International Journal of Tomography & Statistics
Summer 2007, Vol. 6, No. S07; Int. J. Tomogr. Stat.; 122-127
ISSN 0972-9976; Copyright © 2007 by IJTS, ISDER

MODEL PREDICTIVE CONTROL AND REINFORCEMENT LEARNING AS TWO COMPLEMENTARY FRAMEWORKS

Damien Ernst [1] **Mevludin Glavic**
Florin Capitanescu Louis Wehenkel

Dept of Electrical Engineering and Computer Science,
University of Liège, Belgium
Email: {ernst,glavic,capitane,lwh}@montefiore.ulg.ac.be

Abstract: Model predictive control (MPC) and reinforcement learning (RL) are two popular families of methods to control system dynamics. In their traditional setting, they formulate the control problem as a discrete-time optimal control problem and compute a suboptimal control policy. We present in this paper in a unified framework these two families of methods. We run for MPC and RL algorithms simulations on a benchmark control problem taken from the power system literature and discuss the results obtained.

Keywords: Model predictive control, reinforcement learning, interior-point method, fitted Q iteration

1. INTRODUCTION

Reinforcement learning and model predictive control are two families of control techniques which tackle control problems by formalizing them as optimal control problems. While MPC techniques assume that a model of the optimal control problem is available, reinforcement learning techniques assume that the only information available from the model is the one gathered from interaction with the system. Both families of techniques usually compute a suboptimal control policy to the control problem. For MPC techniques, the suboptimalities come mainly from the simplifications done to the original optimal control problem for lightening the computational burdens, while for RL techniques, they also originate from the lack of information on the model.

The paper is an attempt to present some MPC and RL algorithms in a unified framework and reports, for the first time, simulation results obtained by both families of techniques on the same optimal control problem.

Over the last decades researchers from both the MPC and the RL field have proposed numerous algorithms that can be applied to various types of systems (systems of differential equations, Markov decision processes, etc). We have decided to focus in this paper on discrete-time deterministic systems and we have selected, for easing the comparison, one particular type of algorithm for each field. The MPC algorithm selected solves the optimal control problem by computing open-loop policies through the resolution of a non-linear optimization problem in which the system dynamics is represented by equality constraints (Morari and Lee, 1999). Such a choice was in some sense normal since it is the dominant approach to MPC. As RL algorithm, we have chosen an algorithm, known as fitted Q iteration, which computes from the information gathered from interaction with the system, an approximation of the optimal control policy by solving a sequence of batch-mode supervised learning regression problems (Ernst et al., 2005). Two main reasons have motivated this choice. First, this algorithm has been shown to clearly outperform other RL algorithms which makes it a good candidate to become the standard approach to RL. Secondly, in its essence, and this contrary to many other existing RL algorithms which have been designed to solve specifically infinite time horizon control problem, the fitted Q iteration algorithm solves finite time horizon problems. Since such a feature is also shared by MPC techniques, the presentation of RL and MPC in a unified framework gets simplified.

[1] Damien Ernst is a postdoctoral researcher of the Belgian FNRS (Fonds National de la Recherche Scientifique), of which he acknowledges the financial support.

2. MPC AND RL IN THE FINITE HORIZON CASE

2.1 The optimal control problem

We consider a discrete-time system whose dynamics over T-stages is described by a time-invariant equation:

$$x_{t+1} = f(x_t, u_t) \quad t = 0, 1, \cdots, T-1 \qquad (1)$$

where $\forall t$, the state x_t is an element of the state space X and the action u_t is an element of the action space U. $T \in \mathbb{N}_0$ is referred to as the *optimization horizon*.

The transition from t to $t+1$ is associated with an instantaneous cost signal $c_t = c(x_t, u_t) \in \mathbb{R}$ which we assume bounded by a constant B_c, and for every initial state x_0 and for every sequence of actions we define the discounted *cost over T stages* by

$$C_T^{\{u_0, u_1, \cdots, u_{T-1}\}}(x_0) = \sum_{t=0}^{T-1} \gamma^t c(x_t, u_t) \qquad (2)$$

where $\gamma \in [0, 1[$ is the discount factor.

