4

Uncertainty in Reinforcement Learning — Awareness, Quantisation, and Control

Daniel Schneegass, Alexander Hans, and Steffen Udluft
Siemens AG, Corporate Research & Technologies, Intelligent Systems & Control
Germany

1. Introduction

Reinforcement learning (RL) (Sutton & Barto, 1998) is the machine learning answer to the optimal control problem and has been proven to be a promising solution to a wide variety of industrial application domains (e.g., Schaefer et al., 2007; Stephan et al., 2000), including robot control (e.g., Merke & Riedmiller, 2001; Abbeel et al., 2006; Lee et al., 2006; Peters & Schaal, 2008).

In contrast to many classical approaches, building upon extensive domain knowledge, RL aims to derive an optimal policy (i.e., control strategy) from observations only, acquired by the exploration of an unknown environment. For a limited amount of observations the collected information may not be sufficient to fully determine the environment's properties. Assuming the environment to be a Markov decision process (MDP), it is in general only possible to create estimators for the MDP's transition probabilities and the reward function. As the true parameters remain uncertain, the derived policy that is optimal w.r.t. the estimators is in general not optimal w.r.t. the real MDP and may even perform insufficiently. This is unacceptable in industrial environments with high requirements not only on performance, but also robustness and quality assurance.

To overcome this problem, we incorporate the uncertainties of the estimators into the derived Q-function, which is utilised by many RL methods. In order to guarantee a minimal performance with a given probability, as a solution to quality assurance, we present an approach using statistical uncertainty propagation (UP) (e.g., D'Agostini, 2003) on the Bellman iteration to obtain Q-functions together with their uncertainty. In a second step, we introduce a modified Bellman operator, jointly optimising the Q-function and minimising its uncertainty. This method leads to a policy that is no more optimal in the conventional meaning, but maximizes the guaranteed minimal performance and hence optimises the quality requirements. In addition, we show that the approach can be used for efficient exploration as well. In the following we apply the technique exemplarily on discrete MDPs.

This chapter is organised as follows. Within the introduction we give an overview of RL and uncertainty and report on related work. The key section 2 discusses how to bring the concepts of RL and uncertainty together. We explain the application of uncertainty propagation to the Bellman iteration for policy evaluation and policy iteration for discrete MDPs and proceed with section 3, where we introduce the concept of certain-optimality. We further discuss the important observation that certain-optimal policies are stochastic in general (section 4), having a direct impact on the algorithmic solution. Our approach provides a general framework for

different statistical paradigms, we elaborate on this generic view as well as important examples and their advantages and disadvantages in section 5. Section 6 focuses on a possible solution to asymptotically improve the algorithm's time and space complexity and section 7 explains how the proposed concepts can be used to effectively rise to the exploration-exploitation dilemma by seeking uncertain areas of the environment. Finally, in section 8, we focus on the three main application fields quality assurance, exploration, and performance improvement and prove our claims with artificial and industrial benchmarks.

1.1 Reinforcement learning

In (RL) the main objective is to achieve a policy that optimally moves an agent within a Markov decision process (MDP), which is given by state and action spaces S and A as well as the dynamics, defined by a transition probability distribution P_T: $S \times A \times S \rightarrow [0,1]$ depending on the the current state, the chosen action, and the successor state. The agent collects rewards while transiting, whose expected discounted future sum

$$V^{\pi}(s) = \mathrm{E}_s^{\pi}\left(\sum_{i=0}^{\infty}\gamma^i R\left(s^{(i)}, \pi\left(s^{(i)}\right), s^{(i+1)}\right)\right), \tag{1}$$

the value function, has to be maximised over the policy space $\Pi = \{\pi | \pi: S \rightarrow A\}$ for all possible states s, where $0 < \gamma < 1$ is the discount factor, s' the successor state of s, $\pi \in \Pi$ the used policy, and $s = \{s', s'', \ldots, s^{(i)}, \ldots\}$. As an intermediate step one constructs a so-called Q-function $Q^{\pi}(s,a)$ depending on the current state and the chosen action. The optimal $Q^* = Q^{\pi^*}$ is determined by a solution of the Bellman optimality equation

$$Q^*(s,a) = \mathrm{E}_{s'}\left(R(s,a,s') + \gamma V^*(s')\right) \tag{2}$$

$$= \mathrm{E}_{s'}\left(R(s,a,s') + \gamma \max_{a'} Q^*(s',a')\right). \tag{3}$$

Therefore the best policy is $\pi^*(s) = \mathrm{argmax}_a\, Q^*(s, a)$. We define the Bellman operator T as $(TQ)(s, a) = \mathrm{E}_{s'}(R(s, a, s') + \gamma \max_{a'} Q(s', a'))$ for any Q. The fixed point of Q = **Solve**(TQ), i.e., the Bellman operator followed by its projection on the Q-function's hypothesis space, is the approached solution (Sutton & Barto, 1998; Lagoudakis & Parr, 2003; Munos, 2003). Given the parameters of the MDP, i.e., the definitions of the state and the action space, the transition probabilities, and the reward function, this solution can be found using dynamic programming.

For further details and a more broad and general introduction to RL we refer to Sutton & Barto (1998) or Kaelbling et al. (1996).

1.2 Uncertainty

Statistical uncertainty is a crucial issue in many application fields of statistics including the machine learning domain. It is well accepted that any measurement in nature and any conclusion from measurements are affected by an uncertainty. The International Organization for Standardization (ISO) defines uncertainty to be

> "a parameter, associated with the result of a measurement, that characterizes the dispersion of the values that could reasonably be attributed to the measurand" (ISO, 1993).

We focus on the determination, quantisation, and minimisation of uncertainty of the measurements' conclusions in the context of RL, i.e., the uncertainties of Q-functions and policies. The reason for uncertainty in RL is the ignorance about the true environment, i.e., the true MDP. The more observations are collected, the more certain the observer is about the MDP. And the larger the stochasticity, the more uncertainty remains about the MDP for a given amount of observations. And indeed, if the MDP is known to be completely deterministic, everything is known about a state-action pair if it is observed once. There is no uncertainty left. If in contrast the system is highly stochastic, the risk of obtaining a low long-term return in expectation is large.

Note that the mentioned uncertainty is therefore qualitatively different from the MDP's stochasticity leading to the risk of obtaining a low long-term return in the single run. The main difference is that the latter considers the inherent stochasticity of the MDP, whereas uncertainty considers the stochasticity of choosing an MDP from a set of MDPs.

The uncertainty of the measurements, i.e., the transitions and rewards, are propagated to the conclusions, e.g., the Q-function, by uncertainty propagation (UP), which is a common concept in statistics (D'Agostini, 2003). We determine the uncertainty of values $f(\mathrm{x})$ with $f\colon \mathbb{R}^m \to \mathbb{R}^n$ given the uncertainty of their arguments x as

$$\mathrm{Cov}(f) = \mathrm{Cov}(f, f) = D\mathrm{Cov}(\mathrm{x})D^T, \tag{4}$$

where $D_{i,j} = \frac{\partial f_i}{\partial \mathrm{x}_j}$ is the Jacobian matrix of f w.r.t. x and $\mathrm{Cov}(\mathrm{x}) = \mathrm{Cov}(\mathrm{x},\mathrm{x})$ the covariance matrix of the arguments x holding the uncertainty of x.

In the following, we usually work on multi-dimensional objects, having several indices, rather than vectors like f or x, having a single index. Therefore, those objects have to be appropriately vectorised. This can be done by any enumeration and is only of technical importance.

1.3 Related work

There have already been several contributions to estimate generalisation, confidence, and performance bounds in RL. We consider the work of Bertsekas & Tsitsiklis (1996), who gave lower-bounds on the policy's performance by using policy iteration techniques, which were substantially improved by Munos (2003). Kearns et al. (2000) discussed error-bounds for a theoretical policy search algorithm based on trajectory trees. Capacity results on policy evaluation are given by Peshkin & Mukherjee (2001). Antos et al. (2006) provided a broad capacity analysis of Bellman residual minimisation in batch RL. Incorporating prior knowledge about confidence and uncertainty directly into the approached policy were already applied in former work as well in the context of Bayesian RL. We especially mention the work of Engel et al. (2003; 2005), Gaussian process temporal difference learning (GPTD), and a similar approach by Rasmussen & Kuss (2003). They applied Gaussian processes and hence a prior distribution over value functions in RL, which is updated to posteriors by observing samples from the MDP. Ghavamzadeh & Engel recently developed algorithms for Bayesian policy gradient RL (2006) and Bayesian actor-critic RL (2007) as further model-free approaches to Bayesian RL. In all these methods Gaussian processes are applied to obtain the value function and the policy's gradient, respectively.

In model-based approaches, however, one starts with a natural local measure of the uncertainty of the transition probabilities and rewards. One of the first contributions in the

context of RL is provided by Dearden et al. (1998; 1999), who applied Q-learning in a Bayesian framework with an application to the exploration-exploitation trade-off. Poupart et al. (2006) present an approach for efficient online learning and exploration in a Bayesian context, they ascribe Bayesian RL to POMDPs. Besides, statistical uncertainty consideration is similar to, but strictly demarcated from other issues that deal with uncertainty and risk consideration. Consider the work of Heger (1994) and of Geibel (2001). They deal with risk in the context of undesirable states. Mihatsch & Neuneier (2002) developed a method to incorporate the inherent stochasticity of the MDP. Most related to our approach is the recent independent work by Delage & Mannor (2007), who solved the percentile optimisation problem by convex optimization and applied it to the exploration-exploitation trade-off. They suppose special priors on the MDP's parameters, whereas the present work has no such requirements and can be applied in a more general context of RL methods.

