ANNUAL REVIEWS

Annual Review of Fluid Mechanics

Machine Learning for Fluid Mechanics

Steven L. Brunton,[1] Bernd R. Noack,[2,3] and Petros Koumoutsakos[4]

[1]Department of Mechanical Engineering, University of Washington, Seattle, Washington 98195, USA

[2]LIMSI (Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur), CNRS UPR 3251, Université Paris-Saclay, F-91403 Orsay, France

[3]Institut für Strömungsmechanik und Technische Akustik, Technische Universität Berlin, D-10634 Berlin, Germany

[4]Computational Science and Engineering Laboratory, ETH Zurich, CH-8092 Zurich, Switzerland; email: petros@ethz.ch

Annu. Rev. Fluid Mech. 2020. 52:477–508

First published as a Review in Advance on September 12, 2019

The *Annual Review of Fluid Mechanics* is online at fluid.annualreviews.org

https://doi.org/10.1146/annurev-fluid-010719-060214

Keywords

machine learning, data-driven modeling, optimization, control

Abstract

The field of fluid mechanics is rapidly advancing, driven by unprecedented volumes of data from experiments, field measurements, and large-scale simulations at multiple spatiotemporal scales. Machine learning (ML) offers a wealth of techniques to extract information from data that can be translated into knowledge about the underlying fluid mechanics. Moreover, ML algorithms can augment domain knowledge and automate tasks related to flow control and optimization. This article presents an overview of past history, current developments, and emerging opportunities of ML for fluid mechanics. We outline fundamental ML methodologies and discuss their uses for understanding, modeling, optimizing, and controlling fluid flows. The strengths and limitations of these methods are addressed from the perspective of scientific inquiry that considers data as an inherent part of modeling, experiments, and simulations. ML provides a powerful information-processing framework that can augment, and possibly even transform, current lines of fluid mechanics research and industrial applications.

1. INTRODUCTION

Machine learning: algorithms that process and extract information from data; they facilitate automation of tasks and augment human domain knowledge

Supervised learning: learning from data labeled with expert knowledge, providing corrective information to the algorithm

Semisupervised learning: learning with partially labeled data (generative adversarial networks) or by interactions of the machine with its environment (reinforcement learning)

Unsupervised learning: learning without labeled training data

Fluid mechanics has traditionally dealt with massive amounts of data from experiments, field measurements, and large-scale numerical simulations. Indeed, in the past few decades, big data have been a reality in fluid mechanics research (Pollard et al. 2016) due to high-performance computing architectures and advances in experimental measurement capabilities. Over the past 50 years, many techniques were developed to handle such data, ranging from advanced algorithms for data processing and compression to fluid mechanics databases (Perlman et al. 2007, Wu & Moin 2008). However, the analysis of fluid mechanics data has relied, to a large extent, on domain expertise, statistical analysis, and heuristic algorithms.

The growth of data today is widespread across scientific disciplines, and gaining insight and actionable information from data has become a new mode of scientific inquiry as well as a commercial opportunity. Our generation is experiencing an unprecedented confluence of (*a*) vast and increasing volumes of data; (*b*) advances in computational hardware and reduced costs for computation, data storage, and transfer; (*c*) sophisticated algorithms; (*d*) an abundance of open source software and benchmark problems; and (*e*) significant and ongoing investment by industry on data-driven problem solving. These advances have, in turn, fueled renewed interest and progress in the field of machine learning (ML) to extract information from these data. ML is now rapidly making inroads in fluid mechanics. These learning algorithms may be categorized into supervised, semisupervised, and unsupervised learning (see **Figure 1**), depending on the information available about the data to the learning machine (LM).

ML provides a modular and agile modeling framework that can be tailored to address many challenges in fluid mechanics, such as reduced-order modeling, experimental data processing, shape optimization, turbulence closure modeling, and control. As scientific inquiry shifts from first principles to data-driven approaches, we may draw a parallel with the development of numerical methods in the 1940s and 1950s to solve the equations of fluid dynamics. Fluid mechanics stands to benefit from learning algorithms and in return presents challenges that may further advance these algorithms to complement human understanding and engineering intuition.

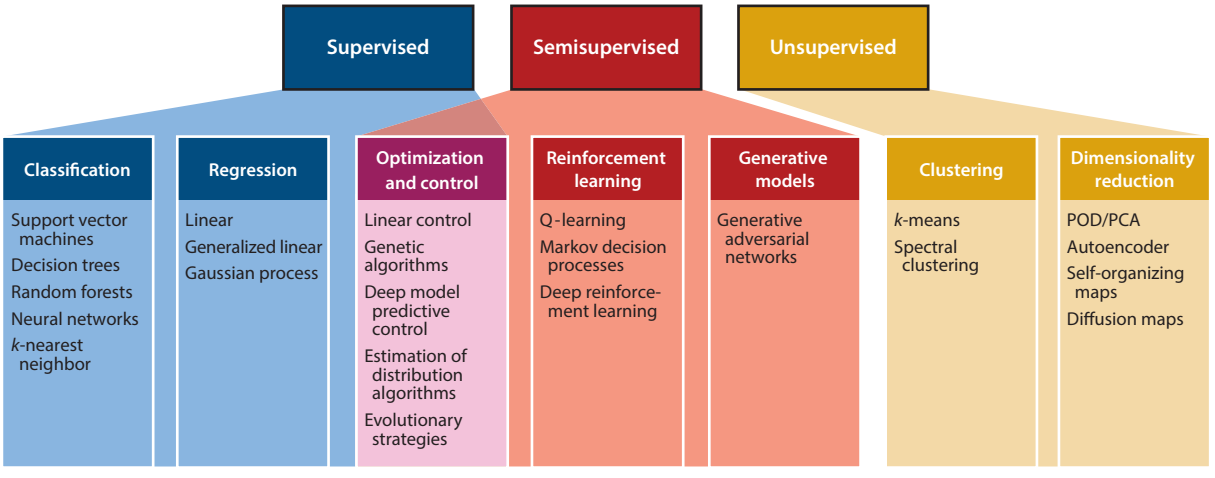

Figure 1

Machine learning algorithms may be categorized into supervised, unsupervised, and semisupervised, depending on the extent and type of information available for the learning process. Abbreviations: PCA, principal component analysis; POD, proper orthogonal decomposition.

In addition to outlining successes, we must note the importance of understanding how learning algorithms work and when these methods succeed or fail. It is important to balance excitement about the capabilities of ML with the reality that its application to fluid mechanics is an open and challenging field. In this context, we also highlight the benefit of incorporating domain knowledge about fluid mechanics into learning algorithms. We envision that the fluid mechanics community can contribute to advances in ML reminiscent of advances in numerical methods in the last century.

Reduced-order model: representation of a high-dimensional system in terms of a low-dimensional one, balancing accuracy and efficiency

Perceptron: the first learning machine; a network of binary decision units used for classification

1.1. Historical Overview

The interface between ML and fluid dynamics has a long and possibly surprising history. In the early 1940s, Kolmogorov, a founder of statistical learning theory, considered turbulence as one of its prime application domains (Kolmogorov 1941). Advances in ML in the 1950s and 1960s were characterized by two distinct developments. On one side, we may distinguish cybernetics (Wiener 1965) and expert systems designed to emulate the thinking process of the human brain, and on the other side, machines such as the perceptron (Rosenblatt 1958) aimed to automate processes such as classification and regression. The use of perceptrons for classification created significant excitement. However, this excitement was quenched by findings that their capabilities had severe limitations (Minsky & Papert 1969): Single-layer perceptrons were only able to learn linearly separable functions and were not capable of learning the XOR function. It was known that multilayer perceptrons could learn the XOR function, but perhaps their advancement was limited given the computational resources of the times (a recurring theme in ML research). The reduced interest in perceptrons was soon accompanied by a reduced interest in artificial intelligence (AI) in general.

Another branch of ML, closely related to budding ideas of cybernetics in the early 1960s, was pioneered by two graduate students: Ingo Rechenberg and Hans-Paul Schwefel at the Technical University of Berlin. They performed experiments in a wind tunnel on a corrugated structure composed of five linked plates with the goal of finding their optimal angles to reduce drag (see **Figure 2**). Their breakthrough involved adding random variations to these angles, where the randomness was generated using a Galton board (an analog random number generator). Most importantly, the size of the variance was learned (increased/decreased) based on the success rate (positive/negative) of the experiments. Despite its brilliance, the work of Rechenberg and Schwefel has received little recognition in the fluid mechanics community, even though a significant number of applications in fluid mechanics and aerodynamics use ideas that can be traced back to their work. Renewed interest in the potential of AI for aerodynamics applications materialized almost simultaneously with the early developments in computational fluid dynamics in the early 1980s. Attention was given to expert systems to assist in aerodynamic design and development processes (Mehta & Kutler 1984).

An indirect link between fluid mechanics and ML was the so-called Lighthill report in 1974 that criticized AI programs in the United Kingdom as not delivering on their grand claims. This report played a major role in the reduced funding and interest in AI in the United Kingdom and subsequently in the United States that is known as the AI winter. Lighthill's main argument was based on his perception that AI would never be able to address the challenge of the combinatorial explosion between possible configurations in the parameter space. He used the limitations of language processing systems of that time as a key demonstration of the failures for AI. In Lighthill's defense, 40 years ago the powers of modern computers as we know them today may have been difficult to fathom. Indeed, today one may watch Lighthill's speech on the internet while an ML algorithm automatically provides the captions.

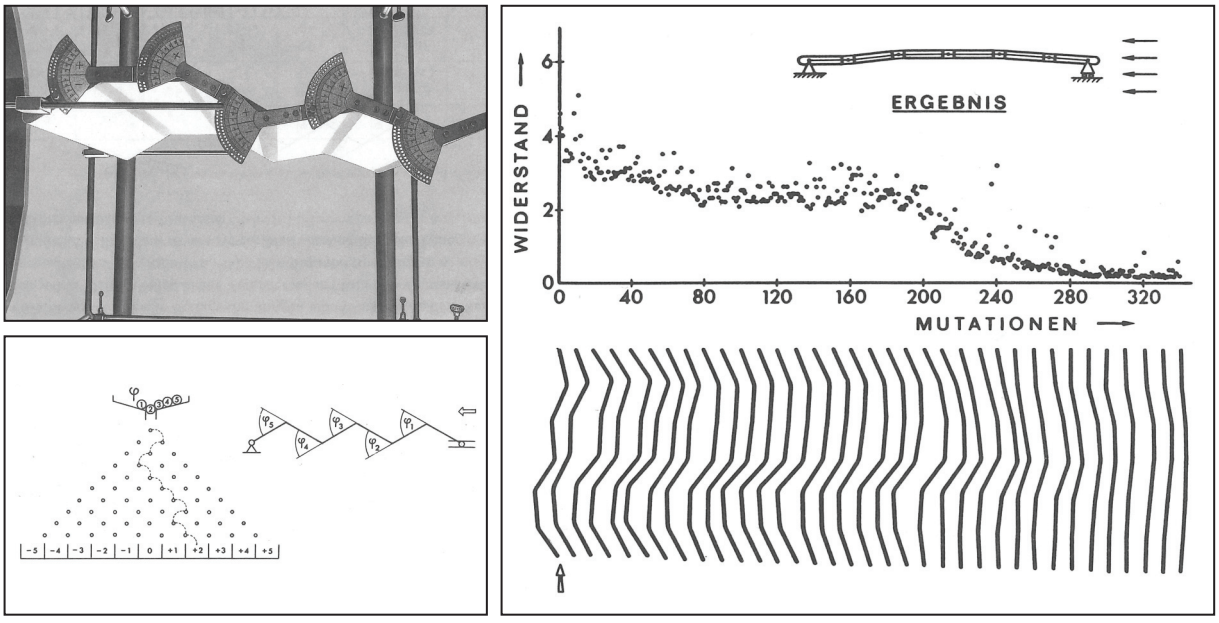

Figure 2

First example of learning and automation in experimental fluid mechanics: Rechenberg's (1964) experiments for optimally corrugated plates for drag reduction using the Galtonbrett (Galton board) as an analog random number generator. Figure reprinted with permission from Rechenberg (1973).

The reawakening of interest in ML, and in neural networks (NNs) in particular, came in the late 1980s with the development of the backpropagation algorithm (Rumelhart et al. 1986). This enabled the training of NNs with multiple layers, even though in the early days at most two layers were the norm. Other sources of stimulus were the works by Hopfield (1982), Gardner (1988), and Hinton & Sejnowski (1986), who developed links between ML algorithms and statistical mechanics. However, these developments did not attract many researchers from fluid mechanics. In the early 1990s a number of applications of NNs in flow-related problems were developed in the context of trajectory analysis and classification for particle tracking velocimetry (PTV) and particle image velocimetry (PIV) (Teo et al. 1991, Grant & Pan 1995) as well as for identifying the phase configurations in multiphase flows (Bishop & James 1993). The link between proper orthogonal decomposition (POD) and linear NNs (Baldi & Hornik 1989) was exploited in order to reconstruct turbulence flow fields and the flow in the near-wall region of a channel flow using wall-only information (Milano & Koumoutsakos 2002). This application was one of the first to also use multiple layers of neurons to improve compression results, marking perhaps the first use of deep learning, as it is known today, in the field of fluid mechanics.

Neural network: a computational architecture, based loosely on biological networks of neurons, for nonlinear regression

Deep learning: neural networks with multiple layers; used to create powerful hierarchical representations at varying levels of abstraction

In the past few years, we have experienced a renewed blossoming of ML applications in fluid mechanics. Much of this interest is attributed to the remarkable performance of deep learning architectures, which hierarchically extract informative features from data. This has led to several advances in data-rich and model-limited fields such as the social sciences and in companies for which prediction is a key financial factor. Fluid mechanics is not a model-limited field, and it is rapidly becoming data rich. We believe that this confluence of first principles and data-driven approaches is unique and has the potential to transform both fluid mechanics and ML.

