Noname manuscript No.
(will be inserted by the editor)

A statistical learning strategy for closed-loop control of fluid flows

Florimond Guéniat · Lionel Mathelin · M. Yousuff Hussaini

Received: date / Accepted: date

Abstract This work discusses a closed-loop control strategy for complex systems utilizing scarce and streaming data. A discrete embedding space is first built using hash functions applied to the sensor measurements from which a Markov process model is derived, approximating the complex system's dynamics. A control strategy is then learned using reinforcement learning once rewards relevant with respect to the control objective are identified. This method is designed for experimental configurations, requiring no computations nor prior knowledge of the system, and enjoys intrinsic robustness. It is illustrated on two systems: the control of the transitions of a Lorenz 63 dynamical system, and the control of the drag of a cylinder flow. The method is shown to perform well.

Keywords Closed-loop control, Reinforcement learning, Machine learning

1 Introduction

While the design and capability of aircraft, and more generally of complex systems, have significantly improved over the years, closed-loop control can bring further improvement in terms of performance and robustness to non-modeled perturbations. In the context of flow control, closed-loop control however suffers from severe limitations preventing its use in many situations. As a paradigmatic example, a typical turbulent flow involves both a large range of spatial scales and

Florimond Guéniat
Department of Mathematics,
Florida State Univ.,
32306-4510 Tallahassee, FL, USA E-mail: florimond@gueniat.fr

Lionel Mathelin
LIMSI-CNRS, rue J. von Neumann,
Campus Universitaire d'Orsay,
91405 Orsay cedex, France

M. Yousuff Hussaini
Department of Mathematics,
Florida State Univ.,
32306-4510 Tallahassee, FL, USA

exhibits a rich and fast dynamics. High frequency phenomena hence require a control command fast enough to adjust to the current state of the quickly evolving flow system. Indeed, frequencies of interest can routinely lie over 1 kHz, leading to a very short period of time for the controller to synthesize the command based on its knowledge of the state of the system.

While flow manipulation and open-loop control are common practice, much fewer successful closed-loop control efforts are reported in the literature. Further, many of them rely on unrealistic assumptions. For example, Model Predictive Control (MPC) approaches require very significant computational resources to solve the governing equations in real-time. If a Reduced-Order Model (ROM) is employed, as is common practice to alleviate the CPU burden, one often needs to observe the whole system for deriving the ROM as, for instance, the velocity or pressure fields with Proper Orthogonal Decomposition (POD), see [1–6]. Hence, flow control with this class of approaches is restricted to numerical simulations or experiments in a wind tunnel equipped with sophisticated visualization tools such as Particle Image Velocimetry (PIV).

This paper discusses a *practical* strategy for closed-loop control of complex flows by alleviating the limitations of current methods. The present work relies on a change of paradigm: we want to derive a general nonlinear closed-loop flow control methodology suitable for *actual* configurations and as realistic as possible. No *a priori* model, nor even a model structure, describing the dynamics of the system is required to be available. The approach proposed is *data-driven only*, with the sole information about the system given by scarce and spatially-constrained sensors. The method then exploits statistical learning methods.

This is the framework one typically deals with in practical situations where the amount of information on the system at hand is limited and usually comes from a few sensors located at the boundary of the fluid domain, *e.g.*, on solid surfaces. The resulting information takes the form of short time-dependent vectors with as many entries as sensors.

Among the few earlier efforts relying on streaming measurements from a few sensors, a trained neural network using surface measurements is employed to reduce the drag of a turbulent channel flow with an opposition control strategy in [7]. In [8], pressure sensors and an auto-regressive model (AutoRegression with eXogenous inputs, ARX) are used to reduce flow-induced cavity tones. An autoregressive approach is also followed in [9] and [10], while a genetic programming technique is adopted in [11] to control a separated boundary layer. Interested readers may refer to [12] for a comprehensive review of the topic. The present approach aims at deriving an efficient, yet robust, nonlinear closed-loop control method compliant with actual situations. Among its distinctive features compared to other methods is a combination of *both* performance and fast learning.

To facilitate *learning* about the system dynamics from the time-dependent measurements, and the subsequent derivation of a control strategy, the problem must be amenable to a finite dimension. Hence, one needs to discretize the infinite-dimensional time-series of the sensors' information. To this end, the streaming data are convolved with a kernel which should result in a discrete image space. Locality-Sensitive Hash (LSH) functions are used for that purpose, [13], which results in a low-dimensional discrete state space. Transitions from state to state in this discrete space describe the dynamics, [14], and allow the analysis to learn, and update in real-time, a Markov process model of the system. A suitable discretization of the

dynamics allows the derivation of a reinforcement learning-based control strategy of the identified Markov process model of the system. The control of Markov processes is a mature field, [15] and reinforcement learning, [16,17], is a suitable class of methods for the control of Markov processes, see for instance [18,19] for the control of 1-D and 2-D chaotic maps. As will be seen in the application examples below, the resulting control strategy is data-driven only, intrinsically robust against perturbations in the flow and does not require significant computational resources nor prior knowledge of the flow. The proposed approach is experiment-oriented and on-going efforts are carried-out to demonstrate it on an experimental open cavity flow in a turbulent regime. This will be the subject of a subsequent publication.

The paper is organized as follows. The framework and basics of how hash functions are used to generate a low-dimensional state space are discussed in Section 2. Section 3 is concerned with learning an efficient control strategy for the system modeled as a stochastic process living in a small dimensional space. The resulting control strategy is illustrated and discussed in the case of the control of a Lorenz 63 system and the drag reduction of a cylinder in a two-dimensional flow in Section 4. Concluding remarks close the paper in Section 5.

2 Hash functions for reduced order modeling

2.1 Preliminaries

Consider a dynamical system evolving on a manifold $\mathcal{D}$:

$$\dot{X}(t) = \mathfrak{f}(X(t)), \qquad X \in \mathcal{D},$$

with X the state of the system and $\mathfrak{f}$ the flow operator. Let $\boldsymbol{g} : \mathcal{D} \to \mathbb{R}^{n_p}$ be a sensor function. In the sequel, the number of sensors n_p will be taken to be one but generalization to more sensors is immediate. The observed data $y \in \mathbb{R}$ is defined as $y(t) := \boldsymbol{g}(X(t))$.

Let Δt be the sampling rate of the measurement system. Sampling has to be fast enough to capture the small time scales of the dynamics of y. The data coming from the sensor are embedded in a reconstructed phase space Ω:

$$(y(t)\ \ y(t-\Delta t) \dots y(t-(n_e-1)\Delta t)) =: \boldsymbol{y}(t) \in \Omega \subset \mathbb{R}^{n_e}.$$

The correlation dimension of y is estimated from the time-series, for instance using the Grassberger-Proccacia algorithm, [20]. It allows the definition of the embedding dimension n_e as, at least, twice the correlation dimension. Under mild assumptions, this resulting embedding dimension ensures there is a diffeomorphism between the phase space $\mathcal{D}$ and the reconstructed phase space, [21], so that $\boldsymbol{y}$ is an observable on the system.

