

Machine learning enhanced real-time aerodynamic forces prediction based on sparse pressure sensor inputs

Junming Duan*

École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland

Qian Wang†

Beijing Computational Science Research Center, 100193 Beijing, China

Jan S. Hesthaven‡

École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland

Accurate prediction of aerodynamic forces in real-time is crucial for autonomous navigation of unmanned aerial vehicles (UAVs). This paper presents a data-driven aerodynamic force prediction model based on a small number of pressure sensors located on the surface of UAV. The model is built on a linear term that can make a reasonably accurate prediction and a nonlinear correction for accuracy improvement. The linear term is based on a reduced basis reconstruction of the surface pressure distribution, where the basis is extracted from numerical simulation data and the basis coefficients are determined by solving linear pressure reconstruction equations at a set of sensor locations. Sensor placement is optimized using the discrete empirical interpolation method (DEIM). Aerodynamic forces are computed by integrating the reconstructed surface pressure distribution. The nonlinear term is an artificial neural network (NN) that is trained to bridge the gap between the ground truth and the DEIM prediction, especially in the scenario where the DEIM model is constructed from simulation data with limited fidelity. A large network is not necessary for accurate correction as the linear model already captures the main dynamics of the surface pressure field, thus yielding an efficient DEIM+NN aerodynamic force prediction model. The model is tested on numerical and experimental dynamic stall data of a 2D NACA0015 airfoil, and numerical simulation data of dynamic stall of a 3D drone. Numerical results demonstrate that the machine learning enhanced model can make fast and accurate predictions of aerodynamic forces using only a few pressure sensors, even for the NACA0015 case in which the simulations do not agree well with the wind tunnel experiments. Furthermore, the model is robust to noise.

*Postdoctoral researcher, Chair of Computational Mathematics and Simulation Science, Bâtiment MA, Station 8, 1015 Lausanne

†Corresponding author. Assistant Professor, Mechanics Division, No. 10 East Xibeiwang Road, Beijing 100193, qian.wang@csrc.ac.cn

‡Professor, Chair of Computational Mathematics and Simulation Science, Bâtiment MA, Station 8, 1015 Lausanne

Nomenclature

A	= amplitude of pitching movement, deg
A_{ref}	= reference area, m^2
\mathbf{b}	= coefficient vector of reduced basis
C_d	= drag coefficient $\frac{\text{drag force}}{\frac{1}{2}\rho_{\infty}V_{\infty}^2A_{\text{ref}}}$
C_l	= lift coefficient $\frac{\text{lift force}}{\frac{1}{2}\rho_{\infty}V_{\infty}^2A_{\text{ref}}}$
C_p	= pressure coefficient $\frac{p-p_{\infty}}{\frac{1}{2}\rho_{\infty}V_{\infty}^2}$
\mathbf{C}_p	= vector of pressure coefficients
$\overline{\mathbf{C}}_p$	= reference vector value of pressure coefficients
f	= pitching frequency, Hz
\mathbf{F}	= force on aircraft in body frame, N
\mathcal{I}	= indices corresponding to the selected sensor locations
$\tilde{\mathcal{I}}$	= candidate indices for DEIM selection
$L(\cdot; \Theta)$	= fully-connected layer
m	= number of candidate sensor locations
M	= number of snapshots
$\mathbf{M}_F^s, \mathbf{M}_F^0$	= pre-computed matrices for DEIM prediction
\mathbf{n}	= normal unit vector pointing into the aircraft in body frame
N	= number of available pressure coefficients on aircraft
n_b	= number of reduced bases
\mathbf{n}_d	= unit vector in drag direction
\mathbf{n}_l	= unit vector in lift direction
n_L	= number of layers in neural network
n_s	= number of selected sensors
p	= pressure, N
p_{∞}	= reference pressure, 1×10^5 N
\mathbf{R}	= reconstruction matrix in DEIM
Re	= Reynolds number
S	= surface area, m^2
\mathbb{S}	= snapshot matrix of pressure coefficients
$\overline{\mathbb{S}}$	= reference matrix of snapshot matrix
t	= time, s

U = reduced basis matrix

V_∞ = freestream velocity, m/s

Greek symbols

α = angle of attack, deg

α_0 = initial angle of attack, deg

$\alpha(\epsilon_{d,\infty}^z)$ = angle of attack corresponding to $\epsilon_{d,\infty}^z$, deg, z = DEIM, NN

$\alpha(\epsilon_{l,\infty}^z)$ = angle of attack corresponding to $\epsilon_{l,\infty}^z$, deg, z = DEIM, NN

ϵ_d^z = ℓ^2 error in drag coefficient, z = DEIM, NN

$\epsilon_{d,\infty}^z$ = ℓ^∞ error in drag coefficient, z = DEIM, NN

ϵ_l^z = ℓ^2 error in lift coefficient, z = DEIM, NN

$\epsilon_{l,\infty}^z$ = ℓ^∞ error in lift coefficient, z = DEIM, NN

ϵ_{proj} = projection error on the reduced space

ν_∞ = freestream kinematic viscosity, m^2/s

φ = activation function

ρ_∞ = freestream density, kg/m^3

σ = singular value

Θ = weights and biases in neural network

Superscripts

DEIM = signifies DEIM

exper = signifies experiment

NN = signifies using neural network

rb = signifies reduced basis approximation

s = signifies sensor

URANS = signifies URANS

Subscripts

i = index of pressure coefficient locations

j = index of layers in neural network

k = index of singular values

ℓ = index of DEIM interpolation points

testing = testing set

training = training set

I. Introduction

AUTONOMOUS navigation of unmanned aerial vehicles (UAVs) is of great importance in real-world applications. Global navigation satellite system (GNSS) aided inertial navigation system (INS) is the most popular navigation system for small UAVs [1]. The INS contains inertial measurement units (IMUs) which are used to measure linear acceleration and angular rates of rotation, based on accelerometers and gyroscopes. From this, position, velocity and attitude can be obtained by integration, but accuracy is limited due to an accumulation of errors in long-time integration, known as drift. Filters in the INS provide a way to fuse the observations from the GNSS, such as time, position, and velocity, to control the drift.

In scenarios when GNSS outages occur, e.g., UAVs passing through tunnels, high buildings, or forests, the drift cannot be controlled and navigation errors increase fast [1, 2], leading to situations where UAVs may even cause danger to objects on the ground during long GNSS outages. There is some past work that attempts to improve navigation performance during GNSS outages. Advanced techniques were used to improve INS error modeling [3–5], but the improvements were limited. Additional sensors were also adopted to aid the navigation system [6–8], adding extra costs and complexity to the system, as well as platform dependence. A vehicle dynamic model (VDM) is a mathematical model describing the dynamics of the platform, constructed based on the physical laws of motion. It does not rely on extra sensors, while takes available quantities such as navigation states, UAV parameters, wind velocity, and control commands as input to compute the aerodynamics loads and then predict navigation states at the next time step that can be further combined with other observations by using filters. VDMs have been integrated into navigation systems [9–14] to improve navigation performance. Compared to the conventional INS-GNSS system, the VDM-INS-GNSS in [15] was shown to give better navigation performance during GNSS outage, where the aerodynamic loads are computed using linear and quadratic polynomials of which the coefficients need to be calibrated for each specific UAV.

When the dynamics become highly nonlinear, e.g., UAVs at high angles of attack or in turbulent flows, it is very difficult for the conventional VDM to work effectively, as the aerodynamic models may be too simple to capture nonlinear dynamics. One growing interest is to use distributed airflow sensors to aid the navigation and control system [16–20], inspired by flying animals, where many different sensors are distributed on the bodies and aerodynamic surfaces. Artificial neural networks (NNs) were used in [16, 18, 19, 21] to learn the dynamics based on sensor inputs. The model in [21] was shown to work also in the dynamic stall region, where the dynamics are highly nonlinear and difficult to capture. In the study of the navigation system based on distributed sensors, it is often not practical or feasible to put too many sensors on the UAVs, and a large NN is necessary if the sensor locations are not well selected. Therefore, the locations of the sensors must be carefully optimized. The sensors are usually placed near the leading edge [22] as the pressure gradient is the highest in this part. For more discussions on sensor arrangements, the reader is referred to the review article [20] and references therein.

Aerodynamic forces can be also computed in real-time by integrating a fast surface flow field reconstruction obtained

through reduced-order modeling and sensor measurements [23, 24]. For a high Reynolds number flow around an aircraft, the contribution of viscous stress to lift and drag forces can be neglected as compared to pressure. Therefore, a surface pressure reconstruction is sufficient for accurate aerodynamic force estimation. The surface pressure can be approximated by a linear combination of reduced basis functions that are extracted from a collection of snapshots via principle component analysis methods such as proper orthogonal decomposition (POD) [25]. The basis coefficients are determined by solving linear pressure reconstruction equations at a set of sensor locations. Sensor placement can be optimized by algorithms such as gappy POD [26], discrete empirical interpolation method (DEIM) [27], generalized empirical interpolation method (GEIM) [28], and particle swarm optimization (PSO) [24]. High resolution in space of the snapshots or basis functions is required for accurate computation of aerodynamic forces as it is based on the integration of the reduced basis approximation. However, it is often very difficult and expensive to put a large number of sensors on the surface of an aircraft. Therefore, snapshots are collected from computational fluid dynamics (CFD) simulations. For vortex-dominated flow phenomena such as dynamic stall, expensive high-fidelity simulations are needed to capture small-scale flow structures and offer good agreement with experiments. Towards an efficient VDM for UAVs, it is necessary to develop algorithms to accurately predict the aerodynamic forces using an inaccurate basis and sparse sensor measurements.

This paper proposes a systematic approach to construct a GNSS-free surrogate aerodynamic model to predict the aerodynamic coefficients based on sparse pressure sensor inputs, employing data fusion from numerical simulations and experiments. The number of sensors is limited, especially in 3D, so we generally cannot use experimental data to select the sensor locations and compute the aerodynamic coefficients by integration on the surface. Usually, there are plenty of numerical simulation data for different aerodynamic states, called snapshots, available on the whole surface of the UAVs, thus the sensor locations can be selected by discrete empirical interpolation method (DEIM) [27] using the pressure coefficients of the numerical snapshots. The real-time pressure sensor values can then be collected at the selected locations, and the surface flow fields and aerodynamic coefficients can be obtained from the DEIM model based on such given sensor inputs. As the DEIM model uses a linear approximation, it is not accurate enough when the number of sensors is small or the dynamics are highly nonlinear. Therefore, an artificial NN is employed to correct the aerodynamic coefficients from the DEIM model with the help of the experimental data. To be specific, the NN learns the map from the pressure sensor inputs to the error in the aerodynamic force prediction, and the size of the NN is small so the training and online prediction can be very efficient. In other words, the linear model obtained by the DEIM provides a rough approximation of the aerodynamic coefficients, and the NN models the unresolved parts, which is calibrated by the experimental data, leading to an accurate and efficient model. The proposed approach is platform-independent and GNSS-free, without additional costs and weights on the UAVs. The test results show that the DEIM+NN model is capable of making fast and accurate aerodynamic force predictions, and not sensitive to noise in the pressure sensor inputs.

This paper is organized as follows. Section II presents the surface pressure reconstruction and sensor placement optimization based on the DEIM. Section III describes the proposed machine learning-enhanced approach for the prediction of the aerodynamic coefficients. The approach is applied to the dynamic stall of a 2D NACA0015 airfoil and a 3D drone in Section IV, and concluding remarks are given in Section V.

II. Surface pressure reconstruction based on discrete empirical interpolation method (DEIM)

Assume that the vector of pressure coefficients $\mathbf{C}_p \in \mathbb{R}^N$ are measured at N locations on the surface of the aircraft, and the matrix \mathbf{S} consists of the corresponding surface area scaled normal vectors

$$\mathbf{S} = \frac{1}{A_{\text{ref}}} [\mathbf{n}_1 S_1, \dots, \mathbf{n}_N S_N] \in \mathbb{R}^{3 \times N},$$

where \mathbf{n}_i is unit normal vector, and S_i is surface area. Notice that in 2D, the last component of \mathbf{n}_i is zero. Then the force coefficients in the body frame due to pressure can be computed by numerical integration

$$\mathbf{F} = \mathbf{SC}_p \in \mathbb{R}^3. \quad (1)$$

The lift and drag coefficients can be obtained by

$$C_l = \mathbf{F} \cdot \mathbf{n}_l, \quad C_d = \mathbf{F} \cdot \mathbf{n}_d. \quad (2)$$

The sensors are usually sparse on the aircraft, in the sense that it is impractical to put too many sensors on the UAVs. Thus it is of interest to use a small number of surface pressure measurements to predict the surface pressure field, and compute the aerodynamic coefficients.

This work chooses to predict the surface pressure coefficients via a reduced basis approximation

$$\mathbf{C}_p \approx \mathbf{C}_p^{\text{rb}} = \overline{\mathbf{C}_p} + \sum_{i=1}^{n_b} b_i \mathbf{u}_i = \overline{\mathbf{C}_p} + \mathbf{Ub},$$

where $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_{n_b}] \in \mathbb{R}^{N \times n_b}$ and $\mathbf{b} = [b_1, b_2, \dots, b_{n_b}]^T \in \mathbb{R}^{n_b}$. The reduced basis is extracted from a collection of snapshots $\mathbb{S} \in \mathbb{R}^{N \times M}$ obtained through numerical simulations, with each column being an instantaneous pressure coefficient vector. The reference value $\overline{\mathbf{C}_p}$ is taken as the average of the columns of \mathbb{S} in this paper. The basis functions can be obtained through proper orthogonal decomposition (POD) [25] based on a singular value decomposition (SVD)

$$\mathbb{S} - \overline{\mathbb{S}} = \widetilde{\mathbf{U}} \Sigma \widetilde{\mathbf{V}}^T, \quad \Sigma = \text{diag}\{\sigma_1, \dots, \sigma_r\},$$

where $\bar{\mathbb{S}} = [\bar{\mathbf{C}}_p, \dots, \bar{\mathbf{C}}_p] \in \mathbb{R}^{N \times M}$, $\sigma_1 \geq \dots \geq \sigma_r \geq 0$, $r = \min\{N, M\}$. The orthonormal reduced basis functions \mathbf{U} are the first n_b columns of $\tilde{\mathbf{U}}$. Given a series of pressure sensor values $\mathbf{C}_p^s \in \mathbb{R}^{n_s}$ at locations $\mathcal{I} = \{\mathcal{I}_1, \dots, \mathcal{I}_{n_s}\} \subset \{1, 2, \dots, N\}$, the basis coefficients \mathbf{b} can be determined by solving a linear system

$$\bar{\mathbf{C}}_p(\mathcal{I}) + \mathbf{U}(\mathcal{I}, :) \mathbf{b} = \mathbf{C}_p^s,$$

where MATLAB notation is used. In this paper, the number of sensors is taken as the number of the basis functions, i.e., $n_s = n_b$, resulting in a square linear system. Therefore, the basis coefficients are $\mathbf{b} = \mathbf{U}^{-1}(\mathcal{I}, :) (\mathbf{C}_p^s - \bar{\mathbf{C}}_p(\mathcal{I}))$ and the predicted surface pressure coefficients are

$$\mathbf{C}_p^{\text{rb}} = \bar{\mathbf{C}}_p + \mathbf{U}\mathbf{U}^{-1}(\mathcal{I}, :) (\mathbf{C}_p^s - \bar{\mathbf{C}}_p(\mathcal{I})) = \bar{\mathbf{C}}_p + \mathbf{R} (\mathbf{C}_p^s - \bar{\mathbf{C}}_p(\mathcal{I})), \quad (3)$$

where $\mathbf{R} = \mathbf{U}\mathbf{U}^{-1}(\mathcal{I}, :) \in \mathbb{R}^{N \times n_s}$. Substituting (3) into (1), the aerodynamic force coefficients are obtained as

$$\mathbf{F} = \mathbf{M}_F^s \mathbf{C}_p^s + \mathbf{M}_F^0, \quad (4)$$

where $\mathbf{M}_F^s = \mathbf{S}\mathbf{R} \in \mathbb{R}^{3 \times n_s}$ and $\mathbf{M}_F^0 = \bar{\mathbf{S}}\bar{\mathbf{C}}_p - \mathbf{S}\mathbf{R}\bar{\mathbf{C}}_p(\mathcal{I}) \in \mathbb{R}^3$. One concludes from (4) that the overall computational cost of the prediction is $O(6n_s)$, as \mathbf{M}_F^s and \mathbf{M}_F^0 can be computed during an offline stage, and real-time online prediction is fast when a small number of sensors are used.