1.2. Challenges and Opportunities for Machine Learning in Fluid Dynamics

Fluid dynamics presents challenges that differ from those tackled in many applications of ML, such as image recognition and advertising. In fluid flows it is often important to precisely quantify the underlying physical mechanisms in order to analyze them. Furthermore, fluid flows exhibit complex, multiscale phenomena the understanding and control of which remain largely unresolved. Unsteady flow fields require algorithms capable of addressing nonlinearities and multiple spatiotemporal scales that may not be present in popular ML algorithms. In addition, many prominent applications of ML, such as playing the game Go, rely on inexpensive system evaluations and an exhaustive categorization of the process that must be learned. This is not the case in fluids, where experiments may be difficult to repeat or automate and where simulations may require large-scale supercomputers operating for extended periods of time.

Reinforcement learning: an agent learns a policy to maximize its long-term rewards by interacting with its environment

ML has also become instrumental in robotics, and algorithms such as reinforcement learning (RL) are used routinely in autonomous driving and flight. While many robots operate in fluids, it appears that the subtleties of fluid dynamics are not presently a major concern in their design. Reminiscent of the pioneering days of flight, solutions imitating natural forms and processes are often the norm (see the sidebar titled Learning Fluid Mechanics: From Living Organisms to Machines). We believe that deeper understanding and exploitation of fluid mechanics will become critical in the design of robotic devices when their energy consumption and reliability in complex flow environments become a concern.

In the context of flow control, actively or passively manipulating flow dynamics for an engineering objective may change the nature of the system, making predictions based on data of uncontrolled systems impossible. Although flow data are vast in some dimensions, such as spatial resolution, they may be sparse in others; for example, it may be expensive to perform parametric studies. Furthermore, flow data can be highly heterogeneous, requiring special care when choosing the type of LM. In addition, many fluid systems are nonstationary, and even for stationary flows it may be prohibitively expensive to obtain statistically converged results.

Fluid dynamics is central to transportation, health, and defense systems, and it is therefore essential that ML solutions are interpretable, explainable, and generalizable. Moreover, it is

LEARNING FLUID MECHANICS: FROM LIVING ORGANISMS TO MACHINES

Birds, bats, insects, fish, whales, and other aquatic and aerial life-forms perform remarkable feats of fluid manipulation, optimizing and controlling their shape and motion to harness unsteady fluid forces for agile propulsion, efficient migration, and other exquisite maneuvers. The range of fluid flow optimization and control observed in biology is breathtaking and has inspired humans for millennia. How do these organisms learn to manipulate the flow environment?

To date, we know of only one species that manipulates fluids through knowledge of the Navier–Stokes equations. Humans have been innovating and engineering devices to harness fluids since before the dawn of recorded history, from dams and irrigation to mills and sailing. Early efforts were achieved through intuitive design, although recent quantitative analysis and physics-based design have enabled a revolution in performance over the past hundred years. Indeed, physics-based engineering of fluid systems is a high-water mark of human achievement. However, there are serious challenges associated with equation-based analysis of fluids, including high dimensionality and nonlinearity, which defy closed-form solutions and limit real-time optimization and control efforts. At the beginning of a new millennium, with increasingly powerful tools in machine learning and data-driven optimization, we are again learning how to learn from experience.

often necessary to provide guarantees on performance, which are presently rare. Indeed, there is a poignant lack of convergence results, analysis, and guarantees in many ML algorithms. It is also important to consider whether the model will be used for interpolation within a parameter regime or for extrapolation, which is considerably more challenging. Finally, we emphasize the importance of cross-validation on withheld data sets to prevent overfitting in ML.

We suggest that this nonexhaustive list of challenges need not be a barrier; to the contrary, it should provide a strong motivation for the development of more effective ML techniques. These techniques will likely impact several disciplines if they are able to solve fluid mechanics problems. The application of ML to systems with known physics, such as fluid mechanics, may provide deeper theoretical insights into algorithms. We also believe that hybrid methods that combine ML and first principles models will be a fertile ground for development.

This review is structured as follows: Section 2 outlines the fundamental algorithms of ML, followed by discussions of their applications to flow modeling (Section 3) and optimization and control (Section 4). We provide a summary and outlook of this field in Section 5.

2. MACHINE LEARNING FUNDAMENTALS

The learning problem can be formulated as the process of estimating associations between inputs, outputs, and parameters of a system using a limited number of observations (Cherkassky & Mulier 2007). We distinguish between a generator of samples, the system in question, and an LM, as in **Figure 3**. We emphasize that the approximations by LMs are fundamentally stochastic, and their learning process can be summarized as the minimization of a risk functional:

$$R(\mathbf{w}) = \int L[\mathbf{y}, \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}, \mathbf{w})]\, p(\mathbf{x}, \mathbf{y})\, d\mathbf{x}d\mathbf{y}, \qquad 1.$$

where the data $\mathbf{x}$ (inputs) and $\mathbf{y}$ (outputs) are samples from a probability distribution p, $\boldsymbol{\phi}(\mathbf{x}, \mathbf{y}, \mathbf{w})$ defines the structure and $\mathbf{w}$ the parameters of the LM, and the loss function L balances the various learning objectives (e.g., accuracy, simplicity, smoothness, etc.). We emphasize that the risk functional is weighted by a probability distribution $p(\mathbf{x}, \mathbf{y})$ that also constrains the predictive capabilities of the LM. The various types of learning algorithms can be grouped into three major categories: supervised, unsupervised, and semisupervised, as in **Figure 1**. These distinctions signify the degree to which external supervisory information from an expert is available to the LM.

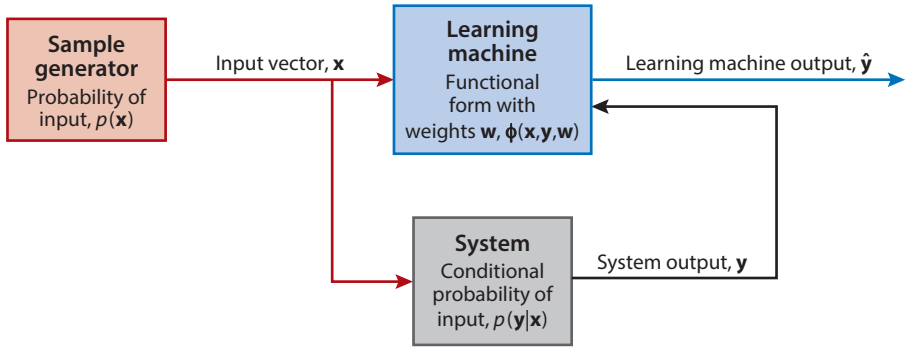

Figure 3

The learning problem. A learning machine uses inputs from a sample generator and observations from a system to generate an approximation of its output. Figure based on an idea from Cherkassky & Mulier (2007).

2.1. Supervised Learning

Supervised learning implies the availability of corrective information to the LM. In its simplest and most common form, this implies labeled training data, with labels corresponding to the output of the LM. Minimization of the cost function, which depends on the training data, will determine the unknown parameters of the LM. In this context, supervised learning dates back to the regression and interpolation methods proposed centuries ago by Gauss (Meijering 2002). A commonly employed loss function is

$$L[\mathbf{y}, \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}, \mathbf{w})] = ||\mathbf{y} - \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}, \mathbf{w})||^2. \quad 2.$$

Alternative loss functions may reflect different constraints on the LM such as sparsity (Hastie et al. 2009, Brunton & Kutz 2019). The choice of the approximation function reflects prior knowledge about the data, and the choice between linear and nonlinear methods directly bears on the computational cost associated with the learning methods.

2.1.1. Neural networks. NNs are arguably the most well-known methods in supervised learning. They are fundamental nonlinear function approximators, and in recent years several efforts have been dedicated to understanding their effectiveness. The universal approximation theorem (Hornik et al. 1989) states that any function may be approximated by a sufficiently large and deep network. Recent work has shown that sparsely connected, deep NNs are information theoretic–optimal nonlinear approximators for a wide range of functions and systems (Bölcskei et al. 2019).

The power and flexibility of NNs emanate from their modular structure based on the neuron as a central building element, a caricature of neurons in the human brain. Each neuron receives an input, processes it through an activation function, and produces an output. Multiple neurons can be combined into different structures that reflect knowledge about the problem and the type of data. Feedforward networks are among the most common structures, and they are composed of layers of neurons, where a weighted output from one layer is the input to the next layer. NN architectures have an input layer that receives the data and an output layer that produces a prediction. Nonlinear optimization methods, such as backpropagation (Rumelhart et al. 1986), are used to identify the network weights to minimize the error between the prediction and labeled training data. Deep NNs involve multiple layers and various types of nonlinear activation functions. When the activation functions are expressed in terms of convolutional kernels, a powerful class of networks emerges, namely convolutional neural networks (CNNs), with great success in image and pattern recognition (Krizhevsky et al. 2012, Goodfellow et al. 2016), Grossberg et al. 1988).

Recurrent neural networks (RNNs), depicted in **Figure 4**, are of particular interest to fluid mechanics. They operate on sequences of data (e.g., images from a video, time series, etc.), and their weights are obtained by backpropagation through time. RNNs have been quite successful for natural language processing and speech recognition. Their architecture takes into account the inherent order of the data, thus augmenting some of the pioneering applications of classical NNs on signal processing (Rico-Martinez et al. 1992). However, their effectiveness has been hindered by diminishing or exploding gradients that emerge during their training. The renewed interest in RNNs is largely attributed to the development of the long short-term memory (LSTM) (Hochreiter & Schmidhuber 1997) algorithms that deploy cell states and gating mechanisms to store and forget information about past inputs, thus alleviating the problems with gradients and the transmission of long-term information from which standard RNNs suffer. An extended architecture called the

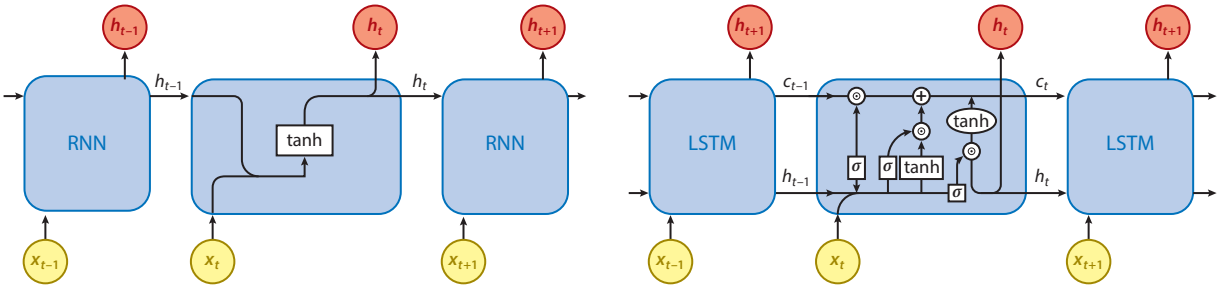

Figure 4

Recurrent neural networks (RNNs) for time series predictions and the long short-term memory (LSTM) regularization. Abbreviations: c_{t-1}, previous cell memory; c_t, current cell memory; h_{t-1}, previous cell output; h_t, current cell output; x_t, input vector; σ, sigmoid. Figure based on an idea from Hochreiter & Schmidhuber (1997).

multidimensional LSTM network (Graves et al. 2007) was proposed to efficiently handle high-dimensional spatiotemporal data. Several potent alternatives to RNNs have appeared over the years; the echo state network has been used with success in predicting the output of several dynamical systems (Pathak et al. 2018).

2.1.2. Classification: support vector machines and random forests. Classification is a supervised learning task that can determine the label or category of a set of measurements from a priori labeled training data. It is perhaps the oldest method for learning, starting with the perceptron (Rosenblatt 1958), which could classify between two types of linearly separable data. Two fundamental classification algorithms are support vector machines (SVMs) (Schölkopf & Smola 2002) and random forests (Breiman 2001), which have been widely adopted in industry. The problem can be specified by the following loss functional, which is expressed here for two classes:

$$L[\mathbf{y}, \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}, \mathbf{w})] = \begin{cases} 0, & \text{if } \mathbf{y} = \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}, \mathbf{w}), \\ 1, & \text{if } \mathbf{y} \neq \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}, \mathbf{w}). \end{cases} \quad 3.$$

The output of the LM is an indicator of the class to which the data belong. The risk functional quantifies the probability of misclassification, and the task is to minimize the risk based on the training data by suitable choice of $\boldsymbol{\phi}(\mathbf{x}, \mathbf{y}, \mathbf{w})$. Random forests are based on an ensemble of decision trees that hierarchically split the data using simple conditional statements; these decisions are interpretable and fast to evaluate at scale. In the context of classification, an SVM maps the data into a high-dimensional feature space on which a linear classification is possible.

2.2. Unsupervised Learning

This learning task implies the extraction of features from the data by specifying certain global criteria, without the need for supervision or a ground-truth label for the results. The types of problems involved here include dimensionality reduction, quantization, and clustering.

