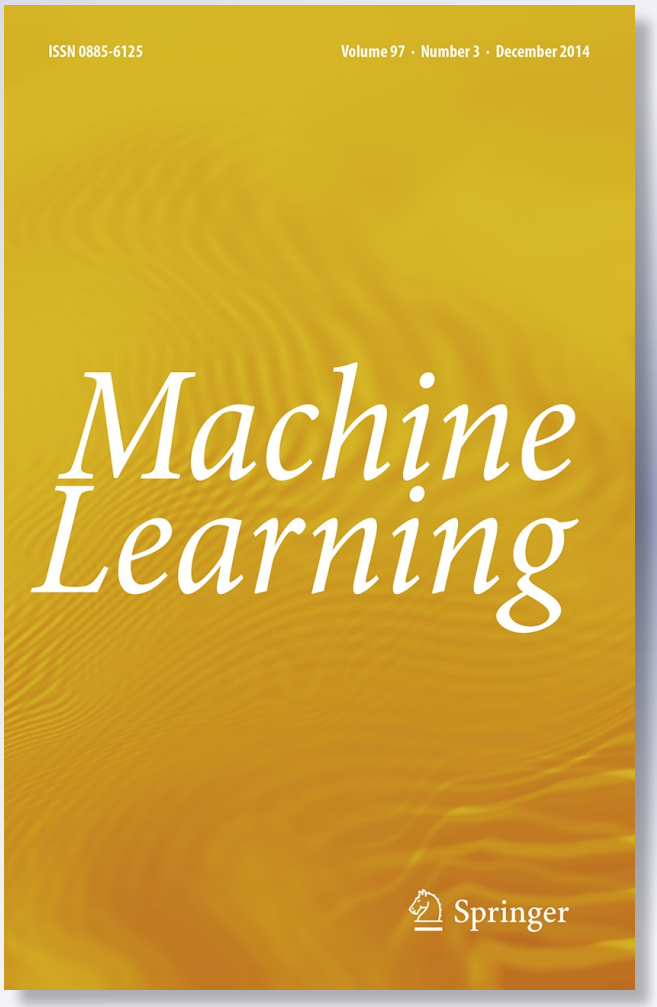
Preference-based reinforcement learning: evolutionary direct policy search using a preference-based racing algorithm

Róbert Busa-Fekete, Balázs Szörényi, Paul Weng, Weiwei Cheng & Eyke Hüllermeier

Machine Learning

ISSN 0885-6125
Volume 97
Number 3

Mach Learn (2014) 97:327-351
DOI 10.1007/s10994-014-5458-8



Springer

Mach Learn (2014) 97:327–351
DOI 10.1007/s10994-014-5458-8

Preference-based reinforcement learning: evolutionary direct policy search using a preference-based racing algorithm

Róbert Busa-Fekete · Balázs Szörényi · Paul Weng · Weiwei Cheng · Eyke Hüllermeier

Received: 3 March 2013 / Accepted: 6 June 2014 / Published online: 2 July 2014

Abstract We introduce a novel approach to preference-based reinforcement learning, namely a preference-based variant of a direct policy search method based on evolutionary optimization. The core of our approach is a preference-based racing algorithm that selects the best among a given set of candidate policies with high probability. To this end, the algorithm operates on a suitable ordinal preference structure and only uses pairwise comparisons between sample rollouts of the policies. Embedding the racing algorithm in a rank-based evolutionary search procedure, we show that approximations of the so-called Smith set of optimal policies can be produced with certain theoretical guarantees. Apart from a formal performance and complexity analysis, we present first experimental studies showing that our approach performs well in practice.

Editors: Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, and Filip Železný.

R. Busa-Fekete · W. Cheng · E. Hüllermeier
Computational Intelligence Group, Department of Mathematics and Computer Science,
University of Marburg, Marburg, Germany

W. Cheng
e-mail: cheng@mathematik.uni-marburg.de

E. Hüllermeier
e-mail: eyke@mathematik.uni-marburg.de

R. Busa-Fekete (✉) B. Szörenyi
MTA-SZTE Research Group on Artificial Intelligence, Tisza Lajos Krt. 103, 6720 Szeged, Hungary
e-mail: busarobi@inf.u-szeged.hu

B. Szörenyi
INRIA Lille - Nord Europe, SequeL project, 40 avenue Halley,
59650 Villeneuve d'Ascq, France
e-mail: szorenyi@inf.u-szeged.hu

P. Weng
Sorbonne Université/s, UPMC Univ Paris 06, UMR 7606, LIP6,
4 Place Jussieu, 75005 Paris, France
e-mail: paul.weng@lip6.fr

Keywords Preference learning · Reinforcement learning · Evolutionary direct policy search · Racing algorithms

1 Introduction

Preference-based reinforcement learning (PBRL) is a novel research direction combining reinforcement learning (RL) and preference learning (Fürnkranz and Hüllermeier 2011). It aims at extending existing RL methods so as to make them amenable to training information and external feedback more general than numerical rewards, which are often difficult to obtain or expensive to compute. For example, anticipating our experimental study in the domain of medical treatment planning, to which we shall return in Sect. 5, how to specify the cost of a patient's death in terms of a reasonable numerical value?

In Akrour et al. (2011) and Cheng et al. (2011), the authors tackle the problem of learning policies solely on the basis of qualitative preference information, namely pairwise comparisons between histories; such comparisons suggest that one system behavior is preferred to another one, but without committing to precise numerical rewards. Building on novel methods for preference learning, this is accomplished by providing the RL agent with qualitative policy models, such as ranking functions. More specifically, Cheng et al. (2011) use a method called *label ranking* to train a model that ranks actions given a state; their approach generalizes classification-based approximate policy iteration (Lagoudakis and Parr 2003). Instead of ranking actions given states, Akrour et al. (2011) learn a preference model on histories, which can then be used for policy optimization.

In this paper, we present a preference-based extension of *evolutionary direct policy search* (EDPS) as proposed by Heidrich-Meisner and Igel (2009, 2009). As a direct policy search method, it shares commonalities with Akrour et al. (2011), but also differs in several respects. In particular, the latter approach (as well as follow-up work of the same authors, such as Akrour et al. (2012)) is specifically tailored for applications in which a user interacts with the learner in an iterative process. Moreover, policy search is not performed in a parametrized policy space directly; instead, preferences on histories are learned in a feature space, in which each history is represented in terms of a feature vector, thereby capturing important background knowledge about the task to be solved.

EDPS casts policy learning as a search problem in a parametric policy space, where the function to be optimized is a performance measure like expected total reward, and evolution strategies (ES) such as CMA- ES (Hansen and Kern 2004; Ostermeier et al. 1994) are used as optimizers. Moreover, since the evaluation of a policy can only be done approximately, namely in terms of a finite number of *rollouts*, the authors make use of *racing algorithms* to control this number in an adaptive manner. These algorithms return a sufficiently reliable ranking over the current set of policies (candidate solutions), which is then used by the ES for updating its parameters and population. A key idea of our approach is to extend EDPS by replacing the *value-based* racing algorithm with a *preference-based* one. Correspondingly, the development of a preference-based racing algorithm can be seen as a core contribution of this paper.

In the next section, we recall the original RL setting and the EDPS framework for policy learning. Our preference-based generalization of this framework is introduced in Sect. 3. A key component of our approach, the preference-based racing algorithm, is detailed and analyzed in Sect. 4. Experiments are presented in Sect. 5. Section 6 provides an overview of related work and Sect. 7 concludes the paper.

2 Evolutionary direct policy search

We start by introducing notation to be used throughout the paper. A Markov Decision Process (MDP) is a 4-tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathbf{P}, r)$, where $\mathcal{S}$ is the (possibly infinite) state space and $\mathcal{A}$ the (possibly infinite) set of actions. We assume that $(\mathcal{S}, \Sigma_{\mathcal{S}})$ and $(\mathcal{A}, \Sigma_{\mathcal{A}})$ are measurable spaces. Moreover,

$$\mathbf{P} : \mathcal{S} \times \mathcal{A} \times \Sigma_S \to [0, 1]$$

is the transition probability kernel that defines the random transitions between states, depending on the action taken. Thus, for each (measurable) $S \in \Sigma_S \subseteq 2^{\mathcal{S}}$, $\mathbf{P}(S \mid \mathbf{s}, a) = \mathbf{P}(\mathbf{s}, a, S)$ is the probability to reach a state $\mathbf{s}' \in S$ when taking action $a \in \mathcal{A}$ in state $\mathbf{s} \in \mathcal{S}$; for singletons $\mathbf{s}' \in \mathcal{S}$, we simply write $\mathbf{P}(\mathbf{s}' \mid \mathbf{s}, a)$ instead of $\mathbf{P}(\{\mathbf{s}'\} \mid \mathbf{s}, a)$. Finally, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the *reward function*, i.e., $r(\mathbf{s}, a)$ defines the reward for choosing action $a \in \mathcal{A}$ in state $\mathbf{s} \in \mathcal{S}$.

