ASME Journal of Dynamic Systems, Measurement, and Control
Online journal at:
https://asmedigitalcollection.asme.org/dynamicsystems

Localization of Upstream Obstacles by Learning From Spectra of the Koopman Operator

Colin Rodwell
Department of Mechanical Engineering,
Clemson University,
Clemson, SC 29634

Phanindra Tallapragada
Department of Mechanical Engineering,
Clemson University,
Clemson, SC 29634
e-mail: ptallap@clemson.edu

Objects moving in water or stationary objects in streams create a vortex wake. An underwater robot encountering the wake created by another body experiences disturbance forces and moments. These disturbances can be associated with the disturbance velocity field and the bodies creating them. Essentially, the vortex wakes encode information about the objects and the flow conditions. Underwater robots that often function with constrained sensing capabilities can benefit from extracting this information from vortex wakes. Many species of fish do exactly this, by sensing flow features using their lateral lines as part of their multimodal sensing capabilities. Besides the necessary sensing hardware, a more important aspect of sensing is related to the algorithms needed to extract the relevant information about the flow. This paper advances a framework for such an algorithm using the setting of a pitching hydrofoil in the wake of a thin plate (obstacle). Using time series pressure measurements on the surface of the hydrofoil and the angular velocity of the hydrofoil, a Koopman operator is constructed that propagates the time series forward in time. Multiple approaches are used to extract dynamic information from the Koopman operator to estimate the plate position and are bench marked against a state-of-the-art convolutional neural network (CNN) applied directly to the time series. We find that using the Koopman operator for feature extraction improves the estimation accuracy compared to the CNN for the same purpose, enabling "blind" sensing using the lateral line. [DOI: 10.1115/1.4066009]

Keywords: estimation, system identification, underwater robotics

1 Introduction

The locomotion of fish and other aquatic swimmers has many desirable characteristics such as energy efficiency, agility, and stealth [1,2], which have inspired mimicry in bioinspired robots [3,4]. Closely related to and aiding the locomotion is the ability of fish to sense and process the spatiotemporal information in the water around them. Objects moving in water or stationary objects in streams create a vortex wake. An underwater robot encountering the wake created by another body experiences disturbance forces and moments. These disturbances can be associated with the disturbance velocity field and the bodies creating them. Essentially, information about fluid flow and the objects that create these flows is encoded in the spatiotemporal evolution of the vortical structures, whether the bodies creating them are cylinders, hydrofoils, underwater robots, or fish. Underwater robots that often function with constrained sensing capabilities can benefit from extracting this information from vortex wakes. Many species of fish do exactly this, by sensing flow features using their lateral lines as part of their multimodal sensing [1,5,6].

The complexity and high (infinite) dimensionality of fluid flows around a swimmer present significant challenges to emulate fish-like hydrodynamic sensing and extract the relevant information from sensor data of the flow. This particular challenge is not restricted to bioinspired fish-like swimmers, but has been present in the broad areas of fluid flow estimation, model reduction, and active flow control. Proper orthogonal decomposition (POD) [7,8] and gappy POD [9] have been tools for model reduction in turbulent flows for decades and have also been applied for unsteady flow sensing past an hydrofoil and estimation of surface pressure [10,11]. Model reduction of complex flows using the Koopman operator approach has extended the POD approach to a dynamical systems framework [12,13]. Subsequent developments in the application of machine learning in dynamical systems have created algorithms for learning the dynamic modes or Koopman modes of a dynamical system from often sparse data [14–16]. Similar methods combining machine learning with dynamical systems are increasingly playing an important role in model reduction in fluid mechanics, flow reconstruction, and flow classification, see, for instance, Refs. [17–26]. Flow estimation in the near field of a body by selecting from known fluid modes by the dynamic mode decomposition (DMD) using surface pressure data has been studied in Ref. [27], and incorporating traditional filtering into this approach was recently shown to allow updating the estimation in real-time [28,29].

This paper considers a different but related problem motivated by underwater robots where on-board sensors such as inertial motion units and pressure sensors can measure only dynamic and kinematic variables of the robot itself or pressure on the surface of the robot but not measure the ambient pressure and velocity fields. We consider the problem of the estimation of the spatial location of an upstream obstacle in a flow past a pitching hydrofoil. It is assumed that pressure on the surface of the hydrofoil can be measured at a small number of fixed locations on the body. This is related to the problems considered in Refs. [30] and [31], where pressure sensors were used to localize a body and a dipole, respectively, by assuming potential

Contributed by the Dynamic Systems Division of ASME for publication in the Journal of Dynamic Systems, Measurement, and Control. Manuscript received December 7, 2023; final manuscript received July 7, 2024; published online August 17, 2024. Assoc. Editor: Efstathios Bakolas.

flow which allows for the application of analytical methods. However, frequently fluid–structure interaction is driven by significant viscous effects including vortex shedding which make purely potential flow models inaccurate. Vortex induced vibrations and coupled fluid–structure models cannot in general be described by simple mathematical models. This motivates data-driven approaches, such as the one introduced in Ref. [32] to localize a source in three dimensions. However, in that work, only a single time snapshot of velocity data is used to perform each estimation, which is possible because the flow was steady except for the movement of the source; for the fluid–structure interaction problem considered in this paper, using a time series of pressure data is necessary due to the unsteady wake generated by the both obstacle and the trailing hydrofoil itself. The field of soft sensing [33] has techniques for regression from time series data, which typically resolve the high-dimensionality of the time series by taking as features its statistical moments (such as mean and variance) or data from a few selected time instants [34]. This approach discards much of the dynamic information from the data, which is often acceptable for systems that are roughly steady, but is a significant drawback when applied to an unsteady fluid system. Based on the success of modal approaches for extracting information about the underlying dynamics of fluid data, we propose a method that extracts information about the system dynamics from the pressure time series, and uses that information as features to identify the location of the obstacle. Using time series pressure measurements on the surface of the hydrofoil, a Koopman operator is constructed that propagates the snapshots of pressure data forward in time, thereby encoding the system dynamics. Multiple approaches are considered to extract the encoded information for use in estimating the position of an upstream obstacle. These include the "direct mode estimation" approach, where the most important modes (eigenvectors) of the operator are input into a dense neural network (DNN), the "spectral image estimation" approach where the spectrum (or the singular vectors) of the operator is extracted and input into a convolutional neural network (CNN), and the "mode-kernel estimation" approach, where the modes constructed from training data are compared to a dictionary of known modes. This is benchmarked against the time CNN [35], a recent black-box CNN architecture designed for classification of multivariate time series, in the "CNN-based estimation" approach.

The remainder of the paper is organized as follows. In Sec. 2, we define the exact fluid-interaction problem considered and discuss its implementation in simulation. In Sec. 3, we review the theory behind standard and exact DMD and its connection to the Koopman operator, and their relevance to the estimation problem are explained in Sec. 4. The training speed and accuracy of the estimation methods are investigated on the training data in Sec. 5.

2 Problem Definition

We consider the problem of flow past a symmetric NACA-0018 hydrofoil of unit chord length, representing a streamlined swimmer, pinned at its leading edge with a linear spring of stiffness $k = 6$ N·m/rad and damping coefficient $c = 2$ N·m s/rad. When the spring is at rest, the hydrofoil is horizontal with the leading edge pointing left. The hydrofoil is immersed in a freestream flow with velocity $U_\infty = 10$ m/s. The fluid is of unit density (equal to that of the hydrofoil) and has viscosity is $\nu = 0.001$ m^2/s. A rectangular bluff body of unit height and width 0.1 m is placed upstream of the hydrofoil and disturbs the incoming freestream flow by shedding and unsteady wake in the fluid. This disturbed flow interacts with the downstream hydrofoil, inducing angular motion and a time-varying pressure profile on the surface. The relative position of the two bodies is defined by the tuple (b, d) as shown in Fig. 1. Along the body, N_s pressure sensors are placed that are evenly spaced in the horizontal direction and record the absolute pressure at a frequency of 40 Hz.

The problem considered in this paper is the estimation of the parameters (b, d) using only measurements of pressure made at the N_s locations on the hydrofoil.

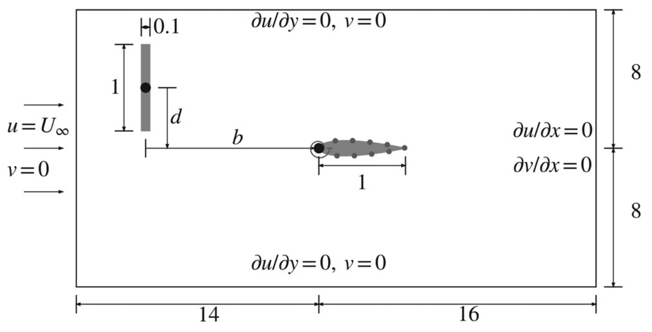


Fig. 1 A hydrofoil is pinned with a torsional spring at its leading edge downstream from a vertical plate, with relative position parameterized by *b* and *d*. On the hydrofoil, N_s=10 pressure sensors are placed evenly whose locations are shown by the filled circles. The figure is not to scale.

