Optimization and passive flow control using single-step deep reinforcement learning

Hassan Ghraieb, Jonathan Viquerat, Aurélien Larcher, P. Meliga, Elie Hachem

Optimization and passive flow control using single-step deep reinforcement learning

A Preprint

H. Ghraieb
MINES Paristech , PSL - Research University, CEMEF

J. Viquerat*
MINES Paristech , PSL - Research University, CEMEF
jonathan.viquerat@mines-paristech.fr

A. Larcher
MINES Paristech , PSL - Research University, CEMEF

P. Meliga
MINES Paristech , PSL - Research University, CEMEF

Elie Hachem
MINES Paristech , PSL - Research University, CEMEF

June 5, 2020

Abstract

This research gauges the ability of deep reinforcement learning (DRL) techniques to assist the optimization and control of fluid mechanical systems. It combines a novel, "degenerate" version of the proximal policy optimization (PPO) algorithm, that trains a neural network in optimizing the system only once per learning episode, and an in-house stabilized finite elements environment implementing the variational multiscale (VMS) method, that computes the numerical reward fed to the neural network. Three prototypical examples of separated flows in two dimensions are used as testbed for developing the methodology, each of which adds a layer of complexity due either to the unsteadiness of the flow solutions, or the sharpness of the objective function, or the dimension of the control parameter space. Relevance is carefully assessed by comparing systematically to reference data obtained by canonical direct and adjoint methods. Beyond adding value to the shallow literature on this subject, these findings establish the potential of single-step PPO for reliable black-box optimization of computational fluid dynamics (CFD) systems, which paves the way for future progress in optimal flow control using this new class of methods.

Keywords Deep Reinforcement Learning; Artificial Neural Networks; Computational fluid dynamics; Flow control; Adjoint method

*Corresponding author

1 Introduction

Flow control, defined as the ability to finesse a flow into a more desired state, is a field of tremendous societal and economical importance. In applications such as ocean shipping or airline traffic, reducing the overall drag by just a few percent while maintaining lift can help reducing fossil fuel consumption and CO_2 emission while saving several billion dollars annually. Many other scenario relevant to fluid mechanical systems call for similarly improved engineering design, e.g., the airline industry is greatly concerned with reducing the structural vibrations and the radiated noise that occur under unsteady flow conditions, while microfluidics and combustion both benefit from enhanced mixing (which can be achieved by promoting unsteadiness in some appropriate manner). All such problems fall under the purview of this line of study.

Flow control is often benchmarked in the context of bluff body drag reduction. Numerous strategies have been implemented, either open-loop with passive appendices (e.g., end plate, splitter plate, small secondary cylinder, or flexible tail), or open-loop with actuating devices (e.g., plasma actuation, steady or unsteady base bleeding, rotation) or closed-loop (e.g. via transverse motion, blowing/suction, rotation, all relying on an appropriate sensing of flow variables). An overview of the recent achievements and perspectives can be found in several comprehensive surveys [1, 2, 3, 4, 5, 6, 7, 8, 9]. Nonetheless, many of the proposed strategies are trial and error, and therefore require extensive and costly experimental or numerical campaigns. This has motivated the development of rigorous mathematical formalisms capable of designing optimal control strategies with minimal effort. The adjoint method is one popular family of such techniques, that has proven efficient at computing accurate, low-cost estimates of the objective gradient with respect to the control variables in large optimization spaces, and is widely used for many applications ranging from oceanography and atmospheric sciences [10] to aerodynamic design [11, 12, 13, 14], by way of fresh developments meant to promote/mitigate the linear amplification of flow disturbances [15, 16, 17, 18, 19, 20].

It is the premise of this research that the task of selecting an optimal subset of control parameters can be alternatively automated with the assistance of supervised learning algorithms (i.e., the simplest and most common form of machine learning, where an agent learns from labeled training data and is provided with corrective feedback). The breakthrough in this field dates back to the introduction of the back-propagation algorithm [21]. It has helped popularize Artificial Neural Networks (ANN), a family of versatile non-parametric tools that can be trained to hierarchically extract informative features from data, and can be used for several purposes, from exploratory data analysis to qualitative and quantitative predictive modeling. Since then, the increased affordability of high performance computational hardware (together with reduced costs for computation and data storage) have allowed leveraging the ever-increasing volume of numerical and experimental data generated for research and engineering purposes into novel insight and actionable information, which in turn has reshaped entire scientific disciplines, such as image analysis [22] or robotics [23, 24]. Because neural networks have produced most remarkable results when applied to stiff large-scale nonlinear problems [25], it is only natural to assume that they are similarly suitable for the state-space models arising from the high-dimensional Machine learning algorithms have thus been making rapid inroads in fluid mechanics, as a mean to solve the Navier–Stokes equations [26] or to predict closure terms in turbulence models [27]; see also Ref. [28] for an overview of the current developments and opportunities.

The focus here is on deep reinforcement learning (DRL), an advanced branch of machine learning in which neural networks specialize in solving decision-making problems (the *deep* terminology being used to weigh on the sizable depth of the network itself). Simply put, the neural network trains in finding out which actions or succession of actions maximize a numerical reward signal, one challenge being that a given action affects not only the immediate reward but also the next states and, thus, all the subsequent rewards. The most well-known success story in this field is AlphaGo, the ANN that defeated the top-level human player at the game of Go [29], but the approach has also proven fruitful to solve computer vision problems (e.g., filtering, or extracting image features) [30] and has enabled breakthroughs in the optimal control of complex dynamic systems [31, 32], including legged robots deployed in real environments [33]. Despite these achievements, DRL remains surprisingly sparsely used in fluid mechanics, with a limited amount of literature barely scratching the surface of the performance improvements to be delivered in flow control [34, 35, 36] and shape optimization problems [37].

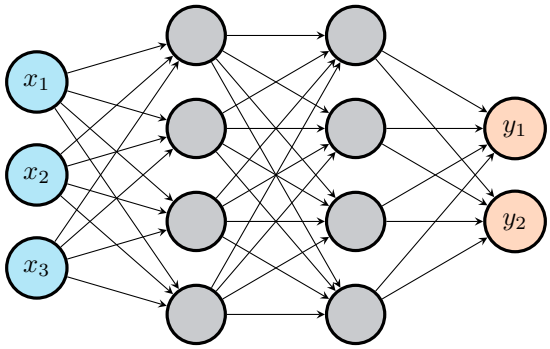

Figure 1: Fully connected neural network with two hidden layers, modeling a mapping from $\mathbb{R}^3$ to $\mathbb{R}^2$.

This researches assesses the feasibility of applying proximal policy optimization (PPO [32]) for flow control and optimization purposes, to both shape the capabilities of the method (PPO is still a relatively newcomer that has quickly emerged as the go-to DRL algorithm due to its data efficiency, simplicity of implementation and reliable performance) and narrow the gap between DRL and advanced numerical methods for multiscale, multi-physics computational fluid dynamics (CFD). We investigate more specifically the "degenerate" single-step PPO algorithm introduced in [37], in which the neural network gets only one attempt per learning episode at finding the optimal. Prototypical examples of separated flows in two dimensions are used as testbed for developing this novel approach (to the best of the authors knowledge, no previous study in the related literature has considered using DRL in a similar fashion), as it has been speculated to hold a high potential as a reliable black-box optimizer for CFD problems (where only the final configuration is of interest), but needs further characterization and fine-tuning.

The organization is as follows: the single-step PPO method is presented in section 2, together with the baseline principles and assumptions of both DRL and PPO. Section 3 outlines the main features of the finite element CFD environment used to compute the numerical reward fed to the neural network, whose efficiency and reliability has been assessed in several previous studies. Three test cases are then organized by increasing level of complexity, namely the flow past a NACA 0012 airfoil (section 4), the flow past a tandem arrangement of two identical cylinders (section 5), and the passive control of a cylinder flow (section 6). For each case, exhaustive results pertaining to the DRL optimization of steady asymptotic, time-averaged, or fluctuating drag and lift are reported and carefully validated against reference data obtained by canonical direct/adjoint methods.

2 Deep reinforcement learning and proximal policy optimization

2.1 Neural networks

A neural network (NN) is a collection of artificial neurons, i.e., connected computational units with universal approximation capabilities [38], that can be trained to arbitrarily well approximate the mapping function between input and output spaces. Each connection provides the output of a neuron as an input to another neuron. Each neuron performs a weighted sum of its inputs, to assign significance to the inputs with regard to the task the algorithm is trying to learn. It then adds a bias to better represent the part of the output that is actually independent of the input. Finally, it feeds an activation function that determines whether and to what extent the computed value should affect the ultimate outcome. As sketched in figure 1, a fully connected network is generally organized into layers, with the neurons of one layer being connected solely to those of the immediately preceding and following layers. The layer that receives the external data is the input layer, the layer that produces the outcome is the output layer, and in between them are zero or more hidden layers.

The design of an efficient neural network requires a proper optimization of the weights and biases, together with a relevant nonlinear activation function. The abundant literature available on this topic (see, e.g., Ref. [39] and the references therein) points to a relevant network architecture (e.g., type

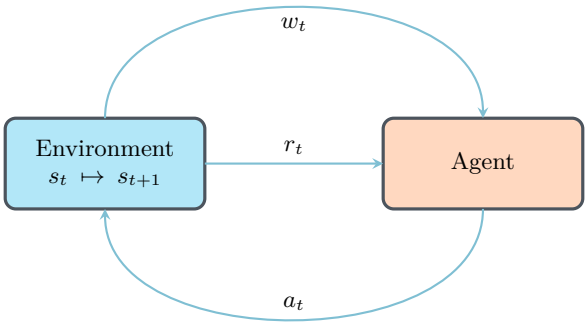

Figure 2: RL agent and its interactions with its environment.

of network, depth, width of each layer), finely tuned hyper parameters (i.e., parameters whose value cannot be estimated from data, e.g., optimizer, learning rate, batch size) and a sufficiently large amount of data to learn from as being the key ingredients for success.

2.2 Deep reinforcement learning

Deep reinforcement learning (DRL) is an advanced branch of machine learning in which deep neural networks train in solving sequential decision-making problems. It is a natural extension of reinforcement learning (RL), in which an agent (the neural network) is taught how to behave in an environment by taking actions and by receiving feedback from it under the form of a reward (to measure how good or bad the taken action was) and information (to gauge how the action has affected the environment). This can be formulated as a Markov Decision Process, for which a typical execution goes as follows (see also figure 2):

- assume the environment is in state $s_t \in \mathcal{S}$ at iteration t, where $\mathcal{S}$ is a set of states,
- the agent uses w_t, an observation of the current environment state (and possibly a partial subset of s_t) to take action $a_t \in \mathcal{A}$, where $\mathcal{A}$ is a set of actions,
- the environment reacts to the action by transitionning from s_t to state $s_{t+1} \in \mathcal{S}$,
- the agent is fed with a reward $r_t \in \mathcal{R}$, where $\mathcal{R}$ is a set of rewards, and a new observation w_{t+1},

This repeats until some termination state is reached, the succession of states and actions defining a trajectory $\tau = \big(s_0, a_0, s_1, a_1, ...\big)$. In any given state, the objective of the agent is to determine the action maximizing its cumulative reward over an episode, i.e., over one instance of the scenario in which the agent takes actions. Most often, the quantity of interest is the discounted cumulative reward along a trajectory defined as

$$R(\tau) = \sum_{t=0}^{T} \gamma^t r_t \,, \tag{1}$$

where T is the horizon of the trajectory, and $\gamma \in [0, 1]$ is a discount factor that weighs the relative importance of present and future rewards (the agent being short-sighted in the limit where $\gamma \to 0$, since it then cares solely about the first reward, and far-sighted in the limit where $\gamma \to 1$, since it then cares equally about all rewards).

There exist two main types of RL algorithms, namely model-based methods, in which the agent tries to build a model of how the environment works to make predictions about what the next state and reward will be before taking any action, and model-free methods, in which the agent conversely interacts with the environment without trying to understand it, and are prominent in the DRL community. Another important distinction to be made within model-free algorithms is that between value-based

methods, in which the agent learns to predict the future reward of taking an action when provided a given state, then selects the maximum action based on these estimates, and policy-based methods, in which it optimizes the expected reward of a decision policy mapping states to actions. Many of the most successful algorithms in DRL (including proximal policy optimization, whose assessment for flow control and optimization purposes is the primary motivation for this research) proceed from policy gradient methods, in which gradient ascent is used to optimize a parameterized policy with respect to the expected return, as further explained in the next section. The reader interested in a more thorough introduction to the zoology of RL methods (together with their respective pros and cons) is referred to Ref. [40].

2.3 From policy methods to Proximal policy optimization

This section intended for the non-specialist reader briefly reviews the basic principles and assumptions of policy gradient methods, together with the various steps taken for improvement.

Policy methods. A policy method maximizes the expected discounted cumulative reward of a decision policy mapping states to actions. It resorts not to a value function, but to a probability distribution over actions given states, that fully defines the behavior of the agent. Since policies are most often stochastic, the following notations are introduced:

- $\pi(s,a)$ is the probability of taking action a in state s under policy π,
- $Q^\pi(s,a)$ is the expected value of the return of the policy after taking action a in state s (also termed state-action value function or Q-function)

$$Q^\pi(s,a) = \mathbb{E}_\pi\big[R(\tau)|s,a\big]\,, \tag{2}$$

- $V^\pi(s)$ is the expected value of the return of the policy in state s (also termed value function or V-function)

$$V^\pi(s) = \mathbb{E}_\pi\big[R(\tau)|s\big]\,. \tag{3}$$

  The V and Q functions are therefore such that

$$V^\pi(s) = \sum_a \pi(s,a) Q^\pi(s,a)\,, \tag{4}$$

  so $V^\pi(s)$ can also be understood as the probability-weighted average of discounted cumulated rewards over all possible actions in state s.