2.2 Hash functions

A hash function is any function that can be used to map an entry $\boldsymbol{y} \in \mathbb{R}^{n_e}$ to a *key* $s \in \mathbb{Z}$. Since the key is an integer, hash functions effectively result in a discrete image space of $\boldsymbol{y}$. Hash functions are often used to efficiently discriminate

two different entries so that slightly different input data should result in a large variation of the associated key, [22]. An important objective in choosing the hash function is to avoid *collisions*, *i.e.*, when two different entries are associated with the same key.

Conversely, the need for identification of similar entries in large databases has led to the use of the Locality-Sensitive Hash functions (LSH) [23,13]. In contrast with most hash functions, they are designed to promote collisions when two entries are close to each other. The idea is that, if two points are close in $\mathbb{R}^{n_e}$, they should be likely to remain close after a projection on a lower dimensional subspace. The Johnson-Lindenstrauss lemma (JLL), [24], provides useful results to reach this objective and motivates the use of LSHs. Specifically, the JLL provides probabilistic guarantees of the near preservation of relative distances between objects in high-dimension after projection onto random low-dimensional spaces.

Let $\boldsymbol{v} \in \mathbb{R}^{n_e}$, $\|\boldsymbol{v}\|_2 = 1$, be a test vector. The function $h^{\boldsymbol{v},w} : \mathbb{R}^{n_e} \to \mathbb{N}$ is a LSH, [23]:

$$h^{\boldsymbol{v},w}(\boldsymbol{y}) := h_0 + \left\lfloor \frac{\boldsymbol{v} \cdot \boldsymbol{y}}{w} \right\rfloor, \tag{1}$$

where $\lfloor \cdot \rfloor$ is the floor operator, $w > 0$ a quantization length and h_0 is such that $h^{\boldsymbol{v},w} > 0$, $\forall \boldsymbol{y} \in \Omega$, $\boldsymbol{v} \in B_2^{n_e}(1)$, with $B_2^{n_e}(1)$ the unit ball of $\mathbb{R}^{n_e}$ in the sense of L^2. As an illustration, following [24,13], if two observables $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ are such that $\|\boldsymbol{y}_1 - \boldsymbol{y}_2\|_2 < r$, with $r = w/2\,\|\boldsymbol{v}\|_2$, they are associated with the same key with a probability p_1 larger than $1/2$. On the other hand, the probability p_2 that two distant points $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ appear close to each other in the sense of $h^{\boldsymbol{v},w}$ is a function of the angle between $(\boldsymbol{y}_2 - \boldsymbol{y}_1)$ and $\boldsymbol{v}$ and is smaller than p_1.

Let us illustrate the LSH with a simple example. Let $n_e = 30$ and $\boldsymbol{y}^a, \boldsymbol{y}^b$ and $\boldsymbol{y}^c$ be three vectors from $\mathbb{R}^{n_e}$ such that:

$$\begin{aligned} y_j^a &= \cos\left(2\,\pi\ (j-1)\right), \qquad 1 \le j \le n_e, \\ y_j^b &= \cos\left(2\,\pi\ (j)\right), \\ y_j^c &= \cos\left(2\,\pi\ (j+2)\right). \end{aligned}$$

Let $\boldsymbol{v}$ be a unit-norm normal Gaussian vector of $\mathbb{R}^{n_e}$, $w = 0.2$ and $h_0 = 0$. The different vectors are drawn in Fig. 1. Upon processing with the LSH, the keys associated with the three vectors $\boldsymbol{y}^a, \boldsymbol{y}^b$ and $\boldsymbol{y}^c$ are:

$$h^{\boldsymbol{v},w}(\boldsymbol{y}^a) = 3, \qquad h^{\boldsymbol{v},w}\left(\boldsymbol{y}^b\right) = 3, \qquad h^{\boldsymbol{v},w}(\boldsymbol{y}^c) = 1.$$

$\boldsymbol{y}^a$ and $\boldsymbol{y}^b$ are closer to each other, in the sense of their L^2-distance, than to $\boldsymbol{y}^c$, and indeed, vectors $\boldsymbol{y}^a$ and $\boldsymbol{y}^b$ are associated with the same key $s = 3$ while $\boldsymbol{y}^c$ is associated with $s = 1$.

To discriminate false neighbors with higher probability, projections on several low-dimensional subspaces can be used. Consider the hash function $\mathfrak{h} : \mathbb{R}^{n_e} \to \mathcal{S} \subset \mathbb{N}$ made of concatenation of n_v keys $\{h^{\boldsymbol{v}_l,w_l}(\boldsymbol{y})\}_{l=1}^{n_v}$. Many choices can be made for the test vectors $\{\boldsymbol{v}_l\}_l$. For instance, they could be the principal axes of the manifold on which the observable $\boldsymbol{y}$ evolves, or may be randomly selected from a Gaussian distribution. Keys (*i.e.*, objects in the image space of $\mathfrak{h}$) generate a Voronoï paving of the observable space. Each key is associated with a cell, or *state* s, and close observables are associated with the same key.

Coarseness of the paving depends on the quantization lengths $\{w_l\}_l$ and the number n_v of test vectors. The set $\{w_l\}_l$ quantifies the minimal length of the

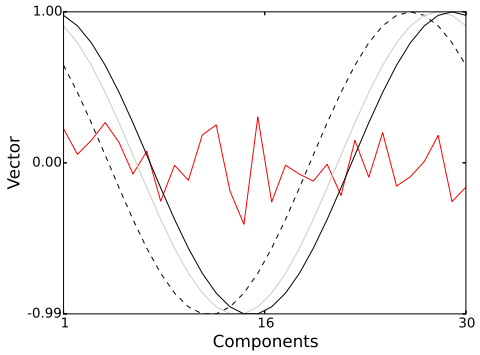


Fig. 1 Plot of the vectors $\boldsymbol{y}^a$ and $\boldsymbol{y}^b$ (solid lines) and $\boldsymbol{y}^c$ (dashed line). The random test vector $\boldsymbol{v}$ is plotted in red.

cells[1] and, as w_l increases, the cardinality $n_{\text{key}} := \text{card}\,(\mathcal{S})$ of the image space of $\mathfrak{h}$ decreases. On the other hand, increasing n_v is equivalent to refining the description, *i.e.*, increasing the cardinality of $\mathcal{S}$.

3 State-driven control

Thanks to the hash function, the original infinite-dimensional system is approximated as a discrete stochastic process whose state space is spanned by the keys. The system is observable in real-time in this discrete space since evaluating the hash function with the streaming sensor data $\boldsymbol{y}$ can be done at no computational cost. Under a discrete control command $a \in \mathcal{A}$ to the actuators, hereafter termed the *action*, the dynamics of the system will be modified and the goal is to find the action which makes the physical system at hand satisfy the control objective, say, a target dynamics. The discretized description of the system with the hash function naturally lends itself to a Markovian framework, [25,26], which is adopted below.