2. Bellman iteration and uncertainty propagation

Our concept of incorporating uncertainty into RL consists in applying UP to the Bellman iteration (Schneegass et al., 2008)

$$Q^m(s_i,a_j) \quad := \quad (TQ^{m-1})(s_i,a_j) \tag{5}$$

$$= \quad \sum_{k=1}^{|S|} P(s_k \mid s_i,a_j)(R(s_i,a_j,s_k)+\gamma V^{m-1}(s_k)), \tag{6}$$

here for discrete MDPs. For policy evaluation we have $V^m(s) = Q^m(s,\pi(s))$, with π the used policy, and for policy iteration $V^m(s) = \max_{a\in A} Q^m(s, a)$ (section 1.1). Thereby we assume a finite number of states s_i, $i \in \{1, \ldots, |S|\}$, and actions a_j, $j \in \{1, \ldots, |A|\}$. The Bellman iteration converges, with $m \to \infty$, to the optimal Q-function, which is appropriate to the estimators P and R. In the general stochastic case, which will be important later, we set $V^m(s) = \sum_{i=1}^{|A|} \pi(s,a_i) Q^m(s,a_i)$ with $\pi(s, a)$ the probability of choosing a in s. To obtain the uncertainty of the approached Q-function, the technique of UP is applied in parallel to the Bellman iteration. With given covariance matrices Cov(P), Cov(R), and Cov(P,R) for the transition probabilities and the rewards, we obtain the initial complete covariance matrix

$$\mathrm{Cov}(Q^0,P,R) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \mathrm{Cov}(P) & \mathrm{Cov}(P,R) \\ 0 & \mathrm{Cov}(P,R)^T & \mathrm{Cov}(R) \end{pmatrix} \tag{7}$$

and the complete covariance matrix after the mth Bellman iteration

$$\mathrm{Cov}(Q^m,P,R) \quad := \quad D^{m-1}\mathrm{Cov}\left(Q^{m-1},P,R\right)\left(D^{m-1}\right)^T \tag{8}$$

with the Jacobian matrix

$$D^m = \begin{pmatrix} D^m_{Q,Q} & D^m_{Q,P} & D^m_{Q,R} \\ 0 & \mathrm{I} & 0 \\ 0 & 0 & \mathrm{I} \end{pmatrix}, \tag{9}$$

$$
\begin{aligned}
(D^m_{Q,Q})_{(i,j),(k,l)} &= \gamma\pi(s_k,a_l)P(s_k \mid s_i,a_j),\\
(D^m_{Q,P})_{(i,j),(l,n,k)} &= \delta_{i,l}\delta_{j,n}\left(R(s_i,a_j,s_k)+\gamma V^m(s_k)\right),\\
(D^m_{Q,R})_{(i,j),(l,n,k)} &= \delta_{i,l}\delta_{j,n}P(s_k \mid s_i,a_j).
\end{aligned}
$$

In combination with the expanded Bellman iteration

$$
\begin{pmatrix} Q^m & P & R \end{pmatrix}^T := \begin{pmatrix} TQ^{m-1} & P & R \end{pmatrix}^T \tag{10}
$$

the presented uncertainty propagation allows to obtain the covariances between Q-function and P and R, respectively. All parameters of Q^m are linear in Q^m, altogether it is a bi-linear function. Therefore, UP is indeed approximately applicable in this setting (D'Agostini, 2003). Having identified the fixed point consisting of Q^* and its covariance $\mathrm{Cov}(Q^*)$, the uncertainty of each individual state-action pair is represented by the square root of the diagonal entries $\sigma Q^* = \sqrt{\mathrm{diag}(\mathrm{Cov}(Q^*))}$, since the diagonal comprises the Q-values' variances.

Finally, with probability $P(\xi)$ depending on the distribution class of Q, the function

$$
Q^*_u(s,a) = (Q^* - \xi\sigma Q^*)(s,a) \tag{11}
$$

provides the guaranteed performance expectation applying action a in state s strictly followed by the policy $\pi^*(s) = \mathrm{argmax}_a\, Q^*(s, a)$. Suppose exemplarily Q to be distributed normally, then the choice $\xi = 2$ would lead to the guaranteed performance with $P(2) \approx 0.977$. The appendix provides a proof of existence and uniqueness of the fixed point consisting of Q^* and $\mathrm{Cov}(Q^*)$.

3. Certain-optimality

The knowledge of uncertainty may help in many areas, e.g., improved exploration (see section 7), a general understanding of quality and risks related to the policy's actual usage, but it does not help to improve the guaranteed performance in a principled manner. By applying $\pi(s) = \mathrm{argmax}_a\, Q^*_u(s, a)$, the uncertainty would not be estimated correctly as the agent is only allowed once to decide for another action than the approached policy suggests. To overcome this problem, we want to approach a so-called certain-optimal policy, which maximises the guaranteed performance. The idea is to obtain a policy π that is optimal w.r.t. a specified confidence level, i.e., which maximises $Z(s, a)$ for all s and a such that

$$
P\left(\bar{Q}^\pi(s,a) > Z(s,a)\right) > P(\xi) \tag{12}
$$

is fulfilled, where $\bar{Q}^\pi$ denotes the true performance function of π and $P(\xi)$ being a prespecified probability. We approach such a solution by approximating Z by Q^π_u and solving

$$
\pi^\xi(s) = \arg\max_\pi \max_a Q^\pi_u(s,a) \tag{13}
$$

$$
= \arg\max_\pi \max_a (Q^\pi - \xi\sigma Q^\pi)(s,a) \tag{14}
$$

under the constraints that $Q^{\pi^\xi} = Q^\xi$ is the valid Q-function for π^ξ, i.e.,

$$Q^\xi(s_i, a_j) = \sum_{k=1}^{|S|} P(s_k \mid s_i, a_j)(R(s_i, a_j, s_k) + \gamma Q^\xi(s_k, \pi^\xi(s_k))). \tag{15}$$

Relating to the Bellman iteration, Q shall be a fixed point not w.r.t. the value function as the maximum over all Q-values, but the maximum over the Q-values minus its weighted uncertainty. Therefore, one has to choose

$$\pi^m(s) \quad := \quad \arg\max_a (Q^m - \xi\sigma Q^m)(s, a) \tag{16}$$

after each iteration, together with an update of the uncertainties according to the modified policy π^m.

4. Stochasticity of certain-optimal policies

Policy evaluation can be applied to obtain deterministic or stochastic policies. In the framework of MDPs an optimal policy which is deterministic always exists (Puterman, 1994). For certain-optimal policies, however, the situation is different. Particularly, for $\xi > 0$ there is a bias on $\xi\sigma Q(s,\pi(s))$ being larger than $\xi\sigma Q(s, a)$, $a \neq \pi(s)$, if π is the evaluated policy, since $R(s,\pi(s), s')$ depends stronger on $V(s') = Q(s',\pi(s'))$ than $R(s, a, s')$, $a \neq \pi(s)$. The value function implies the choice of action $\pi(s)$ for all further occurrences of state s. Therefore, the (deterministic) joint iteration is not necessarily guaranteed to converge. I.e., switching the policy π to π' with $Q(s,\pi'(s)) - \xi\sigma Q(s,\pi'(s)) > Q(s,\pi(s)) - \xi\sigma Q(s,\pi(s))$ could lead to a larger uncertainty of π' at s and hence to $Q'(s,\pi'(s)) - \xi\sigma Q'(s,\pi'(s)) < Q'(s,\pi(s)) - \xi\sigma Q'(s,\pi(s))$ for Q' at the next iteration. This causes an oscillation.

Additionally, there is another effect causing an oscillation when there is a certain constellation of Q-values and corresponding uncertainties of concurring actions. Consider two actions a_1 and a_2 in a state s with similar Q-values but different uncertainties, a_1 having an only slightly higher Q-value but a larger uncertainty. The uncertainty-aware policy improvement step (equation (16)) would alter π^m to choose a_2, the action with the smaller uncertainty. However, the fact that this action is inferior might only become obvious in the next iteration when the value function is updated for the altered π^m (and now implying the choice of a_2 in s). In the following policy improvement step the policy will be changed back to choose a_1 in s, since now the Q-function reflects the inferiority of a_2. After the next update of the Q-function, the values for both actions will be similar again, because now the value function implies the choice of a_1 and the bad effect of a_2 affects $Q(s, a_2)$ only once.

It is intuitively apparent that a certain-optimal policy should be stochastic in general if the gain in value must be balanced with the gain in certainty, i.e., with a decreasing risk of having estimated the wrong MDP. The risk to obtain a low expected return is hence reduced by diversification, a well-known method in many industries and applications.

The value ξ decides about the cost of certainty. If $\xi > 0$ is large, certain-optimal policies tend to become more stochastic, one pays a price for the benefit of a guaranteed minimal performance, whereas a small $\xi \leq 0$ guarantees deterministic certain-optimal policies and uncertainty takes on the meaning of the chance for a high performance. Therefore, we finally define a stochastic uncertainty incorporating Bellman iteration as

$$\begin{pmatrix} Q^m \\ C^m \\ \pi^m \end{pmatrix} := \begin{pmatrix} TQ^{m-1} \\ D_{m-1}C^{m-1}D_{m-1}^T \\ \Lambda(\pi^{m-1}, TQ^{m-1}, m) \end{pmatrix} \tag{17}$$

with

$$\Lambda(\pi, Q, t)(s,a) = \begin{cases} \min(\pi(s,a) + \frac{1}{t}, 1) & : \quad a = a_Q(s) \\ \dfrac{\max(1 - \pi(s, a_Q(s)) - \frac{1}{t}, 0)}{1 - \pi(s, a_Q(s))} \pi(s,a) & : \quad \text{otherwise} \end{cases} \tag{18}$$

and $a_Q(s) = \text{argmax}_a\ (Q - \xi\sigma Q)(s, a)$. The harmonically decreasing change rate of the stochastic policies guarantees reachability of all policies on the one hand and convergence on the other hand. Algorithm 1 summarises the joint iteration.[1]

Algorithm 1 Uncertainty Incorporating Joint Iteration for Discrete MDPs

Require: given estimators P and R for a discrete MDP, initial covariance matrices $\text{Cov}(P)$, $\text{Cov}(R)$, and $\text{Cov}(P, R)$ as well as a scalar ξ