This flow simulation domain D is a rectangle of width 30 m and height 16 m. The leading edge of the hydrofoil defines the origin of the rectangular domain and is located 1 m to the left of and vertically collocated with the domain's center. The center of the rectangular obstacle is placed b m to the left and d m above the origin. At the left boundary of the domain, the horizontal velocity, $u = 20$ m/s, and a vertical velocity, $v = 0$ m/s, are imposed. To allow unimpeded exit of the flow at the right boundary, a zero gradient condition is imposed on the velocity, so $\nabla u = \nabla v = \mathbf{0}$. At the top and bottom of the domain, a "slip" condition enforces that no flow passes through the boundary ($v = 0$) and that there is no shear force at the boundary ($\partial u / \partial y = 0$). On the walls of the plate and hydrofoil, a no-slip condition ensures that the velocity of the flow relative to the bodies is zero at the surface.

The system is simulated in the open-source computational fluid dynamics software OPENFOAM 9. The fluid is modeled by the incompressible two-dimensional Navier–Stokes equations

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\nabla^2\mathbf{u} \tag{1}$$

$$\nabla\cdot\mathbf{u} = \mathbf{0} \tag{2}$$

These are numerically solved with the pressure-implicit with splitting of operators algorithm [36] over finite volumes of fluid. Turbulence is modeled by a large eddy simulation with $k-\omega$ shear stress transport. The fluid domain is discretized with a finite volume square mesh, as shown in Fig. 2. The mesh consists of two identical two-dimensional meshes stacked on top of each other, resulting in a single layer of volumes. Two layers of mesh refinement are used to improve the simulation fidelity in the region between the bodies and

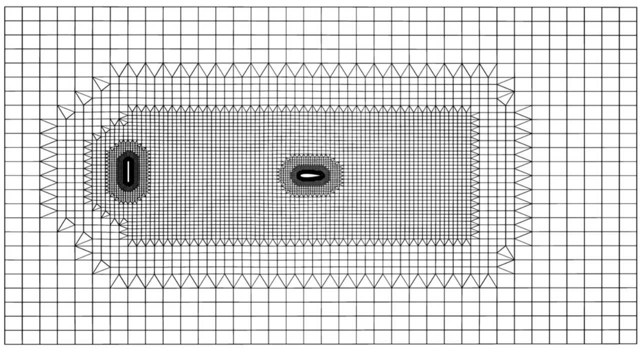


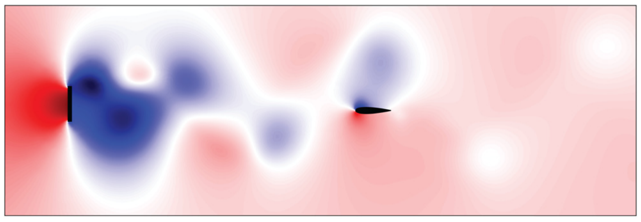
Fig. 2 The computational mesh, with a high density in important flow regions near surfaces and between the bodies

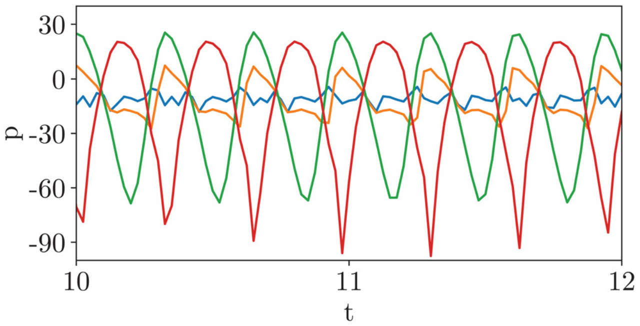
their immediate wake. A further three layers of refinement are applied to capture the flow near the surface. This refinement results in $N = 15,493$ mesh grid points with 80 of these on the surface of the hydrofoil. The moment f_m on the hydrofoil about the pin is calculated by integrating the pressure on its surface. This moment then allows solving the hydrofoil equations of motion

$$I_p\ddot{\theta} + c\dot{\theta} + k\theta = f_m \tag{3}$$

where θ is the angle of the hydrofoil, and I_p is the mass moment of inertia of the hydrofoil about the pin. Through the forcing term and a surface boundary condition, this equation is coupled with the fluid equations (1) and (2), resulting in complex fluid–body interaction. This interaction is solved by iteratively solving the flowfield and body acceleration until convergence is achieved. These states are then integrated forward using the forward Euler method with an adaptive time-step Δt selected such that the Courant number $C < 0.85$ and $\Delta t \leq 0.02$. This is repeated until time $t = 50$ s. When the hydrofoil rotates, the mesh points in a region of 0.3 m from its surface rotate rigidly with it, while the mesh points greater than 1 m from the surface remain fixed.

The pressure field from a sample simulation is shown in Fig. 3(*a*). Regions of alternating low pressure to the right of the flat plate show a $2S$ vortex street. As the vortices interact with the hydrofoil, a high-pressure region is generated that, when combined with the low pressure inside the vortex, generates a net moment on the hydrofoil and thus motion. This pattern repeats in time, symmetrically on both sides of the hydrofoil for $d = 0$ and asymmetrically for $d \neq 0$. The time-varying pressure measured at four of the $N_s = 10$ sensor locations is shown Fig. 3(*b*). The nonzero value of d results in an asymmetrical pressure profile in time: pressure measured at the top center and bottom center of the hydrofoil show differences in temporal evolution besides the obvious phase difference.


(*a*)



(*b*)

Fig. 3 (*a*) The pressure field for a test case where b=8.15 and d=0.18 after the transient period, where red indicates high pressure and blue low pressure. The formation of a vortex wake is evident in the pressure field. (*b*) The pressure at four points in the same simulation on the hydrofoil over time, with blue indicating the trailing edge, orange the top center, green the leading edge, and red the bottom center and blue indicates low pressure (behind the plate and hydrofoil). (Color version online).

3 Dynamic Mode Decomposition

Consider the bodies B_l (leading bluff body) and B_t (trailing hydrofoil) immersed in a two-dimensional fluid flow. We assume that the system has no explicit time dependence, i.e., the states, including the pitch of the hydrofoil, evolve with time, but the underlying dynamic system is time invariant or autonomous. While the flow of the fluid is governed by the Navier–Stokes equation, we will consider the spatially and temporally discretized solution to the boundary value problem. The flow domain is discretized by N grid points $(x_i, y_i) \in \mathbb{R}^2$ with sample times $t \in \{0, \Delta t, 2\Delta t, \ldots, N_t\Delta t\}$. The numerical solutions in Sec. 2 provide the velocity field (u, v) and the pressure field p and the dynamical system $\dot{x} = u(x, y; b, d)$, $\dot{y} = v(x, y; b, d)$ that depends on the parameters b and d. We will consider a time discretization of this dynamical system governed by a flow map F as

$$z_{n+1} = F(z_n) \tag{4}$$

where $z_n = (x(t_n), y(t_n)) \in \mathcal{M} \cong \mathbb{R}^2$ denotes the state at time $t = n\Delta t$. Next, consider $g \in L^2(\mathcal{M})$, the space of square integrable real scalar valued functions. Koopman showed in Ref. [37] that an observable function $g : \mathcal{M} \mapsto \mathbb{R}$ can be propagated by a linear operator, $\mathcal{K} : L^2(\mathcal{M}) \mapsto L^2(\mathcal{M})$ as

$$g(z_{n+1}) = \mathcal{K}g(z_n) := g \circ F(z_n) \tag{5}$$

see Refs. [38] and [39] for a review. The operator $\mathcal{K}$ is linear and known as the Koopman operator.

The scalar valued observable function of interest in the problem is the pressure field, $p(x, y)$. The pressure at the N mesh points at time $t_n = n\Delta t$ is denoted by $h_f(t_n) = p(x_i, y_i, t_n) \in \mathbb{R}^N$. A subset of h_f measured at the pressure sensors on the hydrofoil (shown in Fig. 1) located at the mesh points (ξ_i, η_i) is denoted by h_s. Its snapshot at time t_n is $h_s(t_n) = p(\xi_i, \eta_i, t_n) \in \mathbb{R}^{N_s}$. As a result, the snapshots $h_f(t)$ also contain the snapshots $h_s(t)$. We define two operators K_f and K_s such that $h_f(t_{n+1}) = K_f h_f(t_n)$ and $h_s(t_{n+1}) = K_s h_s(t_n)$. The matrices K_f and K_s are the representations of Koopman operator $\mathcal{K}$ composed with orthogonal projections on spaces spanned by elements of h_f and h_s, respectively.