3.1 Markov Decision Process

Let $p\,(i,l,j)$ be the probability of transition from a state $s^{(i)} \in \mathcal{S}$ to $s^{(j)} \in \mathcal{S}$ under an action $a^{(l)} \in \mathcal{A}$, $\mathcal{A} := \left\{a^{(l)} \in \mathbb{R},\, l \in \mathcal{I}_{\mathcal{A}} := [1, n_a] \subset \mathbb{N}\right\}$ being a uniformly discretized space of possible control actions. Here again, the analysis is restricted to one actuator but generalization to more actuators is immediate. Actions $\left\{a^{(l)}\right\}_{l=1}^{n_a}$ index the discrete commands available to the controller. Define $\mathcal{S} := \left\{s^{(i)}, i \in \mathcal{I}_{\mathcal{S}} := [1, n_{\text{key}}] \subset \mathbb{N}\right\}$ and $s_k \equiv s\,(t_k := k\,\Delta t)$. Similarly, $a_k \equiv a\,(t_k)$. To each transition-action $(i \to j, l)$ is associated a *transition reward* (TR)

[1] Two vectors $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ are associated with two different keys if their L^2-norms differ by more than $w/|\cos\left(\angle\left(\boldsymbol{v}, \boldsymbol{y}_1 - \boldsymbol{y}_2\right)\right)|$, hence the minimal length w of the cells.

$R\left(s^{(i)}, a^{(l)}, s^{(j)}\right)$ [2]. The goal of the control strategy is hence to identify the optimal *policy* $\pi : \mathcal{S} \to \mathcal{A}$, which describes the best action to apply when in a given state so as to maximize the *value*, V_i^π, defined as the sum of the future expected rewards of the policy when starting at state $s^{(i)}$:

$$V_i^\pi := \lim_{K\to\infty} \mathbb{E}\left[\sum_{k=1}^{K} \gamma^{k-1} R\left(s_k, \pi\left(s_k\right), s_{k+1}\right)\right], \qquad s_1 = s^{(i)}, \tag{2}$$

where $\mathbb{E}\left[\cdot\right]$ is the expectation operator over all possible sequences $\{s_k\}_{k=1}^{K}$ under the policy π. Here, γ is the discount rate, $0 < \gamma < 1$. By weakly accounting for TRs occurring far in the future, the discounted rate effectively introduces a time horizon. The values express the expectation of the cumulative TRs of a given policy. Eq. (2) may be reformulated as, [15]:

$$V_i^\pi = R_i^\pi + \gamma \sum_{j\in\mathcal{S}} p(i, \pi\left(s^{(i)}\right), j)\, V_j^\pi, \tag{3}$$

where $R_i^\pi := \mathbb{E}\left[R\left(s_k = s^{(i)}, \pi\left(s_k\right), s_{k+1}\right)\right]$ is the mean TR in state $s_k = s^{(i)}$ under the policy π. This corresponds to the celebrated Bellman equation, [27], written in the discrete settings.

3.2 Identification of the rewards

The transition rewards are unknown *a priori* and depend on the control objective. As perhaps more familiar to the reader, one could define the cost as the opposite of the reward. Consequently, to learn relevant rewards, consider the cost function $\mathcal{J}$ associated with the control objective:

$$\mathcal{J}\left(t_n\right) := \sum_{k=0}^{\infty} \gamma^k \left(\mathcal{C}\left(t_{n+k+1}\right) + \rho\, \left|a\left(t_{n+k}\right)\right|^2\right), \tag{4}$$

with $\mathcal{C}$ the measure of performance, *e.g.*, the drag coefficient in the included example. The contribution of the control intensity $|a|^2$ to the cost with respect to the measure of performance $\mathcal{C}$ is weighted by $\rho > 0$.

Let the immediate rewards $R_{\text{imm}}\left(s_n, a_n, s_{n+1}\right)$ represent, at any given time t_n, the negative of the contribution to the cost $\mathcal{J}\left(t_n\right)$ of the transition from the present state $s_n \equiv s\left(t_n\right)$ to state s_{n+1}, under an action $a_n \equiv a\left(t_n\right)$:

$$R_{\text{imm}}\left(s_n, a_n, s_{n+1}\right) := -\left(\mathcal{C}\left(t_{n+1}\right) + \rho\, \left|a\left(t_n\right)\right|^2\right). \tag{5}$$

R_{imm} is then high when the performance associated with the controlled system is good, and low otherwise. Instead of the reward associated with a particular trajectory in the original, infinite-dimensional, phase space, the *average*, trajectory-independent, reward associated with all trajectories leading to a given transition $(s_n \to s_{n+1}, a_n)$ should be determined. The ergodicity assumption postulates the

[2] We use a slight abuse of notation as we consider $R\left(s^{(i)}, a^{(l)}, s^{(j)}\right) \equiv R\left(i, l, j\right)$ and $p\left(i, a^{(l)}, j\right) \equiv p\left(i, l, j\right)$. Both R and p are $n_{\text{key}} \times n_a \times n_{\text{key}}$ three-way tensors.

equivalence of a temporal and an ensemble average, *i.e.*, here $\lim_{K\to\infty} K^{-1} \sum_{k=0}^{K-1} f(s_k) = \mathbb{E}[f(s)]$. Under this assumption, the mean transition rewards, in the sense of the probability distribution of R_{imm}, are finally determined during the learning stage *via*:

$$R(s_n, a_n, s_{n+1}) \leftarrow (1-\alpha_n) \underbrace{R(s_n, a_n, s_{n+1})}_{\text{old value}} + \alpha_n R_{\text{imm}}(s_n, a_n, s_{n+1}), \qquad (6)$$

with $\alpha_n > 0$ the learning rate. For the TRs to be associated with the average contribution to the cost function of the transition-action $(s_n \to s_{n+1}, a_n)$, the learning rate is set to $\alpha_n = 1/N^{(i,l,j)}$, where $N^{(i,l,j)} \in \mathbb{N}$ corresponds to the number of times the transition-action $\left(s^{(i)} \to s^{(j)}, a^{(l)}\right)$ has occurred so far during the learning process.

3.3 Reinforcement learning

To derive a control strategy, one needs to determine a policy which would give the best control action given the current state of the system, as known through the hash function. The rewards associated with an action when in a given state have been learned and this information is now used to derive a control policy to drive the system along transitions and actions associated with the largest rewards.

When the probabilities of transition from a state to another are known, the policy may be identified by means of a dynamic programming algorithm, [27, 15]. However, the distribution of transition probabilities and values $\{V_i^\pi\}_{i\in\mathcal{I}_\mathcal{S}}$ are difficult to reliably evaluate since neither the control policy nor the transition probabilities are stationary during the learning stage. In this situation, *Reinforcement Learning* is a suitable class of methods for the control of Markov processes, [16,28,29].

Among these methods, the *Q-learning* approach consists in relying on the estimation of the *Q-factors*, or *action-values*, Q^π which evaluate the expected reward of a state-action combination:

$$Q^\pi(i,l) := \left\langle R_i\left(a^{(l)}\right)\right\rangle + \gamma \sum_{j\in\mathcal{S}} p(i,\pi(i),j)\, V_j^\pi, \qquad (7)$$

where $\left\langle R_i\left(a^{(l)}\right)\right\rangle := \mathbb{E}\left[R\left(s_k = s^{(i)}, a^{(l)}, s_{k+1}\right)\right]$ is the empirical mean TR associated with applying the action $a^{(l)}$ when in the state $s^{(i)}$. From Eq. (3), the action-value is hence the expected average reward of applying $a^{(l)}$ while in state $s^{(i)}$ and subsequently following the policy π.

