisely reflect the geometric structure of airfoils; 2) MDF, a discrepant data fusion method, is able to fuse the manifold-features of airfoils in Riemannian space and the flight conditions in Euclidean space effectively.

Fig. 9 shows $C_D$ and $C_L$ predicted by MTL_c, MLP, ACPM, and MDF (four optimal methods in Table V). Subgraphs (a) and (b) show the predicted $C_D$ and $C_L$ in the secondfold, and subgraphs (c) and (d) show the predicted $C_D$ and $C_L$ in the ninthfold. In subgraphs (a) and (c), $C_D$ predicted by MDF are closely clustered around the diagonal. In contrast, $C_D$ predicted by MTL_c, MLP, and ACPM deviate from the diagonal significantly with $C_D$ > 0.03. Subgraphs (a) and (c) indicate that our proposed MDF has a distinct advantage for the prediction of $C_D$ with $C_D$ > 0.03 over other methods. In subgraphs (b) and (d), $C_L$ predicted by four methods are similar. The variations in the predicted $C_D$ and $C_L$ with $\alpha$ (i.e., AoA) are shown in Fig. 10. Because the prediction errors of RBF-GAN are the largest, $C_D$ and $C_L$ values predicted by RBF-GAN are not shown in this figure. Subgraphs (a) and (b) show the variations of $C_D$ and $C_L$ with $\alpha$ in the secondfold, and subgraphs (c) and (d) show the variations of $C_D$ and $C_L$ with $\alpha$ in the ninthfold. Subgraphs (a) and (c) show that $C_D$ values predicted by MDF (the red line) are closer to the real values (the black line) than those predicted by other methods. In subgraphs (b) and (d), $C_L$ values predicted by all methods are relatively similar and close to the real values.

In summary, the results of experiment I demonstrate that the manifold-features extracted by the manifold-based airfoil geometric-feature extraction module indeed reflect the geometric shape of airfoils and that the manifold-features can be used to generate airfoils based on their geometrical properties. In addition, the results of experiment II demonstrate that the manifold-features and flight conditions can be fused by MDF to reduce the prediction errors of $C_D$ while keeping the same prediction accuracy level for $C_L$. 
V. CONCLUSION

The conclusions of our work are as follows:

1) The geometric shape of an airfoil can be approximated by a set of self-intersection-free Bézier curves that are connected end to end to form a segmented smooth Riemannian manifold.

2) The Riemannian metric, as a sort of manifold-feature, can be used to rebuilt smooth and approximated airfoils, and the MSE of rebuilt airfoils is reduced by 41.30% compared with airfoils rebuilt using an Auto-Encoder.

3) The MSE of $C_D$ predicted by MDF is reduced by 54.56% compared with that of $C_D$ predicted by ACPM while keeping the same prediction accuracy level for $C_L$.

The results of the above experiments show that the prediction errors of $C_D$ can be significantly reduced by MDF, which fuses only one type of manifold-feature (i.e., the Riemannian metric) with flight conditions. In the future, more latent manifold-features (e.g., the curvature, torsion, and Riemannian connection) of the segmented smooth manifold built from airfoil curves will be defined and extracted. We firmly believe that 1) multiple manifold-features can further reflect the geometrical properties of airfoils and 2) the fusion of multiple manifold-features with flight conditions can produce more accurate predictions of airfoil coefficient. In addition, other datasets (e.g., Profili airfoil dataset [40] and NACA airfoil dataset [42]) will be used to further validate MDF.

ACKNOWLEDGMENT

The authors would like to thank Dr. W. Wang, Research Fellow, from the University of Electronic Science and Technology of China, for valuable suggestions regarding our project. In addition, the authors would also like to thank Dr. Y. Cheng, Associate Research Fellow, and Dr. W. Qian, Research Fellow, both from the China Aerodynamics Research and Development Center, for their valuable suggestions regarding this article.