We will only consider *undiscounted* and *episodic* MDPs with a finite horizon $T \in \mathbb{N}^+$. In the episodic setup, there is a set of *initial* states $\mathcal{S}_0 \subseteq \mathcal{S}$. $\mathcal{H}^{(T)} = \mathcal{S}_0 \times (\mathcal{A} \times \mathcal{S})^T$ is the set of *histories* with time horizon at most T. A *finite history* or simply *history* is a state/action sequence

$$\mathbf{h} = \left(\mathbf{s}^{(0)}, a^{(1)}, \ldots, a^{(T)}, \mathbf{s}^{(T)}\right) \in \mathcal{H}^{(T)}$$

that starts from an initial state $\mathbf{s}^{(0)} \in \mathcal{S}_0$ drawn from a user-defined *initial state distribution* $\mathbf{P}_0$ over $\mathcal{S}_0$. As a side note, MDPs with terminal states fit in this framework by defining transition functions in terminal states such that those terminal states are repeated at the end of a history (to have exactly length T) if a terminal state is reached before the end of the horizon. Since each history $\mathbf{h}$ uniquely determines a sequence of rewards, a *return function* $V : \mathcal{H}^{(T)} \to \mathbb{R}$ can be defined as

$$V(\mathbf{h}) = \sum_{i=1}^{T} r\left(\mathbf{s}^{(i-1)}, a^{(i)}\right) .$$

A (deterministic) *policy* $\pi : \mathcal{S} \to \mathcal{A}$ prescribes an action to be chosen for each state. We write $\mathbf{h}_\pi$ for a history that was generated by following the policy π, that is, $\pi(\mathbf{s}^{(t-1)}) = a^{(t)}$ for all $t \in \{1, \ldots, T\}$.

2.1 The EDPS framework

We briefly outline the *evolutionary direct policy search* (EDPS) approach introduced by Heidrich-Meisner and Igel (2009). Assume a parametric policy space

$$\Pi = \{\pi_\theta \mid \theta \in \mathbb{R}^p\},$$

i.e., a space of policies parametrized by a vector θ. For example, if $\mathcal{S} \subseteq \mathbb{R}^p$, this could simply be a class of linear policies $\pi_\theta(\mathbf{s}) = \theta^T \mathbf{s}$. Searching a good policy can be seen as an optimization problem where the search space is the parameter space and the target function is a policy performance evaluation, such as expected total reward.

This optimization-based policy search framework, which is called *direct policy search*, has two main branches: *gradient-based* and *gradient-free* methods. Gradient-based methods like the REINFORCE algorithm (Williams 1992) estimate the gradient of the policy parameters to guide the optimizer. Gradient-free methods, on the other hand, make use of a *black-box optimizer* such as *evolution strategies* (Beyer and Schwefel 2002), which gave rise to the EDPS approach.

2.2 Evolutionary optimization

Evolution strategies (ES) are population-based, randomized search techniques that maintain a set of candidate solutions $\theta_1, \ldots, \theta_\mu$ (the population) and a set of (auxiliary) parameters Ω over the search space. An ES optimizer is an iterative method that repeats the following steps in each iteration t:

(i) sample a set of λ candidate solutions $\{\theta_j^{(t+1)}\}_{j=1}^{\lambda}$, called *offspring population*, from the current model defined by $\Omega^{(t)}$ and the *parent population* $\{\theta_i^{(t)}\}_{i=1}^{\mu}$;
(ii) evaluate each offspring solution and select the best μ ones as a new parent population;
(iii) update $\Omega^{(t)}$ based on the new parent population.

The use of evolution strategies proved to be efficient in direct policy search (Heidrich-Meisner and Igel 2008). In the EDPS method by Heidrich-Meisner and Igel (2009), an ES is applied for optimizing the *expected total reward* over the parameter space of linear policies. To this end, the expected total reward of a policy is estimated based on a so-called *rollout set*. More specifically, for an MDP $\mathcal{M}$ with initial distribution $\mathbf{P}_0$, each policy π generates a probability distribution $\mathbf{P}_\pi$ over the set of histories $\mathcal{H}^{(T)}$. Then, the *expected total reward* of π can be written as $\rho_\pi = \mathbb{E}_{\mathbf{h}\sim\mathbf{P}_\pi}[V(\mathbf{h})]$ (Puterman 1994), and the expectation according to $\mathbf{P}_\pi$ can be estimated by the average return over a rollout set $\{\mathbf{h}_\pi^{(i)}\}_{i=1}^{n}$.

From a practical point of view, the size of the rollout set is very important: On the one hand, the learning process gets slow if n is large, while on the other hand, the ranking over the offspring population is not reliable enough if the number of rollouts is too small; in that case, there is a danger of selecting a suboptimal subset of the offspring population instead of the best μ ones. Therefore, Heidrich-Meisner and Igel (2009) proposed to apply an adaptive uncertainty handling scheme, called *racing algorithm*, for controlling the size of rollout sets in a optimal way.

Their EDPS framework is described schematically in Algorithm 1. It bears a close resemblance to ES, but the selection step (line 7) is augmented with a racing algorithm that generates histories for each of the current policies $\pi_{\theta_i^{(t)}}$ by sampling from the corresponding distribution in an adaptive manner until being able to select the best μ policies based on their expected total reward estimates with probability at least $1-\delta$ (see Sect. 2.3). The parameter $n_{\max}$ specifies an upper bound on the number of rollouts for a single policy. The racing algorithm returns a ranking over the policies in the form of a permutation σ.

Algorithm 1 EDPS $(\mathcal{M}, \mu, \lambda, n_{\max}, \delta)$

1: Initialization: select an initial parameter vector $\Omega^{(0)}$ and an initial set of candidate solutions ${\theta_1}^{(0)}, \ldots, {\theta_\mu}^{(0)}$, $\sigma^{(0)}$ is the identity permutation
2: $t = 0$
3: **repeat**
4: $\quad t = t + 1$
5: $\quad$ **for** $\ell = 1, \ldots, \lambda$ **do** ▷ Sample new solutions
6: $\quad\quad \theta_\ell^{(t)} \sim F(\Omega^{(t-1)}, \theta^{(t-1)}_{\sigma^{(t-1)}(1)}, \ldots, \theta^{(t-1)}_{\sigma^{(t-1)}(\mu)})$
7: $\quad \sigma^{(t)} = \textbf{Racing}\left(\mathcal{M}, \pi_{\theta_1^{(t)}}, \ldots, \pi_{\theta_\lambda^{(t)}}, \mu, n_{\max}, \delta\right)$
8: $\quad \Omega^{(t)} = \textbf{Update}(\Omega^{(t-1)}, \theta^{(t)}_{\sigma^{(t)}(1)}, \ldots, \theta^{(t)}_{\sigma^{(t)}(\mu)})$
9: **until** Stopping criterion fulfilled
10: **return** $\pi_{\theta_1^{(t)}}$

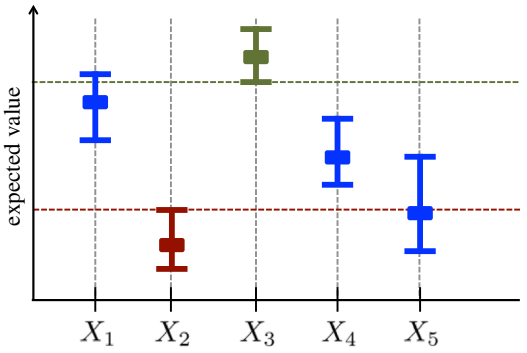


Fig. 1 Illustration of the value-based racing problem: The expectations of the random variables are estimated in terms of confidence intervals that shrink in the course of time. In this example, if two options ought to be selected, then X_2 can be discarded, as it is already worse than three other options (with high probability); likewise, the option X_3 will certainly be an element of the top-2 selection, as it has already outperformed three others. For the other options, a decision cannot yet be made

2.3 Value-based racing

Generating a history in an MDP by following policy π is equivalent to drawing an example from $\mathbf{P}_\pi$. Consequently, a policy along with an MDP and initial distribution can simply be seen as a random variable. Therefore, to make our presentation of the racing algorithm more general, we shall subsequently consider the problem of comparing random variables.

Let $X_1, \ldots, X_K$ be random variables with respective (unknown) distribution functions $\mathbf{P}_{X_1}, \ldots, \mathbf{P}_{X_K}$. These random variables, subsequently also called *options*, are supposed to have finite expected values $\mu_i = \int x d\mathbf{P}_{X_i}(x)$. The racing task consists of selecting, with a predefined confidence $1 - \delta$, a κ-sized subset of the K options with highest expectations. In other words, one seeks a set $I^* \subseteq [K] = \{1, \ldots, K\}$ of cardinality κ maximizing $\sum_{i \in I} \mu_i$, which is equivalent to the following optimization problem:

$$I^* \in \operatorname*{argmax}_{I \subseteq [K]: |I| = \kappa} \sum_{i \in I} \sum_{j \neq i} \mathbb{I}\{\mu_j < \mu_i\}, \tag{1}$$

where the indicator function $\mathbb{I}\{\cdot\}$ maps truth degrees to $\{0, 1\}$ in the standard way. This choice problem must be solved on the basis of random samples drawn from $X_1, \ldots, X_K$.

The Hoeffding race (HR) algorithm (Maron and Moore 1994, 1997) is an adaptive sampling method that makes use of the Hoeffding bound to construct confidence intervals for the empirical mean estimates of the options. Then, in the case of non-overlapping confidence intervals, some options can be eliminated from further sampling. More precisely, if the upper confidence bound for a particular option is smaller than the lower bound of $K - \kappa$ random variables, then it is not included by the solution set I^* in (1) with high probability; the inclusion of an option in I^* can be decided analogously (see Fig. 1 for an illustration). For a detailed implementation of the HR algorithm, see Heidrich-Meisner and Igel (2009).