As stated previously, the transition probabilities can not be accurately estimated. However, an iterative estimation of the Q-factors can be derived, [16,29, 17]. Letting the initial Q-factors be given, the Q-factor associated with a state $s^{(i)}$ and an action $a^{(l)}$ can be updated at time t_n as follows:

$$Q^{\pi}(i,l) \leftarrow \underbrace{Q^{\pi}(i,l)}_{\text{old value}} + \alpha_n \, \Delta Q^{\pi}, \tag{8}$$

$$\Delta Q^{\pi} := \left(R\left(s_n = s^{(i)}, a^{(l)}, s_{n+1}\right) + \gamma \underbrace{\max_{\tilde{l} \in \mathcal{I}_{\mathcal{A}}} Q^{\pi}\left(i, \tilde{l}\right)}_{\text{“best” value}} - \underbrace{Q^{\pi}(i,l)}_{\text{old value}} \right),$$

where $\alpha_n > 0$ is a learning factor. It can be shown that Q^{π} converges to the true Q-factors when $n \to \infty$ if the following conditions hold, [16]: *i)* the TRs R are bounded, *ii)* $0 < \alpha_n < 1$, $\forall n \in \mathbb{N}$, *iii)* $\sum_{n=1}^{\infty} \alpha_n \to \infty$ and *iv)* $\sum_{n=1}^{\infty} (\alpha_n)^2 < \infty$.

The action-value $Q^{\pi}(i,l)$ will increase when the reward associated with the 2-tuple $\left(s^{(i)}, a^{(l)}\right)$ is good, *i.e.*, such that $\Delta Q^{\pi} > 0$, and decrease otherwise. To learn a good policy, the system in different states is stimulated with different actions to estimate the Q-factors. The control policy is the action which, for each state, is associated with the largest Q-factor, [16, 29].

3.4 Robustness

A critical property of any realistic and practically useful control strategy is its resilience with respect to unpredictable events. These events encompass exogenous/external perturbations to the flow, sensor noise, actuator noise, etc. A control strategy robust to these perturbations is passive and brings the flow back to its nominal controlled state after the perturbation is gone. As will be demonstrated in the application example below, see section 4.2.3, the present strategy is robust thanks to two properties:

First, the state of the system, as estimated via the LSH, is discrete. The locality sensitivity property of LSHs implies that a small perturbation $\boldsymbol{\epsilon}$ of the measurement vector $\boldsymbol{y}$ will likely result in the same key as the unperturbed measurement, $h^{\boldsymbol{v},w}(\boldsymbol{y} + \boldsymbol{\epsilon}) = h^{\boldsymbol{v},w}(\boldsymbol{y})$. More precisely, the state estimation is strictly robust to any perturbation of energy $\|\boldsymbol{\epsilon}\|_2$ with probability $\prod_{l=1}^{n_v} \max\left[0, 1 - \|\boldsymbol{\epsilon}\|_2 \, |\cos(\angle(\boldsymbol{\epsilon}, \boldsymbol{v}_l))|/w\right]$.

Second, the proposed method is also robust against perturbations of the flow dynamics. The learning of the control strategy relies on an ensemble average of rewards and Q-factors and then results in an optimal control policy in the ensemble mean-sense. Hence robustness against noise.

4 Results

4.1 Dynamical system

To illustrate the methodology discussed above, we now consider the Lorenz 63 system [30], defined by the following equations:

$$\begin{cases} \frac{\mathrm{d}X_1}{\mathrm{d}t} = \sigma\left(X_2 - X_1\right) + f_{X_1}\left(X, t\right) \\ \frac{\mathrm{d}X_2}{\mathrm{d}t} = X_1\left(r - X_3\right) - X_2 + f_{X_2}\left(X, t\right) \\ \frac{\mathrm{d}X_3}{\mathrm{d}t} = X_1 \times X_2 - bX_3 + f_{X_3}\left(X, t\right), \end{cases} \tag{9}$$

with the common parameters $(\sigma, r, b) = (10, 20, 8/3)$. A chaotic attractor defines the dynamics, structured as two "wings" around two fixed points of the system. The state vector evolves on a wing, turning around a fixed point, before eventually jumping to the other wing. f_{X_i} is the action on the component i of $\boldsymbol{X} := (X_1\, X_2\, X_3)$. The chosen control objective is to remain on the "left" wing, $X_1 \leq 0$. The measure of the performance is given as the distance between the state vector and the left fixed point $\boldsymbol{X}^\star$ of the system, $\mathcal{C} := \left\|\boldsymbol{X} - \boldsymbol{X}^\star\right\|_2$, with $\boldsymbol{X}^\star := \left(-\sqrt{b\left(r-1\right)}, -\sqrt{b\left(r-1\right)}, r-1\right)$.

The observable $\boldsymbol{y}$ is constructed from the time series $\{y\left(t - n\,\Delta t\right)\}_{n=0}^{n_e-1}$ of X_2, sampled every $\Delta t = 0.025$ time units. In this illustration, only f_{X_1} is different from zero so that the Lorenz system is controlled only through the time-derivative $\mathrm{d}X_1/\mathrm{d}t$ of the first component of its state vector. It mimics a realistic scenario, where sensors and actuators are distinct. Actuation values lie between $[-26, 26]$ with a discretization step of 4. This leads to $n_a = 14$ different actuations to control the system. The cost function to minimize is given by Eq. (4) and the control command is penalized with $\rho = 0.1$.

The embedding dimension is set to $n_e = 8$. The first three singular vectors of a $n_e \times N$ Hankel matrix, built on $N = 500$ measures $\{y\left(t - n\,\Delta t\right)\}_{n=0}^{N-1}$, are used as the $n_v = 3$ test vectors $\{\boldsymbol{v}_l\}_{l=1}^{n_v}$ of the LSHs. The quantization length is set to $w = 45$ and leads to a cardinality of the set of keys of $n_{\mathrm{key}} = 14$.

Once the test vectors are determined from the Hankel matrix, the states are to be identified via the LSHs, see Fig. 2(a). It is hence possible to infer both the dynamics of the system from the state transitions and the rewards associated with the objective. The mean transition probabilities from one state to another, after five time increments in order to improve visualization, are plotted in Fig. 2(b). As expected for the Lorenz system, the dynamics are seen to be driven by two main cycles, see Fig. 4. There are two statistically dominant sequences of transitions cycling around all states between 1 (resp. 5) and 4 (resp. 8), corresponding to the right (resp. left) wing of the Lorenz attractor. States 9 to 11 and 12 to 14 represent sub-transitions from a main cluster to another one.