REFERENCES

[1] L. Hu, W. Wang, Y. Xiang, and J. Zhang, "Flow field reconstructions with GANs based on radial basis functions," IEEE Trans. Aerosp. Electron. Syst., vol. 58, no. 4, pp. 3460–3476, Aug. 2022.
[2] J. Li, X. Du, and J. R. R. A. Martins, "Machine learning in aerodynamic shape optimization," Prog. Aerosp. Sci., vol. 134, 2022, Art. no. 100849.
[3] L. Hu, J. Zhang, Y. Xiang, and W. Wang, “Neural networks-based aerodynamic data modeling: A comprehensive review,” IEEE Access, vol. 8, pp. 90805–90823, 2020.
[4] M. Shelton, B. Gregory, S. Lamson, H. Moses, R. Doughty, and T. Kiss, “Optimization of a transonic turbine airfoil using artificial intelligence, CFD and cascade testing,” Turbo Expo., Power Land, Sea, Air, vol. 78903, 1993, Art. no. V03AT15A012.
[5] R. Derksen and T. Rogalsky, “Bezier-parsec: An optimized aerofoil parameterization for design,” Adv. Eng. Softw., vol. 41, no. 7–8, pp. 923–930, 2010.
[6] G. Farin, Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide. San Diego, CA, USA: Elsevier, 2014.
[7] F. Zhu and N. Qin, “Intuitive class/shape function parameterization for airfoils,” AIAA J., vol. 52, no. 1, pp. 17–25, 2014.
[8] Y. Zhang, B. Pang, X. Li, and G. Chen, “Aerodynamic shape optimization with Grassmannian shape parameterization method,” Energies, vol. 15, no. 20, 2022, Art. no. 7722.
[9] A. Adhikari and M. R. Adhikari, Basic Topology 2: Topological Groups. Topology of Manifolds and Lie Groups. Berlin, Germany: Springer Nature, 2022.

[10] E. Afnan's eva and A. Golberg, “Homeomorphisms of finite metric distortion between Riemannian manifolds,” Comput. Methods Funct. Theory, vol. 22, pp. 1–26, 2022.

[11] J. Nagawark, J. Ren, X. Du, L. Leifsson, and S. Koziel, “Single- and multipoint aerodynamic shape optimization using multifidelity models and manifold mapping,” J. Airvic, vol. 58, no. 3, pp. 591–608, 2021.

[12] J. Li, X. Du, and J. R. Martins, “Machine learning in aerodynamic shape optimization,” Prog. Aerosp. Sci., vol. 134, 2022, Art. no. 100849.

[13] X. Du, J. Ren, and L. Leifsson, “Aerodynamic inverse design using multifidelity models and manifold mapping,” Aerosp. Sci. Technol., vol. 85, pp. 371–385, 2019.

[14] V. Raul and L. Leifsson, “Multifidelity modeling similarity conditions for airfoil dynamic stall prediction with manifold mapping,” Eng. Comput., vol. 39, pp. 1180–1205, 2021.

[15] S. P. Dakua, “Performance divergence with data discrepancy,” in Proc. ICCKE, Chin. J. Aeronaut. Theor. Comput. Fluid Dyn., Proc. AIAA Scitech 2020 Forum, to be published, 2020.

[16] M. Deisenroth and J. W. Ng, “Distributed Gaussian processes;,” in Proc. Int. Conf. Mach. Learn., 2015, pp. 1481–1490.

[17] L. Zhang, S. Huang, W. Liu, and D. Tao, “Learning a mixture of granularity-specific experts for fine-grained categorization,” in Proc. IEEE Int. Conf. Comput. Vis., 2019, pp. 8331–8340.

[18] J. Zhang, Y. Thadimari, V. V. Velpula, and J. Bhargav, “Learning interactions and dynamics of swarms,” 2020. Accessed: May 26, 2023. [Online]. Available: https://www.researchgate.net/profile/Jayanth-Bhargav/project/Learning-Interactions-and-Dynamics-of-Swarms/attachment/5eb86e5f155db0001f4ad65/AS.899747358621698@1589143709820/download/CIS_522_Project.pdf

[19] J. Delso, “On gravitomagnetic tensor derived from energy momentum 1-form in a Riemann–Cartan space with a metric connection,” OSP J. Phys. Astron., vol. 3, pp. 126–129, 2022.

[20] S. A. A. A. Said Mad Zain et al., “Generalized fractional Bézier curve with shape parameters,” Mathematics, vol. 9, no. 17, 2021, Art. no. 2141.

[21] Z. Tang, “Metric,” Fundamentals of Riemannian Geometry in Chinese, 1st ed., C. Yue, Ed. Beijing, China: Beijing Normal Univ. Publishing House, 2011.

[22] W. Chen, “Riemannian metric,” in Introduction to Riemann Geometry in Chinese, 1st ed., J. Zhang, Ed. Beijing, China: Beijing Normal Univ. Publishing House, 2004, pp. 83–90.

[23] S. Sommer and A. M. Bronstein, “Horizontal flows and manifold stochastics in geometric deep learning,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 2, pp. 811–822, Feb. 2022.