3 Preference-based EDPS

In Sect. 3.1, we describe an ordinal decision model for comparing policies and discuss some of its decision-theoretic properties. In Sect. 3.2, we analyze this model in the context of Markov Decision Processes.

3.1 Ordinal decision models

The preference-based policy learning settings considered in Fürnkranz et al. (2012), Akrour et al. (2011) proceed from a (possibly partial) preference relation $\succ$ over histories $\mathbf{h} \in \mathcal{H}^{(T)}$, and the goal is to find a policy which tends to generate preferred histories with high probability. In this regard, it is notable that, in the EDPS framework, the precise values of the function to be optimized (in this case the expected total rewards) are actually not used by the evolutionary optimizer. Instead, for updating its current state $(\Omega, \theta_1, \ldots, \theta_\mu)$, the ES only needs the *ranking* of the candidate solutions. The values are only used by the racing algorithm in order to produce this ranking. Consequently, an obvious approach to realizing the idea of a purely preference-based version of evolutionary direct policy search (PB-EDPS) is to replace the original racing algorithm (line 7) by a preference-based racing algorithm that only uses pairwise comparisons between policies (or, more specifically, sample histories generated from these policies). We introduce a racing algorithm of this kind in Sect. 4.

A main prerequisite of such an algorithm is a "lifting" of the preference relation $\succ$ on $\mathcal{H}^{(T)}$ to a preference relation $\gg$ on the space of policies Π; in fact, without a relation of that kind, the problem of ranking policies is not even well-defined. More generally, recalling that we can associate policies with random variables X and histories with realizations $x \in \Xi$, the problem can be posed as follows: Given a (possibly partial) order relation $\succ$ on the set of realizations Ξ, how to define a reasonable order relation on the set of probability distributions over Ξ which is "learnable" by a preference-based racing algorithm?

A natural definition of the preference relation $\gg$ that we shall adopt in this paper is as follows:

$$X \gg Y \text{ if and only if } \mathbf{P}(X \succ Y) > \mathbf{P}(Y \succ X)\,,$$

where $\mathbf{P}(X \succ Y)$ denotes the probability that the realization of X is preferred (with respect to $\succ$) to the realization of Y. We write $X \underline{\gg} Y$ for $X \gg Y$ or $\mathbf{P}(X \succ Y) = \mathbf{P}(Y \succ X)$.

Despite the appeal of $\gg$ as an ordinal decision model, this relation does not immediately solve our ranking task, mainly because it is not necessarily transitive and may even have cycles (Fishburn 1982). The preferential structure induced by $\gg$ is well-studied in social choice theory (Moulin 1988), as it is closely related to the idea of choosing a winner in an election where only pairwise comparisons between candidates are available. We borrow two important notions from social choice theory, namely the *Condorcet winner* and the *Smith set*; in the following, we define these notions in the context of our setting.

Definition 1 A random variable X_i is a Condorcet winner among a set of random variables $X_1, \ldots, X_K$ if $X_i \underline{\gg} X_j$ for all j.

Definition 2 For a set of random variables $\mathcal{X} = \{X_1, \ldots, X_K\}$, the Smith set is the smallest non-empty set $\mathcal{C}^* \subseteq \mathcal{X}$ satisfying $X_i \gg X_j$ for all $X_i \in \mathcal{C}^*$ and $X_j \in \mathcal{X} \setminus \mathcal{C}^*$.

If a Condorcet winner X^* exists, then it is a greatest element of $\underline{\gg}$ and $\mathcal{C}^* = \{X^*\}$. More generally, the Smith set $\mathcal{C}^*$ can be interpreted as the smallest non-empty set of options that are "better" than all options outside $\mathcal{C}^*$.

Due to preferential cycles, the (racing) problem of selecting the κ best options may still not be well-defined for $\gg$ as the underlying preference relation. To overcome this difficulty, we refer to the *Copeland relation* $\gg_C$ as a surrogate. For a set $\mathcal{X} = \{X_1, \ldots, X_K\}$ of random variables, it is defined as follows (Moulin 1988): $X_i \gg_C X_j$ if and only if $d_i > d_j$, where $d_i = \#\{k \mid X_i \gg X_k, X_k \in \mathcal{X}\}$. Its interpretation is again simple: an option X_i is preferred to X_j whenever X_i "beats" (w.r.t. $\gg$) more options than X_j does. Since the preference relation

$\gg_C$ has a numeric representation in terms of the d_i, it is a total preorder. Note that $\gg_C$ is "contextualized" by the set $\mathcal{X}$ of random variables: the comparison of two options X_i and X_j, i.e., whether or not $X_i \gg_C X_j$, also depends on the other alternatives in $\mathcal{X}$.

Obviously, when a Condorcet winner exists, it is the greatest element for $\gg_C$. More generally, the following proposition, which is borrowed from Laslier (1997), establishes an important connection between $\gg$ and $\gg_C$ and legitimates the use of the latter as a surrogate of the former.

Proposition 3 *Let $\mathcal{X} = \{X_1, \ldots, X_K\}$ be a set of random variables with Smith set $\mathcal{C}^*$. Then, for any $X_i \in \mathcal{C}^*$ and $X_j \in \mathcal{X} \setminus \mathcal{C}^*$, $X_i \gg_C X_j$.*

Proof Let $K_{\mathcal{C}^*}$ be the size of $\mathcal{C}^*$. By the definition of the Smith set, $d_i \geq K - K_{\mathcal{C}^*}$ for all $X_i \in \mathcal{C}^*$, since X_i beats all elements of $\mathcal{X} \setminus \mathcal{C}^*$ w.r.t $\gg$. Moreover, $d_j < K - K_{\mathcal{C}^*}$ for all $X_j \in \mathcal{X} \setminus \mathcal{C}^*$, since X_j is beaten by all elements of $\mathcal{C}^*$. Therefore, $d_i > d_j$ for any $X_i \in \mathcal{C}^*$ and $X_j \in \mathcal{X} \setminus \mathcal{C}^*$. □

Therefore, the surrogate relation $\gg_C$ is coherent with the preference order $\gg$ in the sense that the "rational choices", namely the elements of the Smith set, are found on the top of this preorder. In the next section, we shall therefore use $\gg_C$ as an appropriate ordinal decision model for preference-based racing.

3.2 The existence of a Condorcet winner for parametric policy spaces

Recall that our decision model for policies can be written as follows:

$$\pi \gg \pi' \text{ if and only if } S(\mathbf{P}_\pi, \mathbf{P}_{\pi'}) > S(\mathbf{P}_{\pi'}, \mathbf{P}_\pi)\,,$$

where

$$S(\mathbf{P}_\pi, \mathbf{P}_{\pi'}) = \mathop{\mathbb{E}}_{\mathbf{h}\sim\mathbf{P}_\pi, \mathbf{h}'\sim\mathbf{P}_{\pi'}} \left(\mathbb{I}\{\mathbf{h} \succ \mathbf{h}'\}\right)$$

Based on Definition 1, a parametric policy π_θ is a Condorcet winner among $\Pi = \{\pi_\theta \mid \theta \in \Theta\}$, where Θ is a subset of $\mathbb{R}^p$, if $\pi_\theta \gtrdot \pi_{\theta'}$ for all $\theta' \in \Theta$. Although a Condorcet winner does not exist in general (since $\gtrdot$ over policies may be cyclic), we now discuss two situations in which its existence is guaranteed. To this end, we need to make a few additional assumptions.

- **(D)** State space $\mathcal{S}$ is denumerable.
- **(C1)** Transition probabilities $\mathbf{P}(\mathbf{s}' \mid \mathbf{s}, a)$, seen as functions $a \mapsto \mathbf{P}(\mathbf{s}' \mid \mathbf{s}, a)$ of action a for arbitrary but fixed $\mathbf{s}$ and $\mathbf{s}'$, are continuous functions.
- **(C2)** Policies $\pi_\theta(\mathbf{s})$, seen as functions $\theta \mapsto \pi_\theta(\mathbf{s})$ of parameter θ for arbitrary but fixed $\mathbf{s}$, are continuous functions.
- **(K)** Parameter θ is chosen in a non-empty compact subset Θ of $\mathbb{R}^p$.
- **(IA)** Preference relation $\succsim$ over histories is independent of actions, i.e., for any pair of histories $\mathbf{h}_1$ and $\mathbf{h}_2$, if $\mathbf{h}_1 \succsim \mathbf{h}_2$, then $\mathbf{h}'_1 \succsim \mathbf{h}'_2$ when $\mathbf{h}_1$ and $\mathbf{h}'_1$ (resp. $\mathbf{h}_2$ and $\mathbf{h}'_2$) are two histories containing the same sequence of states.

The continuity conditions seem to be quite natural when considering MDPs in continuous domains. Likewise, assumption (**K**) is not a very strong condition. The last condition implies that preferences over histories only depend on visited states. By (**IA**), preferences over histories induce preferences over *trajectories*, i.e., sequences of states. By abuse of notation, we also denote the preference relation over trajectories by $\succsim$.

In the first case, we allow randomization in the application of a policy. In our context, a *randomized policy* is characterized by a probability distribution over parameter space Θ.

Applying a *randomized* policy means selecting a parameter θ according to the probability distribution characterizing the randomized policy first, and applying the policy π_θ on the whole horizon then. In the next proposition, we prove the existence of a Condorcet winner among randomized policies.