From the observations of the system, one learns the $n_{\mathrm{key}} \times n_a \times n_{\mathrm{key}} \simeq 2100$ transition rewards matrix R and the $n_{\mathrm{key}} \times n_a \simeq 150$ Q-factors matrix Q by repeatedly applying Eq. (6) and Eq. (9). The transition matrix is sparse, see Fig. 2(b), and many transitions hence hardly ever occur. Thus, the learned TRs and the Q-factors matrices are also sparse, see Fig. 2(c) and Fig. 2(d). The difference between one wing and the other is clearly discernible in the rewards, the first block being

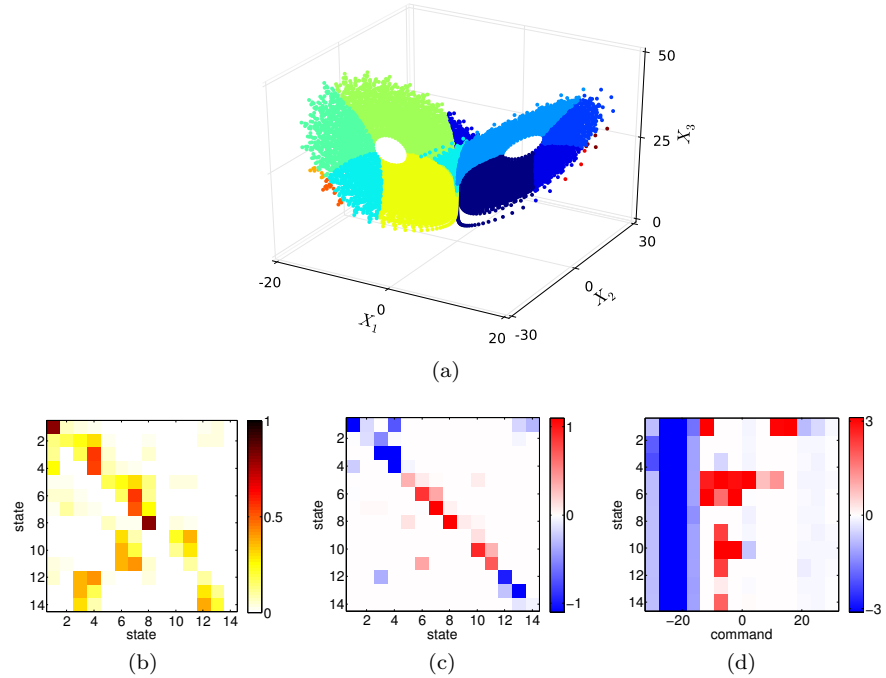


Fig. 2 (a): Identified clusters (color-coded) for the Lorenz 63 system. (b): Mean state transition probabilities. The transition matrix has been iterated five times for readability. (c): Transition rewards associated with a state transition, for the NULL command. The TRs have been rescaled between -2 and 2 and the colormap saturated, in order to improve readability. (d): Q-factors.

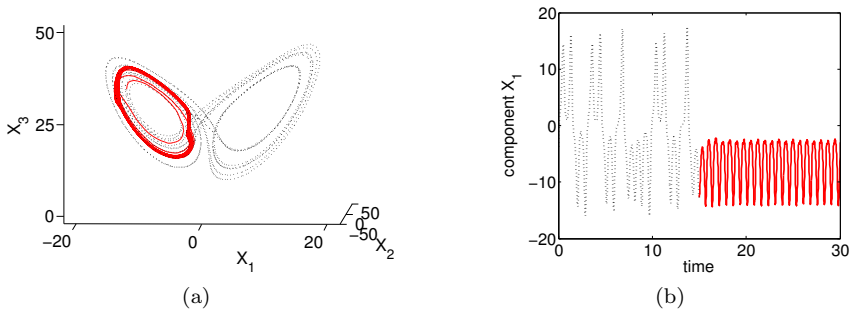


Fig. 3 Uncontrolled Lorenz system in dashed black. Controlled response in solid red. The identified strategy is applied at time $t = 15$. (a): State phase. (b): X_1 component.

associated with negative rewards while the second block is associated with positive rewards. By construction of the rewards, the use of a strong (expensive) command is also discouraged, as can be seen in the Q-factors, Fig. 2(d). The identified control strategy succeeds in staying on the left wing, see Fig. 3.

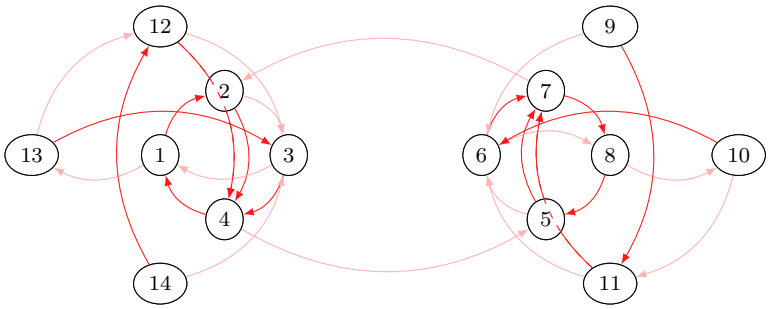


Fig. 4 Transitions between clusters for the Lorenz system. Only the first two most probable transitions are represented. The dark red arrows represent the most probable transitions from one cluster to another, while the pale red represents the second most probable transitions.

4.2 Two-dimensional numerical flow

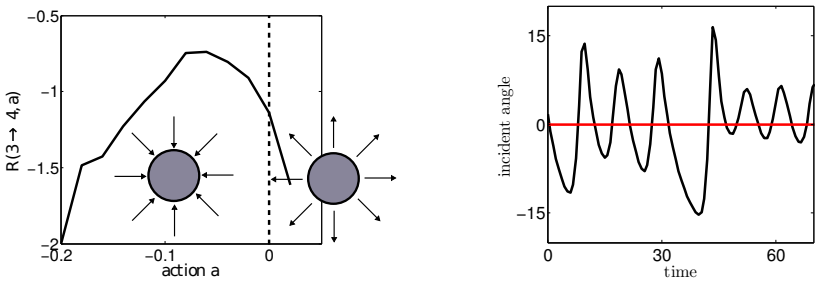
To further illustrate the methodology discussed above, consider a 2-D laminar flow around a circular cylinder, in two situations: a fixed, or random in time angle of the incoming flow.

4.2.1 Configuration of the test case

The considered Reynolds number of the flow is $Re = 200$ based on the cylinder diameter and the upstream flow velocity. Details of the simulation can be found in [31]. The observable $\boldsymbol{y}$ is constructed from the time series $\{y\,(t - n\,\Delta t)\}_{n=0}^{n_e-1}$ of a single pressure sensor, sampled every $\Delta t = 0.75$ time units. This sensor is located on the cylinder surface at an angle of 160 degrees from the upstream stagnation point when the angle is fixed. In this case, the pressure signal oscillates with a period of about 9 time units and half as much for the drag.

Actuation, *i.e.*, control of the flow is achieved via blowing or suction through the whole cylinder surface. Actuation values lie within $[-0.2, 0.02]$, with a discretization step of 0.02, ranging from suction to blowing. It leads to $n_a = 12$ different actuations to control the system. The cost function to minimize is given in Eq. (4), with $\rho = 23$. It corresponds to the drag ($\mathcal{C}$ is the drag coefficient) induced by the cylinder penalized with the intensity of the command. The penalty ρ was chosen so that the resulting command remains within the operating range of the actuator. The embedding dimension is set to $n_e = 14$. The first five singular vectors of a $n_e \times N$ Hankel matrix, built on $N = 500$ measures $\{y\,(t - n\,\Delta t)\}_{n=0}^{N-1}$, are used as the $n_v = 5$ test vectors $\{\boldsymbol{v}_l\}_{l=1}^{n_v}$ of the LSHs. The quantization length is set to $w = 50$ and leads to a cardinality of the set of keys of $n_{\text{key}} = 15$.

Two situations are considered to illustrate the proposed control algorithm. First, the incident angle of the incoming flow is kept constant at its zero nominal value, Sec. 4.2.2. Alternatively, in Sec. 4.2.3, the angle is a random process, smooth in time, whose realizations range from -20 to 20 degrees around the nominal value with a uniform probability distribution, see Fig. 5(b).