2.2.1. Dimensionality reduction I: proper orthogonal decomposition, principal component analysis, and autoencoders. The extraction of flow features from experimental data and large-scale simulations is a cornerstone of flow modeling. Moreover, identifying lower-dimensional representations for high-dimensional data can be used as preprocessing for all tasks in supervised learning algorithms. Dimensionality reduction can also be viewed as an

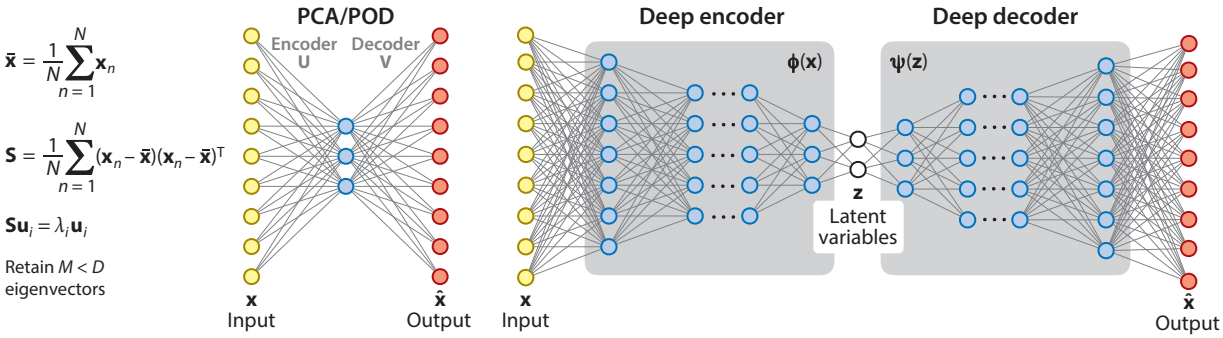

Figure 5

PCA/POD (*left*) versus shallow autoencoders (*center*) and deep autoencoders (*right*). If the node activation functions in the shallow autoencoder are linear, then $\mathbf{U}$ and $\mathbf{V}$ are matrices that minimize the loss function, $\|\hat{\mathbf{x}} - \mathbf{V}\mathbf{U}\mathbf{x}\|$. The node activation functions may be nonlinear, minimizing the loss function, $\|\mathbf{x} - \boldsymbol{\psi}[\boldsymbol{\phi}(\mathbf{x})]\|$. The input $\mathbf{x} \in \mathbb{R}^D$ is reduced to $\mathbf{z} \in \mathbb{R}^M$, with $M \ll D$. Note that PCA/POD requires the solution of a problem-specific eigenvalue equation, while the neuron modules can be extended to nonlinear activation functions and multiple nodes and layers. Abbreviations: PCA, principal component analysis; POD, proper orthogonal decomposition; $\mathbf{S}$, covariance matrix of mean-subtracted data; $\mathbf{U}$, linear encoder; $\mathbf{u}_i$, eigenvector; $\mathbf{V}$, linear decoder; $\mathbf{x}$, input vector; $\mathbf{x}_n$, n-th input vector; $\bar{\mathbf{x}}$, mean of input data; $\hat{\mathbf{x}}$, autoencoder reconstruction; $\mathbf{z}$, latent variable; λ_i, eigenvalue; $\boldsymbol{\phi}(\mathbf{x})$, deep encoder; $\boldsymbol{\psi}(\mathbf{x})$, deep decoder. Figure based on an idea from Bishop & James (1993).

information-filtering bottleneck where the data are processed through a lower-dimensional representation before being mapped back to the ambient dimension. The classical POD algorithm belongs to this category of learning and is discussed more in Section 3. The POD, or linear principal component analysis (PCA) as it is more widely known, can be formulated as a two-layer NN (an autoencoder) with a linear activation function for its linearly weighted input, which can be trained by stochastic gradient descent (see **Figure 5**). This formulation is an algorithmic alternative to linear eigenvalue/eigenvector problems in terms of NNs, and it offers a direct route to the nonlinear regime and deep learning by adding more layers and a nonlinear activation function on the network. Unsupervised learning algorithms have seen limited use in the fluid mechanics community, and we believe that they deserve further exploration. In recent years, the ML community has developed numerous autoencoders that, when properly matched with the possible features of the flow field, can lead to significant insight for reduced-order modeling of stationary and time-dependent data.

2.2.2. Dimensionality reduction II: discrete principal curves and self-organizing maps. The mapping between high-dimensional data and a low-dimensional representation can be structured through an explicit shaping of the lower-dimensional space, possibly reflecting an a priori knowledge about this subspace. These techniques can be seen as extensions of the linear autoencoders, where the encoder and decoder can be nonlinear functions. This nonlinearity may, however, come at the expense of losing the inverse relationship between the encoder and decoder functions that is one of the strengths of linear PCA. An alternative is to define the decoder as an approximation of the inverse of the encoder, leading to the method of principal curves. Principal curves are structures on which the data are projected during the encoding step of the learning algorithm. In turn, the decoding step amounts to an approximation of the inverse of this mapping by adding, for example, some smoothing onto the principal curves. An important version of this process is the self-organizing map (SOM) introduced by Grossberg (1976) and Kohonen (1995). In SOMs the projection subspace is described into a finite set of values with specified connectivity

Autoencoder: a neural network architecture used to compress and decompress high-dimensional data; linear and nonlinear alternative to the proper orthogonal decomposition

architecture and distance metrics. The encoder step amounts to identifying for each data point the closest node point on the SOM, and the decoder step is a weighted regression estimate using, for example, kernel functions that take advantage of the specified distance metric between the map nodes. This modifies the node centers, and the process can be iterated until the empirical risk of the autoencoder has been minimized. The SOM capabilities can be exemplified by comparing it to linear PCA for a two-dimensional set of points. The linear PCA will provide as an approximation the least squares straight line between the points, whereas the SOM will map the points onto a curved line that better approximates the data. We note that SOMs can be extended to areas beyond floating point data, and they offer an interesting way for creating databases based on features of flow fields.

2.2.3. Clustering and vector quantization. Clustering is an unsupervised learning technique that identifies similar groups in the data. The most common algorithm is *k*-means clustering, which partitions data into *k* clusters; an observation belongs to the cluster with the nearest centroid, resulting in a partition of data space into Voronoi cells.

Vector quantizers identify representative points for data that can be partitioned into a predetermined number of clusters. These points can then be used instead of the full data set so that future samples can be approximated by them. The vector quantizer $\boldsymbol{\phi}(\mathbf{x}, \mathbf{w})$ provides a mapping between the data $\mathbf{x}$ and the coordinates of the cluster centers. The loss function is usually the squared distortion of the data from the cluster centers, which must be minimized to identify the parameters of the quantizer,

$$L[\boldsymbol{\phi}(\mathbf{x}, \mathbf{w})] = ||\mathbf{x} - \boldsymbol{\phi}(\mathbf{x}, \mathbf{w})||^2. \quad 4.$$

We note that vector quantization is a data reduction method not necessarily employed for dimensionality reduction. In the latter, the learning problem seeks to identify low-dimensional features in high-dimensional data, whereas quantization amounts to finding representative clusters of the data. Vector quantization must also be distinguished from clustering, as in the former the number of desired centers is determined a priori, whereas clustering aims to identify meaningful groupings in the data. When these groupings are represented by some prototypes, then clustering and quantization have strong similarities.

2.3. Semisupervised Learning

Semisupervised learning algorithms operate under partial supervision, either with limited labeled training data or with other corrective information from the environment. Two algorithms in this category are generative adversarial networks (GANs) and RL. In both cases, the LM is (self-)trained through a game-like process discussed below.

2.3.1. Generative adversarial networks. GANs are learning algorithms that result in a generative model, i.e., a model that produces data according to a probability distribution that mimics that of the data used for its training. The LM is composed of two networks that compete with each other in a zero-sum game (Goodfellow et al. 2014). The generative network produces candidate data examples that are evaluated by the discriminative, or critic, network to optimize a certain task. The generative network's training objective is to synthesize novel examples of data to fool the discriminative network into misclassifying them as belonging to the true data distribution.

The weights of these networks are obtained through a process, inspired by game theory, called adversarial learning. The final objective of the GAN training process is to identify the generative model that produces an output that reflects the underlying system. Labeled data are provided by the discriminator network, and the function to be minimized is the Kullback–Liebler divergence between the two distributions. In the ensuing game, the discriminator aims to maximize the probability of discriminating between true data and data produced by the generator, while the generator aims to minimize the same probability. Because the generative and discriminative networks essentially train themselves, after initialization with labeled training data, this procedure is often called self-supervised. This self-training process adds to the appeal of GANs, but at the same time one must be cautious about whether an equilibrium will ever be reached in the above-mentioned game. As with other training algorithms, large amounts of data help the process, but at the moment, there is no guarantee of convergence.

2.3.2. Reinforcement learning. RL is a mathematical framework for problem solving (Sutton & Barto 2018) that implies goal-directed interactions of an agent with its environment. In RL the agent has a repertoire of actions and perceives states. Unlike in supervised learning, the agent does not have labeled information about the correct actions but instead learns from its own experiences in the form of rewards that may be infrequent and partial; thus, this is termed semisupervised learning. Moreover, the agent is concerned not only with uncovering patterns in its actions or in the environment but also with maximizing its long-term rewards. RL is closely linked to dynamic programming (Bellman 1952), as it also models interactions with the environment as a Markov decision process. Unlike dynamic programming, RL does not require a model of the dynamics, such as a Markov transition model, but proceeds by repeated interaction with the environment through trial and error. We believe that it is precisely this approximation that makes it highly suitable for complex problems in fluid dynamics. The two central elements of RL are the agent's policy, a mapping $a = \pi(s)$ between the state s of the system and the optimal action a, and the value function $V(s)$ that represents the utility of reaching the state s for maximizing the agent's long-term rewards.

Games are one of the key applications of RL that exemplify its strengths and limitations. One of the early successes of RL is the backgammon learner of Tesauro (1992). The program started out from scratch as a novice player, trained by playing a couple of million times against itself, won the computer backgammon olympiad, and eventually became comparable to the three best human players in the world. In recent years, advances in high-performance computing and deep NN architectures have produced agents that are capable of performing at or above human performance at video games and strategy games much more complicated than backgammon, such as Go (Mnih et al. 2015) and the AI gym (Mnih et al. 2015, Silver et al. 2016). It is important to emphasize that RL requires significant computational resources due to the large numbers of episodes required to properly account for the interaction of the agent and the environment. This cost may be trivial for games, but it may be prohibitive in experiments and flow simulations, a situation that is rapidly changing (Verma et al. 2018).

A core remaining challenge for RL is the long-term credit assignment (LTCA) problem, especially when rewards are sparse or very delayed in time (for example, consider the case of a perching bird or robot). LTCA implies inference, from a long sequence of states and actions, of causal relations between individual decisions and rewards. Several efforts address these issues by augmenting the original sparsely rewarded objective with densely rewarded subgoals (Schaul et al. 2015). A related issue is the proper accounting of past experience by the agent as it actively forms a new policy (Novati et al. 2019).

2.4. Stochastic Optimization: A Learning Algorithms Perspective

Optimization is an inherent part of learning, as a risk functional is minimized in order to identify the parameters of the LM. There is, however, one more link that we wish to highlight in this review: that optimization (and search) algorithms can be cast in the context of learning algorithms and more specifically as the process of learning a probability distribution that contains the design points that maximize a certain objective. This connection was pioneered by Rechenberg (1973) and Schwefel (1977), who introduced evolution strategies (ES) and adapted the variance of their search space based on the success rate of their experiments. This process is also reminiscent of the operations of selection and mutation that are key ingredients of genetic algorithms (GAs) (Holland 1975) and genetic programming (Koza 1992). ES and GAs can be considered as hybrids between gradient search strategies, which may effectively march downhill toward a minimum, and Latin hypercube or Monte Carlo sampling methods, which maximally explore the search space. Genetic programming was developed in the late 1980s by J.R. Koza, a PhD student of John Holland. Genetic programming generalized parameter optimization to function optimization, initially coded as a tree of operations (Koza 1992). A critical aspect of these algorithms is that they rely on an iterative construction of the probability distribution, based on data values of the objective function. This iterative construction can be lengthy and practically impossible for problems with expensive objective function evaluations.

Over the past 20 years, ES and GAs have begun to converge into estimation of distribution algorithms (EDAs). The covariance matrix adaptation ES (CMA-ES) algorithm (Ostermeier et al. 1994, Hansen et al. 2003) is a prominent example of ES using an adaptive estimation of the covariance matrix of a Gaussian probability distribution to guide the search for optimal parameters. This covariance matrix is adapted iteratively using the best points in each iteration. The CMA-ES is closely related to several other algorithms, such as mixed Bayesian optimization algorithms (Pelikan et al. 2004), and the reader is referred to Kern et al. (2004) for a comparative review. In recent years, this line of work has evolved into the more generalized information-geometric optimization (IGO) framework (Ollivier et al. 2017). IGO algorithms allow for families of probability distributions whose parameters are learned during the optimization process and maintain the cost function invariance as a major design principle. The resulting algorithm makes no assumption on the objective function to be optimized, and its flow is equivalent to a stochastic gradient descent. These techniques have proven to be effective on several simplified benchmark problems; however, their scaling remains unclear, and there are few guarantees for convergence in cost function landscapes such as those encountered in complex fluid dynamics problems. We note also that there is an interest in deploying these optimization methods to minimize the cost functions often associated with classical ML tasks (Salimans et al. 2017).

2.5. Important Topics We Have Not Covered: Bayesian Inference and Gaussian Processes

There are several learning algorithms that this review does not address but that demand particular attention from the fluid mechanics community. First and foremost, we wish to mention Bayesian inference, which aims to inform the model structure and its parameters from data in a probabilistic framework. Bayesian inference is fundamental for uncertainty quantification, and it is also fundamentally a learning method, as data are used to adapt the models. In fact, the alternative view is also possible, where every ML framework can be cast in a Bayesian framework (Barber 2012, Theodoridis 2015). The optimization algorithms outlined in this review provide a direct link. Whereas optimization algorithms aim to provide the best parameters of a model for given data in a stochastic manner, Bayesian inference aims to provide the full probability distribution. It

may be argued that Bayesian inference may be even more powerful than ML, as it provides probability distributions for all parameters, leading to robust predictions, rather than single values, as is usually the case with classical ML algorithms. However, a key drawback for Bayesian inference is its computational cost, as it involves sampling and integration in high-dimensional spaces, which can be prohibitive for expensive function evaluations (e.g., wind tunnel experiments or large-scale direct numerical simulation). Along the same lines, one must mention Gaussian processes (GPs), which resemble kernel-based methods for regression. However, GPs develop these kernels adaptively based on the available data. They also provide probability distributions for the respective model parameters. GPs have been used extensively in problems related to time-dependent problems, and they may be considered competitors, albeit more costly, to RNNs. Finally, we note the use of GPs as surrogates for expensive cost functions in optimization problems using ES and GAs.