In this context, an optimal control sequence u_0^*, u_1^*, $\cdots$, u_{T-1}^*, is a sequence of actions which minimizes the cost over T stages. [2] We denote by $C_T^\pi(x)$ the cost over T stages associated with a policy π when the initial state is $x_0 = x$. A *T-stage optimal policy* is a policy that leads for every initial state $x_0 = x$ to a minimal $C_T^\pi(x)$. This is the kind of policy we are looking for.

We consider three different types of policies: *open-loop* policies which select at time t the action u_t based only on the initial state x_0 of the system and the current time ($u_t = \pi_o(t, x_0)$), *closed-loop* policies which select the action u_t based on the current time and the current state ($u_t = \pi_c(t, x_t)$) and closed-loop *stationary* policies for which the action is selected only based on the knowledge of the current state ($u_t = \pi_s(x_t)$).

To characterize optimality of these T-stage policies, we define iteratively the sequence of functions $Q_N : X \times U \to \mathbb{R}$, $N = 1, \cdots, T$ as follows:

$$Q_N(x, u) = c(x, u) + \gamma \inf_{u' \in U} Q_{N-1}(f(x, u), u') \qquad (3)$$

with $Q_0(x, u) = 0\,, \forall (x, u) \in X \times U$.

We have the following theorem (see e.g. Bertsekas (2000) for a proof):

Theorem 1. A sequence of actions u_0^*, u_1^*, $\cdots$, u_{T-1}^* is optimal if and only if $Q_{T-t}(x_t, u_t^*) = \inf_{u' \in U} Q_{T-t}(x_t, u')$ $\forall t \in \{0, \cdots, T-1\}$.

Under various sets of additional assumptions (e.g. U finite or see Hernández-Lerma and Lasserre (1996) when U is infinite), the existence of an optimal closed-loop (open-loop) policy which is a T-stage optimal policy is guaranteed. We use the notation $\pi_{c,T}^*$ ($\pi_{o,T}^*$) to refer to a closed-loop (open-loop) T-stage optimal policy. From Theorem 1, we see that every policy $\pi_{c,T}^*$ is such that $\pi_{c,T}^*(x, t) \in \{u \in U | Q_{T-t}(x, u) = \inf_{u' \in U} Q_{T-t}(x, u')\}$. Similarly, for every policy $\pi_{o,T}^*$ we have $\pi_{o,T}^*(x_0, t) \in \{u \in U | Q_{T-t}(x, u) = \inf_{u' \in U} Q_{T-t}(x_t, u')\}$ with $x_t = f(x_{t-1}, \pi_{o,T}^*(x_0, t-1))$, $\forall t = 1, \cdots, T-1$.

2.2 Model predictive control

Model predictive control techniques target an optimal open-loop policy $\pi_{o,T}^*$, and assume that the system dynamics and cost function are available in analytical form.

For a given initial state x_0, the terms $\pi_{o,T}^*(x_0, t)$, $t = 0, 1, \cdots, T-1$ of the optimal open-loop policy may then be computed by solving the minimization problem:

$$\inf_{\substack{(u_0, u_1, \cdots, u_{T-1}, x_1, x_2, \cdots, x_{T-1}) \\ \in U \times \cdots \times U \times X \times \cdots \times X}} \sum_{t=0}^{T-1} \gamma^t c(x_t, u_t) \qquad (4)$$

subject to the T equality constraints (1).

Under appropriate assumptions, the minimization problem (4) can be tackled by classical convex programming algorithms. However, for many practical problems, its resolution may be a difficult task, with no guarantees that the solution found by the used optimizer is indeed optimal.

Therefore, the MPC techniques actually rather produce an approximation $\hat{\pi}_{o,T}^*$ of a T-stage optimal control policy. The better the approximation, the smaller $C_T^{\hat{\pi}_{o,T}^*}(x_0) - C_T^{\pi_{o,T}^*}(x_0)$.