Policy gradient methods. A policy gradient method aims at optimizing a parametrized policy π_θ, where θ denotes the free parameters whose value can be learnt from data (as opposed to the hyper parameters). In practice, one defines an objective function based on the expected discounted cumulative reward

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\big[R(\tau)\big]\,, \tag{5}$$

and seeks the parameterization θ^* maximizing $J(\theta)$, hence such that

$$\theta^* = \arg\max_\theta \mathbb{E}_{\tau\sim\pi_\theta}\big[R(\tau)\big]\,, \tag{6}$$

which can be done on paper by plugging an estimator of the policy gradient $\nabla_\theta J(\theta)$ into a gradient ascent algorithm. This is no small task as one is looking for the gradient with respect to the policy parameters, in a context where the effects of policy changes on the state distribution are unknown (since modifying the policy will most likely modify the set of visited states, which will in turn affect performance in some indefinite manner). The most commonly used estimator, derived in [40] using the log-probability trick, reads

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} \nabla_\theta \log\left(\pi_\theta(s_t,a_t)\right) R(\tau)\right] \sim \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} \nabla_\theta \log\left(\pi_\theta(s_t,a_t)\right) A^\pi(s,a)\right]\,, \tag{7}$$

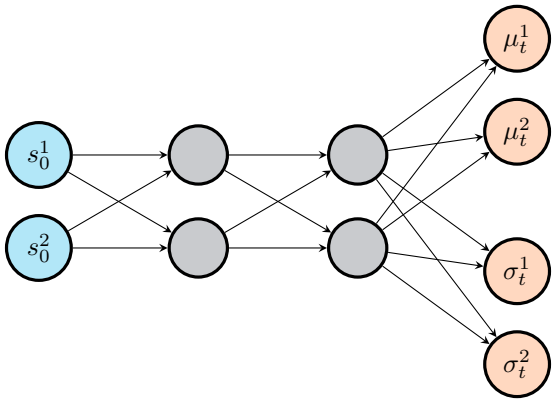

Figure 3: Agent network example used to map states to policy. The input state $\boldsymbol{s_0}$, here of size 2, is mapped to a mean $\boldsymbol{\mu}$ and a standard deviation $\boldsymbol{\sigma}$ vectors, each of size 2. All activation functions are ReLu, except for that of the last layer, which are linear for the μ output, and softplus for the σ output. Orthogonal weights initialization is used throughout the network.

where the rightmost term is a convenient approximation relying on the advantage function

$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)\,, \tag{8}$$

that measures the improvement (if $A > 0$, otherwise the lack thereof) associated with taking action a in state s compared to taking the average over all possible actions. This is because the value function does not depend on θ, so taking it off changes neither the expected value, nor the gradient, but it does reduce the variance, and speeds up the training. Furthermore, when the policy π_θ is represented by a neural network (in which case θ simply denotes the network weights and biases to be optimized), the focus is rather on the policy loss defined as

$$L(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} \log\left(\pi_\theta(a_t|s_t)\right) A^\pi(s,a)\right]\,, \tag{9}$$

whose gradient is equal to the (approximated) policy gradient (7) (since the gradient operator acts only on the log-policy term, not on the advantage) and is computed with respect to each weight and bias by the chain rule, one layer at the time, using the back-propagation algorithm [21].

Trust regions. The performance of policy gradient methods is hurt by the high sensitivity to the learning rate, i.e., the size of the step to be taken in the gradient direction. Indeed, small learning rates are detrimental to learning, but large learning rates can lead to a performance collapse if the agent falls off the cliff and restarts from a poorly performing state with a locally bad policy. This is all the more harmful as the learning rate cannot be tuned locally, meaning that an above average learning rate will speed up learning in some regions of the parameter space where the policy loss is relatively flat, but will possibly trigger an exploding policy update in other regions exhibiting sharper variations. One way to ensure continuous improvement is by imposing a trust region constraint to limit the difference between the current and updated policies, which can be done by determining first a maximum step size relevant for exploration, then by locating the optimal point within this trust region. We will not dwell on the intricate details of the many algorithms developed to solve such trust region optimization problems, e.g., natural policy gradient (NPG [41]), or trust region policy optimization (TRPO [42]). Suffice it to say that they use the minorize-maximization algorithm to maximize iteratively a surrogate policy loss (i.e. a lower bound approximating locally the actual loss at the current policy), but are difficult to implement and can be computationally expensive, as they rely on an estimate of the second-order gradient of the policy log probability.

Proximal policy optimization. Proximal policy optimization (PPO) is another approach with simple and effective heuristics, that uses a probability ratio between the two policies to maximize improvement without the risk of performance collapse [32]. The focus here is on the PPO-clip algorithm[2], that optimizes the surrogate loss

$$L(\theta) = \mathbb{E}_{(s,a)\sim\pi_\theta}\left[\min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)}A^{\pi_\theta}(s,a), g(\epsilon, A^{\pi_\theta}(s,a))\right)\right], \tag{10}$$

where

$$g(\epsilon, A) = \begin{cases}(1+\epsilon)A & A \geq 0\,,\\ (1-\epsilon)A & A < 0\,,\end{cases} \tag{11}$$

and $\epsilon \in [0.1, 0.3]$ is the clipping range, a small hyper parameter defining how far away the new policy is allowed to go from the old. In practice, a state-action pair yielding a positive advantage will cause $\pi_\theta(a|s)$ to increase for the action to become more likely. By doing so, if it increases in a way such that $\pi_\theta(a|s) > (1+\epsilon)\pi_{\theta_{old}}(a|s)$, the min kicks in (10) and its argument hits a ceiling of $(1+\epsilon)A^{\pi_\theta}(s,a)$. Conversely, a state-action pair yielding a negative advantage will cause $\pi_\theta(a|s)$ to decrease for the action to become less likely. If it increases in a way such that $\pi_\theta(a|s) < (1-\epsilon)\pi_{\theta_{old}}(a|s)$, the max kicks in and its argument hits a ceiling of $(1-\epsilon)A^{\pi_\theta}(s,a)$. In neither case does the new policy has any incentive to step too far away from the current policy, which ensures that both policies will behave similarly.

The strengths of the approach lie in the fact that the clipped surrogate itself is cheap to compute, and that it needs only an estimate of the first-order gradient of the policy log probability (like policy gradient methods). Refinements have been proposed recently, for instance the Trust region PPO (TRGPPO) that adaptively adjusts the clipping range within the trust region [43], but classic PPO is generally acknowledged to be one of the most successful DRL method, combining ease of implementation and tuning, together with state-of-the-art performance across a wide range of challenging tasks.

2.4 Single-step PPO algorithm

We now come to the single-step PPO method proposed in [37], a "degenerate" version of PPO in which the agent and the environment get to exchange only one single triplet (s, a, r) per learning episode. This represents a significant difference with respect to classic Markov-based methods. and amounts to view the agent as a parameterized mapping $f_{\boldsymbol{\theta}}$ from state to action.

As sketched in figure 3, the agent is consistently fed with the same input state $\boldsymbol{s}_0$ (usually a vector of zeros), and outputs a policy π that can be represented by mean and standard deviation vectors (denoted by $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, respectively) whose size matches the dimension of the action required by the environment. The optimization loop is as follows: first, the agent implements a random mapping $f_{\boldsymbol{\theta}_0}$ from $\boldsymbol{s}_0$ to an initial policy determined by the initialization of the network parameters $\boldsymbol{\theta}_0$. At each iteration, a population of actions $\boldsymbol{a}_t = f_{\boldsymbol{\theta}_t}(s_0)$ is drawn from the current policy, the reward associated to each set of action is computed and the agent is returned incentives (through the rewards) to update the free parameters in a way such that the next population of actions $\boldsymbol{a_{t+1}} \sim f_{\boldsymbol{\theta}_{t+1}}(s_0)$ will yield higher rewards (see also figure 4). This is done following for the most part the various steps described in section 2.3, only one seeks in practice the optimal mapping $f_{\boldsymbol{\theta_{opt}}}$ such that $\boldsymbol{a}_{opt} \sim f_{\boldsymbol{\theta_{opt}}}(\boldsymbol{s_0})$, not the optimal set of actions $\boldsymbol{a}_{opt}$ itself, and the loss is computed from (10) by substituting a normalized averaged reward for the advantage function.

It is worth noticing that an accurate estimation of the policy gradient requires evaluating a large amount of actions drawn from the current policy, which comes at the expense of computing the same amount of reward evaluations. One key issue in the context of CFD applications is thus the ability to distribute a given set of actions to a parallel environment running on large computer clusters, as we show in the following that the CPU cost of solving the present academic optimization problems in two dimensions ranges from a hundred to several thousand hours.

[2]There is also a PPO-Penalty variant which uses a penalization on the average Kullback–Leibler divergence between the current and new policies, but PPO-clip performs better in practice.

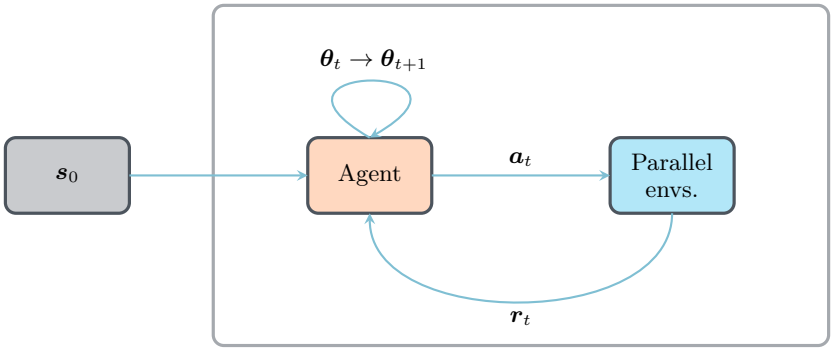

Figure 4: Action loop for single-step PPO. At each episode, the input state $\boldsymbol{s}_0$ is provided to the agent, which in turn provides n actions to n parallel environments. The latter return n rewards, that evaluate the quality of each action taken. Once all the rewards are collected, an update of the agent parameters is made using the PPO loss (10). The process is repeated until convergence of the mapping (i.e., $\boldsymbol{\mu} = \boldsymbol{\mu}^*$ and $\boldsymbol{\sigma} = \mathbf{0}$).

3 Numerical methods

In the following, we investigate two-dimensional (2-D) laminar incompressible flows over various geometries of interest. The flow is described in a Cartesian coordinate system (x, y) with drag force (resp. lift force) positive in the stream-wise $+x$ direction (resp. the cross-wise $+$y direction). The governing equations to be solved numerically are the incompressible Navier–Stokes equations

$$\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0\,, \qquad \partial_t \boldsymbol{u} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \nabla \cdot [p\mathbf{I} - \frac{2}{\mathrm{Re}}\boldsymbol{\varepsilon}(\boldsymbol{u})] = \mathbf{0}\,, \tag{12}$$

where $\boldsymbol{u} = (u, v)$ is the velocity field of component u (resp. v) in the x (resp. y) direction, p is the pressure, $\boldsymbol{\varepsilon}(\boldsymbol{u}) = (\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T)/2$ is the rate of deformation tensor, and Re is the Reynolds number built from the free stream velocity and a to-be-specified length scale.

3.1 The variational multiscale approach (VMS)

In the context of finite element methods (that remain widely used to simulate engineering CFD systems due to their ability to handle complex geometries), direct numerical simulation (DNS) solves the weak form of (12) obtained by integrating by parts the pressure and viscous terms, to give

$$(\partial_t \boldsymbol{u} + \boldsymbol{u} \cdot \nabla \boldsymbol{u}\,,\, \boldsymbol{w}) + \frac{2}{\mathrm{Re}}(\boldsymbol{\varepsilon}(\boldsymbol{u})\,,\, \boldsymbol{\varepsilon}(\boldsymbol{w})) - (p\,,\, \nabla \cdot \boldsymbol{w}) + (\nabla \cdot \boldsymbol{u}\,,\, q) = 0\,, \tag{13}$$

where $(\boldsymbol{w}, q)$ are relevant test functions for the velocity and pressure variables, and $(\ ,\ )$ is the L^2 inner product on the computational domain.

We use here the variational multiscale (VMS) approach [44, 45, 46] to solve a stabilized formulation of (13). This allows circumventing the Babuska—Brezzi condition (that otherwise imposes that different interpolation orders be used to discretize the velocity and pressure variables, while we use here simple continuous piecewise linear P_1 elements for both) and prevents numerical instabilities in advection regimes at high Reynolds numbers (that otherwise quickly spread in the shear regions and pollute the entire solution domain). We shall not go into the extensive details about the derivation of the stabilized formulation, for which the reader is referred to [47]. Suffice it to say here that the flow quantities are split into coarse and fine scale components, that correspond to different levels of resolution. The fine scales are solved in an approximate manner to allow modeling their effect into the large-scale equations, which gives rise to additional terms in the right-hand side of (13), and yields the

following weak form for the large scale

$$(\partial_t \boldsymbol{u} + \boldsymbol{u}\cdot\nabla\boldsymbol{u}\,,\,\boldsymbol{w}) + \frac{2}{\mathrm{Re}}(\boldsymbol{\varepsilon}(\boldsymbol{u})\,,\,\boldsymbol{\varepsilon}(\boldsymbol{w})) - (p\,,\,\nabla\cdot\boldsymbol{w}) + (\nabla\cdot\boldsymbol{u}\,,\,q) = \sum_{K\in\mathcal{T}_h}\left[(\tau_1\mathcal{R}_M\,,\,\boldsymbol{u}\cdot\nabla\boldsymbol{w})_K + (\tau_1\mathcal{R}_M\,,\,\nabla q)_K + (\tau_2\mathcal{R}_C\,,\,\nabla\cdot\boldsymbol{w})_K\right], \tag{14}$$

where $(\,,\,)_K$ is the inner product on element K, $\mathcal{R}_{M,C}$ are the residuals defined by

$$-\mathcal{R}_M = \partial_t\boldsymbol{u} + \boldsymbol{u}\cdot\nabla\boldsymbol{u} + \nabla p \qquad -\mathcal{R}_C = \boldsymbol{\nabla}\cdot\boldsymbol{u}\,, \tag{15}$$

and $\tau_{1,2}$ are ad-hoc mesh-dependent stabilization parameters defined in [48, 49]. Equation (14) is solved here with an in-house VMS solver whose accuracy and flexibility has been assessed in a series of previous papers [47, 50, 51, 49], together with open flow boundary conditions consisting of a uniform free-stream $\mathbf{u} = (1, 0)$ at the inflow, free-surface conditions $\partial_y u = v = 0$ at the upper and lower boundaries, advective conditions at the outflow, and no-slip conditions at all solid surfaces.