3. FLOW MODELING WITH MACHINE LEARNING

First principles, such as conservation laws, have been the dominant building blocks for flow modeling over the past centuries. However, for high Reynolds numbers, scale-resolving simulations using the most prominent model in fluid mechanics, the Navier–Stokes equations, are beyond our current computational resources. An alternative is to perform simulations based on approximations of these equations (as often practiced in turbulence modeling) or laboratory experiments for a specific configuration. However, simulations and experiments are expensive for iterative optimization, and simulations are often too slow for real-time control (Brunton & Noack 2015). Consequently, considerable effort has been placed on obtaining accurate and efficient reduced-order models that capture essential flow mechanisms at a fraction of the cost (Rowley & Dawson 2016). ML provides new avenues for dimensionality reduction and reduced-order modeling in fluid mechanics by providing a concise framework that complements and extends existing methodologies.

We distinguish here two complementary efforts: dimensionality reduction and reduced-order modeling. Dimensionality reduction involves extracting key features and dominant patterns that may be used as reduced coordinates where the fluid is compactly and efficiently described (Taira et al. 2017). Reduced-order modeling describes the spatiotemporal evolution of the flow as a parametrized dynamical system, although it may also involve developing a statistical map from parameters to averaged quantities, such as drag.

There have been significant efforts to identify coordinate transformations and reductions that simplify dynamics and capture essential flow physics; the POD is a notable example (Lumley 1970). Model reduction, such as Galerkin projection of the Navier–Stokes equations onto an orthogonal basis of POD modes, benefits from a close connection to the governing equations; however, it is intrusive, requiring human expertise to develop models from a working simulation. ML provides modular algorithms that may be used for data-driven system identification and modeling. Unique aspects of data-driven modeling of fluid flows include the availability of partial prior knowledge of the governing equations, constraints, and symmetries. With advances in simulation capabilities and experimental techniques, fluid dynamics is becoming a data-rich field, thus becoming amenable to ML algorithms.

In this review, we distinguish ML algorithms to model flow (*a*) kinematics through the extraction flow features and (*b*) dynamics through the adoption of various learning architectures.

3.1. Flow Feature Extraction

Pattern recognition and data mining are core strengths of ML. Many techniques have been developed by the ML community that are readily applicable to spatiotemporal fluid data. We discuss

linear and nonlinear dimensionality reduction techniques, followed by clustering and classification. We also consider accelerated measurement and computation strategies, as well as methods to process experimental flow field data.

3.1.1. Dimensionality reduction: linear and nonlinear embeddings. A common approach in fluid dynamics simulation and modeling is to define an orthogonal linear transformation from physical coordinates into a modal basis. The POD provides such an orthogonal basis for complex geometries based on empirical measurements. Sirovich (1987) introduced the snapshot POD, which reduces the computation to a simple data-driven procedure involving a singular value decomposition. Interestingly, in the same year, Sirovich used POD to generate a low-dimensional feature space for the classification of human faces, which is a foundation for much of modern computer vision (Sirovich & Kirby 1987).

POD is closely related to the algorithm of PCA, one of the fundamental algorithms of applied statistics and ML, to describe correlations in high-dimensional data. We recall that the PCA can be expressed as a two-layer neural network, called an autoencoder, to compress high-dimensional data for a compact representation, as shown in **Figure 5**. This network embeds high-dimensional data into a low-dimensional latent space and then decodes from the latent space back to the original high-dimensional space. When the network nodes are linear and the encoder and decoder are constrained to be transposes of one another, the autoencoder is closely related to the standard POD/PCA decomposition (Baldi & Hornik 1989) (see also **Figure 6**). However, the structure of the NN autoencoder is modular, and by using nonlinear activation units for the nodes, it is possible to develop nonlinear embeddings, potentially providing more compact coordinates. This observation led to the development of one of the first applications of deep NNs to reconstruct the near-wall velocity field in a turbulent channel flow using wall pressure and shear (Milano & Koumoutsakos 2002). More powerful autoencoders are available today in the ML community, and this link deserves further exploration.

On the basis of the universal approximation theorem (Hornik et al. 1989), which states that a sufficiently large NN can represent an arbitrarily complex input–output function, deep NNs

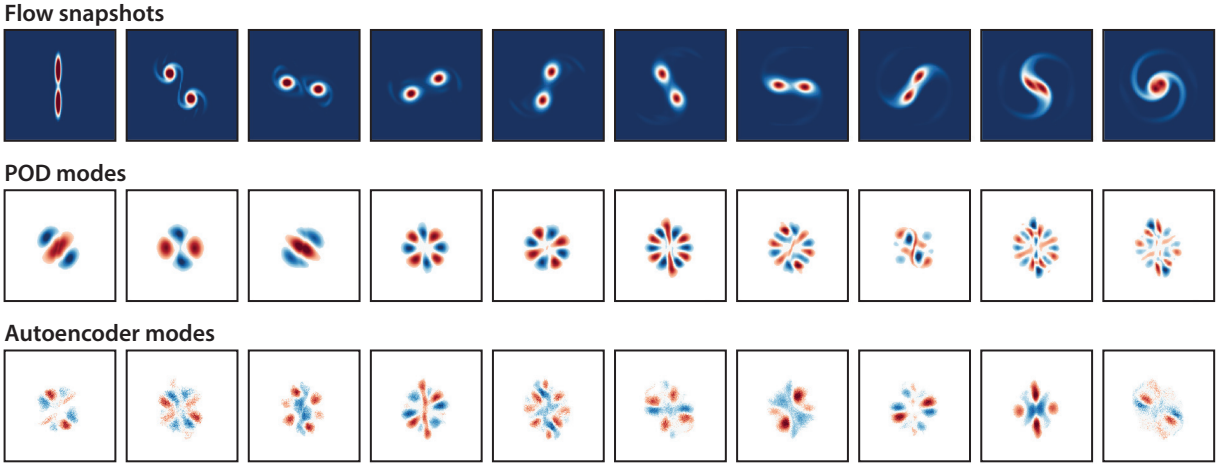

Figure 6

Unsupervised learning example: merging of two vortices (*top*), proper orthogonal decomposition (POD) modes (*middle*), and respective modes from a linear autoencoder (*bottom*). Note that unlike POD modes, the autoencoder modes are not orthogonal and are not ordered.

are increasingly used to obtain more effective nonlinear coordinates for complex flows. However, deep learning often implies the availability of large volumes of training data that far exceed the parameters of the network. The resulting models are usually good for interpolation but may not be suitable for extrapolation when the new input data have different probability distributions than the training data (see Equation 1). In many modern ML applications, such as image classification, the training data are so vast that it is natural to expect that most future classification tasks will fall within an interpolation of the training data. For example, the ImageNet data set in 2012 (Krizhevsky et al. 2012) contained over 15 million labeled images, which sparked the current movement in deep learning (LeCun et al. 2015). Despite the abundance of data from experiments and simulations, the fluid mechanics community is still distanced from this working paradigm. However, it may be possible in the coming years to curate large, labeled, and complete-enough fluid databases to facilitate the deployment of such deep learning algorithms.

Interpretability: the degree to which a model may be understood or interpreted by an expert human

3.1.2. Clustering and classification. Clustering and classification are cornerstones of ML. There are dozens of mature algorithms to choose from, depending on the size of the data and the desired number of categories. The *k*-means algorithm has been successfully employed by Kaiser et al. (2014) to develop a data-driven discretization of a high-dimensional phase space for the fluid mixing layer. This low-dimensional representation, in terms of a small number of clusters, enabled tractable Markov transition models of how the flow evolves in time from one state to another. Because the cluster centroids exist in the data space, it is possible to associate each cluster centroid with a physical flow field, lending additional interpretability. Amsallem et al. (2012) used *k*-means clustering to partition phase space into separate regions, in which local reduced-order bases were constructed, resulting in improved stability and robustness to parameter variations.

Classification is also widely used in fluid dynamics to distinguish between various canonical behaviors and dynamic regimes. Classification is a supervised learning approach where labeled data are used to develop a model to sort new data into one of several categories. Recently, Colvert et al. (2018) investigated the classification of wake topology (e.g., 2S, 2P + 2S, 2P + 4S) behind a pitching airfoil from local vorticity measurements using NNs; extensions have compared performance for various types of sensors (Alsalman et al. 2018). Wang & Hemati (2017) used the *k*-nearest-neighbors algorithm to detect exotic wakes. Similarly, NNs have been combined with dynamical systems models to detect flow disturbances and estimate their parameters (Hou et al. 2019). Related graph and network approaches in fluids by Nair & Taira (2015) have been used for community detection in wake flows (Meena et al. 2018). Finally, one of the earliest examples of ML classification in fluid dynamics by Bright et al. (2013) was based on sparse representation (Wright et al. 2009).

3.1.3. Sparse and randomized methods. In parallel to ML, there have been great strides in sparse optimization and randomized linear algebra. ML and sparse algorithms are synergistic in that underlying low-dimensional representations facilitate sparse measurements (Manohar et al. 2018) and fast randomized computations (Halko et al. 2011). Decreasing the amount of data to train and execute a model is important when a fast decision is required, as in control. In this context, algorithms for the efficient acquisition and reconstruction of sparse signals, such as compressed sensing (Donoho 2006), have already been leveraged for compact representations of wall-bounded turbulence (Bourguignon et al. 2014) and for POD-based flow reconstruction (Bai et al. 2014).

Low-dimensional structure in data also facilitates accelerated computations via randomized linear algebra (Halko et al. 2011, Mahoney 2011). If a matrix has low-rank structure, then there are efficient matrix decomposition algorithms based on random sampling; this is closely related

to the idea of sparsity and the high-dimensional geometry of sparse vectors. The basic idea is that if a large matrix has low-dimensional structure, then with high probability this structure will be preserved after projecting the columns or rows onto a random low-dimensional subspace, facilitating efficient downstream computations. These so-called randomized numerical methods have the potential to transform computational linear algebra, providing accurate matrix decompositions at a fraction of the cost of deterministic methods. For example, randomized linear algebra may be used to efficiently compute the singular value decomposition, which is used to compute PCA (Rokhlin et al. 2009, Halko et al. 2011).

Generalizability: the ability of a model to generalize to new examples including unseen data; Newton's second law, $F = ma$, is highly generalizable

3.1.4. Superresolution and flow cleansing. Much of ML is focused on imaging science, providing robust approaches to improve resolution and remove noise and corruption based on statistical inference. These superresolution and denoising algorithms have the potential to improve the quality of both simulations and experiments in fluids.

Superresolution involves the inference of a high-resolution image from low-resolution measurements, leveraging the statistical structure of high-resolution training data. Several approaches have been developed for superresolution, for example, based on a library of examples (Freeman et al. 2002), sparse representation in a library (Yang et al. 2010), and most recently CNNs (Dong et al. 2014). Experimental flow field measurements from PIV (Adrian 1991, Willert & Gharib 1991) provide a compelling application where there is a tension between local flow resolution and the size of the imaging domain. Superresolution could leverage expensive and high-resolution data on smaller domains to improve the resolution on a larger imaging domain. Large-eddy simulations (LES) (Germano et al. 1991, Meneveau & Katz 2000) may also benefit from superresolution to infer the high-resolution structure inside a low-resolution cell that is required to compute boundary conditions. Recently, Fukami et al. (2018) developed a CNN-based superresolution algorithm and demonstrated its effectiveness on turbulent flow reconstruction, showing that the energy spectrum is accurately preserved. One drawback of superresolution is that it is often extremely costly computationally, making it useful for applications where high-resolution imaging may be prohibitively expensive; however, improved NN-based approaches may drive the cost down significantly. We note also that Xie et al. (2018) recently employed GANs for superresolution.

The processing of experimental PIV and particle tracking has also been one of the first applications of ML. NNs have been used for fast PIV (Knaak et al. 1997) and PTV (Labonté 1999), with impressive demonstrations for three-dimensional Lagrangian particle tracking (Ouellette et al. 2006). More recently, deep CNNs have been used to construct velocity fields from PIV image pairs (Lee et al. 2017). Related approaches have also been used to detect spurious vectors in PIV data (Liang et al. 2003) to remove outliers and fill in corrupt pixels.

3.2. Modeling Flow Dynamics

A central goal of modeling is to balance efficiency and accuracy. When modeling physical systems, interpretability and generalizability are also critical considerations.

3.2.1. Linear models through nonlinear embeddings: dynamic mode decomposition and Koopman analysis. Many classical techniques in system identification may be considered ML, as they are data-driven models that generalize beyond the training data. Dynamic mode decomposition (DMD) (Schmid 2010, Kutz et al. 2016) is a modern approach to extract spatiotemporal coherent structures from time series data of fluid flows, resulting in a low-dimensional linear model for the evolution of these dominant coherent structures. DMD is based on data-driven regression and is equally valid for time-resolved experimental and numerical data. DMD is closely related to

the Koopman operator (Rowley et al. 2009, Mezic 2013), which is an infinite-dimensional linear operator that describes how all measurement functions of the system evolve in time. Because the DMD algorithm is based on linear flow field measurements (i.e., direct measurements of the fluid velocity or vorticity field), the resulting models may not be able to capture nonlinear transients.

Recently, there has been a concerted effort to identify a coordinate system where the nonlinear dynamics appears linear. The extended DMD (Williams et al. 2015) and variational approach of conformation dynamics (Noé & Nüske 2013, Nüske et al. 2016) enrich the model with nonlinear measurements, leveraging kernel methods (Williams et al. 2015) and dictionary learning (Li et al. 2017). These special nonlinear measurements are generally challenging to represent, and deep learning architectures are now used to identify nonlinear Koopman coordinate systems where the dynamics appear linear (Takeishi et al. 2017, Lusch et al. 2018, Mardt et al. 2018, Wehmeyer & Noé 2018). The VAMPnet architecture (Mardt et al. 2018, Wehmeyer & Noé 2018) uses a time-lagged autoencoder and a custom variational score to identify Koopman coordinates on an impressive protein folding example. Based on the performance of VAMPnet, fluid dynamics may benefit from neighboring fields, such as molecular dynamics, which have similar modeling issues, including stochasticity, coarse-grained dynamics, and separation of timescales.