2.3 Reinforcement learning

Reinforcement learning techniques do not assume that the system dynamics and the cost function are given in analytical (or even algorithmic) form. The sole information they assume available about the system dynamics and the cost function is the one that can be gathered from the observation of system trajectories. Reinforcement learning techniques compute from this an *approximation* $\hat{\pi}_{c,T}^*$ of a T-stage optimal (closed-loop) policy since, except for very special conditions, the exact optimal policy can not be decided from such a limited amount of information.

[2] In this problem statement, we did not explicitly consider constraints other than those implied by the system dynamics. Note however that static constraints (e.g. operating limits) can be modeled in this formulation by penalizing the cost function, and dynamic ones (e.g. energy and time limitations) by first introducing additional state variables and adding penalizing terms on these latter.

The *fitted Q iteration* algorithm on which we focus in this paper, actually relies on a slightly weaker assumption, namely that a set of one step system transitions is given, each one providing the knowledge of a new sample of information (x_t, u_t, c_t, x_{t+1}) that we name four-tuple.

We denote by $\mathcal{F}$ the set $\{(x_t^l, u_t^l, c_t^l, x_{t+1}^l)\}_{l=1}^{\#\mathcal{F}}$ of available four-tuples. Fitted Q iteration computes from $\mathcal{F}$ the functions $\hat{Q}_1, \hat{Q}_2, \cdots, \hat{Q}_T$, approximations of the functions $Q_1, Q_2, \cdots, Q_T$ defined by Eqn (3), by solving a sequence of batch-mode supervised learning problems. From these, a policy which satisfies

$$\hat{\pi}^*_{c,T}(t,x) \in \{u \in U | \hat{Q}_{T-t}(x,u) = \inf_{u' \in U} \hat{Q}_{T-t}(x,u')\}$$

is taken as approximation of an optimal control policy.

Posing $\hat{Q}_0(x,u) = 0, \forall (x,u) \in X \times U$, the training set for the Nth supervised learning problem of the sequence $(N \geq 1)$ is

$$\left\{(i^l, o^l) = \left((x_t^l, u_t^l), r_t^l + \gamma \inf_{u \in U} \hat{Q}_{N-1}(x_{t+1}^l, u)\right)\right\}_{l=1}^{\#\mathcal{F}}$$

where the i^l (resp. o^l) denote inputs (resp. outputs). The supervised learning regression algorithm produces from the sample of inputs and outputs the function $\hat{Q}_N$.

3. TIME HORIZON TRUNCATION

Let $\pi^*_{s,T}$ denote a stationary policy such that $\pi^*_{s,T}(x) \in \{u \in U | Q_T(x,u) = \inf_{u' \in U} Q_T(x,u')\}$ and let π^*_T denote a T-stage optimal control policy.

We have the following theorem (proof in Ernst et al. (2006)):

Theorem 2. For any $T, T' \in \mathbb{N}_0$ such that $T' < T$, we have the following bound on the suboptimality of $\pi^*_{s,T'}$ when used as solution of a T-stage optimal control problem:

$$\sup_{x \in X} \left(C_T^{\pi^*_{s,T'}}(x) - C_T^{\pi^*_T}(x)\right) \leq \frac{\gamma^{T'}(4-2\gamma)B_c}{(1-\gamma)^2} \quad (5)$$

Theorem 2 shows that the suboptimality of the policy $\pi^*_{s,T'}$ when used as solution of an optimal control problem with an optimization horizon T $(T > T')$ can be upper bounded by an expression which decreases exponentially with T'.

When dealing with a large or even infinite optimization horizon T, MPC and RL algorithms use this property to truncate the time horizon, so as to reduce computational burdens. In other words, they solve an optimal control problem with a time horizon $T' < T$ and compute an approximation $\hat{\pi}^*_{s,T'}$ of $\pi^*_{s,T'}$.