The sensor locations \mathcal{I} need to be optimized for surface pressure reconstruction accuracy. This work adopts the DEIM [27] in Algorithm 1 to select the sensor locations, which is a greedy algorithm to find the most important interpolation points. Here the n_s sensors are selected from m candidate locations.

Algorithm 1: DEIM

Input: Orthonormal bases $\mathbf{U} \in \mathbb{R}^{N \times n_s}$, candidate indices $\tilde{\mathcal{I}} = \{\tilde{\mathcal{I}}_1, \dots, \tilde{\mathcal{I}}_m\} \subset \{1, 2, \dots, N\}$

Output: Selected indices $\mathcal{I} = \{\mathcal{I}_1, \dots, \mathcal{I}_{n_s}\} \subset \tilde{\mathcal{I}}$

```

1  $\mathcal{I}_1 = \arg \max_{\tilde{\mathcal{I}}} \{|\mathbf{U}(\tilde{\mathcal{I}}, 1)|\}, \mathcal{I} = [\mathcal{I}_1];$ 
2 for  $\ell = 2, \dots, n_s$  do
3   Solve  $\mathbf{U}(\mathcal{I}, : (\ell - 1))\mathbf{c} = \mathbf{U}(\mathcal{I}, \ell)$  for  $\mathbf{c}$ ;
4    $\mathcal{I}_\ell = \arg \max_{\tilde{\mathcal{I}}} \{|\mathbf{U}(:, \ell) - \mathbf{U}(:, : (\ell - 1))\mathbf{c}| \};$ 
5    $\mathcal{I} = [\mathcal{I}, \mathcal{I}_\ell].$ 
6 end

```

III. Aerodynamic coefficients prediction enhanced by machine learning

The characteristics of the flow field can be highly nonlinear and very complex, e.g., in the region of dynamic stall. In such cases, the DEIM prediction is not efficient as a large number of bases need to be used, corresponding to many sensors, since a linear subspace is employed in the reduced basis approximation. The approximation ability of the NN has been exploited in many tasks due to its nonlinear nature, and we propose to use an NN as a correction term together with the DEIM prediction. To be specific,

$$\mathbf{F} = \mathbf{F}^{\text{DEIM}} + \text{NN}(\mathbf{C}_p^s; \Theta),$$

where the input of the NN is the pressure sensor values at the selected locations, and the output is the difference in the aerodynamic force prediction coefficients between the ground truth \mathbf{F} and the DEIM prediction $\mathbf{F}^{\text{DEIM}} = \mathbf{M}_F^s \mathbf{C}_p^s + \mathbf{M}_F^0$.

Figure 1 presents a sketch of the architecture of the fully-connected NN used in this work. The NN can be expressed as

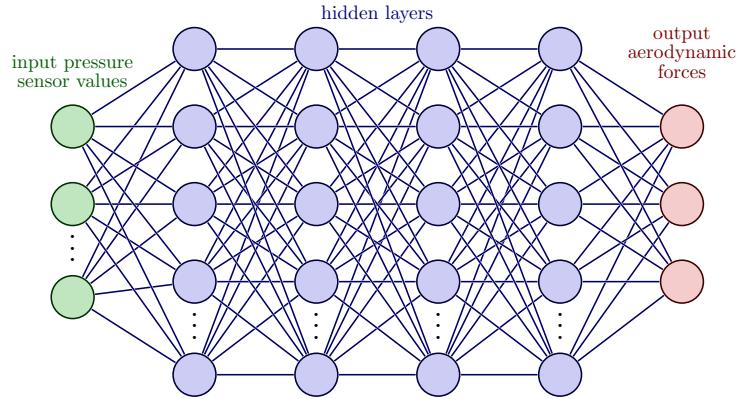


Fig. 1 A sketch of the fully-connected NN used in this work.

$$\text{NN}(\cdot; \Theta) = L_{n_L}(\cdot; \Theta_{n_L}) \circ \varphi_{n_L-1}(L_{n_L-1}(\cdot; \Theta_{n_L-1})) \circ \cdots \varphi_1(L_1(\cdot; \Theta_1)),$$

where L_j is a fully-connected layer, $j = 1, \dots, n_L$, with the weights and biases Θ_j . The activation function φ_j is chosen as ReLU in this paper. As the main dynamics has been captured by the DEIM model and the input dimension n_s is low, the size of the NN is not large, which means that the offline training of the NN is efficient and the online evaluation is fast. It should be mentioned that the ground truth is obtained from numerical or experimental data in this paper, and the NN correction term can be viewed as a closure for the complex behavior between the DEIM prediction and ground truth, e.g., viscous forces, deviation of the numerical simulations, uncertainty in the experiments, or other missing effects.

The NN is trained to minimize the error in the training set

$$\Theta = \arg \min \sum_{\substack{t \in t_{\text{training}} \\ f \in f_{\text{training}}}} \|\mathbf{F}^{\text{DEIM}}(t, f) + \text{NN}(\mathbf{C}_p^s(t, f); \Theta) - \mathbf{F}(t, f)\|_2^2.$$

Our implementation of the NN is built on the PyTorch library [29]. In all the tests, the mini-batch ADAM optimizer with an initial learning rate of 0.001 is adopted to train the NN, and the StepLR scheduler with step size 50 and decay rate 0.95 is used, so that the learning rate is $\lambda = 0.001 \times 0.95^{\lfloor n/50 \rfloor}$, where n is the number of epochs. An early stopping technique is used to avoid overfitting, i.e., the training is terminated if the error in the validation set has not improved for 100 epochs. We also use the weight decay technique implemented in PyTorch as a regularization. Notice that the force coefficients in the body frame are used to train the NN, so that the angle of attack is not included in the input of the NN. The final outputs are transformed based on the angle of attack to obtain the lift and drag coefficients

$$C_l^{\text{NN}} = (\mathbf{F}^{\text{DEIM}} + \text{NN}(\mathbf{C}_p^s; \Theta)) \cdot \mathbf{n}_l, \quad C_d^{\text{NN}} = (\mathbf{F}^{\text{DEIM}} + \text{NN}(\mathbf{C}_p^s; \Theta)) \cdot \mathbf{n}_d.$$

The entire workflow is shown in Fig. 2.

IV. Applications

A. Numerical simulation setup

In this work, the numerical snapshots of the transient aerodynamic flows are computed by using the open-source CFD solver OpenFOAM [30] (v2112), which solves the incompressible unsteady Reynolds-Averaged Navier–Stokes equations (URANS) in an arbitrary Lagrangian-Eulerian (ALE) framework, with the $k-\omega$ -Shear Stress Transport (SST) turbulence model [31] as a closure. The cyclic arbitrary mesh interface (AMI) is used to model the sliding interface between the static zone and the rotating zone, where the airfoil or drone is located at the center of the rotating zone. The PIMPLE solver is adopted for the transient simulations, while the flow fields are initialized by using the steady-state solutions from the SIMPLE solver. The movement of the 2D airfoil and 3D drone is modeled by the harmonic pitching

$$\alpha(t) = \alpha_0 + A \sin(2\pi ft).$$

A sketch of the numerical simulation of the 2D NACA0015 airfoil is shown in Fig. 3

B. 2D NACA0015 airfoil

The approach is first verified by predicting the aerodynamic coefficients in dynamic stall of a 2D NACA0015 airfoil. The experimental data were provided by He et al. [32], collected based on a recirculating wind tunnel with an open

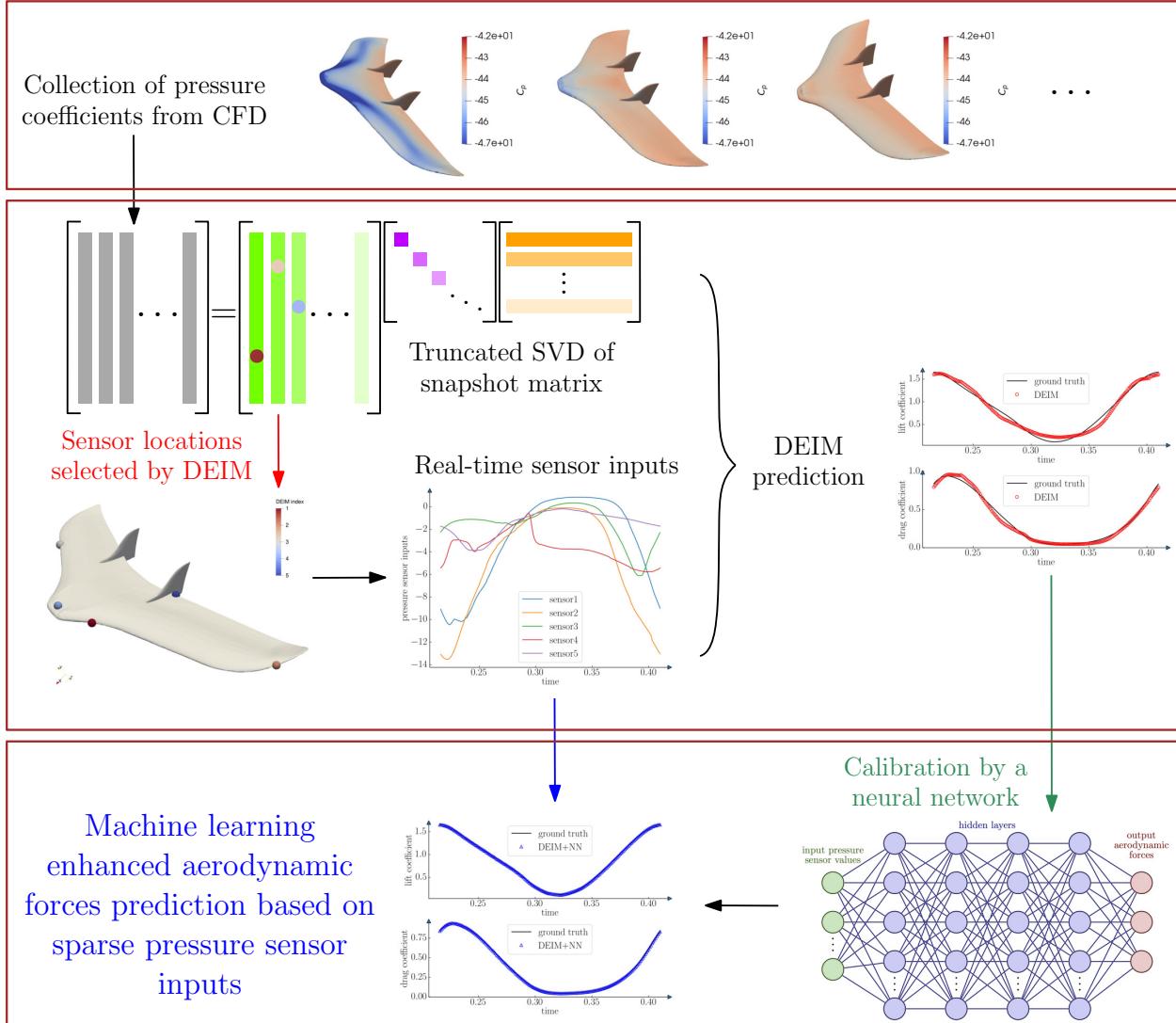


Fig. 2 The workflow of the proposed machine learning-enhanced aerodynamic forces prediction based on sparse pressure sensor inputs.

jet test section. This paper only uses the data in the pitching movement of the airfoil, while the experiment in [32] also considered the flapping of the trailing edge. It should be mentioned that due to the use of the open jet test section, the experimental aerodynamic coefficients must be corrected to recover the case of a full wind tunnel. Two cases are considered, where the DEIM models are built on the pressure coefficients from the experiment and URANS simulation, respectively. The former is used to verify the effectiveness of using NN as a correction term, while the latter follows the proposed approach based on data fusion from the numerical simulation and experiment. Note that the input pressure coefficients do not contain viscous forces, so the NN is also used to model the viscous effects. In the second case, the sensor locations and DEIM model are obtained from the URANS simulation data, while the prediction takes the experimental pressure coefficients as input, thus the NN also models the deviation between the URANS simulation and

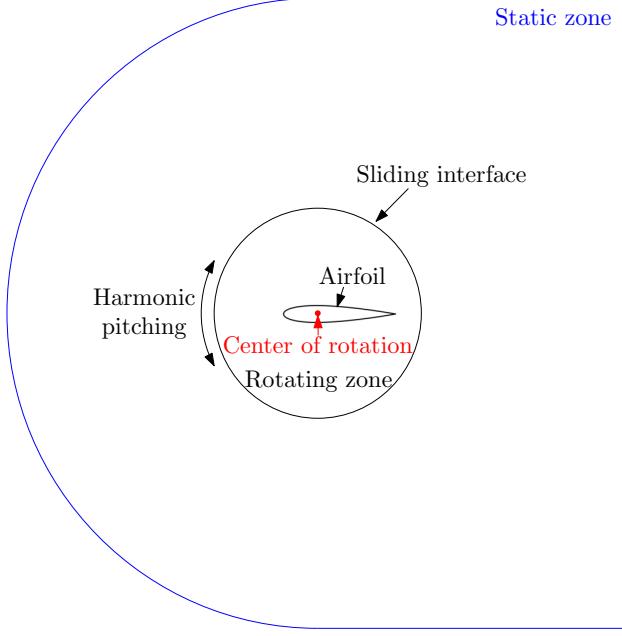
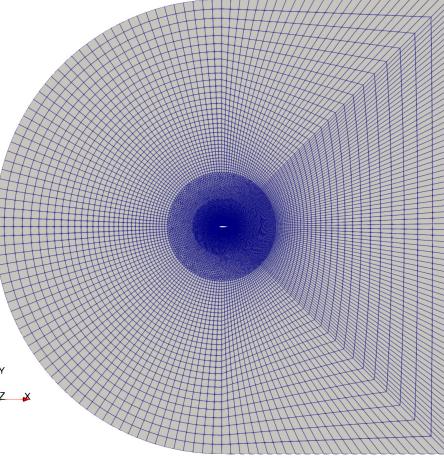


Fig. 3 A sketch of the numerical simulation of the 2D NACA0015 airfoil.

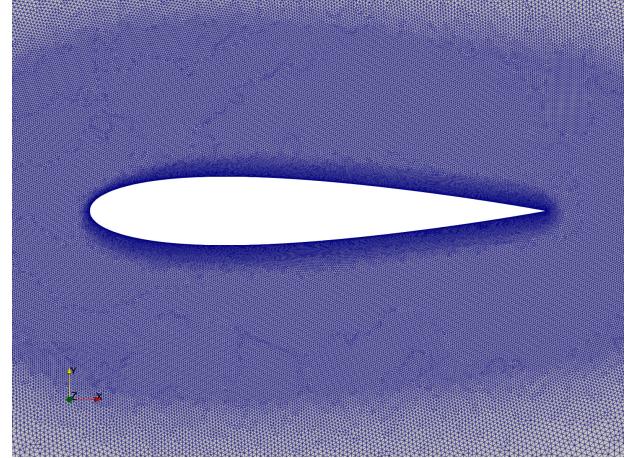
the experiment.