3.2 Remeshing strategy

For each test case documented in the following, a 2-D computational domain is defined as

$$\Omega = \{(x, y) \mid x_{-\infty} \leq x \leq x_\infty\,;\ |y| \leq y_\infty\,;\ \Gamma_s(x, y) < 0\}\,, \tag{16}$$

where Γ_s is a level set function that depends on the control parameters, whose zero iso-contour gives the position of the solid surface S (the convention used here being to have $\Gamma_s(x, y) < 0$ inside the solid domain and $\Gamma_s \geq 0$ outside). The dependency of Γ_s on the control parameters is one key issue, as it imposes that a different mesh be used for each action taken by the agent over the course of the PPO optimization, each of which must be equally capable of accurately capturing the global and local spatio-temporal features at play. Here, this is achieved using the immerse volume method (IVM), widely used to immerse and represent complex geometries inside a unique mesh, and that has been implemented in the context of finite element VMS methods (see, e.g., [50, 49] for a detailed presentation of the mathematical formulation relevant to a rigid body immersed in an incompressible fluid). For each geometry and each value of the control parameters, the level set function is computed at all nodes of the mesh using the algorithm presented in [52]. The mesh at the interface is then refined anisotropically using the gradient of the level set, and the physical properties of each domain are set with appropriate mixing laws. Each mesh is adapted under the constraint of a fixed, case-dependent number of edges. Representative interface regions at the end of the anisotropic adaptation process are shown in figure 5. In practice, the adaptation process consists of adding nodes locally in the vicinity of the interface. The rest of the elements keep the same background size, that increases with the distance via three successive refinement steps to accurately capture the near-wake region; see figure 5(d).

3.3 Reward computations

This study is on the optimization of the non-dimensional forces exerted on the immersed body, as measured by the drag and lift coefficients per unit span length, denoted by D and L, respectively, and defined as

$$2\oint_S(-p\boldsymbol{n} + \frac{2}{\mathrm{Re}}\boldsymbol{\varepsilon}(\boldsymbol{u})\cdot\boldsymbol{n})\cdot\boldsymbol{e}_\theta\,\mathrm{d}l\,, \tag{17}$$

where $\boldsymbol{e}_\theta = (1-\theta)\boldsymbol{e}_x + \theta\boldsymbol{e}_y$ and θ is a Boolean used to specify the quantity subjected to optimization ($\theta = 0$ for drag, $\theta = 1$ for lift). The latter is computed as

$$2\oint_{\partial\Omega}(-p\boldsymbol{n} + \frac{2}{\mathrm{Re}}\boldsymbol{\varepsilon}(\boldsymbol{u})\cdot\boldsymbol{n})\cdot\boldsymbol{w}\,\mathrm{d}l = -2\int_\Omega(-p\boldsymbol{n} + \frac{2}{\mathrm{Re}}\boldsymbol{\varepsilon}(\boldsymbol{u})\cdot\boldsymbol{n}) : \nabla\boldsymbol{w}\,\mathrm{d}s - 2\int_\Omega(\nabla\boldsymbol{u}\cdot\boldsymbol{u})\cdot\boldsymbol{w}\,\mathrm{d}s\,, \tag{18}$$

using the variational approach described in [53], that consists of multiplying the momentum equations by a test function $\boldsymbol{w}$ equal to $\boldsymbol{e}_\theta$ on the nodes lying on S and 0 otherwise (hence vanishing everywhere except on the first layer of elements around S), then of integrating by parts to bring up only volume integral terms, reportedly more accurate and less sensitive to the approximation of the body interface their surface counterparts [54].

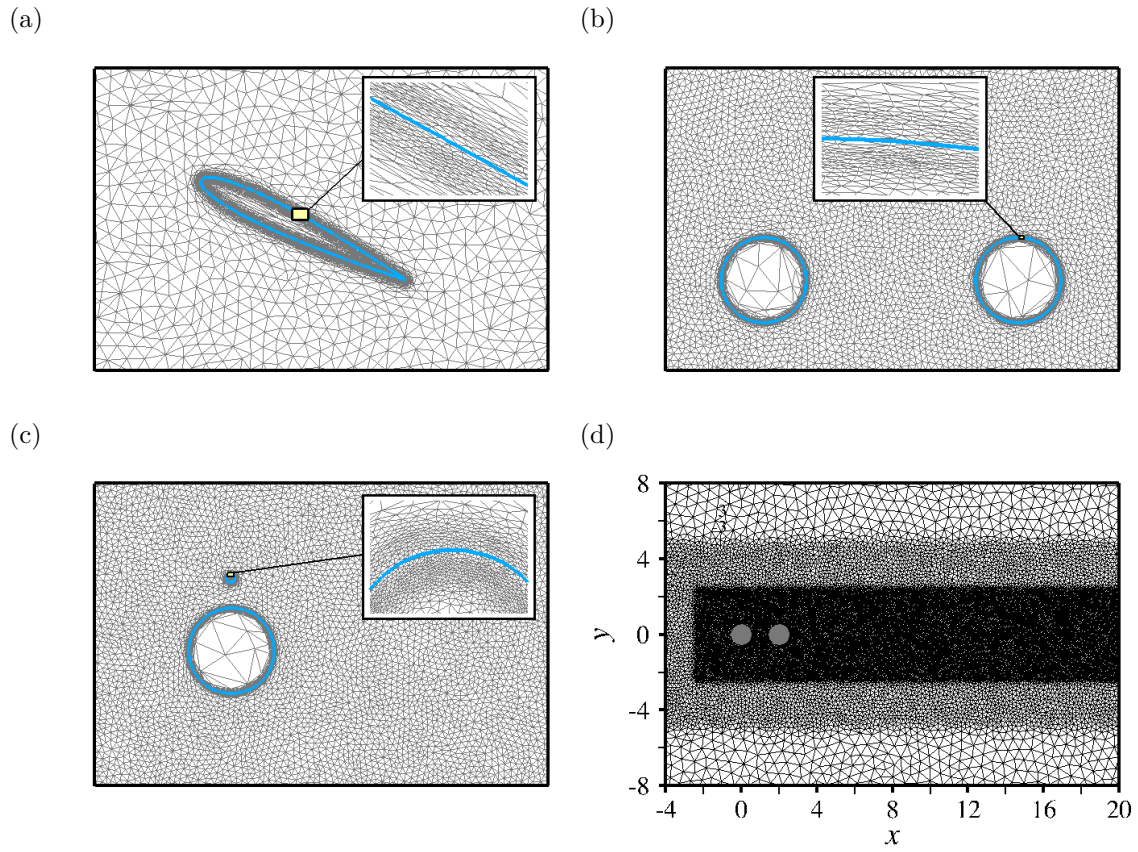

Figure 5: $(a-c)$ Details of the boundary layer mesh around (a) a NACA 0012 airfoil, (b) two circular tandem cylinders, and (c) a two-circular cylinder system, as obtained at the end of the anisotropic adaptation process. The blue line indicates the zero iso-contour of the level set function. (d) Background mesh of the tandem configuration evidencing the three successive refinement steps.

3.4 Neural network and PPO implementation

The present workflow relies on the online PPO implementation of Stable Baselines, a toolset of reinforcement learning algorithms dedicated to the research community and industry [55], for which a custom OpenAI environment has been designed using the Gym library [56]. The instant reward r_t used to train the neural network is simply the quantity subjected to optimization (i.e., either the steady asymptotic, time-averaged, or fluctuating value or drag and lift, modulo a plus or minus sign to tackle both maximization and minimization problems). A moving average reward is also computed on the fly as the sliding average over the 100 latest values of r_t, and we assume convergence when the absolute variation in the moving average reward between two consecutive episodes is thrice smaller than 10^{-3} out of 4 consecutive values. All other relevant hyper parameters are documented in the next sections, with the exception of the discount factor (since single-step PPO computes only one single reward per episode).

4 Optimal angle of attack of a NACA 0012 airfoil

We consider first the NACA 0012 airfoil at incidence in a uniform stream depicted in figure 6. All variables are made non-dimensional using the straight chord distance c between the leading and trailing

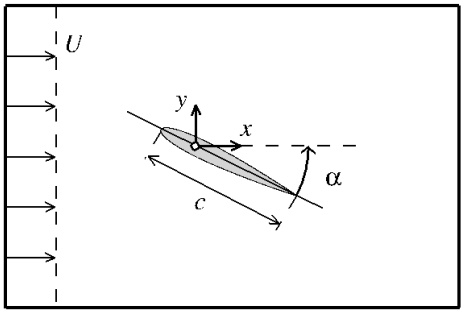

Figure 6: Schematic diagram of the flow past a NACA 0012 at incidence.

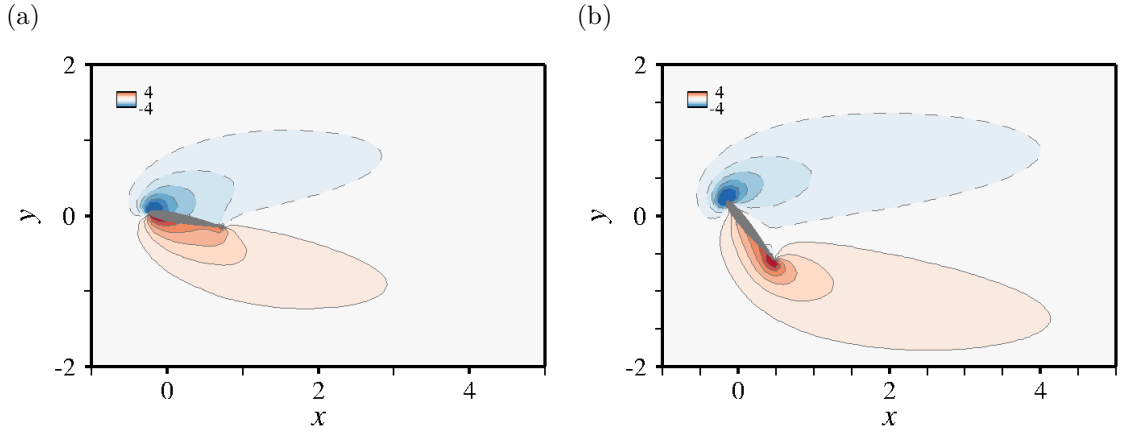

Figure 7: Flow past a NACA 0012 at Re = 10. Iso-contours of the steady vorticity for (a) $\alpha = 11°$, and (b) $\alpha = 51°$.

edges and the free-stream velocity. This is a simple problem whose sole control parameter[3] is the angle of attack α, i.e., the angle between the chord and the stream-wise directions (with $\alpha > 0$ corresponding to positive lift), which allows comparing DRL/PPO predictions directly to DNS[4] reference data. The origin of the coordinate system is at the airfoil pivot-point, set at quarter chord length from the leading edge (that is, the eading edge is at $(-0.25, 0)$ under zero incidence). The dimensions of the computational domain are $x_{-\infty} = -15$, $x_{\infty} = 40$ and $y_{\infty} = 15$, which was found to be large enough not to have a discernible influence on the results. All meshes are adapted under the constraint of a fixed number of edges $n_{el} = 115,000$. The wall-normal size of the closest near-wall element is $\sim 1.5 \times 10^{-3}$, which yields $20 - 25$ boundary-layer grid points at mid-chord under zero incidence (depending on the Reynolds number). The agent for this case is a fully-connected network with two hidden layers, each holding 4 neurons. The learning rate is set to 5×10^{-3}, and the PPO loss clipping range is $\epsilon = 0.3$.

4.1 Steady optimization at Re=10

The Reynolds number is set first to Re = 10 for the flow to remain steady regardless of the angle of incidence. As could have been expected, the upper boundary layer remains attached even for positive, non-small values of α (in practice up to $\alpha \sim 40° \pm 5°$), after which it separates and pressure drops along

[3]This is not strictly speaking a control problem because the angle of attack cannot always be tuned to improve performances. The methodology however carries over straightforwardly to related control problems, such as the optimization of multi-element high-lift systems.

[4]A deliberate abuse of language in the attempt to ease the reading. The reference data are actually VMS, not DNS, but the difference is clear from context.

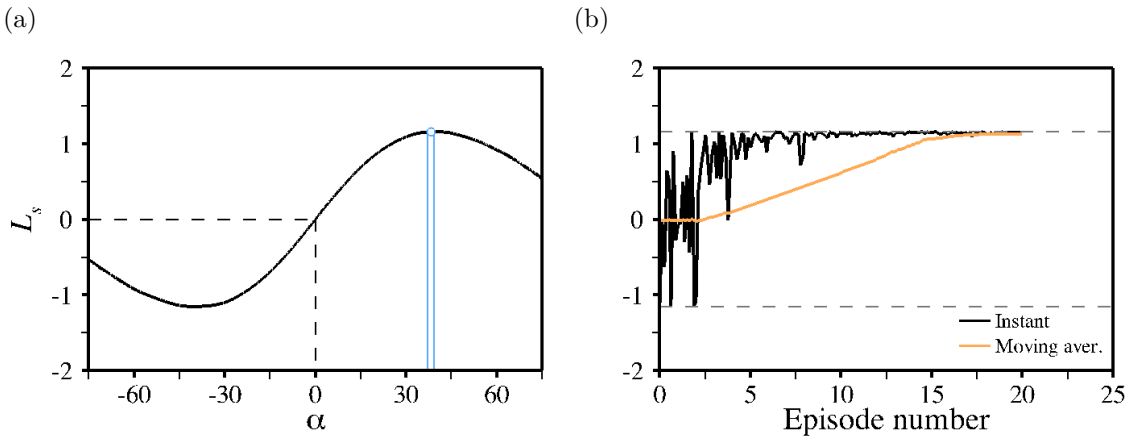

Figure 8: Flow past a NACA 0012 at Re = 10. (a) Lift against the angle of attack, as computed by DNS. The vertical blue lines are the lower and upper boundaries of the optimal range of angles obtained using DRL to solve the lift maximization problem, with the blue circle indicating the associated optimal lift. (b) Evolution per episode of the instant reward (in black) and of the moving average reward (in orange). The DNS values associated to min-max reward are reported as the horizontal dashed lines.

the upper surface, hence creating lift. This is illustrated in figure 7 showing iso-contours of vorticity distributions computed at $\alpha = 11°$, for which the boundary layer remains attached, and 51°, for which it separates. Of course, for negative values of α, the same is true of the lower boundary layer, whose separation creates a pressure drop along the lower surface, hence creating negative lift.