Since $\pi_s(x) = \pi^*_{o,T'}(x,0)$ is a $\pi^*_{s,T'}$ policy, MPC methods produce the following policy: $\hat{\pi}^*_{s,T'}(x) = \hat{\pi}^*_{o,T'}(x,0)$. Similarly, the RL algorithm outputs a policy $\hat{\pi}^*_{s,T'}(x) = \hat{\pi}^*_{c,T'}(x,0)$.

4. EXPERIMENTS

In this section, we present simulation results obtained by using MPC and RL techniques. The section starts with a description of the optimal control problem considered. Then, we run experiments on this optimal control problem with the reinforcement learning algorithm and, afterwards, with the MPC algorithm.

4.1 Control problem

We consider the problem of controlling the benchmark power system represented on Fig. 1a. This academic power system example is composed of a generator connected to a machine of infinite size through a transmission line. On the transmission line is installed a variable reactance which influences the amount of electrical power transmitted through the line. The system has two state variables, the angle δ and the speed ω of the generator. The dynamics of these two state variables is given by the set of differential equations:

$$\begin{aligned} \dot{\delta} &= \omega \\ \dot{\omega} &= \frac{P_m - P_e}{M} \quad \text{with} \quad P_e = \frac{EV}{u + X_{system}} \sin\delta \end{aligned}$$

where P_m, M, E, V and X_{system} are parameters equal respectively to 1, 0.03183, 1, 1 and 0.4. P_m represents the mechanical power of the machine, M its inertia, E its terminal voltage, V the voltage of the terminal bus system and X_{system} the overall system reactance. When the uncontrolled system is driven away from its equilibrium point $(\delta_e, \omega_e) = (\arcsin(\frac{X_{system} P_m}{EV}), 0)$, undamped electrical power (P_e) oscillations appear in the line (Fig. 1b). A variable reactance u has been installed in series with the overall reactance X_{system}. The control problem consists in finding a control policy for this variable reactance which damps the electrical power oscillations.

From this continuous time control problem, we define a discrete-time optimal control problem with infinite time optimization horizon $(T \rightarrow \infty)$ such that policies π leading to small costs $\lim_{T \rightarrow \infty} C_T^{\pi}(x)$, also tend to produce good damping of P_e.

The discrete-time dynamics is obtained by discretizing the time with the time between t and $t+1$ chosen equal to $0.050\,s$. The value of u is chosen constant during each $0.050\,s$ interval. If δ_{t+1} and ω_{t+1} are such that they do not belong anymore to the stability domain of the uncontrolled $(u = 0)$ system (Fig. 1c), that is if

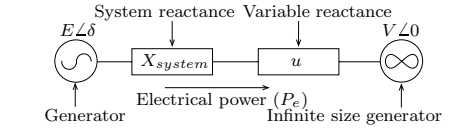
$$\frac{1}{2}M\omega_{t+1}^2 - P_m\delta_{t+1} - \frac{EV\cos(\delta_{t+1})}{X_{system}} > -0.438788,$$

then a *terminal state* is supposed to be reached. A terminal state is a state in which the system remains stuck, i.e. *term. state* $= f($*term. state*$, u)$ $\forall u \in U$.

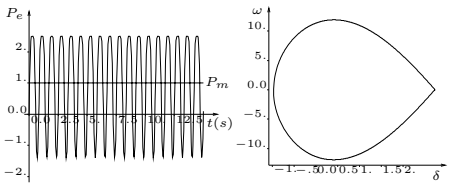
The state space X is thus composed of the uncontrolled system's stability domain (Fig. 1c) plus one terminal state. The action space U is chosen such that $U = \{u \in \mathbb{R} | u \in [-0.16, 0]\}$. The decay factor γ is equal to 0.95. The cost function $c(x, u)$ was chosen to penalize deviations of the electrical power from its steady state value (P_e at steady state $= P_m$). Furthermore, since control actions should not drive the system towards the terminal state, a very large cost is associated with such actions. The value of this latter cost was chosen such that a policy which drives the system outside of the stability domain, whatever the optimization horizon T, is necessarily suboptimal. The exact cost function is:

$$c(x_t, u_t) = \begin{cases} (P_{e t+1} - P_m)^2 & \text{if } x_{t+1} \neq \textit{term. state} \\ 0 & x_t = \textit{term. state} \\ 1000 & \text{otherwise} \end{cases} \quad (6)$$

where $P_{e t+1} = \frac{EV}{X_{system}+u_t}\sin(\delta_{t+1})$.



(a) Representation of the power system.



(b) Electrical power oscillations when at $t = 0$ $(\delta, \omega) = (0, 8)$ and $u = 0$

(c) Stability domain of the uncontrolled ($u = 0$) power system.

Fig. 1. Some insights into the control problem.

4.2 Reinforcement learning

Fitted Q iteration algorithm. The fitted Q iteration algorithm computes $\hat{\pi}_{s,T'}$ by solving sequentially T' batch mode supervised learning problems. As batch-mode supervised learning algorithm, we have chosen the Extra-Trees algorithm (Ernst et al., 2005; Geurts et al., 2006). At each iteration of the fitted Q iteration algorithm, one also needs to compute $\inf_{u \in U} \hat{Q}(x_{t+1}^l, u)$ for each $l \in \{1, 2, \cdots, \#\mathcal{F}\}$. To compute this minimum, we have considered a subset $U'\{0, -0.016, \cdots, -0.16\}$ of U and computed the minimum over this subset. Similarly, after T' iterations of the algorithm, the policy $\hat{\pi}_{s,T'}$ output by the RL algorithm is chosen equal to $\hat{\pi}_{s,T'}(x) = \arg\inf_{u \in U'} \hat{Q}_{T'}(x, u)$.

	Value of T'			
	1	5	20	100
RL, $\#\mathcal{F} = 10^5$	$44.3^\dagger$	39.3	20.5	20.9
RL, $\#\mathcal{F} = 10^3$	$43.9^\dagger$	47.2	27.7	26.9
MPC	$55.2^\dagger$	38.2	20.3	20.3

Table 1. Value of $\lim_{T\to\infty} C_T^{\hat{\pi}^*_{s,T'}}((0, 8))$ estimated by simulating the system with $\hat{\pi}^*_{s,T'}$ from $(\delta, \omega) = (0, 8)$ over 1000 time steps. The suffix † indicates that the policy drives the system to the *terminal state* before $t = 1000$. Non-linear optimization algorithm did not converge at time $t = 164$ when $T' = 20$.

Four-tuples generation. To collect the four-tuples we have considered 100,000 one step episodes with x_0 and u_0 for each episode drawn at random in $X \times U$.

