Congealed Deep Neural Network-based System Identification for Morphing Aircraft

Hao-Chi Che
School of Automation Science and Electrical Engineering, Beihang University, Beijing, 100191, China

Huai-Ning Wu*
School of Automation Science and Electrical Engineering, Beihang University, Beijing, 100191, China
Hangzhou International Innovation Institute, Beihang University, Hangzhou, 311115, China

Email: chehaochi@126.com, whn@buaa.edu.cn
*Corresponding Author

Abstract

This paper proposes a new deep neural network (DNN) architecture called the congealed DNN for system identification of morphing aircraft. The developed DNN consists of two parts: the invariant features of the inner layers and the time-varying weights of the output layer. For the inner invariant features, a novel meta-learning with adversarial optimization framework is developed to derive a common representation function shared by different deformation conditions. For the time-varying weights, we consider them to be composed of congealed weights and time-varying perturbations. The congealed weights are estimated using standard adaptive techniques, while a sliding mode-like function is employed to attenuate time-varying disturbance terms. The experimental results indicate that the proposed method demonstrates more precise and faster adaptation capabilities to the morphing aircraft system compared to other methods.

Keywords: Congealed deep neural network, Meta-learning, Time-varying parameters, System identification

1. Introduction

Model-based approaches provide a formalized method with almost-global stability guarantees for aircraft controller design. However, morphing aircraft can adaptively change their structural shape, thereby dynamically adjusting their aerodynamic configuration. The deformation process can lead to changes in the inherent structural parameters of the aircraft [1], which increases the complexity of aircraft modelling. The resulting complexity and instability in dynamics can have a significantly adverse impact on the model-based controller design for morphing aircraft. Therefore, it is imperative in model-based controller design to accurately identify the unmodeled dynamics.

Neural networks (NNs) have garnered significant interest within the field of control systems, particularly in nonlinear control, due to their ability to approximate complex nonlinear functions [2]. Besides, some studies demonstrate that deep neural networks (DNNs), through more complex architectures, may achieve superior function approximation capabilities [3], [4], [5]. However, the model of morphing aircraft represents a more complex nonlinear spatiotemporal system. Due to the unavailability of time-dependent variables in morphing aircraft, the aforementioned NN may struggle to achieve ideal results when approximating spatiotemporal uncertainty. In recent years, the paradigm of combining meta-learning and adaptive control has shown promising prospects in estimating spatiotemporal uncertainty, addressing domain transfer challenges, and adapting to new environments in real-time [6]. Nevertheless, the above results have limited capability in dealing spatiotemporal systems with rapidly changing parameters.

Significant advancements have been achieved in adaptive estimation methodologies for spatiotemporal systems. The recent developments in this area can primarily be classified into two trends. One of these is based on the so-called robust adaptive laws or switching σ modifications [7], [8], while the other involves filtering transformations and projection operation [9]. When the unknown parameters continue to change, the aforementioned methods cannot guarantee zero error regulation. In [10], a technique known as the congelation of variables (CV) has been introduced and advanced. The CV method treats unknown time-varying parameters as nominally constant parameters subject to disturbances, where the deviation between the actual parameter values and the nominal value is regarded as a time-varying disturbance. Accordingly, the controller design is decomposed into two parts: a conventional adaptive method for nominally constant parameters, and a damping component that suppresses the time-varying disturbance. Additionally, this approach is compatible with a wide range of adaptive control frameworks based on parameter

estimation, as it preserves the original parameter update laws applicable to time-invariant parameters.

Inspired by the aforementioned CV method, we design a congealed DNN (ConDNN) architecture to identify the unmodeled dynamics in morphing aircraft. This method not only improves the estimation accuracy of classical adaptive methods but also expands their applicability in more complex and dynamic spatiotemporal environments. Thus, this study presents two key innovations:

(1) Unlike the methods that combine NN and adaptive estimation in [4], [5], [6], an estimator based on ConDNN is designed for time-varying linear parameters. This structure demonstrates better estimation accuracy under rapidly changing conditions of deformable structures.

(2) Compared to the results of DNN in [4], [5], [6], we further consider the impact of representation errors brought about by DNN. The developed learning framework enables more accurate model estimation.

2. Problem Statement

The morphing aircraft model can be represented as the following general affine nonlinear spatiotemporal system:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\xi}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u} \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ describes the system state, $\boldsymbol{u} \in \mathbb{R}^m$ is the control input, $\boldsymbol{f}: \mathbb{R}^n \times \mathbb{R}^s \to \mathbb{R}^n$ represents the unmodeled dynamics, $\boldsymbol{g}: \mathbb{R}^n \to \mathbb{R}^{n\times m}$ indicates the known control effectiveness matrix with $n \geq m$, $\boldsymbol{\xi} \in \mathbb{R}^s$ is a hidden state related to deformation. To simplify notation, function arguments (e.g., t, $\boldsymbol{x}$) may be omitted where there is no ambiguity.