Proposition 4 *Under* (**D**), (**C1**), (**C2**), (**K**) *and* (**IA**), *there exists a randomized policy* π^* *which is a Condorcet winner, that is, for any (randomized or not) policy* π, *it holds that* $S(\mathbf{P}_{\pi^*}, \mathbf{P}_\pi) \geq S(\mathbf{P}_\pi, \mathbf{P}_{\pi^*})$.

Proof This result was proved in Kreweras (1961) for a finite set of action, and the corresponding proof can be easily extended to an infinite set of action. A Condorcet winner can be seen as a Nash equilibrium in the following two-player symmetric continuous zero-sum game: The set of strategies is defined as the set of (non-randomized) parametric policies Π, which can be identified to Θ. The payoff for strategy $\pi_{\theta'}$ against π_θ is defined by $u(\theta, \theta') = S(\mathbf{P}_{\pi_\theta}, \mathbf{P}_{\pi_{\theta'}}) - S(\mathbf{P}_{\pi_{\theta'}}, \mathbf{P}_{\pi_\theta})$. For any $\theta \in \Theta$, $\mathbf{P}_\theta(\tau)$ denotes the probability that parametric policy π_θ generates trajectory $\tau = (s^{(0)}, s^{(1)}, \ldots, s^{(T)})$. This probability can be written as:

$$\mathbf{P}_\theta(\tau) = \prod_{i=1}^{T} \mathbf{P}(s^{(i)} \mid s^{(i-1)}), \pi_\theta(s^{(i-1)})$$

Then, payoff $u(\theta, \theta')$ can be written as:

$$u(\theta, \theta') = \sum_\tau \sum_{\tau' \precsim \tau} \mathbf{P}_\theta(\tau)\mathbf{P}_{\theta'}(\tau') - \sum_\tau \sum_{\tau' \precsim \tau} \mathbf{P}_{\theta'}(\tau)\mathbf{P}_\theta(\tau')$$

thanks to (**D**) and our initial assumption that the horizon is finite.

The continuity conditions of (**C1**) and (**C2**) entail that $\mathbf{P}_\theta$ is a continuous function and therefore so is $u(\theta, \theta')$. Then, by Glicksberg's generalization (Fudenberg and Tirole 1991) of the Kakutani fixed point theorem, there exists a mixed Nash equilibrium, i.e., in our context, a randomized policy that is a Condorcet winner for $\gtrsim$. □

In the second case, we introduce two other conditions in order to guarantee the existence of a Condorcet winner among the (non-randomized) policies. Before presenting them, we recall two definitions. A function $f : E \to \mathbb{R}$ is said to be *quasiconcave* if $E \subset \mathbb{R}^p$ is convex and

$$\forall \lambda \in [0, 1],\ \forall x, y \in \mathbb{R}^p :\ f(\lambda x + (1 - \lambda)y) \geq \min\big(f(x), f(y)\big).$$

A family of functions $f_{\upsilon \in \Upsilon}$ is said to be *uniformly quasi-concave* if $f_\upsilon : E \to \mathbb{R}$ is quasiconcave for all $\upsilon \in \Upsilon$ and, moreover, for all $x, y \in E$ either of the following conditions holds:

$$\forall \upsilon \in \Upsilon :\ \min(f_\upsilon(x), f_\upsilon(y)) = f_\upsilon(x)$$
$$\forall \upsilon \in \Upsilon :\ \min(f_\upsilon(x), f_\upsilon(y)) = f_\upsilon(y)$$

For any $(\mathbf{s}, \mathbf{s}') \in \mathcal{S} \times \mathcal{S}$, let $f_{\mathbf{s},\mathbf{s}'}(\theta)$ denote $\mathbf{P}(\mathbf{s}' \mid \mathbf{s}, \pi_\theta(\mathbf{s}))$ the composition of transition probability $\mathbf{P}(S \mid \mathbf{s}, \cdot)$ with parametric policy π_θ.

(**C**) Parameter space Θ is convex.
(**UQC**) The family of functions $f_{(\mathbf{s},\mathbf{s}') \in \mathcal{S} \times \mathcal{S}}(\theta)$ is uniformly quasiconcave.

While the convexity condition does not seem to be very restrictive, the condition (**UQC**) is quite strong. It excludes the existence of states $\mathbf{s}_1, \mathbf{s}'_1, \mathbf{s}_2, \mathbf{s}'_2$ and parameters θ, θ' such that

$$\mathbf{P}(\mathbf{s}'_1 \mid \mathbf{s}_1, \pi_\theta(\mathbf{s}_1)) > \mathbf{P}(\mathbf{s}'_1 \mid \mathbf{s}_1, \pi_{\theta'}(\mathbf{s}_1)) \text{ and}$$
$$\mathbf{P}(\mathbf{s}'_2 \mid \mathbf{s}_2, \pi_\theta(\mathbf{s}_2)) < \mathbf{P}(\mathbf{s}'_2 \mid \mathbf{s}_2, \pi_{\theta'}(\mathbf{s}_2)).$$

These two conditions along with the previous conditions (**D**), (**C1**), (**C2**), (**K**) and (**IA**) are sufficient for the existence of a (non-randomised) policy that is a Condorcet winner:

Proposition 5 *Under* (**D**), (**C1**), (**C2**), (**K**), (**IA**), (**C**) *and* (**UQC**), *there exists a parameter* $\theta^* \in \Theta$ *such that* π_{θ^*} *is a Condorcet winner among* $\Pi = \{\pi_\theta | \theta \in \Theta\}$.

Proof In game theory (Fudenberg and Tirole 1991), it is known that if payoff function u (defined in proof of Proposition 4) is quasiconcave in its first argument (as the game is symmetric, it is also in its second argument), then there exists a pure Nash equilibrium.

Sums of uniformly quasiconcave functions are also quasiconcave (Prékopa et al. 2011). Moreover, products of nonnegative uniformly quasiconcave functions are also quasiconcave. If $f_{(\mathbf{s},\mathbf{s}')\in\mathcal{S}\times\mathcal{S}}(\theta)$ is uniformly quasiconcave, then $\mathbf{P}_\theta$ (seen as a function of θ) is quasiconcave as well, and so is u. □

4 Preference-based racing algorithm

This section is devoted to our preference-based racing algorithm (PBR). Section 4.1 describes the concentration property of the estimate of $\mathbf{P}(X \succ Y)$, which is a cornerstone of our approach. Section 4.2 provides a simple technique to handle incomparability of random samples. Section 4.3 outlines the PBR algorithm as a whole, and Sect. 4.4 provides a formal analysis of this algorithm.

4.1 An efficient estimator of $\mathbf{P}(X \succ Y)$

In Sect. 3.1, we introduced an ordinal decision model specified by the order relation $\gg_C$. Sorting a set of random variables $X_1, \ldots, X_K$ according to $\gg_C$ first of all requires an *efficient estimator* of $S(X_i, X_j) = \mathbf{P}(X_i \succ X_j)$.

A two-sample U-statistic called the *Mann-Whitney U-statistic* (also known as the *Wilcoxon 2-sample statistic*) is an unbiased estimate of $S(\cdot,\cdot)$ (Serfling 1980). Given independent samples $\mathbf{X} = \{x^{(1)}, \ldots, x^{(n)}\}$ and $\mathbf{Y} = \{y^{(1)}, \ldots, y^{(n)}\}$ of two independent random variables X and Y (for simplicity, we assume equal sample sizes), it is defined as

$$\widehat{S}(\mathbf{X}, \mathbf{Y}) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbb{I}\{x^{(i)} \succ y^{(j)}\} \ . \tag{2}$$

Apart from being an unbiased estimator of $S(X, Y)$, (2) possesses concentration properties resembling those of the sum of independent random variables.[1]

Theorem 6 (Hoeffding (1963), §5b) *For any* $\epsilon > 0$, *using the notations introduced above,*

$$\mathbf{P}\left(\left|\widehat{S}(\mathbf{X}, \mathbf{Y}) - S(X, Y)\right| \geq \epsilon\right) \leq 2\exp(-2n\epsilon^2) \ .$$

[1] Although $\widehat{S}$ is a sum of n^2 random values, these values are combinations of only $2n$ independent values. This is why the convergence rate is not better than the usual one for a sum of n independent variables.

An equivalent formulation of this theorem is as follows: For any $0 < \delta < 1$, the interval

$$\Bigg[\underbrace{\widehat{S}(\mathbf{X},\mathbf{Y}) - \sqrt{\frac{1}{2n}\ln\frac{2}{\delta}}}_{L(\mathbf{X},\mathbf{Y})}, \underbrace{\widehat{S}(\mathbf{X},\mathbf{Y}) + \sqrt{\frac{1}{2n}\ln\frac{2}{\delta}}}_{U(\mathbf{X},\mathbf{Y})} \Bigg] \tag{3}$$

contains $S(X, Y)$ with probability at least $1 - \delta$. For more details on the U-statistic, see Appendix 9.1.