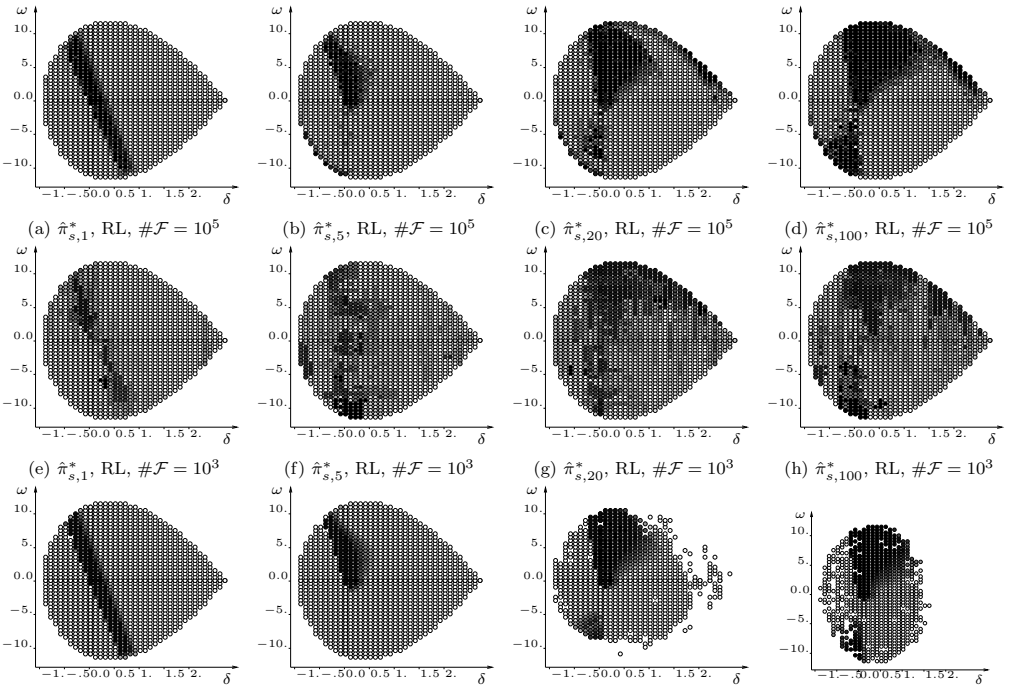
Results. On Figs 2a-c, we have represented the policy $\hat{\pi}_{s,T'}$ computed for increasing values of T'. As we may observe, the policy considerably changes with T'. To assess the influence of T' on the ability of the policy $\hat{\pi}_{s,T'}$ to approximate an optimal policy over an infinite time horizon, we have computed for different values of T', the value of $\lim_{T\to\infty} C_T^{\hat{\pi}_{s,T'}}((\delta, \omega) = (0, 8.))$. The results are reported on the first line of Table 1. We see that the cost tends to decrease when T' increases. That means that the suboptimality of the policy $\hat{\pi}_{s,T'}$ tend to decrease with T', as suggested by Theorem 2.

Performances of the fitted Q iteration algorithm are influenced by the information it has on the optimal control problem, represented by the set of $\mathcal{F}$. Usually, the less information, the larger $\lim_{T\to\infty} C_T^{\hat{\pi}^*_{s,T'}}(x) - \lim_{T\to\infty} C_T^{\pi^*_{s,T'}}(x)$, assuming that T' is sufficiently large. To illustrate this, we have run the RL algorithm by considering this time a 1000 element set of four-tuples, with the four-tuples generated in the same conditions as before. The resulting policies $\hat{\pi}^*_{s,T'}$ are drawn on Figs 2d-e. As shown on the second line of Table 1, these policies tend indeed to lead to higher costs than those observed by considering 100,000 four-tuples.

We were mentioning before, when defining the optimal control problem, that a policy leading to small costs was also leading to good damping of P_e. This is illustrated on Fig. 3a.

4.3 Model predictive control

Non-linear optimization problem. At the core of the MPC approach, there is the resolution

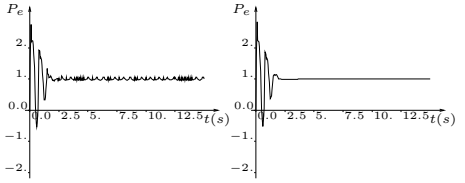


(a) $\hat{\pi}^*_{s,1}$, RL, $\#\mathcal{F} = 10^5$ (b) $\hat{\pi}^*_{s,5}$, RL, $\#\mathcal{F} = 10^5$ (c) $\hat{\pi}^*_{s,20}$, RL, $\#\mathcal{F} = 10^5$ (d) $\hat{\pi}^*_{s,100}$, RL, $\#\mathcal{F} = 10^5$

(e) $\hat{\pi}^*_{s,1}$, RL, $\#\mathcal{F} = 10^3$ (f) $\hat{\pi}^*_{s,5}$, RL, $\#\mathcal{F} = 10^3$ (g) $\hat{\pi}^*_{s,20}$, RL, $\#\mathcal{F} = 10^3$ (h) $\hat{\pi}^*_{s,100}$, RL, $\#\mathcal{F} = 10^3$

(i) $\hat{\pi}^*_{s,1}$, MPC (j) $\hat{\pi}^*_{s,5}$, MPC (k) $\hat{\pi}^*_{s,20}$, MPC (l) $\hat{\pi}^*_{s,100}$, MPC