For model-based controller design, it is essential to have prior knowledge of the system model, specifically dynamic $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\xi})$. When the system model remains unavailable, real-time model estimation via online identification becomes necessary. For estimation in real-time systems, we have two urgent issues to address. First, existing adaptive estimation methods based on DNN (such as [4], [5] and [6]) require the latent variables to change slowly. Second, the DNN architecture designed in the above results does not take into account the impact of representation errors.

3. ConDNN Design

The following content provides a detailed explanation for identifying spatiotemporal uncertainty and the derivation process of the ConDNN structure.

3.1. DNN structure

The unmodeled dynamics term $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\xi})$ can be approximated by the following DNN:

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\xi}) = \boldsymbol{W}_{\boldsymbol{f}}^T \boldsymbol{\varrho}(\boldsymbol{\phi}(\boldsymbol{x})) + \boldsymbol{\epsilon}_{\boldsymbol{a}}(\boldsymbol{x}, t) \tag{2}$$

where $\boldsymbol{W}_{\boldsymbol{f}} = [\boldsymbol{w}_1\ \boldsymbol{w}_2\ \cdots\ \boldsymbol{w}_n] \in \mathbb{R}^{h\times n}$ is an ideal output-layer weight, $\boldsymbol{\phi}(\boldsymbol{x}): \mathbb{R}^n \to \mathbb{R}^p$ describes the DNN's inner-layer features, $\boldsymbol{\epsilon}_{\boldsymbol{a}}: \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ denotes the function reconstruction error with $\bar{\epsilon}_{\boldsymbol{a}} = \sup_{t\geq 0}\|\boldsymbol{\epsilon}_{\boldsymbol{a}}(\boldsymbol{x}, t)\|$, $\boldsymbol{\varrho}: \mathbb{R}^p \to \mathbb{R}^h$ is the activation function vector of the output layer. Specifically, $\boldsymbol{\phi}(\boldsymbol{x})$ can be articulated as $\boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{W}_{\iota}\boldsymbol{\varrho}_{\iota}(\boldsymbol{W}_{\iota-1}\boldsymbol{\varrho}_{\iota-1}(\boldsymbol{W}_{\iota-2}\boldsymbol{\varrho}_{\iota-2}(\cdots \boldsymbol{x})))$, where ι indicates the total number of hidden layers, $\boldsymbol{W}_{\iota}$ and $\boldsymbol{\varrho}_{\iota}$ correspond to the ideal weights and activation functions at layer ι, respectively. For any analytic function $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\xi})$, it can be decomposed into a $\boldsymbol{\xi}$-invariant component $\boldsymbol{\Phi}(\boldsymbol{x})$ and a $\boldsymbol{\xi}$-dependent component $\boldsymbol{a}(\boldsymbol{\xi})$ with arbitrary accuracy [6]. Inspired by this, Eq. (2) can be rewritten as

$$\begin{aligned}\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\xi}) &= \boldsymbol{I}_n \otimes \boldsymbol{\varrho}\big(\boldsymbol{\phi}(\boldsymbol{x})\big)\mathrm{vec}\big(\boldsymbol{W}_{\boldsymbol{f}}\big) + \boldsymbol{\epsilon}_{\boldsymbol{a}}(\boldsymbol{x}, t)\\ &= \boldsymbol{\Phi}(\boldsymbol{x})\boldsymbol{a}(\boldsymbol{\xi}) + \boldsymbol{\epsilon}_{\boldsymbol{a}}(\boldsymbol{x}, t)\end{aligned}$$

where $\boldsymbol{\Phi}(\boldsymbol{x}) = \boldsymbol{I}_n \otimes \boldsymbol{\varrho}\big(\boldsymbol{\phi}(\boldsymbol{x})\big) \in \mathbb{R}^{n\times nh}$ is a representation function shared by all morphing conditions, $\boldsymbol{a}(\boldsymbol{\xi}) = \mathrm{vec}\big(\boldsymbol{W}_{\boldsymbol{f}}\big) \in \mathbb{R}^{nh}$ is the linear coefficient update for each morphing condition, $\mathrm{vec}\big(\boldsymbol{W}_{\boldsymbol{f}}\big)$ represents the vectorization of matrix $\boldsymbol{W}_{\boldsymbol{f}}$, $\otimes$ is the Kronecker product.

3.1. Output-layer weight congelation

In this section, we introduce a weight congelation scheme to handle time-varying weight $\boldsymbol{a}(t)$. Let the linear coefficient $\boldsymbol{a}(t)$ is denoted as $\boldsymbol{a}(t) \in \Omega_{\boldsymbol{a}}$, where $\Omega_{\boldsymbol{a}} \subseteq \mathbb{R}^{nh}$ denotes a compact set with an unknown radius $\delta_{\boldsymbol{a}}$. Also, define the constant congealed parameter $\boldsymbol{\ell}_{\boldsymbol{a}} \in \mathbb{R}^{nh}$ with respect to $\boldsymbol{a}(t)$, which can be interpreted as the mean value of $\boldsymbol{a}(t)$. The time-varying perturbation with respect to $\boldsymbol{\ell}_{\boldsymbol{a}}$ is defined as $\Delta_{\boldsymbol{a}}(t) = \boldsymbol{a}(t) - \boldsymbol{\ell}_{\boldsymbol{a}} \in \mathbb{R}^{nh}$. Therefore, we have $\delta_{\boldsymbol{a}} = \sup_{t\geq 0}||\Delta_{\boldsymbol{a}}(t)||$. The time-varying weight structure of the output layer is shown in Figure 1. The unknown dynamic $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\xi})$ in Eq. (1) can be approximated by the ConDNN, which is defined by Definition 1.