Ensure: calculates a certain-optimal Q-function Q and policy π under the assumption of the observations and the posteriors given by $\text{Cov}(P)$, $\text{Cov}(R)$, and $\text{Cov}(P, R)$

set $\text{C} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \text{Cov}(P) & \text{Cov}(P,R) \\ 0 & \text{Cov}(P,R)^T & \text{Cov}(R) \end{pmatrix}$

set $\forall i, j : Q(s_i, a_j) = 0$, $\forall i, j : \pi(s_i, a_j) = \frac{1}{|A|}$, $t = 0$

while the desired precision is not reached **do**

set $t = t + 1$

set $\forall i, j : (\sigma Q)(s_i, a_j) = \sqrt{C_{i|A|+j, i|A|+j}}$

find $\forall i : a_{i,\max} = \arg\max_{a_j} (Q - \xi\sigma Q)(s_i, a_j)$

set $\forall i : d_{i,\text{diff}} = \min\left(\frac{1}{t}, 1 - \pi(s_i, a_{i,\max})\right)$

set $\forall i : \pi(s_i, a_{i,\max}) = \pi(s_i, a_{i,\max}) + d_{i,\text{diff}}$

set $\forall i : \forall a_j \neq a_{i,\max} : \pi(s_i, a_j) = \frac{1 - \pi(s_i, a_{i,\max})}{1 - \pi(s, a_{i,\max}) + d_{i,\text{diff}}} \pi(s_i, a_j)$

set $\forall i, j : Q'(s_i, a_j) = \sum_{k=1}^{|S|} P(s_k | s_i, a_j) \left(R(s_i, a_j, s_k) + \gamma \sum_{l=1}^{|A|} \pi(s_k, a_l) Q(s_k, a_l)\right)$

set $Q = Q'$

set $D = \begin{pmatrix} D_{Q,Q} & D_{Q,P} & D_{Q,R} \\ 0 & \text{I} & 0 \\ 0 & 0 & \text{I} \end{pmatrix}$

set $C = DCD^T$

end while

return $Q - \xi\sigma Q$ and π

[1] Sample implementations of our algorithms and benchmark problems can be found at: http: //ahans.de/publications/robotlearning2010uncertainty/

The function $Q_u^{\xi}(s, a)=(Q_x - \xi\sigma Q_x)(s, a)$ with (Q_x, C_x, π_x) as the fixed point of the (stochastic) joint iteration for given ξ provides, with probability $P(\xi)$ depending on the distribution class of Q, the guaranteed performance applying action a in state s strictly followed by the stochastic policy π_x. First and foremost, π_x maximises the guaranteed performance and is therefore called a certain-optimal policy.

5. The initial covariance matrix – statistical paradigms

The initial covariance matrix

$$\mathrm{Cov}((P,R)) = \begin{pmatrix} \mathrm{Cov}(P,P) & \mathrm{Cov}(P,R) \\ \mathrm{Cov}(P,R)^T & \mathrm{Cov}(R,R) \end{pmatrix} \tag{19}$$

has to be designed by problem dependent prior belief. If, e.g., all transitions from different state-action pairs and the rewards are assumed to be mutually independent, all transitions can be modelled as multinomial distributions. In a Bayesian context one supposes a priorly known distribution (D'Agostini, 2003; MacKay, 2003) over the parameter space $P(s_k \mid s_i, a_j)$ for given i and j. The Dirichlet distribution with density

$$P(P(s_1 \mid s_i, a_j), \ldots, P(s_{|S|} \mid s_i, a_j))_{\alpha_{1,i,j},\ldots,\alpha_{|S|,i,j}} = \frac{\Gamma(\alpha_{i,j})}{\prod_{k=1}^{|S|}\Gamma(\alpha_{k,i,j})} \prod_{k=1}^{|S|} P(s_k \mid s_i, a_j)^{\alpha_{k,i,j}-1} \tag{20}$$

and $\alpha_{i,j} = \sum_{k=1}^{|S|} \alpha_{k,i,j}$ is a conjugate prior in this case with posterior parameters

$$\alpha_{k,i,j}^d = \alpha_{k,i,j} + n_{s_k \mid s_i, a_j} \tag{21}$$

in the light of the observations occurring $n_{s_k \mid s_i, a_j}$ times a transition from s_i to s_k by using action a_j. The initial covariance matrix for P then becomes

$$(\mathrm{Cov}(P))_{(i,j,k),(l,m,n)} = \delta_{i,l}\delta_{j,m} \frac{\alpha_{k,i,j}^d(\delta_{k,n}\alpha_{i,j}^d - \alpha_{n,i,j}^d)}{(\alpha_{i,j}^d)^2(\alpha_{i,j}^d + 1)}, \tag{22}$$

assuming the posterior estimator $P(s_k \mid s_i, a_j) = \alpha_{k,i,j}^d / \alpha_{i,j}^d$. Similarly, the rewards might be distributed normally with the normal-gamma distribution as a conjugate prior.
As a simplification or by using the frequentist paradigm, it is also possible to use the relative frequency as the expected transition probabilities with their uncertainties

$$(\mathrm{Cov}(P))_{(i,j,k),(l,m,n)} = \delta_{i,l}\delta_{j,m} \frac{P(s_k \mid s_i, a_j)(\delta_{k,n} - P(s_n \mid s_i, a_j))}{n_{s_i, a_j} - 1} \tag{23}$$

with n_{s_i, a_j} observed transitions from the state-action pair (s_i, a_j).
Similarly, the rewards expectations become their sample means and $\mathrm{Cov}(R)$ a diagonal matrix with entries

$$\mathrm{Cov}(R(s_i, a_j, s_k)) = \frac{\mathrm{Var}(R(s_i, a_j, s_k))}{n_{s_k \mid s_i, a_j} - 1}. \tag{24}$$

The frequentist view and the conjugate priors have the advantage of being computationally feasible, nevertheless, the method is not restricted to them, any meaningful covariance matrix Cov((*P*,*R*)) is allowed. Particularly, applying covariances between the transitions starting from different state-action pairs and between states and rewards is reasonable and interesting, if there is some measure of neighbourhood over the state-action space. Crucial is finally that the prior represents the user's belief.

6. Improving asymptotic performance

The proposed algorithm's time complexity per iteration is of higher order than the standard Bellman iteration's one, which needs $O(|S|^2|A|)$ time ($O(|S|^2|A|^2)$ for stochastic policies). The bottleneck is the covariance update with a time complexity of $O((|S||A|)^{2.376})$ (Coppersmith & Winograd, 1990), since each entry of Q depends only on $|S|$ entries of P and R. The overall complexity is hence bounded by these magnitudes.

This complexity can limit the applicability of the algorithm for problems with more than a few hundred states. To circumvent this issue, it is possible to use an approximate version of the algorithm that considers only the diagonal of the covariance matrix. We call this variant the *diagonal approximation of uncertainty incorporating policy iteration* (DUIPI) (Hans & Udluft, 2009). Only considering the diagonal neglects the correlations between the state-action pairs, which in fact are small for many RL problems, where on average different state-action pairs share only little probability to reach the same successor state.

DUIPI is easier to implement and, most importantly, lies in the same complexity class as the standard Bellman iteration. In the following we will derive the update equations for DUIPI. When neglecting correlations, the uncertainty of values $f(x)$ with $f : \mathbb{R}^m \to \mathbb{R}^n$, given the uncertainty of the arguments x as σx, is determined as

$$(\sigma f)^2 = \sum_i \left(\frac{\partial f}{\partial x_i} \right)^2 (\sigma x_i)^2. \tag{25}$$

This is equivalent to equation (4) of full-matrix UP with all non-diagonal elements set equal to zero.

The update step of the Bellman iteration,

$$Q^m(s,a) := \sum_{s'} P(s' \mid s,a) \Big[R(s,a,s') + \gamma V^{m-1}(s') \Big], \tag{26}$$

can be regarded as a function of the estimated transition probabilities P and rewards R, and the Q-function of the previous iteration $Q^{m\text{-}1}$ ($V^{m\text{-}1}$ is a subset of $Q^{m\text{-}1}$), that yields the updated Q-function Q^m. Applying UP as given by equation (25) to the Bellman iteration, one obtains an update equation for the Q-function's uncertainty:

$$\begin{aligned} (\sigma Q^m(s,a))^2 \quad := \quad & \sum_{s'} (D_{Q,Q})^2 (\sigma V^{m-1}(s'))^2 + \\ & \sum_{s'} (D_{Q,P})^2 (\sigma P(s' \mid s,a))^2 + \\ & \sum_{s'} (D_{Q,R})^2 (\sigma R(s,a,s'))^2, \end{aligned} \tag{27}$$

$$D_{Q,Q} = \gamma P(s' \mid s,a),\ D_{Q,P} = R(s,a,s') + \gamma V^{m-1}(s'),\ D_{Q,R} = P(s' \mid s,a). \tag{28}$$

V^m and σV^m have to be set depending on the desired type of the policy (stochastic or deterministic) and whether policy evaluation or policy iteration is performed. E.g., for policy evaluation of a stochastic policy π

$$V^m(s) = \sum_a \pi(a \mid s) Q^m(s,a), \tag{29}$$

$$(\sigma V^m(s))^2 = \sum_a \pi(a \mid s)^2 (\sigma Q^m(s,a))^2. \tag{30}$$

For policy iteration, according to the Bellman optimality equation and resulting in the Q-function Q^* of an optimal policy, $V^m(s) = \max_a Q^m(s, a)$ and $(\sigma V^m(s))^2 = (\sigma Q^m(s,\text{argmax}_a Q^m(s, a)))^2$.

Using the estimators P and R with their uncertainties σP and σR and starting with an initial Q-function Q^0 and corresponding uncertainty σQ^0, e.g., $Q^0 := 0$ and $\sigma Q^0 := 0$, through the update equations (26) and (27) the Q-function and corresponding uncertainty are updated in each iteration and converge to Q^π and σQ^π for policy evaluation and Q^* and σQ^* for policy iteration.