(a) (b)

Fig. 5 (a): Transition rewards associated to the transition $3 \to 4$, with respect to the command. The TRs have been rescaled between -2 and 2. The drawings illustrate the actuation (suction or blowing). (b): Time-evolution of the incident angle of the incoming flow, for the noiseless (red) and noisy (black) case.

4.2.2 Noiseless case: nominal incidence

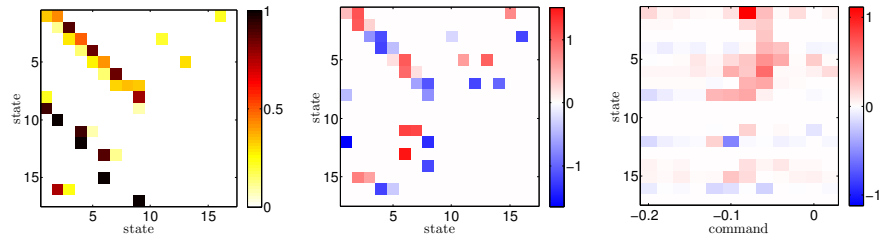
Once the test vectors are determined from the Hankel matrix, and the states are identified via $\mathfrak{h}$, it is possible to infer the dynamics of the system and the associated rewards. The learned transition reward $R\left(3, a^{(l)}, 4\right)$ is plotted in Fig. 5(a). It clearly exhibits a maximum which corresponds to the best compromise between a sufficient decrease of $\mathcal{C}$ while a reasonable increase of $|a|^2$. The estimated transition probabilities from one state to another are plotted in Fig. 6(a). In the present case, the dynamics are seen to be rather periodic, with a statistically dominant sequence of transitions cycling around all states between 1 and 8, see Fig. 7. Other states are "transient" states between two stages of the main cycle. For instance, a transition from state 1 to state 16 can occur with a low probability but the next transition will be to state 2 (or, less likely, state 3). Further analysis of the transition matrix can give more insights into the dynamics and on the relevance of other sequences, *e.g.*, via stability analysis, [14], symbolic dynamics based on keys, [32], or the Kullback-Leibler entropy, [14].

From the observations of the system, one learns the $n_{\text{key}} \times n_a \times n_{\text{key}} \simeq 2700$ transition rewards matrix R and the $n_{\text{key}} \times n_a \simeq 180$ Q-factors matrix Q by repeatedly applying Eq. (6) and Eq. (9). As in the Lorenz system case (see above in Sec. 4.1), the transition matrix is sparse, as can be appreciated from Fig. 6(a), and many transitions hence hardly ever occur. Thus, the learned TRs and the Q-factors matrices are also sparse, see Fig. 6(b) and Fig. 6(c).

The performance of the control strategy is assessed in terms of the performance indicator η defined as the difference between the time-averaged cost $\langle \mathcal{J} \rangle := \langle \mathcal{J}(t) \rangle_t$ and the time-averaged cost with a NULL strategy ($a = 0$), $\langle \mathcal{J}_{\text{NULL}} \rangle$. The performance indicator is scaled with the difference between $\langle \mathcal{J}_{\text{NULL}} \rangle$ and the cost $\langle \mathcal{J}_{\text{ORA}} \rangle$ of an optimal time-invariant control strategy:

$$\eta := \left(\langle \mathcal{J}_{\text{NULL}} \rangle - \langle \mathcal{J} \rangle \right) / \left(\langle \mathcal{J}_{\text{NULL}} \rangle - \langle \mathcal{J}_{\text{ORA}} \rangle \right). \tag{10}$$

The evolution of η as a function of the learning effort, defined as the number N of measurements during the learning phase, is plotted in Fig. 8(a). The performance indicator increases with the amount of learning for computing the TRs, but quickly reaches a plateau. The TR matrix being sparse, the information of



(a) (b) (c)

Fig. 6 (a): Mean state transition probabilities. (b): Transition rewards associated with a state transition, for the NULL command. The TRs have been rescaled between -2 and 2 and the colormap saturated, in order to improve readability. (c): Q-factors.

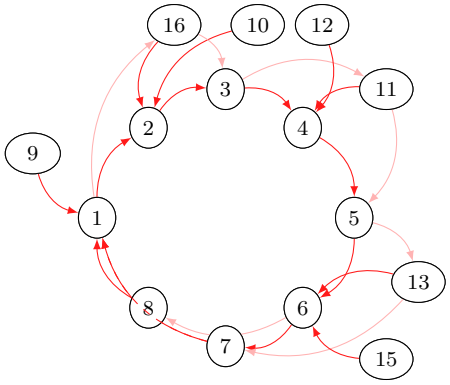


Fig. 7 Transitions between clusters for the cylinder flow. Only the first two most probable transitions are represented. The dark red arrows represent the most probable transitions from a cluster to another one, while the pale red represents the second most probable transitions.

the learning stage focuses onto a limited number of unknowns and the Q-factors quickly converge.

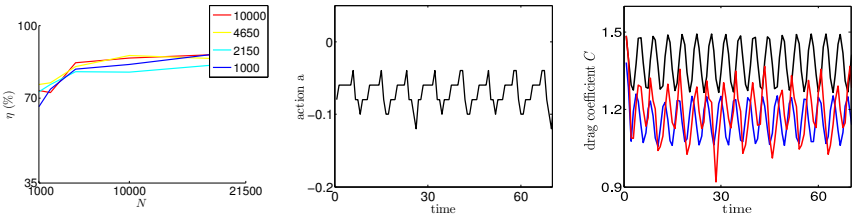
As expected, the values $\{V_i^\pi\}_{i\in\mathcal{I}_\mathcal{S}}$, computed with the learned policies are larger than the values computed with the ORA policy, on average, see Tab. 1.

Policy	NULL	ORA	Present strategy
Average value $\langle V^\pi \rangle$	-15.8	9.5	13.2

Table 1 Average expected value $\langle V^\pi \rangle := \mathbb{E}_{i\in\mathcal{I}_\mathcal{S}}\left[V_i^\pi\right]$ associated with the NULL, ORA and present strategy policies. The learning effort for the rewards (resp. the Q-factors) was 10000.

The resulting control command is plotted in Fig. 8(b) and is seen to exhibit an oscillatory behavior. The associated drag coefficient of the cylinder flow is plotted in Fig. 8(c) for the present approach as well the NULL and the ORA strategy. The identified control is seen to perform well. The drag coefficient for the identified control resembles the one given by the ORA control.

To illustrate the impact of the control on the system, the time-averaged pressure field around the cylinder is plotted in Fig. 9, both with and without control.



(a) (b) (c)

Fig. 8 Illustration of the control policy. (a): Performance indicator with respect to the learning effort of the transition rewards matrix R. Colors (from blue to red) encode different efforts in learning Q. (b): Actuation as a function of time, $a(t)$. (c): Time-evolution of the drag coefficient under three different control policies: zero command (NULL, black), ORA strategy (blue), and the optimal policy from the present approach (red).

