Learning Against Uncertainty in Control Engineering

Mazen Alamir

Learning Against Uncertainty in Control Engineering ⋆

Mazen Alamir [a]

[a] *Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble.*

Abstract

In this paper, some data-based control design options that can be used to accommodate for the presence of uncertainties in continuous-state engineering systems are recalled and discussed. Focus is made on reinforcement learning, stochastic model predictive control and certification via randomized optimization. Some thoughts are also shared regarding the positioning of the control community in a data and AI-dominated period for which some suggestions and risks are highlighted.

Keywords Reinforcement Learning; Stochastic Model Predictive Control; Probabilistic Certification.

1 Introduction

Feedback in engineering is all about facing uncertainties affecting dynamical systems. These uncertainties result from incomplete knowledge of the dynamics or from non modeled unpredictable exogenous factors. This is the reason why, a discussion regarding learning-based tools and ideas involving uncertainty management in control should unavoidably operate some necessary choices unless it encompasses all the history of control design. This is because any classical control-related management of uncertainties can be viewed as specific instantiation of learning from the measured quantities. Such a choice explicitly excludes many topics and focuses on others.

Let us face it, the recent burst of the keyword *learning* in the control literature is mainly due to two facts, namely: 1) The recent impressive success stories of Deep Reinforcement Learning (DRL) in the games area (GO, Chess, etc.) and since, in many fancy application areas including painting and Music! and 2) the availability of easy-to-use and efficient general purpose learning tools such as `scikit-learn` [52] and Deep Neural Networks (DNN) learner tools such as Keras [22] and GOOGLE's `tensorflow` [1].

Therefore, while RL [13] should be one single topic among many others on which the above mentioned choice has to operate, it is a fact that the *unreasonably over-optimistic* [1] expectations and widespread beliefs regarding the extent to which RL might be successful without almost no specific a priori knowledge, does alter in a dramatic way and probably for many years to come, the state-of-mind of young researchers in the control engineering community.

That is the reason why, without a (necessarily too short) balanced discussion towards moderating the expectations and scope of validity of RL (and especially model-free RL) in the specific domain of control engineering, no claim involving other options can be even heard since sentences, like the one below, can always be used to disqualify any possible realistic and real-life compatible innovative solution:

> *Why should I examine any suggestion when I am told that Reinforcement Learning solves any problem even without having the slightest knowledge on the dynamics or any a priori assumption while using exclusively ground-truth real-life measurements?*

Therefore, this paper starts by first recalling briefly what RL is about in the framework of control systems with continuous state set under uncertainties (Section 2.1). Focus is made on the computation issues in the model-based setting (Section 2.2.1) and then in the model-free setting (Section 2.2.2). Capitalizing on this recall some

⋆ This paper was not presented at any IFAC meeting. Corresponding author M. Alamir. Tel. +33476826326. Fax +33476826388. This work was supported by MIAI @ Grenoble Alpes under Grant ANR-19-P3IA-0003.

Email address: `mazen.alamir@gipsa-lab.inpg.fr` (Mazen Alamir).

[1] To the author's opinion!

recent works showing interactions between control and RL are briefly described in Section 2.3, in particular, RL-based identification of input/output linearizing state feedback (Section 2.3.1) as well as control invariance based framework for safe learning during RL (Section 2.3.2) are discussed. Section 3 explores some scalable solutions mainly involving a combination of stochastic MPC, probabilistic certification and clustering. The paper ends with a general discussion regarding some claims and practices related to data-driven solutions as well as some humble recommendation regarding the positioning of the control community in a data/AI dominated period.

2 Reinforcement learning? why not? but ...

2.1 Brief refresher on Reinforcement learning

Consider a discrete-time dynamical system with state vector $x \in \mathbb{R}^n$, control input vector $u \in \mathbb{R}^{n_u}$. In the following reminder and for the sake of homogeneity of notation throughout the whole scope of the paper, the cost minimization paradigm is used instead of the reward maximization that is commonly invoked in the RL literature. Therefore, it is assumed that a *measurable* stage cost $\ell(x,u)$ is associated to the pair (x,u) being visited by the controlled system. The result of the integration of this stage cost over some time interval is assumed to be an indicator of the quality of the system behavior over this interval (lower values are better).

The starting point in any control design approach is to consider the so called sets-of-interest $\mathbb{X}$ and $\mathbb{U}$ that contain all possible realizations of the state and the input vectors values. Choosing such sets is definitively not an easy task but this difficulty is not specific to RL and materializes in any possible framework.

In its so-called *deterministic-policy* form [23] [2], the objective of RL design is to find an optimal state-feedback *policy* $\pi : \mathbb{X} \to \mathbb{U}$ such that the following cost is minimized [23]:

$$J(\pi) := \int_{\mathbb{X}} \rho^{\pi}(x)\ell(x,\pi(x))dx \tag{1}$$

where $\rho^{\pi}(x)$ stands for the *discounted* probability density function of the state under the control strategy π, namely:

$$\rho^{\pi}(x) := \int_{\mathbb{X}} \sum_{k=1}^{\infty} \gamma^{k-1} p_0(s) p(s \to x, k, \pi) ds \tag{2}$$

which simply sums (with a discount factor γ) the probabilities of all the paths starting at some initial state s (with probability density $p_0(s)$) and passes through x after k sampling periods. The probability density of such an event under the feedback strategy π is denoted by $p(s \to x, k, \pi)$. The use of probabilities implicitly acknowledges that the underlying controlled system's dynamics is uncertain.

The ambition of RL formulations is to provide a state-feedback *policy* $\pi^{\star}$ that is optimal considering the statistics of possible realizations of the system future life-impacting components (state, uncertainties). This is precisely what is expressed through the cost function (1) and this is precisely why RL is (conceptually) viewed as the perfect answer to the presence of uncertainties in the controlled systems.

The search for such an optimal policy commonly starts by choosing some parametrized form, say π_θ, of the feedback strategy; and to look for an optimal parameter vector $\theta^{\star}$ that minimizes the corresponding cost $J(\pi_\theta)$ defined by (1). This minimization is commonly performed using gradient descent in which the gradient of the cost function w.r.t the design parameter θ is given by [23]:

$$\nabla_\theta J(\pi_\theta) = \int_{\mathbb{X}} \rho^{\pi_\theta}(x) \big[\nabla_\theta \pi_\theta(x)\big] \nabla_u Q^{\pi_\theta}(x, \pi_\theta(x)) dx \tag{3}$$

$$= \mathbb{E}\Big[\big[\nabla_\theta \pi_\theta(\cdot)\big] \nabla_u Q^{\pi_\theta}(\cdot, \pi_\theta(\cdot))\Big] \tag{4}$$

where $Q^{\pi}(x,u)$ is the so-called action value function defined by:

$$Q^{\pi}(x,u) := \ell(x,u) + \mathbb{E}\Big[\sum_{k=0}^{\infty} \gamma^k \ell(x_{k+1}, \pi_\theta(x_{k+1}))\Big] \quad \text{under } (x_0,u_0) = (x,u) \tag{5}$$

Note that the definition (5) of Q^{π} implies (by the Bellman optimality argument) that when the feedback strategy is optimal one gets the so-called *Q-learning* equation characterizing the corresponding optimal action-value map:

$$Q^{\pi^{\star}}(x,u) = \ell(x,u) + \gamma \min_{v \in \mathbb{U}} \mathbb{E}\Big[Q^{\pi^{\star}}(x^+, v)\Big] \tag{6}$$

in which the expectation (which differs from the one involved in the very definition (5) of Q^{π} itself) refers to the uncertainties on the prediction of the next state x^+. Equation (6) recalls the Stochastic Dynamic Programming formulation and suggests that the optimal action value function can also be (at least conceptually) computed by a fixed-point iteration [26,4], namely, using a current estimation of the map Q^{π}, one can derive a corrected version from the evaluation of the r.h.s of (6).

[2] The slight difference in the various RL forms is not relevant to the current discussion since the overall assessment and conclusion remain valid.

Note that having the optimal action value map $Q^\star = Q^{\pi^\star}$ enables to derive the optimal policy and the optimal value function by:

$$u^\star(x) := \arg\min_{u\in\mathbb{U}} Q^\star(x,u) \; ; \; V^*(x) = Q^\star(x,\pi^\star(x)) \quad (7)$$

2.2 The challenges of RL implementation

In this section, the challenges that are associated to the implementation of the above formulation in the presence of uncertainties are recalled in two contexts of use, namely 1) in the presence of uncertain dynamic model and 2) in the purely data-driven context, called also *model-free* RL. Note that the literature on RL involves a huge amount of possible variants that slightly differ in many technicalities. For a detailed discussion, the reader might refer to the excellent recent books [55,51]. The following necessarily over-simplified presentation simply helps conveying general messages that, presumably hold true among the different versions.