One objective of interest here is to maximize the steady asymptotic value of the lift coefficient L_s reached after the initial transient. We show in figure 8(a) a curve of lift against the angle of attack. It has been obtained by DNS from a sample of 30 value of α uniformly distributed in the range $[-75°; 75°]$, for the point of attack to remain at the leading edge. In each simulation, the initial condition is marched in time up to $t = 100$ with time step $\Delta t = 0.125$, at which point lift is checked to vary by less than 0.5%. The obtained distribution is perfectly antisymmetric with respect to $\alpha = 0$, as should be, and exhibits a maximum for $\alpha_{opt} = 39.5$, associated to $L_{s,opt} = 1.16$.

The DRL/single-step PPO resolution of the lift maximization problem ($r_t = L_s$) uses 8 environments and 4 steps mini-batches to update the network for 32 epochs. Convergence is achieved after 20 episodes, as indicated in figure 8(b) showing the evolution of the instant reward (in black) and of the moving average reward (in orange). This represents 160 simulations, each of which is performed on 1 core following the exact same procedure as above, and lasts 1h 25min,[5] hence $\sim$ 225h of total CPU cost to achieve convergence. The DRL singles out an optimal angle of attack in the range $\alpha_{opt} \in [37.5°; 40°]$ associated which $L_{s,opt} \sim 1.16$, which is perfectly in line with the DNS results; see the blue hue in figure 8(a).

4.2 Unsteady optimization at Re=100

The Reynolds number is now set to Re = 100 to add complexity by triggering flow unsteadiness, at least for angles of incidence above a certain threshold value. This is best seen in figure 9 showing instantaneous iso-contours of vorticity distributions computed at $\alpha = 26°$, for which the flow settles to a steady regime similar to that observed at Re=10 (although the vorticity sheets spread much further downstream), and 51°, for which the flow develops to a time-periodic, vortex-shedding state.

A relevant objective here is thus to maximize the time-averaged lift coefficient

$$\bar{L} = \frac{1}{\tau} \int_{t_i}^{t_i+\tau} L(t)\, \mathrm{d}t\,, \tag{19}$$

[5]This could scaled down to a few tens of minutes using an iterative Newton-like method, but the scope of this research is broader than just steady flows.

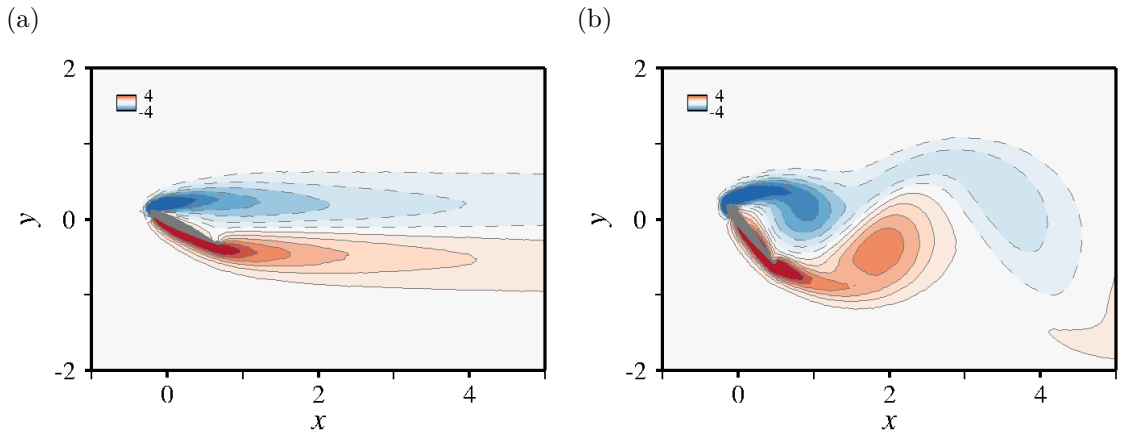

Figure 9: Flow past a NACA 0012 at Re = 100. Iso-contours of the instantaneous vorticity for (a) $\alpha = 26°$, and (b) $\alpha = 51°$.

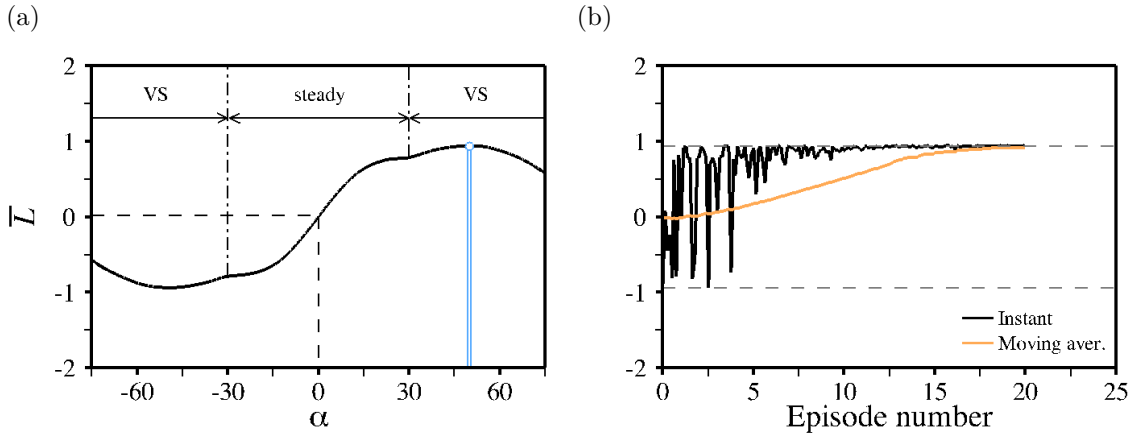

Figure 10: Flow past a NACA 0012 at Re = 100. (a) Mean lift against the angle of attack, as computed by DNS. The vertical blue lines are the lower and upper boundaries of the optimal range of angles obtained using DRL to solve the mean lift maximization problem, with the blue circle indicating the associated optimal lift. (b) Evolution per episode of the instant reward (in black) and of the moving average reward (in orange). The DNS values associated to min-max reward are reported as the horizontal dashed lines.

that mesures the mean lift force exerted on the airfoil, and reduces to the steady asymptotic lift for those values of α yielding a steady state (as long as t_i is large enough to dismiss the initial transient). figure 10(a) presents a curve of mean lift against α that has been obtained by DNS using the same sample of 30 values of $\alpha \in [-75°; 75°]$. In each simulation, the initial condition is marched in time up to $t = 50$ to leave out the transient, then up to $t = 150$ to compute (a posteriori) accurate averages. The averaging time $\tau = 100$ represents $16 - 20$ shedding cycles (depending on the angle of incidence) and has been checked to determine accurately the mean lift within 3%. The obtained distribution is again perfectly antisymmetric with respect to $\alpha = 0$, as should be. It loses continuity at $\alpha \sim 30°$, which reflects the bifurcation from the steady to the time-periodic regime through a classical linear instability mechanism (the critical value being very close to that reported in [57]), and triggers a substantial increase of lift at higher angles of incidence. The maximum is reached for $\alpha_{opt} = 50.8$, associated to $\bar{L}_{opt} = 0.94$.

Again, the DRL/single-step PPO resolution of the lift maximization problem ($r_t = \bar{L}$) uses 8 environments and 4 steps mini-batches to update the network for 32 epochs. Convergence is achieved after 22 episodes; see figure 10(b) showing the evolutions of the instant reward (in black) and of the

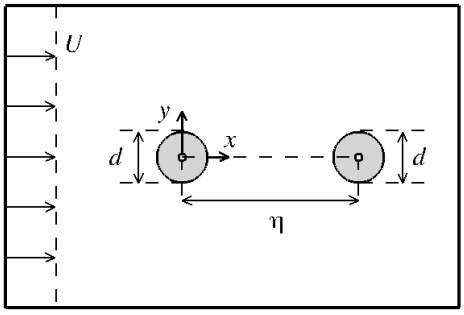

Figure 11: Schematic diagram of the flow past a tandem arrangement of two circular cylinders at zero-incidence.

moving average reward (in orange). This represents 176 simulations, each of which is performed on 1 core following the exact same procedure as above, and lasts 4h30 on average, hence approximately $\sim$ 790h of CPU cost. This is a dramatic increase with respect to the previous case at Re = 10, but one that only reflects the difference between time-dependent and time-independent CFD, as the difference is merely by 2 episodes from the standpoint of learning. The DRL returns an optimal in a range $\alpha_{opt} \in [48.4°; 50.7°]$ for which $\bar{L}_{opt} = 0.94$, again perfectly in line with the DNS values; see the blue hue in figure 10(a).

5 Optimal tandem arrangement of two circular cylinders

As a second test case, we consider the tandem arrangement of two identical circular cylinders in a uniform stream at zero-incidence. A schematic of the configuration is provided in figure 11. All variables are made non-dimensional using the cylinders diameter d and the free-stream velocity, and the sole control parameter is the center distance η between the centers of the two cylinders. While this remains a simple problem on paper (especially as the control parameter space is small enough that it again allows direct comparisons between DRL/PPO and DNS), we shall see that it brings a significant layer of complexity due to the topology of the cost functional. The origin of the coordinate system is set at the center of the upstream (main) cylinder, that is, the center of the downstream (or surrounding) cylinder is at $(\eta, 0)$. The dimensions of the computational domain are $x_{-\infty} = 15$, $x_{\infty} = 40$ and $y_{\infty} = 15$, which was found to be large enough not to have a discernible influence on the results in the range of center distances specified hereinafter. The number of mesh edges for this case is $n_{el} = 125,000$. The wall-normal size of the closest near-wall element is $\sim 10^{-3}$, which yields $20 - 25$ grid points in the boundary-layer of the main cylinder, just prior to separation (depending on the center distance and the Reynolds number). The agent for this case is similar in every respect to that used in section 4, i.e., a fully-connected network with two hidden layers, each holding 4 neurons, learning rate equal to 5×10^{-3}, and PPO loss clipping range of 0.3.

5.1 Steady optimization at Re=10

As has been done in the NACA test case, the Reynolds number is set first to Re = 10 for the flow to remain steady regardless of the center distance. This is best seen in figure 12 showing iso-contours of vorticity computed at $\eta = 3$, a small value for which the shear layer separating from the main cylinder simply engulfs its downstream counterpart, and $\eta = 11$, a value large enough for each cylinder to develop independent vorticity sheets (only those of the surrounding cylinder is of slightly lower magnitude since the wake velocity it is subjected to is smaller than the uniform stream velocity).

Since lift is inherently zero (using basic symmetry considerations), one meaningful objective for this case is to maximize the steady asymptotic value of the total drag coefficient reached after the initial transient (i.e., the non-dimensional drag force exerted on the two-cylinder system). We show in figure 13(a) a curve of drag against the center distance, that has been obtained by DNS from a sample of 11 values

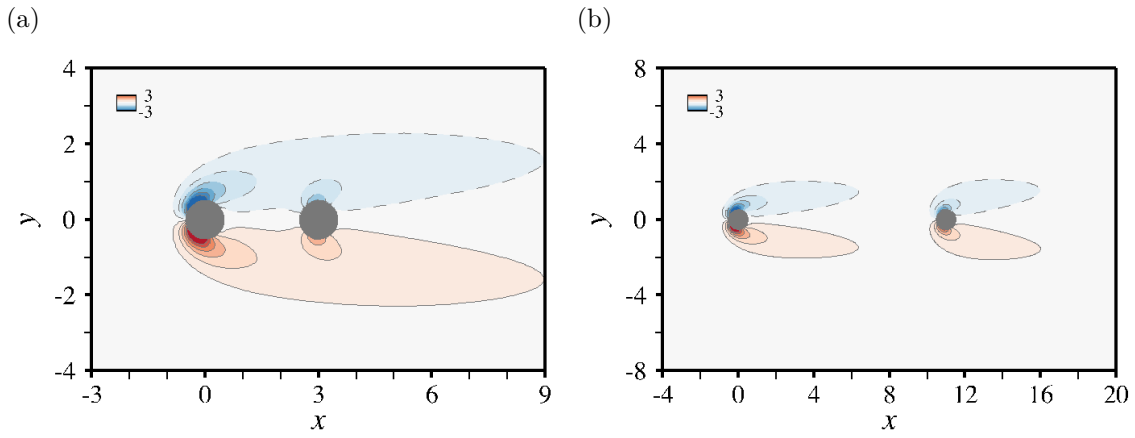

Figure 12: Flow past the tandem arrangement of two circular cylinders at Re = 10. Iso-contours of the steady vorticity for (a) $\eta = 3$, and (b) $\eta = 11$.