4.2 Handling incomparability

Recall that $\succ$ is only assumed to be a partial order and, therefore, allows for incomparability $x \perp y$ between realizations x and y of random variables (histories generated by policies). In such cases we have $\mathbb{I}\{x \succ y\} = \mathbb{I}\{y \succ x\} = 0$ and, consequently, $\widehat{S}(\mathbf{X},\mathbf{Y}) + \widehat{S}(\mathbf{Y},\mathbf{X}) < 1$. Since this inequality is inconvenient and may complicate the implementation of the algorithm, we use a modified version of the indicator function as proposed by Hemelrijk (1952):

$$\mathbb{I}^{\mathrm{INC}}\{x \succ x'\} = \mathbb{I}\{x \succ x'\} + \frac{1}{2}\mathbb{I}\{x \perp x'\} \tag{4}$$

A more serious problem caused by incomparability is a complication of the variance estimation for $\widehat{S}(\mathbf{X},\mathbf{Y})$ (Hemelrijk 1952). Therefore, it is not clear how Bernstein-like bounds (Audibert et al. 2007), where the empirical variance estimate is used in the concentration inequality, could be applied.

4.3 Preference-based racing algorithm

Our preference-based racing setup assumes K random variables $X_1, \ldots, X_K$ with distributions $\mathbf{P}_{X_1}, \ldots, \mathbf{P}_{X_K}$, respectively, and these random variables take values in a partially ordered set $(\Xi, \succ)$. Obviously, the value-based racing setup described in Sect. 2.3 is a special case, with $\Xi = \mathbb{R}$ and $\succ$ reduced to the standard $>$ relation on the reals (comparing rollouts in terms of their rewards). The goal of our preference-based racing (PBR) algorithm is to find the best κ random variables with respect to the surrogate decision model $\gg_C$ introduced in Sect. 3.1. This leads to the following optimization task:

$$I^* \in \operatorname*{argmax}_{I \subseteq [K]: |I| = \kappa} \sum_{i \in I} \sum_{j \neq i} \mathbb{I}\{X_i \gg_C X_j\} \tag{5}$$

which can be rewritten by using $\gg$ as

$$I^* \in \operatorname*{argmax}_{I \subseteq [K]: |I| = \kappa} \sum_{i \in I} \sum_{j \neq i} \mathbb{I}\{X_i \gg X_j\} \tag{6}$$

Thanks to the indicator function (4), we have $S(X_i, X_j) = 1 - S(X_j, X_i)$ and hence $\mathbb{I}\{X_i \gg X_j\} = \mathbb{I}\{S(X_i, X_j) > 1/2\} = \mathbb{I}\{S(X_j, X_i) < 1/2\}$, which simplifies our implementation.

Algorithm 2 shows the pseudocode of PBR. It assumes as inputs the number κ, an upper bound $n_{\max}$ on the number of realizations an option is allowed to sample, and an upper bound δ on the probability of making a mistake (i.e., returning a suboptimal selection). We will concisely write $s_{i,j} = S(X_i, X_j)$ and $\widehat{s}_{i,j}$ for its estimate. The confidence interval (3) of $\widehat{s}_{i,j}$ for confidence level $1 - \delta$ is denoted by $[\ell_{i,j}, u_{i,j}]$. The set A consists of those index pairs for which the preference can not yet be determined with high probability

(i.e., $1/2 \in [\ell_{i,j}, u_{i,j}]$), but that are possibly relevant for the final outcome. Initially, A contains all $K^2 - K$ pairs of indices (line 1).

PBR first samples each pair of options whose indices appear (at least once) in A (lines 4–5). Then, in lines 6–10, it calculates $\widehat{s}_{i,j}$ for each pair of options according to (2), and the confidence intervals $[\ell_{i,j}, u_{i,j}]$ based on (3).

Algorithm 2 PBR$(X_1, \ldots, X_K, \kappa, n_{\max}, \delta)$

1: $A = \{(i, j) | \, i \neq j, 1 \leq i, j \leq K\}$
2: $n = 0$
3: **while** $(n \leq n_{\max}) \wedge (|A| > 0)$ **do**
4: **for all** i appearing in A **do**
5: $x_i^{(n)} \sim X_i$ ▷ Draw a random sample
6: **for all** $(i, j) \in A$ **do**
7: Update $\widehat{s}_{i,j}$ with the new samples according to (2)
8: using the indicator function $\mathbb{I}^{\text{INC}}\{., .\}$ from (4)
9: $c_{i,j} = \sqrt{\frac{1}{2n} \log \frac{2K^2 n_{\max}}{\delta}}$
10: $u_{i,j} = \widehat{s}_{i,j} + c_{i,j}$, $\ell_{i,j} = \widehat{s}_{i,j} - c_{i,j}$
11: **for** $i = 1 \rightarrow K$ **do**
12: $z_i = |\{j \mid \ell_{i,j} > 1/2, j \neq i\}|$ ▷ Number of options that are beaten by i
13: $o_i = |\{j \mid u_{i,j} < 1/2, j \neq i\}|$ ▷ Number of options that beat i
14: $C = \left\{i \mid K - \kappa < \left|\{j \mid K - z_j < o_i\}\right|\right\}$ ▷ select
15: $D = \left\{i \mid \kappa < \left|\{j \mid K - o_j < z_i\}\right|\right\}$ ▷ discard
16: **for** $(i, j) \in A$ **do**
17: **if** $(i, j \in C \cup D) \vee (1/2 \notin [\ell_{i,j}, u_{i,j}])$ **then**
18: $A = A \setminus (i, j)$
19: ▷ Do not update $\hat{s}_{i,j}$ any more
20: $n = n + 1$
21: σ is a permutation that sorts the options in decreasing order based on $\widehat{d}_i = \#\{j \mid \ell_{i,j} > 1/2\}$.
22: **return** σ

Next, for each X_i, we compute the number z_i of random variables that are worse with high enough probability—that is, for which $\ell_{i,j} > 1/2$, $j \neq i$ (line 12). Similarly, for each option X_i, we also compute the number o_i of options X_j that are preferred to it with high enough probability—that is, for which $u_{i,j} < 1/2$ (line 13). Note that, for each X_j, there are always at most $K - z_j$ options that can be better. Therefore, if $\#\{j \mid K - z_j < o_i\} > K - \kappa$, then X_i is a member of the solution set I^* of (6) with high probability (see line 14). The indices of these options are collected in C. One can also discard options based on a similar argument (line 15); their indices are collected in D. Note that a selection or exclusion of an option requires at most K different confidence bounds to be bigger or smaller than $1/2$, and since we can select or discard an option at any time, the confidence level δ has to be divided by $K^2 n_{\max}$ (line 9). In Sect. 5, we will describe a less conservative confidence correction that adjusts the confidence level dynamically based on the number of selected and discarded options.

In order to update A, we note that, for those options in $C \cup D$, it is already decided with high probability whether or not they belong to I^*. Therefore, if two options X_i and X_j both belong to $C \cup D$, then $s_{i,j}$ does not need to be sampled any more, and thus the index pair (i, j) can be excluded from A. Additionally, if $1/2 \notin [\ell_{i,j}, u_{i,j}]$, then the pairwise relation of X_i and X_j is known with high enough probability, so (i, j) can again be excluded from A. These filter steps are implemented in line 17.

We remark that the condition for termination in line 3 is as general as possible and cannot be relaxed. Indeed, termination must be based on those preferences that are already decided (with high probability). Thus, assuming the options to be ordered according to $\gg_C$, the algorithm can only stop if $\min\{z_1, \ldots, z_\kappa\} \geq \max\{K - o_{\kappa+1}, \ldots, K - o_K\}$ or $\min\{o_{\kappa+1}, \ldots, o_K\} \leq \max\{K - z_1, \ldots, K - z_\kappa\}$. Both conditions imply that $C \cup D = [K]$ and hence that A is empty.

4.4 Analysis of the PBR algorithm

Recall that PBR returns a permutation σ, from which the set of options B deemed best by the racing algorithm (in terms of $\gg_C$) can be obtained as $B = \{X_{\sigma(i)} | 1 \leq i \leq \kappa\}$. In the following, we consider the top-κ set B as the output of PBR.

In the first part of our analysis, we upper bound the expected number of samples taken by PBR. Our analysis is similar to the sample complexity analysis of PAC-bandit algorithms (Even-Dar et al. 2002). Technically, we have to make the assumption that $S(X_i, X_j) \neq 1/2$ for all $i, j \in [K]$, which may appear quite restrictive at first sight. In practice, however, the value of $S(X_i, X_j)$ will indeed almost never *exactly* equal $1/2$.[2]

Theorem 7 *Let $X_1, \ldots, X_K$ be random variables such that $S(X_i, X_j) \neq 1/2$ for all $i, j \in [K]$, and define*

$$n_i = \left\lceil \frac{1}{4 \min_{j \neq i} \Delta_{i,j}^2} \log \frac{2K^2 n_{\max}}{\delta} \right\rceil ,$$

where $\Delta_{i,j} = S(X_i, X_j) - 1/2$. Then, whenever $n_i \leq n_{\max}$ for all $i \in [K]$, PBR outputs the κ best options (with respect to $\gg_C$) with probability at least $1 - \delta$ and generates at most $\sum_{i=1}^K n_i$ samples.

Proof According to (3), for any i, j and round n, the probability that $s_{i,j}$ is not included in

$$\left[\underbrace{\widehat{s}_{i,j} - \sqrt{\frac{1}{2n} \ln \frac{2K^2 n_{\max}}{\delta}}}_{\ell_{i,j}}, \underbrace{\widehat{s}_{i,j} + \sqrt{\frac{1}{2n} \ln \frac{2K^2 n_{\max}}{\delta}}}_{u_{i,j}} \right] \tag{7}$$

is at most $\delta/(K^2 n_{\max})$. Thus, with probability at least $1 - \delta$, $s_{i,j} \in [\ell_{i,j}, u_{i,j}]$ for every i and j throughout the whole run of the algorithm. Therefore, if the PBR returns a ranking represented by permutation σ and $n_{i,j} \leq n_{\max}$ for all $i, j \in [K]$, then $\{1 \leq i \leq K | \sigma(i) \leq \kappa\}$ is the solution set of (6) with probability at least $1 - \delta$. Thus, the PBR algorithm is correct.