Like the full-matrix algorithm DUIPI can be used with any choice of estimator, e.g., a Bayesian setting using Dirichlet priors or the frequentist paradigm (see section 5). The only requirement is the possibility to access the estimator's uncertainties σP and σR. In Hans & Udluft (2009) and section 8.2 we give results of experiments using the full-matrix version and DUIPI and compare the algorithms for various applications.

Algorithm 2 Diagonal Approximation of Uncertainty Incorporating Policy Iteration

Require: estimators P and R for a discrete MDP, their uncertainties σP and σR, a scalar ξ
Ensure: calculates a certain-optimal policy π

```
set ∀i,j : Q(s_i,a_j) = 0, (σQ)^2(s_i,a_j) = 0
set ∀i,j : π(s_i,a_j) = 1/|A|, t = 0
while the desired precision is not reached do
    set t = t + 1
    set ∀s : a_{s,max} = argmax_a Q(s,a) − ξ sqrt((σQ)^2(s,a))
    ∀s : d_s = min(1/t, 1 − π(a_{s,max}|s))
    set ∀s : π(a_{s,max}|s) = π(a_{s,max}|s) + d_s
    set ∀s : ∀a ≠ a_{s,max} : π(a|s) = (1 − π(a_{s,max}|s)) / (1 − π(a_{s,max}|s) + d_s) π(a|s)
    set ∀s : V(s) = Σ_a π(s,a) Q(s,a)
    set ∀s : (σV)^2(s) = Σ_a π(s,a) (σQ)^2(s,a)
    set ∀s,a : Q'(s,a) = Σ_{s'} P(s'|s,a) [R(s,a,s') + γ V(s')]
    set ∀s,a : (σQ')^2(s,a) =
      Σ_s (D_{Q,Q})^2 (σV)^2(s') + (D_{Q,P})^2 (σP)^2(s'|s,a) + (D_{Q,R})^2 (σR)^2(s,a,s')
    set Q = Q', (σQ)^2 = (σQ')^2
end while
return π
```

7. Uncertainty-based exploration

Since RL is usually used with an initially unknown environment, it is necessary to explore the environment in order to gather knowledge. In that context the so-called *exploration-exploitation dilemma* arises: when should the agent stop trying to gain more information (explore) and start to act optimally w.r.t. already gathered information (exploit)? Note that this decision does not have to be a binary one. A good solution of the exploration-exploitation problem could also gradually reduce the amount of exploration and increase the amount of exploitation, perhaps eventually stopping exploration altogether.
The algorithms proposed in this chapter can be used to balance exploration and exploitation by combining existing (already gathered) knowledge and uncertainty about the environment to further explore areas that seem promising judging by the current knowledge. Moreover, by aiming at obtaining high rewards and decreasing uncertainty at the same time, good online performance is possible (Hans & Udluft, 2010).

7.1 Efficient exploration in reinforcement learning

There have been many contributions considering efficient exploration in RL. E.g., Dearden et al. (1998) presented *Bayesian Q-learning*, a Bayesian model-free approach that maintains probability distributions over Q-values. They either select an action stochastically according to the probability that it is optimal or select an action based on *value of information*, i.e., select the action that maximises the sum of Q-value (according to the current belief) and expected gain in information. They later added a Bayesian model-based method that maintains a distribution over MDPs, determines value functions for sampled MDPs, and then uses those value functions to approximate the true value distribution (Dearden et al., 1999). In *model-based interval estimation* (MBIE) one tries to build confidence intervals for the transition probability and reward estimates and then optimistically selects the action maximising the value within those confidence intervals (Wiering & Schmidhuber, 1998; Strehl & Littman, 2008). Strehl & Littman (2008) proved that MBIE is able to find near-optimal policies in polynomial time. This was first shown by Kearns & Singh (1998) for their E^3 algorithm and later by Brafman & Tennenholtz (2003) for the simpler *R-Max* algorithm. R-Max takes one parameter C, which is the number of times a state-action pair (s, a) must have been observed until its actual Q-value estimate is used in the Bellman iteration. If it has been observed fewer times, its value is assumed as $Q(s, a) = R_{\max}/(1 - \gamma)$, which is the maximum possible Q-value ($R_{\max}$ is the maximum possible reward). This way exploration of state-action pairs that have been observed fewer than C times is fostered. Strehl & Littman (2008) presented an additional algorithm called *model-based interval estimation with exploration bonus* (MBIE-EB) for which they also prove its optimality. According to their experiments, it performs similarly to MBIE. MBIE-EB alters the Bellman equation to include an exploration bonus term $\beta / \sqrt{n_{s,a}}$, where β is a parameter of the algorithm and $n_{s,a}$ the number of times state-action pair (s, a) has been observed.

7.2 Uncertainty propagation for exploration

Using full-matrix uncertainty propagation or DUIPI with the parameter ξ set to a negative value it is possible to derive a policy that balances exploration and exploitation:

$$\pi^{\xi}(s) := \arg\max_a (Q^* - \xi\sigma Q^*)(s, a). \tag{31}$$

However, like in the quality assurance context, this would allow to consider the uncertainty only for one step. To allow the resulting policy to plan the exploration, it is necessary to include the uncertainty-aware update of the policy in the iteration as described in section 3.
Section 3 proposes to update the policy π^m using Q^m and σQ^m in each iteration and then using π^m in the next iteration to obtain Q^{m+1} and σQ^{m+1}. This way Q-values and uncertainties are not mixed, the Q-function remains the valid Q-function of the resulting policy. Another possibility consists in modifying the Q-values in the iteration with the ξ-weighted uncertainty. However, this leads to a Q-function that is no longer the Q-function of the policy, as it contains not only the sum of (discounted) rewards, but also uncertainties. Therefore, using a Q and σQ obtained this way it is not possible to reason about expected rewards and uncertainties when following this policy. Moreover, when using a negative ξ for exploration the Q-function does not converge in general for this update scheme, because in each iteration the Q-function is increased by the ξ-weighted uncertainty, which in turn leads to higher uncertainties in the next iteration. On the other hand, by choosing ξ and γ to satisfy $\xi + \gamma < 1$ we were able to keep Q and σQ from diverging. Used with DUIPI this update scheme gives rise to a DUIPI variation called *DUIPI with Q-modification* (DUIPI-QM) which has proven useful in our experiments (section 8.2), as DUIPI-QM works well even for environments that exhibit high correlations between different state-action pairs, because through this update scheme of mixing Q-values and uncertainties the uncertainty is propagated through the Q-values.

8. Applications

The presented techniques offer at least three different types of application, which are important in various practical domains.

8.1 Quality assurance and competitions

With a positive ξ one aims at a guaranteed minimal performance of a policy. To optimise this minimal performance, we introduced the concept of certain-optimality. The main practical motivation is to avoid delivering an inferior policy. To simply be aware of the quantification of uncertainty helps to appreciate how well one can count on the result. If the guaranteed Q-value for a specified start state is insufficient, more observations must be provided in order to reduce the uncertainty.
If the exploration is expensive and the system critical such that the performance probability has definitely to be fulfilled, it is reasonable to bring out the best from this concept. This can be achieved by a certain-optimal policy. One abandons "on average" optimality in order to perform as good as possible at the specified confidence level.
Another application field, the counter-part of quality assurance, are competitions, which is symmetrical to quality assurance by using negative ξ. The agent shall follow a policy that gives it the chance to perform exceedingly well and thus to win. In this case, certain-optimality comes again into play as the performance expectation is not the criterion, but the percentile performance.

8.1.1 Benchmarks

For demonstration of the quality assurance and competition aspects as well as the properties of certain-optimal policies, we applied the joint iteration on (fixed) data sets for two simple

classes of MDPs. Furthermore, we sampled over the space of allowed MDPs from their (fixed) prior distribution. As a result we achieve a posterior of the possible performances for each policy.

We have chosen a simple bandit problem with one state and two actions and a class of two-state MDPs with each two actions. The transition probabilities are assumed to be distributed multinomially for each start state, using the maximum entropy prior, i.e., the Beta distribution with $\alpha = \beta = 1$. For the rewards we assumed a normal distribution with fixed variance $\sigma_0 = 1$ and a normal prior for the mean with $\mu = 0$ and $\sigma = 1$. Transition probabilities and rewards for different state-action-pairs are assumed to be mutually independent. For the latter benchmark, for instance, we defined to have made the following observations (states **s**, actions **a**, and rewards **r**) over time:

$$\mathbf{s} = (1, 1, 1, 1, 1, 2, 2, 2, 2, 2,2), \tag{32}$$

$$\mathbf{a} = (1, 1, 2, 2, 1, 1, 1, 2, 2,2), \tag{33}$$

$$\mathbf{r} = (1.35, 1, 1, 1, 1, 1, 0, 0, 1, -1). \tag{34}$$

On the basis of those observations we deployed the joint Bellman iteration for different values of ξ, each leading to a policy π^{ξ} that depends on ξ only. The estimates for P and R as well as the initial covariance matrix C^0 are chosen in such a way, that they exactly correspond with the above mentioned posterior distributions. Concurrently, we sampled MDPs from the respective prior distribution. On each of these MDPs we tested the defined policies and weighted their performance probabilities with the likelihood to observe the defined observations given the sampled MDP.

8.1.2 Results

Figure 1 shows the performance posterior distributions for different policies on the two-state MDP problem. Obviously, expectation and variance adopt different values per policy. The expectation-optimal policy reaches the highest expectation whereas the certain and stochastic policies show a lower variance and the competition policy has a wider performance distribution. Each of these properties is exactly the precondition for the aspired behaviour of the respective policy type.

The figures 2 left (bandit problem) and 2 right (two-state MDP problem) depict the percentile performance curves of different policies. In case of the two-state MDP benchmark, these are the same policies as in figure 1 (same colour, same line style), enriched by additional ones. The cumulative distribution of the policies' performances is exactly the inverse function of the graphs in figure 2. Thereby we facilitate a comparison of the performances on different percentiles. The right figure clearly states that the fully stochastic policy shows superior performance at the 10th percentile whereas a deterministic policy, different from the expectation-optimal one, achieves the best performance at the 90th percentile.