In this test, the parameters in the pitching movement are chosen as $\alpha_0 = 20$ deg, $A = 8$ deg, and the freestream conditions are $\rho_\infty = 1.146 \text{ kg/m}^3$, $V_\infty = 30 \text{ m/s}$, $v_\infty = 1.655 \times 10^{-5} \text{ m}^2/\text{s}$. The reference surface area is $A_{\text{ref}} = 0.0225 \text{ m}^2$ with the chord length 0.3 m, so that the Reynolds number based on the chord length is $Re = 5.4 \times 10^5$. The pitching frequencies in the experiment are $f = 0.796, 2.387, 3.183, 4.775, 1.592, 3.979 \text{ Hz}$, where the first 4 frequencies are used for training the NN, and the last two are used for validation and testing, respectively. In the first case, the training frequencies are used to build the DEIM model. In the second case, the URANS results with 10 pitching frequencies uniform in 0.5 to 5 Hz are used to obtain the DEIM model. The computational mesh is generated using Gmsh [33], consisting of about 2.35×10^5 cells and 8.38×10^5 faces, as shown in Fig. 4. The visualization of the velocity field with a specific frequency and time using surface line integral convolution (LIC) is also presented, which clearly shows that the complex flow separation happens near the upper surface and trailing edge of the airfoil.

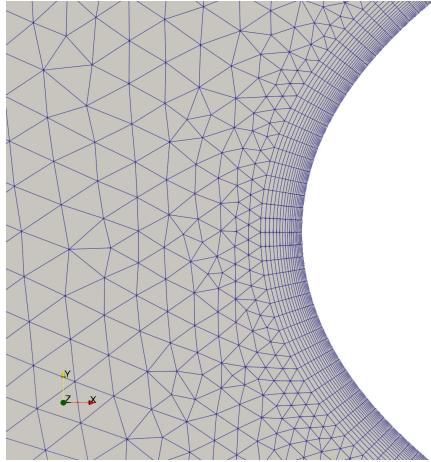
There are 36 sensor locations on the airfoil in the experiment, which serve as the candidate locations to be selected by the DEIM based on the experimental data in the first case, see Fig. 5 with $n_s = 5, 8, 10$. In the second case, the mesh centers, among a total of 1316 surface elements, nearest to the experimental sensor locations are used as the candidate locations, and the sensor locations are selected by the DEIM based on the URANS data, shown in Fig. 6 with $n_s = 5, 8, 10$. In the figures, the indices are the order of importance of the locations resulting from the selection in the DEIM. The selected locations predominantly lie on the upper surface of the airfoil and concentrate near the leading and trailing edges, which is consistent with the flow characteristics in Fig. 4.



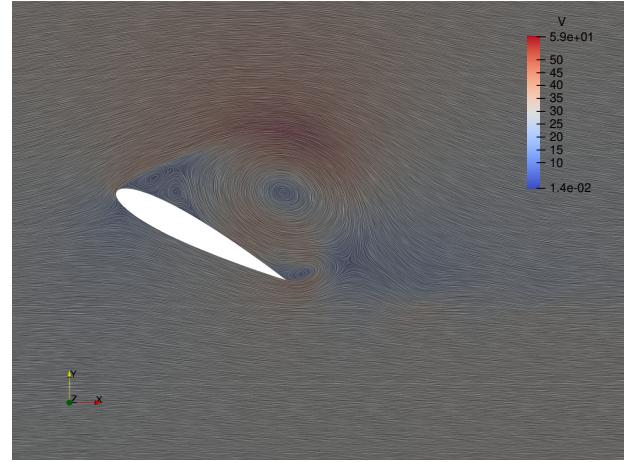
(a) The whole computational domain.



(b) Close view of the mesh.



(c) Further close view of the mesh.



(d) Visualization using surface line integral convolution (LIC) colored by the magnitude of the velocity.

Fig. 4 Computational mesh and URANS results with $f = 3.979 \text{ Hz}$ at $t = 0.3 \text{ s}$.

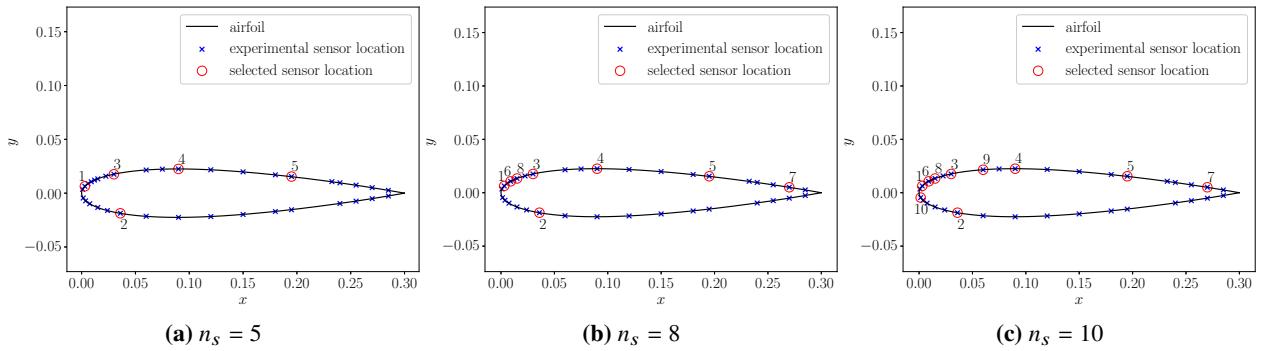


Fig. 5 Sensor locations on the airfoil selected by the DEIM based on the experimental data.

The comparisons of the corrected and uncorrected experimental, and URANS aerodynamic coefficients are shown in Fig. 7. One observes that the URANS results can capture the main flow features, but deviate from the corrected experimental results. To further examine whether the reduced subspace generated by the URANS simulation can

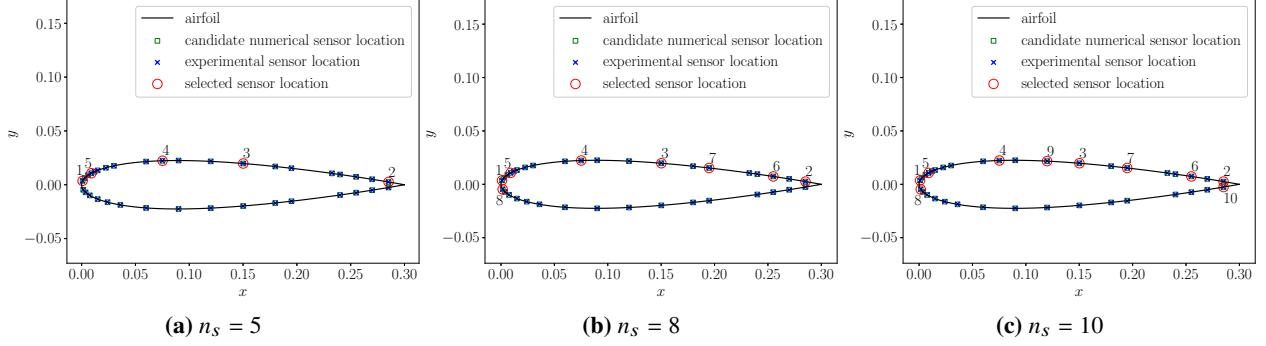


Fig. 6 Sensor locations on the airfoil selected by the DEIM based on the URANS data.

represent the main features of the surface pressure field, the projection errors ϵ_{proj} of the experimental data onto the reduced space are computed

$$\epsilon_{\text{proj}} = \frac{1}{N_{t_{\text{exper}}} N_{f_{\text{exper}}}} \sum_{\substack{t \in t_{\text{exper}} \\ f \in f_{\text{exper}}}} \frac{\left\| (I_{n_s} - \widehat{\mathbf{U}}\widehat{\mathbf{U}}^T) (\mathbf{C}_p^{\text{exper}}(t, f) - \overline{\mathbf{C}_p}(\mathcal{I})) \right\|_2}{\left\| \mathbf{C}_p^{\text{exper}}(t, f) - \overline{\mathbf{C}_p}(\mathcal{I}) \right\|_2},$$

where $\widehat{\mathbf{U}} \in \mathbb{R}^{m \times n_s}$ are the orthonormal reduced basis functions obtained by performing the Gram-Schmidt method on $\mathbf{U}(\mathcal{I}, :)$. The scaled singular values in the SVD of the URANS data are plotted in Fig. 8, which shows that a larger basis improves the approximation of the numerical snapshots as expected. One also observes from Fig. 8 that the projection errors decay as n_b increases, reflecting that a larger reduced basis can express the experimental data better.

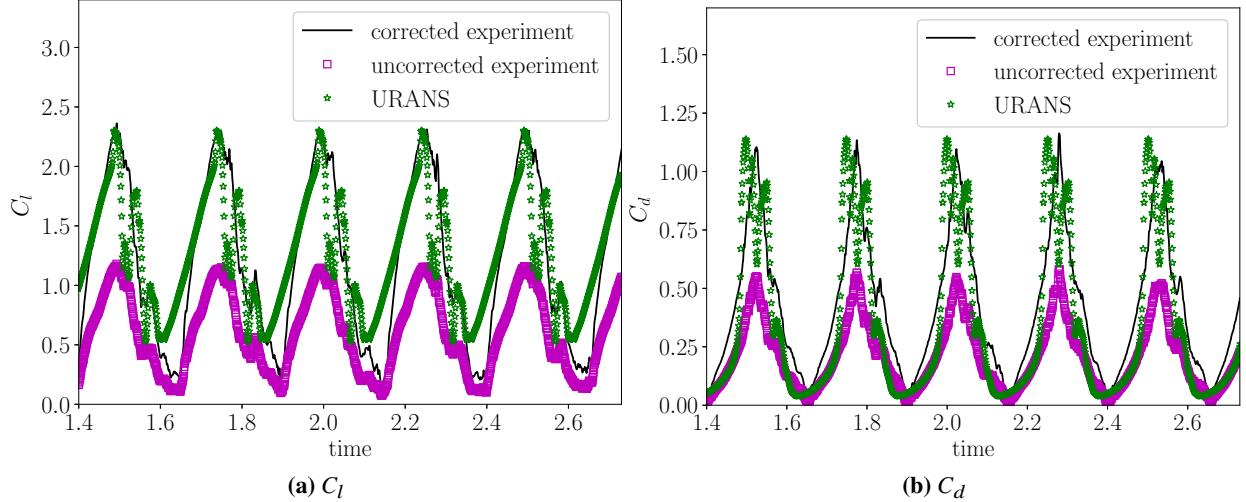


Fig. 7 2D airfoil: Comparison of the corrected and uncorrected experimental, and URANS aerodynamic coefficients.

To find a preferred architecture of the NN, we perform a grid search with 2, 3, 4 hidden layers, 10, 20, 30, 40 neurons in each layer, 10^{-5} , 10^{-6} , 10^{-7} weight decay, with a mini-batch size of 64 in the training. When the DEIM model is built

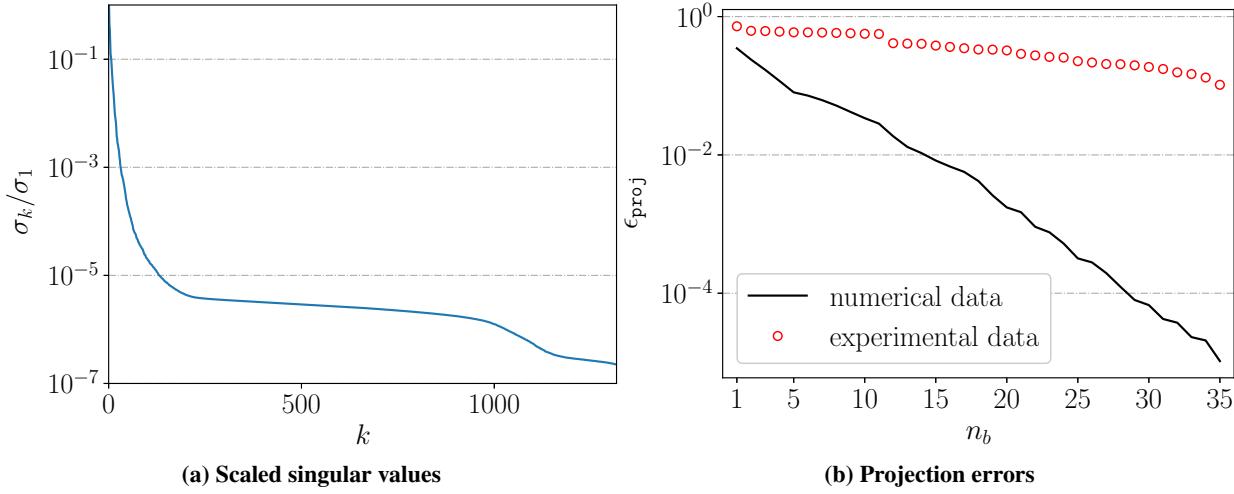


Fig. 8 2D airfoil: The scaled singular values and the projection errors ϵ_{proj} w.r.t. n_b of the URANS and experimental data on the reduced space generated by the URANS data.

on the URANS data, the best NN architecture is obtained when $n_s = 10$, consisting of 2 hidden layers with 10 neurons in each layer, and a weight decay as 10^{-5} . Figures 9-10 show the lift and drag coefficients C_l, C_d with respect to the angle of attack α for different n_s in two cases, where only one whole period is presented. Figures 11-12 plot the evolution of C_l, C_d with respect to time. The maximal C_l appears at $\alpha \approx 26$ deg. When the airfoil is near the dynamic stall region with α in [24, 28] deg, the errors of the DEIM prediction become larger. The lift and drag coefficients C_l, C_d from the DEIM prediction deviate from the experimental data, especially near the minimal and maximal α , and increasing n_s does not improve the results. It is observed that the predictions from the DEIM models based on the experimental data are more accurate, as expected. The URANS data based DEIM model can capture the main dynamics, so it makes sense to calibrate the model by adding a correction term. The results obtained by adding the NN corrections are much closer to the experimental data in both cases, especially near the minimum of C_l, C_d , see Figs. 11-12. This verifies the effectiveness of the NN correction term and our proposed approach employing the data fusion from the numerical simulation and experiment.