Fig. 2. Representation of $\hat{\pi}^*_{s,T'}(x)$. The evaluation is carried out for $\{(\delta,\omega) \in X |\ \exists i,j \in \mathbb{Z} | (\delta,\omega) = (0.1 * i, 0.5 * j)\}$. The grey level of a bullet associated with a state x gives information about the magnitude of $|\hat{\pi}^*_{s,T'}(x)|$. Black bullets corresponds to the largest possible value of $|\hat{\pi}^*_{s,T'}|$ (-0.16), white bullets to the smallest one (0) and grey to intermediate values with the larger the magnitude of $|\hat{\pi}^*_{T'}|$, the darker the grey. Figures a-d gives for different values of T' the policies $\hat{\pi}^*_{s,T'}(x)$ obtained by the RL algorithm with 100,000 four-tuples, figures g-h with the RL algorithm with 1000 four-tuples and Figs i-l with the MPC algorithm. On these latter four figures, states for which the MPC algorithm failed to converge are not represented.



(a) $\hat{\pi}^*_{s,100}$, RL, $\#\mathcal{F}{=}10^5$ (b) $\hat{\pi}^*_{s,100}$, MPC

Fig. 3. Evolution of P_e when the system starts from $(\delta_0, \omega_0) = (0, 8)$ and is controlled by the policy $\hat{\pi}^*_{s,100}$.

of a non-linear programming problem. It has been stated as follows:

$$\inf_{(\delta_1,\cdots,\delta_{T'},\omega_1,\cdots,\omega_{T'},u_0,\cdots,u_{T'-1})\in\mathbb{R}^{3T'}} \Big(\sum_{t=0}^{T'-1} \gamma^t \Big(\frac{EV}{X_{system}+u_t}\sin(\delta_{t+1}) - P_m\Big)^2\Big) \qquad (7)$$

subject to $2T'$ equality constraints ($t = 0, 1, \cdots, T'-1$):

$$\delta_{t+1} - \delta_t - (h/2)\omega_t - (h/2)\omega_{t+1} = 0 \qquad (8)$$

$$\omega_{t+1} - \omega_t - (h/2)\frac{1}{M}\left(P_m - \frac{EV\sin\delta_t}{u_t + X_{system}}\right) - (h/2)\frac{1}{M}\left(P_m - \frac{EV\sin\delta_{t+1}}{u_t + X_{system}}\right) = 0 \qquad (9)$$

with $h = 0.05\,s$ and $3T'$ inequality constraints ($t = 0, 1, \cdots, T'-1$):

$$u_t \le 0 \qquad (10)$$

$$-u_t \le 0.16 \qquad (11)$$

$$\frac{1}{2}M\omega_{t+1}^2 - P_m\delta_{t+1} - \frac{EV}{X_{system}}\cos(\delta_{t+1}) + 0.438788 \le 0 \qquad (12)$$

Several choices have been made to state the non-linear optimization problem. First, the cost of 1000 that occurs when the system reaches a terminal state does not appear in the cost functional (7) and has been replaced by the inequality constraints (12). These constraints limit the search for an optimal policy to open-loop policies that guarantee that the system stays inside the stability domain of the uncontrolled system. It does not lead to any suboptimality since we know that

policies driving the system to a terminal state are suboptimal policies. Another choice that deserves explanations is the way the equality constraints $x_{t+1} = f(x_t, u_t)$ are modeled. To model them, we have relied on a trapezoidal method, with a step size of $0.05\,s$, the time between t and $t+1$ (Eqns (8-9)).

Non-linear optimization algorithm. As non-linear optimizer, we have used the primal-dual Interior-Point Method (Kortanek et al., 1991).

Results. Figures 2i-l represent the different policies $\hat{\pi}_{s,T'}$ computed for increasing values of T'. To compute the value of $\hat{\pi}_{s,T'}$ for a specific state x, we need to run the optimization algorithm. As reported on these figures, for some states, the algorithm fails to converge. Also, the larger the value of T', the more frequently the algorithm fails to converge, which is not surprising since the complexity of the optimization problem tends to increase with T'. For example, on the 1304 states x for which the policy $\hat{\pi}_{s,T'}$ was estimated, 1 failed to converge for $T' = 1$, 2 for $T' = 5$, 340 for $T' = 20$ and 474 when $T' = 100$. One may reasonably wonder whether the convergence problems are not due to the fact that for some values of x_0 and T', there are no solutions to the optimization problem. If the equality constraints $x_{t+1} = f(x_t, u_t)$ were perfectly represented, it would not be the case since when $u_t = 0$, $t = 0, 1, \cdots, T'-1$, we know that the trajectory stays inside the stability domain. However, with a discrete dynamics approximated by Eqns (8-9), it may not be the case anymore. To know whether most of these convergence problems were indeed caused by approximating the equality constraints, we have simulated the system modeled by Eqns (8-9) over 100 time steps and with $u_t = 0$, $\forall t$. We found out that for the 1304 states for which the policy was estimated, 20 were indeed leading to a violation of the constraints when chosen as initial state. This is however a small number compared to the 474 states for which the non-linear optimization algorithm failed to converge.

It is interesting to notice that, if we take apart the states for which convergence did not occur, MPC policies look similar to RL policies computed with a large number of samples. This is not surprising, since both methods aim to approximate $\pi_{s,T'}$ policies. Table 1 reports the costs over an infinite time horizon obtained by using MPC policies for various values of T'. When $T' = 20$, the MPC algorithm failed to converge for some states met, which is why the value of $\lim_{T\to\infty} C_T^{\hat{\pi}_{s,20}}$ is not given. We observe that for $T' = 5$ and $T' = 100$, the policies computed by MPC outperform those computed by RL. Figure 3 shows that, while the $\hat{\pi}_{s,100}$ policy computed by the RL algorithm produces some residual oscillations, the MPC $\hat{\pi}_{s,100}$ policy is able to damp them completely. This is explained, among others, by the ability the MPC algorithm has to exploit the continuous nature of the action space while the RL algorithm discretizes it into a finite number of values.

5. CONCLUSIONS

We have presented in this paper reinforcement learning and model predictive control in a unified framework and run, for both techniques, experiments on a same optimal control problem. From these experiments, two main conclusions can be drawn. Firstly, the RL algorithm considered, known as fitted Q iteration, was offering good performances. This suggests that it may perhaps be a good alternative to MPC approaches when the system dynamics and/or the cost function are totally or partially unknown. Indeed, rather than use first the information gathered from interaction to identify the system dynamics and/or the cost function, one could directly extract from this information the control policy by using RL algorithms. Secondly, MPC approaches usually compute from the model a suboptimal policy by solving a non-linear optimization problem. However, we found out that non-linear optimizers may fail to find a good solution, especially when the optimization horizon grows. In some sense, we could have circumvented these convergence problems by using the fitted Q iteration algorithm together with a procedure to generate the four-tuples from the model. More generally, we may question whether it would not be preferable when the model is known to rely more on algorithms exploiting the dynamic programming principle, as fitted Q iteration does, than to compute immediately an optimal sequence of actions by solving a non-linear optimization problem for which the system dynamics is represented through equality constraints.

REFERENCES

D.P. Bertsekas. *Dynamic Programming and Optimal Control*, volume I. Athena Scientific, Belmont, MA, 2nd edition, 2000.

D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, April 2005.

D. Ernst, M. Glavic, F. Capitanescu, and L. Wehenkel. Reinforcement learning and model predictive control: a comparison. *Submitted*, 2006.

P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *To appear in Machine Learning*, 2006.

O. Hernández-Lerma and J.B. Lasserre. *Discrete-Time Markov Control Processes. Basic Optimality Criteria*. Springer, New-York, 1996.

K. Kortanek, F. Potra, and Y. Ye. On some efficient interior point methods for nonlinear convex programming. *Linear Algebra and its Applications*, 169-189, April 1991.

M. Morari and J.H. Lee. Model predictive control: past, present and future. *Computers and Chemical Engineering*, 23:667–682, 1999.