In table 1 we listed the derived policies and the estimated percentile performances (given by the Q-function) for different ξ for the two-state MDP benchmark. They approximately match the certain-optimal policies on each of the respective percentiles. With increasing ξ (decreasing percentile) the actions in the first state become stochastic at first and later on the actions in the second state as well. For decreasing ξ the (deterministic) policy switches its action in the first state at some threshold whereas the action in the second state stays the same. These observations can be comprehended from both the graph and the table.

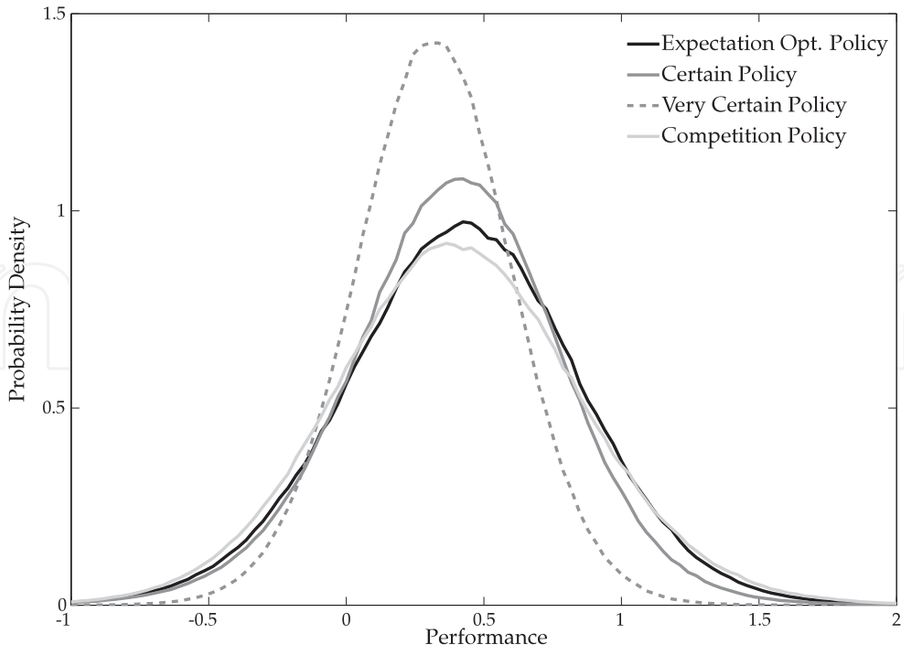


Fig. 1. Performance distribution for different (stochastic) policies on a class of simple MDPs with two states and two actions. The performances are approximately normally distributed. The expectation is highest for the expectation-optimal policy whereas the certain and most stochastic policy features the lowest variance and the highest percentile performance below a certain threshold.

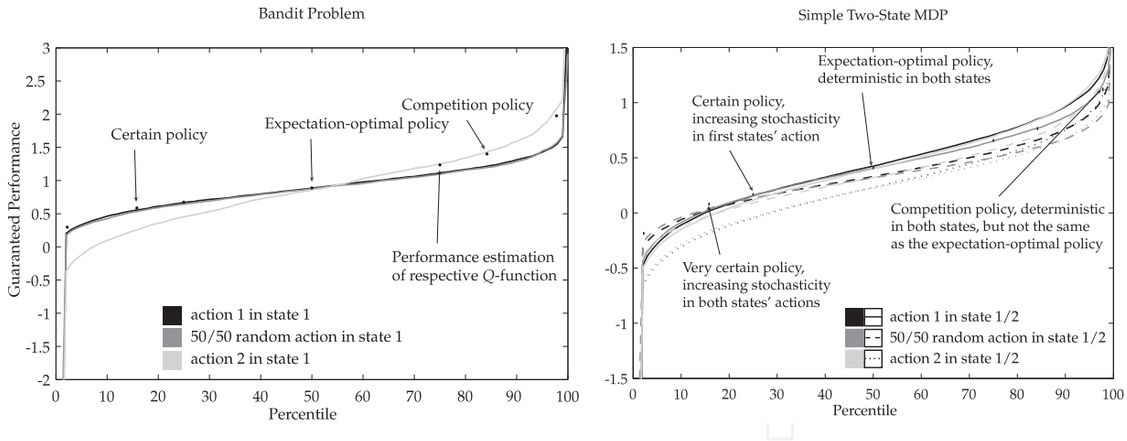


Fig. 2. Percentile performance for simple MDPs and joint iteration results. The different graphs show the percentile performance curves achieved by different policies (i.e., the inverse of the cumulative performance distribution). The grey scale value and the line style depict what action to choose on the state/both states. The dots show the estimated Q-values for the derived certain-optimal policies at the specified percentile. Q-values are distributed normally. The percentiles have been specified by values of $\xi \in \{2, 1$ (*certain policy*), $2/3, 0$ (*expectation-optimal policy*), $-2/3, -1$ (*competition policy*), $-2\}$ for the bandit problem and $\xi \in \{2, 1.5$ (*very certain policy*), $1, 2/3$ (*certain policy*), 0 (*expectation-optimal policy*), $-2/3, -1, -1.5$ (*competition policy*), $-2\}$ on the simple two-state MDP.

ξ	Percentile Performance	π(1,1)	π(1,2)	π(2,1)	π(2,2)	Entropy
4	− 0.663	0.57	0.43	0.52	0.48	0.992
3	− 0.409	0.58	0.42	0.55	0.45	0.987
2	− 0.161	0.59	0.41	0.60	0.40	0.974
1	0.106	0.61	0.39	0.78	0.22	0.863
2/3	0.202	0.67	0.33	1	0	0.458
0	0.421	1	0	1	0	0
− 2/3	0.651	1	0	1	0	0
− 1	0.762	1	0	1	0	0
− 2	1.103	1	0	1	0	0
− 3	1.429	0	1	1	0	0
− 4	1.778	0	1	1	0	0

Table 1. Derived certain-optimal policies for different values of ξ on the above mentioned dataset (equations (32), (33) and (34)) and the assumed prior for the two-state MDP benchmark problem. In addition the estimated percentile performances and the policies' entropies are given. The results are consistent with figure 2 (right), i.e., the derived policies approximately match the actually certain-optimal policies on the respective percentiles.

8.2 Exploration

As outlined in section 7 our approach can also be used for efficient exploration by using a negative ξ. This leads to a policy that explores state-action pairs where $Q_u^{\xi}(s,a)$ is large more intensively, since the estimator of the Q-value is already large but the true performance of the state-action pair could be even better as the uncertainty is still large as well.

To demonstrate the functionality of our approach for exploration we conducted experiments using two benchmark applications from the literature. We compare the full-matrix version, classic DUIPI, DUIPI with Q-function modification, and two established algorithms for exploration, R-Max (Brafman & Tennenholtz, 2003) and MBIE-EB (Strehl & Littman, 2008). Furthermore, we present some insight of how the parameter ξ influences the agent's behaviour. Note that the focus here is not only gathering information about the environment but also balancing exploration and exploitation in order to provide good online performance.

8.2.1 Benchmarks

The first benchmark is the *RiverSwim* domain from Strehl & Littman (2008), which is an MDP consisting of six states and two actions. The agent starts in one of the first two states (at the beginning of the row) and has the possibility to swim to the left (with the current) or to the right (against the current). While swimming to the left always succeeds, swimming to the right most often leaves the agent in the same state, sometimes leads to the state to the right, and occasionally (with small probability) even leads to the left. When swimming to the left in the very left state, the agent receives a small reward. When swimming to the right in the very right state, the agent receives a very large reward, for all other transitions the reward is zero. The optimal policy thus is to always swim to the right. See figure 3 for an illustration.

The other benchmark is the *Trap* domain from Dearden et al. (1999). It is a maze containing 18 states and four possible actions. The agent must collect flags and deliver them to the goal. For each flag delivered the agent receives a reward. However, the maze also contains a trap

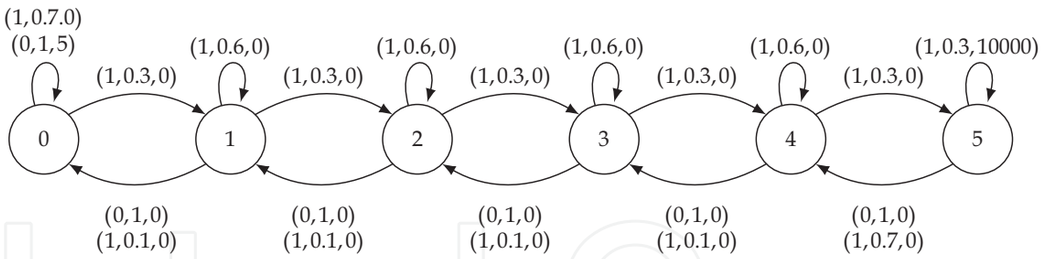


Fig. 3. Illustration of the RiverSwim domain. In the description (*a*, *b*, *c*) of a transition *a* is the action, *b* the probability for that transition to occur, and *c* the reward.

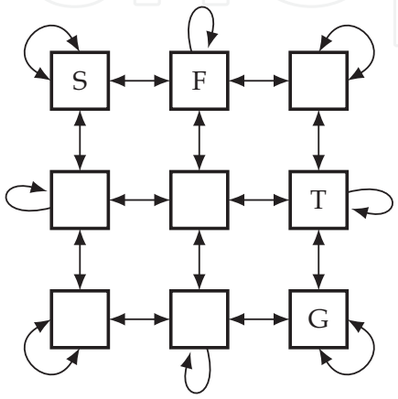


Fig. 4. Illustration of the Trap domain. Starting in state S the agent must collect the flag from state F and deliver it to the goal state G. Once the flag is delivered to state G, the agent receives a reward and is transferred to the start state S again. Upon entering the trap state T a large negative reward is given. In each state the agent can move in all four directions. With probability 0.9 it moves in the desired direction, with probability 0.1 it moves in one of the perpendicular directions with equal probability.

state. Entering the trap state results in a large negative reward. With probability 0.9 the agent's action has the desired effect, with probability 0.1 the agent moves in one of the perpendicular directions with equal probability. See figure 4 for an illustration.
For each experiment we measured the cumulative reward for 5000 steps. The discount factor was set $\gamma = 0.95$ for all experiments. For full-matrix UP, DUIPI, and DUIPI-QM we used Dirichlet priors (section 5). The algorithms were run whenever a new observation became available, i.e., in each step.