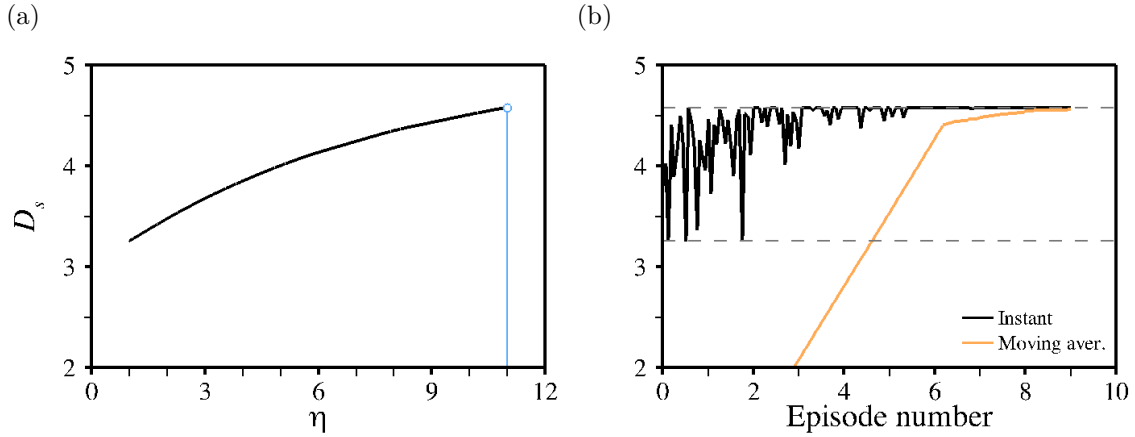

Figure 13: Flow past the tandem arrangement of two circular cylinders at Re = 10. (a) Drag against the center distance, as computed by DNS. The vertical blue line is the optimal distance obtained using DRL to solve the drag maximization problem, with the blue circle indicating the associated optimal drag. (b) Evolution per episode of the instant reward (in black) and of the moving average reward (in orange). The DNS values associated to min-max reward are reported as the horizontal dashed lines.

of η uniformly distributed in the range $[1; 11]$. We did not consider pushing the second cylinder further downstream to keep the computational costs affordable (especially in view of the time-dependent cases documented in the following), as this would have required extending the computational domain and increasing the numbers of grid points accordingly - plus we do not anticipate such large distances to be relevant from the standpoint of optimization since the interaction between both cylinders weakens increasingly, although it can take up to several tens of diameters for the sensitivity to η to go to zero. In each simulation, the initial condition is marched in time up to $t = 100$ with time step $\Delta t = 0.125$, at which point drag is checked to vary by less than 0.5%. In figure 13(a), drag is seen to increase monotonically with η, which we ascribe to the fact that the surrounding cylinder experiences negative drag if sufficiently close to the main cylinder, on behalf of the low pressures prevailing in the gap [58]. In return, the maximum is reached at $\eta_{opt} = 11$, associated to $D_{s,opt} = 4.58$.

The DRL/single-step PPO resolution of the drag maximization problem ($r_t = D_s$) uses 16 environments and 4 steps mini-batches to update the network for 32 epochs. Convergence is achieved after 9 episodes, as illustrated in figure 13(b) showing the evolutions of the instant and moving average rewards. This represents 144 simulations, each of which is performed on 1 core following the exact same procedure as above, and lasts 2h30, hence 360h of CPU cost to achieve convergence. On behalf

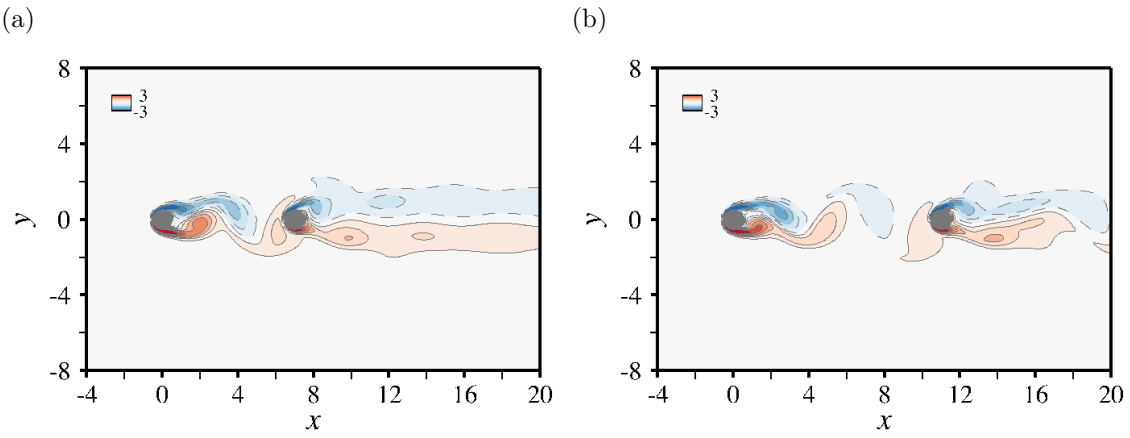

Figure 14: Flow past the tandem arrangement of two circular cylinders at Re = 100. Iso-contours of the instantaneous vorticity for (a) $\eta = 7$, and (b) $\eta = 11$.

of the monotonical behavior of drag, it is not too surprising that the DRL converges to the optimal distance $\eta_{opt} = 11$ associated to $D_{s,opt} \sim 4.58$, identical to the DNS values.

5.2 Unsteady optimization at Re=100

The Reynolds number is now set to Re = 100 to trigger flow unsteadiness. However, only a limited range of distances $\eta = [7; 11]$ is investigated because the surrounding cylinder inhibits vortex shedding from the main cylinder for sufficiently small values of η (the critical value at Re = 100 being $\eta = 3.75$-4 [59]). We use here a safe lower bound of 7 to dismiss the lose-lose outcome of near-critical distances, for which one preserves either accuracy to the detriment of longer simulation times (for slowly growing disturbances to reach saturation, which increases dramatically the computational burden), or cost, likely to the detriment of discontinuous aerodynamic forces. In the range of center distances considered, the flow has been checked to always quickly develop into a vortex-shedding state, although the pattern itself depends subtly on the center distance. Indeed, if vortex shedding is consistently observed in the wake of the main cylinder, with roughly one pair of vortices fully developing in the gap flow between the cylinders before impinging on the surrounding cylinder, a binary vortex street forms the smallest center distances; see figure 14(a) whose instantaneous iso-contour of vorticity has been computed at $\eta = 7$, while a single vortex street is recovered at larger distances; see figure 14(b) for $\eta = 11$.

One relevant objective here (in connection with fatigue reduction and energy harvesting problems in fluid-structure systems) is to maximize the root mean square (rms) lift coefficient

$$L' = \left(\frac{1}{\tau} \int_{t_i}^{t_i+\tau} (L(t) - \bar{L})^2 \, \mathrm{d}t \right)^{1/2}, \tag{20}$$

that measures the magnitude of the fluctuating lift force exerted on the two-cylinder system. figure 15(a) shows a curve of rms lift against the center distance, that has been obtained by DNS from a reduced sample of 9 values of $\eta \in [7; 11]$. In each simulation, the initial condition is marched in time up to $t = 100$ to leave out the transient, then up to $t = 200$ to compute (a posteriori) meaningful oscillation amplitudes. The averaging time $\tau = 100$ represents ~ 20 shedding cycles, and allows converging the rms lift within 5%. A maximum is reached for $\eta_{opt} = 8.8$, associated to $L'_{opt} = 0.91$.

Again, the DRL/single-step PPO resolution of the lift maximization problem ($r_t = L'$) uses 16 environments and 4 steps mini-batches. Convergence is achieved after 12 episodes, as illustrated by the instant and moving average rewards in figure 15(b). This represents 192 simulations, each of which is performed on 1 core following the exact same procedure as above, and lasts 7h30 (much longer than for the NACA 0012 test case, because all tandem solutions are unsteady), hence approximately ~ 1440h of CPU cost to achieve convergence. The so-determined optimal distance is in a range $\eta_{opt} \in [8.69; 8.83]$

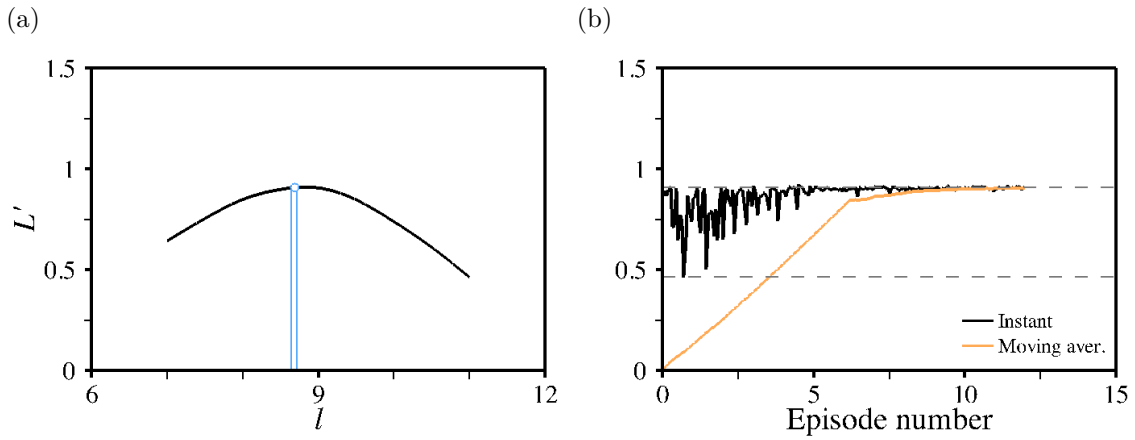

Figure 15: Flow past the tandem arrangement of two circular cylinders at Re = 100. (a) Root mean square value of lift against the center distance, as computed by DNS. The vertical blue lines are the lower and upper boundaries of the optimal range of angles obtained using DRL to solve the rms lift maximization problem, with the blue circle indicating the associated optimal lift. (b) Evolution per episode of the instant reward (in black) and of the moving average reward (in orange). The DNS values associated to min-max reward are reported as the horizontal dashed lines.

for which $L'_{opt} \sim 0.91$, almost identical to the DNS values as the discrepancy in the optimal distance is by 1%.

5.3 Unsteady optimization at Re=300

Finally, the Reynolds number is set to Re = 300 for the flow to develop into its vortex-shedding state regardless of the center distance. The full range $\eta \in [1; 11]$ is considered, therefore the flow pattern now depends dramatically on the center distance. We shall not go into the full details here, as this has been the subject of several numerical studies [60, 61, 62, 59, 63], but it is worth noticing that for small distances $\eta \lesssim 2$, the separated shear layers from the main cylinder engulf their downstream counterparts and trigger shedding vortices far from the cylinders, although the gap flow between the cylinders remains steady; see figure 16(a) showing iso-contours of vorticity computed at $\eta = 2$. For $\eta \sim 3$, the gap flow becomes unsteady, but the gap vortices are not fully developed by the time they impinge on the surrounding cylinder, hence a single vortex street is observed in the far wake; see figure 16(b) obtained for $\eta = 3$. For $\eta \sim 4$, one pair of gap vortices fully develops, then impinges on the downstream cylinder. This in turn triggers a complex interaction in the near wake of the surrounding cylinder before a vortex street eventually forms further downstream; see figure 16(c) for $\eta = 4$. Finally, for $\eta \gtrsim 5$, the wake of the surrounding cylinder becomes unsteady again, and both cylinders shed vortices that are synchronized and close to anti-phase; see figure 16(d) for $\eta = 5$.

This broad variety of flow patterns triggers different features of flow unsteadiness, which shows in the curve of rms lift against the center distance provided in 17(a). Such results have been obtained from an extended sample of 30 values of $\eta \in [1; 11]$, with each initial condition marched in time up to $t = 200$ to leave out the transient, then up to $t = 300$. The maximum is reached for $\eta_{opt} = 3.35$ (i.e., a value for which the gap vortices are close to being fully developed before impingement), associated to $L'^{opt} = 1.99$, but a close second lies at $\eta_{\sim opt} = 7.25$, associated to $L'_{\sim opt} = 1.36$. The DRL/single-step PPO resolution of the lift maximization problem uses 16 environments and 4 steps mini-batches, and convergence is achieved after 20 episodes, as illustrated by the instant and moving average rewards in figure 17(b). This represents 320 simulations, each of which is performed on 1 core following the exact same procedure as above, and lasts 10h30 (much longer than at Re=100 due to the increased simulation time; more on this in the following), hence approximately 3300h of CPU cost to achieve convergence. Interestingly, DRL converges to an optimal distance $\eta_{opt} \in [7.32; 7.38]$ for which $L'_{opt} \sim 1.39$, i.e., it fails to identify the global optimal, but does converge to a value close to the secondary DNS maximum.

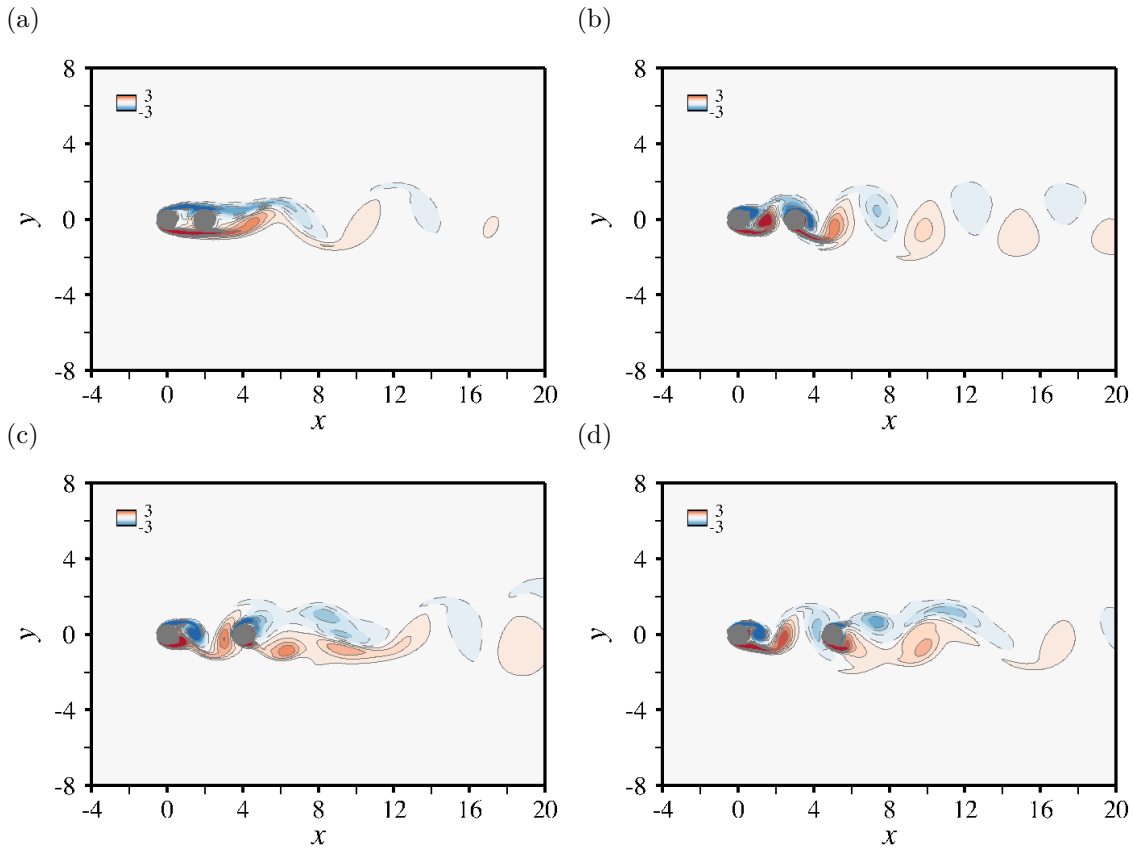

Figure 16: Flow past the tandem arrangement of two circular cylinders at Re = 300. Iso-contours of the instantaneous vorticity for (a) $\eta = 2$, (b) $\eta = 3$, (c) $\eta = 4$ and (d) $\eta = 5$.