Definition 1. (Congealed Deep Neural Network) A ConDNN with the learned representation $\boldsymbol{\Phi}(\boldsymbol{x})$ and the congealed weight vector $\boldsymbol{\ell}_{\boldsymbol{a}}$ is capable of approximating the spatiotemporal uncertainty $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\xi})$ as

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\xi}) = \boldsymbol{\Phi}(\boldsymbol{x})\boldsymbol{\ell}_{\boldsymbol{a}} + \boldsymbol{\Phi}(\boldsymbol{x})\Delta_{\boldsymbol{a}}(t) + \boldsymbol{\epsilon}_{\boldsymbol{a}}(\boldsymbol{x}, t). \tag{3}$$

The overall structure of the ConDNN is shown in Figure 1.

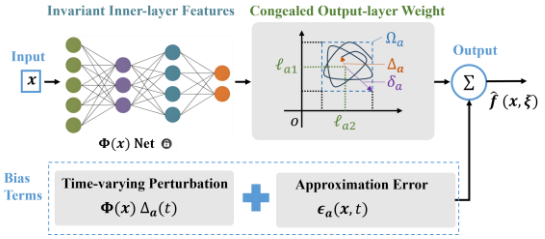

Figure 1 Description of the ConDNN architecture.

Drawing inspiration from the CV method [10], the constant congealed parameter $\boldsymbol{\ell}_{\boldsymbol{a}}$ can be approximated using the traditional adaptive estimation methods, resulting in an "estimate" represented as $\hat{\boldsymbol{a}}$. The time-varying perturbation term $\Delta_{\boldsymbol{a}}(t)$ and reconstruction error $\boldsymbol{\epsilon}_{\boldsymbol{a}}$ can be compensated for by inserting sliding mode-like terms.

4. Offline Meta-Learning

This part provides a concise overview of domain adversarial invariant meta-learning (DAIML) method [6] used for training $\boldsymbol{\Phi}(\boldsymbol{x})$. We consider K distinct deformation conditions, each characterized by unmodeled dynamics with varying parameters. Data can be obtained from each of these conditions. The dataset is given as $D = \{D_1, \cdots, D_K\}$, where each subset $D_k = \{\boldsymbol{x}_k(i), \boldsymbol{y}_k(i)\}_{i=1}^{N_k}$ comprises N_k pairs of input $\boldsymbol{x}$ and output data $\boldsymbol{y}$. A challenge in multi-task learning is the presence of domain shift, where $\boldsymbol{\Phi}(\boldsymbol{x})$ might memorize the distribution of $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\xi})$ under different deformation conditions for state $\boldsymbol{x}$ rather than $\boldsymbol{\xi}$. To address this issue, we present the subsequent adversarial optimization structure:

$$\max_{\boldsymbol{h}} \min_{\boldsymbol{\Phi}, \widehat{\boldsymbol{a}}_k} \sum_{k=1}^{K} \sum_{i=1}^{N_k} \{||\boldsymbol{y}_k(i) - \boldsymbol{\Phi}(\boldsymbol{x}_k(i), \boldsymbol{W})\widehat{\boldsymbol{a}}_k||^2 - \varpi \text{loss}(\boldsymbol{h}(\boldsymbol{\Phi}(\boldsymbol{x}_k(i))), k)\} \tag{4}$$

where $\boldsymbol{h} \in \mathbb{R}^K$ denotes an auxiliary DNN designed to act as a discriminator, predicting the deformation condition index from $\boldsymbol{\Phi}(\boldsymbol{x})$, $\varpi \in \mathbb{R}_{\geq 0}$ serves as a hyper-parameter to control the regularization strength. The DNN $\boldsymbol{h}$ outputs a K dimensional probability vector corresponding to K predefined $\boldsymbol{\xi}_k$, the term $\text{loss}(\cdot)$ represents the cross-entropy loss. The specific steps are shown in Algorithm 1, which performs following two main steps:

The *adaptation step* (lines 5-8) solves a least squares (LS) problem regarding $\widehat{\boldsymbol{a}}$ for each adaptive set $D_k^{\boldsymbol{a}}$. Specifically, we randomly select B data points from each sequence data D_k as the adaptive set $D_k^{\boldsymbol{a}}$, while the remaining $N_k - B$ data points $D_k^{\boldsymbol{W}}$ serve as the training set. On the adaptive set $D_k^{\boldsymbol{a}}$, we use LS to obtain $\boldsymbol{a}^*$ under specific deformation conditions.