8.2.2 Results

Table 2 summarises the results for the considered domains and algorithms obtained with the respective parameters set to the optimal ones found.
For RiverSwim all algorithms except classic DUIPI perform comparably. By considering only the diagonal of the covariance matrix, DUIPI neglects the correlations between different state-action pairs. Those correlations are large for state-action pairs that have a significant probability of leading to the same successor state. In RiverSwim many state-action pairs have this property. Neglecting the correlations leads to an underestimation of the uncertainty, which prevents DUIPI from correctly propagating the uncertainty of Q-values of the right most state to states further left. Thus, although Q-values in state 5 have a

	RiverSwim	Trap
R-Max	$3.02 \pm 0.03 \times 10^6$	469 ± 3
MBIE-EB	$3.13 \pm 0.03 \times 10^6$	558 ± 3
full-matrix UP	$2.59 \pm 0.08 \times 10^6$	521 ± 20
DUIPI	$0.62 \pm 0.03 \times 10^6$	554 ± 10
DUIPI-QM	$3.16 \pm 0.03 \times 10^6$	565 ± 11

Table 2. Best results obtained using the various algorithms in the RiverSwim and Trap domains. Shown is the cumulative reward for 5000 steps averaged over 50 trials for full-matrix UP and 1000 trials for the other algorithms. The used parameters for R-Max were C = 16 (RiverSwim) and C = 1 (Trap), for MBIE-EB β = 0.01 (RiverSwim) and β = 0.01 (Trap), for full-matrix UP α = 0.3, $\xi = -1$ (RiverSwim) and α = 0.3, $\xi = -0.05$ (Trap), for DUIPI α = 0.3, $\xi = -2$ (RiverSwim) and α = 0.1, $\xi = -0.1$ (Trap), and for DUIPI-QM α = 0.3, $\xi = -0.049$ (RiverSwim) and α = 0.1, $\xi = -0.049$ (Trap).

large uncertainty throughout the run, the algorithm settles for exploiting the action in the left most state giving the small reward if it has not found the large reward after a few tries. DUIPI-QM does not suffer from this problem as it modifies Q-values using uncertainty. In DUIPI-QM, the uncertainty is propagated through the state space by means of the Q-values. In the Trap domain the correlations of different state-action pairs are less strong. As a consequence, DUIPI and DUIPI-QM perform equally well. Also the performance of MBIE-EB is good in this domain, only R-Max performs worse than the other algorithms. R-Max is the only algorithm that bases its explore/exploit decision solely on the number of executions of a specific state-action pair. Even with its parameter set to the lowest possible value, it often visits the trap state and spends more time exploring than the other algorithms.

8.2.3 Discussion

Figure 5 shows the effect of ξ for the algorithms. Except DUIPI-QM the algorithms show "inverted u"-behaviour. If ξ is too large (its absolute value too small), the agent does not explore much and quickly settles on a suboptimal policy. If, on the other hand, ξ is too small (its absolute value too large), the agent spends more time exploring. We believe that DUIPI-QM would exhibit the same behaviour for smaller values for ξ, however, those are not usable as they would lead to a divergence of Q and σQ.

Figure 6 shows the effect ξ using DUIPI in the Trap domain. While with large ξ the agent quickly stops exploring the trap state and starts exploiting, with small ξ the uncertainty keeps the trap state attractive for more time steps, resulting in more negative rewards.

Using uncertainty as a natural incentive for exploration is achieved by applying uncertainty propagation to the Bellman equation. Our experiments indicate that it performs at least as good as established algorithms like R-Max and MBIE-EB. While most other approaches to exploration assume a specific statistical paradigm, our algorithm does not make such assumptions and can be combined with any estimator. Moreover, it does not rely on state-action pair counters, optimistic initialisation of Q-values, or explicit exploration bonuses. Most importantly, when the user decides to stop exploration, the same method can be used to obtain certain-optimal policies for quality assurance by setting ξ to a positive value.

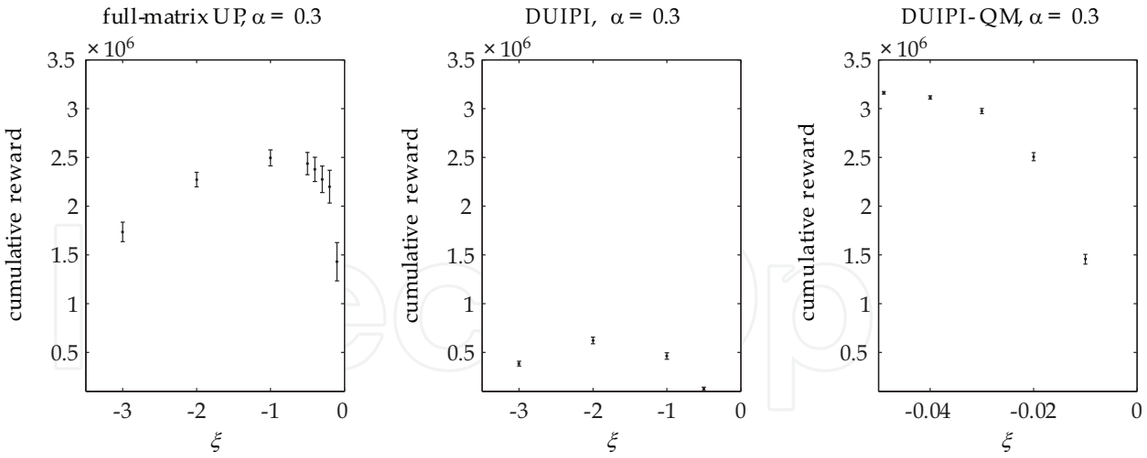


Fig. 5. Cumulative rewards for RiverSwim obtained by the algorithms for various values of ξ. The values for full-matrix UP are averaged over 50 trials, for the values for DUIPI and DUIPI-QM 1000 trials of each experiment were performed.

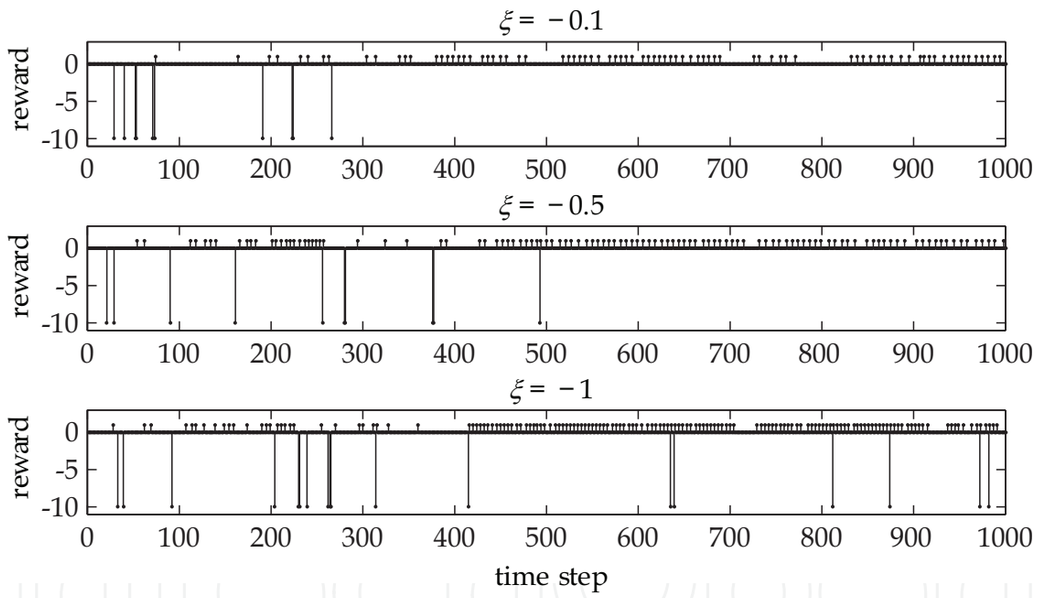


Fig. 6. Immediate rewards of exemplary runs using DUIPI in the Trap domain. When delivering a flag, the agent receives reward 1, when entering the trap state it receives -10. While with $\xi = -0.1$ after less than 300 steps the trap state does not seem worth exploring anymore, setting $\xi = -0.5$ makes the agent explore longer due to uncertainty. With $\xi = -1$ the agent does not stop exploring the trap state in the depicted 1000 time steps.

	full-matrix UP	**DUIPI**	**DUIPI-QM**
time	7 min	14 s	14 s

Table 3. Computation time for 5000 steps in the RiverSwim domain using a single core of an Intel Core 2 Quad Q9550 processor. The policy was updated in every time step.

While the full-matrix UP is the more fundamental and theoretically more sound method, its computational cost is considerable (see table 3). If used with care, however, DUIPI and DUIPI-QM constitute valuable alternatives that proved well in practice. Although our experiments are rather small, we expect DUIPI and DUIPI-QM to also perform well on larger problems.

8.3 Increasing the expected performance

Incorporating uncertainty in RL can even improve the expected performance for concrete MDPs in many practical and industrial environments, where exploration is expensive and only allowed within a small range. The available amount of data is hence small and exploration takes place in an, in part extremely, unsymmetrical way. Data is particularly collected in areas where the operation is already preferable. Many of the insufficiently explored so-called on-border states are undesirable in expectation, but might, by chance, give a high reward in the singular case. If the border is sufficiently large this might happen at least a few times and such an outlier might suggest a high expected reward. Note that in general the size of the border region will increase with the dimensionality of the problem. Carefully incorporating uncertainty avoids the agent to prefer those outliers in its final operation.