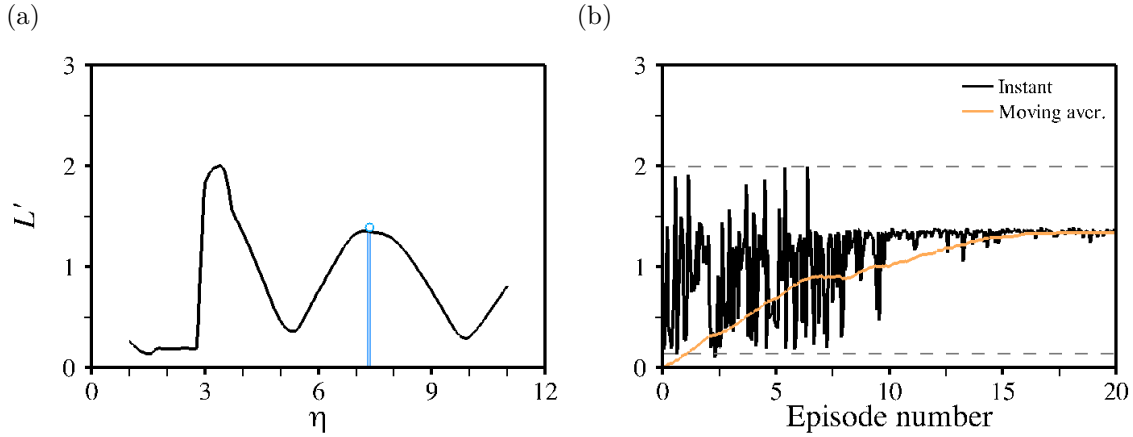

Figure 17: Flow past the tandem arrangement of two circular cylinders at Re = 300. (a) Root mean square lift against the center distance, as computed by DNS. The vertical blue lines are the lower and upper boundaries of the optimal range of angles obtained using DRL to solve the rms lift maximization problem, with the blue circle indicating the associated optimal lift. (b) Evolution per episode of the instant reward (in black) and of the moving average reward (in orange). The DNS values associated to min-max reward are reported as the horizontal dashed lines.

This half-failure can be explained by the steepness of the reward gradients with respect to the control variable in the vicinity of the global optimal, that reflects the existence of a secondary instability

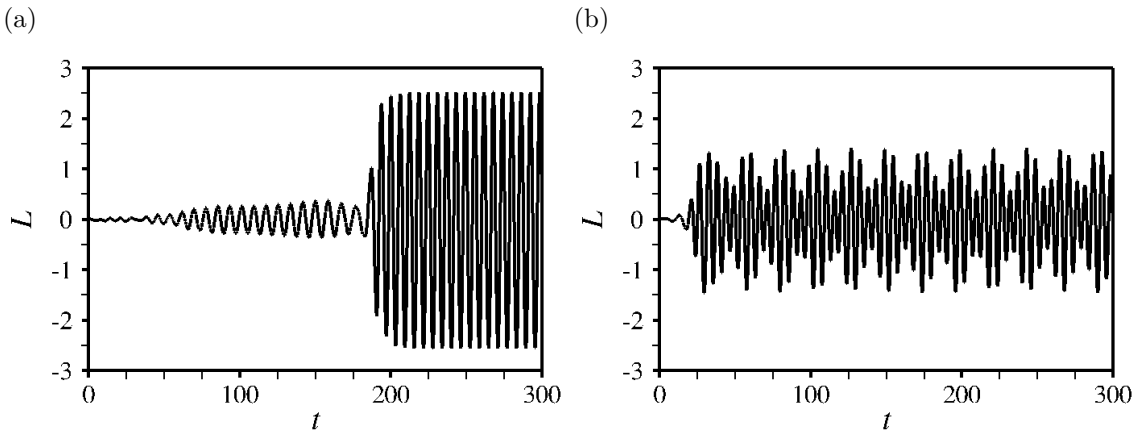

Figure 18: Flow past the tandem arrangement of two circular cylinders at Re = 300: time history of lift for (a) $\eta = 3$, (b) $\eta = 9$.

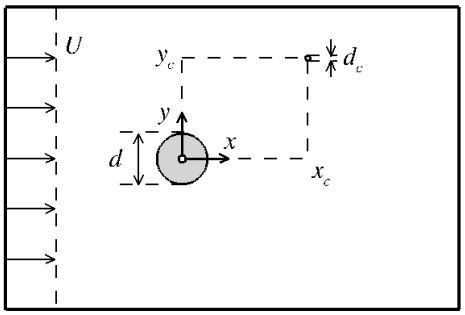

Figure 19: Schematic diagram of the flow past a two-circular cylinder system consisting of a fixed cylinder and movable control cylinder.

mechanism at play in a narrow range of center distances. This is illustrated in figure 18(a) showing that for $\eta \sim 3$, the flow settles to a first time-periodic solution, then bifurcates to a second time-periodic solution associated with increased lift oscillations (hence the large value of t_i used for this case). Actually, the DRL does identify high reward positions close to $\eta = 3$, as evidenced by the early peaks in the instant reward curve in figure 17(b), whose value $L' \sim 2$ is consistent with the global optimal. The problem is that the PPO loss is specifically built to prevent large updates of the policy (to avoid a performance collapse). In return, because there are very few times where the global maximum is met during the exploration phase (compared to the secondary one, which again stems from the topology of the reward function), the clipped policy updates only lead to limited exploration and trap the optimization process in a local maximum. It is worth insisting that these results do not cast doubt on the applicability of the method to practically meaningful high Reynolds flows, since low to moderate Reynolds numbers are likely required for such instability cascade scenario to occur. They do stress, however, that the method can benefit from carefully tuning the trade-off between exploration and exploitation, which we defer to a future publication (see also section 7 for additional comments on this matter).

6 Passive control of the cylinder flow

The third test case if that of a circular cylinder in a uniform stream, for which we assess the effect of inserting a second smaller (control) cylinder, as depicted in figure 19. All variables are made non-dimensional using the diameter d of the main cylinder and the free-stream velocity. The origin of the

coordinate system is set at the center of the main cylinder. The diameter of the control cylinder is set to $d_c = 0.1$, and the sole control parameter is the position of its center denoted by (x_c, y_c). While this may not seem overly complicated, the parameter space is now large enough do dismiss mapping the best positions for placement of the control cylinder by classical numerical procedures (to give a taste, a tens of thousands of DNS are required to cover merely a few diameters around the main cylinder). In the following, the DRL/PPO results are thus compared to theoretical predictions obtained by the adjoint method, that has proven fruitful to gain insight into the most efficient region from the linear sensitivity of the uncontrolled flow (i.e., the flow past the main cylinder), without ever calculating the controlled states. We shall not go into the technicalities of how to derive said sensitivity by the adjoint method, as this is a topic thoroughly covered in a series of recent papers [20, 64, 65] to which the interested reader is referred for further deepening. The line of thought is conversely to take the output sensitivity as a given to assess the relevance of DRL/PPO.

The agent for this case is a fully-connected network with two hidden layers, each holding 2 neurons, with the same learning rate 5×10^{-3} and PPO loss clipping range 0.3 as in sections 4 and 5. The dimensions of the rectangular computational domain are $x_{-\infty} = -15$, $x_\infty = 15$ and $y_\infty = 40$. The number of mesh edges for this case is $n_{el} = 190,000$, a substantial increase compared to the cylinder tandem case, due to the need to mesh accurately in the vicinity of the small control cylinder. The wall-normal size of the closest near-wall element is $\sim 1.5 \times 10^{-3}$ at the main cylinder, and $\sim 2 \times 10^{-3}$ at control cylinder, which yields $25 - 40$ grid points in the boundary-layer of the control cylinder (depending on the position of the control cylinder and the Reynolds number).

6.1 Steady optimization at Re=40

The Reynolds number is set first to $\mathrm{Re} = 40$ for the flow to remain steady regardless of the position of the control cylinder. This is best seen in figures 20(a)−20(c) showing iso-contours of vorticity computed for three representative positions, namely $(x_c, y_c) = (-1.1, 0)$, for which the main and control cylinder are arranged in tandem at zero incidence with the control cylinder upstream, $(x_c, y_c) = (0.02, 0.9)$ for which they are arranged side by side, and $(x_c, y_c) = (1.4, 0)$, for which they are again arranged in tandem at zero incidence, but with the main cylinder upstream. The intended objective here is to minimize the steady asymptotic value of the total drag (i.e., the non-dimensional drag force exerted on the two-cylinder system). This requires comparing the DRL values of D_s, that inherently pertain to the two-cylinder system, to those values $D_{s0} + \delta D_s$ computed by the adjoint method, where D_{s0} is the drag exerted on the main cylinder in the absence of control cylinder, and δD_s is a control-induced variation computed simultaneously for all possible positions of the control cylinder as

$$\delta D_s = \int_\Omega (\boldsymbol{u}^\dagger + 2\boldsymbol{e}_x) \cdot \delta \boldsymbol{f} \, \mathrm{d}s \,. \tag{21}$$

In equation (21), $\boldsymbol{u}^\dagger$ is a steady adjoint velocity, solution to

$$\nabla \cdot \boldsymbol{u}^\dagger = 0 \,, \qquad -\nabla \boldsymbol{u}^\dagger \cdot \boldsymbol{u} + \nabla \boldsymbol{u}^T \cdot \boldsymbol{u}^\dagger + \nabla \cdot \boldsymbol{\sigma}(-p^\dagger, \boldsymbol{u}^\dagger) = \boldsymbol{0} \,, \tag{22}$$

together with boundary condition $\boldsymbol{u}^\dagger = 2\boldsymbol{e}_x$ at the surface of the main cylinder, that we compute with the finite-element solver presented in [20]. Also, $\delta \boldsymbol{f}$ is a simple model of the force exerted by the control cylinder on the flow, namely a force opposite to the drag felt by the control cylinder in a uniform flow at the local velocity, carefully crafted to numerical and experimental data, whose relevance to mimic with good accuracy the effect of a true control cylinder in the range of Reynolds numbers considered is discussed extensively in [20].

A map of the so-computed drag variations is shown in figure 20(d), where the successful positions of the control cylinder (those for which $D_s < D_{s0}$, or equivalently $\delta D_s < 0$) are indicated by the blue hue. The map is symmetric with respect to the centerline $y = 0$ and indicates that the total drag is reduced in two distinct flow regions, a first one located approximately 1 diameter upstream of the main cylinder, and a second one located in the vicinity of the rear stagnation point and extending downstream over 3 diameters, along the outer boundary of the recirculation region. Interestingly, both regions achieve a similar drag reduction by $\sim 2\%$, although the reduction achieved in the upstream region is slightly larger. The DRL/single-step PPO resolution of the drag minimization problem ($r_t = -D_s$) uses 8 environments and a single-step mini-batch to update the network for 32 epochs,

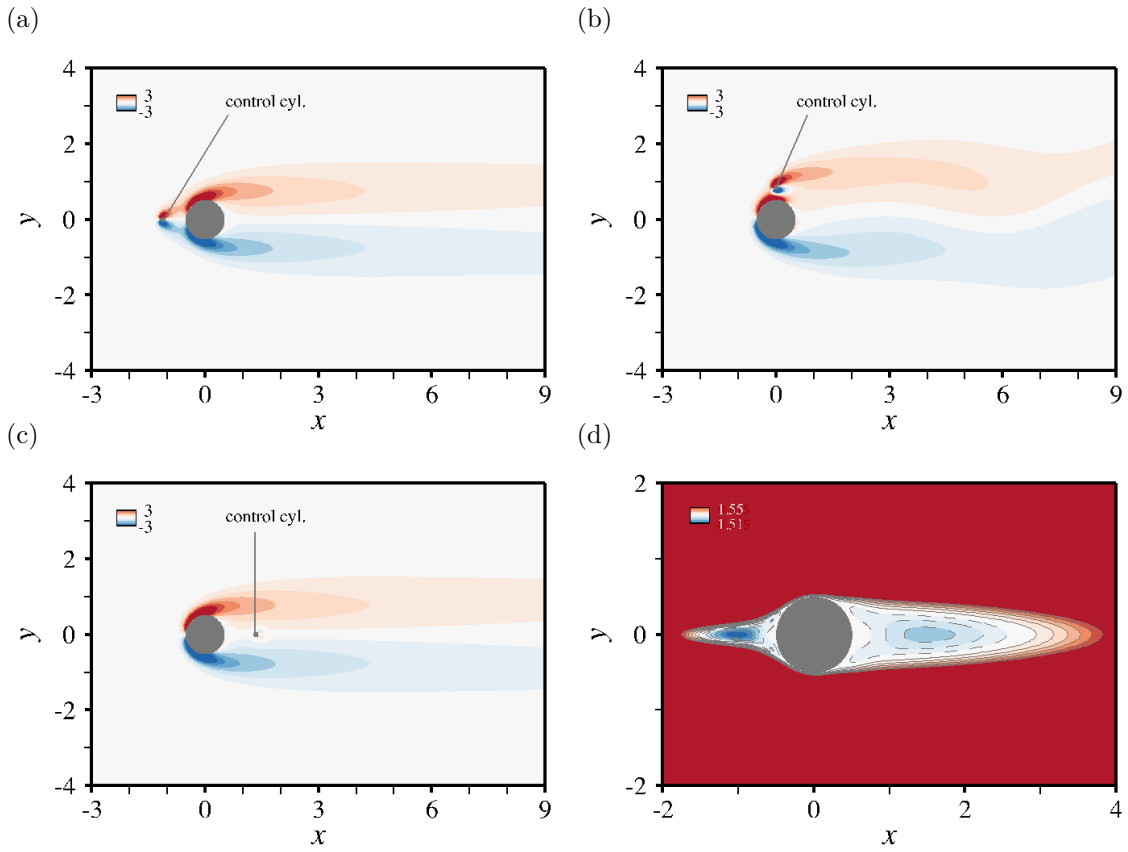

Figure 20: Flow past a two-circular cylinder system at Re = 40. $(a-c)$ Iso-contours of the steady vorticity for (a) $(x_c, y_c) = (-1.1, 0)$, (b) $(x_c, y_c) = (0.02, 0.9)$ and (c) $(x_c, y_c) = (1.4, 0)$. (d) Theoretical mean drag variation induced by a control cylinder of diameter $d_c = 0.1$, as computed using a steady adjoint method, modeling the presence of the control cylinder by a pointwise reacting force localized at the same location where the control cylinder is placed.