We use the DMD algorithm to construct an approximate operator K_s that propagates the snapshots of h_s. Consider first m time snapshots of the surface pressure measurements $p(\xi_i, \eta_i)$, which are few in number. We construct two matrices $X_s \in \mathbb{R}^{N_s \times m}$ and $Y_s \in \mathbb{R}^{N_s \times m}$ of these measurements

$$X_s = \begin{bmatrix} p(\xi_i, \eta_i, t_n) & p(\xi_i, \eta_i, t_{n+1}) \cdots p(\xi_i, \eta_i, t_{n+m-1}) \end{bmatrix} \tag{6}$$

$$Y_s = \begin{bmatrix} p(\xi_i, \eta_i, t_{n+1}) & p(\xi_i, \eta_i, t_{n+2}) \cdots p(\xi_i, \eta_i, t_{n+m}) \end{bmatrix} \tag{7}$$

The operator K_s that propagates h_s is then given by

$$K_s = \underset{K}{\operatorname{argmin}} \, \| Y_s - KX_s \|_F^2 \tag{8}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. When the measurements are sufficiently large and richly sampled, the dominant spatiotemporal modes of the system are identified by K_s [12,40,41]. Essentially, as the number of sampling points increases, the Koopman operator calculated by DMD or the extended dynamic mode decomposition would converge to the orthogonal projection of the action of the Koopman operator acting on the space of the observables chosen as basis functions, see, for example, Ref. [41] for a discussion. We note that other errors besides $\| Y_s - KX_s\|_F^2$ can be used to calculate K_s. For example, instead of the average error considered in Eq. (8), one may prioritize having a tight upper bound on the error as in Ref. [42]. When the argument of Eq. (8) is zero, K_s exactly propagates the snapshots h_s forward in time. This problem has a convex solution

$$K_s = Y_s X_s^+ \tag{9}$$

where + represents the Moore–Penrose pseudoinverse. When the rank of X_s is greater than or equal to the number of columns of K, this solution is unique. The modes of the operator K_s are then calculated as the eigenvectors and eigenvalues of K_s

$$K_s\Phi = \Lambda\Phi \tag{10}$$

where

$$\Phi = \begin{bmatrix} \phi_1 & \phi_2 & \cdots & \phi_{N_s} \end{bmatrix} \tag{11}$$

is the matrix of the eigenvectors and

$$\Lambda = \mathrm{diag}\begin{bmatrix} \lambda_1 & \lambda_2 & \cdots & \lambda_{N_s} \end{bmatrix} \tag{12}$$

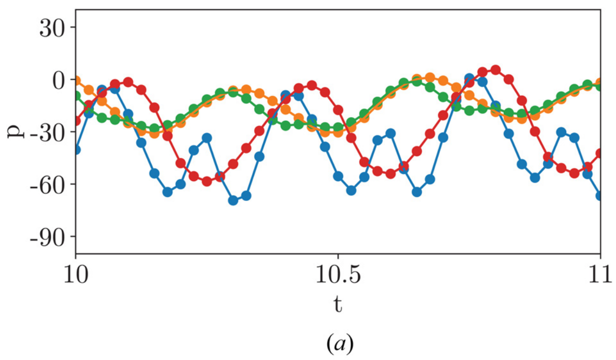
the diagonal matrix of eigenvalues. By convention, the L^2 norm of each component of Φ is unity, $||\phi_i||_2 = 1$. The modes and eigenvalues of the operator K_s can be used to reconstruct the pressure field on the surface of the cylinder m time-steps after an initial data snapshot $h_s(t_n)$ by

$$h_s(t_{m+n}) \approx \Phi\Lambda^m\Phi^{-1}h_s(t_n) = \Phi\Lambda^m\boldsymbol{\alpha} \tag{13}$$

where

$$\boldsymbol{\alpha}_f = \Phi^{-1}h_s(t_n) = \begin{bmatrix} \alpha_{f,1} & \alpha_{f,2} & \cdots & \alpha_{f,N_s} \end{bmatrix} \tag{14}$$

is a vector of complex numbers defining the relative magnitude of the modes, as well as their relative phases. The magnitude of α_i is roughly equivalent to the concept of "mode energy" in POD, and in the case that most of the "energy" is concentrated in a small number of modes, the flow can be reconstructed with high accuracy using only those few modes. Similarly, neglecting modes with small corresponding values of α causes only minimal reconstruction error. This reconstruction ability is demonstrated in Fig. 4, where an operator is generated using data from time 10 s to 13.75 s, and reconstructs the pressure starting at a time of 10 s. Compared to the actual pressure, the reconstructed pressure is qualitatively correct, with noticeable error only emerging in the high frequency component of the dynamics. This reconstruction is nearly the same even for



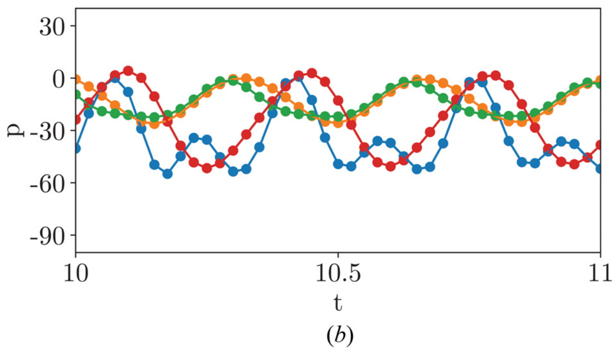


Fig. 4 **(*a*) Actual pressure over time for *b*=5.31 and *d*=0.26. (*b*) The pressure predicted by the operator calculated based on Eq. (13), where t_n=10 s and N_s=10.**

$N_s = 100$, implying that a small number of measurements capture many of essential features of the pressure distribution.

While this approach, often referred to as "exact DMD" [40], is applicable for the surface measurements because N is small, applying this method on the field data would be computationally intractable because of the size of K_f and the resulting complexity of calculating its eigenvalues. For the field measurements, we use "standard DMD," which introduces a degree of truncation to greatly increase the speed of the calculation. Shifted data matrices are first constructed similarly to Eq. (7)

$$X_f = \begin{bmatrix} p(x_i, y_i, t_n) & p(x_i, y_i, t_{n+1}) \cdots p(x_i, y_i, t_{n+m-1}) \end{bmatrix}$$
$$Y_f = \begin{bmatrix} p(x_i, y_i, t_{n+1}) & p(x_i, x_i, t_{n+2}) \cdots p(x_i, y_i, t_{n+m}) \end{bmatrix}$$

Instead of directly calculating K_f using the pseudoinverse, first the singular value decomposition is computed

$$X_f = U\Sigma V^* \tag{15}$$

where U and V are the left and right singular matrices, respectively, and Σ is a diagonal matrix containing the singular values. To improve computation speed, Σ can be truncated starting with its smallest singular values to a smaller $r \times r$ matrix Σ_t, and the corresponding columns of U and V can be removed, resulting in truncated matrices U_t and V_t, respectively. This allows approximating the matrix K_f (in a reduced-dimensional space) as

$$\tilde{A} = U_t^* Y_f V_t \Sigma_t^{-1} \tag{16}$$

and its eigenvalues and eigenvectors are computed as

$$\tilde{A}\Psi_r = \Lambda_f \Psi_r \tag{17}$$

The eigenvectors in the reduced space can be projected back to the full space using the truncated left singular matrix

$$\Psi = U_t\Psi_r \tag{18}$$

where this Ψ physically corresponds to modes of the fluid field

$$\Psi = \begin{bmatrix} \psi_1 & \psi_2 & \cdots & \psi_r \end{bmatrix} \tag{19}$$

Using a known initial condition, it is again possible to reconstruct the flow field using these truncated modes as

$$h_f(t_{n+m}) \approx \Psi\Lambda_f^m\Psi^{-1}h_f(t_n) = \Psi\Lambda^q\boldsymbol{\alpha}_f \tag{20}$$

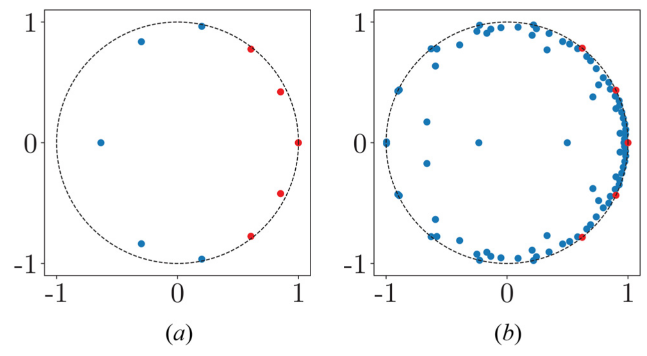
where

$$\boldsymbol{\alpha}_f = \Psi^{-1}h_f(t_n) = \begin{bmatrix} \alpha_{f,1} & \alpha_{f,2} & \cdots & \alpha_{f,r} \end{bmatrix} \tag{21}$$

and $h_f(t_n)$ is a snapshot of the flow field at $t = n\Delta t$.

3.1 Dominant Pressure Modes.

Modal analysis of flows reveals that a few modes contain a very large fraction of the energy [40]. An analysis of the modes of the operators K_f and K_s reveal the same. We sort the modes of K_f and K_s such that they are labeled in descending order of magnitude, $|\alpha_{f,1}| \geq |\alpha_{f,2}| \ldots \geq |\alpha_{f,r}|$ and $|\alpha_{s,1}| \geq |\alpha_{s,2}| \ldots \geq |\alpha_{s,N_s}|$, respectively. The components of Ψ and Λ_f are also sorted to be in the same order as $\boldsymbol{\alpha}_f$, and the components of Φ and Λ_s are sorted in the same order as that of $\boldsymbol{\alpha}_s$. Figures 5(*a*) and 5(*b*) show the eigenvalues Λ_s and Λ_f on the complex plane. Many of the eigenvalues Λ_f are inside the unit circle (Fig. 5(*b*)) indicating that the corresponding modes decay. The rest of the eigenvalues on the unit circle occur as complex conjugates with the corresponding modal magnitudes α_f and α_s also occurring as complex conjugate pairs. The small number of surface pressure measurements used introduces approximation error that causes slight deviations in the



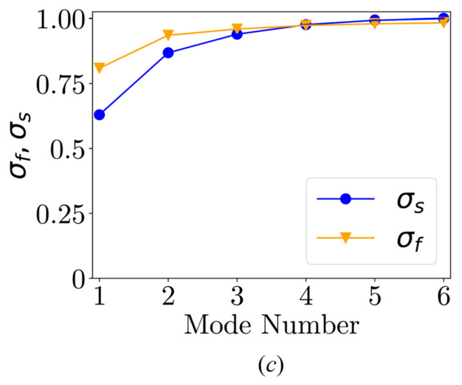


Fig. 5 Eigenvalues (*a*) Λ_s and (*b*) Λ_f of the modes on the complex plane from the simulation for a representative (b, d), where the 3 corresponding to the highest magnitudes of α, along with their complex conjugates, are represented by red circles. (*c*) σ_f (blue circles) and σ_s (red triangles), with $\sigma_f(3) > 0.93$ and $\sigma_s(3) > 0.93$. (Color version online).

magnitude of the eigenvalues of the even second dominant modes, which lie close to the boundary of the unit circle (in Fig. 5(*a*)). However, these modes decay very slowly and for short time intervals persist in being one of the dominant modes. The first eigenvalues $\lambda_{f,1}$ and $\lambda_{s,1}$ are both real. The corresponding modes ψ_1 and ϕ_1 and modal coefficients $\alpha_{f,1}$ and $\alpha_{s,1}$ are real. A measure of the sparsity of the modes containing most of the information of flow is revealed by the ratios

$$\sigma_f(n) = \frac{\sum_{i=1}^{n} |\alpha_{f,2i-1}|}{\sum_{i=1}^{r} |\alpha_{f,i}|}, \quad n \leq r/2 \tag{22}$$

$$\sigma_s(n) = \frac{\sum_{i=1}^{n} |\alpha_{s,2i-1}|}{\sum_{i=1}^{N_s} |\alpha_{s,i}|}, \quad n \leq \frac{N_s}{2} \tag{23}$$

Figure 5(*c*) shows a plot of $\sigma_s(n)$ and $\sigma_f(n)$, revealing that $\sigma_s(3) > 0.9$ and $\sigma_f(3) > 0.9$, i.e., for the first three modes of both the fluid field pressure and surface pressure contain much of the information about the respective fields. The first three modes ψ_1, ψ_2, and ψ_3 of the pressure field in the fluid domain are shown in Figs. 6(*a*)–6(*c*) which also reveal the physical features of the flow. The first mode ψ_1 corresponds to the pressure field of the steady flow, with the next two modes identifying changes in pressure due to the vortex shedding past the obstacle (leading bluff body). The eigenvalues Λ_s and Λ_f (in Figs. 5(*a*) and 5(*b*)) corresponding to the dominant modes have similar phases: one with a phase of zero, one with a phase near 0.45 (corresponding to the vortex shedding frequency), and another with a phase near 0.90 (corresponding to the second harmonic of the vortex shedding). The modes ϕ_1, ϕ_2, and ϕ_3 for the surface pressure on the hydrofoil are also shown in Figs. 6(*d*)–6(*f*) for the case where $N_s = 10$.

4 Parameter Estimation

We use four different approaches all based on supervised learning to the problem of estimating the parameters (b, d). The first of these is based on using a convolutional neural network (the "TIME" CNN [43]) which takes as input the time series pressure measurements at the N_s locations on the trailing hydrofoil. This approach sets a

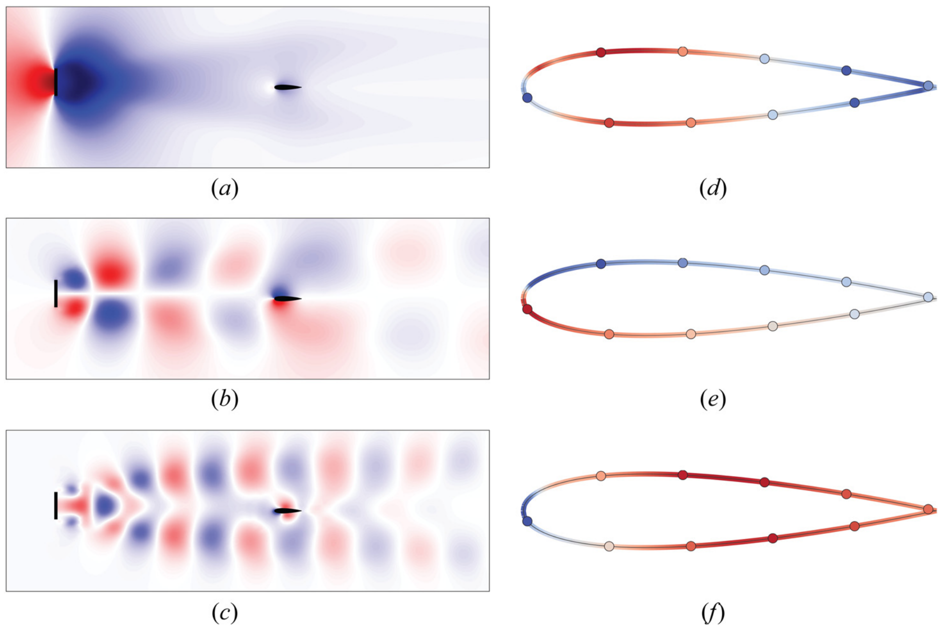


Fig. 6 (*a*)–(*c*) The first three modes of pressure field in the fluid domain. Due to the phase of modes being arbitrary, the red (dark) and blue (light) colors are interchangeable, and white signifies a value of zero. (*d*)–(*e*) The first three modes of the surface pressure field. Parameter values are b=8.15 and d=0.18. (Color version online).

benchmark that uses one of the latest machine learning methods for time series data. The next three approaches are based on combining calculations of DMD with machine learning.

Supervised learning necessitates collecting multiple sets of simulations with nonoverlapping values of (b, d): one set to train the estimator (the "training" set), one set to validate how well trained the estimator is (the "validation" set), and one set to test the accuracy of the resulting estimator (the "testing" set). The parameter values used for each dataset are shown in Fig. 7. The training set has parameters on an even 6×6 grid ($a = 36$ points total) filling the region $4 \leq b \leq 9$ and $0 \leq d \leq 1$, which is selected because the wake in that region is well-developed and has not fully dissipated. The even spacing of these training points is designed to sample the entire space. However, in practical applications, the parameters would likely not fall on a grid, but rather be scattered throughout the acceptable parameter space. For this reason, the validation and testing data are selected from a uniform probability distribution in the region $4 \leq b \leq 9$ and $0 \leq d \leq 1$, with $c = 20$ points for validation and $e = 38$ points for testing.

From each simulation, pressure is known at $N = 15{,}493$ points in the domain, with 80 of those points on the surface of the body. The pressures are interpolated with a spline function to determine the pressures at ten evenly spaced points, which represent physical pressure sensors and is the only data from the simulation for the parameter estimation, in training, validation, and testing.

Suppose the pressure from the measurements from the surface of the hydrofoil for the kth pair of parameters (b_k, d_k) at time t_j is denoted by $p_k(\xi_i, \eta_i, t_j)$. Data from the simulations are stored in three different sets

$$\mathbb{T} = \{T^1, T^2, \ldots, T^a\} \tag{24}$$

$$\mathbb{V} = \{V^1, V^2, \ldots, V^c\} \tag{25}$$

$$\mathbb{E} = \{E^{a+c+1}, E^{a+c+2}, \ldots, E^e\} \tag{26}$$

where T^k is a matrix corresponding to the kth pair of training parameters, (b_k, d_k), and $T^k_{ij} = p_k(\xi_i, \eta_i, t_j)$ is the pressure at the ith pressure sensor and jth time interval. Similarly, the kth validation dataset V^k is the matrix $V^k_{ij} = p_{(k+a)}(\xi_i, \eta_i, t_j)$, and the kth test (or evaluation) dataset E^k is the matrix $E^k_{ij} = p_{(k+a+c)}(\xi_i, \eta_i, t_j)$, respectively. We define a sampling function S which acts on these sets and returns a random window of 150 time-steps of continuous data from a random element, as well as the parameters (b, d) at that element. For instance, $S(\mathbb{T}) = (T^k_{ij}, b(k), d(k))$, where k is a random variable uniformly distributed over the set $\{1, 2, \ldots, 36\}$, i is the range of integers where $1 \leq i \leq 10$, and j is a range $\{j_0, j_0 + 1, \ldots, j_0 + 150\}$ such that j_0 is a uniformly distributed random integer in the range $t_0 \leq j_0 \leq t_f - 150$. The initial time-step is chosen as $t_0 = 400$ which allows transient flow conditions to decay, and the final time is selected as $t_f = 1000$. Therefore, each matrix of data in training, validation, and test sets have 150 time-steps of data with the length of each time-step being $\Delta t = 0.025$ s.

The estimation problem considered here is to construct an estimator that predicts (b, d) as

$$(b_p, d_p) = \mathcal{D}_\theta(\mathcal{F}_\omega(P)) \tag{27}$$

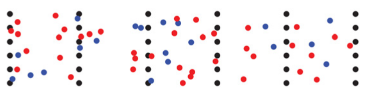


Fig. 7 Positions of the pinned leading edge of the hydrofoil relative to the upstream plate for training (black circles spaced at uniform horizontal spacing in vertical lines), validation (blue or light colored), and testing (red or dark colored) data sets. Training points are on a grid for even coverage of the parameter space, while validation and testing points are placed randomly. (Color version online).

$$(P, b, d) = S(\mathbb{T}) \tag{28}$$

where $\mathcal{F}$ is a function that extracts features from the time series, and $\mathcal{D}$ is a function that maps those features to an estimate of (b, d). Both θ and ω are sets of parameters (referred to as "weights") that support those functions. The objective is to select the parameters θ and ω such that the expected value of the mean squared error (L) in the estimation is minimized, or

$$\min_{\theta,\omega} L_e \text{ where } L_e = \boldsymbol{E}[(b_p - b)^2 + (d_p - d)^2)] \tag{29}$$

Below, four different architectures for the feature extractor $\mathcal{F}$ are introduced, two of which are nonparametric (so $\omega = \varnothing$). After optimizing the parameters for each, the overall estimation error values are investigated to determine which feature extractor is most effective. To isolate the effect of changing $\mathcal{F}$, the architecture of $\mathcal{D}$ is kept the same for all feature extractors, though θ is re-optimized for each $\mathcal{F}$.

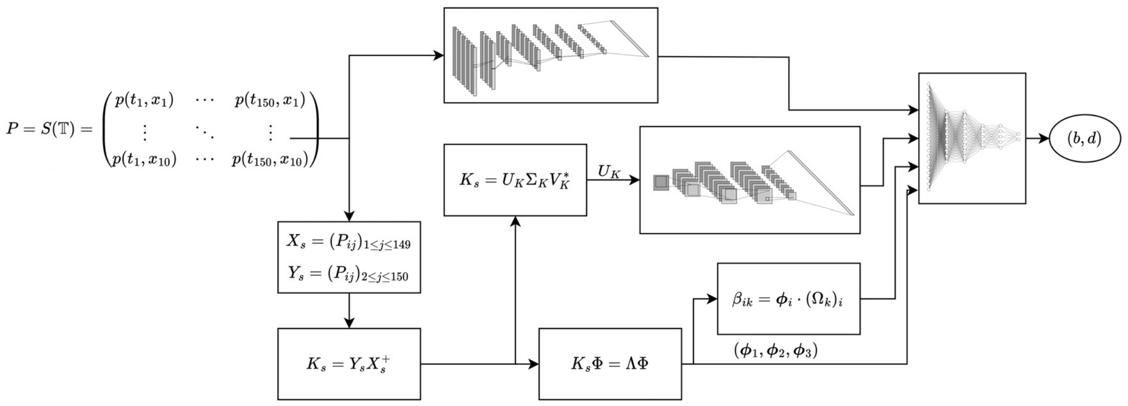
The function $\mathcal{D}$ is a DNN, selected due to their general utility in estimation problems. It is fully connected and uses "Rectified linear unit" activation, with sequential layers containing 200, 100, 100, 50, and 50 nodes. The output layer has two nodes, representing estimates of b and d, and has a linear activation function.

4.1 Convolutional Neural Network-Based Estimation.

For time series classification problems, neural network-based parametric approaches have been found to have performance comparable to common nonparametric approaches [43]. The best suited networks for this task are recursive neural networks such as long short-term memory networks and CNNs, which extract features from time series data by repeatedly applying a single-dimensional kernel. We use a CNN as a benchmark feature extractor, using the hyperparameters determined to be optimal for time series classification in Ref. [35], a network architecture known as the "time CNN," which has been benchmarked against other time series classification algorithms in Ref. [43]. Though we consider estimation problems instead of classification, estimation can in a sense be viewed as a subset of parameter classification, as simply averaging the classification probability distribution on different "bins" of parameter values yields an estimate. As a result, we expect a network architecture designed for classification to be effective at estimation. In summary, the architecture requires two layers of alternating one-dimensional convolutional neural networks with sigmoid activation and average pooling operations. The filters have length 7, and the pooling operations operate on groups of three values, reducing the dimension of the latent space by roughly a factor of 3 with every application. No padding is used on the convolutional steps. The single input time series vector is split into six vectors using six separate filters at the first step, and this is increased to 12 vectors for the second step. After the final pooling operation, the vectors are concatenated into a single feature vector of length 168, which serves as the output of $\mathcal{F}$. This procedure is visualized in the top row of Fig. 8.

4.2 Direct Mode Estimation.

Though the CNN can extract features from dynamic data, it uses a black-box approach that does not reveal any specific information about what dynamics it has identified. By contrast, the highest magnitude Koopman modes of dynamic data also form a reduced basis of the flow dynamics, which may also serve as features for further estimation. Further, an association between the local modes of the surface pressure and the modes of the entire fluid field are visually obvious, as shown in Figs. 6(*d*)–6(*f*). The surface pressure modes in Figs. 6(*d*)–6(*f*) calculated from $N_s = 100$ observations are shown as a continuous mode, and their values at the smaller number $N_s = 10$ of sensor locations are shown as circles. Despite the different quantities of observations, these modes are qualitatively very similar. Because the modes of the larger flow field are clearly strongly affected by (b, d), this motivates the possibility that significant information



$P = S(\mathbb{T}) = \begin{pmatrix} p(t_1,x_1) & \cdots & p(t_{150},x_1) \\ \vdots & \ddots & \vdots \\ p(t_1,x_{10}) & \cdots & p(t_{150},x_{10}) \end{pmatrix}$
$X_s = (P_{ij})_{1\leq j\leq 149}$
$Y_s = (P_{ij})_{2\leq j\leq 150}$
$K_s = Y_s X_s^+$
$K_s = U_K \Sigma_K V_K^*$
U_K
$K_s\Phi = \Lambda\Phi$
$\beta_{ik} = \phi_i \cdot (\Omega_k)_i$
(ϕ_1, ϕ_2, ϕ_3)
(b,d)

Fig. 8 A schematic of the estimation methods. The top row illustrates the CNN-based estimation approach. The next row down shows spectral image estimation approach. The second to bottom row shows the mode-kernel estimation approach, and the bottom row shows the direct mode estimation approach.

about (b,d) has been passed from the larger fluid modes to the pressure modes on the surface, which can be evaluated from a sparse set of observations.

Because the dominant three modes have been demonstrated to contain the majority of the information needed to reconstruct the flow, they are selected as features of the data, to be input into the DNN which performs the final estimation. However, to ensure performance, they must first be standardized. This takes two forms: standardizing their order and standardizing their phase. The motivation for standardizing the order is simple: if for one (b,d) pair the first mode input to the DNN corresponds to the zero phase mode, but for an adjacent (b,d) pair the first mode input to the DNN corresponds to the second harmonic, the large difference between those mode shapes due to their representing different physical phenomena conceals the subtle changes in the mode shapes that would be useful in estimating (b,d). More formally, we label modes such that for a mode $\boldsymbol{\phi}_1^1$ with parameters (b,d) and $\boldsymbol{\phi}_1^2$ with parameters $(b+\epsilon_1, d+\epsilon_2)$, for small values of (ϵ_1,ϵ_2) we expect $\langle \boldsymbol{\phi}_1^1, \boldsymbol{\phi}_1^2 \rangle \approx 1$, and their corresponding eigenvalues $\boldsymbol{\lambda}_1^1/\boldsymbol{\lambda}_1^2 \approx 1 + 0i$.

The modes are sorted by first neglecting the complex conjugates, which is done by removing from Φ, Λ, and $\boldsymbol{\alpha}$ all entries with indices i that satisfy $\Im(\lambda_i) < 0$. Modes are then labeled using the procedure

$$\boldsymbol{\phi}_1 = \phi_i \text{ where } \arg\max_i(|\alpha_i|) \text{ subject to } \angle\lambda_i = 0$$
$$\boldsymbol{\phi}_2 = \phi_i \text{ where } \arg\max_i(|\alpha_i|) \text{ subject to } 0 < \angle\lambda_i \leq 0.6$$
$$\boldsymbol{\phi}_3 = \phi_i \text{ where } \arg\max_i(|\alpha_i|) \text{ subject to } 0.6 < \angle\lambda_i \leq 1.2$$

The values of $\boldsymbol{\lambda}_1$, $\boldsymbol{\lambda}_2$, and $\boldsymbol{\lambda}_3$ are labeled by the same procedure.

The second inconsistency between the modes that must be standardized is their phases. This is a result of the properties of eigenvectors: an eigenvector scaled by any arbitrary complex number remains an eigenvector. As the modes are eigenvectors, the phase of any individual element of the mode vector is arbitrary, though the relative phases of different elements are not. For simplicity, we discard the phase information and use only the magnitude of the mode elements in the estimation. The feature information is concatenated into a vector

$$Z = \begin{bmatrix} |\boldsymbol{\phi}_1| & |\boldsymbol{\phi}_2| & |\boldsymbol{\phi}_3| & |\boldsymbol{\lambda}_1| & |\boldsymbol{\lambda}_2| & |\boldsymbol{\lambda}_3| & \angle\boldsymbol{\lambda}_1 & \angle\boldsymbol{\lambda}_2 & \angle\boldsymbol{\lambda}_3 \end{bmatrix} \tag{30}$$

where $|\phi|$ denotes the elementwise magnitude of ϕ, and as a result each element is a real scalar. The vector Z of length 36 is then the output of the feature extractor function $\mathcal{F}$.

4.3 Mode-Kernel Estimation.

Because modes can be interpreted as features of the system that contain a condensed representation of the system dynamics, it is likely that on top of parametric approaches (like DNNs), nonparametric algorithms which compare the test data directly with the stored training data may also be applicable. Many such algorithms exist, but here we attempt a kernel-based approach. For each parameter pair k in the training dataset, sorted benchmark modes $[\boldsymbol{\phi}_1\ \boldsymbol{\phi}_2\ \boldsymbol{\phi}_3]|_k$ are identified using the procedure for exact DMD and the sorting from Sec. 4.2, except using the entire simulation data T^k instead of a sample of T^k. These modes are stored in a library Ω such that $\Omega_k = [\boldsymbol{\phi}_1\ \boldsymbol{\phi}_2\ \boldsymbol{\phi}_3]|_k$.

The estimation procedure works by estimating the similarity of the modes $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \boldsymbol{\phi}_3$ from the sample $S(\mathbb{T})$ with the dictionary of benchmark modes using a kernel function. In this case, the kernel function is the inner product, so the similarity between modes is defined as $\beta_{ik} = |\langle \boldsymbol{\phi}_i, (\Omega_k)_i \rangle|$, where a value near 1 indicates a high similarity. This similarity can be used to estimate (b,d) through simple algorithms, for example, the K-nearest neighbors algorithm. However, because of the high performance of parametric approaches to time series [43], as well as to keep consistency with the other approaches, we instead flatten β into a feature vector of length 108 to be output from $\mathcal{F}$ into $\mathcal{D}$.

4.4 Spectral Image Estimation.

While in this case it is simple to order the modes to perform a direct estimation as was done in both Secs. 4.2 and 4.3, that may not be the case for more complex fluid flows where more dominant modes are present, or for larger ranges of the estimation parameters where the modes change enough throughout the range to be not easily recognizable. The need for a human to derive the heuristics used to sort the modes for a given flow is also an obstacle to using this approach on an autonomous vehicle. Here, we present a method extract information from K_s without any heuristics or requirement to sort the modes. We first extract the real-valued spectra U_K of the operator, where

$$K_s = U_K \Sigma_K V_K^* \tag{31}$$

is the singular value decomposition of K_s. The matrix U_K is then passed into a two-dimensional CNN. Because two-dimensional CNNs are most often used for image classification, this can be interpreted as treating U_K as an image. For consistency, the hyperparameters of this network are chosen to be similar to those in Sec. 4.1: the convolutional steps have a filter size of 7, and max pooling is performed with a pool size of 3. However, because the operations are performed on an input matrix of size 10×10 instead of a vector of size 150, zero padding is required to allow applying two layers of convolution without the input becoming smaller than the filter. For the same reason, only one pooling operation is applied, located after both of the convolutional layers. After the pooling operation, the result is flattened into a feature vector also of length 108, which is then output from $\mathcal{F}$.

4.5 Training.

Each of the four estimation approaches is trained separately, with unique weights for each. For methods that have parameters ω associated with the feature extraction, ω is trained jointly with θ. Because of the risk of weight optimization finding local minima with different final loss values, a batch of ten estimators is trained for each of the estimation approaches, with the training process of each individual estimator termed a "run." In total, 40 weight optimizations are performed.

Before training, a training dataset is constructed by evaluating many realizations of the sampling function

$$(\boldsymbol{P}_i, \boldsymbol{b}_i, \boldsymbol{d}_i) = S(\mathbb{T}) \, \forall \, i \in \{1, 2, \ldots, 3600\} \tag{32}$$

Validation and testing datasets are constructed in a similar manner on $\mathcal{V}$ and $\mathcal{E}$, respectively, with the validation dataset constructed of 400 realizations of $S(\mathcal{V})$ and the testing dataset constructed of 760 realizations of $S(\mathcal{E})$. The training and validation datasets are recomputed for every run, but the testing dataset is only calculated once for consistency. The predicted values are given by

$$(\boldsymbol{b}_{p,i}, \boldsymbol{d}_{p,i}) = \mathcal{D}_\theta(\mathcal{F}_\omega(\boldsymbol{P})) \tag{33}$$

where $(\boldsymbol{b}_p, \boldsymbol{d}_p)$ represent vectors of predicted values of (b, d). The weights are updated such that

$$\min_{\theta,\omega} L \text{ where } L = \sum_{i=i_0}^{i_f} \left((\boldsymbol{b}_{p,i} - \boldsymbol{b}_i)^2 + (\boldsymbol{d}_{p,i} - \boldsymbol{d}_i)^2 \right) \tag{34}$$

which minimizes the error between predicted and actual values of (b, d) in a mean-squares sense. Instead of performing this summation over the entire set at once, it is efficient to iteratively perform the summation over smaller batches. Here, we chose a batch size of $i_f - i_0 = 50$. This batch-based optimization allows the use of the *adam* algorithm, which calculates the parameter updates at every step using a combination of the gradient of the current batch and the "momentum" from gradients calculated on past batches. After the entire dataset has been iterated over (known collectively as an "epoch"), an early-stopping criterion is checked, and either the training stops or continues with a new set of batches.

The early-stopping algorithm used was first introduced in Ref. [44]. After every epoch, the loss on the validation dataset L_v is calculated. If $L_v < L_{\min,i}$, where $L_{\min,i}$ is the lowest validation error encountered so far in run i, then $L_{\min,i} := L_v$ and the weights $\sigma_{\min,i} = (\theta, \omega)$ are saved before continuing. However, if at any epoch $L_v > 1.3 L_{\min,i}$ and at least 200 epochs have passed, it is assessed that the network is overtrained and the training run is terminated. The representative weights of the estimation approach, σ_{opt}, are then defined as

$$\sigma_{\text{opt}} = \sigma_{\min,i_{\text{opt}}} \tag{35}$$

$$i_{\text{opt}} = \arg\min_i L_{\min,i} \tag{36}$$

which selects the weights form the run with the lowest loss.

5 Results

The loss values during training on the validation data for the estimation approaches are shown in Fig. 9. The mode-kernel estimation approach has the highest average loss after epoch 50, though the lowest mode-kernel estimation approach loss is still lower than the average loss for all of the other three estimation approaches at the end of training, indicating that the overall difference between the estimation approaches is small. The other three estimation approaches have very similar average losses by the end of their training. However, the direct mode estimation approach has the run with the lowest loss by a small margin. A t-test is used to

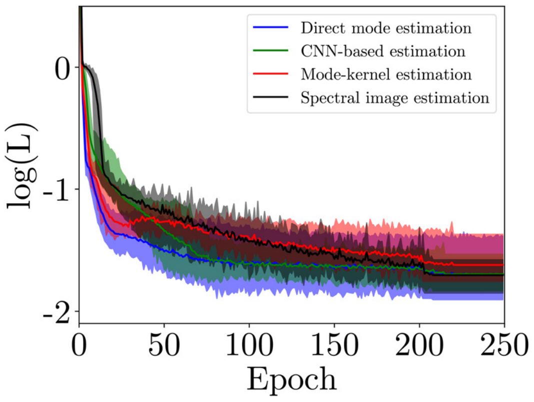


Fig. 9 The training curve for the four estimation methods. The shaded region represents the range of loss values on the validation data for the ten networks trained for each method, and the solid lines reflect the average. The mode-kernel estimation approach consistently performs poorly, and other three methods have roughly equal average loss, with the direct mode estimation approach having the lowest minimum loss.

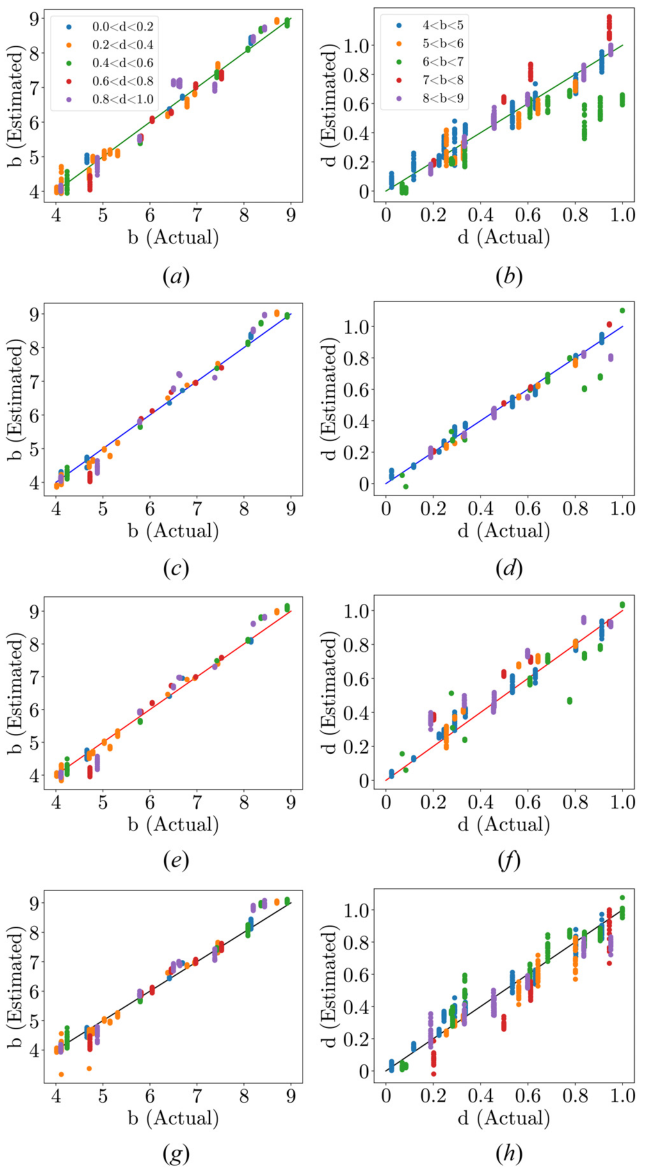


Fig. 10 Predicted versus real values of ((*a*), (*c*), (*e*), (*g*)) *b* and ((*b*), (*d*), (*f*), (*h*)) *d* using (*a*) and (*b*) the time series-CNN approach, (*c*) and (*d*) the direct mode estimation approach, (*e*) and (*f*) the mode-kernel estimation approach, and (*g*) and (*h*) the spectral image estimation approach

determine whether the mean minimum validation loss is statistically different between the methods. The result is that none of the methods used have a statistically significant difference in their mean minimum loss values from the CNN-based estimation approach (using $P = 0.05$ as the cutoff for significance).

The optimal weights σ_{opt} can be evaluated on the testing data to compare the best estimated values of (b, d) for each approach with the true values. These predicted and real values are shown in Fig. 10. The number of unique (b, d) pairs in the testing data (38) is much less than the number of samples evaluated (760), leading to many different estimations of the true value distributed along a vertical line. The CNN-based estimation (Figs. 10(*a*) and 10(*b*)) and spectral image estimation (Figs. 10(*g*) and 10(*h*)) approaches are imprecise in their estimates of d, with a large range of predicted values arising from different samples of data associated with the same true value. By contrast, the mode and kernel-based approaches are precise, with different samples from the same simulation giving similar outputs. This is consistent with the theoretical motivation of the Koopman operator; a given system has exactly one Koopman operator $\mathcal{K}$ that maps its dynamics forward by a specific length of time, which is valid both during the transient period and during steady-state. The estimated operator K_s does in practice change slightly depending on which window of data is selected, but is generally much more consistent than the time series itself. The spectral image estimation approach may be inconsistent because the singular value decomposition is not unique. We make it unique (to within a sign) by constraining the diagonal of Σ_K to be decreasing from the left to the right. However, this presents a problem for the use of U_K for estimation: very small changes in K_s can result in a reordering of Σ_K, which in turn results in a reordering of U_k. This reordering is likely responsible for the large range of estimates for d.

More quantitatively, differences between the estimation methods can be investigated by considering the loss on the testing dataset. The CNN-based estimation has the highest loss value of 0.0393, followed by the spectral image estimation at 0.0343, the direct mode estimation at 0.0340, and the mode-kernel estimation at 0.0325. The relative accuracy of the estimation methods is inconsistent with that calculated on the validation data; this inconsistency could indicate that different estimation approaches are suited to different regions of the parameter space, which is sampled differently by the testing and validation data due to the random location of samples. This may be particularly true for the kernel method, which may have difficulty with testing (b, d) values that are far from those in the training dataset.

6 Conclusion

We have shown that the Koopman operator can extract features amenable for estimation from a dynamic system more accurately than a state-of-the-art black-box convolutional neural network feature extractor. Multiple approaches to extract features from the Koopman operator are considered, and both directly inputting the modes to a DNN and inputting the spectrum of the operator into a CNN were found to be effective in estimating the parameters of the flow, with the approach using the dynamic modes performing slightly better and having shorter training times. This motivates the use of dynamic mode decomposition in the classification and estimation of parameters for multivariate time series which are generated by dynamic systems. The significant advantage of the Koopman operator-based approaches for parameter estimation over the approach using machine learning directly on time series is the physical insights offered into the estimation. Training data can be better selected, and sensitivity of estimation to changes in parameters can be better understood in light of such physics-informed machine learning. Future work will consider in more depth these aspects and the relationship between the measurable (surface) modes and the broader modes of the fluid.

This work opens several future avenues of research to make the application of the sensing and localization approach described in this paper practically useful in the mobile robotics context. The calculation of the Koopman operator and the classification of obstacle distance have to be done with smaller snapshots of data. This is a significant challenge that requires further optimization to the computation of the Koopman operator and the subsequent machine learning. Besides this, the presence of more complex flows with disturbances or the existence of more than one obstacle present more challenges to the sensing framework described in this paper. We currently envision this framework as one that augments other sensing modalities. Further research on such sensor fusion can be expected to improve flow sensing by underwater robots.

Funding Data

- Office of Naval Research (Grant No. 13204704; Funder ID: 10.13039/100000006).