We applied the joint iteration on a simple artificial archery benchmark with the "border phenomenon". The state space represents an archer's target (figure 7). Starting in the target's middle, the archer has the possibility to move the arrowhead in all four directions and to shoot the arrow. The exploration has been performed randomly with short episodes. The dynamics were simulated with two different underlying MDPs. The arrowhead's moves are either stochastic (25 percent chance of choosing another action) or deterministic. The event of making a hit after shooting the arrow is stochastic in both settings. The highest probability for a hit is with the arrowhead in the target's middle. The border is explored quite rarely, such that a hit there misleadingly causes the respective estimator to estimate a high reward and thus the agent to finally shoot from this place.

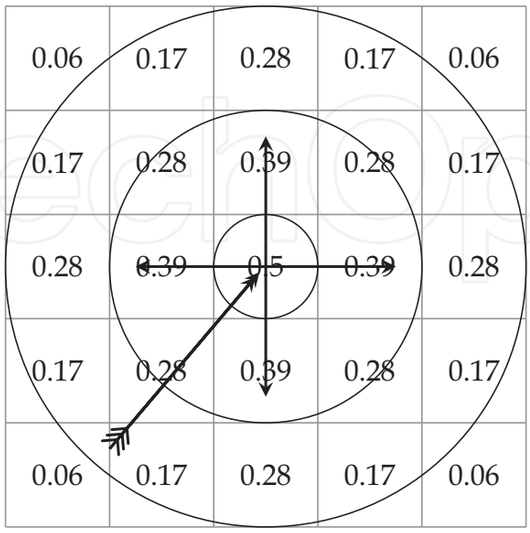


Fig. 7. Visualisation of the archery benchmark. The picture shows the target consisting of its 25 states, together with their hitting probabilities.

Table 4. Average reward for the archery and gas turbine benchmark.

Setting	**Model**	**Discr.**	**# Obs.**	$\xi = 0$	$\xi = 0.5$	$\xi = 1$	$\xi = 2$
Archery	Frequentist		100	0.14	**0.16**	0.13	0.05
(Stochastic)			500	0.17	0.20	**0.25**	0.22
			1000	0.21	0.26	**0.29**	0.27
			2500	0.27	0.29	**0.31**	**0.31**
Archery	Deterministic		100	0.35	**0.38**	0.23	0.17
(Deterministic)	Dirichlet Prior		500	0.32	0.38	0.39	**0.41**
	$\forall i : \alpha_i = 0$		1000	0.35	0.41	0.44	**0.45**
			2500	0.44	0.46	0.48	0.49
Turbine	Frequentist	coarse	10^4	0.736	0.758	0.770	0.815
		medium	10^4	0.751	0.769	0.784	0.816
		fine	10^4	0.767	0.785	0.800	0.826
Turbine	Maximum Entropy	coarse	10^4	0.720	0.767	0.814	0.848
	Dirichlet Prior	medium	10^4	0.713	0.731	0.749	0.777
	$\forall i : \alpha_i = 1$	fine	10^4	0.735	0.773	0.789	**0.800**
	For Comparison			RefCon	RQL	RPS	RFuzzy
Turbine		coarse	10^5		0.680	0.657	0.662
		medium	10^5	0.53	0.687	0.745	0.657
		fine	10^5		0.717	0.729	0.668

In table 4 the performance, averaged over 50 trials (two digits precision), for the frequentist setting (in the stochastic case) and the deterministic prior (in the deterministic case) for the transition probabilities are listed.
The table shows that the performance indeed increases with ξ until a maximum and then decreases rapidly. The position of the maximum apparently increases with the number of observations. This can be explained by the decreasing uncertainty. The performance of the theoretical optimal policy is 0.31 for the stochastic archery benchmark and 0.5 for the deterministic one. They are achieved in average by the certain-optimal policy based on 2500 observations with $1 \leq \xi \leq 2$ in the stochastic case and for $3 \leq \xi \leq 4$ in the deterministic case.

8.4 An industrial application

We further applied the uncertainty propagation together with the joint iteration on an application to gas turbine control (Schaefer et al., 2007) with a continuous state and a finite action space, where it can be assumed that the "border phenomenon" appears as well. We discretised the internal state space with three different precisions (coarse (4^4 = 256 states), medium (5^4 = 625 states), fine (6^4 = 1296 states)), where the high-dimensional state space has already been reduced to a four-dimensional approximate Markovian state space, called "internal state space". A detailed description of the problem and the construction of the internal state space can be found in Schaefer et al. (2007). Note that the Bellman iteration and the uncertainty propagation is computationally feasible even with 6^4 states, since P and Cov((P,R)) are sparse.
We summarise the averaged performances (50 trials with short random episodes starting from different operating points, leading to three digits precision) in table 4 on the same uninformed priors as used in section 8.3. The rewards were estimated with an uninformed normal-gamma distribution as conjugate prior with $\sigma = \infty$ and $\alpha = \beta = 0$.
In contrary to the archery benchmark, we left the number of observations constant and changed the discretisation. The finer the discretisation, the larger is the uncertainty. Therefore the position of the maximum tends to increase with decreasing number of states. The performance is largest using the coarse discretisation. Indeed, averaged over all discretisations, the results for the frequentist setting tend to be better than for the maximum entropy prior. The overall best performance can be achieved with the coarse discretisation and the frequentist setting with $\xi = 5$, but using the maximum entropy prior leads to comparable results even with $\xi = 3$.
The theoretical optimum is not known, but for comparison we show the results of the recurrent Q-learning (RQL), prioritised sweeping (RPS), fuzzy RL (RFuzzy), neural rewards regression (RNRR), policy gradient NRR (RPGNRR), and control neural network (RCNN) (Schaefer et al., 2007; Appl & Brauer, 2002; Schneegass et al., 2007). The highest observed performance is 0.861 using 10^5 observations, which has almost been achieved by the best certain-optimal policy using 10^4 observations.

9. Conclusion

A new approach incorporating uncertainty in RL is presented, following the path from *awareness* to *quantisation* and *control*. We applied the technique of uncertainty propagation

(awareness) not only to understand the reliability of the obtained policies (quantisation) but also to achieve certain-optimality (control), a new optimality criterion in RL and beyond. We exemplarily implemented the methodology on discrete MDPs, but want to stress on its generality, also in terms of the applied statistical paradigm. We demonstrated how to realistically deal with large-scale problems without a substantial loss of performance. In addition, we have shown that the method can be used to guide exploration (control). By changing a single parameter the derived policies change from certain-optimal policies for quality assurance to policies that are certain-optimal in a reversed sense and can be used for information-seeking exploration.

Current and future work considers several open questions as the application to other RL paradigms and function approximators like neural networks and support vector machines. Another important issue is the utilisation of the information contained in the full covariance matrix rather than only the diagonal. This enhancement can be seen as a generalisation of the local to a global measure of uncertainty. It can be shown that the guaranteed minimal performance for a specific selection of states depends on the covariances between the different states, i.e., the non-diagonal entries of the covariance matrix.

Last but not least the application to further industrial environments is strongly aspired. Definitely, as several laboratory conditions, such as the possibility of an extensive exploration or the access on a sufficiently large number of observations, are typically not fulfilled in practice, we conclude that the knowledge of uncertainty and its intelligent utilisation in RL is vitally important to handle control problems of industrial scale.

10. References

Abbeel, P., Coates, A., Quigley, M. & Ng, A. Y. (2006). An application of reinforcement learning to aerobatic helicopter flight, *Proc. of the 20th Conference on Neural Information Processing Systems*, MIT Press, pp. 1–8.

Antos, A., Szepesvári, C. & Munos, R. (2006). Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path, *Proc. of the Conference on Learning Theory*, pp. 574–588.

Appl, M. & Brauer, W. (2002). Fuzzy model-based reinforcement learning, *Advances in Computational Intelligence and Learning*, pp. 211–223.

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

Brafman, R. I. & Tennenholtz, M. (2003). R-Max - a general polynomial time algorithm for near-optimal reinforcement learning, *Journal of Machine Learning Research* 3: 213–231.

Coppersmith, D. & Winograd, S. (1990). Matrix multiplication via arithmetic progressions, *Journal of Symbolic Computation* 9: 251–280.

D'Agostini, G. (2003). *Bayesian Reasoning in Data Analysis: A Critical Introduction*, World Scientific Publishing.

Dearden, R., Friedman, N. & Andre, D. (1999). Model based Bayesian exploration, *Proc. of the Conference on Uncertainty in Artificial Intelligence*, pp. 150–159.

Dearden, R., Friedman, N. & Russell, S. J. (1998). *Bayesian Q-learning, Proc. of the Innovative Applications of Artificial Intelligence Conference of the Association for the Advancement of Artificial Intelligence*, pp. 761–768.

Delage, E. & Mannor, S. (2007). Percentile optimization in uncertain Markov decision processes with application to efficient exploration, *Proc. of the International Conference on Machine Learning*, pp. 225–232.

Engel, Y., Mannor, S. & Meir, R. (2003). Bayes meets Bellman: The Gaussian process approach to temporal difference learning, *Proc. of the International Conference on Machine Learning*, pp. 154–161.

Engel, Y., Mannor, S. & Meir, R. (2005). Reinforcement learning with Gaussian processes, *Proc. of the International Conference on Machine learning*, pp. 201–208.

Geibel, P. (2001). Reinforcement learning with bounded risk, *Proc. of the 18th International Conference on Machine Learning*, Morgan Kaufmann, San Francisco, CA, pp. 162–169.

Ghavamzadeh, M. & Engel, Y. (2006). Bayesian policy gradient algorithms, *Advances in Neural Information Processing Systems 19*, pp. 457–464.

Ghavamzadeh, M. & Engel, Y. (2007). Bayesian actor-critic algorithms, *Proc. of the International Conference on Machine learning*, pp. 297–304.

Hans, A. & Udluft, S. (2009). Efficient uncertainty propagation for reinforcement learning with limited data, *Proc. of the International Conference on Artificial Neural Networks*, Springer, pp. 70–79.