In order to upper bound the expected sample complexity, let us note that based on the confidence interval in (7), one can compute a sample size $\widetilde{n}_{i,j}$ for some i and j so that both X_i and X_j are sampled for at least $\widetilde{n}_{i,j}$ times, then $[\ell_{i,j}, u_{i,j}]$ does not contain $1/2$ with probability at most $\delta/(K^2 n_{\max})$. A simple calculation yields

$$\widetilde{n}_{i,j} = \left\lceil \frac{1}{4\Delta_{i,j}^2} \log \frac{2K^2 n_{\max}}{\delta} \right\rceil .$$

Furthermore, if all preferences against other options are decided for some i (i.e., $\ell_{i,j} > 1/2$ or $u_{i,j} < 1/2$ for all $j \neq i$), then X_i will not be sampled any more. Therefore, by using the union bound, X_i is sampled at most $\max_{j \neq i} \widetilde{n}_{i,j} < n_i$ with probability at most δ/K.

[2] For example, if $S(X_i, X_j)$ is considered as a random variable with continuous density on $[0, 1]$, then the probability of $S(X_i, X_j) \neq 1/2$ is 0.

The theorem follows by putting these observations together. □

Remark 8 We remark that Theorem 7 remains valid despite the fact that statistical independence is not assured, neither for the terms in $\widehat{s}_{i,j}$ nor for $\widehat{s}_{i,j}$ and $\widehat{s}_{i,j'}$ with $i, j, j' \in [K]$. First, the confidence interval of each $\widehat{s}_{i,j}$ is obtained based on the concentration property of the U-statistic (Theorem 6). Second, the confidence intervals of $\widehat{s}_{i,j}$ are calculated separately for all $i, j \in [K]$ in every iteration, and the subsequent application of the union bound does not require independence.

In the second part of our analysis, we investigate the relation between the outcome of PBR and the decision model $\gg$. Theorem 7 and Proposition 3 have the following immediate consequence for PBR.

Corollary 9 *Let $\mathcal{X} = \{X_1, \ldots, X_K\}$ be a set of random variables with Smith set $\mathcal{C}^* \subseteq \mathcal{X}$. Then, under the conditions of Theorem 7, with probability at least $1 - \delta$,* PBR *outputs a set of options $B \subseteq \mathcal{X}$ satisfying the following: If $|\mathcal{C}^*| \leq \kappa$, then $\mathcal{C}^* \subseteq B$ (Smith efficiency), otherwise $B \subseteq \mathcal{C}^*$.*

Proof The result follows immediately from Theorem 7 and Proposition 3. □

Thus, PBR finds the Smith set with high probability provided κ is set large enough; otherwise, it returns at least a subset of the Smith set. This indeed justifies the use of $\gg_C$ as a decision model. Nevertheless, as pointed out in Sect. 8 below, other surrogates of the $\gg$ relation are conceivable, too.

5 Implementation and practical issues

In this section, we describe three "tricks" to make the implementation of the ES along with the preference-based racing framework more efficient. These tricks are taken from Hendrich-Meisner and Igel (2009) and adapted from the setting of value-based to the one of preference-based racing.

1. Consider the confidence interval $I = [\ell_{i,j}, u_{i,j}]$ for a pair of objects i and j. Since an update $I' = [\ell'_{i,j}, u'_{i,j}] = [\widehat{s}_{i,j} + c_{i,j}, \widehat{s}_{i,j} - c_{i,j}]$ based on the current estimate $\widehat{s}_{i,j}$ will not only shrink but also shift this interval, one of the two bounds might be worse than it was before. To take full advantage of previous estimates, one may update the confidence interval with the intersection $I'' = I' \cap I'' = [\max(\ell_{i,j}, \ell'_{i,j}), \min(u_{i,j}, u'_{i,j})]$.
   In order to justify this update, first note that the confidence parameter δ in the PBR algorithm was set in such a way that, for each time step, the confidence interval $[\ell_{i,j}, u_{i,j}]$ for any pair of options includes $s_{i,j}$ with probability at least $1-\delta/(n_{\max}K^2)$ (see (7)). Now, consider the intersection of confidence intervals for options i and j that are calculated up to iteration $n_{\max}$. This interval contains $s_{i,j}$ with probability at least $1-\delta/K^2$, an estimate that follows immediately from the union bound. Thus, it defines a valid confidence interval with confidence level $1 - \delta/K^2$.
2. The correction of confidence level δ by $K^2 n_{\max}$ is very conservative. When an option i is selected or discarded in iteration t, that is, its index is contained in $C \cup D$ in line 17 of the PBR algorithm, none of the $\widehat{s}_{i,1}, \ldots \widehat{s}_{i,K}$ will be recomputed and used again. Based on this observation, one can adjust the correction of the confidence level dynamically. Let m_t be the number of options that are discarded or selected ($|C \cup D|$) up to iteration t. Then, in iteration t, the correction

$$c_t = (K-1)\sum_{\ell=0}^{t-1} m_t + (K-1)(n_{\max} - t + 1)m_t \tag{8}$$

can be used instead of $K^2 n_{\max}$. Note that the second term in (8) upper bounds the number of active pairs contained by A in the implementation of PBR in every time step. Therefore, this is a valid correction.

3. The parameter $n_{\max}$ specifies an upper bound on the sample size that can be taken from an option. If this parameter is set too small, then the ranking returned by the racing algorithm is not reliable enough. Heidrich-Meisner and Igel (2009) therefore suggested to dynamically adjust $n_{\max}$ so as to find the smallest appropriate value for this parameter. In fact, if a non-empty set of options remains to be sampled at the end of a race (A is not empty), the time horizon $n_{\max}$ was obviously not large enough (case I). On the other hand, if the racing algorithm ends (i.e., each option is either selected or discarded) even before reaching $n_{\max}$, this indicates that the parameter could be decreased (case II). Accordingly, a simple policy can be used for parameter tuning, i.e., for adapting $n_{\max}$ for the next iteration of PB-EDPS (the race between the individuals of the next population) based on experience from the current iteration: $n_{\max}$ is set to $n_{\max} = \alpha n_{\max}$ in case I and to $n_{\max} = \alpha^{-1} n_{\max}$ in case II, where $\alpha > 1$ is a user-defined parameter. In our implementation, we use $\alpha = 1.25$. Moreover, we initialize $n_{\max}$ by 3 and never exceed $n_{\max} = 100$, even if our adjustment policy suggests a further increase.

The above improvements essentially aim at decreasing the sample complexity of the racing algorithm. In our experiments, we shall therefore investigate their effect on empirical sample complexity.

6 Experiments

In Sect. 6.1, we compare our PBR algorithm with the original Hoeffding race (HR) algorithm in terms of empirical sample complexity on synthetic data. In Sect. 6.2, we test our PB-EDPS method on a benchmark problem that was introduced in previous work on preference-based RL (Cheng et al. 2011).

6.1 Results on synthetic data

Recall that our preference-based racing algorithm is more general than the original value-based one and, therefore, that PBR is more widely applicable than the Hoeffding race (HR) algorithm. This is an obvious advantage of PBR, and indeed, our preference-based generalization of the racing problem is mainly motivated by applications in which the value-based setup cannot be used. Seen from this perspective, PBR has an obvious justification, and there is in principle no need for a comparison to HR. Nevertheless, such a comparison is certainly interesting in the standard numerical setting where both algorithms can be used.

More specifically, the goal of our experiments was to compare the two algorithms in terms of their empirical sample complexity. This comparison, however, has to be done with caution, keeping in mind that PBR and HR are solving different optimization tasks (namely (1) and (6), respectively): HR selects the κ best options based on the means, whereas the goal of PBR is to select κ options based on $\gg_C$. While these two objectives coincide in some cases, they may differ in others. Therefore, we considered the following two test scenarios:

1. *Normal distributions:* each random variable X_i follows a normal distribution $\mathcal{N}((k/2)m_i, v_i)$, where $m_i \sim U[0, 1]$ and $v_i \sim U[0, 1]$, $k \in \mathbb{N}^+$;
2. *Bernoulli distributions with random drift:* each X_i obeys a Bernoulli distribution $Bern(1/2) + d_i$, where $d_i \sim (k/10)\, U[0, 1]$ and $k \in \mathbb{N}^+$.

In both scenarios, the goal is to rank the distributions by their means.[3] For both racing algorithms, the following parameters were used in each run: $K = 10$, $\kappa = 5$, $n_{\max} = 300$, $\delta = 0.05$.

Strictly speaking, HR is not applicable in the first scenario, since the support of a normal distribution is not bounded; we used $R = 8$ as an upper bound, thus conceding to HR a small probability for a mistake.[4] For Bernoulli, the bounds of the supports can be readily determined.