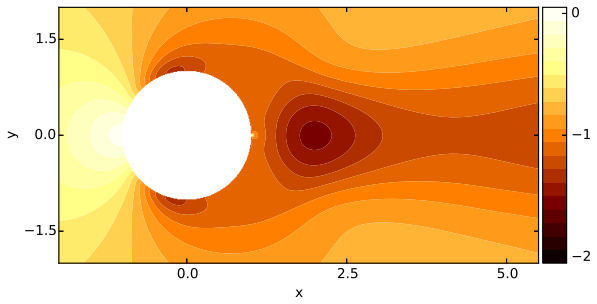
When control is applied, the pressure difference between the upstream stagnation point and the rear cylinder vicinity is significantly reduced. Further insights about the control effect can be gained by examining Fig. 10 where the time-averaged streamlines are plotted. With control, *i.e.*, with a negative normal velocity at the cylinder surface, small recirculation bubbles significantly weaken, or even vanish, and the separation of the boundary layer from the cylinder surface is postponed further downstream, reducing the effective width of the wake. The length L_r of the recirculation bubble drops from $L_r = 1.11$ in the case of the NULL command (no control), see Fig. 9(a), to $L_r = 0.95$ when the command identified by the present control approach is applied, see Fig. 9(b). The length of the recirculation bubble is hence reduced by 15%. Suction at the cylinder surface tends to slightly increase the viscous drag (thinner boundary layers with larger velocity gradients) but significantly decreases the width of the wake and the pressure defect at the back of the cylinder, producing a lower drag.

4.2.3 Measurements with noise

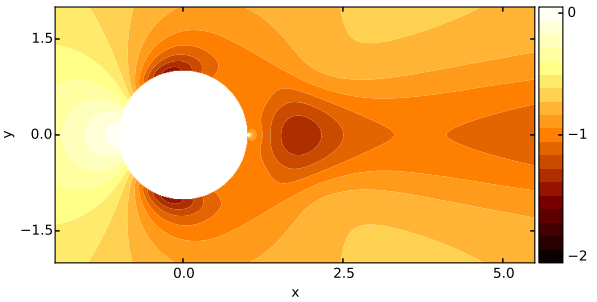
To investigate the robustness of our control strategy, the angle of the incident flow is made to vary randomly between -20 and 20 degrees around its nominal value with a uniform probability distribution and a smooth time-evolution, see Fig. 5(b) for a typical realization. This mimics a typical class of perturbations to the system at hand and allows the determination of the robustness of the control strategy.

As in the noiseless case considered in Sec. 4.2.2 above, the transition probability, rewards and Q-factors matrices are all sparse (not shown for sake of brevity). The performance indicator η of the derived policy is depicted in Fig. 11(a). In contrast with the noiseless case, η is here strongly dependent on the effort in learning. At early stages of the learning process, the influence of the noise is prominent and the dynamics are not properly captured, which ultimately leads to poor control strategies.

Similarly as in Sec. 4.2.2, the values $\{V_i^\pi\}_{i\in\mathcal{I}_\mathcal{S}}$, computed from the identified strategy, are larger, on average, than the values from the ORA policy, see Tab. 2. This result indicates that the present approach is able to uncover an efficient control strategy, here achieving a significantly lower cost than an optimal time-invariant control strategy, even in a noisy environment. The resulting control command is plotted in Fig. 11(b) and is seen to exhibit an oscillatory behavior.



(a)



(b)

Fig. 9 Mean pressure field. (a): for the NULL command. (b): for the present identified control strategy. Note that only a part of the computational domain is plotted.

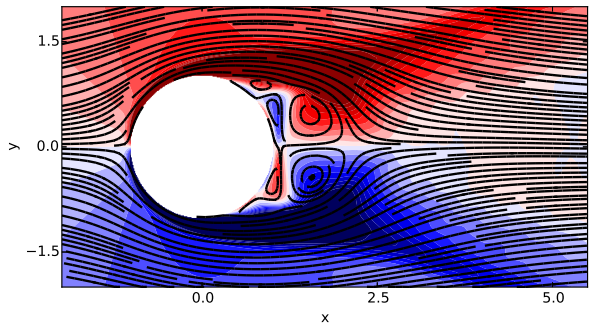
Policy	NULL	ORA	Present strategy
Average value $\langle V^\pi \rangle$	-6.8	3.0	8.0

Table 2 Average expected value $\langle V^\pi \rangle := \mathbb{E}_{i \in \mathcal{I}_\mathcal{S}} \left[V_i^\pi \right]$ associated with the NULL, ORA and present strategy policies. The learning effort for the rewards (resp. the Q-factors) was 10000.

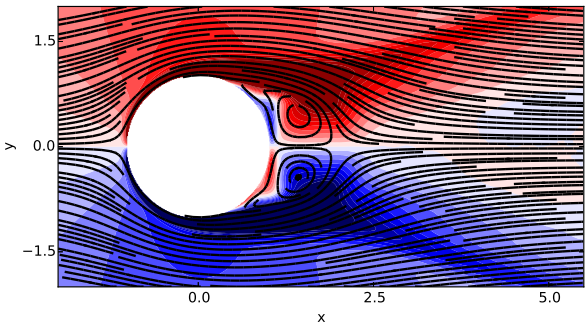
The associated drag coefficient of the cylinder flow is plotted in Fig. 11(c) for the present approach as well the NULL and the ORA strategy. Again, the identified control is seen to perform well and resembles that given by ORA.

5 Concluding remarks

This work has presented an experiment-oriented control strategy which does not require any prior knowledge of the physical system to be controlled nor significant computational resources. This strategy allows the learning of a control policy from scarce and point sensors with very limited information on the system at hand. From the sensors' streaming data, a phase space is built using hash functions as a kernel the measurements are convoluted with in real-time. This allows the derivation of a discrete, low-dimensional, space in which the dynamics of the system are approximated. Ensemble-averaged rewards associated with transitions from one discrete state to another are estimated during an online learning sequence. They

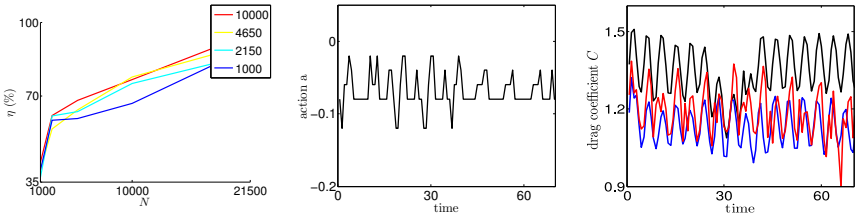


(a)



(b)

Fig. 10 Vorticity field and streamlines for the mean field. (a): for the NULL command. (b): for the present identified control strategy. Note that only a part of the computational domain is plotted. Colors encode the sign of the vorticity.



(a) (b) (c)

Fig. 11 Illustration of the control policy. (a): Performance indicator with respect to the learning effort of the transition rewards matrix R. Colors (from blue to red) encode different efforts in learning Q. (b): Actuation as a function of time, $a(t)$. (c): Time-evolution of the drag coefficient under three different control policies: zero command (NULL, black), ORA strategy (blue), and the optimal policy from the present approach (red).

are directly related to the control objective and tend to sort state transitions based on their impact on the control cost function. A reinforcement learning algorithm is then used to derive the optimal control policy, promoting transitions associated with good rewards.