The following ℓ^2 and ℓ^∞ errors in the lift and drag coefficients for testing times t_{testing} and frequencies f_{testing} are evaluated,

$$\epsilon_a^z = \sqrt{\frac{1}{N_{t_{\text{testing}}} N_{f_{\text{testing}}}} \sum_{\substack{t \in t_{\text{testing}} \\ f \in f_{\text{testing}}}} |C_a^{\text{exper}}(t, f) - C_a^z(t, f)|^2},$$

$$\epsilon_{a,\infty}^z = \max_{\substack{t \in t_{\text{testing}} \\ f \in f_{\text{testing}}}} |C_a^{\text{exper}}(t, f) - C_a^z(t, f)|,$$

$$\alpha(\epsilon_{a,\infty}^z) = \arg \max_{\substack{t \in t_{\text{testing}} \\ f \in f_{\text{testing}}}} |C_a^{\text{exper}}(t, f) - C_a^z(t, f)|,$$

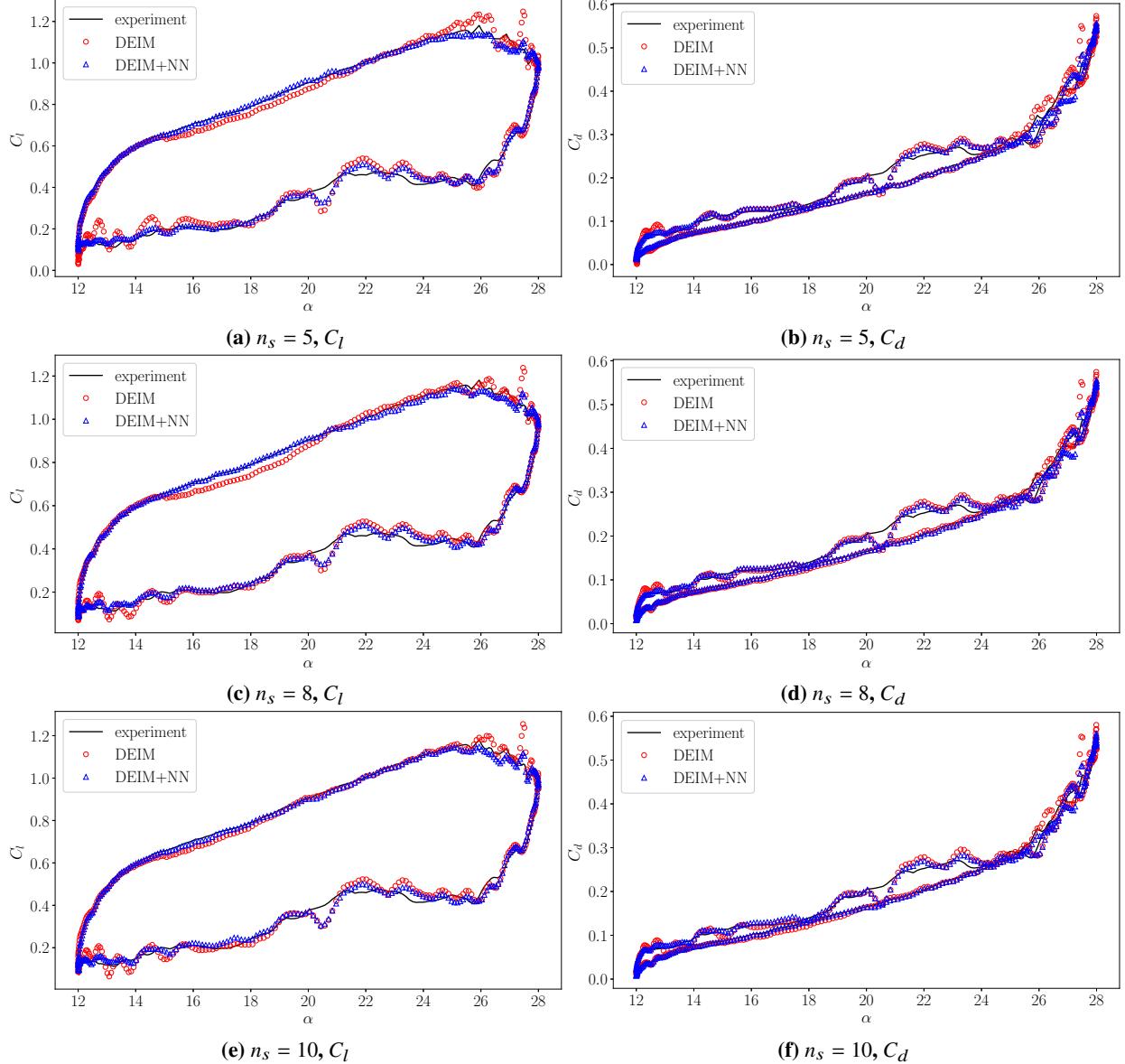


Fig. 9 2D airfoil: C_l, C_d w.r.t. α for the testing experimental data without noise in the pressure sensor inputs. The DEIM models are based on the experimental data.

where $N_{t_{\text{testing}}}$ and $N_{f_{\text{testing}}}$ are the numbers of the testing times and frequencies, respectively, with $a = l, d$ and $z = \text{DEIM}, \text{NN}$. The errors are listed in Tables 1-2. The results show that the DEIM combined with the NN gives more accurate predictions than only using the DEIM in both cases, indicating that the NN correction term can accurately bridge the gap between the DEIM prediction and the ground truth. Although the DEIM models based on experimental data are more accurate, the number of sensors in 3D is limited so the experimental data cannot be used to obtain optimal sensor locations. One observes that in the second case, the smallest errors in C_l and C_d with the NN correction are 2.21×10^{-2} and 1.07×10^{-2} , respectively, and the DEIM+NN gives about 7 and 4 times more accurate C_l and C_d than DEIM, respectively. The online CPU time costs of the DEIM and NN parts are recorded in Table 5, highlighting the

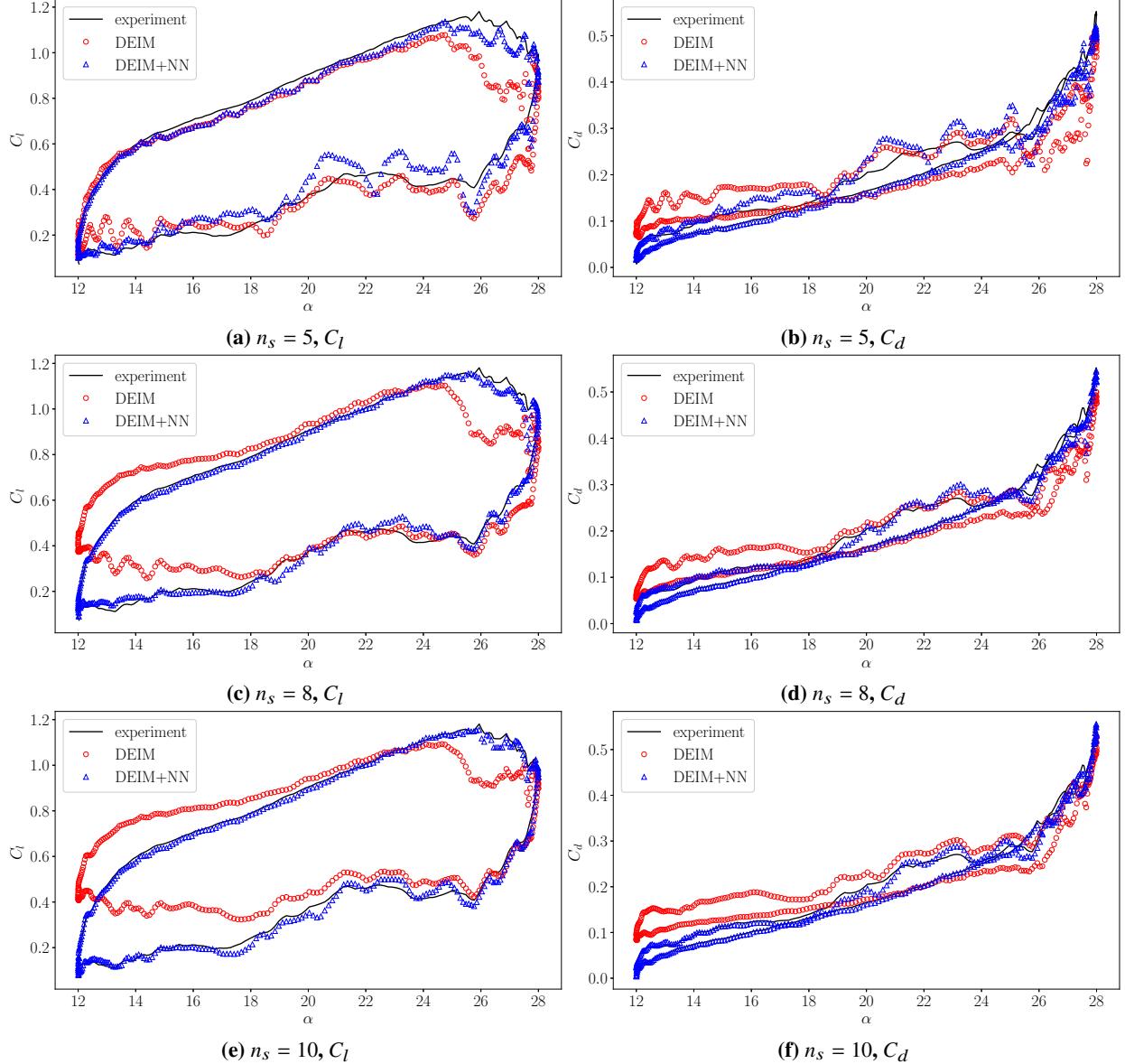


Fig. 10 2D airfoil: C_l, C_d w.r.t. α for the testing experimental data without noise in the pressure sensor inputs. The DEIM models are based on the URANS data.

high computational efficiency of the proposed model.

The predicted lift and drag coefficients C_l, C_d are also corrected and compared with the URANS results, shown in Fig. 13. One observes that the URANS and DEIM prediction deviate from the corrected experimental results, while the DEIM+NN gives accurate results.

To test the robustness of the proposed model, predictions with 1.5% noise in the pressure sensor inputs are considered [34]. The results are shown in Figs. 14-17, which demonstrate that the model can still predict the aerodynamic coefficients well and is not influenced by the noise. From Tables 1-2 one can also see that the smallest errors in C_l and C_d of the URANS data based DEIM prediction with NN correction are 2.24×10^{-2} and 1.07×10^{-2} , which are very

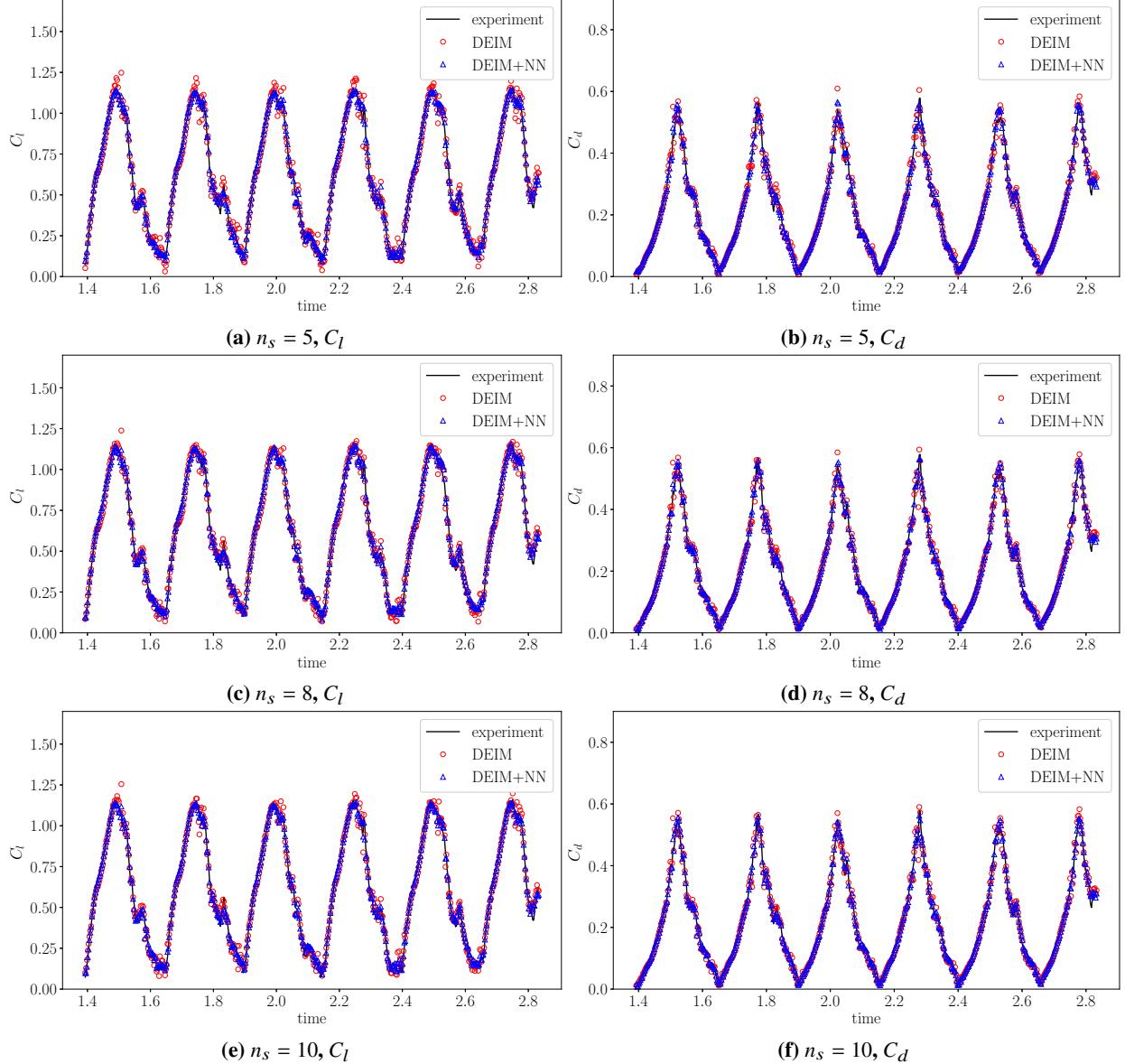


Fig. 11 2D airfoil: C_l, C_d w.r.t. time for the testing experimental data without noise in the pressure sensor inputs. The DEIM models are based on the experimental data.

close to the smallest errors of the predictions without noise, confirming that the model is not sensitive to noise.

C. 3D drone

In this section, the DEIM+NN approach is used to predict the aerodynamic coefficients during dynamic stall of a 3D drone. As there is no experimental data, the pressure coefficients from the URANS simulation are used to select the sensor locations and build the DEIM model, and then the numerical pressure coefficients at the selected locations serve as the sensor inputs. It is worth mentioning that the sensor inputs do not consider the viscous forces, so the NN also models the viscous effects.

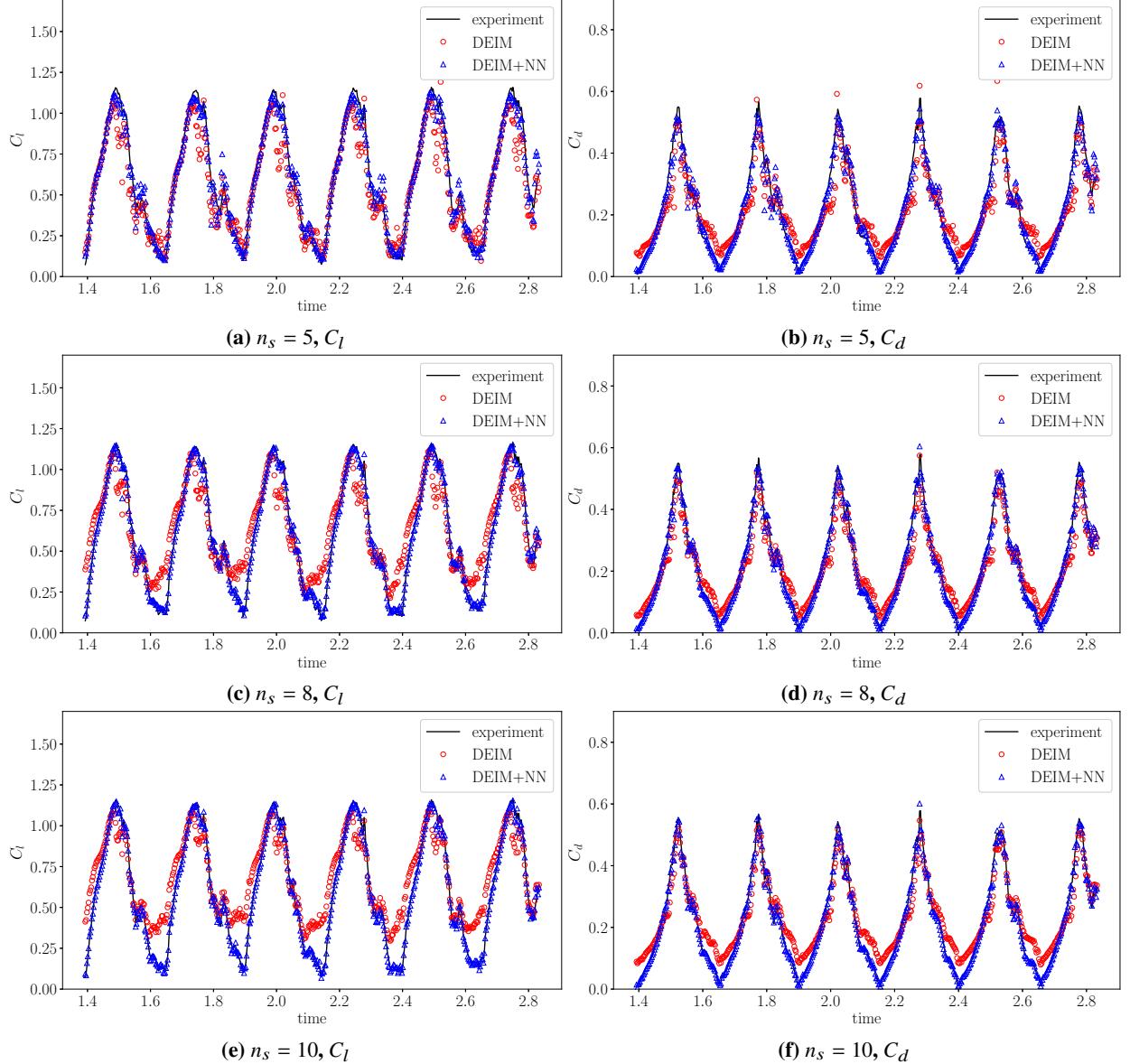


Fig. 12 2D airfoil: C_l, C_d w.r.t. time for the testing experimental data without noise in the pressure sensor inputs. The DEIM models are based on the URANS data.