Hans, A. & Udluft, S. (2010). Uncertainty propagation for efficient exploration in reinforcement learning, *Proc. of the European Conference on Artificial Intelligence*

Heger, M. (1994). Consideration of risk in reinforcement learning, *Proc. 11th International Conference on Machine Learning*, Morgan Kaufmann, pp. 105–111.

ISO (1993). *Guide to the Expression of Uncertainty in Measurement*, International Organization for Standardization.

Kaelbling, L. P., Littman, M. L. & Moore, A. W. (1996). Reinforcement learning: A survey, *Journal of Artificial Intelligence Research* 4: 237–285.

Kearns, M., Mansour, Y. & Ng, A. Y. (2000). Approximate planning in large POMDPs via reusable trajectories, *Advances in Neural Information Processing Systems 12*.

Kearns, M. & Singh, S. (1998). Near-optimal reinforcement learning in polynomial time, *Proceedings of the 15th International Conference on Machine Learning*, pp. 260–268.

Lagoudakis, M. G. & Parr, R. (2003). Least-squares policy iteration, *Journal of Machine Learning Research* pp. 1107–1149.

Lee, H., Shen, Y., Yu, C.-H., Singh, G. & Ng, A. Y. (2006). Quadruped robot obstacle negotiation via reinforcement learning, *Proc. of the 2006 IEEE International Conference on Robotics and Automation, ICRA 2006, May 15-19, 2006, Orlando, Florida, USA*, pp. 3003–3010.

MacKay, D. J. C. (2003). *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, Cambridge.

Merke, A. & Riedmiller, M. A. (2001). Karlsruhe brainstormers - a reinforcement learning approach to robotic soccer, *RoboCup 2001: Robot Soccer World Cup V*, Springer, pp. 435– 440.

Mihatsch, O. & Neuneier, R. (2002). Risk-sensitive reinforcement learning, *Machine Learning* 49(2–3): 267–290.

Munos, R. (2003). Error bounds for approximate policy iteration., *Proc. of the International Conference on Machine Learning*, pp. 560–567.

Peshkin, L. & Mukherjee, S. (2001). Bounds on sample size for policy evaluation in Markov environments, *Proc. of Annual Conference on Computational Learning Theory, COLT and the European Conference on Computational Learning Theory*, Vol. 2111, Springer, Berlin, pp. 616–629.

Peters, J. & Schaal, S. (2008). Reinforcement learning of motor skills with policy gradients, *Neural Networks* 21(4): 682–697.

Poupart, P., Vlassis, N., Hoey, J. & Regan, K. (2006). An analytic solution to discrete Bayesian reinforcement learning, *Proc. of the International Conference on Machine Learning*, pp. 697–704.

Puterman, M. L. (1994). *Markov Decision Processes*, John Wiley & Sons, New York.

Rasmussen, C. E. & Kuss, M. (2003). Gaussian processes in reinforcement learning, *Advances in Neural Information Processing Systems 16*, pp. 751–759.

Schaefer, A. M., Schneegass, D., Sterzing, V. & Udluft, S. (2007). A neural reinforcement learning approach to gas turbine control, *Proc. of the International Joint Conference on Neural Networks*.

Schneegass, D., Udluft, S. & Martinetz, T. (2007). Improving optimality of neural rewards regression for data-efficient batch near-optimal policy identification, *Proc. of the International Conference on Artificial Neural Networks*, pp. 109–118.

Schneegass, D., Udluft, S. & Martinetz, T. (2008). Uncertainty propagation for quality assurance in reinforcement learning, *Proc. of the International Joint Conference on Neural Networks*, pp. 2589–2596.

Stephan, V., Debes, K., Gross, H.-M., Wintrich, F. & Wintrich, H. (2000). A reinforcement learning based neural multi-agent-system for control of a combustion process, *Proc. of the International Joint Conference on Neural Networks*, pp. 217–222.

Strehl, A. L. & Littman, M. L. (2008). An analysis of model-based interval estimation for Markov decision processes., *Journal of Computer and System Sciences* 74(8): 1309–1331.

Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*, MIT Press, Cambridge.

Wiering, M. & Schmidhuber, J. (1998). Efficient model-based exploration, *Proceedings of the 5th International Conference on Simulation of Adaptive Behavior: From Animals to Animats 5*, MIT Press/Bradford Books, Montreal, pp. 223–228.

Appendix

Theorem 1 *Suppose a finite MDP* $M = (S,A,P,R)$ *with discount factor* $0 < \gamma < 1$ *and* C^0 *an arbitrary initial symmetric and positive definite covariance matrix. Then the function*

$$\left(Q^m, C^m\right) = \left(TQ^{m-1}, D^{m-1}C^{m-1}(D^{m-1})^T\right) \tag{35}$$

provides a unique fixed point (Q^*,C^*) *almost surely, independent of the initial Q, for policy evaluation and policy iteration.*

Proof: It has already been shown that $Q^m = TQ^{m-1}$ converges to a unique fixed point Q^* (Sutton & Barto, 1998). Since Q^m does not depend on C^k or the Jacobi matrix D^k for any

iteration $k<m$, it remains to show that C^* unambiguously arises from the fixed point iteration. We obtain

$$C^m = \prod_{i=0}^{m-1} D^i C^0 \prod_{i=0}^{m-1} (D^i)^T \tag{36}$$

after m iterations. Due to convergence of Q^m, D^m converges to D^* as well, which leads to

$$C^* = \prod_{i=0}^{\infty} D^* C_{\text{conv}} \prod_{i=0}^{\infty} (D^*)^T \tag{37}$$

with C_{conv} the covariance matrix after convergence of Q. By successive matrix multiplication we obtain

$$(D^*)^n = \begin{pmatrix} (D^*)^n_{Q,Q} & \sum_{i=0}^{n} (D^*)^i_{Q,Q} (D^*)_{Q,P} & \sum_{i=0}^{n} (D^*)^i_{Q,Q} (D^*)_{Q,R} \\ 0 & \mathrm{I} & 0 \\ 0 & 0 & \mathrm{I} \end{pmatrix} \tag{38}$$

eventually leading to

$$(D^*)^\infty = \begin{pmatrix} (D^*)^\infty_{Q,Q} & \sum_{i=0}^{\infty} (D^*)^i_{Q,Q} (D^*)_{Q,P} & \sum_{i=0}^{\infty} (D^*)^i_{Q,Q} (D^*)_{Q,R} \\ 0 & \mathrm{I} & 0 \\ 0 & 0 & \mathrm{I} \end{pmatrix} \tag{39}$$

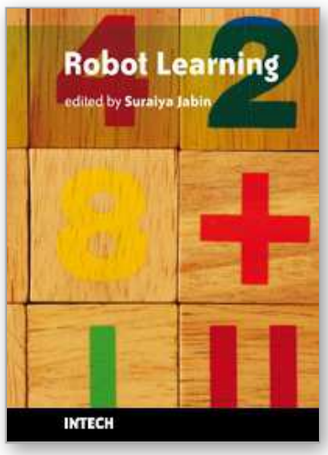
$$= \begin{pmatrix} 0 & (\mathrm{I} - (D^*)_{Q,Q})^{-1} (D^*)_{Q,P} & (\mathrm{I} - (D^*)_{Q,Q})^{-1} (D^*)_{Q,R} \\ 0 & \mathrm{I} & 0 \\ 0 & 0 & \mathrm{I} \end{pmatrix} \tag{40}$$

since all eigenvalues of $(D^*)_{Q,Q}$ are strictly smaller than 1 and $\mathrm{I} - (D^*)_{Q,Q}$ is invertible for all but finitely many $(D^*)_{Q,Q}$. Therefore, almost surely, $(D^*)^\infty$exists, which implies that C^* exists as well. We finally obtain

$$C^*_{Q,Q} = (\mathrm{I} - (D^*)_{Q,Q})^{-1} \begin{pmatrix} (D^*)_{Q,P} & (D^*)_{Q,R} \end{pmatrix} \tag{41}$$

$$\begin{pmatrix} \mathrm{Cov}(P,P) & \mathrm{Cov}(P,R) \\ \mathrm{Cov}(P,R)^T & \mathrm{Cov}(R,R) \end{pmatrix} \begin{pmatrix} (D^*)^T_{Q,P} \\ (D^*)^T_{Q,R} \end{pmatrix} \left((\mathrm{I} - (D^*)_{Q,Q})^{-1} \right)^T . \tag{42}$$

The fixed point C^* depends on the initial covariance matrices Cov(P), Cov(R), and Cov(P,R) solely, but not on Cov(Q,Q), Cov(Q,P), or Cov(Q,R) and is therefore independent of the operations necessary to reach the fixed point Q^*. □



Robot Learning

Edited by Suraiya Jabin

ISBN 978-953-307-104-6
Hard cover, 150 pages
Publisher Sciyo
Published online 12, August, 2010
Published in print edition August, 2010

Robot Learning is intended for one term advanced Machine Learning courses taken by students from different computer science research disciplines. This text has all the features of a renowned best selling text. It gives a focused introduction to the primary themes in a Robot learning course and demonstrates the relevance and practicality of various Machine Learning algorithms to a wide variety of real-world applications from evolutionary techniques to reinforcement learning, classification, control, uncertainty and many other important fields. Salient features: - Comprehensive coverage of Evolutionary Techniques, Reinforcement Learning and Uncertainty. - Precise mathematical language used without excessive formalism and abstraction. - Included applications demonstrate the utility of the subject in terms of real-world problems. - A separate chapter on Anticipatory-mechanisms-of-human-sensory-motor-coordination and biped locomotion. - Collection of most recent research on Robot Learning.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Daniel Schneegass, Alexander Hans and Steffen Udluft (2010). Uncertainty in Reinforcement Learning - Awareness, Quantisation, and Control, Robot Learning, Suraiya Jabin (Ed.), ISBN: 978-953-307-104-6, InTech, Available from: http://www.intechopen.com/books/robot-learning/uncertainty-in-reinforcement-learning-awareness-quantisation-and-control

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen