A CAUTIOUS APPROACH TO GENERALIZATION IN REINFORCEMENT LEARNING

Raphael Fonteneau[1], Susan A. Murphy[2], Louis Wehenkel[1] and Damien Ernst[1]
[1] *Department of Electrical Engineering and Computer Science, University of Liège, Belgium*

[2] *Department of Statistics, University of Michigan, USA*

Keywords: Reinforcement learning, Prior knowledge, Cautious generalization.

Abstract: In the context of a deterministic Lipschitz continuous environment over continuous state spaces, finite action spaces, and a finite optimization horizon, we propose an algorithm of polynomial complexity which exploits weak prior knowledge about its environment for computing from a given sample of trajectories and for a given initial state a sequence of actions. The proposed Viterbi-like algorithm maximizes a recently proposed lower bound on the return depending on the initial state, and uses to this end prior knowledge about the environment provided in the form of upper bounds on its Lipschitz constants. It thereby avoids, in way depending on the initial state and on the the prior knowledge, those regions of the state space where the sample is too sparse to make safe generalizations. Our experiments show that it can lead to more cautious policies than algorithms combining dynamic programming with function approximators. We give also a condition on the sample sparsity ensuring that, for a given initial state, the proposed algorithm produces an optimal sequence of actions in open-loop.

1 INTRODUCTION

Since the late sixties, the field of Reinforcement Learning (RL) (Sutton and Barto, 1998) has studied the problem of inferring from the sole knowledge of observed system trajectories, near-optimal solutions to optimal control problems. The original motivation was to design computational agents able to learn by themselves how to interact in a rational way with their environment. The techniques developed in this field have appealed researchers trying to solve sequential decision making problems in many fields such as Finance (Ingersoll, 1987), Medicine (Murphy, 2003; Murphy, 2005) or Engineering (Riedmiller, 2005).

RL algorithms are challenged when dealing with large or continuous state spaces. Indeed, in such cases they have to generalize the information contained in a generally sparse sample of trajectories. The dominating approach for generalizing this information is to combine RL algorithms with function approximators (Bertsekas and Tsitsiklis, 1996; Lagoudakis and Parr, 2003; Ernst et al., 2005). Usually, these approximators generalize the information contained in the sample to areas poorly covered by the sample by implicitly assuming that the properties of the system in those areas are similar to the properties of the system in the nearby areas well covered by the sample. This in turn often leads to low performance guarantees on the inferred policy when large state space areas are poorly covered by the sample. This can be explained by the fact that when computing the performance guarantees of these policies, one needs to take into account that they may actually drive the system into the poorly visited areas to which the generalization strategy associates a favorable environment behavior, while the environment may actually be particularly adversarial in those areas. This is corroborated by theoretical results which show that the performance guarantees of the policies inferred by these algorithms degrade with the sample sparsity where, loosely speaking, the sparsity can be seen as the radius of the largest non-visited state space area.[1]

We propose an algorithm for reinforcement learn-

[1]Usually, these theoretical results do not give lower bounds per se but a distance between the return of the inferred policy and the optimal return. However, by adapting in a straightforward way the proofs behind these results, it is often possible to get a bound on the distance between the estimate of the return of the inferred policy computed by the RL algorithm and its actual return and, from there, a lower bound on the return of the inferred policy.

ing that derives action sequences which tend to avoid regions where the performance is uncertain given the available set and weak prior knowledge. This weak prior knowledge is given in the form of upper bounds on the Lipschitz constants of the environment. To compute these sequences of actions, the algorithm exploits a lower bound on the performance of the agent when it uses a certain open-loop sequence of actions while starting from a given initial condition, as proposed in (Fonteneau et al., 2009). More specifically, it computes an open-loop sequence of actions to be used from a given initial state to maximize that lower bound. To this end we derive a Viterbi-like algorithm, of polynomial computational complexity in the size of the dataset and the optimization horizon. Our algorithm does not rely on function approximators and it computes, as a byproduct, a lower bound on the return of its open-loop sequence of decisions. It essentially adopts a cautious approach to generalization in the sense that it computes decisions that avoid driving the system into areas of the state space that are not well enough covered by the available dataset, according to the prior information about the dynamics and reward function. Our algorithm – named CGRL for Cautious Generalization (oriented) Reinforcement Learning – assumes a finite action space and a deterministic dynamics and reward function, and it is formulated for finite time-horizon problems. We also provide a condition on the sample sparsity ensuring that, for a given initial state, the proposed algorithm produces an optimal sequence of actions in open-loop, and we suggest directions for leveraging our approach to a larger class of problems in RL.

The rest of the paper is organized as follows. Section 2 briefly discusses related work. In Section 3, we formalize the inference problem we consider. In Section 4, we adapt the results of (Fonteneau et al., 2009) to compute a lower bound on the return of an open-loop sequence of actions. Section 5 proposes a polynomial algorithm for inferring a sequence of actions maximizing this lower bound and Section 5 states a condition on the sample sparsity for its optimality. Section 6 illustrates the features of the proposed algorithm and Section 7 discusses its interest, while Section 8 concludes.

2 RELATED WORK

The CGRL algorithm outputs sequences of decisions that, given the prior knowledge it has about its environment in terms of upper bounds on its Lipschitz constants, are likely to drive the agent only towards areas well enough covered by the sample. Heuristic strategies have already been proposed in the RL literature to infer policies that exhibit such a conservative behavior. As a way of example, some of these strategies associate high negative rewards to trajectories falling outside of the well covered areas. The CGRL algorithm can be seen as a min-max approach to solve the generalization task which exploits in a rational way prior knowledge in the form of upper bounds on its Lipschitz constants. Other works in RL have already developped min-max strategies when the environment behavior is partially unknown. However, these strategies usually consider problems with finite state spaces where the uncertainities come from the lack of knowledge of the transition probabilities (Delage and Mannor, 2006; Csáji and Monostori, 2008). In model predictive control (MPC) where the environment is supposed to be fully known (Ernst et al., 2009), min-max approaches have been used to determine the optimal sequence of actions with respect to the "worst case" disturbance sequence occuring (Bemporad and Morari, 1999). The CGRL algorithm relies on a methodology for computing a lower bound on the return in a deterministic setting with a mostly unknown environment. In this, it is related to works in the field of RL which try to get from a sample of trajectories lower bounds on the returns of inferred policies (Mannor et al., 2004; Qian and Murphy, 2009).

3 PROBLEM STATEMENT

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

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

where for all t, the state x_t is an element of the normed vector state space $\mathcal{X}$ and u_t is an element of the finite (discrete) action space $\mathcal{U}$. $T \in \mathbb{N}_0$ is referred to as the *optimization horizon*. An instantaneous reward $r_t = \rho(x_t, u_t) \in \mathbb{R}$ is associated with the action u_t taken while being in state x_t. For every initial state x and for every sequence of actions $(u_0, \ldots, u_{T-1}) \in \mathcal{U}^T$, the cumulated reward over T stages (also named return over T stages) is defined as

$$J^{u_0,\ldots,u_{T-1}}(x) = \sum_{t=0}^{T-1} \rho(x_t, u_t) \ .$$

We assume that the system dynamics f and the reward function ρ are Lipschitz continuous, i.e. that there exist finite constants $L_f, L_\rho \in \mathbb{R}$ such that: $\forall x, x' \in \mathcal{X}, \forall u \in \mathcal{U}$,

$$\begin{aligned} \|f(x,u) - f(x',u)\| &\leq L_f \|x - x'\| \ , \\ |\rho(x,u) - \rho(x',u)| &\leq L_\rho \|x - x'\| \ . \end{aligned}$$

We further suppose that: (i) the system dynamics f and the reward function ρ are unknown, (ii) a set of one-step transitions $\mathcal{F} = \{(x^l, u^l, r^l, y^l)\}_{l=1}^{|\mathcal{F}|}$ is known where each one-step transition is such that $y^l = f(x^l, u^l)$ and $r^l = \rho(x^l, u^l)$, (iii) $\forall a \in \mathcal{U}$, $\exists (x, u, r, y) \in \mathcal{F} : u = a$ (each action $a \in \mathcal{U}$ appears at least once in $\mathcal{F}$) and (iv) two constants L_f and L_ρ satisfying the above-written inequalities are known.[2]

Let $J^*(x) = \max_{(u_0,\ldots,u_{T-1}) \in \mathcal{U}^T} J^{u_0,\ldots,u_{T-1}}(x)$. An optimal sequence of actions $u_0^*(x), \ldots, u_{T-1}^*(x)$ is such that $J^{u_0^*(x),\ldots,u_{T-1}^*(x)}(x) = J^*(x)$. The goal is to compute, for any initial state $x \in \mathcal{X}$, a sequence of actions $(\hat{u}_0^*(x), \ldots, \hat{u}_{T-1}^*(x)) \in \mathcal{U}^T$ such that $J^{\hat{u}_0^*(x),\ldots,\hat{u}_{T-1}^*(x)}$ is as close as possible to $J^*(x)$.

4 LOWER BOUND ON THE RETURN OF A GIVEN SEQUENCE OF ACTIONS

In this section, we present a method for computing, from a given initial state, a dataset of transitions, and weak prior knowledge about the environment, a maximal lower bound on the T-stage return of a given sequence of actions $u_0, \ldots, u_{T-1}$. The method is adapted from (Fonteneau et al., 2009). In the following, we denote by $\mathcal{F}^T_{u_0,\ldots,u_{T-1}}$ the set of all sequences of one-step system transitions $[(x^{l_0}, u^{l_0}, r^{l_0}, y^{l_0}), \ldots, (x^{l_{T-1}}, u^{l_{T-1}}, r^{l_{T-1}}, y^{l_{T-1}})]$ that may be built from elements of $\mathcal{F}$ and that are compatible with $u_0, \ldots, u_{T-1}$, i.e. for which $u^{l_t} = u_t$, $\forall t \in [\![0, T-1]\!]$. First, we compute a lower bound on the return of the sequence $u_0, \ldots, u_{T-1}$ from any given element τ from $\mathcal{F}^T_{u_0,\ldots,u_{T-1}}$. This lower bound $B(\tau, x)$ is made of the sum of the T rewards corresponding to τ ($\sum_{t=0}^{T-1} r^{l_t}$) and T negative terms. Every negative term is associated with a one-step transition. More specifically, the negative term corresponding to the transition $(x^{l_t}, u^{l_t}, r^{l_t}, y^{l_t})$ of τ represents an upper bound on the variation of the cumulated rewards over the remaining time steps that can occur by simulating the system from a state x^{l_t} rather than $y^{l_{t-1}}$ (with $y^{l_{-1}} = x$). By maximizing $B(\tau, x)$ over $\mathcal{F}^T_{u_0,\ldots,u_{T-1}}$, we obtain a maximal lower bound on the return of $u_0, \ldots, u_{T-1}$ whose tightness can be characterized in terms of the sample sparsity.

[2]These constants do not necessarily have to be the smallest ones satisfying these inequalities (i.e., the Lispchitz constants), however, the smaller they are, the higher the lower bound on the return of the policy outputted by the CGRL algorithm will be.

4.1 Computing a Bound from a Given Sequence of One-Step Transitions

We have the following lemma.

Lemma 4.1. *Let $u_0, \ldots, u_{T-1}$ be a sequence of actions. Let $\tau = [(x^{l_t}, u^{l_t}, r^{l_t}, y^{l_t})]_{t=0}^{T-1} \in \mathcal{F}^T_{u_0,\ldots,u_{T-1}}$. Then,*

$$J^{u_0,\ldots,u_{T-1}}(x) \geq B(\tau, x) \, ,$$

with

$$B(\tau, x) \doteq \sum_{t=0}^{T-1} \left[r^{l_t} - L_{Q_{T-t}} \| y^{l_{t-1}} - x^{l_t} \| \right] ,$$
$$y^{l_{-1}} = x \, ,$$
$$L_{Q_{T-t}} = L_\rho \sum_{i=0}^{T-t-1} (L_f)^i \, .$$

The proof is given in Appendix 8.1. The lower bound on $J^{u_0,\ldots,u_{T-1}}(x)$ derived in this lemma can be interpreted as follows. The sum of the rewards of the "broken" trajectory formed by the sequence of one-step system transitions τ can never be greater than $J^{u_0,\ldots,u_{T-1}}(x)$, provided that every reward r^{l_t} is penalized by a factor $L_{Q_{T-t}} \| y^{l_{t-1}} - x^{l_t} \|$. This factor is in fact an upper bound on the variation of the $(T-t)$-state-action value function (see Appendix 8.1) that can occur when "jumping" from $(y^{l_{t-1}}, u_t)$ to (x^{l_t}, u_t). An illustration of this is given in Figure 1.

4.2 Tightness of Highest Lower Bound Over all Compatible Sequences of One-Step Transitions

We define

$$B^{u_0,\ldots,u_{T-1}}(x) = \max_{\tau \in \mathcal{F}^T_{u_0,\ldots,u_{T-1}}} B(\tau, x)$$

and we analyze in this subsection the tightness of the lower bound $B^{u_0,\ldots,u_{T-1}}(x)$ as a function of the sample sparsity. The sample sparsity is defined as follows: let $\mathcal{F}_a = \{(x^l, u^l, r^l, y) \in \mathcal{F} | u^l = a\}$ ($\forall a$, $\mathcal{F}_a \neq \emptyset$ according to assumption (iii) given in Section 3) and let us suppose that $\exists\, \alpha \in \mathbb{R}^+$:

$$\forall a \in \mathcal{U}\, , \sup_{x' \in \mathcal{X}} \Big\{ \min_{(x^l, u^l, r^l, y^l) \in \mathcal{F}_a} \| x^l - x' \| \Big\} \leq \alpha \, . \qquad (1)$$

The smallest α which satisfies equation (1) is named the sample sparsity and is denoted by α^* (the existence of α and α^* implies that $\mathcal{X}$ is bounded). We have the following theorem.

Theorem 4.2 (Tightness of highest lower bound). $\exists\, C > 0 : \forall (u_0, \ldots, u_{T-1}) \in \mathcal{U}^T$,

$$J^{u_0,\ldots,u_{T-1}}(x) - B^{u_0,\ldots,u_{T-1}}(x) \leq C\alpha^* .$$

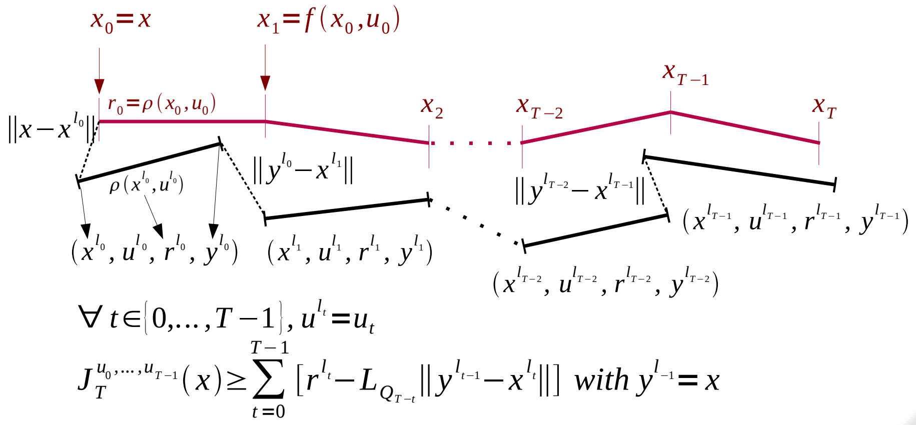

$x_0 = x$
$x_1 = f(x_0, u_0)$
$r_0 = \rho(x_0, u_0)$
x_2
x_{T-2}
x_{T-1}
x_T
$\|x - x^{l_0}\|$
$\rho(x^{l_0}, u^{l_0})$
$\|y^{l_0} - x^{l_1}\|$
$\|y^{l_{T-2}} - x^{l_{T-1}}\|$
$(x^{l_0}, u^{l_0}, r^{l_0}, y^{l_0})$
$(x^{l_1}, u^{l_1}, r^{l_1}, y^{l_1})$
$(x^{l_{T-2}}, u^{l_{T-2}}, r^{l_{T-2}}, y^{l_{T-2}})$
$(x^{l_{T-1}}, u^{l_{T-1}}, r^{l_{T-1}}, y^{l_{T-1}})$
$\forall t \in \{0, \ldots, T-1\}, u^{l_t} = u_t$
$J_T^{u_0, \ldots, u_{T-1}}(x) \geq \sum_{t=0}^{T-1} [r^{l_t} - L_{Q_{T-t}} \|y^{l_{t-1}} - x^{l_t}\|]$ with $y^{l_{-1}} = x$

Figure 1: A graphical interpretation of the different terms composing the bound on $J^{u_0,\ldots,u_{T-1}}(x)$ computed from a sequence of one-step transitions.

The proof of Theorem 4.2 is given in Appendix 8.2. The lower bound $B^{u_0,\ldots,u_{T-1}}(x)$ thus converges to the T-stage return of the sequence of actions $u_0, \ldots, u_{T-1}$ when the sample sparsity α^* decreases to zero.

5 COMPUTING A SEQUENCE OF ACTIONS MAXIMIZING THE HIGHEST LOWER BOUND

Let $\mathfrak{B}^*(x) = \{(u_0, \ldots, u_{T-1}) \in \mathcal{U}^T | B^{u_0,\ldots,u_{T-1}}(x) = \max_{(u'_0,\ldots,u'_{T-1}) \in \mathcal{U}^T} B^{u'_0,\ldots,u'_{T-1}}(x)\}$. The CGRL algorithm computes for each initial state x a sequence of actions $\hat{u}^*_0(x), \ldots, \hat{u}^*_{T-1}(x)$ that belongs to $\mathfrak{B}^*(x)$. From what precedes, it follows that the actual return $J^{\hat{u}^*_0(x),\ldots,\hat{u}^*_{T-1}(x)}(x)$ of this sequence is lower-bounded by $\max_{(u_0,\ldots,u_{T-1}) \in \mathcal{U}^T} B^{u_0,\ldots,u_{T-1}}(x)$. Due to the tightness of the lower bound $B^{u_0,\ldots,u_{T-1}}(x)$, the value of the return which is guaranteed will converge to the true return of the sequence of actions when α^* decreases to zero. Additionaly, we prove in Section 5.1 that when the sample sparsity α^* decreases below a particular threshold, the sequence $\hat{u}^*_0(x), \ldots, \hat{u}^*_{T-1}(x)$ is optimal. To identify a sequence of actions that belongs to $\mathfrak{B}^*(x)$ without computing for all sequences $u_0, \ldots, u_{T-1}$ the value $B^{u_0,\ldots,u_{T-1}}(x)$, the CGRL algorithm exploits the fact that the problem of finding an element of $\mathfrak{B}^*(x)$ can be reformulated as a shortest path problem.

5.1 Convergence of $\hat{\mathbf{u}}^*_0(\mathbf{x}), \ldots, \hat{\mathbf{u}}^*_{\mathbf{T-1}}(\mathbf{x})$ Towards an Optimal Sequence of Actions

We prove hereafter that when α^* gets lower than a particular threshold, the CGRL algorithm can only output optimal policies.

Theorem 5.1 (Convergence of CGRL algorithm). *Let*

$$\mathfrak{J}^*(x) = \{(u_0, \ldots, u_{T-1}) \in \mathcal{U}^T | J^{u_0,\ldots,u_{T-1}}(x) = J^*(x)\},$$

and let us suppose that $\mathfrak{J}^*(x) \neq \mathcal{U}^T$ (if $\mathfrak{J}^*(x) = \mathcal{U}^T$, the search for an optimal sequence of actions is indeed trivial). We define

$$\varepsilon(x) = \min_{u_0,\ldots,u_{T-1} \in \mathcal{U}^T \setminus \mathfrak{J}^*(x)} \{J^*(x) - J^{u_0,\ldots,u_{T-1}}(x)\}.$$

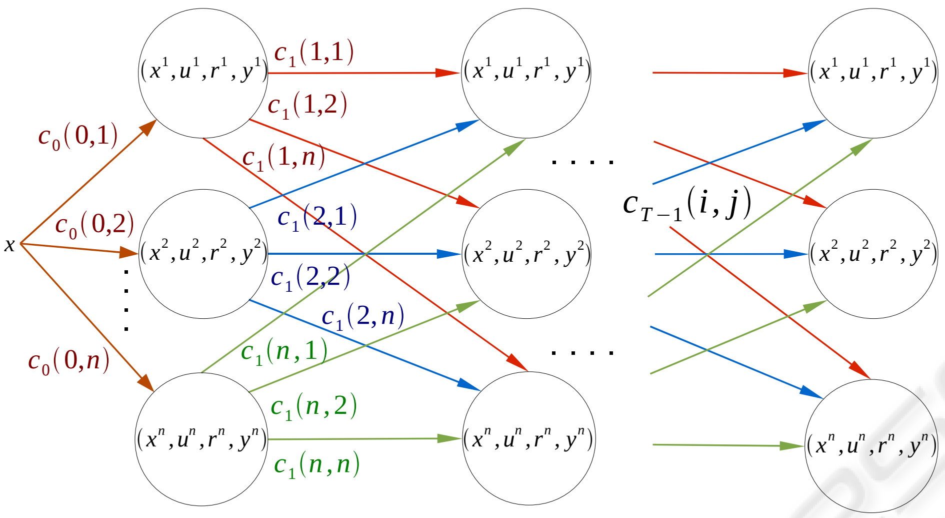
Then

$$C\alpha^* < \varepsilon(x) \implies (\hat{u}^*_0(x), \ldots, \hat{u}^*_{T-1}(x)) \in \mathfrak{J}^*(x).$$

The proof of Theorem 5.1 is given in Appendix 8.3.

5.2 Cautious Generalization Reinforcement Learning Algorithm

The CGRL algorithm computes an element of the set $\mathfrak{B}^*(x)$ defined previously. Let $\mathcal{D} : \mathcal{F}^T \to \mathcal{U}^T$ be the operator that maps a sequence of one-step system transitions $\tau = [(x^{l_t}, u^{l_t}, r^{l_t}, y^{l_t})]_{t=0}^{T-1}$ into the sequence of actions $u^{l_0}, \ldots, u^{l_{T-1}}$. Using this operator, we can write $\mathfrak{B}^*(x) = \{(u_0, \ldots, u_{T-1}) \in \mathcal{U}^T | \exists \tau \in \arg\max_{\tau \in \mathcal{F}^T} B(\tau, x)$ for which $\mathcal{D}(\tau) = (u_0, \ldots, u_{T-1})\}$. Or,



$$l_0^*,\ldots,l_{T-1}^* \in \underset{l_0,\ldots,l_{T-1}}{argmax}\ c_0(0,l_0)+c_1(l_0,l_1)+\ldots+c_{T-1}(l_{T-2},l_{T-1})$$

$$with\ c_t(i,j)=-L_{Q_{T-t}}\|y^i-x^j\|+r^j\ ,\ y^0=x \longrightarrow \hat{u}_0^*(x),\ldots,\hat{u}_{T-1}^*(x)=u^{l_0^*},\ldots,u^{l_{T-1}^*}$$

Figure 2: A graphical interpretation of the CGRL algorithm (notice that $n = |\mathcal{F}|$).

equivalently

$$\mathfrak{B}^*(x) = \Big\{(u_0,\ldots,u_{T-1}) \in \mathcal{U}^T \,|$$
$$\exists \tau \in \arg\max_{\tau\in\mathcal{F}^T} \sum_{t=0}^{T-1} \left[r^{l_t} - L_{Q_{T-t}}\|y^{l_{t-1}} - x^{l_t}\|\right] \text{ for which}$$
$$\mathcal{D}(\tau) = (u_0,\ldots,u_{T-1})\Big\} .$$

From this expression, we can notice that a sequence of one-step transitions τ such that $\mathcal{D}(\tau)$ belongs to $\mathfrak{B}^*(x)$ can be obtained by solving a shortest path problem on the graph given in Figure 2. The CGRL algorithm works by solving this problem using the Viterbi algorithm and by applying the operator $\mathcal{D}$ to the sequence of one-step transitions τ corresponding to its solution. Its complexity is quadratic with respect to the cardinality of the input sample $\mathcal{F}$ and linear with respect to the optimization horizon T.

6 ILLUSTRATION

In this section, we illustrate the CGRL algorithm on a variant of the puddle world benchmark introduced in (Sutton, 1996). In this benchmark, a robot whose goal is to collect high cumulated rewards navigates on a plane. A puddle stands in between the initial position of the robot and the high reward area. If the robot is in the puddle, it gets highly negative rewards. An optimal navigation strategy drives the robot around the puddle to reach the high reward area. Two datasets of one-step transitions have been used in our example. The first set $\mathcal{F}_1$ contains elements that uniformly cover the area of the state space that can be reached within T steps. The set $\mathcal{F}_2$ has been obtained by removing from $\mathcal{F}_1$ the elements corresponding to the highly negative rewards.[3] The full specification of the benchmark and the exact procedure for generating $\mathcal{F}_1$ and $\mathcal{F}_2$ are given in Appendix 8.4. On Figure 3, we have drawn the trajectory of the robot when following the sequence of actions computed by the CGRL algorithm. Every state encountered is represented by a white square. The plane upon which the robot navigates has been colored such that the darker the area, the smaller the corresponding rewards are. In particular, the puddle area is colored in dark grey/black. We see that the CGRL policy drives the robot around the puddle to reach the high-reward area – which is represented by the light-grey circles. The CGRL algorithm also computes a lower-bound on the cumulated rewards obtained by this action sequence. Here, we found out that this lower bound was rather conservative.

[3]Although this problem might be treated by on-line learning methods, in some settings - for whatever reason - on-line learning may be impractical and all one will have is a batch of trajectories

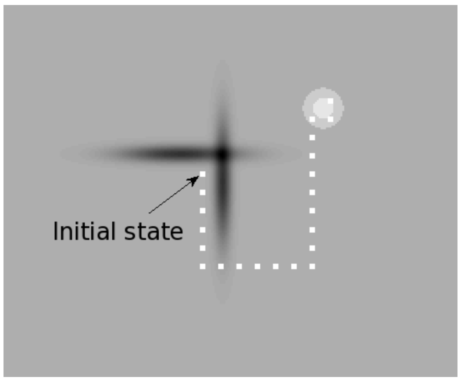


Figure 3: CGRL with $\mathcal{F}_1$.

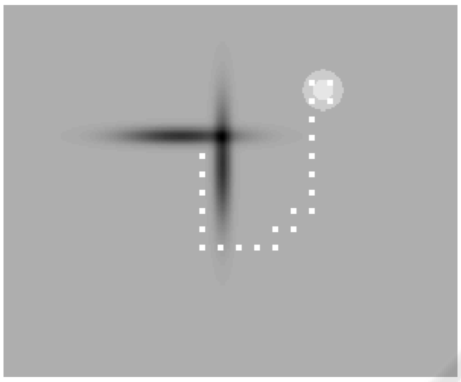


Figure 4: FQI with $\mathcal{F}_1$.

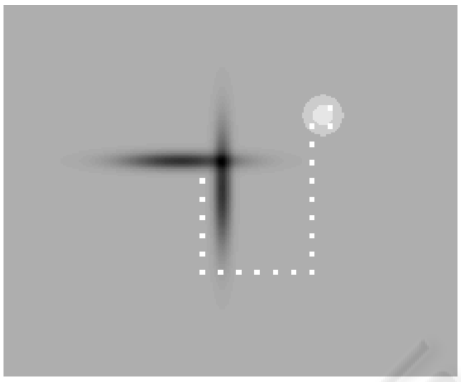


Figure 5: CGRL with $\mathcal{F}_2$.

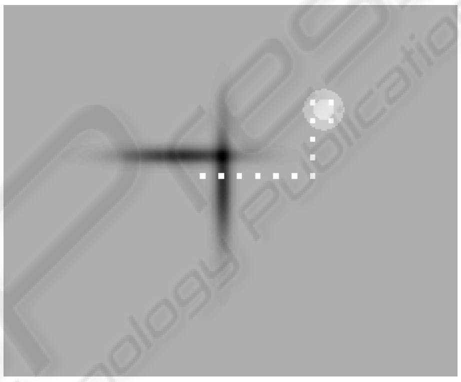


Figure 6: FQI with $\mathcal{F}_2$.

Figure 4 represents the policy inferred from $\mathcal{F}_1$ by using the (finite-time version of the) Fitted Q Iteration algorithm (FQI) combined with extremely randomized trees as function approximators (Ernst et al., 2005). The trajectories computed by the CGRL and FQI algorithms are very similar and so are the sums of rewards obtained by following these two trajectories. However, by using $\mathcal{F}_2$ rather that $\mathcal{F}_1$, the CGRL and FQI algorithms do not lead to similar trajectories, as it is shown on Figures 5 and 6. Indeed, while the CGRL policy still drives the robot around the puddle to reach the high reward area, the FQI policy makes the robot cross the puddle. In terms of optimality, this latter navigation strategy is much worse. The difference between both navigation strategies can be explained as follows. The FQI algorithm behaves as if it were associating to areas of the state space that are not covered by the input sample, the properties of the elements of this sample that are located in the neighborhood of these areas. This in turn explains why it computes a policy that makes the robot cross the puddle. The same behavior could probably be observed by using other algorithms that combine dynamic programming strategies with kernel-based approximators or averagers (Boyan and Moore, 1995; Gordon, 1999; Ormoneit and Sen, 2002). The CGRL algorithm generalizes the information contained in the dataset, by assuming, given the intial state, the most adverse behavior for the environment according to its weak prior knowledge about the environment. This results in the fact that the CGRL algorithm penalizes sequences of decisions that could drive the robot in areas not well covered by the sample, and this explains why the CGRL algorithm drives the robot around the puddle when run with $\mathcal{F}_2$.

7 DISCUSSION

The CGRL algorithm outputs a sequence of actions as well as a lower bound on its return. When $L_f > 1$ (e.g. when the system is unstable), this lower bound will decrease exponentially with T. This may lead to very low performance guarantees when the optimization horizon T is large. However, one can also observe that the terms $L_{Q_{T-t}}$ – which are responsible for the exponential decrease of the lower bound with the optimization horizon – are multiplied by the distance between the end state of a one-step transition and the beginning state of the next one-step transition of the sequence τ ($\|y^{l^*_{t-1}} - x^{l^*_t}\|$) solution of the shortest path problem of Figure 2. Therefore, if these states $y^{l^*_{t-1}}$ and $x^{l^*_t}$ are close to each other, the CGRL algorithm can lead to good performance guarantees even for large values of T. It is also important to notice that this lower bound does not depend explicitly on the sample sparsity α^*, but depends rather on the initial state for which the sequence of actions is computed. Therefore, this may lead to cases where the CGRL algorithm provides good performance guarantees for some specific initial states, even if the sample does not cover every area of the state space well enough.

Other RL algorithms working in a similar setting as the CGRL algorithm, while not exploiting the weak prior knowledge about the environment, do not output a lower bound on the return of the policy h they infer from the sample of trajectories $\mathcal{F}$. However, some lower bounds on the return of h can still be computed. For instance, this can be done by exploiting the results of (Fonteneau et al., 2009) upon which the CGRL algorithm is based. However, one can show that following the strategy described in (Fonteneau et al., 2009) would necessarily lead to a bound lower than the lower bound associated to the sequence of actions computed by the CGRL algorithm. Another strategy would be to design global lower bounds on their policy by adapting proofs used to establish the consistency of these algorithms. As a way of example, by proceeding like this, we can design a lower-bound on the return of the policy given by the FQI algorithm when combined with some specific approximators which have, among others, Lipschitz continuity properties. These algorithms compute a sequence of state-action value functions $\hat{Q}_1, \hat{Q}_2, \ldots, \hat{Q}_T$ and compute the policy $h : \{0, 1, \ldots, T-1\} \times X$ defined as follows : $h(t, x_t) \in \underset{u \in \mathcal{U}}{\arg\max} \hat{Q}_{T-t}(x_t, u)$. For instance when using kernel-based approximators (Ormoneit and Sen, 2002), we have as result that the return of h when starting from a state x is larger than $\hat{Q}_T(x, h(0, x)) - (C_1 T + C_2 T^2) \cdot \alpha^*$ where C_1 and C_2 depends on L_f, L_ρ, the Lipschtiz constants of the class of approximation and an upper bound on ρ. The explicit dependence of this lower bound on α^* as well as the large values of C_1 and C_2 tend to lead to a very conservative lower bound, especially when $\mathcal{F}$ is sparse.

8 CONCLUSIONS

We have proposed a new strategy for RL using a batch of system transitions and some weak prior knowledge about the environment behavior. It consists in maximizing lower bounds on the return that may be inferred by combining information from a dataset of observed system transitions and upper bounds on the Lipschitz constants of the environment. The proposed algorithm is of polynomial complexity and avoids regions of the state space where the sample density is too low according to the prior information. A simple example has illustrated that this strategy can lead to cautious policies where other batch-mode RL algorithms fail because they unsafely generalize the information contained in the dataset.

From the results given in (Fonteneau et al., 2009), it is also possible to derive in a similar way upper bounds on the return of a policy. In this respect, it would also be possible to adopt an optimistic generalization strategy by inferring policies that maximize these upper bounds. We believe that exploiting together the policy based on a cautious generalization strategy and the one based on an optimistic generalization strategy could offer interesting possibilities for addressing the exploitation-exploration tradeoff faced when designing intelligent agents. For example, if the policies coincide, it could be an indication that further exploration is not needed.

When using batch mode reinforcement learning algorithms to design autonomous intelligent agents, a problem arises. After a long enough time of interaction with their environment, the sample the agents collect may become so large that batch mode RL-techniques may become computationally impractical, even with small degree polynomial algorithms. As suggested by (Ernst, 2005), a solution for addressing this problem would be to retain only the most "informative samples". In the context of the proposed algorithm, the complexity for computing the optimal sequence of decisions is quadratic in the size of the dataset. We believe that it would be interesting to design lower complexity algorithms based on subsampling the dataset based on the initial state information.

Finally, while in this paper we have considered deterministic environments and open-loop strategies for interacting with them, we believe that extending our framework to closed-loop strategies (revising at each time step the first stage decision in a receding horizon approach) and study their performances, in particular in the context of stochastic environments, is a very promising direction of further research.

ACKNOWLEDGEMENTS

Raphael Fonteneau acknowledges the financial support of the FRIA (Fund for Research in Industry and Agriculture). Damien Ernst is a research associate of the FRS-FNRS. This paper presents research results of the Belgian Network BIOMAGNET (Bioinformatics and Modeling: from Genomes to Networks), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office.We also acknowledge financial support from NIH grants P50 DA10075 and R01 MH080015. The scientific responsibility rests with its authors.

REFERENCES

Bemporad, A. and Morari, M. (1999). Robust model predictive control: A survey. *Robustness in Identification and Control*, 245:207–226.

Bertsekas, D. and Tsitsiklis, J. (1996). *Neuro-Dynamic Programming*. Athena Scientific.

Boyan, J. and Moore, A. (1995). Generalization in reinforcement learning: Safely approximating the value function. In *Advances in Neural Information Processing Systems 7*, pages 369–376. MIT Press.

Csáji, B. C. and Monostori, L. (2008). Value function based reinforcement learning in changing markovian environments. *J. Mach. Learn. Res.*, 9:1679–1709.

Delage, E. and Mannor, S. (2006). Percentile optimization for Markov decision processes with parameter uncertainty. *Operations Research*.

Ernst, D. (2005). Selecting concise sets of samples for a reinforcement learning agent. In *Proceedings of the Third International Conference on Computational Intelligence, Robotics and Autonomous Systems (CIRAS 2005)*, page 6.

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

Ernst, D., Glavic, M., Capitanescu, F., and Wehenkel, L. (April 2009). Reinforcement learning versus model predictive control: a comparison on a power system problem. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 39:517–529.

Fonteneau, R., Murphy, S., Wehenkel, L., and Ernst, D. (2009). Inferring bounds on the performance of a control policy from a sample of trajectories. In *Proceedings of the 2009 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (IEEE ADPRL 09)*, Nashville, TN, USA.

Gordon, G. (1999). *Approximate Solutions to Markov Decision Processes*. PhD thesis, Carnegie Mellon University.

Ingersoll, J. (1987). *Theory of Financial Decision Making*. Rowman and Littlefield Publishers, Inc.

Lagoudakis, M. and Parr, R. (2003). Least-squares policy iteration. *Jounal of Machine Learning Research*, 4:1107–1149.

Mannor, S., Simester, D., Sun, P., and Tsitsiklis, J. (2004). Bias and variance in value function estimation. In *Proceedings of the 21st International Conference on Machine Learning*.

Murphy, S. (2003). Optimal dynamic treatment regimes. *Journal of the Royal Statistical Society, Series B*, 65(2):331–366.

Murphy, S. (2005). An experimental design for the development of adaptive treatment strategies. *Statistics in Medicine*, 24:1455–1481.

Ormoneit, D. and Sen, S. (2002). Kernel-based reinforcement learning. *Machine Learning*, 49(2-3):161–178.

Qian, M. and Murphy, S. (2009). Performance guarantee for individualized treatment rules. *Submitted*.

Riedmiller, M. (2005). Neural fitted Q iteration - first experiences with a data efficient neural reinforcement learning method. In *Proceedings of the Sixteenth European Conference on Machine Learning (ECML 2005)*, pages 317–328.

Sutton, R. (1996). Generalization in reinforcement learning: Successful examples using sparse coding. In *Advance in Neural Information Processing Systems 8*, pages 1038–1044. MIT Press.

Sutton, R. and Barto, A. (1998). *Reinforcement Learning*. MIT Press.

APPENDIX

8.1 Proof of Lemma 4.1

Before proving Lemma 4.1 in Section 8.1.2, we prove in Section 8.1.1 a preliminary result related to the Lipschitz continuity of state-action value functions.

8.1.1 Lipschitz Continuity of the N-stage State-action Value Functions

For $N = 1, \dots, T$, let us define the family of state-action value functions $Q_N^{u_0,\dots,u_{T-1}} : X \times U \to \mathbb{R}$ as follows:

$$Q_N^{u_0,\dots,u_{T-1}}(x,u) = \rho(x,u) + \sum_{t=T-N+1}^{T-1} \rho(x_t,u_t),$$

where $x_{T-N+1} = f(x,u)$. $Q_N^{u_0,\dots,u_{T-1}}(x,u)$ gives the sum of rewards from instant $t = T-N$ to instant $T-1$ when (i) the system is in state x at instant $T-N$, (ii) the action chosen at instant $T-N$ is u and (iii) the actions chosen at instants $t > T-N$ are u_t. The function $J^{u_0,\dots,u_{T-1}}$ can be deduced from $Q_N^{u_0,\dots,u_{T-1}}$ as follows:

$$\forall x \in X,\ J^{u_0,\dots,u_{T-1}}(x) = Q_T^{u_0,\dots,u_{T-1}}(x,u_0). \qquad (2)$$

We also have $\forall x \in X, \forall u \in U$,

$$Q_{N+1}^{u_0,\ldots,u_{T-1}}(x,u) = \rho(x,u) + Q_N^{u_0,\ldots,u_{T-1}}(f(x,u),u_{T-N}) \quad (3)$$

Lemma [Lipschitz continuity of $Q_N^{u_0,\ldots,u_{T-1}}$] $\forall N \in \{1,\ldots,T\}, \forall x,x' \in X, \forall u \in U$,

$$|Q_N^{u_0,\ldots,u_{T-1}}(x,u) - Q_N^{u_0,\ldots,u_{T-1}}(x',u)| \le L_{Q_N}\|x-x'\|,$$

with $L_{Q_N} = L_\rho \sum_{t=0}^{N-1} L_f^t$.

Proof We consider the statement $\mathcal{H}(N)$: $\forall x,x' \in X, \forall u,u' \in U$,

$$|Q_N^{u_0,\ldots,u_{T-1}}(x,u) - Q_N^{u_0,\ldots,u_{T-1}}(x',u)| \le L_{Q_N}\|x-x'\|.$$

We prove by induction that $\mathcal{H}(N)$ is true $\forall N \in \{1,\ldots,T\}$. For the sake of clarity, we denote $|Q_N^{u_0,\ldots,u_{T-1}}(x,u) - Q_N^{u_0,\ldots,u_{T-1}}(x',u)|$ by Δ_N.

Basis $(N=1)$: We have $\Delta_N = |\rho(x,u) - \rho(x',u)|$, and the Lipschitz continuity of ρ allows to write $\Delta_N \le L_\rho\|x-x'\|$. This proves $\mathcal{H}(1)$.

Induction step: We suppose that $\mathcal{H}(N)$ is true, $1 \le N \le T-1$. Using equation (3), we can write $\Delta_{N+1} = \left|Q_{N+1}^{u_0,\ldots,u_{T-1}}(x,u) - Q_{N+1}^{u_0,\ldots,u_{T-1}}(x',u)\right|$ $= \left|\rho(x,u) - \rho(x',u) + Q_N^{u_0,\ldots,u_{T-1}}(f(x,u),u_{T-N}) - Q_N^{u_0,\ldots,u_{T-1}}(f(x',u),u_{T-N})\right|$ and, from there, $\Delta_{N+1} \le \left|\rho(x,u) - \rho(x',u)\right| + \left|Q_N^{u_0,\ldots,u_{T-1}}(f(x,u),u_{T-N}) - Q_N^{u_0,\ldots,u_{T-1}}(f(x',u),u_{T-N})\right|$.
$\mathcal{H}(N)$ and the Lipschitz continuity of ρ give

$$\Delta_{N+1} \le L_\rho\|x-x'\| + L_{Q_N}\|f(x,u) - f(x',u)\|.$$

The Lipschitz continuity of f gives

$$\Delta_{N+1} \le L_\rho\|x-x'\| + L_{Q_N}L_f\|x-x'\|,$$

then $\Delta_{N+1} \le L_{Q_{N+1}}\|x-x'\|$ since $L_{Q_{N+1}} = L_\rho + L_{Q_N}L_f$. This proves $\mathcal{H}(N+1)$ and ends the proof.

8.1.2 Proof of Lemma 4.1

By assumption we have $u^{l_0} = u_0$, then we use equation (2) and the Lipschitz continuity of $Q_T^{u_0,\ldots,u_{T-1}}$ to write

$$|J^{u_0,\ldots,u_{T-1}}(x) - Q_T^{u_0,\ldots,u_{T-1}}(x^{l_0},u_0)| \le L_{Q_T}\|x-x^{l_0}\|.$$

It follows that

$$Q_T^{u_0,\ldots,u_{T-1}}(x^{l_0},u_0) - L_{Q_T}\|x-x^{l_0}\| \le J^{u_0,\ldots,u_{T-1}}(x).$$

According to equation (3), we have $Q_T^{u_0,\ldots,u_{T-1}}(x^{l_0},u_0) = \rho(x^{l_0},u_0) + Q_{T-1}^{u_0,\ldots,u_{T-1}}(f(x^{l_0},u_0),u_1)$ and from there

$$Q_T^{u_0,\ldots,u_{T-1}}(x^{l_0},u_0) = r^{l_0} + Q_{T-1}^h(y^{l_0},u_1).$$

Thus,

$$Q_{T-1}^{u_0,\ldots,u_{T-1}}(y^{l_0},u_1) + r^{l_0} - L_{Q_T}\|x-x^{l_0}\| \le J_T^{u_0,\ldots,u_{T-1}}(x).$$

The Lipschitz continuity of $Q_{T-1}^{u_0,\ldots,u_{T-1}}$ with $u_1 = u^{l_1}$ gives

$$\begin{aligned}|Q_{T-1}^{u_0,\ldots,u_{T-1}}(y^{l_0},u_1) &- Q_{T-1}^{u_0,\ldots,u_{T-1}}(x^{l_1},u^{l_1})| \\ &\le L_{Q_{T-1}}\|y^{l_0} - x^{l_1}\|.\end{aligned}$$

This implies that

$$\begin{aligned}Q_{T-1}^{u_0,\ldots,u_{T-1}}(x^{l_1},u_1) &- L_{Q_{T-1}}\|y^{l_0} - x^{l_1}\| \\ &\le Q_{T-1}^{u_0,\ldots,u_{T-1}}(y^{l_0},u_1).\end{aligned}$$

We have therefore

$$\begin{aligned}Q_{T-1}^{u_0,\ldots,u_{T-1}}(x^{l_1},u_1) + r^{l_0} &- L_{Q_T}\|x-x^{l_0}\| \\ -L_{Q_{T-1}}\|y^{l_0} - x^{l_1}\| &\le J^{u_0,\ldots,u_{T-1}}(x).\end{aligned}$$

The proof is completed by developing this iteration.

8.2 Proof of Theorem 4.2

Let $(x_0,u_0,r_0,x_1,u_1,\ldots,x_{T-1},u_{T-1},r_{T-1},x_T)$ be the trajectory of an agent starting from $x_0 = x$ when following the open-loop policy $u_0,\ldots,u_{T-1}$. Using equation (1), we define $\tau = [(x^{l_t},u^{l_t},r^{l_t},y^{l_t})]_{t=0}^{T-1} \in \mathcal{F}_{u_0,\ldots,u_{T-1}}^T$ that satisfies $\forall t \in \{0,1,\ldots,T-1\}$

$$\|x^{l_t} - x_t\| = \min_{l \in \{1,\ldots,|\mathcal{F}|\}} \|x^l - x_t\| \le \alpha^*. \quad (4)$$

We have $B(\tau,x) = \sum_{t=0}^{T-1}[r^{l_t} - L_{Q_{T-t}}\|y^{l_{t-1}} - x^{l_t}\|]$ with $y^{l_{-1}} = x$. Let us focus on $\|y^{l_{t-1}} - x^{l_t}\|$. We have $\|y^{l_{t-1}} - x^{l_t}\| = \|x^{l_t} - x_t + x_t - y^{l_{t-1}}\|$, and hence $\|y^{l_{t-1}} - x^{l_t}\| \le \|x^{l_t} - x_t\| + \|x_t - y^{l_{t-1}}\|$. Using inequality (4), we can write

$$\|y^{l_{t-1}} - x^{l_t}\| \le \alpha^* + \|x_t - y^{l_{t-1}}\|. \quad (5)$$

For $t=0$, one has $\|x_t - y^{l_{t-1}}\| = \|x_0 - x_0\| = 0$. For $t > 0$ $\|x_t - y^{l_{t-1}}\| = \|f(x_{t-1},u_{t-1}) - f(x^{l_{t-1}},u_{t-1})\|$ and the Lipschitz continuity of f implies that $\|x_t - y^{l_{t-1}}\| \le L_f\|x_{t-1} - x^{l_{t-1}}\|$. So, as $\|x_{t-1} - x^{l_{t-1}}\| \le \alpha^*$, we have

$$\forall t > 0,\ \|x_t - y^{l_{t-1}}\| \le L_f\alpha^*. \quad (6)$$

Equations (5) and (6) imply that for $t > 0$, $\|y^{l_{t-1}} - x^{l_t}\| \le \alpha^*(1+L_f)$ and, for $t=0$, $\|y^{l_{-1}} - x^{l_0}\| \le \alpha^* \le \alpha^*(1+L_f)$. This gives

$$B(\tau,x) \ge \sum_{t=0}^{T-1}[r^{l_t} - L_{Q_{T-t}}\alpha^*(1+L_f)] \doteq B.$$

We also have, by definition of $B^{u_0,\ldots,u_{T-1}}(x)$,

$$J^{u_0,\ldots,u_{T-1}}(x) \ge B^{u_0,\ldots,u_{T-1}}(x) \ge B(\tau,x) \ge B.$$

Thus,
$|J^{u_0,\ldots,u_{T-1}}(x) - B^{u_0,\ldots,u_{T-1}}(x)| \leq |J^{u_0,\ldots,u_{T-1}}(x) - B|$
$= J^{u_0,\ldots,u_{T-1}}(x) - B$
$= |\sum_{t=0}^{T-1}[(r_t - r^{l_t}) + L_{Q_{T-t}}\alpha^*(1+L_f)]|$
$\leq \sum_{t=0}^{T-1}[|r_t - r^{l_t}| + L_{Q_{T-t}}\alpha^*(1+L_f)]$.
The Lipschitz continuity of ρ allows to write

$$|r_t - r^{l_t}| = |\rho(x_t, u_t) - \rho(x^{l_t}, u_t)| \leq L_\rho \|x_t - x^{l_t}\|,$$

and using inequality (4), we have $|r_t - r^{l_t}| \leq L_\rho \alpha^*$. Finally, we obtain
$J^{u_0,\ldots,u_{T-1}}(x) - B \leq \sum_{t=0}^{T-1}[L_\rho \alpha^* + L_{Q_{T-t}}\alpha^*(1+L_f)]$
$\leq T L_\rho \alpha^* + \sum_{t=0}^{T-1} L_{Q_{T-t}}\alpha^*(1+L_f)$
$\leq \alpha^*\big(T L_\rho + \sum_{t=0}^{T-1} L_{Q_{T-t}}(1+L_f)\big)$.
Thus

$$\begin{aligned} J^{u_0,\ldots,u_{T-1}}(x) &- B^{u_0,\ldots,u_{T-1}}(x) \\ &\leq \Big(T L_\rho + (1+L_f)\sum_{t=0}^{T-1} L_{Q_{T-t}}\Big)\alpha^*, \end{aligned}$$

which completes the proof.

8.3 Proof of Theorem 5.1

Let us prove that by *Reductio ad absurdum*. Let us suppose that the algorithm does not return an optimal sequence of actions, which means that

$$J^{\hat{u}_0^*(x),\ldots,\hat{u}_{T-1}^*(x)}(x) \leq J^*(x) - \varepsilon(x) \ .$$

Let us consider a sequence $u_0^*(x), \ldots, u_{T-1}^*(x) \in \mathfrak{J}^*(x)$. Then $J^{u_0^*(x),\ldots,u_{T-1}^*(x)}(x) = J^*(x)$. The lower bound $B^{u_0^*(x),\ldots,u_{T-1}^*(x)}(x)$ satisfies the relationship

$$J^*(x) - B^{u_0^*(x),\ldots,u_{T-1}^*(x)}(x) \leq C\alpha.$$

Knowing that $C\alpha < \varepsilon(x)$, we have

$$B^{u_0^*(x),\ldots,u_{T-1}^*(x)}(x) > J^*(x) - \varepsilon.$$

By definition of ε,

$$J_T^*(x) - \varepsilon \geq J^{\hat{u}_0^*(x),\ldots,\hat{u}_{T-1}^*(x)}(x),$$

and since

$$J^{\hat{u}_0^*(x),\ldots,\hat{u}_{T-1}^*(x)}(x) \geq B^{\hat{u}_0^*(x),\ldots,\hat{u}_{T-1}^*(x)}(x),$$

we have

$$B^{u_0^*(x),\ldots,u_{T-1}^*(x)}(x) > B^{\hat{u}_0^*(x),\ldots,\hat{u}_{T-1}^*(x)}(x) \ ,$$

which contradicts the fact that the algorithm returns the sequence that leads to the highest lower bound. This ends the proof.

8.4 Experimental Specifications

The puddle world benchmark is defined by

$$\begin{aligned} &\mathcal{X} = \mathbb{R}^2, \\ &\mathcal{U} = \{(0.1 \quad 0), (-0.1 \quad 0), (0 \quad 0.1), (0 \quad -0.1)\}, \\ &f(x,u) = x + u, \\ &\rho(x,u) = k_1 \mathcal{N}_{\mu_1,\Sigma_1}(x) - k_2 \mathcal{N}_{\mu_2,\Sigma_2}(x) - k_3 \mathcal{N}_{\mu_3,\Sigma_3}(x) \ , \end{aligned}$$

with

$$\begin{aligned} &\mathcal{N}_{\mu,\Sigma}(x) = \frac{1}{2\pi\sqrt{|\Sigma|}} e^{\frac{-(x-\mu)\Sigma^{-1}(x-\mu)'}{2}}, \\ &\mu_1 = (1 \quad 1), \\ &\mu_2 = (0.225 \quad 0.75), \\ &\mu_3 = (0.45 \quad 0.6), \\ &\Sigma_1 = \begin{pmatrix} 0.005 & 0 \\ 0 & 0.005 \end{pmatrix}, \\ &\Sigma_2 = \begin{pmatrix} 0.05 & 0 \\ 0 & 0.001 \end{pmatrix}, \\ &\Sigma_3 = \begin{pmatrix} 0.001 & 0 \\ 0 & 0.05 \end{pmatrix} \end{aligned}$$

and $k_1 = 1, k_2 = k_3 = 20$. The Euclidian norm is used. $L_f = 1$, $L_\rho = 1.3742 * 10^6$, $T = 25$, initial state $x = (0.35, 0.65)$. The sets of one-step system transitions are
$\mathcal{F}_1 = \{(x, u, \rho(x,u), f(x,u)) | x = (-2.15 + i * 5/203, -1.85 + j * 5/203), i, j = 1 : 203\}$,
$\mathcal{F}_2 = \mathcal{F}_1 \backslash \{(x,u,r,y) \in \mathcal{F}_1 | x \in [0.4, 0.5] \times [0.25, 0.95] \cup [-0.1, 0.6] \times [0.7, 0.8]\}$.

The FQI algorithm combined with extremely randomized trees is run using its default parameters given in (Ernst et al., 2005).