The parameters in the pitching movement are chosen as $\alpha_0 = 20$ deg, $A = 15$ deg, and the freestream conditions are $\rho_\infty = 1.146 \text{ kg/m}^3$, $V_\infty = 20 \text{ m/s}$, $\nu_\infty = 1.655 \times 10^{-5} \text{ m}^2/\text{s}$. The reference surface area and chord length of the drone are $A_{\text{ref}} = 0.39 \text{ m}^2$ and 0.3 m, respectively, so the Reynolds number based on the chord length is $Re = 3.6 \times 10^5$. The URANS simulation is performed with 9 pitching frequencies uniform in [4, 8] for training, and two random frequencies in the same domain for validation and testing, respectively. The snappyHexMesh utility in OpenFOAM is used to generate the computational mesh, consisting of about 3.58×10^6 cells and 1.13×10^7 faces. The surface mesh of the drone is shown in Fig. 18, where the surface pressure field and streamlines are also presented for $f = 6.786 \text{ Hz}$ at $t = 0.2 \text{ s}$.

Table 1 2D airfoil: the ℓ^2 errors, ℓ^∞ errors in C_l , and the angles of attack corresponding to the ℓ^∞ errors for different n_s .

	n_s	DEIM			DEIM+NN		
		ϵ_l^{DEIM}	$\epsilon_{l,\infty}^{\text{DEIM}}$	$\alpha(\epsilon_{l,\infty}^{\text{DEIM}})$	ϵ_l^{NN}	$\epsilon_{l,\infty}^{\text{NN}}$	$\alpha(\epsilon_{l,\infty}^{\text{NN}})$
Experimental data based DEIM model							
without noise	5	3.97×10^{-2}	1.82×10^{-1}	27.5 deg	2.35×10^{-2}	9.97×10^{-2}	28.0 deg
	8	3.49×10^{-2}	1.72×10^{-1}	27.5 deg	2.26×10^{-2}	9.46×10^{-2}	24.6 deg
	10	3.41×10^{-2}	1.89×10^{-1}	27.5 deg	2.17×10^{-2}	9.41×10^{-2}	24.6 deg
1.5% noise	5	4.02×10^{-2}	1.80×10^{-1}	27.5 deg	2.41×10^{-2}	9.72×10^{-2}	28.0 deg
	8	3.56×10^{-2}	1.71×10^{-1}	27.5 deg	2.34×10^{-2}	9.50×10^{-2}	24.6 deg
	10	3.45×10^{-2}	1.87×10^{-1}	27.5 deg	2.23×10^{-2}	9.60×10^{-2}	24.8 deg
URANS data based DEIM model							
without noise	5	1.14×10^{-1}	4.83×10^{-1}	27.5 deg	5.37×10^{-2}	2.89×10^{-1}	21.9 deg
	8	1.45×10^{-1}	3.29×10^{-1}	12.3 deg	2.55×10^{-2}	1.71×10^{-1}	27.6 deg
	10	1.56×10^{-1}	3.57×10^{-1}	12.4 deg	2.21×10^{-2}	1.22×10^{-1}	27.0 deg
1.5% noise	5	1.14×10^{-1}	4.79×10^{-1}	27.6 deg	5.41×10^{-2}	2.87×10^{-1}	21.9 deg
	8	1.45×10^{-1}	3.31×10^{-1}	27.6 deg	2.59×10^{-2}	1.72×10^{-1}	27.6 deg
	10	1.56×10^{-1}	3.56×10^{-1}	12.4 deg	2.24×10^{-2}	1.22×10^{-1}	27.0 deg

Table 2 2D airfoil: the ℓ^2 errors, ℓ^∞ errors in C_d , and the angles of attack corresponding to the ℓ^∞ errors for different n_s .

	n_s	DEIM			DEIM+NN		
		ϵ_d^{DEIM}	$\epsilon_{d,\infty}^{\text{DEIM}}$	$\alpha(\epsilon_{d,\infty}^{\text{DEIM}})$	ϵ_d^{NN}	$\epsilon_{d,\infty}^{\text{NN}}$	$\alpha(\epsilon_{d,\infty}^{\text{NN}})$
Experimental data based DEIM model							
without noise	5	1.60×10^{-2}	9.64×10^{-2}	28.0 deg	1.16×10^{-2}	6.98×10^{-2}	28.0 deg
	8	1.41×10^{-2}	9.43×10^{-2}	27.5 deg	9.69×10^{-3}	4.97×10^{-2}	28.0 deg
	10	1.47×10^{-2}	9.65×10^{-2}	27.5 deg	9.60×10^{-3}	4.46×10^{-2}	24.5 deg
1.5% noise	5	1.61×10^{-2}	1.01×10^{-1}	28.0 deg	1.17×10^{-2}	6.88×10^{-2}	28.0 deg
	8	1.43×10^{-2}	9.13×10^{-2}	27.5 deg	9.90×10^{-3}	4.87×10^{-2}	28.0 deg
	10	1.48×10^{-2}	9.36×10^{-2}	27.5 deg	9.82×10^{-3}	4.49×10^{-2}	24.5 deg
URANS data based DEIM model							
without noise	5	5.87×10^{-2}	2.49×10^{-1}	27.5 deg	2.40×10^{-2}	1.39×10^{-1}	27.1 deg
	8	4.40×10^{-2}	1.66×10^{-1}	27.6 deg	1.13×10^{-2}	8.14×10^{-2}	27.6 deg
	10	4.83×10^{-2}	1.07×10^{-1}	26.4 deg	1.07×10^{-2}	6.26×10^{-2}	27.9 deg
1.5% noise	5	5.87×10^{-2}	2.48×10^{-1}	27.5 deg	2.41×10^{-2}	1.34×10^{-1}	27.1 deg
	8	4.40×10^{-2}	1.68×10^{-1}	27.6 deg	1.14×10^{-2}	8.18×10^{-2}	27.6 deg
	10	4.83×10^{-2}	1.07×10^{-1}	26.4 deg	1.07×10^{-2}	6.44×10^{-2}	27.9 deg

The scaled singular values in the SVD are shown in Fig. 19, indicating that the linear subspace is not efficient in capturing the dynamics.

In the test, all surface mesh centers (299, 771 in total) in the computational mesh are taken as candidate locations,

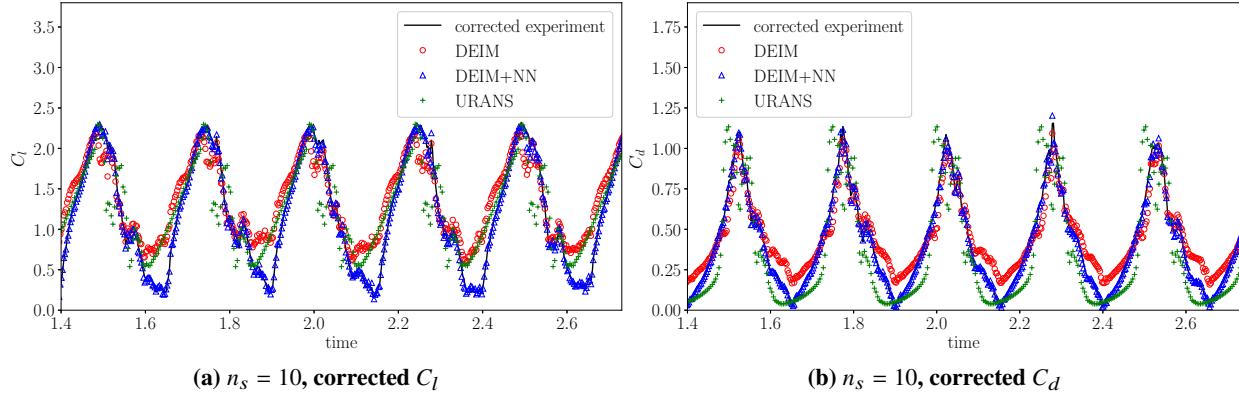


Fig. 13 2D airfoil: The corrected C_l, C_d w.r.t. time for the testing experimental data without noise in the pressure sensor inputs. The DEIM model is based on the URANS data.

and the DEIM is used to select n_s locations. Figure 20 presents the selected sensor locations with $n_s = 5, 10, 15$, where the color corresponds to the order during the selection in the DEIM. One observes that the selected locations lie on the upper surface, and the locations on the leading edges are preferred, similar to the 2D airfoil case.

In the training of the NN, the mini-batch size is 32. We perform a grid search to find a preferred architecture, with 2, 3, 4 hidden layers, 10, 20, 30, 40 neurons each layer, and $10^{-5}, 10^{-6}, 10^{-7}$ weight decay. The best model is obtained with $n_s = 10$, 4 hidden layers with 40 neurons in each layer and a weight decay as 10^{-6} . Figure 21 gives the lift and drag coefficients C_l, C_d with respect to the angle of attack α for different n_s , and Fig. 22 plots the evolution of C_l, C_d with respect to time. The numerical results are only shown during one whole period, as the URANS simulation is periodic in time. The maximal C_l appears at $\alpha \approx 30$ deg, and the drone is near the region of dynamic stall with α in [28, 35] deg. The lift and drag coefficients C_l, C_d predicted by the DEIM model deviate from the URANS simulation results, and the results do not improve with larger n_s . After adding the NN correction, the results are much better, and very close to the URANS simulation, thus the NN correction term is vital in the improvement of the accuracy. The results with 1.5% noise in the pressure sensor inputs are also shown in Figs. 23-24. One observes that the lift and drag coefficients are still well predicted by the DEIM+NN without obvious oscillations.

Similar to the 2D airfoil case, the ℓ^2 and ℓ^∞ errors in C_l, C_d and corresponding α are computed to examine the performance of the proposed approach, listed in Tables 3-4. One observes that when no noise is added, the smallest errors in C_l and C_d by adding the NN correction are 1.67×10^{-3} and 7.32×10^{-4} , respectively, and the DEIM+NN yields at least 30 times more accurate C_l and C_d than the DEIM, which verifies the high accuracy of the proposed approach. When 1.5% noise is added to the pressure sensor inputs, the errors in the lift and drag coefficients are still below 1.0×10^{-2} , suggesting that the approach is robust to noise. The online CPU time costs in Table 5 confirm that the DEIM+NN model is efficient and can be used for real-time prediction.

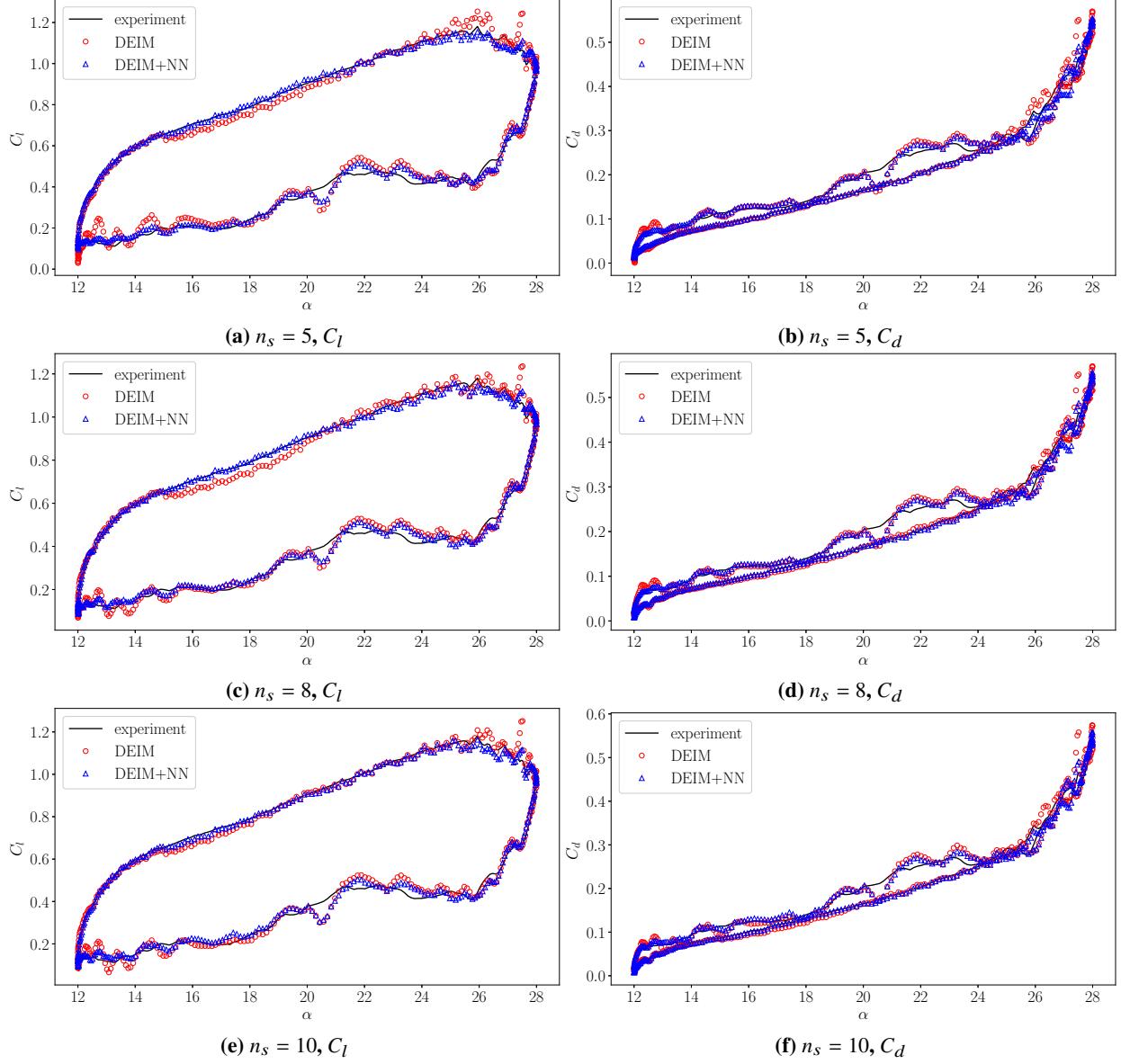


Fig. 14 2D airfoil: C_l, C_d w.r.t. α for the testing experimental data with 1.5% noise in the pressure sensor inputs. The DEIM models are based on the experimental data.

V. Conclusion

In the navigation and control of UAVs, accurate and efficient real-time aerodynamic prediction based on sensor inputs plays an important role. This paper presents a systematic approach for the construction of a data-driven aerodynamic model combined with discrete empirical interpolation method (DEIM) to predict aerodynamic coefficients. Pressure coefficients on the aircraft surface from URANS simulations serve as snapshots and are used to obtain a set of reduced basis. The sensor locations are optimized by the DEIM, and the basis coefficients are computed based on real-time pressure sensor inputs at the selected locations. The aerodynamic forces are computed by integrating the reduced basis reconstruction of the surface pressure distribution. The DEIM model captures main dynamics, and a neural network is

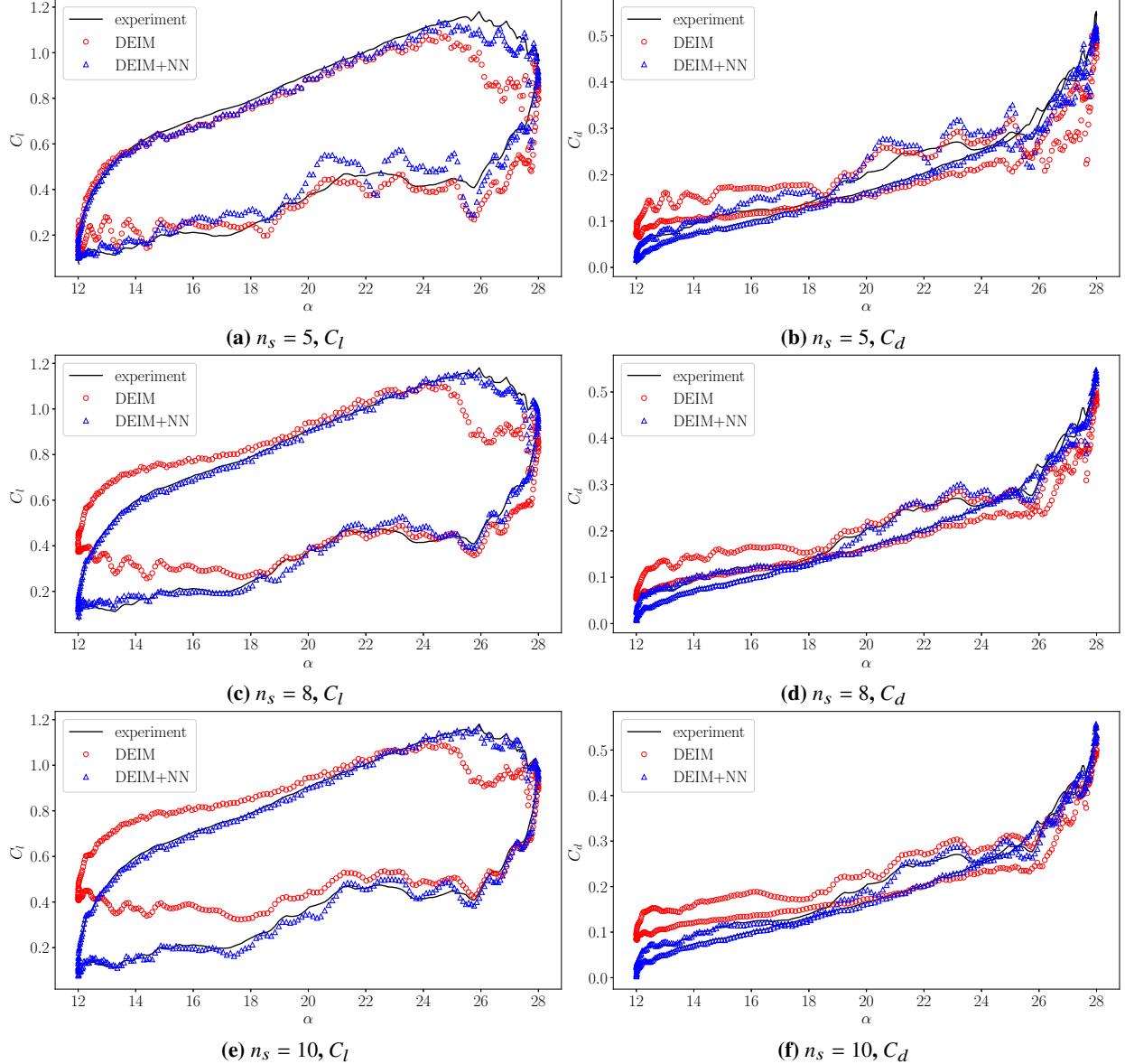


Fig. 15 2D airfoil: C_l, C_d w.r.t. α for the testing experimental data with 1.5% noise in the pressure sensor inputs. The DEIM models are based on the URANS data.

proposed to bridge the gap between the aerodynamic forces of the ground truth and the DEIM prediction, resulting in a more accurate model. The approach is tested on the URANS simulation and experimental data of the dynamic stall of a 2D NACA0015 airfoil, and URANS simulation data of the dynamic stall of a 3D drone. The results show that the approach yields fast and accurate real-time predictions of the aerodynamic coefficients, and the neural network-based correction term improves the accuracy significantly. Furthermore, we demonstrated that the approach is not sensitive to noise in the pressure sensor inputs.

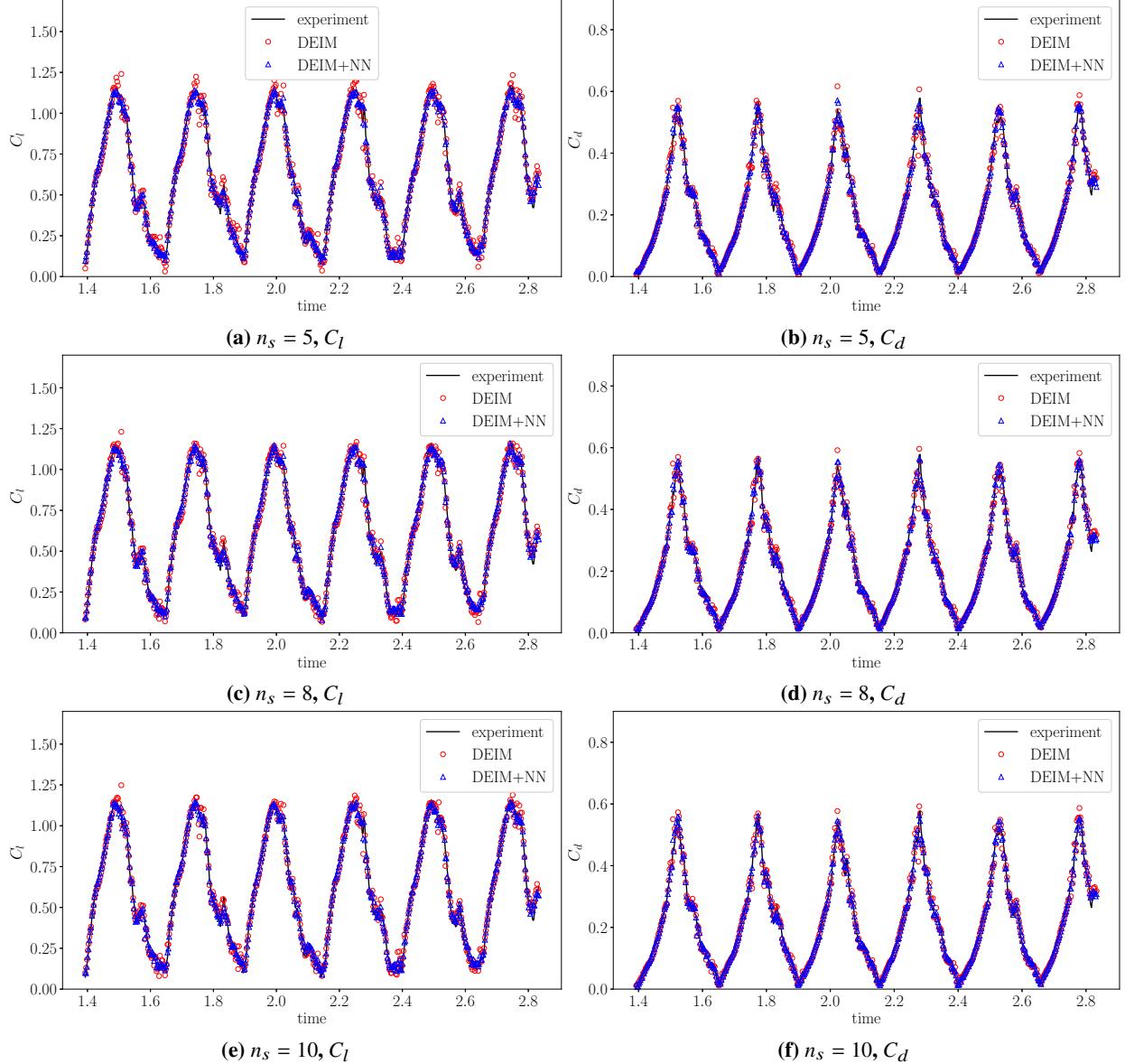


Fig. 16 2D airfoil: C_l, C_d w.r.t. time for the testing experimental data with 1.5% noise in the pressure sensor inputs. The DEIM models are based on the experimental data.

Funding Sources

The first and second authors were partially supported by the Sense Dynamics Project (SD11, Collaborative Data Science Project, Swiss Data Science Center). The second author is also partially supported by Project U2230402 of NSFC.

Acknowledgments

The authors thank Dr. Guosheng He and Prof. Karen Mullenens for providing the wind tunnel experiments data. We are grateful for the inspiring discussions with Dr. Nicolò Ripamonti.

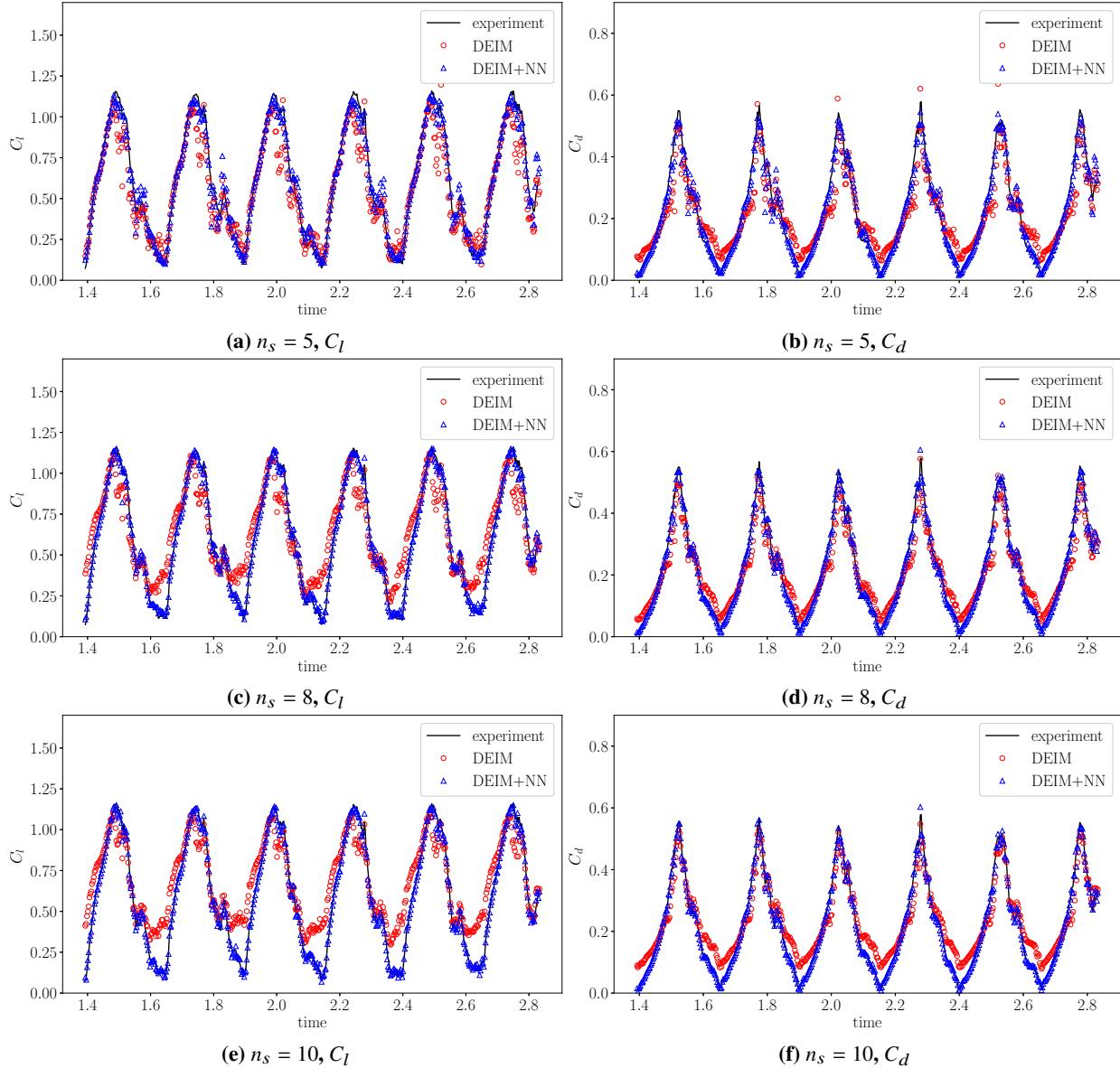
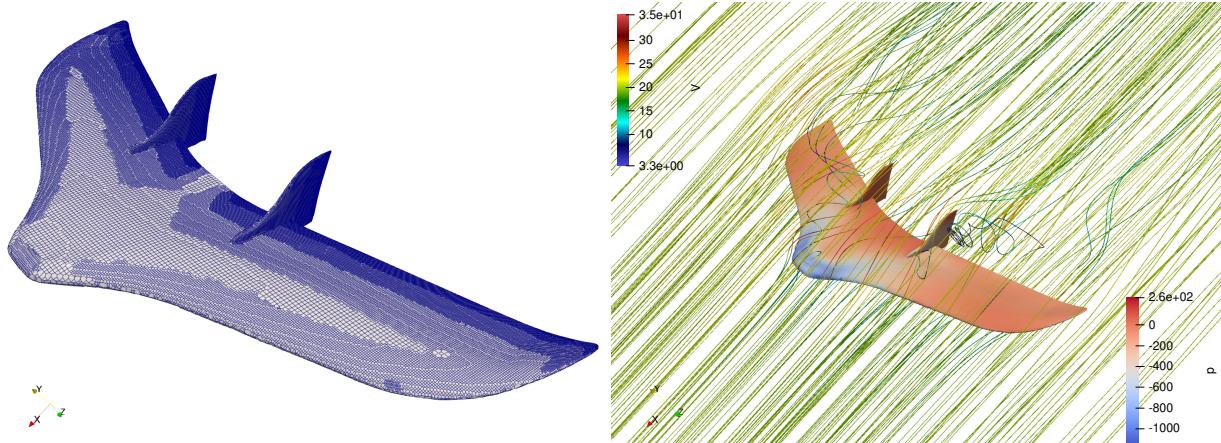


Fig. 17 2D airfoil: C_l, C_d w.r.t. time for the testing experimental data with 1.5% noise in the pressure sensor inputs. The DEIM models are based on the URANS data.

References

- [1] Bryson, M., and Sukkarieh, S., “UAV localization using inertial sensors and satellite positioning systems,” *Handbook of Unmanned Aerial Vehicles*, edited by K. P. Valavanis and G. J. Vachtsevanos, Springer Netherlands, Dordrecht, 2015, pp. 433–460. https://doi.org/10.1007/978-90-481-9707-1_3, URL https://doi.org/10.1007/978-90-481-9707-1_3.
- [2] Lau, T. K., Liu, Y. H., and Lin, K. W., “Inertial-based localization for unmanned helicopters against GNSS outage,” *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 49, No. 3, 2013, pp. 1932–1949. <https://doi.org/10.1109/taes.2013.6558029>.
- [3] Nassar, S., and El-Sheimy, N., “Wavelet analysis for improving INS and INS/DGPS navigation accuracy,” *Journal of Navigation*,



(a) The surface mesh of the drone.

(b) The surface pressure and streamlines colored by the magnitude of the velocity.

Fig. 18 Computational mesh and URANS results with $f = 6.786 \text{ Hz}$ at $t = 0.2 \text{ s}$.