The *training step* (lines 9-12) updates $\boldsymbol{\Phi}$ using the training dataset $D_k^{\boldsymbol{W}}$. This is achieved by solving the optimization problem presented in Eq. (4) that includes regularization. Additionally, spectral normalization is added to ensure the stability and convergence of NN training. The training procedure alternates between optimizing $\boldsymbol{h}$ and $\boldsymbol{\Phi}$ in two sequential phases: First, $\boldsymbol{\Phi}$ undergoes adjustment (line 9) while keeping $\boldsymbol{h}$ unchanged, subsequently $\boldsymbol{h}$ is refined (line 11) while $\boldsymbol{\Phi}$ remains fixed. A tuning parameter η $(0 < \eta \leq 1)$ governs the update frequency of the discriminator $\boldsymbol{h}$ within this alternating scheme.

Algorithm 1 The algorithm of DAIML

Input: $D = \{D_1, \cdots, D_K\}$, $\varpi \geq 0$, $0 < \eta \leq 1$, $\gamma > 0$
Output: $\boldsymbol{\Phi}$ and $\boldsymbol{h}$
1: Initialize $\boldsymbol{\Phi}$ and $\boldsymbol{h}$ with random values
2: Iteratively perform lines 3-12 until convergence
3: Sample dataset D_k from D randomly,
4: Select two disjoint batches $D_k^{\boldsymbol{a}}$ and $D_k^{\boldsymbol{W}}$ from D_k randomly
5: Solve the LS problem $\boldsymbol{a}^* = \arg\min_{\widehat{\boldsymbol{a}}} \sum_{i \in D_k^{\boldsymbol{a}}} (\|\boldsymbol{y}_k(i) - \boldsymbol{\Phi}(\boldsymbol{x}_k(i), \boldsymbol{W})\widehat{\boldsymbol{a}}\|^2)$
6: **if** $\|\boldsymbol{a}^*\| > \gamma$ **then**
7: $\boldsymbol{a}^* = \gamma \frac{\boldsymbol{a}^*}{\|\boldsymbol{a}^*\|}$
8: **end if**
9: Train DNN $\boldsymbol{\Phi}$ with loss $\sum_{i \in D_k^{\boldsymbol{W}}} \{||\boldsymbol{y}_k(i) - \boldsymbol{\Phi}(\boldsymbol{x}_k(i), \boldsymbol{W})\widehat{\boldsymbol{a}}_k||^2 - \varpi \text{loss}(\boldsymbol{h}(\boldsymbol{\Phi}(\boldsymbol{x}_k(i))), k)\}$ by stochastic gradient descent and spectral normalization
10: **if** $rand() < \eta$ **then**
11: Train DNN $\boldsymbol{h}$ with $\sum_{i \in D_k^{\boldsymbol{W}}} \text{loss}(\boldsymbol{h}(\boldsymbol{\Phi}(\boldsymbol{x}_k(i))), k)$
12: **end if**

5. Online Adaptation

5.1. Parameter update

To estimate the drift dynamics, this paper develops the following state estimator:

$$\dot{\widehat{\boldsymbol{x}}} = \widehat{\boldsymbol{f}}(\boldsymbol{x}, \widehat{\boldsymbol{a}}, \hat{\delta}_{\boldsymbol{a}}, \hat{\epsilon}_{\boldsymbol{a}}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u} + \boldsymbol{G}_{\boldsymbol{x}}\tilde{\boldsymbol{x}} \tag{5}$$

where $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \widehat{\boldsymbol{x}}$ is state estimation error, and $\boldsymbol{G}_{\boldsymbol{x}} \in \mathbb{R}^{n \times n}$ is a user-defined positive definite (PD) observer gain matrix, $\widehat{\boldsymbol{f}}(\boldsymbol{x}, \widehat{\boldsymbol{a}}, \hat{\delta}_{\boldsymbol{a}}, \hat{\epsilon}_{\boldsymbol{a}})$ is the ConDNN-based estimate of the unknown dynamics $\boldsymbol{f}$, which is defined as

$$\widehat{\boldsymbol{f}}(\boldsymbol{x}, \widehat{\boldsymbol{a}}, \hat{\delta}_{\boldsymbol{a}}, \hat{\epsilon}_{\boldsymbol{a}}) = \boldsymbol{\Phi}(\boldsymbol{x})\widehat{\boldsymbol{a}} + \boldsymbol{\Phi}(\boldsymbol{x})\text{sgm}(\boldsymbol{\Phi}^T(\boldsymbol{x})\tilde{\boldsymbol{x}}, \tau)\hat{\delta}_{\boldsymbol{a}} + \text{sgm}(\tilde{\boldsymbol{x}}, \tau)\hat{\epsilon}_{\boldsymbol{a}} \tag{6}$$

where $\widehat{\boldsymbol{a}} \in \mathbb{R}^{nh}$, $\hat{\delta}_{\boldsymbol{a}} \in \mathbb{R}$ and $\hat{\epsilon}_{\boldsymbol{a}} \in \mathbb{R}$ denote the estimates of the congealed parameter $\boldsymbol{\ell}_{\boldsymbol{a}}$, the radius of $\boldsymbol{a}$ (i.e., $\delta_{\boldsymbol{a}}$) and upper bound of $\boldsymbol{\epsilon}_{\boldsymbol{a}}$ (i.e., $\bar{\epsilon}_{\boldsymbol{a}}$), respectively. By utilizing (1), (3) and (6), we have

$$\dot{\tilde{\boldsymbol{x}}} = \boldsymbol{\Phi}(\boldsymbol{x})\tilde{\boldsymbol{a}} + \boldsymbol{\Phi}(\boldsymbol{x})\Delta_{\boldsymbol{a}}(t) + \boldsymbol{\epsilon}_{\boldsymbol{a}}(\boldsymbol{x}, t) - \boldsymbol{G}_{\boldsymbol{x}}\tilde{\boldsymbol{x}} - \boldsymbol{\Phi}(\boldsymbol{x})\text{sgm}(\boldsymbol{\Phi}^T(\boldsymbol{x})\tilde{\boldsymbol{x}}, \tau)\hat{\delta}_{\boldsymbol{a}} - \text{sgm}(\tilde{\boldsymbol{x}}, \tau)\hat{\epsilon}_{\boldsymbol{a}} \tag{7}$$

where $\tilde{\boldsymbol{a}} = \boldsymbol{\ell}_{\boldsymbol{a}} - \widehat{\boldsymbol{a}}$ is defined as the error between the congealed weight vector and the weight estimates. The weight estimate adaptation law is designed as

$$\dot{\widehat{\boldsymbol{a}}} = \boldsymbol{\Pi}_{\boldsymbol{a}}\boldsymbol{\Phi}^T(\boldsymbol{x})\tilde{\boldsymbol{x}} \tag{8a}$$

$$\dot{\hat{\delta}}_{\boldsymbol{a}} = \gamma_\delta \tilde{\boldsymbol{x}}^T \boldsymbol{\Phi}(\boldsymbol{x})\text{sgm}(\boldsymbol{\Phi}^T(\boldsymbol{x})\tilde{\boldsymbol{x}}, \tau) \tag{8b}$$

$$\dot{\hat{\epsilon}}_{\boldsymbol{a}} = \gamma_\epsilon \tilde{\boldsymbol{x}}^T \text{sgm}(\tilde{\boldsymbol{x}}, \tau) \tag{8c}$$

$\boldsymbol{\Pi}_{\boldsymbol{a}} \in \mathbb{R}^{nh \times nh}$ is a constant PD adaptation gain matrix, $\gamma_\delta \in \mathbb{R}_{>0}$ and $\gamma_\epsilon \in \mathbb{R}_{>0}$ are constant adaptation gains.

Remark 1. The function $\text{sgm}(\cdot, \tau)$ denotes a class of sigmoid functions, parameterized by a constant $\tau \in \mathbb{R}_{\geq 0}$. For the state $\boldsymbol{x} \in \mathbb{R}^n$, $\text{sgm}(\cdot, \tau)$ satisfies the inequality:

$$\|\boldsymbol{x}\| - \boldsymbol{x}^T \text{sgm}(\boldsymbol{x}, \tau) \leq \rho(\tau) \tag{9}$$

where $\rho(\cdot)$ is a class $\mathcal{K}$ function. As τ approaches zero, $\text{sgm}(\boldsymbol{x}, \tau)$ approaches the sign function $\text{sgn}(\cdot)$. For example, we can choose $\text{sgm}(\boldsymbol{x}, \tau) = \frac{\boldsymbol{x}}{\sqrt{\|\boldsymbol{x}\|^{2+\tau^2}}}$.

5.2. Convergence analysis

Theorem 1. Given the uncertain spatiotemporal system (1), the state estimator (5) and the parameter update laws (8), the state estimation error $\tilde{\boldsymbol{x}}$, for any $t_2 > t_1 \geq 0$ satisfies:

$$\frac{1}{t_2-t_1}\int_{t_1}^{t_2}\|\tilde{\boldsymbol{x}}\|^2 dt \le \frac{p_0}{t_2-t_1}+p_1 \tag{11}$$

where p_0 and p_1 are positive constants given in subsequent proof.

Proof. The Lyapunov function candidate (LFC) V_0 is defined as

$$V_0(\boldsymbol{z}) = \frac{1}{2}\tilde{\boldsymbol{x}}^T\tilde{\boldsymbol{x}} + \frac{1}{2}\tilde{\boldsymbol{a}}^T(\boldsymbol{\Pi_a})\tilde{\boldsymbol{a}} + \frac{1}{2}\gamma_\delta^{-1}\tilde{\delta}_a^2 + \frac{1}{2}\gamma_\epsilon^{-1}\tilde{\epsilon}_a^2$$

where $\boldsymbol{z} = [\tilde{\boldsymbol{x}}^T, \tilde{\boldsymbol{a}}^T, \tilde{\delta}_a, \tilde{\epsilon}_a]^T$, $\tilde{\delta}_a = \delta_a - \hat{\delta}_a$, $\tilde{\epsilon}_a = \bar{\epsilon}_a - \hat{\epsilon}_a$. The time derivative of LFC is articulated as follows:

$$\dot{V}_0 = \tilde{\boldsymbol{x}}^T\dot{\tilde{\boldsymbol{x}}} - \tilde{\boldsymbol{a}}^T(\boldsymbol{\Pi_a})\dot{\hat{\boldsymbol{a}}} - \tilde{\delta}_a\gamma_\delta^{-1}\dot{\hat{\delta}}_a - \tilde{\epsilon}_a\gamma_\epsilon^{-1}\dot{\hat{\epsilon}}_a \tag{12}$$

Substituting (7) and (8a) into (12), $\dot{V}_0$ in (12) can be bounded as

$$\begin{aligned}\dot{V}_0 \le\ & \tilde{\boldsymbol{x}}^T(\boldsymbol{\Phi}(\boldsymbol{x})\tilde{\boldsymbol{a}} - \boldsymbol{G_x}\tilde{\boldsymbol{x}} - \mathrm{sgm}(\tilde{\boldsymbol{x}},\tau)\hat{\epsilon}_a \\ & -\boldsymbol{\Phi}(\boldsymbol{x})\mathrm{sgm}(\boldsymbol{\Phi}^T(\boldsymbol{x})\tilde{\boldsymbol{x}},\tau)\hat{\delta}_a) + \|\boldsymbol{\Phi}^T(\boldsymbol{x})\tilde{\boldsymbol{x}}\|\delta_a \\ & +\|\tilde{\boldsymbol{x}}\|\bar{\epsilon}_a - \tilde{\boldsymbol{a}}^T\boldsymbol{\Phi}^T(\boldsymbol{x})\tilde{\boldsymbol{x}} - \tilde{\delta}_a\gamma_\delta^{-1}\dot{\hat{\delta}}_a - \tilde{\epsilon}_a\gamma_\epsilon^{-1}\dot{\hat{\epsilon}}_a.\end{aligned}$$

With the help of inequality (9), one can obtain

$$\begin{aligned}\dot{V}_0 \le\ & -\tilde{\boldsymbol{x}}^T\boldsymbol{G_x}\tilde{\boldsymbol{x}} + \tilde{\boldsymbol{x}}^T\mathrm{sgm}(\tilde{\boldsymbol{x}},\tau)\bar{\epsilon}_a \\ & +\tilde{\boldsymbol{x}}^T\boldsymbol{\Phi}(\boldsymbol{x})\mathrm{sgm}(\boldsymbol{\Phi}^T(\boldsymbol{x})\tilde{\boldsymbol{x}},\tau)\tilde{\delta}_a - \tilde{\delta}_a\gamma_\delta^{-1}\dot{\hat{\delta}}_a \\ & -\tilde{\epsilon}_a\gamma_\epsilon^{-1}\dot{\hat{\epsilon}}_a + (\delta_a+\bar{\epsilon}_a)\rho(\tau).\end{aligned}$$

Exploiting (8b) and (8c), we finally obtain that

$$\dot{V}_0 \le -\lambda_{\boldsymbol{G}}||\tilde{\boldsymbol{x}}||^2 + l \tag{13}$$

where $\lambda_{\boldsymbol{G}}$ is the minimum eigenvalue of $\boldsymbol{G_x}$, $l = (\delta_a + \bar{\epsilon}_a)\rho(\tau)$. Furthermore, the time derivative of LFC satisfies $\dot{V}_0 \le 0$, when $\|\tilde{\boldsymbol{x}}\| \ge \sqrt{\frac{l}{\lambda_{\boldsymbol{G}}}} > 0$, which indicate that $\tilde{\boldsymbol{x}}$ is uniformly bounded $\forall t \in \mathbb{R}_{\ge 0}$. Solving the differential inequality (13) on $[t_1, t_2)$, for some $t_2 > t_1 \ge 0$, we have $\lambda_{\boldsymbol{G}}\int_{t_1}^{t_2}\|\tilde{\boldsymbol{x}}\|^2 dt \le V_0(t)|_{t_1}^{t_2} + (t_2-t_1)l$. Thus, we can obtain the result in (11), where $p_0 = \lambda_{\boldsymbol{G}}^{-1}\sup_{t\ge 0}V_0(t)$ and $p_1 = \lambda_{\boldsymbol{G}}^{-1}l$. Given that p_1 is adjustable, we can make $\tilde{\boldsymbol{x}} \in \mathcal{L}_2$ by adjusting $\tau \in \mathcal{L}_2$. Consequently, the proposed estimator ensures the asymptotic convergence property [11]. ■

6. Simulation

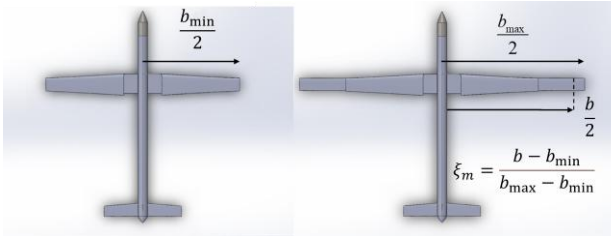

Figure 2. Variable span morphing aircraft.