2.2.1 In the presence of an uncertain model

When one disposes of an uncertain model in the sense that x^+ can be computed (with uncertainty) from the knowledge of (x,u), an *off-line* solution can be an option.

More precisely, **for the current value of the policy parameter** θ, the expectation invoked in the r.h.s of (4) can be approximated via averaging over a cloud of realizations of the state. For each state x in this cloud, The estimation of the value of $Q(x,\pi_\theta(x))$ needs the estimation of the expectation of the integrated cost (5) which involves model-based simulations in which the sampled quantity is the realization of the uncertainties for the current sample of x. The number of samples in these two averaging processes can be determined following the recommendations of [59] depending on the targeted quality of the approximations. Recall that all the above cascaded evaluations are to be done to get the evaluation of the gradient at the current value θ during the gradient-based optimization process.

This means that even if the gradient descent converges ultimately to the right solution, this might need a number of simulations [involved in (5)] that grows exponentially in the state dimension. This is the price to pay if one would like to stick to the ideal promises of RL which is to deliver the stochastically optimal state feedback STRATEGY defined as a function of the state over the set of interest $\mathbb{X}$.

The discussion above applies to the case where the gradient approach is used. When an uncertain model is available, it is also possible to compute the expectation invoked in the r.h.s of (6) in order to implement the fixed-point iteration (since a cloud of x^+ realizations can be computed based on the uncertain model) [3] . In this case, the curse of dimensionality still applies since the cardinality of the supporting grid in the state space that is needed to enforce the equality (6) over $\mathbb{X}$ increases exponentially in the state space.

The possibility of simulating the dynamics offers an obvious advantage over model-free design since any pair of (x,u) can be visited by forcing a simulation that starts precisely at (x,u). Therefore by extensively using Graphical Processing Unit (GPU) to perform parallel simulations and by waiting sufficiently long time and provided that a discrete state setting is considered and assuming that there is no uncertainties [4] , one can achieve very good sub-optimal solutions. But for continuous state uncertain systems, this goes rapidly beyond acceptable limits should the original promises of RL be targeted. This is because not only should any region of the (x,u) continuous space be visited, but it should be visited many times with different realizations of the uncertainties in order to compute the associated expectation approximatively.

Therefore, the above discussion can be summarized by the following statement:

Fact 1

When considering the control related continuous state applications, if the initial ambitious promises of RL are to be fulfilled, namely computing a stochastically optimal **policy** over a continuous set of states, the curse of dimensionality is unavoidable. RL inherits the standard limitations of Approximate Dynamic Programming (ADP). The beauty of the gradient theorem and the universality of deep neural networks does not help overcoming this fundamental obstacle.

The combination of a faithful model, appropriate ad-hoc simplifications and choices can be very effective in specific situations. A recent illustrative example has been proposed in [36] where an actor/critic RL approach has been used to design a magnetic control of tokamak plasmas. Note however that no specific uncertainty handling is used although the resulting feedback does show robustness to non modeled dynamics as in any feedback control framework.

[3] This is sometimes referred to as Q-learning.

[4] This can scale up to several days of extensive GPU to learn a simple ATARI game.

2.2.2 Model-free setting

The presumed ability of RL schemes to derive state-feedback policies in the absence of the slightest knowledge of the underlying dynamics has a lot to do with the high expectation they raise in the minds of many decision makers, engineers and researchers. In this section, the simplest version of model-free implementation of RL is recalled before we can examine the extent to which the commonly shared statement regarding model-free RL are justified.

Note first of all that none of the possibilities invoked in the previous section applies to the model-free case since model-based simulations are no more possible meaning that neither the whole cloud of trajectories involved in (5) nor even a single step as in (6) can be obtained through model simulation. Only real-life measurement-based updating steps can be used. More precisely, the algorithm can only use the values of triplets of the form:

$$\left\{z^{(i)}, \ell(z^{(i)})\right\}_{i\in\mathcal{I}} \qquad z^{(i)} = (x^{(i)}, u^{(i)}) \tag{8}$$

that are *visited* by the real system during some data collection experiment or during the system life-time should a continuous RL adaptation be adopted.

In its simplest form, the model-free RL starts by assuming a w-parametrized structure $Q^w(\cdot,\cdot)$ of the action value map and then implements an alternate improvement of the policy π_θ and the action-value map Q^w according to the following updating rule (also referred to as the actor/critic method):

$$\delta^{(i)} = \left[\ell(z^{(i)}) + \gamma Q^{w^{(i)}}(z^{(i+1)})\right] - Q^{w^{(i)}}(z^{(i)}) \tag{9a}$$

$$w^{(i+1)} = w^{(i)} - \alpha_w \delta^{(i)} \left[\nabla_w Q^{w^{(i)}}(z^{(i)})\right] \tag{9b}$$

$$\theta^{(i+1)} = \theta^{(i)} - \alpha_\theta \nabla_\theta \pi_\theta(x^{(i)}) \nabla_u Q^{w^{(i)}}(z^{(i)}) \tag{9c}$$

where (9b) is a gradient step that intends to update w (and hence Q^w) towards the satisfaction of (6) based on the error $\delta^{(i)}$ on the Bellman equality at the current iteration i. On the other hand, the updating rule (9c) implements a gradient step in the policy parameter θ which aims to decrease the action value map, given its current estimation at iteration i.

Note however that in order to explore in a sufficiently relevant manner the space of (x, u) so that the expectation can be correctly estimated, actions u_k that are different from the ones suggested by $\pi_\theta(x_k)$ should be regularly applied. The way and the frequency at which the exploration should be done is still an open and largely unsolved problem and will probably remain as such forever. This is because only random-like variations can be applied around the current value $u = \pi_{\theta_i}(x^{(i)}) + \nu^{(i)}$ since, *In the absence of an underlying dynamical model*, there is strictly no way of knowing what is the correction $\nu^{(i)}$ to be applied in order to force the system to visit a still unvisited region of the state-action space.

Even discarding this major difficulty, one can easily admit that for a controlled system with continuous state and control, a satisfactory exploration is a major issue to achieve in a reasonable time when the state and the control dimension go above some very moderate sizes.

Another fundamental limitation stems from the fact that since only purely experimental data collection is involved, one can only encounter *the realizations of the uncertainties that are decided by the fate during the specific interval of time during which data is being collected!*. In other words, the outcome of the iterations of model-free RL depends on an unavoidably limited set of realizations of the uncertainties that might have no statistical relevance given the true set of possible realizations that the original statement of the RL was intended to explicitly account for.

Beside this structural lack of *effective* handling of the uncertainties, there is another drawback that is associated to the gradient approach used in (9a)-(9c). Indeed, not-too-young readers probably still recall these *ancient* times where the problem of local minima associated to the gradient descent methods were largely acknowledged and largely experimented. It seems however that the astonishing success [5] of Stochastic Gradient Descent (SGD) in the context of Deep Learning (DL) induced a collective forgetting of this simple fact. The recent better understanding of the behavior of SGD iterations in the context of DL [64] suggests that the convergence comes from the combination of two facts, namely: the use of *sufficiently large number* of hidden layers leading to an over-parameterization of the underlying Deep Neural Network (DNN) on one hand and the use of SGD on the other hand. Unfortunately, the use of over-parametrization in the context of model-free control system design while increasing the probability of avoiding very bad local minima obviously comes at the price of much larger required number of real-life iterations making the framework inappropriate in many control related situations if not in the majority of them.

Last but not least, one must keep in mind that the model-free approach can only be used for situations where the problem can be stated in terms that only involve the measured quantities. In many situations, there are constraints to be handled, or terms in the stage cost, that involves internal states that are not directly measured. These components of the state are generally reconstructed using model-based observers or through

[5] Quite often if not always, this success is encountered in rather non critical applications where committing error is not fatal for the underlying context. This is almost never the case in engineering world.

simulation-based identification of relationships between the measurement profile and these components [8]. This is obviously impossible in a totally model-free setting.

Therefore, the above discussion can be summarized by the following statement:

Fact 2

As far as controlled continuous state systems are concerned, model-free RL frameworks *rarely* achieve the original promises in terms of uncertainty handling over the state space of interest. At best, they are possible heuristics with fragile convergence assessment and weak statistical relevance. Nevertheless, they can be quite valuable on a specific set of contexts.

For all the above reasons, **only model-based frameworks are considered** in the remainder of this paper.

2.3 Mixing RL & control ingredients

In some recent works, the paradigm of RL is invoked to solve control specific problems or control-based accommodations are used to enable a safe RL. In this section, we build on the previous recall on RL in order to discuss representative instances of such works, namely, the RL-based design of feedback linearization [20,61] and the Design of safe-learning frameworks [24,21,32].

2.3.1 Learning feedback linearization via RL

Recall that the feedback linearization of a nonlinear dynamics $\dot{x} = f(x)+g(x)u$ amounts to find an output map $y = h(x)$ and a state feedback $\pi_\theta(x, v)$ such that the following equality (11) holds which transforms the nonlinear system into a chain of integrators controlled by v [6]:

$$y^{(\nu)} = L_f^\nu h(x) + L_g L_f^{\nu-1} h(x)\pi_\theta(x, v) \quad (10)$$
$$= W_\theta(x, v) = v \quad (11)$$

where $y^{(\nu)}$ denote the ν-derivative of y while v is a feedforward term.

Note that equation (10) holds provided that the relative degree [7] associated to the output y is equal to ν [35]. This suggests to use the following definition of the stage cost as a quality indicator for θ to define a RL framework:

$$\ell(x, v, \theta) := \|y^{(\nu)} - v\|_2^2 \quad (12)$$

[6] The notation $L_f h$ stands for $\frac{\partial h}{\partial x} f$, $L_f^2 h = L_f(L_f h)$, etc.

[7] The relative degree associated to an output is the lowest order of derivation that makes u appear explicitly in the r.h.s of the corresponding higher derivative's expression.

which is assumed to be a measured quantity in the related works [20,61]. Since model-free RL requires *only* the measurement of the stage cost ℓ (and the state vector x! used in $\pi_\theta(x, v)$), the associated schemes can be used in this context to progressively *discover* the linearizing state strategy π_θ from real-life experiments.

Similar recent ideas towards learning linearizing feedback have been proposed in [58] to achieve event-triggered learning based on Gaussian process modeling.

It is worth underlying that several, *quite questionable*, assumptions are needed for the approach to apply, namely, 1) the existence and the explicit knowledge of a measurable output map $y = h(x)$ for which the *relative degree* ν, *assumed to be known, is invariant over all possible realizations of the unknown dynamics* and 2) the possibility to measure the whole state as well as the high derivative of the output $y^{(\nu)}$ which might be unrealistic for relative order higher than or equal to 2 because of the unavoidable measurement noise. Nevertheless, the scheme might be of some help in some very specific applications.

2.3.2 Control-based solutions for Safe learning

While in section 2.2.2, attention is focused on the computational issue associated to the curse of dimensionality in RL and the difficult task of exploring, without a supporting model, the action-state space, a crucial problem was left aside, namely the *safety* of the controlled system during the learning exploratory phase.

Without claiming to solve the curse of dimensionality nor the exploration issue (Section 2.2), a series of works [27,2,24] attempted nevertheless to address the safety issue for a rather restricted class of small size dynamical systems of the form:

$$\dot{x} = f(x, u, d(x)) \quad (13)$$

where f is supposed to be known while d is an unknown map for which a bounding set $\hat{\mathcal{D}}(x)$ is supposed to be known such that $d(x) \in \hat{\mathcal{D}}(x)$. A so-called safety set is supposed to be defined by $\mathcal{S} := \{x : g(x) \geq 0\}$ which is supposed to be a *robust control invariant set* under some state feedback strategy $\kappa^\star(x)$ to be determined. This means that under $\kappa^\star(x)$, any trajectory that starts in $\mathcal{S}$ remains in $\mathcal{S}$. Note that the main difficulty lies in the computation of the strategy $\kappa^\star$ while the map $d(\cdot)$ is not known. This is done by solving, in the unknown $V(\cdot, \cdot)$, the Hamilton-Jacobi-Isaac equation [15] which only involves the presumably known bounding set $\hat{\mathcal{D}}(x)$:

$$0 = \min\Big\{ g(x) - V(x,t), \frac{\partial V}{\partial t}(x,t) + \max_{u\in\mathbb{U}} \min_{d\in\hat{\mathcal{D}}(x)} \frac{\partial V}{\partial x} f(x,u,d) \Big\} \quad (14)$$

with the boundary condition $V(x,T) = g(x)$. Once such a map V is computed, the strategy $\kappa^\star$ is derived by:

$$\kappa^\star(x) := \arg\max_{u\in\mathbb{U}} \min_{d\in\hat{\mathcal{D}}(x)} \frac{\partial V}{\partial x} f(x,u,d) \qquad (15)$$

having this *safe feedback strategy* $\kappa^\star$, a cautious exploration can be implemented during which the worst consequence of applying the RL suggested control $u = \pi_{\theta^{(i)}}(x) + \nu^{(i)}$ (Section 2.2.2 page 4) is evaluated in terms of safety and if violation is possible, the safe strategy $\kappa^\star(x^{(i)})$ is applied instead. Probabilistic evaluation (see Section 3.1.2) can also be used instead of the worst-case evaluation in order to avoid over pessimistic design that might lead to reduced exploration performance. In this case, the safe strategy is used only if the constraint violation probability goes beyond some threshold.

The framework is assessed experimentally in [24] to address the problem of safe exploration of RL strategy applied to the problem of the scalar vertical displacement of a quadrotor drone. In this example, the uncertain vertical dynamics (13) takes the following simple form:

$$\dot{x}_1 = x_2 \quad ; \quad \dot{x}_2 = k_T u + g + k_0 + d(x). \qquad (16)$$

where x_1 stands for the altitude z.

It is worth taking few minutes to examine this example since it is iconic of the impact of RL on our research community. Indeed, Given that the objective is to regulate the variable $x_1 = z$ governed by (16) in spite of the absence of knowledge of $d(x)$ for which an upper bound $\hat{\mathcal{D}}(x)$ is known, an *old fashioned* control designer would simply have used the following simple law (assuming, without loss of generality, that $u \in [-\bar{u}, +\bar{u}]$ and $k_T > 0$:

$$u = \bar{u} \cdot \tanh\Big(-\beta\Big[\dot{z} + \lambda_S(\dot{z} - \lambda(z_d - z))\Big]\Big) \qquad (17)$$

since this would steer the system to the manifold $S(z) = \dot{z} - \lambda(z_d - z) = 0$ which achieves the regulation task without the need for any learning and a fortiori any safe learning concern. Another more general scheme with provable convergence in the absence of almost no knowledge on the system's model has been recently proposed [50] to address a wider class of problems that includes (16) as a particular instance.

This example is iconic of a general attitude that can be stated as follows:

> *In some recent works, the already available and purely control-related solutions are too easily forgotten when it comes to contributing in any possible way to the RL buzzword induced euphoria.*

An example of deep understanding via classical control concepts and tools of the achievable performance via adaptation and high gain control can be found in the recent excellent survey [30].

Other recent works focused on control-oriented solutions to the safe learning problem based on the use of Barrier functions [21,33,60], robust MPC [63] or via projection on safe sets [29] are worth examining for interested readers.

2.4 Discussion

Let us take a step back to look at the big picture! Recall that in the nineties, the nonlinear control design via analytic Lyapunov methods [40] was the dominant option. The emergent Nonlinear Model Predictive Control (NMPC) [46] was sometimes even denied the qualification of state feedback by some nonlinear systems theorists [8] because of its implicit nature (no explicit expression of the feedback nor of the associated optimal cost function). The difficult, if not impossible, derivation of Lyapunov-based solutions to general nonlinear systems incited [34] (even in 2013!) to classify systems for which, modeling involving high nonlinearities is necessary, as legitimate candidates for data-driven control design.

The success of NMPC and the unavoidable constraints handling task achieved convincing our community that NMPC is probably not so bad an option. However, the desire to have explicit representations that can be computed once for all resisted in the linear case leading to the design of Explicit Linear MPC computation tools [57]. Several years were necessary to acknowledge the non scalability of this option and the comparable complexity to on-line computation even in linear low-dimensional case [17].

Having this recall in mind, it is hard not to see in the RL a new avatar of this buried desire to be able to compute the control as a strategy (pre-computed function of the state) that arms the designer against the uncertainties in an explicit, guaranteed, pre-computed and on the top of it, optimal manner.

The previous section suggests that this option is strongly questionable, at least in its general scope claim. The remainder of this paper is dedicated to model-based learning options that might appear to be *less ambitious* but which are, to the author's opinion, more appropriate for real-life engineering problems.

[8] Including some members of the author's PhD examination committee in 1995!.

3 Scalable model-based learning for control design and certification under uncertainties

The frameworks discussed in this section follow the conclusion of the discussion of Section 2.4. The ambition of designing a STOCHASTIC OPTIMAL STRATEGY that holds over a whole subset of the state space is abandoned in favor of one of the following three alternatives:

(1) Either by dropping the search for a strategy in favor of the real-time computation of point-wise (given the current state) on-line solution of a stochastic optimization problem. This option repetitively looks for the best sequence of actions (in an approximate stochastic sense) given the current state and applies the first action. At the next instant the process is repeated leading to the so-called stochastic MPC (Section 3.2.1).

(2) Or by dropping the optimality induced characterization that lies under the stochastic dynamic programming formulation (Q-learning) in favor of more pragmatic and scalable output feedback oriented heuristics (Section 3.2.2).

(3) Or by explicitly adopting a parameterized suboptimal solution where a state feedback is designed in a problem-dependent step while leaving some few parameters of the solution to be tuned via randomized optimization. This solution keeps the ambition of deriving a feedback strategy over some subset of the state space but intentionally drops the optimality requirement in order to get a tractable design procedure (Section 3.2.3).

From the above presentation, it comes out that two frameworks need to be briefly recalled before the associated solutions can be discussed, namely, Stochastic Model Predictive Control (SMPC) and stochastic certification via randomized optimization. This is the object of the next section.

3.1 Recalls

Hereafter, only brief recalls are proposed for the sake of completeness, interested readers are invited to consult the proposed reference for a detailed exposition.

3.1.1 Stochastic Model Predictive Control

Consider the class of systems governed by:

$$x^+ = f(x, u, w) \tag{18}$$

where x, u and w stand for the state, control input and uncertainties/disturbance vectors respectively. The map f is supposed to be known. All the uncertainties are gathered in w. Stochastic Model Predictive Control (SMPC) amounts to compute an implicit feedback control based on the repetitive solution of the following optimization problem [9], expressed at instant k where the state of the system is x_k:

$$\mathcal{P}(x_k, \eta): \quad \mathbf{u}^\star(x_k) \leftarrow \min_{\mathbf{u}\in\mathbb{U}^N} \mathbb{E}\Big[J(\mathbf{u} \mid x_k, \cdot)\Big] \tag{19}$$

$$\text{under } \Pr\Big[\mathbf{g}(\mathbf{u}, x_k, \cdot) \leq 0\Big] \geq 1 - \eta \tag{20}$$

where the expectation in (19) and the probability in (20) refer to the realization of the uncertainties/disturbance vector profile $\boldsymbol{w}$. The condensed expression $\mathbf{g}(\mathbf{u}, x_k, \mathbf{w}) \leq 0$ refers to the vector of constraints to be satisfied. This might gather stage constraints over the prediction horizon as well as terminal constraints at the end of the prediction horizon. This is the reason why the expression involves the control and the uncertainty profiles $\mathbf{u}$ and $\mathbf{w}$. The prediction horizon length is denoted by N while $\mathbb{U}$ stands for the set of admissible control values.

Note that SMPC can also be formulated as the problem of finding a feedback strategy rather than a control profile (see [47] for more details). We stick to the latter formulation in accordance with the previous discussion. Note finally that the parameter η in (19)-(20) introduces constraint relaxation as a probability of constraint violation is allowed provided that it is lower than η.

Denoting by $\mathbf{u}^\star(x_k) := (u_0^\star, \dots u_{N-1}^\star) \in \mathbb{U}^N$ a solution to $\mathcal{P}(x_k, \eta)$, the applied feedback is given by $u_k = u_0^\star$ in accordance with the receding horizon principle. This control is applied during the time interval $(k, k+1)$. At the next sampling instant, a new optimization problem $\mathcal{P}(x_{k+1}, \eta)$ is defined and solved, the first action in the optimal sequence is applied over $(k+1, k+2)$ and so on.

Note that the concrete handling of the probability term in (20) over high dimensional uncertainty vector is not straightforward. The probabilistic certification framework discussed in the next section gives appropriate and concrete tools to manage this issue.

3.1.2 Probabilistic certification

Probabilistic certification paradigm addresses the problem of relaxing an original optimization problem including a robust constraint satisfaction requirement involving a decision variable $\theta \in \Theta \subset \mathbb{R}^{n_\theta}$ and an uncertainty vector p of the form:

$$\min_{\theta\in\Theta} J(\theta) \quad \text{under} \quad (\forall p) \quad I(\theta, p) = 0 \tag{21}$$

[9] There are several alternatives regarding the definition of SMPC and the way the constraints related concerns have to be stated.

where $I(\theta, p)$ is defined as follows:

$$I(\theta, p) := \begin{cases} 0 \text{ if specification are satisfied} \\ 1 \text{ otherwise} \end{cases} \tag{22}$$

and where a probability measure $\mathcal{P}$ is associated to the uncertainty vector p that is assumed to belong to some admissible set $\mathbb{P}$.

The randomized method replaces the original hard problem (21) by the following relaxed problem:

$$\min_{\theta \in \Theta} J(\theta) \quad \text{under} \quad \Pr\{I(\theta, p) = 0\} \geq 1 - \eta \tag{23}$$

where $\Pr\{I(\theta, p) = 1\}$ represents the probability of the *event* $I(\theta, p) = 1$ (violation of the requirement) given some statistics of realization of the uncertainty vector p.

Now since the computation of the probability term is a rather involved and expensive task, the randomized method [11,12] simplifies (23) by replacing the probability by the mean value over N_s drawn independent identically distributed (i.i.d) samples of p in $\mathbb{P}$, namely the new optimization problem becomes:

$$\min_{\theta \in \Theta} J(\theta) \quad \text{under} \quad \sum_{\ell=1}^{N_s} I(\theta, p^{(\ell)}) \leq m \tag{24}$$

which simply replaces the constraint on the probability by a different constraint stating that the mean value of $I(\theta, p^{(\ell)})$ over N_s random samples to be lower than m/N_s, or to state it differently that at most m between the total number N_s of samples lead to the violation of the specification. It comes therefore that, for any given admissible number of failures $m \in \mathbb{N}$, N_s must be such that $\dfrac{m}{N_s} \leq \eta$ which is obviously only a necessary condition. This is because N_s must also be sufficiently large so that the fulfillment of (24) implies that the condition (23) on the probability is satisfied with a probability greater than $1 - \delta$ with a pre-specified small value δ. That is the reason why the minimum value of N_s that makes this implication true involves both the precision specified by η and the confidence level specified by δ.

In [11,12], several expressions for the value of N_s are given under different assumptions. An example of upper bounds on N_s is given below in the case where Θ is a discrete set of cardinality n_Θ. In this case, the following proposition holds [12]:

Proposition 3.1 *Let $m \in \mathbb{N}$ be any integer. Let $\delta \in (0, 1)$ be a targeted confidence parameter and $\eta \in (0, 1)$ be a targeted precision parameter. Assuma a design parameter that belong to a discrete set Θ of cardinality n_Θ.*

n_Θ	$\eta = 0.1$	$\eta = 0.05$	$\eta = 0.01$	$\eta = 0.001$
5	154	308	1536	15354
10	163	326	1628	16280
100	193	386	1930	19299
10000	252	503	2515	25148

Table 1
Evolution of the sample size N_s as a function of the precision parameter η and the cardinality n_Θ of the design parameter set Θ. A confidence parameter $\delta = 10^{-3}$ is used while the number of failures $m = 1$ is used in (25).

Take N_s satisfying:

$$N_s \geq \frac{1}{\eta}\left(m + \ln(\frac{n_\Theta}{\delta}) + \left(2m\ln(\frac{n_\Theta}{\delta})\right)^{1/2}\right) \tag{25}$$

then any solution θ to (24) in which the $\{p^{(\ell)}\}_{\ell=1}^{N_s}$ are randomly i.i.d drawn using the probability measure $\mathcal{P}$ satisfies the constraint in (23) with a probability $\geq 1 - \delta$.

A remarkable property of the expression (25) enabling the computation of N_s is that it is totally independent of the the dimension of the vector of parameters p. This is of a tremendous importance in the context of uncertain models involving high number of uncertain parameters. Another interesting feature of Proposition 3.1 is that the confidence parameter δ appears through logarithmic terms which means that one can seek highly confident assertions without dramatic increase in the number of samples. Table 1 shows examples of lower bounds on the number N_s of required scenarios to achieve the certification for different values of the pair (n_Θ, η) when the high confidence parameter value $\delta = 10^{-3}$ is used.

3.2 Approximate SMPC schemes

The SMPC topic deserves a survey on its own [47,48] and frequent updating is necessary because of the rapid undergoing development. Some instances are shown here motivated by the discussion above. The basic difficulty in solving (19)-(20) stems from the expectation and the probability that are involved in the formulation. Indeed, a precise approximation of these quantities needs a high number of samples of the uncertainty realizations to be associated to the current state x_k in order to formulate a faithful approximation of the problem to be solved in real-time. The approaches revisited hereafter proposes different ways of addressing this issue in a non yet totally satisfactory manner.

3.2.1 Uncertainties clustering-based solutions

One of the SMPC that is widely used is related to the so-called multi-stage scenario-tree decomposition (see

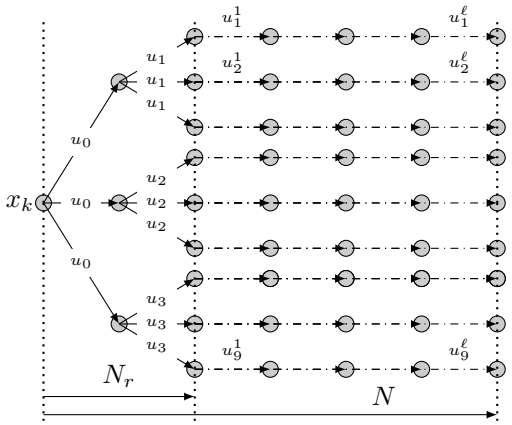

Fig. 1. Illustration of the tree of scenarios for $N = 6$, $N_r = 2$, $\ell = N - N_r = 4$ and $q = 3$.

[41,45] and the references therein). In this approach, the basic assumption is that the value $w \in \mathbb{R}^{n_w}$ of the uncertainty/disturbance vector at each sampling instant lies in a discrete set of moderate cardinality q, say $\mathbb{W} := \{w^{(1)}, \ldots, w^{(q)}\}$ where $w^{(i)} \in \mathbb{R}^{n_w}$. Consequently, the uncertainty profile over a prediction horizon of length N takes values in a set of cardinality q^N. The resulting exponential explosion is limited by the assumption according to which the value pf w becomes constant after a limited number of steps N_r (defining the so-called robust horizon) as shown in Figure 1. This reduces the number of scenarios to $n_s := q^{N_r}$ instead of q^N. The resulting tree is shown in Figure 1 for $N = 6$, $N_r = 2$ and $q = 3$ where a control profile is associated to each uncertainties profile with some constraints that are discussed below. It is also assumed that the available statistics of the uncertainties enables to associate a probability ω_i to each scenario i among the $n_s = q^{N_r}$ scenarios under consideration so that a weighted cost function can be defined at instant k by:

$$\sum_{i=1}^{n_s} \omega_i J(\mathbf{u}^{(i)} \,|x_k, \mathbf{w}^{(i)}) \tag{26}$$

where $\{\mathbf{w}^{(i)}\}_{i=1}^{n_s}$ is the set of scenarios while $\mathbf{u}^{(i)}$ is the control profile associated to the i-th scenario. Note however that these control profiles are constrained by the fact that all the control inputs starting from the same state (root) have to be equal. This constraint is referred to as the *non anticipation* constraint. This is clearly shown on Figure 1 through the notation u_0, u_1, u_2 and u_3. Note that the expression (26) is supposed to be an approximation of the expected cost as defined by (19). Similar weighted constraints can be similarly defined in order to replace (20).

Obviously, the assumption regarding the finite number q^{N_r} of scenarios might be viewed as a brute force assumption. As a matter of fact, the discrete set of q possible values of w can be viewed as a result of a clustering operation and the corresponding probabilities $\{\omega_i\}_{i=1}^{n_s}$ can be obtained as a by-product of the same clustering step using the ratios of population sizes of the clusters to the total number of randomly drawn samples.

Clustering algorithms (K-Means, Mean-shift, DBSCAN, to cite but few available algorithms in the `scikit-learn` library [52]) generally perform an *unsupervised* clustering in the sense that they consider only internal relationships between the elements of the set of samples $\mathcal{W} := \{\mathbf{w}^{[1]}, \ldots, \mathbf{w}^{[s]}\}$ where $s \gg n_s$ to be used in the clustering operation (finding the centers of the clusters and their associated weights):

$$\mathcal{W} \xrightarrow[\text{clustering}]{\text{unsupervised}} \mathbb{W}^N \tag{27}$$

In a recent work [5], it has been suggested that a supervised clustering can be achieved by inducing the clustering of the elements of $\mathcal{W}$ from the *unsuprvised clustering* of the optimal solutions $\mathcal{U}^\star := \{\mathbf{u}^{[i]\star}\}_{i=1}^{s}$ of the deterministic optimal control problems associated to the elements $\mathbf{w}^{[i]}$ of the set $\mathcal{W}$:

$$\mathcal{W} \xrightarrow[\text{clustering}]{\text{unsupervised}} \mathcal{U}^\star \xrightarrow[\text{clustering}]{\text{induced}} \mathbb{W}^N \tag{28}$$

The rationale behind this solution is that uncertainty/disturbance profiles should be considered as *similar* if they induce the *similar* optimal solutions even if the disturbances are not close in their own space.

Beside the supervised clustering feature, it is proposed in [5] to continuously update a FIFO buffer of clusters so that the clustering can be made state-dependent.

The heuristics summarized in this section have been applied to relevant problems such as the problem of uncertainty aware type-1 diabetes [25] as well as the combined therapy of cancer [5] to cite but few examples. Note that both frameworks involve at some stage, the solution of a deterministic nonlinear optimal control problem that can be achieved using excellent and freely available solvers that are gathered within unified optimization framework such as `Casadi` [14].

3.2.2 Learning output-feedback stochastic NMPC from clouds of deterministic solutions

The online computation of the optimization problems invoked in Section 3.2.1 might be incompatible with real-time implementation for a class of systems showing *fast* dynamics requiring small sampling periods. An intuitive option is to then simulate off-line a high number of *closed-loop* scenarios and then to learn the resulting feedback using nonlinear modeling structures such as DNN, Random Forest or any other regression model that are available in any Machine Learning package. Once a candidate fitted feedback is available, its performance (in-

cluding constraints satisfaction assessment) can be certified using the probabilistic certification framework recalled in Section 3.1.2.

Such a solution is proposed in [38] in order to learn the robust feedback associated to the robust multi-stage tree scenarios approach recalled in Section 3.2.1 and to certify the resulting approximate feedback. More precisely, during the simulated closed-loop scenarios, NMPC feedback based on the solution of the DETERMINISTIC NLP defined by (26) and the associated constraint is used. Once a sufficient learning data is obtained, a DNN is used to capture the structure of the feedback. During the closed-loop simulation, an extended Kalman filter is used to get an output feedback so that the certification implicitly includes the state estimation error. The framework is validated using the quite challenging example of controlling a wind kite under a restrictive altitude constraint. A tightening parameter η (back off on the lower bound of the altitude) is used that can be viewed as a hyper parameter for the control design [10] which is optimized via probabilistic certification.

The framework of [38,39] heavily relies on the observability assumption which might be quite questionable in the presence of high uncertainties affecting the model's parameters. This being said, it should be emphasized that the learning literature rarely addresses the need for an output feedback and the availability of the state is unfortunately a too frequently used assumption in the learning-based control literature.

A different alternative has been recently proposed [6] that is based on the off-line solution of deterministic problems to build a learning data for the identification of an uncertainty-aware dynamic output feedback.

More precisely, a cloud of pairs $\{z = (x_0, \boldsymbol{w})\}_{z\in\mathbb{Z}}$ is considered in which each element z is a pair of initial state x_0 and an uncertainty vector profile $\boldsymbol{w}$. Obviously, each z defines a deterministic optimal control problem $\mathcal{P}(x_0, \boldsymbol{w})$ in which both x_0 and $\boldsymbol{w}$ are supposed to be known. Figure 2 shows how the optimal solution of the deterministic optimal control problem $\mathcal{P}(z)$ can be used to construct a learning data $\mathcal{D}(z)$ associated to z of the form:

$$\mathcal{D}(z) := \left\{ \boldsymbol{y}^{\star,(-)}_{M+j}(z), \boldsymbol{u}^{\star}_{M+j}(z) \right\}_{j=0}^{m-1} \tag{29}$$

in which $\boldsymbol{u}^{\star}(z) := (\boldsymbol{u}^{\star}_0(z), \ldots, \boldsymbol{u}^{\star}_{N-1}(z))$ is the optimal control profile (should x_0 and $\boldsymbol{w}$ be perfectly known) while $\boldsymbol{y}^{\star,(-)}_{M+j} := (\boldsymbol{y}^{\star}_j, \ldots, \boldsymbol{y}^{\star}_{M+j})$ stands for the sequence of measurement (including the input) gathered during the interval $[j, M+j]$ (see Figure 2).

[10] Namely, η is a component of the decision variable θ in the certification framework as recalled in Section 3.1.2

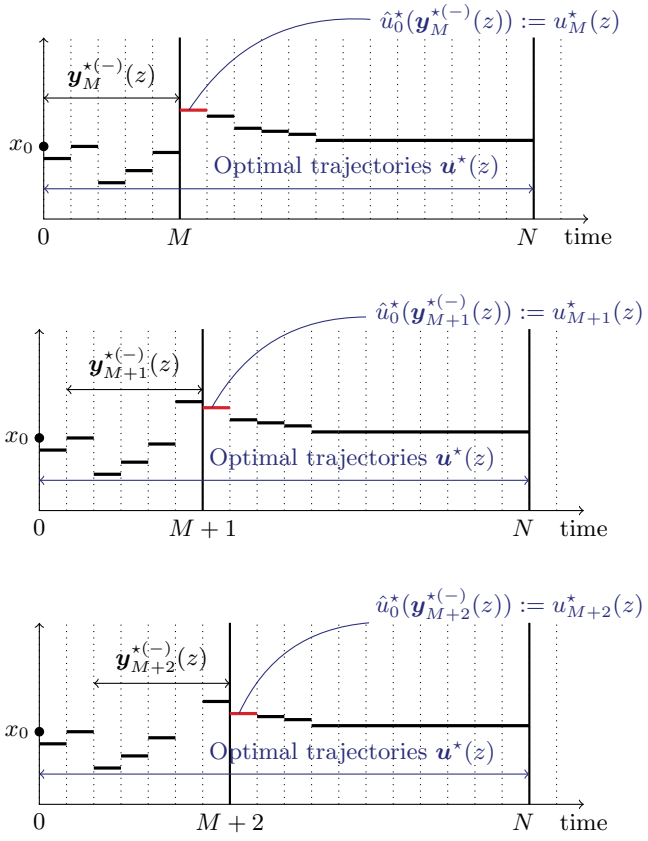

Fig. 2. Construction of the learning data $\mathcal{D}(z)$: by going forward in a moving window along the optimal trajectory computed for a single pair $z = (x_0, w)$ it is possible to generate a high number of different samples of pair $(\boldsymbol{y}^{(-)}, u)$ that can be used in the construction of the learning data for the identification of uncertainty-aware output feedback.

The rationale behind this choice it that if $N = \infty$, by the Bellman principle, the control input $\boldsymbol{u}^{\star}_{M+j}(z)$ would be the first action in the exact optimal solution associated to the pair $(\boldsymbol{x}^{\star}_{M+j}(z), \boldsymbol{w})$ that would be reconstructed using the previous the output profile $\boldsymbol{y}^{\star,(-)}_{M+j}$ should the extended observability condition holds true. For finite prediction horizon N this corresponds to an approximation. This is what is meant by the notation $\hat{\boldsymbol{u}}^{\star}_0(\boldsymbol{y}^{\star,(-)}_{M+j}(z)) := \boldsymbol{u}^{\star}_M(z) \approx \boldsymbol{u}^{\star}_0(\boldsymbol{y}^{\star,(-)}_{M+j}(z))$ in Figure 2. By so doing, a single NLP solution enables to gather m instances in the learning data that aims to identify the approximate optimal control as a function of the previously measured output. Repeating this process for a cloud of $n_z = \mathrm{card}(\mathbb{Z})$ samples of z, a learning data of size $m \cdot n_z$ can be built from the solution of n_z standard deterministic NLP solutions:

$$\mathrm{card}\left(\bigcup_{z\in\mathbb{Z}} D(z) \right) = m n_z \tag{30}$$

Here again, once an output feedback map is fitted, its

performance can be certified following the same guidelines invoked previously since high number of closed-loop simulations can be efficiently conducted in an optimization-free way. In a further stage, it is possible to re-inject the values of z for which the performances show inadequate in the learning process iteratively in order to improve the quality of the fitted map.

The use of the measurement profile $\boldsymbol{y}^{(-)}$ as a feature vector is obviously inspired by the extended observability idea (joint observation of the state and the uncertain parameter vector) but the observability is not a constructive condition for the heuristic implementation. In the absence of strict observability, the fitting step would find the *best* compromise given the dispersion of the indistinguishable features in the learning data.

In [6], the above framework has been used to derive an uncertainty-aware output feedback applied to address an economic MPC design. Comparisons have been conducted against a perfect-knowledge ideal NMPC on one hand and a nominal NMPC. It comes out that, despite large uncertainties ranging from 20% to 180% of the nominal values, the proposed design recovers 78% of the advantage of having full knowledge of the parameters values compared to the nominal one.

3.2.3 Suboptimal certified strategies via problem-dependent parameterization

As already mentioned in Section 2.3.2, quite often, the availability of computation power and user-friendly software that can address complex and ambitious formulation makes it easy to forget simple and efficient solutions that might appropriately address the problem in a simple and elegant way. Indeed, in many real-life situations, one disposes of an uncertain model of the form:

$$\dot{\xi} = f_1(\xi, u, \varphi(\eta)) \tag{31}$$
$$\dot{\eta} = f_2(\xi, \eta, w) \tag{32}$$
$$y = h(\xi, u) \tag{33}$$

where f_1 is a known function involving uncertainty through an uncertain map φ that depends on a variable that obeys an uncertain dynamics f_2 that also depends on some exogenous signal w. ξ and η stand for two sub-state vectors. Assume that

$\mathbf{H}_1$) the control objective and constraints can be expressed in terms of (ξ, u) only, namely:

$$J(\boldsymbol{\xi}, \boldsymbol{u}) \quad , \quad g(\boldsymbol{\xi}, \boldsymbol{u}) \leq 0 \tag{34}$$

where $\boldsymbol{\xi}$, $\boldsymbol{u}$ stand for the trajectories of ξ and u over some prediction horizon.

$\mathbf{H}_2$) the system defined by:

$$\dot{\xi} = f_1(\xi, u, z) \tag{35}$$
$$\dot{z} = 0 \tag{36}$$

is observable;

$\mathbf{H}_3$) The dynamics (32) is unconditionally bounded.

Under these conditions, it is possible to generically design a Moving-Horizon-Estimator (MHE) [54] of the state (ξ, z) of the dynamical system (35)-(36) which provides an estimation of the unknown term [11]:

$$\begin{pmatrix} \hat{\xi} \\ \hat{\varphi} \end{pmatrix} = \mathrm{MHO}(\boldsymbol{y}^{(-)}) \tag{37}$$

The estimation can be used to define a parameterized OUTPUT FEEDBACK of the form

$$k_\theta(\hat{\xi}, \hat{\varphi}) = k_\theta \circ \mathrm{MHO}(\boldsymbol{y}^{(-1)})$$

By so doing it is possible to simulate closed-loop scenarios for different value of the unknown dynamics parameters as well as the initial guess of the observer in order to perform randomized optimal choice of the control parameter θ following the guidelines of section 3.1.2 where p is defined by:

$$p := (\xi(0), \eta(0), \hat{\xi}(0), \hat{\eta}(0), \boldsymbol{w}) \tag{38}$$

which clearly enables to write J and g invoked in (34) as function of (θ, p). This framework is rigorously detailed in [38] but appeared previously, totally or partially, in many application oriented works such as [10,3,9] related respectively to automotive control, cancer treatment and propofol-based control of BIS during anesthesia. It is shown how in each specific case, a problem-dependent design of the parameterized control law $k_\theta()$ can be derived enabling a faster simulation of the high number of closed-loop scenarios required for the certification and probabilistic optimization task.

3.3 Miscellaneous: Non covered related topics

As mentioned in the introduction, *learning for control* is so vast a topic that only a tiny selected set of items has been shortly discussed. Among the interesting topics that are not mentioned in this paper, it is worth mentioning those related to the construction of non parametric models (kernel-based [42], kinky inference-based [18,43] among many others) and the approximation of associated bounds (maximum modeling error, Lypschitz constants to cite but few ones) that allow the use of standard control design or proof arguments that require the

[11] The input measurement is included in the y vector.

knowledge of such bounds. Recent works related to the propagation of uncertainties [65], anticipating the learning process [19], joint extended state/parameters estimation error certification [7], combining system identification and reinforcement learning [44] or MPC-induced parametrization of reinforcement learning [28] are worth examining and might be very promising topics in the near future.

Regarding the computational aspects, the use of parallel computation involving GPU/FPGA [37,31,62,53] which play key role in deep learning will certainly play an even increasingly crucial role in all design and certification methods that involve high number of simulations that can be conducted in parallel (only some of them have been discussed above).

4 Discussion

In this last section, some key issues are discussed which are worth examining carefully when it comes to considering the coupling between learning and control.

4.1 Is appropriate modeling really expensive?

Too many papers advocating for massive use of data-driven solutions start by asserting that *modeling is an expensive step that is worth avoiding*. This is probably true when high fidelity models are thought of, but it is not so accurate if one seeks parameterized models that can be structurally sound in so far as they correspond to a family of behaviors that contains the true one for a combination (although unknown) of parameter values. These models can then be used inside the many uncertain model-based solutions which acknowledge and accommodate for the presence of uncertain parameters as it has been shown in the previous sections. After all, no engineering system falls from the sky, somebody has designed it following our solid understanding of the underlying governing laws.

In order to assess the previous statement, it is worth recalling that there are several domain-specific and available libraries that help modeling real-life systems in a modular and drag-and-drop way: A non exhaustive list of such libraries would include: the MATLAB SIMSCAPE multi-discipline library [12] (electrical, multi-body, fluids, etc), the unified physical system modeling Modelica [49], the SIMULINK SIMCRYO library [16] dedicated to the modeling of cryogenic refrigerators, the PSIM simulator [13] dedicated to microgrids modeling to cite but few examples among so many.

[12] https://fr.mathworks.com/products/simscape.html

[13] https://psim.powersimtech.com/webinar-microgrid-design-and-simulation

With the development of parametric uncertainty aware solutions such as the ones recalled in the present paper, it is possible that further specific development in control-oriented structural modeling might be the appropriate direction to undertake which can reveal consistent in no more than few years. Such solid representation of the processes is currently highly appreciated by the industrial partners of our technologies and algorithms. Moreover, knowledge-based representations are already favorably welcomed as a step towards explainability that is systematically opposed to purely data-driven black-box approaches.

It is quite surprising to witness the emergence of increasingly wide spreading beliefs according to which discarding our first principles-based understanding of the physical world might be an efficient option. A nice discussion regarding this dangerous drift is provided in [56].

4.2 Heuristic vs provable settings

The emergence of data-driven solutions and tools obviously questions the necessary positioning of the control community in the landscape of data-based solutions providers. One of the options that is frequently suggested claims that the control community is very good when it comes to providing provable statements in the form of guaranteed performance/stability and constraints satisfaction.

While this is probably true, the implication of this positioning might be quite risky. This is in particular true when confusion is made between two fundamentally different achievements:

- Providing ad-hoc assumption-based sufficient conditions for a statement to hold;
- Providing a set of checkable, verifiable and real-life compatible [14] conditions.

Sticking to the first item is obviously much easier and very much corresponds to a rather common practice in the control community. A danger that lies underneath is to prevent clever and efficient heuristics (including control-culture's inspired ones) that are not proof-friendly to emerge in our publication supports leaving the field of the only solutions that matter in real-life to computer science originated contributions. Another probably more harmful danger is to forget the uncheckable quality of the assumptions and rely on the papers titles to consider that the problem is solved while it is still totally open form a practical and real-life point of view.

[14] that might hold at least for one realistic existing real-life system!

4.3 Sharing codes and benchmarks

While a mathematical proof can be carefully read and checked, data-related results highly depend on many - sometimes hidden- steps (Randomness in data generation, splitting the data into learning and validation [15], tuning of the model's hyper-parameters, features selections, normalization, choice of the cloud of possibilities, etc). This suggests that a good practice is to share the whole code and data in an accessible form to the readers of accepted publications.

While such practices are commonly encouraged, they should probably become mandatory, at least when the contributions meet some conditions in terms of data-dependent content.

Similarly, as mentioned in the paper, very often, quite involved and complex frameworks are assessed using *toy ad-hoc* examples that might be handled using standard control tools and methods. The arguments implicitly used is that, *these are simply illustrative examples but the proposed framework does scale to tackle real-life problems*. The suggestion here is just to use only such relevant examples in the first place. Such examples can be built by our community and shared as unquestionably relevant benchmarks that enable to rank the proposed solutions or at least check, in far more convincing way, their effectiveness and scalability.

4.4 Teaching Data-related tools in control courses

This paper hopefully underlined many potential fertile combinaisons of control-based ideas and data-related tools and concepts. However, in order for such a potential to materialize in cross-discipline contributions, it is mandatory that data-mining, Machine Learning and AI tools enters the corps of the basic teaching programs of any control-oriented course. This should not be done on a tools developing level but rather in a user oriented manner. In other words, it is crucial to render common and easy to a control designer to experiment different ways of including data-oriented modules in his/her control-inspired solutions.

Such a better understanding and practicing of data-related tools enable to come out with highly effective surprising solutions but can also help demystifying some other options that might sometimes be wrongly viewed as universal magical solutions to almost any problem.

References

[1] M Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L; Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Anayo K. Akametalu, Jaime F. Fisac, Jeremy H. Gillula, Shahab Kaynama, Melanie N. Zeilinger, and Claire J. Tomlin. Reachability-based safe learning with gaussian processes. In *53rd IEEE Conference on Decision and Control*, pages 1424–1431, 2014.

[3] M. Alamir. On probabilistic certification of combined cancer therapies using strongly uncertain models. *Journal of Theoretical Biology*, 384:59–69, 2015.

[4] M. Alamir. Explicit approximation of stochastic optimal feedback control for combined therapy of cancer, 2020.

[5] M. Alamir. On the use of supervised clustering in stochastic nmpc design. *IEEE Transactions on Automatic Control*, 65(12):5392–5398, 2020.

[6] M. Alamir. A heuristic for dynamic output predictive control design for uncertain nonlinear systems. arxiv:2102.02268, 2021.

[7] M. Alamir. Partial extended observability certification and optimal design of moving horizon estimators. *IEEE Transactions on Automatic Control*, 2021.

[8] M. Alamir, D. Alberer, and L. Del Re. Identification of a class of nonlinear dynamic relationships: Application to the identification of engine emission models. *International Journal of Engine Research*, 15(8):898–905, 2014.

[9] M. Alamir, M. Fiacchini, Queinnec I., S. Tarbouriech, and M. Mazerolles. Feedback law with probabilistic certification for propofol-based control of bis during anesthesia. *International Journal of Robust and Nonlinear Control*, 28:6254–6266, 2018.

[10] M. Alamir, A. Murilo, R. Amari, P. Tona, R. Fürhapter, and P. Ortner. *On the Use of Parameterized NMPC in Real-time Automotive Control*, pages 139–149. Springer London, London, 2010.

[11] T. Alamo, R. Tempo, and E. F. Camacho. Randomized strategies for probabilistic solutions of uncertain feasibility and optimization problems. *Automatic Control, IEEE Transactions on*, 54(11):2545–2559, 2009.

[12] T. Alamo, R. Tempo, A. Luque, and Ramirez D. R. Randomized methods for design of uncertain systems: sample complexity and sequential algorithms. *Automatica*, to appear 2015.

[13] B. Amos, D. Stanton, S.and Yarats, and A. G. Wilson. On the model-based stochastic value gradient for continuous reinforcement learning learning. In Ali Jadbabaie, John Lygeros, George J. Pappas, Pablo A.;Parrilo, Benjamin Recht, Claire J. Tomlin, and Melanie N. Zeilinger, editors, *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, volume 144 of *Proceedings of Machine Learning Research*, pages 6–20. PMLR, 07 – 08 June 2021.

[14] Gillis J. Horn G. et al Andersson, J.A.E. Casadi: a software framework for nonlinear optimization and optimal control. *Journal of Theoretical Biology*, 384:59 – 69, 2015.

[15] T. Basar and G. J. Olsder. *Dynamic noncooperative game theory*. Society for Industrial and Applied Mathematics, 1999.

[15] with or without shuffle.

[16] François Bonne, Mazen Alamir, Christine Hoa, Patrick Bonnay, Michel Bon-Mardion, and Lionel Monteiro. A simulink library of cryogenic components to automatically generate control schemes for large cryorefrigerators. *IOP Conference Series: Materials Science and Engineering*, 101:012171, dec 2015.

[17] F. Borrelli, M. Baoti, J. Pekar, and G. Stewart. On the complexity of explicit mpc laws. In *2009 European Control Conference (ECC)*, pages 2408–2413, 2009.

[18] J. P. Calliess, S. J. Roberts, C. E. Rasmussen, and J. Maciejowski. Lazily adapted constant kinky inference for nonparametric regression and model-reference adaptive control. *Automatica*, 122:109216, 2020.

[19] A. Capone and S. Hirche. Anticipating the long-term effect of online learning in control. In *2020 American Control Conference (ACC)*, pages 3865–3872, 2020.

[20] F. Castaeda, M. Wulfman, A. Agrawal, T. Westenbroek, C. J. Tomlin, Shankar S., and K. Sreenath. Improving input-output linearizing controllers for bipedal robots via reinforcement learning, 2020.

[21] J. Choi, C. Fernando, C. J. Tomlin, and K. Sreenath. Reinforcement learning for safety-critical control under model uncertainty, using control lyapunov functions and control barrier functions, 2020.

[22] Francois Chollet et al. Keras, 2015.

[23] S. David, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 387–395, Bejing, China, 22–24 Jun 2014. PMLR.

[24] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 64(7):2737–2752, 2019.

[25] Jose Garcia-Tirado, John P. Corbett, Dimitri Boiroux, John Bagterp Jrgensen, and Marc D. Breton. Closed-loop control with unannounced exercise for adults with type 1 diabetes using the ensemble model predictive control. *Journal of Process Control*, 80:202–210, 2019.

[26] M. Geist and B. Scherrer. Anderson acceleration for reinforcement learning. *CoRR*, abs/1809.09501, 2018.

[27] Jeremy H. Gillula and Claire J. Tomlin. Guaranteed safe online learning via reachability: tracking a ground target using a quadrotor. In *2012 IEEE International Conference on Robotics and Automation*, pages 2723–2730, 2012.

[28] S. Gros and M. Zanon. Data-driven economic nmpc using reinforcement learning. *IEEE Transactions on Automatic Control*, 65(2):636–648, 2020.

[29] Sebastien Gros, Mario Zanon, and Alberto Bemporad. Safe reinforcement learning via projection on a safe set: How to achieve optimality? *IFAC-PapersOnLine*, 53(2):8076–8081, 2020. 21st IFAC World Congress.

[30] Lei Guo. Feedback and uncertainty: Some basic problems and results. *Annual Reviews in Control*, 49:27–36, 2020.

[31] Handan Grsoy and Mehmet nder Efe. Control system implementation on an fpga platform. *IFAC-PapersOnLine*, 49(25):425–430, 2016. 14th IFAC Conference on Programmable Devices and Embedded Systems PDES 2016.

[32] L. Hewing, K. P. Waberisch, M. Menner, and M. N. Zeilinger. Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:260–296, 2020.

[33] T. Hirshberg, S. Vemprala, and A. Kapoor. Safety considerations in deep control policies with safety barrier certificates under uncertainty. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6245–6251, 2020.

[34] Zhong-Sheng Hou and Zhuo Wang. From model-based control to data-driven control: Survey, classification and perspective. *Information Sciences*, 235:3–35, 2013. Data-based Control, Decision, Scheduling and Fault Diagnostics.

[35] A. Isidori. *Nonlinear Control Systems*. Springer-Verlag, 1995.

[36] Degrave J., F. Felici, and J. et al. Buchli. Magnetic control of tokamak plasmas through deep reinforcement learning. *nature*, 602:414–419, 2022.

[37] T. C. Jansen. Gpu++: an embedded gpu development system for general-purpose computations. In *Computer Science*, 2007.

[38] B. Karg, T. Alamo, and S. Lucia. Probabilistic performance validation of deep learningbased robust nmpc controllers. *International Journal of Robust and Nonlinear Control*, Jul 2021.

[39] B. Karg and S. Lucia. Learning-based approximation of robust nonlinear predictive control with state estimation applied to a towing kite. In *2019 18th European Control Conference (ECC)*, pages 16–22, 2019.

[40] H.K. Khalil. *Nonlinear Systems*. Prentice Hall, 2002.

[41] S. Lucia, T. Finkler, D. Basak, and S. Engell. A new robust nmpc scheme and its application to a semi-batch reactor example*. *IFAC Proceedings Volumes*, 45(15):69–74, 2012. 8th IFAC Symposium on Advanced Control of Chemical Processes.

[42] E. T. Maddalena and C. N. Jones. Learning non-parametric models with guarantees: A smooth lipschitz regression approach. *IFAC-PapersOnLine*, 53(2):965–970, 2020. 21st IFAC World Congress.

[43] J. M. Manzano, D. Limon, D. Muoz de la Pea, and J. P. Calliess. Robust learning-based mpc for nonlinear constrained systems. *Automatica*, 117:108948, 2020.

[44] Andreas B. Martinsen, Anastasios M. Lekkas, and Sbastien Gros. Combining system identification with reinforcement learning-based mpc. *IFAC-PapersOnLine*, 53(2):8130–8135, 2020. 21st IFAC World Congress.

[45] Rubn Mart, Sergio Lucia, Daniel Sarabia, Radoslav Paulen, Sebastian Engell, and Csar de Prada. Improving scenario decomposition algorithms for robust nonlinear model predictive control. *Computers & Chemical Engineering*, 79:30–45, 2015.

[46] D. Q. Mayne, J.B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

[47] A. Mesbah. Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, 36(6):30–44, 2016.

[48] A. Mesbah. Stochastic model predictive control with active uncertainty learning: A survey on dual control. *Annual Reviews in Control*, 45:107 – 117, 2018.

[49] Modelica Association. Modelica - a unified object-oriented language for physical systems modeling. Tutorial, December 2000.

[50] A. T. Mohamed and M. Alamir. Robust output feedback controller for a class of nonlinear systems with actuator dynamicsthis work is funded by innov-hydro project. *IFAC-PapersOnLine*, 51(25):275–280, 2018. 9th IFAC Symposium on Robust Control Design ROCOND 2018.

[51] Bertsekas D. P. *Reinforcement learning and optimal control*. Athena Scientific, July 2019.

[52] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[53] K. M. M Rathai, O. Sename, and M. Alamir. Gpu-based parameterized nmpc scheme for control of half car vehicle with semi-active suspension system. *IEEE Control Systems Letters*, 3(3):631–636, 2019.

[54] James B. Rawlings. *Moving Horizon Estimation*, pages 1–7. Springer London, London, 2013.

[55] Sutton R. S. and Barto A. G. *Reinforcement learning: An introduction*. MIT Press, November 2018.

[56] R. Sepulchre. To know or to predict. *IEEE Control Systems*, April 2020.

[57] P. Tondel, T. Arne-Johansen, and A. Bemporad. An algorithm for multi-parametric quadratic programming and explicit mpc solutions. *Automatica*, 39(3):489–497, 2003.

[58] J. Umlauft and S. Hirche. Feedback linearization based on gaussian processes with event-triggered online learning. *IEEE Transactions on Automatic Control*, 65(10):4154–4169, 2020.

[59] M. Vidyasagar. Randomized algorithms for robust controller synthesis using statistical learning theory. *Automatica*, 37(10):1515–1528, 2001.

[60] Chuanzheng Wang, Yinan Li, Yiming Meng, Stephen L. Smith, and Jun Liu. Learning control barrier functions with high relative degree for safety-critical control, 2020.

[61] T. Westenbroek, D. Fridovich-Keil, E. Mazumdar, S. Arora, V. Prabhu, S. S. Sastry, and C. J. Tomlin. Feedback linearization for uncertain systems via reinforcement learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1364–1371, 2020.

[62] L Yu, A. Goldsmith, and S. Di Cairano. Efficient convex optimization on gpus for embedded model predictive control. In *Proceedings of the General Purpose GPUs*, 2017.

[63] Mario Zanon and Sebastien Gros. Safe reinforcement learning using robust mpc. *IEEE Transactions on Automatic Control*, 66(8):3638–3652, 2021.

[64] A.Z. Zeyuan, L. Yuanzhi, and S. Zhao. A convergence theory for deep learning via over-parameterization. *CoRR*, abs/1811.03962, 2018.

[65] J. Zhu, K. Muandet, M. Diehl, and B. Schölkopf. A new distribution-free concept for representing, comparing, and propagating uncertainty in dynamical systems with kernel probabilistic programming. In *IFAC PapersOnLine*, pages 7240–7247. IFAC, 2020.