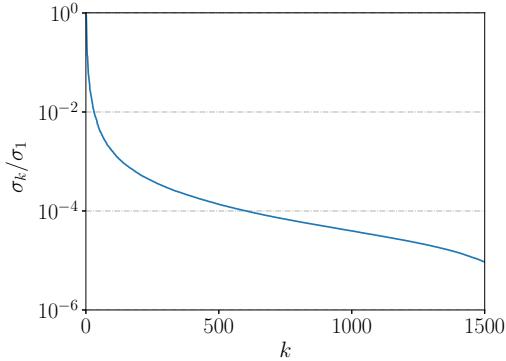
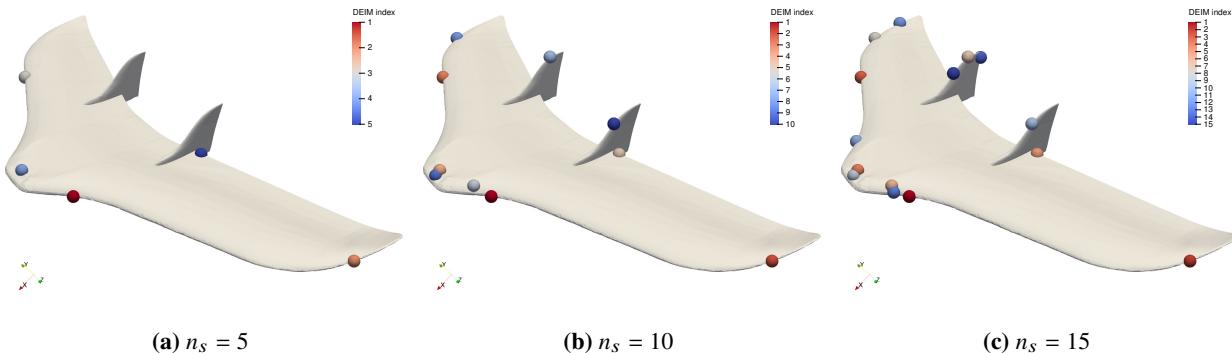


Fig. 19 3D drone: The scaled singular values in the SVD.



(a) $n_s = 5$

(b) $n_s = 10$

(c) $n_s = 15$

Fig. 20 Sensor locations selected by the DEIM on the drone.

Vol. 58, No. 1, 2005, pp. 119–134. <https://doi.org/10.1017/s0373463304003005>, URL https://www.cambridge.org/core/product/identifier/S0373463304003005/type/journal_article.

- [4] Nassar, S., and El-Sheimy, N., “A combined algorithm of improving INS error modeling and sensor measurements for accurate INS/GPS navigation,” *GPS Solutions*, Vol. 10, No. 1, 2006, pp. 29–39. <https://doi.org/10.1007/s10291-005-0149-3>, URL

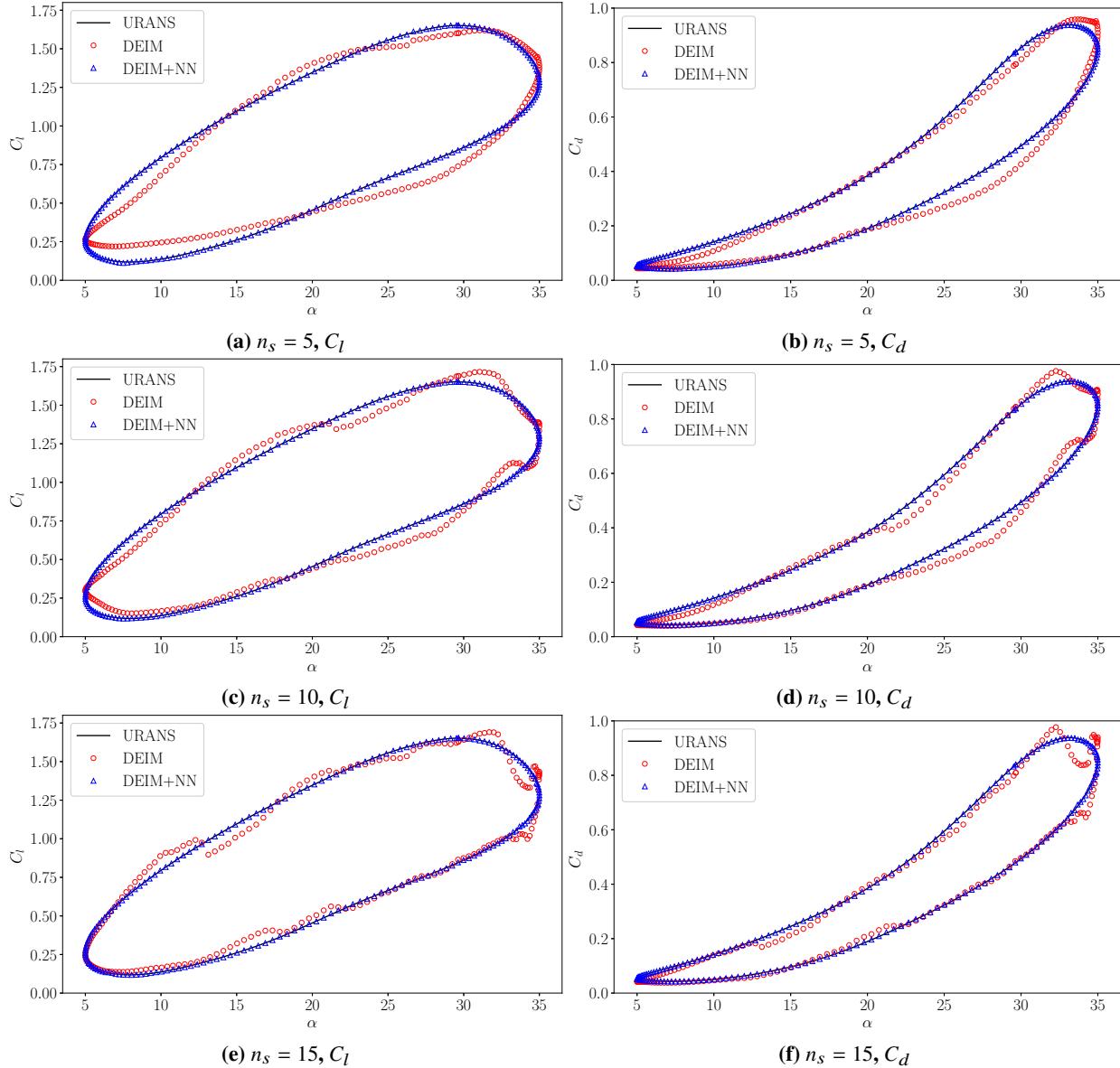


Fig. 21 3D drone: C_l, C_d w.r.t. α for the testing data without noise in the pressure sensor inputs.

<https://doi.org/10.1007/s10291-005-0149-3>.

- [5] Noureldin, A., Karamat, T. B., Eberts, M. D., and El-Shafie, A., “Performance enhancement of MEMS-based INS/GPS integration for low-cost navigation applications,” *IEEE Transactions on Vehicular Technology*, Vol. 58, No. 3, 2009, pp. 1077–1096. <https://doi.org/10.1109/tvt.2008.926076>.
- [6] Yun, S., Lee, Y. J., and Sung, S., “IMU/Vision/Lidar integrated navigation system in GNSS denied environments,” *2013 IEEE Aerospace Conference*, IEEE, 2013, pp. 1–10. <https://doi.org/10.1109/aero.2013.6497403>, ISSN: 1095-323X.
- [7] Madany, Y. M., Elkamchouchi, H. M., and Ahmed, M. M., “Modelling and simulation of robust navigation for unmanned air

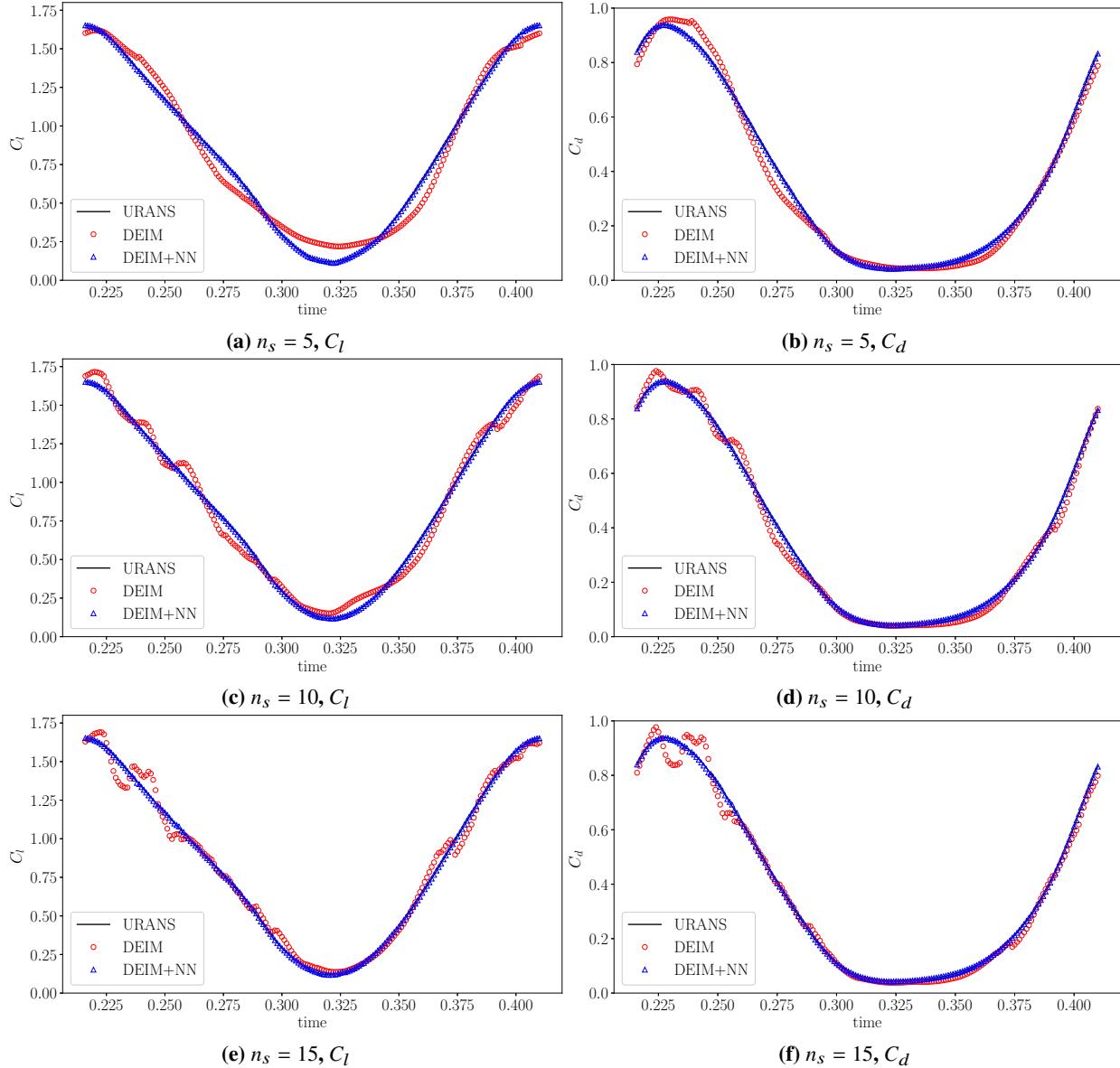


Fig. 22 3D drone: C_l, C_d w.r.t. time for the testing data without noise in the pressure sensor inputs.

systems (UASs) based on integration of multiple sensors fusion architecture,” *2013 European Modelling Symposium*, IEEE, 2013, pp. 719–724. <https://doi.org/10.1109/ems.2013.121>.

- [8] Uijt de Haag, M., Duan, P., and Vadlamani, A., “Flight test and simulation results of an integrated dual airborne laser scanner and inertial navigator for UAV applications,” *2014 IEEE 11th International Multi-Conference on Systems, Signals & Devices (SSDI14)*, IEEE, 2014, pp. 1–6. <https://doi.org/10.1109/ssd.2014.6808889>.
- [9] Koifman, M., and Bar-Itzhack, I., “Inertial navigation system aided by aircraft dynamics,” *IEEE Transactions on Control Systems Technology*, Vol. 7, No. 4, 1999, pp. 487–493. <https://doi.org/10.1109/87.772164>.
- [10] Bryson, M., and Sukkarieh, S., “Vehicle model aided inertial navigation for a UAV using low-cost sensors,” *Proceedings of the*

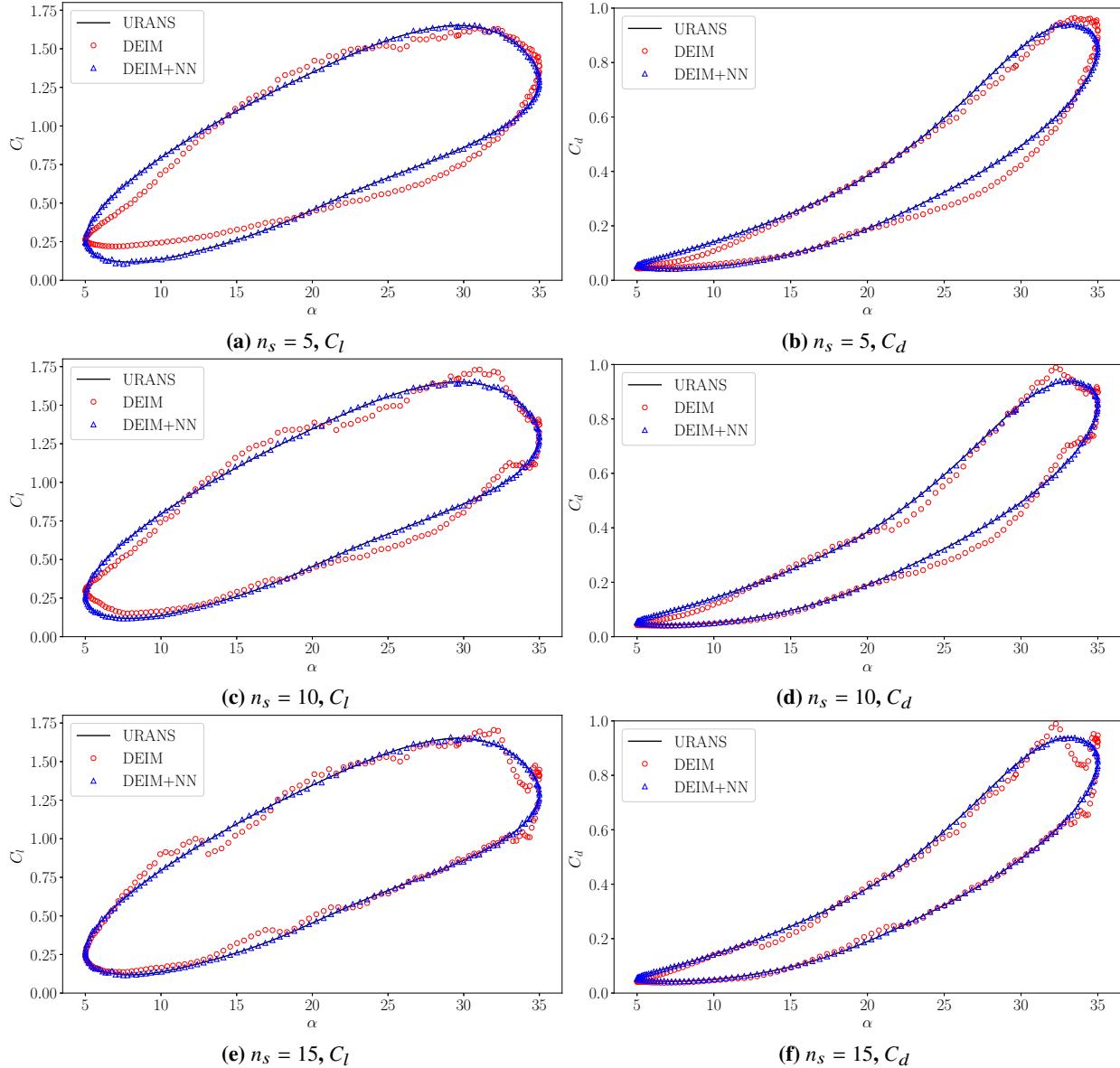


Fig. 23 3D drone: C_l, C_d w.r.t. α for the testing data with 1.5% noise in the pressure sensor inputs.