The morphing aircraft model in [12] is adopted in this paper (Figure 2). Its longitudinal nonlinear dynamic model can be simplified into an affine nonlinear model $\dot{\boldsymbol{x}}_m = \boldsymbol{f}_m(\boldsymbol{x}_m, \xi_m) + \boldsymbol{g}_m(\boldsymbol{x}_m)\boldsymbol{u}_m$, where $\boldsymbol{x}_m = [V\ \alpha\ \theta\ q\ h]^T \in \mathbb{R}^5$ denotes the state, and $\boldsymbol{u}_m = [\delta_e\ \delta_t]^T \in \mathbb{R}^2$ is the control input, V, α, θ, q and h are flight velocity, attack angle, pitch angle, pitch angular velocity and flight altitude, respectively, δ_t is the throttle opening and δ_e is elevator deflection angle, ξ_m is the morphing ratio (Figure 2). Considering the equilibrium point $(\boldsymbol{x}_e, \boldsymbol{u}_e, \xi_e)$ satisfy $\boldsymbol{f}_m(\boldsymbol{x}_e, \xi_e) + \boldsymbol{g}_m(\boldsymbol{x}_e) = \boldsymbol{0}$, we can construct the system as shown in (1), where $\boldsymbol{x} = \boldsymbol{x}_m - \boldsymbol{x}_e, \boldsymbol{u} = \boldsymbol{u}_m - \boldsymbol{u}_e, \xi = \xi_m - \xi_e$. For the detailed analysis process and simulation parameters, please refer to reference [13].

The simulated control process is as follows: the initial altitude of the morphing aircraft is 4980 m, the speed is 35 m/s, and both the initial angle attack and the initial pitch angle are 0.01 rad, the pitch angular velocity is 0 rad/s, i.e., $\boldsymbol{x}_m(0) = [35\ 0.01\ 0.01\ 0\ 4980]^T$. At this moment, the wingspan deformation ratio is 0. At the 20th second, a command for wingspan length deformation is applied, requiring that the aircraft's wingspan length to undergo rapid periodic changes over 80 seconds (Figure 3). Assuming $\xi_e = 1$, the equilibrium points can be determined a priori by LS fitting in [12], namely $\boldsymbol{x}_e = [40\ 0.0453\ 0.0453\ 0\ 5000]^T$ and $\boldsymbol{u}_e = [-0.3839\ 27.9397]^T$. In this paper, the dynamic related to deformation rate of the wingspan is unknown. It is estimated through the design of associated ConDNN.

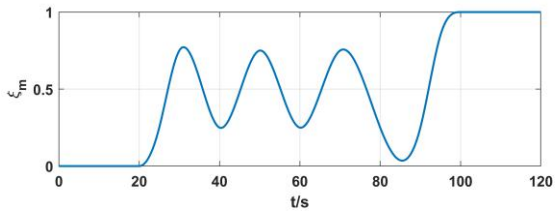

Figure 3. Commands for the wingspan deformation.

6.1. *Data collection*

MATLAB is used for numerical simulation of the flight process of a morphing aircraft to collect training data. We allow an aircraft to fly for 5 minutes along random trajectories under six different morphing conditions (Table 1), with the wingspan morphing ratio ranging from 0 to 1. Data are sampled at 100 Hz, generating a total of 30 000 data points. We collect data $\boldsymbol{x}$ and $\boldsymbol{u}$ along each trajectory, and use numerical differentiation to compute $\boldsymbol{x}$. Combining $\boldsymbol{g}(\boldsymbol{x})$ with (1), the measurement data $\boldsymbol{f}$ can be obtained.

Table 1 Offline data generation sampling points.

ξ_m	0.0	0.2	0.4	0.6	0.8	1.0
b/m	10.18	12.22	14.25	16.29	18.32	20.39

6.2. *DNN architecture and training details*

We take the state $\boldsymbol{x}$ as the input to the DNN, and its output is the unmodeled dynamics $\boldsymbol{f}$. The DNN in (6) is structured as $5 \to 32 \to 32 \to 5 \to 5$, where the final layer activation function employs a linear transfer function. The DNN's inner layer features $\boldsymbol{\Phi}$ follow the architecture

$5 \to 32 \to 32 \to 5$ and ReLU activation. The DNN $\boldsymbol{h}$ follow the architecture $5 \to 64 \to 32 \to 6$ and ReLU activation. The selected hyper-parameters include a discriminator training frequency $\eta = 0.5$, a normalization constant $\gamma = 10$, and a regularization strength $\varpi = 0.1$. We follow Algorithm 1 to train the common representation function $\boldsymbol{\Phi}$. In online adaption, the learning gains are selected as $\boldsymbol{G_x} = 10 \times \boldsymbol{I}_{5\times5}$, $\gamma_\delta = 0.11$, $\gamma_\epsilon = 0.5$ and $\boldsymbol{\Pi_a} = 20 \times \boldsymbol{I}_{25\times25}$. The initial values of parameters are selected as $\hat{\delta}_{\boldsymbol{a}}(0) = 0.1$, $\hat{\epsilon}_{\boldsymbol{a}}(0) = 0.1$ and $\hat{\boldsymbol{a}}(0) = 0.1 \times \boldsymbol{1}_{25}$, respectively, where $\boldsymbol{1}$ is unit vector. The constant of sigmoid function is selected as $\tau = 0.9$. The unknown dynamic estimation by Eq. (6), with their parameter updates shown in Eq. (8)

6.3. *Result and comparison with other method*

The "Baseline" refers to the classical DNN-based adaptive approximation technique employed in [4]. Its common representation function is also trained offline using DAIML in Section 4. Figure 4 shows a comparison of the errors between the measured values and the estimated values, indicating that under rapidly changing deformation parameters, our scheme demonstrates smaller estimation errors and faster convergence speed.

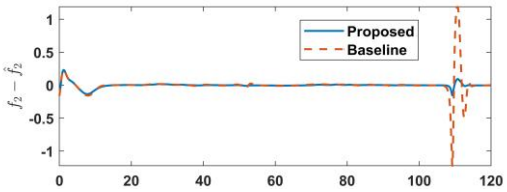

Figure 4. Comparison of estimation errors.

7. Conclusion

This paper proposes a novel ConDNN to approximate spatiotemporal uncertainty of morphing aircraft. Using the proposed DNN architecture, we estimate the unmodeled dynamics of the morphing aircraft. Simulation results confirm the effectiveness of the proposed approach. Compared to classical methods, it demonstrates superior performance under rapid parameter variations.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grants 92271115 and 62473021, in part by the Research Startup Funds of Hangzhou International Innovation Institute of Beihang University under Grant 2024KQ059.

References

1. S. Barbarino, O. Bilgen, R. M. Ajaj, et al., A review of morphing aircraft, Journal Intel. Mat. Syst. Str., vol. 22, no. 9, pp. 823–877, 2011.
2. F. Zhang, Y.-Y. Chen and Y. Zhang, Neural network boundary approximation for uncertain nonlinear spatiotemporal systems and its application of tracking control, IEEE T. Neur. Net. Lear., vol. 35, no. 5, pp. 7238-7243, 2024.
3. Y. LeCun, Y. Bengio and G. Hinton, Deep learning, Nature, vol. 521, no. 7553, pp. 436–444, 2015.
4. M. L. Greene, Z. I. Bell, S. Nivison et al., Deep neural network-based approximate optimal tracking for unknown nonlinear systems. IEEE T. Automat. Contr., vol. 68 no. 5, pp. 3171–3177, 2023.
5. R. Sun, M. L. Greene, D. M. Le et al., Lyapunov-based real-time and iterative adjustment of deep neural networks. IEEE Control Systems Letters, vol. 6, pp. 193-198, 2021.
6. M. O'Connell, G. Shi, X. Shi, et al. Neural-fly enables rapid learning for agile flight in strong winds. Sci. Robot., vol. 7, no. 66, p. eabm6597, 2022.
7. P. A. Ioannou and J. Sun. Robust Adaptive Control, volume 1. PTR Prentice-Hall Upper Saddle River, NJ, 1996.
8. K. Tsakalis and P. Ioannou. Adaptive control of linear time-varying plants. Automatica, vol. 23, no. 4, pp. 459–468, 1987.
9. M. Krstic, I. Kanellakopoulos, and P. Kokotović, Nonlinear and Adaptive Control Design. New York, NY, USA: Wiley, 1995.
10. K. Chen and A. Astolfi. Adaptive control of systems with time-varying parameters, IEEE T. Automat. Contr., vol. 66, no. 5, pp. 1986–2001, 2021.
11. Z. Zhang, Y. Dong, and G. Duan, Global asymptotic fault-tolerant tracking for time-varying nonlinear complex systems with prescribed performance, Automatica, vol. 159, p. 111345, 2024.
12. M. Yin, Coordinated control of deformation and flight for morphing aircraft, Nanjing University of Aeronautics and Astronautics, 2015.
13. H.-C. Che and H.-N. Wu, Coordinated control of deformation and flight for morphing aircraft via meta-learning and coupled state-dependent Riccati equations, IEEE T. Aero. Elec. Sys., Early access, doi:10.1109/taes.2025.3598801, 2025.

Authors Introduction

Mr. Hao-Chi Che

He received his M.S. degree in control theory and control engineering from the School of Mechanical and Electrical Engineering, Soochow University, Suzhou, China, in 2021. He is currently pursuing the Ph.D. degree in Beihang University, Beijing, China.

Dr. Huai-Ning Wu

He received the B.E. degree in automation from Shandong Institute of Building Materials Industry, Jinan, China, and the Ph.D. degree in control theory and control engineering from Xi'an Jiaotong University, Xi'an, China, in 1992 and 1997, respectively.

From August 1997 to July 1999, he was a Post-Doctoral Research Fellow with the Department of Electronic Engineering, Beijing Institute of Tech nology, Beijing, China. Since August 1999, he has been with the School of Automation Science and Electrical Engineering, Beihang University (formerly Beijing University of Aeronautics and Astronautics), Beijing, China. From December 2005 to May 2006, he was a Senior Research Associate with the City University of Hong Kong (CityU), Hong Kong. From October 2006 to December 2008 and from July 2010 to August 2013, he was a Research Fellow with CityU. He is currently a Professor with Beihang University. His current research interests include robust and fault-tolerant control, intelligent control, distributed parameter systems, and human-in-the loop systems.

Dr. Wu serves as an Associate Editor for the IEEE Transactions on Fuzzy Systems.