3.2.2. Neural network modeling. Over the last three decades, NNs have been used to model dynamical systems and fluid mechanics problems. Early examples include the use of NNs to learn the solutions of ordinary and partial differential equations (Dissanayake & Phan-Thien 1994, Gonzalez-Garcia et al. 1998, Lagaris et al. 1998). We note that the potential of these works has not been fully explored, and in recent years there have been further advances (Chen et al. 2018, Raissi & Karniadakis 2018), including discrete and continuous-in-time networks. We note also the possibility of using these methods to uncover latent variables and reduce the number of parametric studies often associated with partial differential equations (Raissi et al. 2019). NNs are also frequently employed in nonlinear system identification techniques such as NARMAX, which are often used to model fluid systems (Glaz et al. 2010, Semeraro et al. 2016). In fluid mechanics, NNs were widely used to model heat transfer (Jambunathan et al. 1996), turbomachinery (Pierret & Van den Braembussche 1999), turbulent flows (Milano & Koumoutsakos 2002), and other problems in aeronautics (Faller & Schreck 1996).

RNNs with LSTMs (Hochreiter & Schmidhuber 1997) have been revolutionary for speech recognition, and they are considered one of the landmark successes of AI. They are currently being used to model dynamical systems and for data-driven predictions of extreme events (Vlachas et al. 2018, Wan et al. 2018). An interesting finding of these studies is that combining data-driven and reduced-order models is a potent method that outperforms each of its components on several studies. GANs (Goodfellow et al. 2014) are also being used to capture physics (Wu et al. 2018). GANs have potential to aid in the modeling and simulation of turbulence (Kim et al. 2018), although this field is nascent.

Despite the promise and widespread use of NNs in dynamical systems, several challenges remain. NNs are fundamentally interpolative, and so the function is well approximated only in the span (or under the probability distribution) of the sampled data used to train them. Thus, caution should be exercised when using NN models for an extrapolation task. In many computer vision and speech recognition examples, the training data are so vast that nearly all future tasks may be viewed as an interpolation on the training data, although this scale of training has not been achieved to date in fluid mechanics. Similarly, NN models are prone to overfitting, and care must be taken to cross-validate models on a sufficiently chosen test set; best practices are discussed by Goodfellow et al. (2016). Finally, it is important to explicitly incorporate partially known physics, such as symmetries, constraints, and conserved quantities.

3.2.3. Parsimonious nonlinear models. Parsimony is a recurring theme in mathematical physics, from Hamilton's principle of least action to the apparent simplicity of many governing equations. In contrast to the raw representational power of NNs, ML algorithms are also being employed to identify minimal models that balance predictive accuracy with model complexity, preventing overfitting and promoting interpretability and generalizability. Genetic programming was recently used to discover conservation laws and governing equations (Schmidt & Lipson 2009). Sparse regression in a library of candidate models has also been proposed to identify dynamical systems (Brunton et al. 2016) and partial differential equations (Rudy et al. 2017, Schaeffer 2017). Loiseau & Brunton (2018) identified sparse reduced-order models of several flow systems, enforcing energy conservation as a constraint. In both genetic programming and sparse identification, a Pareto analysis is used to identify models that have the best trade-off between model complexity, measured in number of terms, and predictive accuracy. In cases where the physics is known, this approach typically discovers the correct governing equations, providing exceptional generalizability compared with other leading algorithms in ML.

3.2.4. Closure models with machine learning. The use of ML to develop turbulence closures is an active area of research (Duraisamy et al. 2019). The extreme separation of spatiotemporal scales in turbulent flows makes it exceedingly costly to resolve all scales in simulation, and even with Moore's law, we are decades away from resolving all scales in relevant configurations (e.g., aircraft, submarines, etc.). It is common to truncate small scales and model their effect on the large scales with a closure model. Common approaches include Reynolds-averaged Navier–Stokes (RANS) and LES. However, these models may require careful tuning to match data from fully resolved simulations or experiments.

ML has been used to identify and model discrepancies in the Reynolds stress tensor between a RANS model and high-fidelity simulations (Ling & Templeton 2015, Parish & Duraisamy 2016, Ling et al. 2016b, Xiao et al. 2016, Singh et al. 2017, Wang et al. 2017). Ling & Templeton (2015) compared SVMs, Adaboost decision trees, and random forests to classify and predict regions of high uncertainty in the Reynolds stress tensor. Wang et al. (2017) used random forests to build a supervised model for the discrepancy in the Reynolds stress tensor. Xiao et al. (2016) leveraged sparse online velocity measurements in a Bayesian framework to infer these discrepancies. In related work, Parish & Duraisamy (2016) developed the field inversion and ML modeling framework that builds corrective models based on inverse modeling. This framework was later used by Singh et al. (2017) to develop an NN enhanced correction to the Spalart–Allmaras RANS model, with excellent performance. A key result by Ling et al. (2016b) employed the first deep network architecture with many hidden layers to model the anisotropic Reynolds stress tensor, as shown in **Figure 7**. Their novel architecture incorporates a multiplicative layer to embed Galilean invariance into the tensor predictions. This provides an innovative and simple approach to embed known physical symmetries and invariances into the learning architecture (Ling et al. 2016a), which we believe will be essential in future efforts that combine learning for physics. For LES closures, Maulik et al. (2019) have employed artificial NNs to predict the turbulence source term from coarsely resolved quantities.

3.2.5. Challenges of machine learning for dynamical systems. Applying ML to model physical dynamical systems poses several unique challenges and opportunities. Model interpretability and generalizability are essential cornerstones in physics. A well-crafted model will yield hypotheses for phenomena that have not been observed before. This principle is for example exhibited in the parsimonious formulation of classical mechanics in Newton's second law.

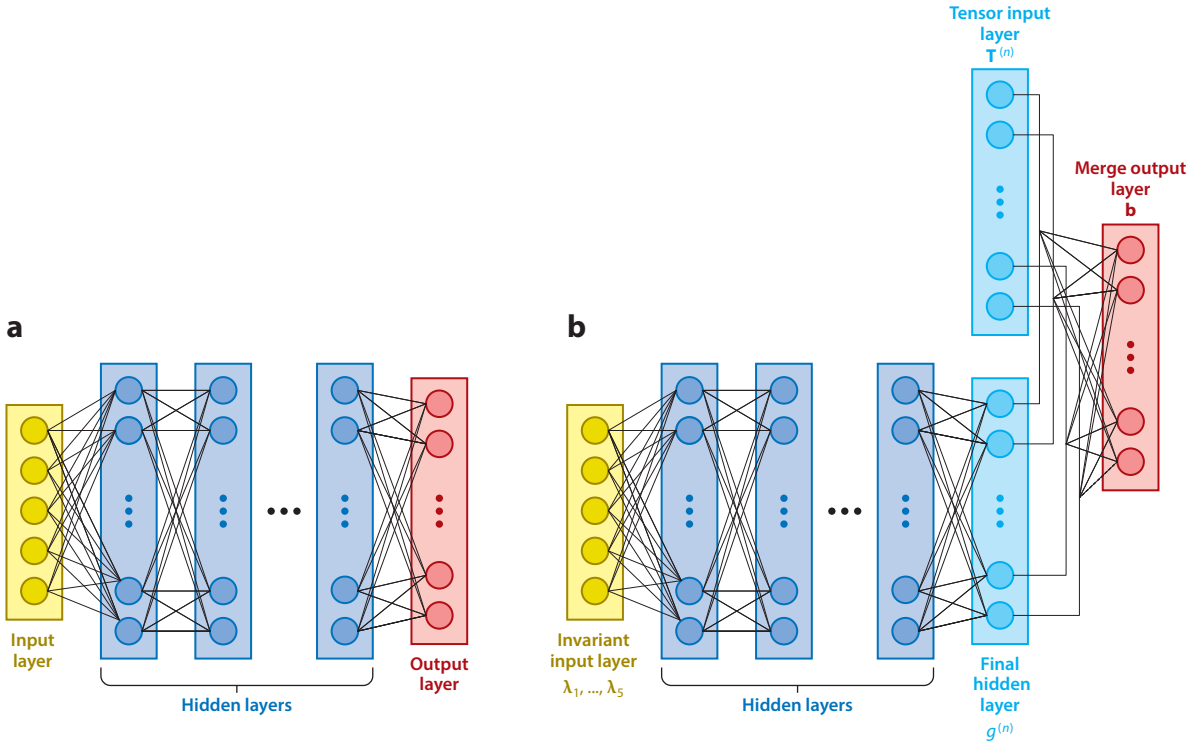

Figure 7

Comparison of standard neural network architecture (*a*) with modified neural network for identifying Galilean invariant Reynolds stress models (*b*). Abbreviations: **b**, anisotropy tensor; $g^{(n)}$, scalar coefficients weighing the basis tensors; $\mathbf{T}^{(n)}$, isotropic basis tensors; $\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_5$, five tensor invariants. Figure adapted with permission from Ling et al. (2016b).

High-dimensional systems, such as those encountered in unsteady fluid dynamics, have the challenges of multiscale dynamics, sensitivity to noise and disturbances, latent variables, and transients, all of which require careful attention when applying ML techniques. In ML for dynamics, we distinguish two tasks: discovering unknown physics and improving models by incorporating known physics. Many learning architectures cannot readily incorporate physical constraints in the form of symmetries, boundary conditions, and global conservation laws. This is a critical area for continued development, and several recent works have presented generalizable physics models (Battaglia et al. 2018).

4. FLOW OPTIMIZATION AND CONTROL USING MACHINE LEARNING

Learning algorithms are well suited to flow optimization and control problems involving black-box or multimodal cost functions. These algorithms are iterative and often require several orders of magnitude more cost function evaluations than gradient-based algorithms (Bewley et al. 2001). Moreover, they do not offer guarantees of convergence, and we suggest that they be avoided when techniques such as adjoint methods are applicable. At the same time, techniques such as RL have been shown to outperform even optimal flow control strategies (Novati et al. 2019). Indeed, there are several classes of flow control and optimization problems where learning algorithms may be the method of choice, as described below.

OPTIMIZATION AND CONTROL: BOUNDARIES ERASED BY FAST COMPUTERS

Optimization and control are intimately related, and the boundaries are becoming even less distinct with increasingly fast computers, as summarized by Tsiotras & Mesbahi (2017, p. 195):

> Interestingly, the distinction between optimization and control is largely semantic and (alas!) implementation-dependent. If one has the capability of solving optimization problems fast enough on the fly to close the loop, then one has (in principle) a feedback control law... Not surprisingly then, the same algorithm can be viewed as solving an optimization or a control problem, based solely on the capabilities of the available hardware. With the continued advent of faster and more capable computer hardware architectures, the boundary between optimization and control will become even more blurred. However, when optimization is embedded in the implementation of feedback control, the classical problems of control such as robustness to model uncertainty, time delays, and process and measurement noise become of paramount importance, particularly for high-performance aerospace systems.

In contrast to flow modeling, learning algorithms for optimization and control interact with the data sampling process in several ways. First, in line with the modeling efforts described in earlier sections, ML can be applied to develop explicit surrogate models that relate the cost function and the control/optimization parameters. Surrogate models such as NNs can then be amenable to even gradient-based methods, although they often get stuck in local minima. Multifidelity algorithms (Perdikaris et al. 2016) can also be employed to combine surrogates with the cost function of the complete problem. As the learning progresses, new data are requested as guided by the results of the optimization. Alternatively, the optimization or control problem may be described in terms of learning probability distributions of parameters that minimize the cost function. These probability distributions are constructed from cost function samples obtained during the optimization process. Furthermore, the high-dimensional and nonconvex optimization procedures that are currently employed to train nonlinear LMs are well suited to the high-dimensional, nonlinear optimization problems in flow control.

We remark that the lines between optimization and control are becoming blurred by the availability of powerful computers (see the sidebar titled Optimization and Control: Boundaries Erased by Fast Computers). However, the range of critical spatiotemporal scales and the nonlinearity of the underlying processes will likely render real-time optimization for flow control a challenge for decades to come.

4.1. Stochastic Flow Optimization: Learning Probability Distributions

Stochastic optimization includes ES and GAs, which were originally developed based on bio-inspired principles. However, in recent years these algorithms have been placed in a learning framework (Kern et al. 2004).

Stochastic optimization has found widespread use in engineering design, in particular as many engineering problems involve black-box-type cost functions. A much-abbreviated list of applications includes aerodynamic shape optimization (Giannakoglou et al. 2006), uninhabited aerial vehicles (UAVs) (Hamdaoui et al. 2010), shape and motion optimization in artificial swimmers (Gazzola et al. 2012, Van Rees et al. 2015), and improved power extraction in crossflow turbines (Strom et al. 2017). We refer readers to the review article by Skinner & Zare-Behtash (2018) for an extensive comparison of gradient-based and stochastic optimization algorithms for aerodynamics.

These algorithms involve large numbers of iterations, and they can benefit from massively parallel computer architectures. Advances in automation have also facilitated their application in experimental (Strom et al. 2017, Martin & Gharib 2018) and industrial settings (Bueche et al.

2002). We note that stochastic optimization algorithms are well suited to address the experimental and industrial challenges associated with uncertainty, such as unexpected system behavior, partial descriptions of the system and its environment, and exogenous disturbances. Hansen et al. (2009) proposed an approach to enhance the capabilities of evolutionary algorithms for online optimization of a combustor test rig.

Stochastic flow optimization will continue to benefit from advances in computer hardware and experimental techniques. At the same time, convergence proofs, explainability, and reliability are outstanding issues that need to be taken into consideration when deploying such algorithms in fluid mechanics problems. Hybrid algorithms that combine in a problem-specific manner stochastic techniques and gradient-based methods may offer the best strategy for flow control problems.

4.2. Flow Control with Machine Learning

Feedback flow control modifies the behavior of a fluid dynamic system through actuation that is informed by sensor measurements. Feedback is necessary to stabilize an unstable system, attenuate sensor noise, and compensate for external disturbances and model uncertainty. Challenges of flow control include a high-dimensional state, nonlinearity, latent variables, and time delays. ML algorithms have been used extensively in control, system identification, and sensor placement.