Note that the complexity of the racing problem is controlled by the parameter k, with a higher k indicating a less complex task; we varied k between 1 and 10. Since the complexity of the task is not known in practice, an appropriate time horizon $n_{\max}$ might be difficult to determine. If one sets $n_{\max}$ too low (with respect to the complexity of task), the racing algorithm might not be able to assure the desired level of accuracy. We designed our synthetic experiments so as to challenge the racing algorithms with this problem; amongst others, this allows us to assess the deterioration of their accuracy if the task complexity is underestimated. For doing this, we kept $n_{\max} = 300$ fixed and varied the complexity of task by tuning k. Note that if the $\Delta_{i,j} = s_{i,j} - 1/2$ values that we used to characterize the complexity of the racing task in our theoretical analysis were known, a lower bound for $n_{\max}$ could be calculated based on the Theorem 7.

Our main goal in this experiment was to compare the HR and PBR methods in terms of *accuracy*, which is the percentage of true top-κ variables among the predicted top-κ, and in terms of *empirical sample complexity*, which is the number of samples drawn by the racing algorithm for a fixed racing task. For a fixed k, we generated a problem instance as described above and ran both racing algorithms on this instance. We repeated this process 1,000 times and averaged the empirical sample complexity and accuracy of both racing algorithms. In this way, we could assess the performance of the racing algorithms for a fixed k.

Figure 2 plots the empirical sample complexity versus accuracy for various k. First we ran plain HR and PBR algorithms without the improvements described in Sect. 5. As we can see from the plots (Fig. 2a, c), PBR achieves a significantly lower sample complexity than HR, whereas its accuracy is on a par or better in most cases. While this may appear surprising at first sight, it can be explained by the fact that the Wilcoxon 2-sample statistic is *efficient* (Serfling 1980), just like the mean estimate in the case of the normal and Poisson distribution, but its asymptotic behavior can be better in terms of constants. That is, while the variance of the mean estimate scales with $1/n$ (according to the central limit theorem), the variance of the Wilcoxon 2-sample statistic scales with $1/Rn$ for equal sample sizes, where $R \geq 1$ depends on the value estimated by the statistic (Serfling 1980).

Second, we ran HR and PBR in their improved implementation as described in Sect. 5. That is, δ was corrected dynamically during the run, and the confidence intervals are updated by intersecting them with the previously computed intervals. The results are plotted in Fig. 2b

[3] In order to show that the ranking based on means and $\gg$ coincide for a set of options $X_1, \ldots, X_K$ with means $\mu_1, \ldots, \mu_K$, it is enough to see that for any X_i and X_j, $\mu_i < \mu_j$ implies $S(X_i, X_j) > 1/2$. In the case of the normal distribution, this follows from the symmetry of the density function. Now, let us consider two Bernoulli distributions with parameters p_1 and p_2, where $p_1 < p_2$. Then, a simple calculation shows that the value of $S(.,.)$ is $(p_2 - p_1 + 1)/2$, which is greater than $1/2$. This also holds if we add a drift $d_1, d_2 \in [0, 1]$ to the value of the random variables.

[4] The probability that all samples remain inside the range is larger than 0.99 for $K = 10$ and $n_{\max} = 300$.

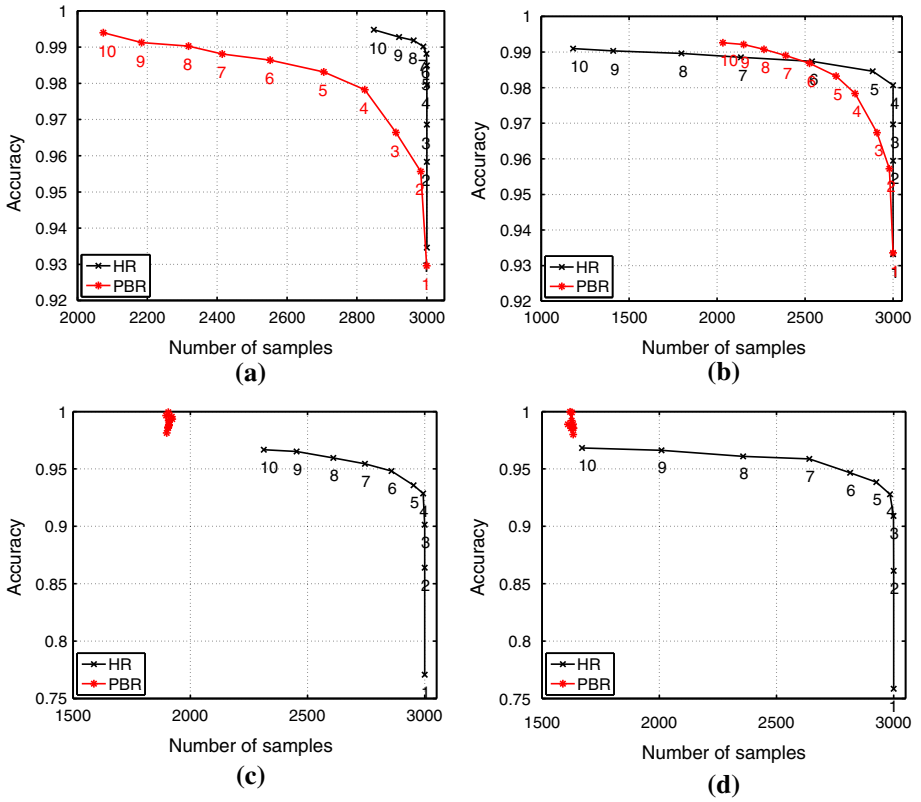


Fig. 2 The accuracy is plotted against the empirical sample complexities for the Hoeffding race algorithm (HR) and PBR, with the complexity parameter k shown below the markers. Each result is the average of 1,000 repetitions. **a** Normal distributions. **b** Normal distributions/improved. **c** Bernoulli distributions. **d** Bernoulli distributions/improved

and d. Thanks to these implementation tricks, the empirical sample complexities of both racing algorithms are reduced. Interestingly, the relative improvements of HR are larger than those of PBR. This can be explained by the fact that HR completely stops sampling an option once it has been selected or discarded. Since PBR may still continue sampling such options, the effect of correcting δ on the empirical sample complexity could be less pronounced.

In the Bernoulli case, one may wonder why the sample complexity of PBR hardly changes with k (see the red point cloud in Fig. 2c). This can be explained by the fact that the two sample U-statistic $\widehat{S}$ in (2) does not depend on the magnitude of the drift d_i (as long as it is smaller than 1).

6.2 Medical treatment design

Here, we tackle a problem that has been used in previous work on preference-based RL (Cheng et al. 2011; Akrour et al. 2012), namely the medical treatment design for cancer clinical trials. The problem is to learn an optimal treatment policy π mapping states $\boldsymbol{s} = (S, X) \in \mathcal{S} = \mathbb{R}^2_+$ to actions in the form of a dosage level $d \in [0, 1]$; the drug is given once a month, and a patient is simulated over a fixed time horizon. We conducted our experiments with six

months. A state $s = (S, X)$ describes the health condition of the patient: S is the tumor size and X the level of toxicity, which is inversely related to the wellness of the patient. These two properties constitute conflicting criteria: An increase of the dosage level will reduce the tumor size but increase toxicity and therefore affect the patient's wellness. A corresponding simulation model based on first-order difference equations was originally introduced in Zhao et al. (2009). In this model, the state transitions are defined as follows:

$$S_{t+1} = S_t + \Big[a_1 \cdot \max(X_t, X_0) - b_1 \cdot (D_t - d_1) \Big] \times \mathbb{I}\{S_t > 0\} ,$$
$$X_{t+1} = X_t + a_2 \cdot \max(S_t, S_0) + b_2 \cdot (D_t - d_2) ,$$

where S_t and X_t denote the tumor size and toxicity level after the t-th month of treatment, respectively. The dosage level chosen in the t-th month is denoted by D_t. The model parameters $a_1, a_2, b_1, b_2, d_1, d_2$ are positive real-valued numbers. The probability that a patient dies in the t-th month of the treatment follows a Bernoulli distribution with parameter $1 - \exp(-\exp(c_0 + c_1 \cdot S_t + c_2 \cdot X_t))$, where c_0, c_1, c_2 are also real-valued parameters. The parameters of the medical treatment model were set to $a_1 = 0.1, a_2 = 0.15, b_1 = b_2 = 1.2, d_1 = d_2 = 0.5, c_0 = -4, c_1 = c_2 = 1$. The reward function is defined as

$$5 \cdot \Big[\mathbb{I}\{X_{t+1} - X_t < 1/2\} - \mathbb{I}\{X_{t+1} - X_t > 1/2\} \Big]$$
$$+15 \cdot \mathbb{I}\{S_t \leq 0\} + 5 \cdot \Big[\mathbb{I}\{S_{t+1} - S_t < 1/2\} - \mathbb{I}\{S_{t+1} - S_t > 1/2\} \Big]$$

unless the patient dies, in which case the reward is -60. The first term expresses the change of wellness of the patient in terms of toxicity, the second term corresponds to the healing of the patient (tumor size is 0), and the third term expresses the change of the tumor size.