and restricts to positions of the control cylinder in the upper half-plane ($y > 0$) to take advantage of the problem symmetries. Convergence is achieved after 75 episodes, as illustrated by the instant and moving average rewards in figure 21(b). This represents 600 simulations in which the initial condition is marched in time up to $t = 150$ time step $\Delta t = 0.1$. Each simulation is performed on 1 core and lasts 1h30, hence approximately 900h of CPU cost to achieve convergence. This number of episodes may be deemed large compared to the other test cases documented herein, but it is easily explained by the complexity of the functional topology, that consists of two distinct valleys of comparable depth. From this point of view, the fact that the DRL achieves convergence is simply a matter of chance, as it can in theory oscillate forever between both. The positions investigated by the DRL are reported as the grey circles in figure 21(a), together with those optimal positions associated with the minimum drag $D_{s,opt} \sim 1.51$ in red. Despite some limited discrepancies regarding the spatial extent of the upstream region, the reported agreement is very satisfying, and establishes the ability of DRL to identify both regions of interest, and to predict the drag reduction achieved in these regions.

6.2 Unsteady optimization at Re=100

We set now the Reynolds number to Re = 100 for the flow to consistently develop to time-periodic vortex-shedding, i.e., the control cylinder is unable to inhibit vortex shedding from the main cylinder. In contrast, the flow past the control cylinder remains steady, as illustrated in figures 22(a)−22(c) showing iso-contours of vorticity computed for representative positions of the contour cylinder. This is

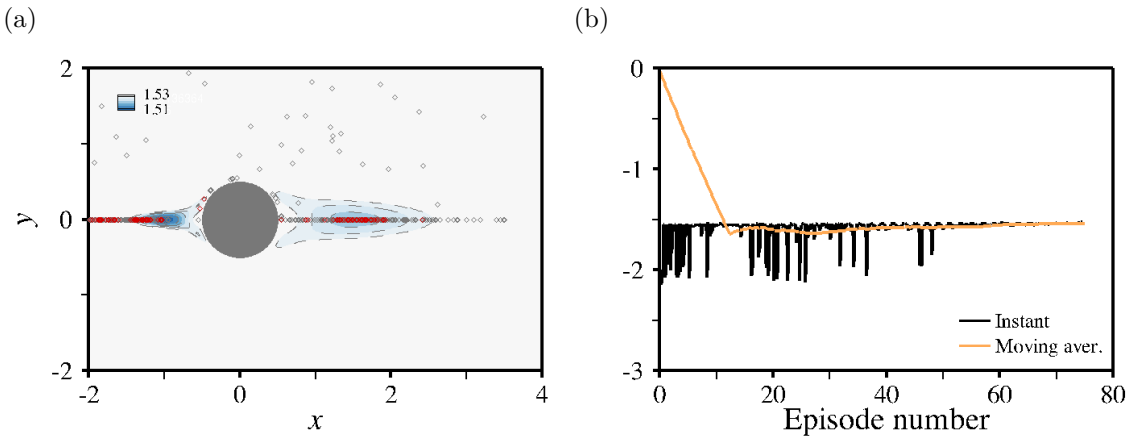

Figure 21: Flow past a two-circular cylinder system at Re = 40. (a) Variation of the total drag induced by a control cylinder of diameter $d_c = 0.1$. The grey circles are the positions investigated by the DRL. The red circles single out those DRL positions associated with an optimal reduction. The negative iso-contours of the adjoint-based variation are superimposed from figure 20(d). (b) Evolution per episode of the instant reward (in black) and of the moving average reward (in orange).

easily explained by the fact that the Reynolds number is the wake of the latter is roughly scaled by the ratio of the cylinder diameters, and therefore remains below 15 regardless of the position of the control cylinder. The intended objective here is to minimize the total mean drag. This requires comparing the DRL values of $\bar{D}$, that again inherently pertain to the two-cylinder system, to those values $\bar{D}_0 + \delta\bar{D}$ computed by the adjoint method, where $\bar{D}_0$ is the mean drag exerted on the main cylinder in the absence of control cylinder, and $\delta\bar{D}$ is the control-induced variation computed simultaneously for all possible positions of the control cylinder as

$$\delta\bar{D} = \frac{1}{\tau}\int_{t_i}^{t_i+\tau}\int_{\Omega}(\boldsymbol{u}^\dagger(t) + 2\boldsymbol{e}_x)\cdot\delta\boldsymbol{f}(t)\,\mathrm{d}s\,\mathrm{d}t\,, \tag{23}$$

with $\boldsymbol{u}^\dagger$ the time-dependent adjoint velocity, solution to

$$\nabla\cdot\boldsymbol{u}^\dagger = 0\,, \qquad -\partial_t\boldsymbol{u}^\dagger - \nabla\boldsymbol{u}^\dagger\cdot\boldsymbol{u} + \nabla\boldsymbol{u}^T\cdot\boldsymbol{u}^\dagger + \nabla\cdot\boldsymbol{\sigma}(-p^\dagger,\boldsymbol{u}^\dagger) = \boldsymbol{0}\,, \tag{24}$$

with boundary condition $\mathbf{u}^\dagger = 2\boldsymbol{e}_y$ at the surface of the main cylinder.

A map of the so-computed drag variations is reproduced from [66] in figure 22(d), where the successful positions of the control cylinder (those for which $\bar{D} < \bar{D}_0$, or equivalently $\delta\bar{D} < 0$) are indicated by the blue hue. The map is again symmetric with respect to the centerline, and indicates that the total drag is reduced in two distinct flow regions, a first one approximately 1 diameter upstream of the main cylinder, that extends upstream over 2 diameters (hence further upstream than at Re = 40) and in which drag is reduced by roughly 2%, and a second one in the vicinity of the rear stagnation point, that extends downstream over 2 diameters along the outer boundary of the mean recirculation region (shorter than that found at Re = 40), and in which drag is reduced by almost 8%. The DRL/single-step PPO resolution of the lift maximization problem($r_t = -\bar{D}$) uses again 8 environments and a single-step mini-batch, and convergence is achieved after 35 episodes, as illustrated by the instant and moving average rewards in figure 23(b). This represents 280 simulations in which the initial condition is marched in time with time step $\Delta t = 0.1$, up to $t = 100$ to leave out the transient, then up to $t = 200$ to compute relevant averages. Each simulation is performed on 1 core and lasts 4h30, hence approximately 1260h of CPU cost to achieve convergence. The reduced cost compared to the steady case is ascribed to the simpler functional topology, that features now two valleys with clearly different depths. This shows up in figure 23(a) depicting the positions investigated by the DRL (grey circles) together with those optimal positions associated with the minimum drag $\bar{D}_{opt} \sim 1.23$ in red. One sees that while a few optimal points are identified in the upstream region, DRL quickly settles for the most efficient downstream region, and more specifically the core of the mean recirculation region, in striking agreement with the adjoint-based results.

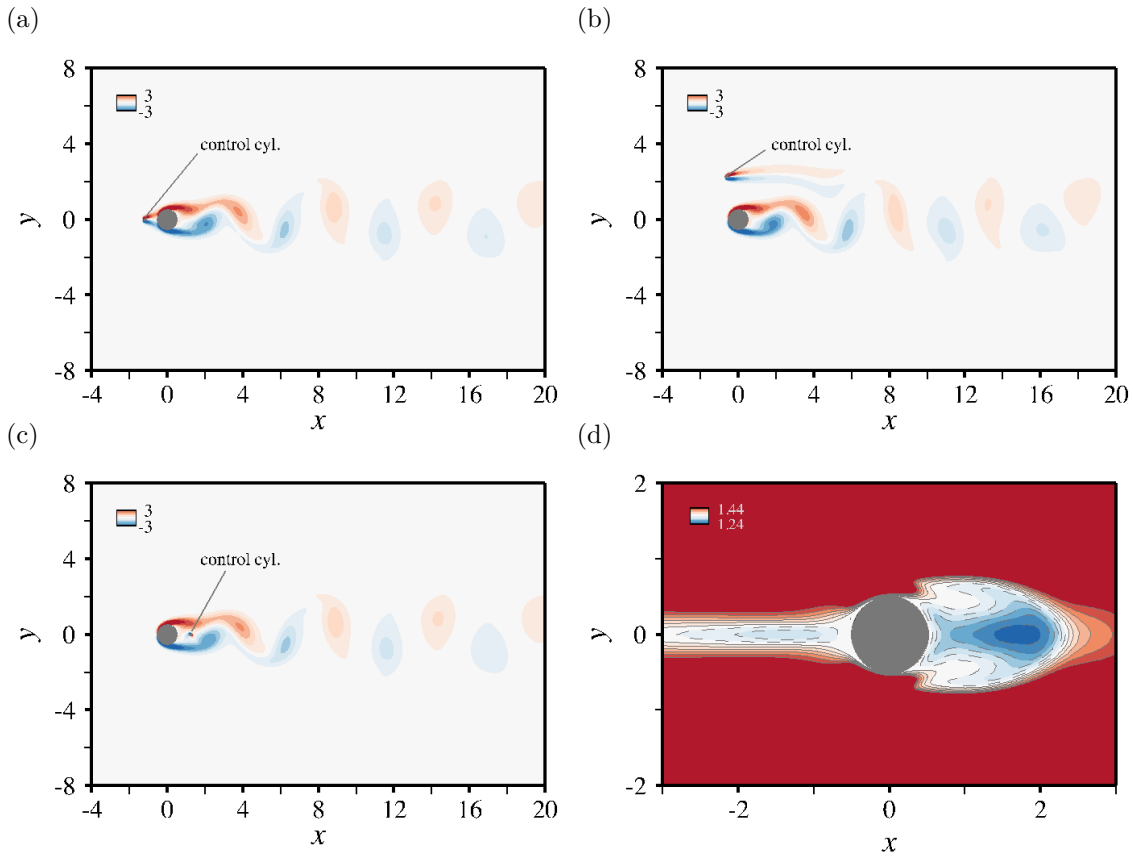

Figure 22: Flow past a two-circular cylinder system at Re = 100. $(a-c)$ Iso-contours of the instantaneous vorticity for (a) $(x_c, y_c) = (-1.2, 0)$, (b) $(x_c, y_c) = (-0.65, 2.2)$ and (c) $(x_c, y_c) = (1.2, 0)$. (d) Theoretical mean drag variation induced by a control cylinder of diameter $d_c = 0.1$, as computed using the time-varying adjoint method, modeling the presence of the control cylinder by a pointwise reacting force localized at the same location where the control cylinder is placed. Reproduced from [66].

7 Discussion and concluding remark

Fluid dynamicists have just begun to gauge the relevance of deep reinforcement learning techniques to assist the design of optimal flow control strategies. This research weighs in on this issue and shows that the proposed single-step PPO holds a high potential as a reliable, go-to black-box optimizer for complex CFD problems. The findings reported herein add value to the shallow literature on this subject, as the combination of PPO and CFD has proven fruitful to tackle several time-independent and time-dependent test cases, some of which are not tractable using canonical direct methods. Despite these achievements, there is still a need to consolidate the acquired knowledge, and to further develop, characterize and fine-tune the method, whether it be via an improved balance between exploration and exploitation to deal with steep global maxima (for instance using Trust Region-Guided PPO, as it effectively encourages the policy to explore more on the potential valuable actions, no matter whether they were preferred by the previous policies or not [43]), via non-normal probability density functions to deal with multiple global maxima, or via coupling with a surrogate model trained on-the-fly.