Australasian Conference on Robotics and Automation, Australian Robotics and Automation Association, 2004, pp. 1–9.

- [11] Vasconcelos, J. F., Silvestre, C., Oliveira, P., and Guerreiro, B., “Embedded UAV model and LASER aiding techniques for inertial navigation systems,” *Control Engineering Practice*, Vol. 18, No. 3, 2010, pp. 262–278. <https://doi.org/10.1016/j.conengprac.2009.11.004>, URL <https://www.sciencedirect.com/science/article/pii/S0967066109002202>.
- [12] Crocoll, P., Seibold, J., Scholz, G., and Trommer, G. F., “Model-aided navigation for a quadrotor helicopter: A novel navigation system and first experimental results,” *NAVIGATION*, Vol. 61, No. 4, 2014, pp. 253–271. <https://doi.org/10.1002/navi.68>, URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/navi.68>.
- [13] Crocoll, P., and Trommer, G. F., “Quadrotor inertial navigation aided by a vehicle dynamics model with in-flight parameter

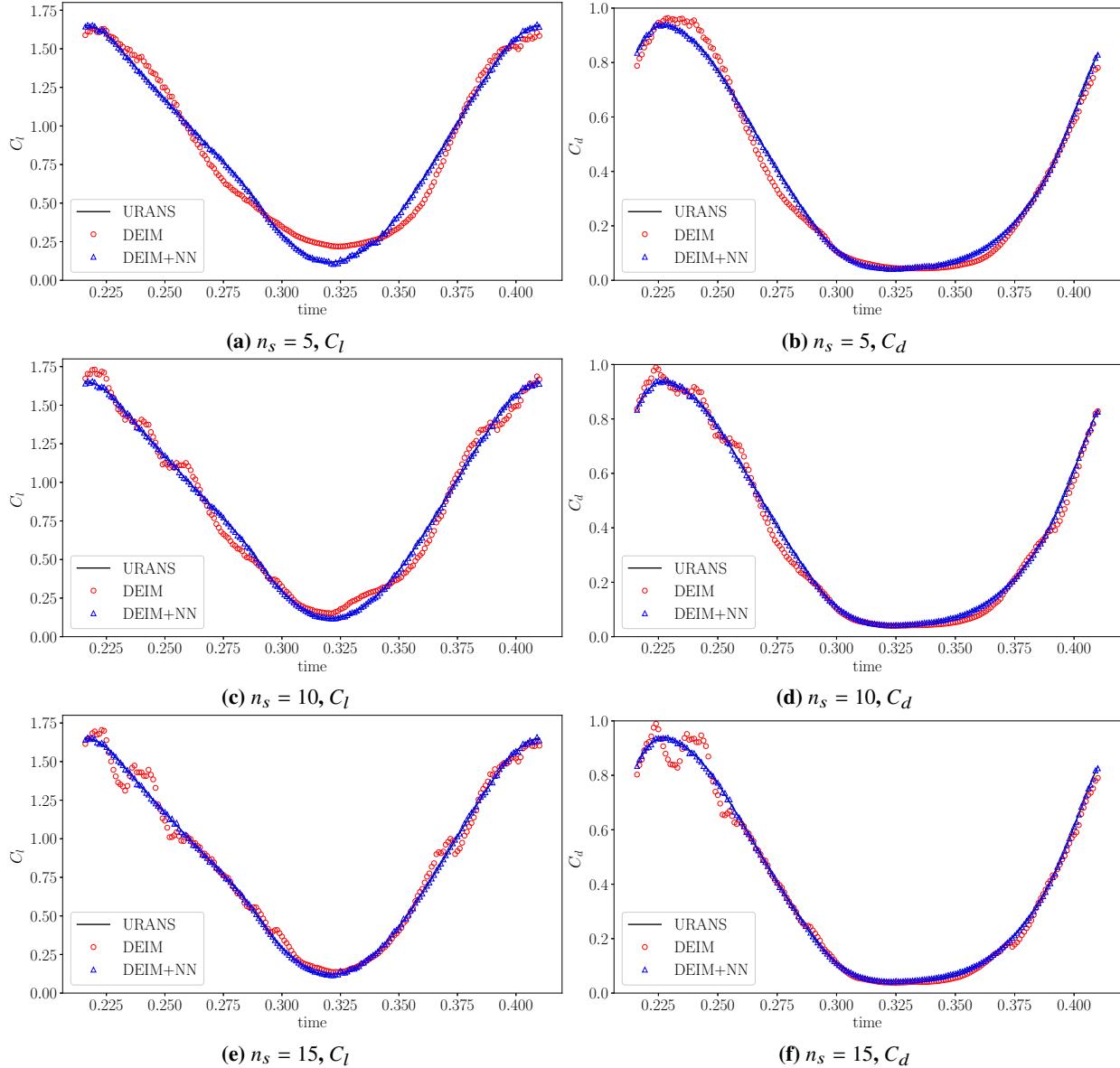


Fig. 24 3D drone: C_l, C_d w.r.t. time for the testing data with 1.5% noise in the pressure sensor inputs.

estimation,” *Proceedings of the 27th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2014)*, 2014, pp. 1784–1795.

- [14] Khaghani, M., and Skaloud, J., “Autonomous vehicle dynamic model-based navigation for small UAVs,” *NAVIGATION*, Vol. 63, No. 3, 2016, pp. 345–358. <https://doi.org/10.1002/navi.140>, URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/navi.140>.
- [15] Khaghani, M., and Skaloud, J., “Assessment of VDM-based autonomous navigation of a UAV under operational conditions,” *Robotics and Autonomous Systems*, Vol. 106, 2018, pp. 152–164. <https://doi.org/10.1016/j.robot.2018.05.007>, URL <https://www.sciencedirect.com/science/article/pii/S0921889017303792>.
- [16] Fei, H. P., Zhu, R., Zhou, Z. Y., and Wang, J. D., “Aircraft flight parameter detection based on a neural network using multiple

Table 3 3D drone: the ℓ^2 errors, ℓ^∞ errors in C_l , and the angles of attack corresponding to the ℓ^∞ errors for different n_s .

	n_s	DEIM			DEIM+NN		
		ϵ_l^{DEIM}	$\epsilon_{l,\infty}^{\text{DEIM}}$	$\alpha(\epsilon_{l,\infty}^{\text{DEIM}})$	ϵ_l^{NN}	$\epsilon_{l,\infty}^{\text{NN}}$	$\alpha(\epsilon_{l,\infty}^{\text{NN}})$
without noise	5	7.46×10^{-2}	1.47×10^{-1}	8.0 deg	2.25×10^{-3}	7.36×10^{-3}	7.5 deg
	10	5.66×10^{-2}	1.10×10^{-1}	28.1 deg	1.67×10^{-3}	7.30×10^{-3}	17.7 deg
	15	5.28×10^{-2}	1.55×10^{-1}	35.0 deg	2.78×10^{-3}	1.11×10^{-2}	34.9 deg
1.5% noise	5	7.56×10^{-2}	1.51×10^{-1}	8.0 deg	6.27×10^{-3}	2.23×10^{-2}	5.0 deg
	10	5.76×10^{-2}	1.10×10^{-1}	32.3 deg	6.35×10^{-3}	1.82×10^{-2}	25.8 deg
	15	5.37×10^{-2}	1.62×10^{-1}	35.0 deg	6.18×10^{-3}	1.94×10^{-2}	34.8 deg

Table 4 3D drone: the ℓ^2 errors, ℓ^∞ errors in C_d , and the angles of attack corresponding to the ℓ^∞ errors for different n_s .

	n_s	DEIM			DEIM+NN		
		ϵ_d^{DEIM}	$\epsilon_{d,\infty}^{\text{DEIM}}$	$\alpha(\epsilon_{d,\infty}^{\text{DEIM}})$	ϵ_d^{NN}	$\epsilon_{d,\infty}^{\text{NN}}$	$\alpha(\epsilon_{d,\infty}^{\text{NN}})$
without noise	5	3.22×10^{-2}	7.67×10^{-2}	28.9 deg	7.61×10^{-4}	3.71×10^{-3}	33.7 deg
	10	2.61×10^{-2}	6.93×10^{-2}	28.1 deg	7.32×10^{-4}	4.19×10^{-3}	34.8 deg
	15	2.88×10^{-2}	9.38×10^{-2}	33.9 deg	1.41×10^{-3}	7.97×10^{-3}	34.9 deg
1.5% noise	5	3.26×10^{-2}	7.82×10^{-2}	34.9 deg	1.82×10^{-3}	8.35×10^{-3}	32.3 deg
	10	2.63×10^{-2}	6.52×10^{-2}	28.1 deg	2.48×10^{-3}	8.95×10^{-3}	34.8 deg
	15	2.92×10^{-2}	9.70×10^{-2}	35.0 deg	2.51×10^{-3}	1.16×10^{-2}	34.8 deg

Table 5 Averaged CPU times (s) for the prediction of lift and drag coefficients. The tests are performed on a Linux server with Intel® Xeon® Gold 6148 CPU @ 2.40GHz.

n_s	2D airfoil		3D drone		
	DEIM	NN	n_s	DEIM	NN
5	2.41×10^{-6}	7.89×10^{-5}	5	3.60×10^{-6}	1.19×10^{-4}
8	2.28×10^{-6}	7.05×10^{-5}	10	2.41×10^{-6}	7.89×10^{-5}
10	2.86×10^{-6}	7.30×10^{-5}	15	2.41×10^{-6}	7.89×10^{-5}

hot-film flow speed sensors,” *Smart Materials and Structures*, Vol. 16, No. 4, 2007, pp. 1239–1245. <https://doi.org/10.1088/0964-1726/16/4/035>, URL <https://dx.doi.org/10.1088/0964-1726/16/4/035>.

- [17] Shen, H., Xu, Y., and Remeikas, C., “Pitch control of a micro air vehicle with micropressure sensors,” *Journal of Aircraft*, Vol. 50, No. 1, 2013, pp. 239–248. <https://doi.org/10.2514/1.c031894>, URL <https://arc.aiaa.org/doi/10.2514/1.C031894>.
- [18] Magar, K. T., Reich, G. W., Kondash, C., Slinker, K., Pankonien, A. M., Baur, J. W., and Smyers, B., “Aerodynamic parameters from distributed heterogeneous CNT hair sensors with a feedforward neural network,” *Bioinspiration & Biomimetics*, Vol. 11, No. 6, 2016, p. 066006. <https://doi.org/10.1088/1748-3190/11/6/066006>, URL <https://dx.doi.org/10.1088/1748-3190/11/6/066006>.

- [19] Wood, K. T., Araujo-Estrada, S., Richardson, T., and Windsor, S., “Distributed pressure sensing–based flight control for small fixed-wing unmanned aerial systems,” *Journal of Aircraft*, Vol. 56, No. 5, 2019, pp. 1951–1960. <https://doi.org/10.2514/1.C035416>, URL <https://arc.aiaa.org/doi/10.2514/1.C035416>.
- [20] Mark, A., Xu, Y., and Dickinson, B. T., “Review of microscale flow-sensor-enabled mechanosensing in small unmanned aerial vehicles,” *Journal of Aircraft*, Vol. 56, No. 3, 2019, pp. 962–973. <https://doi.org/10.2514/1.c034979>, URL <https://arc.aiaa.org/doi/10.2514/1.C034979>.
- [21] Araujo-Estrada, S. A., and Windsor, S. P., “Aerodynamic state and loads estimation using bioinspired distributed sensing,” *Journal of Aircraft*, Vol. 58, No. 4, 2021, pp. 704–716. <https://doi.org/10.2514/1.c036224>, URL <https://arc.aiaa.org/doi/10.2514/1.C036224>.
- [22] Araujo-Estrada, S. A., Salama, F., Greatwood, C. M., Wood, K. T., Richardson, T. S., and Windsor, S. P., “Bio-inspired distributed strain and airflow sensing for small unmanned air vehicle flight control,” *AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, 2017. <https://doi.org/10.2514/6.2017-1487>, URL <https://arc.aiaa.org/doi/10.2514/6.2017-1487>.
- [23] Gong, H., Chen, Z., Maday, Y., and Li, Q., “Optimal and fast field reconstruction with reduced basis and limited observations: Application to reactor core online monitoring,” *Nuclear Engineering and Design*, Vol. 377, 2021, p. 111113. <https://doi.org/10.1016/j.nucengdes.2021.111113>.
- [24] Zhao, X., Deng, Z. C., and Zhang, W. W., “Sparse reconstruction of surface pressure coefficient based on compressed sensing,” *Experiments in Fluids*, Vol. 63, No. 10, 2022, p. 156. <https://doi.org/10.1007/s00348-022-03505-7>.
- [25] Berkooz, G., Holmes, P., and Lumley, J. L., “The proper orthogonal decomposition in the analysis of turbulent flows,” *Annual Review of Fluid Mechanics*, Vol. 25, No. 1, 1993, pp. 539–575. <https://doi.org/10.1146/annurev.fl.25.010193.002543>.
- [26] Willcox, K., “Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition,” *Computers & fluids*, Vol. 35, No. 2, 2006, pp. 208–226. <https://doi.org/10.1016/j.compfluid.2004.11.006>.
- [27] Chaturantabut, S., and Sorensen, D. C., “Nonlinear model reduction via discrete empirical interpolation,” *SIAM Journal on Scientific Computing*, Vol. 32, No. 5, 2010, pp. 2737–2764. <https://doi.org/10.1137/090766498>.
- [28] Argaud, J. P., Bouriquet, B., De Caso, F., Gong, H., Maday, Y., and Mula, O., “Sensor placement in nuclear reactors based on the generalized empirical interpolation method,” *Journal of Computational Physics*, Vol. 363, 2018, pp. 354–370. <https://doi.org/10.1016/j.jcp.2018.02.050>.
- [29] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S., “PyTorch: An imperative style, high-performance deep learning library,” *Advances in Neural Information Processing Systems*, Vol. 32, Curran Associates, Inc., 2019, p. 1711. [https://doi.org/10.1016/s0140-6736\(19\)32614-5](https://doi.org/10.1016/s0140-6736(19)32614-5), URL <https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html>.

- [30] ESI-OpenCFD, “OpenFOAM,” , dec 2022. URL <https://www.openfoam.com/>.
- [31] Menter, F. R., Kuntz, M., and Langtry, R., “Ten years of industrial experience with the SST turbulence model,” *Turbulence, heat and mass transfer*, Vol. 4, No. 1, 2003, pp. 625–632.
- [32] He, G. S., Deparday, J., Siegel, L., Henning, A., and Mulleners, K., “Stall delay and leading-edge suction for a pitching airfoil with trailing-edge flap,” *AIAA Journal*, Vol. 58, No. 12, 2020, pp. 5146–5155. <https://doi.org/10.2514/1.j059719>, URL <https://doi.org/10.2514/1.J059719>.
- [33] Geuzaine, C., and Remacle, J.-F., “Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities,” *International Journal for Numerical Methods in Engineering*, Vol. 79, No. 11, 2009, pp. 1309–1331. <https://doi.org/10.1002/nme.2579>.
- [34] Zhao, X., Du, L., Peng, X. H., Deng, Z. C., and Zhang, W. W., “Research on refined reconstruction method of airfoil pressure based on compressed sensing,” *Theoretical and Applied Mechanics Letters*, Vol. 11, No. 2, 2021, p. 100223. <https://doi.org/10.1016/j.taml.2021.100223>, URL <https://www.sciencedirect.com/science/article/pii/S2095034921000283>.