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

References

[1] Triantafyllou, M. S., Weymouth, G. D., and Miao, J., 2016, "Biomimetic Survival Hydrodynamics and Flow Sensing," Annu. Rev. Fluid Mech., **48**(1), pp. 1–24.
[2] Gazzola, M., Argentina, M., and Mahadevan, L., 2015, "Gait and Speed Selection in Slender Inertial Swimmers," Proc. Natl. Acad. Sci. U.S.A., **112**(13), pp. 3874–3879.
[3] Ijspeert, I. A., 2014, "Biorobotics: Using Robots to Emulate and Investigate Agile Locomotion," Science, **346**(6206), pp. 196–203.
[4] Kelasidi, E., Liljeback, P., Pettersen, K. Y., and Gravdahl, J. T., 2016, "Innovation in Underwater Robots: Biologically Inspired Swimming Snake Robots," IEEE Rob. Autom. Mag., **23**(1), pp. 44–62.
[5] Pitcher, T. J., Partridge, B., and Wardle, C. S., 1976, "A Blind Fish Can School," Science, **194**(4268), pp. 963–965.
[6] Bleckmann, H., and Zelick, R., 2009, "Lateral Line System of Fish," Integr. Zool., **4**(1), pp. 13–25.
[7] Sirovich, L., 1987, "Turbulence and the Dynamics of Coherent Structures. Part 1: Coherent Structures," Q. Appl. Math., **45**(3), pp. 561–571.
[8] Berkooz, G., Holmes, P., and Lumley, J. L., 2003, "The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows," Annu. Rev. Fluid Mech., **25**(1), pp. 539–575.
[9] Everson, R., and Sirovich, L., 1995, "Karhunen–Loève Procedure for Gappy Data," J. Opt. Soc. Am. A, **12**(8), pp. 1657–1664.
[10] Bui-Thanh, T., Damodaran, M., and Willcox, K. E., 2004, "Aerodynamic Data Reconstruction and Inverse Design Using Proper Orthogonal Decomposition," AIAA J., **42**(8), pp. 1505–1516.
[11] Willcox, K. E., 2006, "Unsteady Flow Sensing and Estimation Via the Gappy Proper Orthogonal Decomposition," Comput. Fluids, **35**(2), pp. 208–226.
[12] Rowley, C. W., Mezić, I., Bagheri, S., Schlatter, P., and Henningson, D. S., 2009, "Spectral Analysis of Nonlinear Flows," J. Fluid Mech., **641**, pp. 115–127.
[13] Schmid, P. J., 2010, "Dynamic Mode Decomposition of Numerical and Experimental Data," J. Fluid Mech., **656**, pp. 5–28.
[14] Li, Q., Dietrich, F., Bollt, E. M., and Kevrekidis, I. G., 2017, "Extended Dynamic Mode Decomposition With Dictionary Learning: A Data-Driven Adaptive Spectral Decomposition of the Koopman Operator," Chaos, **27**(10), p. 103111.
[15] Otto, S., and Rowley, C., 2019, "Linearly Recurrent Autoencoder Networks for Learning Dynamics," SIAM J. Appl. Dyn. Syst., **18**(1), pp. 558–593.
[16] Champion, K., Lusch, B., Nathan Kutz, J., and Brunton, S. L., 2019, "Data-Driven Discovery of Coordinates and Governing Equations," PNAS, **116**(45), pp. 22445–22451.
[17] Raissi, M., Yazdani, A., and Karniadakis, G., 2020, "Hidden Fluid Mechanics: Learning Velocity and Pressure Fields From Flow Visualizations," Science, **367**(6481), pp. 1026–1030.
[18] Brunton, S. L., Noack, B. R., and Koumoutsakos, P., 2019, "Machine Learning for Fluid Mechanics," Annu. Rev. Fluid Mech., **52**, p. 2020.
[19] Callaham, J. L., Maeda, K., and Brunton, S. L., 2019, "Robust Flow Reconstruction From Limited Measurements Via Sparse Representation," Phys. Rev. Fluids, **4**(10), p. 103907.
[20] Alsalman, M., Colvert, B., and Kanso, E., 2018, "Training Bioinspired Sensors to Classify Flows," Bioinspiration Biomimetics, **14**(1), p. 016009.
[21] Colvert, B., Alsalman, M., and Kanso, E., 2018, "Classifying Vortex Wakes Using Neural Networks," Bioinspiration Biomimetics, **13**(2), p. 025003.
[22] Pollard, B., and Tallapragada, P., 2020, "Sensing and Classification of Ambient Vortex Wake From the Kinematics of a Bioinspired Swimming Robot Using Neural Networks," ASME Paper No. DSCC2020-3282.
[23] Pollard, B., and Tallapragada, P., 2021, "Learning Hydrodynamic Signatures Through Proprioceptive Sensing by Bioinspired Swimmers," Bioinspiration Biomimetics, **16**(2), p. 026014.

[24] Rodwell, C., Pollard, B., and Tallapragada, P., 2023, "Proprioceptive Wake Classification by a Body With a Passive Tail," Bioinspiration Biomimetics, **18**(4), p. 046001.
[25] Rodwell, C., and Tallapragada, P., 2022, "Embodied Hydrodynamic Sensing and Estimation Using Koopman Modes in an Underwater Environment," American Control Conference (ACC), Atlanta, GA, June 8–10, pp. 1632–1637.
[26] Rodwell, C., Sourav, K., and Tallapragada, P., 2024, "Feel the Force: From Local Surface Pressure Measurement to Flow Reconstruction in Fluid–Structure Interaction," Phys. Fluids, **36**(1), p. 013606.
[27] Bright, I., Lin, G., and Kutz, J. N., 2013, "Compressive Sensing Based Machine Learning Strategy for Characterizing the Flow Around a Cylinder With Limited Pressure Measurements," Phys. Fluids, **25**(12), p. 127102.
[28] Gomez, D. F., Lagor, F. D., Kirk, P. B., Lind, A. H., Jones, A. R., and Paley, D. A., 2019, "Data-Driven Estimation of the Unsteady Flowfield Near an Actuated Airfoil," J. Guid., Control, Dyn., **42**(10), pp. 2279–2287.
[29] Lidard, J. M., Goswami, D., Snyder, D., Sedky, G., Jones, A. R., and Paley, D. A., 2021, "Output Feedback Control for Lift Maximization of a Pitching Airfoil," J. Guid., Control, Dyn., **44**(3), pp. 587–594.
[30] Yen, W.-K., Huang, C.-F., Chang, H.-R., and Guo, J., 2020, "Localization of a Leading Robotic Fish Using a Pressure Sensor Array on Its Following Vehicle," Bioinspiration Biomimetics, **16**(1), p. 016007.
[31] Abdulsadda, A. T., and Tan, X., 2013, "Underwater Tracking of a Moving Dipole Source Using an Artificial Lateral Line: Algorithm and Experimental Validation With Ionic Polymer–Metal Composite Flow Sensors," Smart Mater. Struct., **22**(4), p. 045010.
[32] Wolf, B. J., van de Wolfshaar, J., and van Netten, S. M., 2020, "Three-Dimensional Multi-Source Localization of Underwater Objects Using Convolutional Neural Networks for Artificial Lateral Lines," J. R. Soc. Interface, **17**(162), p. 20190616.
[33] Souza, F. A., Araújo, R., and Mendes, J., 2016, "Review of Soft Sensor Methods for Regression Applications," Chemom. Intell. Lab. Syst., **152**, pp. 69–79.
[34] Facco, P., Doplicher, F., Bezzo, F., and Barolo, M., 2009, "Moving Average PLS Soft Sensor for Online Product Quality Estimation in an Industrial Batch Polymerization Process," J. Process Control, **19**(3), pp. 520–529.
[35] Zhao, B., Lu, H., Chen, S., Liu, J., and Wu, D., 2017, "Convolutional Neural Networks for Time Series Classification," J. Syst. Eng. Electron., **28**(1), pp. 162–169.
[36] Issa, R. I., 1986, "Solution of the Implicitly Discretised Fluid Flow Equations by Operator-Splitting," J. Comput. Phys., **62**(1), pp. 40–65.
[37] Koopman, B. O., 1931, "Hamiltonian Systems and Transformation in Hilbert Space," Proc. Natl. Acad. Sci., **17**(5), pp. 315–318.
[38] Lasota, A., and Mackey, M. C., 1994, *Chaos, Fractals, and Noise: Stochastic Aspects of Dynamics*, Springer, New York.
[39] Budisic, M., Mohr, R., and Mezić, I., 2012, "Applied Koopmanism," Chaos: Interdiscip. J. Nonlinear Sci., **22**(4), p. 047510.
[40] Tu, J. H., Rowley, C. W., Luchtenburg, D. M., Brunton, S. L., and Kutz, J. N., 2014, "On Dynamic Mode Decomposition: Theory and Applications," J. Comput. Dyn., **1**(2), pp. 391–421.
[41] Korda, M., and Mezic, I., 2017, "On Convergence of Extended Dynamic Mode Decomposition to the Koopman Operator," J. Nonlinear Sci., **28**(2), pp. 687–710.
[42] Haseli, M., and Cortes, J., 2023, "Temporal Forward–Backward Consistency, Not Residual Error, Measures the Prediction Accuracy of Extended Dynamic Mode Decomposition," IEEE Control Syst. Lett., **7**, pp. 649–654.
[43] Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., and Muller, P.-A., 2019, "Deep Learning for Time Series Classification: A Review," Data Min. Knowl. Discovery, **33**(4), pp. 917–963.
[44] Prechelt, L., 1998, "Early Stopping—But When?," *Neural Networks: Tricks of the Trade*, Springer, Berlin, Germany, pp. 55–69.