One point worth being emphasized is that while we do report exhaustive data regarding the computational efficiency of DRL with the hope to foster future comparisons, we did not seek to optimize said efficiency, neither by optimizing the hyper parameters, nor by using pre-trained deep learning models (as is done in transfer learning). Actually, even the most successful DRL algorithm will not be able to compete with the state-of-the art adjoint methods one generally opts for when parameter spaces are large enough to dismiss direct approaches, at least from the standpoint of CPU cost. This is because

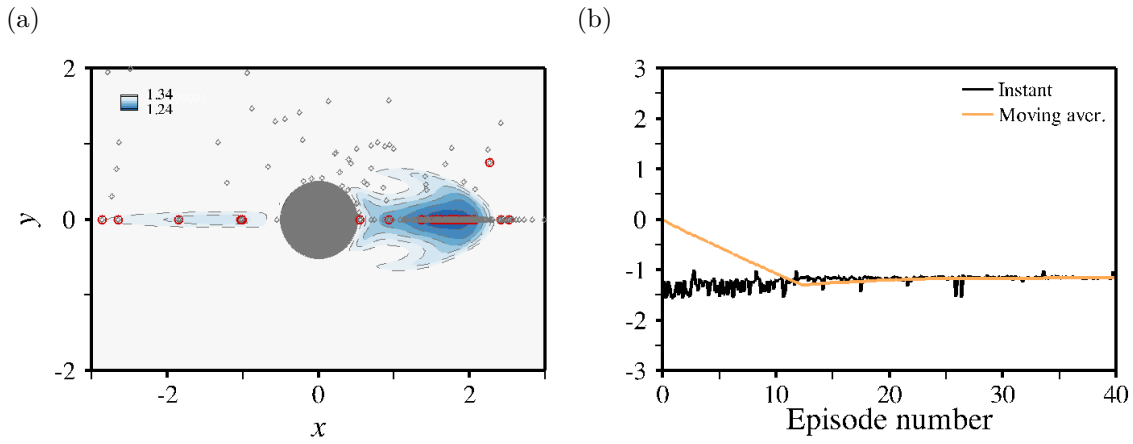

Figure 23: Flow past a two-circular cylinder system at Re = 100. (a) Variation of the total mean drag induced by a control cylinder of diameter $d_c = 0.1$. The grey circles are the positions investigated by the DRL. The red circles single out those DRL positions associated with an optimal reduction. The negative iso-contours of the adjoint-based variation are superimposed from figure 22(d). (b) Evolution per episode of the instant reward (in black) and of the moving average reward (in orange).

the adjoint method involves only two numerical simulations (one for the uncontrolled solution, one for the adjoint solution), so convergence would need to be achieved in less than two episodes (using single-step PPO) to match that cost. The one area where DRL can outperform the adjoint method is applicability. On the one hand, adjoint methods are increasingly difficult to apply rigorously as turbulence sets in, because the high sensitivity to initial conditions yields exponentially diverging adjoint solutions as soon as the length of the adjoint simulation exceeds the predictability time scale [67]. On the other hand, they suffer from the reversal in space-time directionality (evidenced by the minus sign ahead of the material derivative term in (24)) that imposes to march the adjoint equations backwards in time. This in turn requires knowledge of the history of the cylinder flow solution through the entire time-span of the adjoint simulation, which is inconvenient and very costly in terms of storage. To give a taste, it takes 45 Gb to solve the passive control case at Re = 100 using the method documented in [20] (solving first the Navier–Stokes equations, writing all time steps to disk, solving the adjoint equations over the same time interval and with the same time step, and discarding the early and late time steps corresponding to transients of the DNS and adjoint solutions to compute meaningful time averages of the adjoint-based integrands). It would therefore take up to 2 Tb to handle a simple three-dimensional (3-D) case with 40 points distributed in the span-wise direction, which is beyond feasible limits. In contrast, it would likely take minimum effort to apply DRL-CFD to 3-D, turbulent flows, as the only prerequisite is the ability to compute accurate numerical solutions (which behoves the CFD solver, not the reinforcement learning algorithm), and the storage cost of an episode in the single-step PPO is simply that of a CFD run, times the number of environments. This should opens a number of research directions, including applying the methodology to systems of industrial relevance, featuring, e.g., high Reynolds numbers, multiple phases, non-Newtonian fluids and/or coupled fluid-thermic processes.

Acknowledgements

This work is supported by the Carnot M.I.N.E.S. Institute through the M.I.N.D.S. project.

References

[1] M. Gad-el Hak. Modern developments in flow control. Appl. Mech. Rev., 49(7):365–379, 1996.

[2] J. Lumley and P. Blossey. Control of turbulence. Annu. Rev. Fluid Mech., 30:311–327, 1998.

[3] A. Glezer and M. Amitay. Synthetic jets. Annu. Rev. Fluid Mech., 34(1):503–529, 2002.

[4] S. S. Collis, R. D. Joslin, A. Seifert, and V. Theofilis. Issues in active flow control: theory, control, simulation, and experiment. Prog. Aerosp. Sci., 40:237–289, 2004.

[5] J. Kim and T. R. Bewley. A linear systems approach to flow control. Annu. Rev. Fluid Mech., 39:383–417, 2007.

[6] H. Choi, W.-P. Jeon, and J. Kim. Control of flow over a bluff body. Annu. Rev. Fluid Mech., 40:113–139, 2008.

[7] T. C. Corke, C. L. Enloe, and S. P. Wilkinson. Dielectric barrier discharge plasma actuators for flow control. Annu. Rev. Fluid Mech., 42:505–529, 2010.

[8] L. N. Cattafesta and M. Sheplak. Actuators for active flow control. Annu. Rev. Fluid Mech., 43:247–272, 2011.

[9] A. Seifert. Boundary layer separation control: Experimental perspective and modeling outlook. Annu. Rev. Fluid Mech., 50:null, 2018.

[10] M.C.G. Hall. Application of adjoint sensitivity theory to an atmospheric general circulation model. J. Atmospheric Sci., 43:2644–2651, 1986.

[11] A. Jameson. Aerodynamic design via control theory. J. Sci. Comput., 3:233–260, 1998.

[12] A. Jameson, L. Martinelli, and N.A. Pierce. Fluid dynamics optimum aerodynamic design using the Navier–Stokes equations. Theor. Comput. Fluid Dyn., 10:213–237, 1998.

[13] M. D. Gunzburger. Perspectives in flow control and optimization. SIAM, Philadelphia, 2002.

[14] B. Mohammadi and O. Pironneau. Shape optimization in fluid mechanics. Annu. Rev. Fluid Mech., 36:255–279, 2004.

[15] D. C. Hill. A theoretical approach for analyzing the restabilization of wakes. Technical memorandum NASA-TM-103858, NASA, 1992.

[16] F. Giannetti and P. Luchini. Structural sensitivity of the first instability of the cylinder wake. J. Fluid Mech., 581:167–197, 2007.

[17] O. Marquet, D. Sipp, and L. Jacquin. Sensitivity analysis and passive control of cylinder flow. J. Fluid Mech., 615:221–252, 2008.

[18] J. O. Pralits, L. Brandt, and F. Giannetti. Instability and sensitivity of the flow around a rotating circular cylinder. J. Fluid Mech., 650:513–536, 2010.

[19] D. Sipp, O. Marquet, P. Meliga, and A. Barbagallo. Dynamics and control of global instabilities in open-flows: a linearized approach. Appl. Mech. Rev., 63:030801, 2010.

[20] P. Meliga, E. Boujo, G. Pujals, and F. Gallaire. Sensitivity of aerodynamic forces in laminar and turbulent flow past a square cylinder. Phys. Fluids, 26:104101, 2014.

[21] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. Nature, 323:533–536, 1986.

[22] A. Krizhevsky, Ilya Sutskever, and G.E. Hinton. Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst., 25:1097?1105, 2012.

[23] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. The International Journal of Robotics Research, 32(11):1238–1274, 2013.

[24] V. Mnih, K. Kavukcuoglu, D. Silver, Rusu. A.A., Veness J., M.G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. Nature, 518:7540, 2015.

[25] B. Lusch, J. N. Kutz, and S. L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. Nature communications, 9(1):1–10, 2018.

[26] M. Raissi, A. Yazdani, and G. E. Karniadakis. Hidden Fluid Mechanics: A Navier-Stokes Informed Deep Learning Framework for Assimilating Flow Visualization Data. arXiv, 2018.

[27] A. D. Beck, D. G. Flad, and C.-D. Munz. Deep Neural Networks for Data-Driven Turbulence Models. arXiv, 2018.

[28] S. L. Brunton, B.R. Noack, and P. Koumoutsakos. Machine learning for fluid mechanics. Annu. Rev. Fluid Mech., 52:477–508, 2020.

[29] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of go without human knowledge. Nature, 550(7676):354–359, 2017.

[30] A.V. Bernstein and E.V. Burnaev. Reinforcement learning in computer vision. Proc. SPIE 10696, 10th International Conference on Machine Vision (ICMV 2017), 2018.

[31] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. arXiv e-prints, 2015.

[32] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms. arXiv e-prints, July 2017.

[33] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter. Learning agile and dynamic motor skills for legged robots. Science Robotics, 4(26):eaau5872, 2019.

[34] P. Ma, Y. Tian, Z. Pan, B. Ren, and D. Manocha. Fluid directed rigid body control using deep reinforcement learning. ACM Transactions on Graphics (TOG), 37(4):1–11, 2018.

[35] J. Rabault, M. Kuchta, A. Jensen, U. Réglade, and N. Cerardi. Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control. Journal of Fluid Mechanics, 865:281–302, 2019.

[36] H. Tang, J. Rabault, A. Kuhnle, Y. Wang, and T. Wang. Robust active flow control over a range of Reynolds numbers using an artificial neural network trained through deep reinforcement learning. Phys. Fluids, 32:053605, 2020.

[37] J. Viquerat, J. Rabault, A. Kuhnle, H. Ghraieb, and E. Hachem. Direct shape optimization through deep reinforcement learning. arXiv preprint arXiv:1908.09885, 2019.

[38] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. Neural Networks, 2(5):359–366, 1989.

[39] I. Goodfellow, Y. Bengio, and A. Courville. The Deep Learning Book. MIT Press, 2017.

[40] Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. MIT press, 2018.

[41] A. Kakade. A natural policy gradient. Adv. Neural Inf. Process Syst., 14:1531–1538, 2001.

[42] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust Region Policy Optimization. arXiv e-prints, February 2015.

[43] Y. Wang, H. He, X. Tan, and Y. Gan. Trust region-guided proximal policy optimization. arXiv preprint arXiv:1901.10314, 2019.

[44] T. J. R. Hughes, G. R. Feijóo, L. Mazzei, and J.-B. Quincy. The variational multiscale method - a paradigm for computational mechanics. Comput. Methods Appl. Mech. Engrg., 166:3–24, 1998.

[45] R. Codina. Stabilization of incompressibility and convection through orthogonal sub-scales in finite element methods. Comput. Methods Appl. Mech. Engrg., 190:1579–1599, 2000.

[46] Y. Bazilevs, V. M. Calo, J. A. Cottrell, T. J. R. Hughes, A. Reali, and G. Scovazzi. Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. Comput. Methods Appl. Mech. Engrg., 197:173–201, 2007.

[47] E. Hachem, B. Rivaux, T. Kloczko, H. Digonnet, and T. Coupez. Stabilized finite element method for incompressible flows with high Reynolds number. J. Comput. Phys., 229(23):8643–8665, 2010.

[48] R. Codina. Stabilized finite element approximation of transient incompressible flows using orthogonal subscales. Comput. Methods Appl. Mech. Engrg., 191:4295–4321, 2002.

[49] E. Hachem, S. Feghali, R. Codina, and T. Coupez. Immersed stress method for fluid-structure interaction using anisotropic mesh adaptation. Int. J. Numer. Meth. Eng., 94:805–825, 2013.

[50] E. Hachem, H. Digonnet, E. Massoni, and T. Coupez. Immersed volume method for solving natural convection, conduction and radiation of a hat-shaped disk inside a 3d enclosure. International Journal of numerical methods for heat & fluid flow, 2012.

[51] T. Coupez, G. Jannoun, N. Nassif, H. C. Nguyen, H. Digonnet, and E. Hachem. Adaptive time-step with anisotropic meshing for incompressible flows. J. Comput. Phys., 241:195–211, 2013.

[52] J. Bruchon, H. Digonnet, and T. Coupez. Using a signed distance function for the simulation of metal forming processes: formulation of the contact condition and mesh adaptation. Int. J. Numer. Meth. Eng., 78:980–1008, 2004.

[53] V. John. Reference values for drag and lift of a two-dimensional time-dependent ow around a cylinder. Int. J. Numer. Meth. Fluids, 44:777–788, 2004.

[54] V. John. Parallele Lösung der inkompressiblen Navier–Stokes Gleichungen auf adaptiv verfeinerten Gittern. PhD thesis, Otto-von-Guericke-Universität Magdeburg, Fakultät für Mathematik, 1997.

[55] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu. Stable baselines. https://github.com/hill-a/stable-baselines, 2018.

[56] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.

[57] G. Dergham. Étude de la stabilité du lâcher tourbillonnaire d?un profil NACA 0012. PhD thesis, Master Thesis Université Pierre et Marie Curie, 2007.

[58] M.M. Zdravkovich. Review of interference-induced oscillations in flow past two parallel circular cylinders in various arrangements. J. Wind Eng. Ind. Aerod., 28:183–199, 1988.

[59] K. Lee, K.-S. Yang, and D.-H. Yoon. Flow-induced forces on two circular cylinders in proximity. Comput. Fluids, 38:111–120, 2009.

[60] S. Mittal, V. Kumar, and A. Raghuvanshi. Unsteady incompressible flows past two cylinders in tandem and staggered arrangements. Int. J. Numer. Meth. Fl., 25:1315–1344, 1997.

[61] J.R. Meneghini, F. Saltara, C.L.R. Siqueira, and J.A. Ferrari. Numerical simulation of flow interference between two circular cylinders in tandem and side-by-side arrangements. J. Fluids Struct., 15:327–350, 2001.

[62] B. Sharman, F.-S. Lien, L. Davidson, and C. Norberg. Numerical predictions of low reynolds number flows over two tandem circular cylinders. Int. J. Numer. Meth. Fl., 47:423–447, 2005.

[63] W. Zhang, H.-S. Dou, and Y. Li. Unsteady characteristics of low-Re flow past two tandem cylinders. Theor. Comput. Fluid Dyn., 32:475–493, 2018.

[64] X. Mao. Sensitivity of forces to wall transpiration in flow past an aerofoil. Proc. R. Soc. A, 471:20150618, 2015.

[65] P. Meliga, E. Boujo, M. Meldi, and F. Gallaire. Revisiting the drag reduction problem using adjoint-based dis- tributed forcing of laminar and turbulent flows over a circular cylinder. Eur. J. Mech. B-Fluid, 72:123–134, 2018, accepted.

[66] P. Meliga. Computing the sensitivity of drag and lift in flow past a circular cylinder: time-stepping vs. self-consistent analysis. Phys. Rev. Fluids, 2:073905, 2017.

[67] Q. Wang and J.-H. Gao. The drag-adjoint field of a circular cylinder wake at reynolds numbers 20, 100 and 500. J. Fluid Mech., 730:145–161, 2013.