The resulting method is compliant with actual configurations where instrumentation is limited and spatially-constrained. Owing to the use of kernel hash

functions and effective state-aggregation in a discrete phase space, the method runs in real-time and allows closed-loop control. Its discrete and ensemble-averaged nature also brings intrinsic robustness against perturbations in the flow.

This approach has been illustrated on two test cases. The control of a Lorenz system is achieved, by measuring only one component. To mimic a realistic scenario, the actuation is done on a different component. The method is also illustrated by the two-dimensional flow around a circular cylinder. Measurements were provided by a single wall-mounted pressure sensor and actuation was achieved by blowing or suction of fluid at the cylinder surface. The drag coefficient was significantly reduced, reaching essentially the same performance as the control policy given by the ORA strategy.

More generally, the method presented in this work is readily applicable to physical systems with a causal link between the actuators and the cost functional as evaluated from the sensors. It does not rely on a prior model and instead learns directly from observing the system under stimulation by the actuators, hence being suitable for practical configurations. An identified limitation of the proposed approach is the number of keys one needs to consider if the relevant dynamics of the system is very rich (large n_{key}) and/or a very fine control law is required (large n_a). In this situation, the number of entries of the Q matrix grows and hence possibly requires more data for learning. Approximation techniques have however been used to alleviate this limitation [33].

Current efforts concern the experimental control of the turbulent flow over an open cavity using the present approach and will be the subject of a subsequent publication. Further developments focus on the convolution kernels, an improved evaluation of suitable rewards and milder assumptions for the reinforcement learning of the control policy.

References

1. J. Gerhard, M. Pastoor, R. King, B.R. Noack, A. Dillmann, M. Morzynski, and G. Tadmor. Model-based control of vortex shedding using low-dimensional galerkin models. *AIAA J.*, 4262(2003):115–173, 2003.
2. M. Bergmann and L. Cordier. Optimal control of the cylinder wake in the laminar regime by trust-region methods and pod reduced-order models. *J. Comp. Phys.*, 227(16):7813–7840, 2008.
3. Z. Ma, S. Ahuja, and C.W. Rowley. Reduced-order models for control of fluids using the eigensystem realization algorithm. *Theo. Comp. Fluid Dyn.*, 25(1-4):233–247, 2011.
4. W.T. Joe, T. Colonius, and D.G. MacMynowski. Feedback control of vortex shedding from an inclined flat plate. *Theo. Comp. Fluid Dyn.*, 25(1-4):221–232, 2011.
5. L. Mathelin, L. Pastur, and O. Le Maître. A compressed-sensing approach for closed-loop optimal control of nonlinear systems. *Theo. Comp. Fluid Dyn.*, 26(1-4):319–337, 2012.
6. L. Cordier, B.R. Noack, G. Tissot, G. Lehnasch, J. Delville, M. Balajewicz, G. Daviller, and R.K. Niven. Identification strategies for model-based control. *Exp. Fluids*, 54(8):1–21, 2013.
7. C. Lee, J. Kim, D. Babcock, and R. Goodman. Application of neural networks to turbulence control for drag reduction. *Phys. Fluids*, 9(6):1740–1747, 1997.
8. M.A. Kegerise, R.H. Cambell, and L.N. Cattafesta. Real time feedback control of flow-induced cavity tones - part 2: Adaptive control. *J. Sound Vib.*, 307:924–940, 2007.
9. S.-C. Huang and J. Kim. Control and system identification of a separated flow. *Phys. Fluids*, 20(10):101509, 2008.
10. A. Hervé, D. Sipp, P.J. Schmid, and M Samuelides. A physics-based approach to flow control using system identification. *J. Fluid Mech.*, 702:26–58, 2012.

11. N. Gautier, J.-L. Aider, T. Duriez, B.R. Noack, M. Segond, and M.W. Abel. Closed-loop separation control using machine learning. *J. Fluid Mech.*, 770:442–457, 2015.
12. S. Brunton and B. Noack. Closed-loop turbulence control: Progress and challenges. *App. Mech. Rev.*, 67(5):050801, 2015.
13. M. Slaney and M. Casey. Locality-sensitive hashing for finding nearest neighbors [lecture notes]. *IEEE Signal Process. Mag.*, 25(2):128–131, 2008.
14. E. Kaiser, B.R. Noack, L. Cordier, A. Spohn, M. Segond, M. Abel, G. Daviller, J. Östh, S. Krajnović, and R.K. Niven. Cluster-based reduced-order modelling of a mixing layer. *J. Fluid Mech.*, 754:365–414, 9 2014.
15. P. Mandl. Estimation and control in markov chains. *Adv. App. Probab.*, pages 40–60, 1974.
16. C. Watkins and P. Dayan. Q-learning. *Mach. Learn.*, 8(3-4):279–292, 1992.
17. A. Gosavi. Target-sensitive control of markov and semi-markov processes. *Int. J. Control Autom.*, 9(5):941–951, 2011.
18. C.T. Lin and C.P. Jou. Controlling chaos by ga-based reinforcement learning neural network. *IEEE T. Neural Networ.*, 10(4):846–859, 1999.
19. S. Gadaleta and G. Dangelmayr. Optimal chaos control through reinforcement learning. *Chaos*, 9(3):775–788, 1999.
20. P. Grassberger and I. Procaccia. Measuring the strangeness of strange attractors. *Physica D*, 9:189–208, 1983.
21. F. Takens, D.A. Rand, and L.S. Young. Dynamical systems and turbulence. *Lect. Notes Math.*, 898(9):366, 1981.
22. J.L. Carter and M.N. Wegman. Universal classes of hash functions. In *Proc. 9th Ann. ACM Theor. Comp.*, pages 106–112. ACM, 1977.
23. A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Proc. 47th Ann. IEEE Found. Comp. Sci.*, pages 459–468. IEEE, 2006.
24. W. Johnson and J. Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemp. Math.*, 26:189–206, 1984.
25. E.A. Novikov. Two-particle description of turbulence, markov property, and intermittency. *Phys. Fluids*, 1(2):326–330, 1989.
26. C. Renner, J. Peinke, and R. Friedrich. Experimental indications for markov properties of small-scale turbulence. *J. Fluid Mech.*, 433:383–409, 2001.
27. R. Bellman. On the theory of dynamic programming. *P. Natl. Acad Sci. USA*, 38(8):716, 1952.
28. W. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.
29. F. Lewis and D. Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *Circuits Syst. Mag., IEEE*, 9(3):32–50, 2009.
30. E.N. Lorenz. Deterministic nonperiodic flow. *J. Atmos. Sci.*, 20(2):130–141, 1963.
31. O.P. Le Maître, R.H. Scanlan, and O.M. Knio. Estimation of the flutter derivatives of an NACA airfoil by means of Navier–Stokes simulation. *J. Fluids Struct.*, 17(1):1–28, 2003.
32. F. Lusseyran, L.R. Pastur, and C. Letellier. Dynamical analysis of an intermittency in an open cavity flow. *Phys. Fluids*, 20(11):114101, 2008.
33. A.A. Gorodetsky, S. Karaman, and Y.M. Marzouk. Efficient high-dimensional stochastic optimal motion control using tensor-train decomposition. In *Robotics: Science and Systems XI, Sapienza University of Rome, Italy, July 13-17*, 2015.