4.2.1. Neural networks for control. NNs have received significant attention for system identification (see Section 3) and control, including applications in aerodynamics (Phan et al. 1995). The application of NNs to turbulence flow control was pioneered by Lee et al. (1997). The skin-friction drag of a turbulent boundary layer was reduced using local wall-normal blowing and suction based on few skin-friction sensors. A sensor-based control law was learned from a known optimal full-information controller, with little loss in overall performance. Furthermore, a single-layer network was optimized for skin-friction drag reduction without incorporating any prior knowledge of the actuation commands. Both strategies led to a conceptually simple local opposition control. Several other studies employ NNs, e.g., for phasor control (Rabault et al. 2019) or even frequency cross-talk. The need to optimize many parameters is the price for the theoretical advantage of approximating arbitrary nonlinear control laws. NN control may require exorbitant computational or experimental resources for configurations with complex high-dimensional nonlinearities and many sensors and actuators. At the same time, the training time of NNs has been improved by several orders of magnitude since these early applications, which warrant further investigation into their potential for flow control.

4.2.2. Genetic algorithms for control. GAs have been deployed to solve several flow control problems. They require that the structure of the control law be prespecified and contain only a few adjustable parameters. An example of the use of GA for control design in fluids was used for experimental mixing optimization of the backward-facing step (Benard et al. 2016). As with NN control, the learning time increases with the number of parameters, making it challenging or even prohibitive for controllers with nonlinearities (e.g., a constant-linear-quadratic law), signal history (e.g., a Kalman filter), or multiple sensors and actuators.

Genetic programming has been used extensively in active control for engineering applications (Dracopoulos 1997, Fleming & Purshouse 2002) and in recent years in several flow control plants. This includes the learning of multifrequency open-loop actuation, multi-input sensor feedback, and distributed control. We refer readers to Duriez et al. (2016) for an in-depth description of the method and to Noack (2018) for an overview of the plants. We remark that most control laws have been obtained within 1,000 test evaluations, each requiring only a few seconds in a wind tunnel.

4.3. Flow Control via Reinforcement Learning

In recent years, RL has advanced beyond the realm of games and has become a fundamental mode of problem solving in a growing number of domains, including to reproduce the dynamics of hydrological systems (Loucks et al. 2005), actively control the oscillatory laminar flow around bluff bodies (Guéniat et al. 2016), study the individual (Gazzola et al. 2014) or collective motion of fish (Gazzola et al. 2016, Novati et al. 2017, Verma et al. 2018), maximize the range of simulated (Reddy et al. 2016) and robotic (Reddy et al. 2018) gliders, optimize the kinematic motion of UAVs (Kim et al. 2004, Tedrake et al. 2009), and optimize the motion of microswimmers (Colabrese et al. 2017, 2018). **Figure 8** provides a schematic of RL with compelling examples related to fluid mechanics.

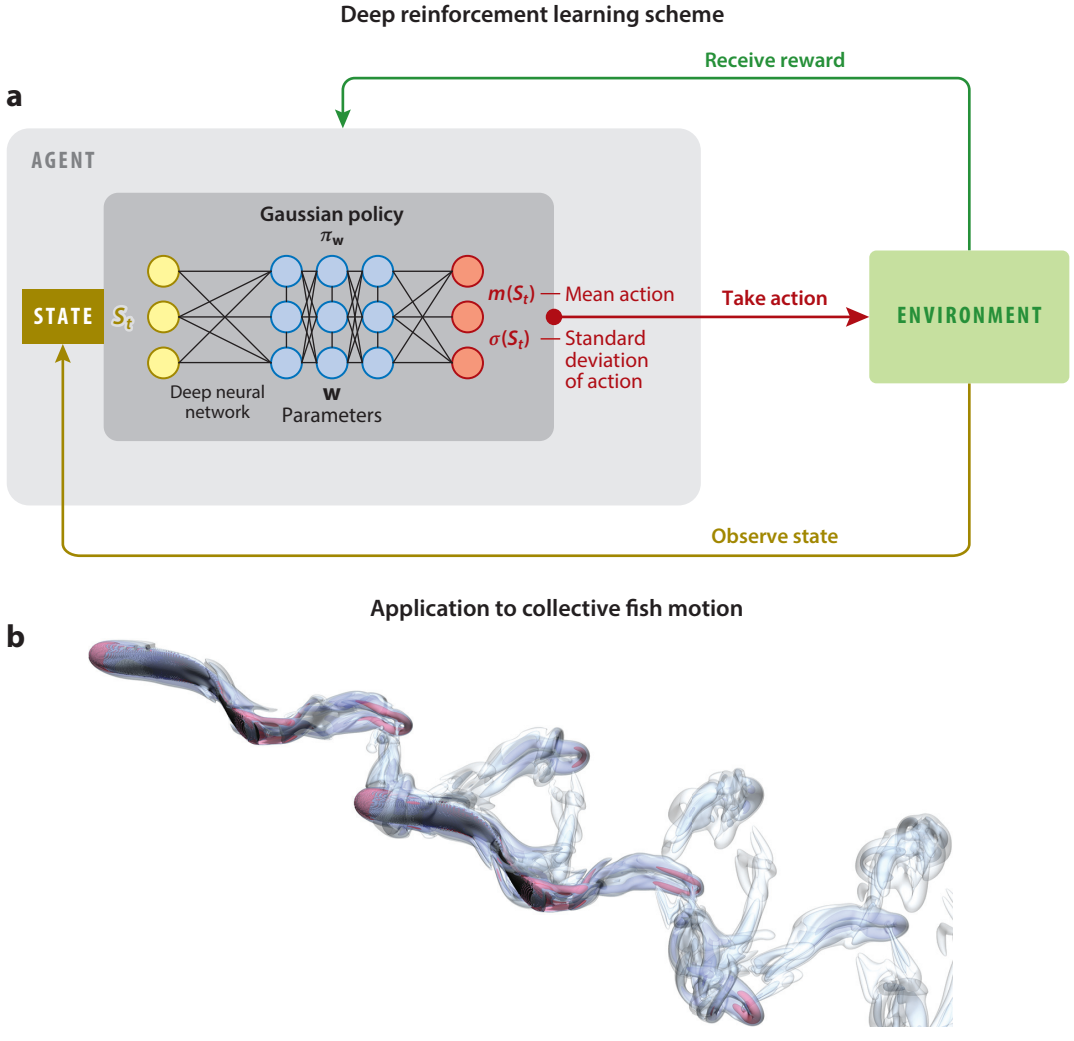

Figure 8

Deep reinforcement learning schematic (*a*) and application to the study of the collective motion of fish via the Navier–Stokes equations (*b*). Panel *b* adapted from Verma et al. (2018).

Fluid mechanics knowledge is essential for applications of RL, as success or failure hinges on properly selecting states, actions, and rewards that reflect the governing mechanisms of the flow problem. Natural organisms and their sensors, such as the visual system in a bird or the lateral line in a fish, can guide the choice of states. As sensor technologies progress at a rapid pace, the algorithmic challenge may be that of optimal sensor placement (Papadimitriou & Papadimitriou 2015, Manohar et al. 2018). The actions reflect the flow actuation device and may involve body deformation or wing flapping. Rewards may include energetic factors, such as the cost of transport or proximity to the center of a fish school to avoid predation. The computational cost of RL remains a challenge to its widespread adoption, but we believe this deficiency can be mediated by the parallelism inherent in RL. There is growing interest in methods designed to be transferable from low-accuracy (e.g., two-dimensional) to high-accuracy (e.g., three-dimensional) simulations (Verma et al. 2018) or from simulations to related real-world applications (Richter et al. 2016, Bousmalis et al. 2017).

5. DISCUSSION AND OUTLOOK

This review presents ML algorithms for the perspective of fluid mechanics. The interface of the two fields has a long history and has attracted a renewed interest in the last few years. This review addresses applications of ML in problems of flow modeling, optimization, and control in experiments and simulations. It highlights some successes of ML in critical fluid mechanics tasks, such as dimensionality reduction, feature extraction, PIV processing, superresolution, reduced-order modeling, turbulence closure, shape optimization, and flow control. It discusses lessons learned from these efforts and justifies the current interest in light of the technological advances of our times. Our goal is to provide a deeper understanding of ML and its context in fluid mechanics. ML comprises data-driven optimization and applied regression techniques that are well suited for high-dimensional, nonlinear problems, such as those encountered in fluid dynamics; fluid mechanics expertise will be necessary to formulate these optimization and regression problems.

ML algorithms present an arsenal of tools, largely unexplored in fluid mechanics research, that can augment existing modes of inquiry. Fluid mechanics knowledge and centuries-old conservation laws remain relevant in the era of big data. Such knowledge can help frame more precise questions and assist in reducing the large computational cost often associated with the application of ML algorithms in flow control and optimization. The exploration and visualization of high-dimensional search spaces can be simplified by ML and increasingly abundant high-performance computing resources.

In the near future, experience with ML algorithms will help frame new questions in fluid mechanics, extending decades-old linearized models and linear approaches to the nonlinear regime. The transition to the nonlinear realm of ML is facilitated by the abundance of open source software and methods and the prevalent openness of the ML community. In the long term, ML will undoubtedly offer a fresh look into old problems of fluid mechanics under the light of data. Interpreting the ML solutions, and refining the problem statement, will again require fluid mechanics expertise.

A word of caution is necessary to balance the current excitement about data-driven research and the (almost magical) powers of ML. After all, an ML algorithm will always provide some kind of answer to any question that is based on its training data—data that may not even be relevant to the question at hand. Properly formulating the question and selecting the data, the LM, and its training are all critical components of the learning process. Applying ML algorithms to fluid

mechanics faces numerous outstanding challenges (and opportunities!). Although many fields of ML are concerned with raw predictive performance, applications in fluid mechanics often require models that are explainable and generalizable and have guarantees.

Reproducibility: the process of documenting procedures and archiving code and data so that others can fully reproduce scientific results

Although deep learning will undoubtedly become a critical tool in several aspects of flow modeling, not all ML is deep learning. It is important to consider several factors when choosing methods, including the quality and quantity of data, the desired inputs and outputs, the cost function to be optimized, whether the task involves interpolation or extrapolation, and how important it is for models to be explainable. It is also important to cross-validate ML models; otherwise, results may be prone to overfitting. It is also important to develop and adapt ML algorithms that are not only physics informed but also physics consistent, a major outstanding challenge in AI. This review concludes with a call for action in the fluid mechanics community to further embrace open and reproducible research products and standards. Reproducibility is a cornerstone of science, and several frameworks are currently developed to render this into a systematic scientific process (Barber 2015). It is increasingly possible to document procedures, archive code, and host data so that others can reproduce results. Data are essential for ML; thus, creating and curating benchmark data sets and software will spur interest among researchers in related fields, driving progress. These fluid benchmarks are more challenging than the traditional image data sets encountered in ML: Fluid data are multimodal and multifidelity, they have high resolution in some dimensions and are sparse in others, many tasks balance multiple objectives, and foremost, data come from dynamical systems, where many tasks do not admit postmortem analysis.

We are entering a new and exciting era in fluid mechanics research. Centuries of theoretical developments based on first principles are now merging with data-driven analysis. This fusion will provide solutions to many long-sought problems in fluid dynamics, first and foremost the enhanced understanding of its governing mechanisms.

SUMMARY POINTS

1. Machine learning (ML) entails powerful information-processing algorithms that are relevant for modeling, optimization, and control of fluids. Effective problem solvers will have expertise in ML and in-depth knowledge of fluid mechanics.
2. Fluid mechanics has been traditionally concerned with big data. For decades it has used ML to understand, predict, optimize, and control flows. Currently, ML capabilities are advancing at an incredible rate, and fluid mechanics is beginning to tap into the full potential of these powerful methods.
3. Many tasks in fluid mechanics, such as reduced-order modeling, shape optimization, and feedback control, may be posed as optimization and regression tasks. ML can improve optimization performance and reduce convergence time. ML is also used for dimensionality reduction and identifying low-dimensional manifolds and discrete flow regimes, which benefit understanding.
4. Flow control strategies have been traditionally based on the precise sequence from understanding to modeling and then to control. The ML paradigm suggests more flexibility and iterates between data-driven and first principle approaches.

FUTURE ISSUES

1. ML algorithms often come without guarantees for performance, robustness, or convergence, even for well-defined tasks. How can interpretability, generalizability, and explainability of the results be achieved?
2. Incorporating and enforcing known flow physics is a challenge and opportunity for ML algorithms. Can we hybridize data-driven and first principle approaches in fluid mechanics?
3. There are many possibilities to discover new physical mechanisms, symmetries, constraints, and invariances from fluids data.
4. Data-driven modeling may be a potent alternative in revisiting existing empirical laws in fluid mechanics.
5. ML encourages open sharing of data and software. Can this assist the development of frameworks for reproducible and open science in fluid mechanics?
6. Fluids researchers will benefit from interfacing with the ML community, where the latest advances are reported at peer-reviewed conferences.

DISCLOSURE STATEMENT

The authors are not aware of any biases that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGMENTS

S.L.B. acknowledges funding from the Army Research Office (ARO W911NF-17-1-0306, W911NF-17-1-0422) and the Air Force Office of Scientific Research (AFOSR FA9550-18-1-0200). B.R.N. acknowledges funding from LIMSI-CNRS, Université Paris Sud (SMEMaG), the French National Research Agency (ANR-11-IDEX-0003-02, ANR-17-ASTR-0022), and the German Research Foundation (CRC880, SE 2504/2-1, SE 2504/3-1). P.K. acknowledges funding from an ERC Advanced Investigator Award (FMCoBe, No. 34117), the Swiss National Science Foundation, and the Swiss National Supercomputing Centre. The authors are grateful for discussions with Nathan Kutz (University of Washington), Jean-Christophe Loiseau (ENSAM ParisTech, Paris), François Lusseyran (LIMSI-CNRS, Paris), Guido Novati (ETH Zurich), Luc Pastur (ENSTA ParisTech, Paris), and Pantelis Vlachas (ETH Zurich).

LITERATURE CITED

Adrian RJ. 1991. Particle-imaging techniques for experimental fluid mechanics. *Annu. Rev. Fluid Mech.* 23:261–304

Alsalman M, Colvert B, Kanso E. 2018. Training bioinspired sensors to classify flows. *Bioinspiration Biomim.* 14:016009

Amsallem D, Zahr MJ, Farhat C. 2012. Nonlinear model order reduction based on local reduced-order bases. *Int. J. Numer. Meth. Eng.* 92:891–916

Bai Z, Wimalajeewa T, Berger Z, Wang G, Glauser M, Varshney PK. 2014. Low-dimensional approach for reconstruction of airfoil data via compressive sensing. *AIAA J.* 53:920–33

Baldi P, Hornik K. 1989. Neural networks and principal component analysis: learning from examples without local minima. *Neural Netw.* 2:53–58

Barber D. 2012. *Bayesian Reasoning and Machine Learning*. Cambridge, UK: Cambridge Univ. Press

Reproducible science: a framework.

Barber RF, Candes EJ. 2015. Controlling the false discovery rate via knock-offs. *Ann. Stat.* 43:2055–85

Battaglia PW, Hamrick JB, Bapst V, Sanchez-Gonzalez A, Zambaldi V, et al. 2018. Relational inductive biases, deep learning, and graph networks. arXiv:1806.01261 [cs.LG]

Bellman R. 1952. On the theory of dynamic programming. *PNAS* 38:716–19

Benard N, Pons-Prats J, Periaux J, Bugeda G, Braud P, et al. 2016. Turbulent separated shear flow control by surface plasma actuator: experimental optimization by genetic algorithm approach. *Exp. Fluids* 57:22

Bewley TR, Moin P, Temam R. 2001. DNS-based predictive control of turbulence: an optimal benchmark for feedback algorithms. *J. Fluid Mech.* 447:179–225

Bishop CM, James GD. 1993. Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nucl. Instrum. Methods Phys. Res.* 327:580–93

Theoretical analysis of the approximation properties of deep neural networks.

Bölcskei H, Grohs P, Kutyniok G, Petersen P. 2019. Optimal approximation with sparsely connected deep neural networks. *SIAM J. Math. Data Sci.* 1:8–45

Bourguignon JL, Tropp JA, Sharma AS, McKeon BJ. 2014. Compact representation of wall-bounded turbulence using compressive sampling. *Phys. Fluids* 26:015109

Bousmalis K, Irpan A, Wohlhart P, Bai Y, Kelcey M, et al. 2017. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. arXiv:1709.07857 [cs.LG]

Breiman L. 2001. Random forests. *Mach. Learn.* 45:5–32

Bright I, Lin G, Kutz JN. 2013. Compressive sensing based machine learning strategies for characterizing the flow around a cylinder with limited pressure measurements. *Phys. Fluids* 25:127102

Brunton SL, Kutz JN. 2019. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. New York: Cambridge Univ. Press

Brunton SL, Noack BR. 2015. Closed-loop turbulence control: progress and challenges. *Appl. Mech. Rev.* 67:050801

Brunton SL, Proctor JL, Kutz JN. 2016. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *PNAS* 113:3932–37

Bueche D, Stoll P, Dornberger R, Koumoutsakos P. 2002. Multi-objective evolutionary algorithm for the optimization of noisy combustion problems. *IEEE Trans. Syst. Man Cybern. C* 32:460–73

Chen TQ, Rubanova Y, Bettencourt J, Duvenaud DK. 2018. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems 31*, ed. S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, R Garnett, pp. 6571–83. Red Hook, NY: Curran Assoc.

Cherkassky V, Mulier FM. 2007. *Learning from Data: Concepts, Theory, and Methods*. Hoboken, NJ: John Wiley & Sons

Colabrese S, Gustavsson K, Celani A, Biferale L. 2017. Flow navigation by smart microswimmers via reinforcement learning. *Phys. Rev. Lett.* 118:158004

Colabrese S, Gustavsson K, Celani A, Biferale L. 2018. Smart inertial particles. *Phys. Rev. Fluids* 3:084301

Colvert B, Alsalman M, Kanso E. 2018. Classifying vortex wakes using neural networks. *Bioinspiration Biomim.* 13:025003

Dissanayake M, Phan-Thien N. 1994. Neural-network-based approximations for solving partial differential equations. *Comm. Numer. Meth. Eng.* 10:195–201

Dong C, Loy CC, He K, Tang X. 2014. Learning a deep convolutional network for image super-resolution. In *Computer Vision—ECCV 2014*, ed. D Fleet, T Pajdla, B Schiele, T Tuytelaars, pp. 184–99. Cham, Switz.: Springer

Donoho DL. 2006. Compressed sensing. *IEEE Trans. Inf. Theory* 52:1289–306

Dracopoulos DC. 1997. *Evolutionary Learning Algorithms for Neural Adaptive Control*. London: Springer-Verlag

Duraisamy K, Iaccarino G, Xiao H. 2019. Turbulence modeling in the age of data. *Annu. Rev. Fluid Mech.* 51:357–77

Duriez T, Brunton SL, Noack BR. 2016. *Machine Learning Control: Taming Nonlinear Dynamics and Turbulence*. Cham, Switz.: Springer

Faller WE, Schreck SJ. 1996. Neural networks: applications and opportunities in aeronautics. *Prog. Aerosp. Sci.* 32:433–56

Fleming PJ, Purshouse RC. 2002. Evolutionary algorithms in control systems engineering: a survey. *Control Eng. Pract.* 10:1223–41

Freeman WT, Jones TR, Pasztor EC. 2002. Example-based super-resolution. *IEEE Comput. Graph.* 22:56–65

Fukami K, Fukagata K, Taira K. 2018. Super-resolution reconstruction of turbulent flows with machine learning. arXiv:1811.11328 [physics.flu-dyn]

Gardner E. 1988. The space of interactions in neural network models. *J. Phys. A* 21:257

Gazzola M, Hejazialhosseini B, Koumoutsakos P. 2014. Reinforcement learning and wavelet adapted vortex methods for simulations of self-propelled swimmers. *SIAM J. Sci. Comput.* 36:B622–39

Gazzola M, Tchieu A, Alexeev D, De Brauer A, Koumoutsakos P. 2016. Learning to school in the presence of hydrodynamic interactions. *J. Fluid Mech.* 789:726–49

Gazzola M, Van Rees WM, Koumoutsakos P. 2012. C-start: optimal start of larval fish. *J. Fluid Mech.* 698:5–18

Germano M, Piomelli U, Moin P, Cabot WH. 1991. A dynamic subgrid-scale eddy viscosity model. *Phys. Fluids* 3:1760–65

Giannakoglou K, Papadimitriou D, Kampolis I. 2006. Aerodynamic shape design using evolutionary algorithms and new gradient-assisted metamodels. *Comput. Methods Appl. Mech. Eng.* 195:6312–29

Glaz B, Liu L, Friedmann PP. 2010. Reduced-order nonlinear unsteady aerodynamic modeling using a surrogate-based recurrence framework. *AIAA J.* 48:2418–29

Gonzalez-Garcia R, Rico-Martinez R, Kevrekidis I. 1998. Identification of distributed parameter systems: a neural net based approach. *Comput. Chem. Eng.* 22:S965–68

Goodfellow I, Bengio Y, Courville A. 2016. *Deep Learning*. Cambridge, MA: MIT Press

Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, et al. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, ed. Z Ghahramani, M Welling, C Cortes, ND Lawrence, KQ Weinberger, pp. 2672–80. Red Hook, NY: Curran Assoc.

Powerful deep learning architecture that learns through a game between a network that can generate new data and a network that is an expert classifier.

Grant I, Pan X. 1995. An investigation of the performance of multi layer, neural networks applied to the analysis of PIV images. *Exp. Fluids* 19:159–66

Graves A, Fernández S, Schmidhuber J. 2007. Multi-dimensional recurrent neural networks. In *Artificial Neural Networks—ICANN 2007*, ed. JM de Sa, LA Alexandre, W Duch, D Mandic, pp. 549–58. Berlin: Springer

Grossberg S. 1976. Adaptive pattern classification and universal recoding, I: parallel development and coding of neural feature detectors. *Biol. Cybernet.* 23:121–34

Grossberg S. 1988. Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural Netw.* 1:17–61

Guéniat F, Mathelin L, Hussaini MY. 2016. A statistical learning strategy for closed-loop control of fluid flows. *Theor. Comput. Fluid Dyn.* 30:497–510

Halko N, Martinsson PG, Tropp JA. 2011. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.* 53:217–88

Hamdaoui M, Chaskalovic J, Doncieux S, Sagaut P. 2010. Using multiobjective evolutionary algorithms and data-mining methods to optimize ornithopters' kinematics. *J. Aircraft* 47:1504–16

Hansen N, Müller SD, Koumoutsakos P. 2003. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evol. Comput.* 11:1–18

Hansen N, Niederberger AS, Guzzella L, Koumoutsakos P. 2009. A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. *IEEE Trans. Evol. Comput.* 13:180–97

Hastie T, Tibshirani R, Friedman J, Hastie T, Friedman J, Tibshirani R. 2009. *The Elements of Statistical Learning*, Vol. 2. New York: Springer

Hinton GE, Sejnowski TJ. 1986. Learning and relearning in Boltzmann machines. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, ed. DE Rumelhart, J McClelland, pp. 282–317. Cambridge, MA: MIT Press

Hochreiter S, Schmidhuber J. 1997. Long short-term memory. *Neural Comput.* 9:1735–80

Regularization of recurrent neural networks and major contributor to the success of Google Translate.

Holland JH. 1975. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Ann Arbor: Univ. Mich. Press

Hopfield JJ. 1982. Neural networks and physical systems with emergent collective computational abilities. *PNAS* 79:2554–58

Hornik K, Stinchcombe M, White H. 1989. Multilayer feedforward networks are universal approximators. *Neural Netw.* 2:359–66

Hou W, Darakananda D, Eldredge J. 2019. *Machine learning based detection of flow disturbances using surface pressure measurements*. Paper presented at AIAA Atmospheric Flight Mechanics Conference 2019, San Diego, Calif., AIAA Pap. 2019-1148

Jambunathan K, Hartle S, Ashforth-Frost S, Fontama V. 1996. Evaluating convective heat transfer coefficients using neural networks. *Int. J. Heat Mass Transf.* 39:2329–32

Kaiser E, Noack BR, Cordier L, Spohn A, Segond M, et al. 2014. Cluster-based reduced-order modelling of a mixing layer. *J. Fluid Mech.* 754:365–414

Kern S, Müller SD, Hansen N, Büche D, Ocenasek J, Koumoutsakos P. 2004. Learning probability distributions in continuous evolutionary algorithms—a comparative review. *Nat. Comput.* 3:77–112

Kim B, Azevedo VC, Thuerey N, Kim T, Gross M, Solenthaler B. 2018. Deep fluids: a generative network for parameterized fluid simulations. arXiv:1806.02071 [cs.LG]

Kim HJ, Jordan MI, Sastry S, Ng AY. 2004. Autonomous helicopter flight via reinforcement learning. In *Advances in Neural Information Processing Systems 17*, ed. B Scholkopf, Y Vis, JC Platt, pp. 799–806. Cambridge, MA: MIT Press

Knaak M, Rothlubbers C, Orglmeister R. 1997. A Hopfield neural network for flow field computation based on particle image velocimetry/particle tracking velocimetry image sequences. *IEEE Int. Conf. Neural Netw.* 1:48–52

Kohonen T. 1995. *Self-Organizing Maps*. Berlin: Springer-Verlag

Kolmogorov A. 1941. The local structure of turbulence in incompressible viscous fluid for very large Reynolds number. *Cr. Acad. Sci. USSR* 30:301–5

Koza JR. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Boston: MIT Press

Krizhevsky A, Sutskever I, Hinton GE. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, ed. F Pereira, CJC Burges, L Bottou, KQ Weinberger, pp. 1097–105. Red Hook, NY: Curran Assoc.

Kutz JN, Brunton SL, Brunton BW, Proctor JL. 2016. *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. Philadelphia: SIAM

Labonté G. 1999. A new neural network for particle-tracking velocimetry. *Exp. Fluids* 26:340–46

Lagaris IE, Likas A, Fotiadis DI. 1998. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Trans. Neural Netw.* 9:987–1000

LeCun Y, Bengio Y, Hinton G. 2015. Deep learning. *Nature* 521:436–44

Lee C, Kim J, Babcock D, Goodman R. 1997. Application of neural networks to turbulence control for drag reduction. *Phys. Fluids* 9:1740–47

Lee Y, Yang H, Yin Z. 2017. PIV-DCNN: cascaded deep convolutional neural networks for particle image velocimetry. *Exp. Fluids* 58:171

Li Q, Dietrich F, Bollt EM, Kevrekidis IG. 2017. Extended dynamic mode decomposition with dictionary learning: a data-driven adaptive spectral decomposition of the Koopman operator. *Chaos* 27:103111

Liang D, Jiang C, Li Y. 2003. Cellular neural network to detect spurious vectors in PIV data. *Exp. Fluids* 34:52–62

Ling J, Jones R, Templeton J. 2016a. Machine learning strategies for systems with invariance properties. *J. Comput. Phys.* 318:22–35

Ling J, Kurzawski A, Templeton J. 2016b. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *J. Fluid Mech.* 807:155–66

Ling J, Templeton J. 2015. Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty. *Phys. Fluids* 27:085103

Loiseau JC, Brunton SL. 2018. Constrained sparse Galerkin regression. *J. Fluid Mech.* 838:42–67

Loucks D, van Beek E, Stedinger J, Dijkman J, Villars M. 2005. *Water Resources Systems Planning and Management: An Introduction to Methods*, Vol. 2. Cham, Switz.: Springer

Lumley J. 1970. *Stochastic Tools in Turbulence*. New York: Academic

Lusch B, Kutz JN, Brunton SL. 2018. Deep learning for universal linear embeddings of nonlinear dynamics. *Nat. Commun.* 9:4950

Mahoney MW. 2011. Randomized algorithms for matrices and data. *Found. Trends Mach. Learn.* 3:123–224

Manohar K, Brunton BW, Kutz JN, Brunton SL. 2018. Data-driven sparse sensor placement for reconstruction: demonstrating the benefits of exploiting known patterns. *IEEE Control Syst. Mag.* 38:63–86

Mardt A, Pasquali L, Wu H, Noé F. 2018. VAMPnets for deep learning of molecular kinetics. *Nat. Commun.* 9:5

Martin N, Gharib M. 2018. Experimental trajectory optimization of a flapping fin propulsor using an evolutionary strategy. *Bioinspiration Biomim.* 14:016010

Maulik R, San O, Rasheed A, Vedula P. 2019. Subgrid modelling for two-dimensional turbulence using neural networks. *J. Fluid Mech.* 858:122–44

Meena MG, Nair AG, Taira K. 2018. Network community-based model reduction for vortical flows. *Phys. Rev. E* 97:063103

Mehta UB, Kutler P. 1984. *Computational aerodynamics and artificial intelligence*. NASA Tech. Rep. NASA-TM-85994, Ames Res. Cent., Moffett Field, CA

Meijering E. 2002. A chronology of interpolation: from ancient astronomy to modern signal and image processing. *Proc. IEEE* 90:319–42

Meneveau C, Katz J. 2000. Scale-invariance and turbulence models for large-eddy simulation. *Annu. Rev. Fluid Mech.* 32:1–32

Mezic I. 2013. Analysis of fluid flows via spectral properties of the Koopman operator. *Annu. Rev. Fluid Mech.* 45:357–78

Milano M, Koumoutsakos P. 2002. Neural network modeling for near wall turbulent flow. *J. Comput. Phys.* 182:1–26

Minsky M, Papert SA. 1969. *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA: MIT Press

Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518:529–33

Nair AG, Taira K. 2015. Network-theoretic approach to sparsified discrete vortex dynamics. *J. Fluid Mech.* 768:549–71

Noack BR. 2018. Closed-loop turbulence control—from human to machine learning (and retour). In *Fluid-Structure-Sound Interactions and Control: Proceedings of the 4th Symposium on Fluid-Structure-Sound Interactions and Control*, ed. Y Zhou, M Kimura, G Peng, AD Lucey, L Hung, pp. 23–32. Singapore: Springer

Noé F, Nüske F. 2013. A variational approach to modeling slow processes in stochastic dynamical systems. *SIAM Multiscale Model Simul.* 11:635–55

Novati G, Mahadevan L, Koumoutsakos P. 2019. Controlled gliding and perching through deep-reinforcement-learning. *Phys. Rev. Fluids* 4:093902

Novati G, Verma S, Alexeev D, Rossinelli D, Van Rees WM, Koumoutsakos P. 2017. Synchronisation through learning for two self-propelled swimmers. *Bioinspiration Biomim.* 12:aa6311

Nüske F, Schneider R, Vitalini F, Noé F. 2016. Variational tensor approach for approximating the rare-event kinetics of macromolecular systems. *J. Chem. Phys.* 144:054105

Ollivier Y, Arnold L, Auger A, Hansen N. 2017. Information-geometric optimization algorithms: a unifying picture via invariance principles. *J. Mach. Learn. Res.* 18:564–628

Ostermeier A, Gawelczyk A, Hansen N. 1994. Step-size adaptation based on non-local use of selection information. In *International Conference on Parallel Problem Solving from Nature 1994 Oct 9*, pp. 189–98. Berlin: Springer

Ouellette NT, Xu H, Bodenschatz E. 2006. A quantitative study of three-dimensional Lagrangian particle tracking algorithms. *Exp. Fluids* 40:301–13

Papadimitriou DI, Papadimitriou C. 2015. Optimal sensor placement for the estimation of turbulent model parameters in CFD. *Int. J. Uncert. Quant.* 5:545–68

Parish EJ, Duraisamy K. 2016. A paradigm for data-driven predictive modeling using field inversion and machine learning. *J. Comput. Phys.* 305:758–74

Pathak J, Hunt B, Girvan M, Lu Z, Ott E. 2018. Model-free prediction of large spatiotemporally chaotic systems from data: a reservoir computing approach. *Phys. Rev. Lett.* 120:024102

Pelikan M, Ocenasek J, Trebst S, Troyer M, Alet F. 2004. Computational complexity and simulation of rare events of Ising spin glasses. In *Genetic and Evolutionary Computation—GECCO 2004*, pp. 36–47. Berlin: Springer

Perdikaris P, Venturi D, Karniadakis G. 2016. Multifidelity information fusion algorithms for high-dimensional systems and massive data sets. *SIAM J. Sci. Comput.* 38:B521–38

Perlman E, Burns R, Li Y, Meneveau C. 2007. Data exploration of turbulence simulations using a database cluster. In *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing*, p. 23. New York: ACM

Phan MQ, Juang JN, Hyland DC. 1995. On neural networks in identification and control of dynamic systems. In *Wave Motion, Intelligent Structures and Nonlinear Mechanics*, ed. A Guran, DJ Inman, pp. 194–225. Singapore: World Sci.

Pierret S, Van den Braembussche R. 1999. Turbomachinery blade design using a Navier-Stokes solver and artificial neural network. *J. Turbomach.* 121:326–32

Pollard A, Castillo L, Danaila L, Glauser M. 2016. *Whither Turbulence and Big Data in the 21st Century?* Cham., Switz.: Springer

Rabault J, Kuchta M, Jensen A, Réglade U, Cerardi N. 2019. Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control. *J. Fluid Mech.* 865:281–302

Raissi M, Karniadakis GE. 2018. Hidden physics models: machine learning of nonlinear partial differential equations. *J. Comput. Phys.* 357:125–41

Raissi M, Perdikaris P, Karniadakis G. 2019. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* 378:686–707

Rechenberg I. 1964. Kybernetische lösungsansteuerung einer experimentellen forschungsaufgabe. In *Annual Conference of the WGLR at Berlin in September*, Vol. 35, p. 33

Rechenberg I. 1973. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der Biologischen Evolution*. Stuttgart, Ger.: Frommann-Holzboog

Reddy G, Celani A, Sejnowski TJ, Vergassola M. 2016. Learning to soar in turbulent environments. *PNAS* 113:E4877–84

Reddy G, Wong-Ng J, Celani A, Sejnowski TJ, Vergassola M. 2018. Glider soaring via reinforcement learning in the field. *Nature* 562:236–39

Richter SR, Vineet V, Roth S, Koltun V. 2016. Playing for data: ground truth from computer games. In *European Conference on Computer Vision 2016*, ed. B Leibe, J Matas, N Sebe, M Welling, pp. 102–18. Cham, Switz.: Springer

Rico-Martinez R, Krischer K, Kevrekidis IG, Kube MC, Hudson JL. 1992. Discrete- vs. continuous-time nonlinear signal processing of Cu electrodissolution data. *Chem. Eng. Commun.* 118:25–48

Rokhlin V, Szlam A, Tygert M. 2009. A randomized algorithm for principal component analysis. *SIAM J. Matrix Anal. Appl.* 31:1100–24

First example of a simple binary network with learning capabilities.

Rosenblatt F. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol. Rev.* 65:386–408

Rowley CW, Dawson S. 2016. Model reduction for flow analysis and control. *Annu. Rev. Fluid Mech.* 49:387–417

Rowley CW, Mezić I, Bagheri S, Schlatter P, Henningson D. 2009. Spectral analysis of nonlinear flows. *J. Fluid Mech.* 645:115–27

Rudy SH, Brunton SL, Proctor JL, Kutz JN. 2017. Data-driven discovery of partial differential equations. *Sci. Adv.* 3:e1602614

Rumelhart DE, Hinton GE, Williams RJ. 1986. Learning representations by back-propagating errors. *Nature* 323:533–36

Salimans T, Ho J, Chen X, Sutskever I. 2017. Evolution strategies as a scalable alternative to reinforcement learning. arXiv:1703.03864 [stat.ML]

Schaeffer H. 2017. Learning partial differential equations via data discovery and sparse optimization. *Proc. R. Soc. A* 473:20160446

Schaul T, Horgan D, Gregor K, Silver D. 2015. Universal value function approximators. In *Proceedings of the 32nd International Conference on Machine Learning*, ed. F Bach, DM Blei, pp. 1312–20. Red Hook, NY: Curran Assoc.

Schmid PJ. 2010. Dynamic mode decomposition for numerical and experimental data. *J. Fluid Mech.* 656:5–28

Schmidt M, Lipson H. 2009. Distilling free-form natural laws from experimental data. *Science* 324:81–85

Schölkopf B, Smola AJ. 2002. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press

Schwefel HP. 1977. *Numerische Optimierung von Computer-Modellen Mittels der Evolutionsstrategie*. Basel, Switz.: Birkhäuser

Semeraro O, Lusseyran F, Pastur L, Jordan P. 2016. Qualitative dynamics of wavepackets in turbulent jets. arXiv:1608.06750 [physics.flu-dyn]

Silver D, Huang A, Maddison CJ, Guez A, Sifre L, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529:484–89

Singh AP, Medida S, Duraisamy K. 2017. Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils. *AIAA J.* 55:2215–27

Sirovich L. 1987. Turbulence and the dynamics of coherent structures, parts I–III. *Q. Appl. Math.* 45:561–90

Sirovich L, Kirby M. 1987. A low-dimensional procedure for the characterization of human faces. *J. Opt. Soc. Am. A* 4:519–24

Skinner SN, Zare-Behtash H. 2018. State-of-the-art in aerodynamic shape optimisation methods. *Appl. Soft. Comput.* 62:933–62

Strom B, Brunton SL, Polagye B. 2017. Intracycle angular velocity control of cross-flow turbines. *Nat. Energy* 2:17103

Sutton RS, Barto AG. 2018. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press. 2nd ed.

The classic book on reinforcement learning.

Taira K, Brunton SL, Dawson S, Rowley CW, Colonius T, et al. 2017. Modal analysis of fluid flows: an overview. *AIAA J.* 55:4013–41

Takeishi N, Kawahara Y, Yairi T. 2017. Learning Koopman invariant subspaces for dynamic mode decomposition. In *Advances in Neural Information Processing Systems 30*, ed. I Guyon, UV Luxburg, S Bengio, H Wallach, R Fergus, et al., pp. 1130–40. Red Hook, NY: Curran Assoc.

Tedrake R, Jackowski Z, Cory R, Roberts JW, Hoburg W. 2009. *Learning to fly like a bird*. Paper presented at the 14th International Symposium on Robotics Research, Lucerne, Switz.

Teo C, Lim K, Hong G, Yeo M. 1991. A neural net approach in analysing photographs in PIV. *IEEE Trans. Syst. Man Cybern.* 3:1535–38

Tesauro G. 1992. Practical issues in temporal difference learning. *Mach. Learn.* 8:257–77

Theodoridis S. 2015. *Machine Learning: A Bayesian and Optimization Perspective*. San Diego, CA: Academic

A monograph linking machine learning and Bayesian inference algorithms.

Tsiotras P, Mesbahi M. 2017. Toward an algorithmic control theory. *J. Guid. Control Dyn.* 40:194–96

Van Rees WM, Gazzola M, Koumoutsakos P. 2015. Optimal morphokinematics for undulatory swimmers at intermediate reynolds numbers. *J. Fluid Mech.* 775:178–88

Verma S, Novati G, Koumoutsakos P. 2018. Efficient collective swimming by harnessing vortices through deep reinforcement learning. *PNAS* 115:5849–54

Vlachas PR, Byeon W, Wan ZY, Sapsis TP, Koumoutsakos P. 2018. Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proc. R. Soc. A* 474:20170844

Wan ZY, Vlachas P, Koumoutsakos P, Sapsis T. 2018. Data-assisted reduced-order modeling of extreme events in complex dynamical systems. *PLOS ONE* 13:e0197704

Wang JX, Wu JL, Xiao H. 2017. Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data. *Phys. Rev. Fluids* 2:034603

Wang M, Hemati MS. 2017. Detecting exotic wakes with hydrodynamic sensors. arXiv:1711.10576

Wehmeyer C, Noé F. 2018. Time-lagged autoencoders: deep learning of slow collective variables for molecular kinetics. *J. Chem. Phys.* 148:241703

Wiener N. 1965. *Cybernetics or Control and Communication in the Animal and the Machine*, Vol. 25. Cambridge, MA: MIT Press

Willert CE, Gharib M. 1991. Digital particle image velocimetry. *Exp. Fluids* 10:181–93
Williams MO, Rowley CW, Kevrekidis IG. 2015. A kernel approach to data-driven Koopman spectral analysis. *J. Comput. Dyn.* 2:247–65
Wright J, Yang A, Ganesh A, Sastry S, Ma Y. 2009. Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell.* 31:210–27
Wu H, Mardt A, Pasquali L, Noe F. 2018. Deep generative Markov State Models. *Adv. Neural Inf. Process. Syst.* 31:3975–84
Wu X, Moin P. 2008. A direct numerical simulation study on the mean velocity characteristics in turbulent pipe flow. *J. Fluid Mech.* 608:81–112
Xiao H, Wu JL, Wang JX, Sun R, Roy C. 2016. Quantifying and reducing model-form uncertainties in Reynolds-averaged Navier–Stokes simulations: a data-driven, physics-informed Bayesian approach. *J. Comp. Phys.* 324:115–36
Xie Y, Franz E, Chu M, Thuerey N. 2018. tempoGAN: a temporally coherent, volumetric GAN for super-resolution fluid flow. *ACM Trans. Graph.* 37:95
Yang J, Wright J, Huang TS, Ma Y. 2010. Image super-resolution via sparse representation. *IEEE Trans. Image Process.* 19:2861–73