[24] P. Hansen, B. Eltzner, and S. Sommer, “Diffusion means and heat kernel on manifolds,” in Proc. Int. Conf. Geometric Sci. Inf., 2021, pp. 111–118.

[25] S. Alexakis, T. Balehowsky, and A. Nachman, “Determining a Riemannian metric from minimal areas,” Adv. Math., vol. 366, 2020, Art. no. 107025.

[26] Z. Li and D. Wang, “Analysis and experimental study on different airfoils in heavy unmanned aerial vehicle,” Aerosp. Syst., vol. 4, no. 1, pp. 37–48, 2021.

[27] X. Du, P. He, and J. R. Martins, “A b-spline-based generative adversarial network model for fast interactive airfoil aerodynamic optimization,” in Proc. AIAA Scitech 2020 Forum, 2020, Art. no. 2128.

[28] K. Yonekura, K. Wada, and K. Suzuki, “Generating various airfoils with required lift coefficients by combining NACA and Joukowski airfoils using conditional variational autoencoders,” Eng. Appl. Artif. Intell., vol. 108, 2022, Art. no. 105460.

[29] K. Yonekura and K. Suzuki, “Data-driven design exploration method using conditional variational autoencoder for airfoil design,” Struct. Multidisciplinary Optim., vol. 64, no. 2, pp. 613–624, 2021.

[30] R. Li, Y. Zhang, and H. Chen, “Physically interpretable feature learning of supercritical airfoils based on variational autoencoders,” AIAA J., vol. 60, no. 11, pp. 6168–6182, 2022.

[31] M. Morimoto, K. Fukami, K. Zhang, A. G. Nair, and K. Fukagata, “Convolutional neural networks for fluid flow analysis: Toward effective metamodeling and low dimensionalization,” Theor. Comput. Fluid Dyn., vol. 35, no. 5, pp. 633–658, 2021.

[32] M. Yanin and Z. A. K. Ramadhan, “Neural network approach for predicting aerodynamic performance of naca airfoil at low Reynolds number,” Jurnal Polimesin, vol. 20, no. 2, pp. 190–193, 2022.

[33] R. Li, Y. Zhang, and H. Chen, “Transfer learning from two-dimensional supercritical airfoils to three-dimensional transonic swept wings,” Chin. J. Aeronaut., to be published, doi: 10.1016/j.cja.2023.04.008.

[34] Z. Wei, B. Guillard, P. Fua, and M. Bauerheim, “Latent representation of CFD meshes and application to 2D airfoil aerodynamics,” AIAA J., 2023.
Yu Xiang (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electronic engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, Sichuan, China, in 1995, 1998, and 2003, respectively. He joined the UESTC in 2003 and became an Associate Professor in 2006. From 2014 to 2015, he was a Visiting Scholar with the University of Melbourne, Australia. His current research interests include computer networks, intelligent transportation systems, and deep learning.

Liwei Hu (Student Member, IEEE) received the B.S. degree in software engineering from the School of Software, Hebei Normal University, Hebei, China, in 2014, as well as the M.S. degree in computer technology from the University of Electronic Science and Technology of China (UESTC), Chengdu, Sichuan, China, in 2018. He is currently working toward the Ph.D. degree in computer science and technology from the School of Computer Science and Engineering, UESTC.

His research fields include aerodynamic data modeling, deep learning and pattern recognition.

Guangbo Zhang received the B.S. degree in network engineering from the School of Computer Science, South-Central Minzu University, Wuhan, Hubei, China, in 2020. He is currently working toward the M.S. degree in computer science and technology with the School of Computer Science and Engineering, UESTC.

His research fields include aerodynamic data modeling and deep learning.

Jun Zhang received the B.S. and M.S. degrees in electronic engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 1995 and 1998, respectively.

From 1998 to 2008, he was a Senior Researcher and an Engineer in CERNET. He is currently a Lecturer at the School of Computer Science and Engineering, UESTC. His current research interests include software-defined networks, machine learning applied in network traffic engineering, and aerodynamics.

Wenyong Wang (Member, IEEE) received the B.S. degree in computer science from BeiHang University, Beijing, China, in 1988 and the M.S. and Ph.D. degrees in computer science from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 1991 and 2011, respectively.

He has been a Professor in computer science and engineering with UESTC since 2006. He is also a special-term Professor at Macau University of Science and Technology, a senior member of the Chinese Computer Federation, a member of the expert board of the China Education and Research Network (CERNET), and China next generation Internet. His main research interests include next generation Internet, software-defined networks, and software engineering.