As argued by the authors of Cheng et al. (2011), the numerical rewards assigned to different health states of a patient (including the extreme case of death) are quite arbitrary in this model. Therefore, they propose an alternative and more realistic formalization, in which histories are compared in a qualitative way:

- Giving full priority to the survival of a patient, $\mathbf{h}' \preceq \mathbf{h}$ if the patient survives in $\mathbf{h}$ but not in $\mathbf{h}'$, and both histories are incomparable ($\mathbf{h}' \perp \mathbf{h}$) if the patient does neither survive in $\mathbf{h}'$ nor in $\mathbf{h}$.
- Otherwise, if the patient survives in both histories, preference depends on the worst wellness of the patient and the final tumor size: Let C_X and C'_X denote, respectively, the *maximal* toxicity during the whole treatment in $\mathbf{h}$ and $\mathbf{h}'$, and C_S and C'_S the respective size of the tumor *at the end of the therapy*. Then, preference is defined via Pareto dominance: $\mathbf{h}' \preceq \mathbf{h}$ if (and only if) $C_X \leq C'_X$ and $C_S \leq C'_S$.

Let us again emphasize that $\preceq$ thus defined, as well as the induced strict order $\prec$, are only *partial* order relations. We used the same experimental setup as in previous work (Cheng et al. 2011; Akrour et al. 2012).

We run the implementation of Heidrich-Meisner and Igel (2009) with the Hoeffding race algorithm and CMA- ES (Hansen and Kern 2004); we refer to this implementation as EDPS. We set $\lambda = 7$ and $\mu = 3$ according to Hansen and Kern (2004). The initial global step size σ_0 in CMA- ES was selected from $\{0.1, 1, 5, 10, 15, 25, 50, 100\}$. The racing algorithm has two hyper-parameters, the confidence term δ and the maximum number of samples allowed for a single option, $n_{\max}$. We optimized δ in the range $\{0.01, 0.05, 0.5, 0.1, 0.2\}$, while $n_{\max}$ was initialized with 3 and then adapted as described in Sect. 5. All parameter values were determined by means of a grid search, repeating the training process in each grid

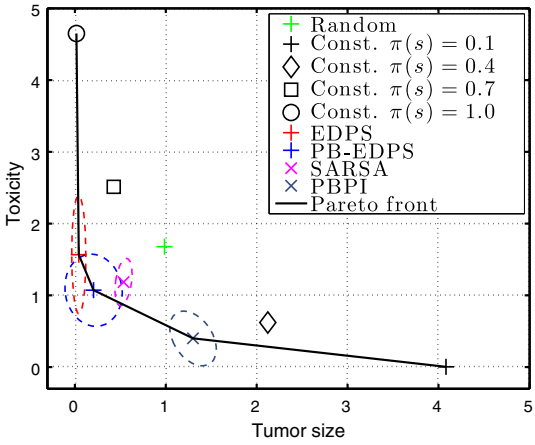


Fig. 3 Illustration of patient status under different treatment policies. On the x axis is the tumor size after 6 months, on the y axis the highest toxicity during the treatment

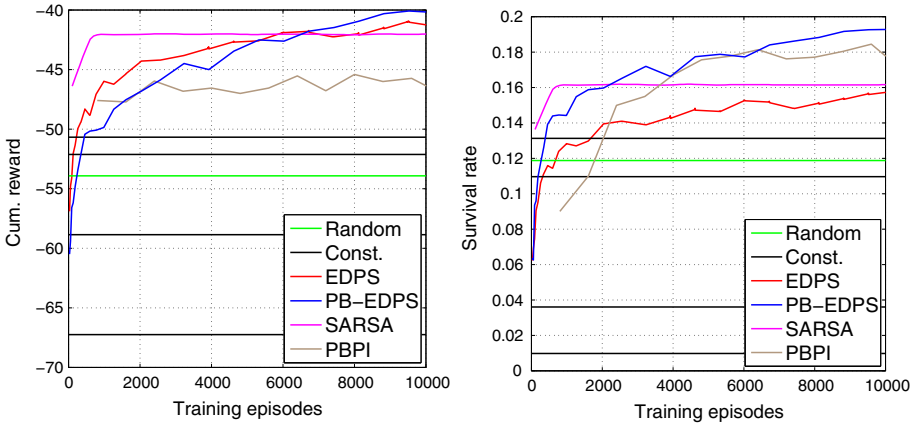
point (parameter setting) 100 times, and evaluating each model on 300 patients in terms of expected utility; we found $\sigma_0 = 10$, $\delta = 0.2$ to be optimal.

Our preference-based variant PB-EDPS as introduced in Sect. 3 was run with the same parameters. We used a sigmoidal policy space defined as $\pi_\theta(\mathbf{s}) = 1/(1 + \exp(-\theta^T \mathbf{s}))$. As baseline methods, we run the discrete uniform random policy (randomly choosing a dosage $d \in D' = \{0.1, 0.4, 0.7, 1.0\}$ each month) and the constant policies that take the same dosage $d \in D'$ independently of the patient's health state. As a more sophisticated baseline, we furthermore used SARSA(λ) (Rummery and Niranjan 1994) with discrete action set according to the original setup.[5] Finally, we included the preference-based policy iteration (PBPI) method of Fürnkranz et al. (2012) with the parameters reported by the authors.

Each policy learning method (EDPS, PB- EDPS, PBPI and SARSA(λ)) was run until reaching a limit 10,000 training episodes, and each policy found was evaluated on 300 virtual patients. That is, following the policy found by the learner, we generated 300 histories that represent the history of 300 virtual patients' fitness under the 6-months treatment. Based on these 300 treatment histories, we calculated the averages for C_X, the maximum toxicity level, as well as C_S, the tumor size at the end of the treatment for each policy. We repeated this process 100 times for each policy search method. Then, we plotted its mean and the 95 % confidence regions (assuming a multivariate normal distribution), which represent the uncertainty coming from the repetitions of the training process. As can be seen in Fig. 3, our approach is performing quite well and lies on the Pareto front of all methods (which remains true when adding the death rate, reported in Fig. 4b, as a third criterion).

As we were also interested in the rate of convergence of different policy learning method, we periodically evaluated the policies found by the methods during the learning process after a certain number of training episodes. Note that not each policy learner can be evaluated after an arbitrary number of training episodes. In the case of policy learners based on evolution strategies, such as EDPS and PB- EDPS, the set of candidate policies does change within an ES iteration (when the candidate policies are compared by using the racing algorithm). Therefore, we only evaluated the best policy taken from the current candidate policies

[5] We used an ϵ-greedy policy for exploration. Initially, the learning rate α, the exploration term ϵ and the parameter of the replacing traces λ were set to 0.1, 0.2 and 0.95 respectively, and decreased gradually with a decay factor $1/\lceil \frac{10}{\tau} \rceil$, where τ is the number of training episodes. We discretized each dimension of the state space into 20 bins and used a tile coding to represent the action-value function. We refer to Szepesvári (2010) for more details.



(a) Cumulative reward **(b)** Survival rate

Fig. 4 Convergence of different methods in terms of training episodes

after each ES iteration. Moreover, since the parameter $n_{\max}$ was adjusted dynamically (see Sect. 5), the number of training episodes generated within one ES iteration was also varying. In this way, after each run, we obtained a set of pairs (T_t, p_t), where T_t is the number of training episodes generated so far, and p_t is the performance of the current best policy after the t-th ES iteration (performance evaluation of a single policy is explained below).

Since SARSA updates the action-value estimates after each episode, policy evaluation could in principle be done after each training episode. Yet, for the sake of efficiency, we opted for a less costly alternative and evaluated in steps of 100 training episodes. PBPI generates 800 training episodes in a policy iteration step. Thus, we could evaluate it only after every 800 training episodes.

For a fair comparison, we evaluated the policies during the learning phase in terms of the cumulative reward (even if the method is preference-based, such as PBPI and PB- EDPS) and the survival rate on 300 virtual patients. These values were plotted as a function of the number of training episodes generated. We repeated the training process 100 times for each method. In the case of SARSA and PBPI, we simple plotted the average curve. For the ES-based methods, for which the number of training episodes is varying, we took all (T, p) pairs from each run and applied smoothing to obtain an average curve.

As one can see from the plots in Fig. 4, SARSA achieves the fastest convergence, and its performance in terms of cumulative reward is on a par with other methods. PBPI is outperformed by other policy learners in terms of cumulative reward. Nevertheless, the preference-based methods achieve a better survival rate than the value-based methods for a training episode smaller than 4,000.

6.3 Experiments with higher survival rate

According to our qualitative criterion, two histories are incomparable with probability at least p^2 if the death rate is p. Thus, even if incomparability does provide some information, too, the training data in the previous test scenario was not very informative for our preference-based policy learner.

Therefore, to obtain a higher survival rate, we changed the model parameters as follows: $a_1 = 0.1, a_2 = 0.15, b_1 = b_2 = 1.2, d_1 = d_2 = 0.5, c_0 = -8, c_1 = c_2 = 1.5$. Then,

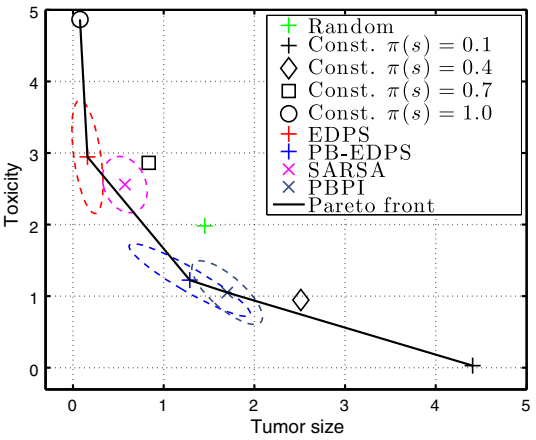
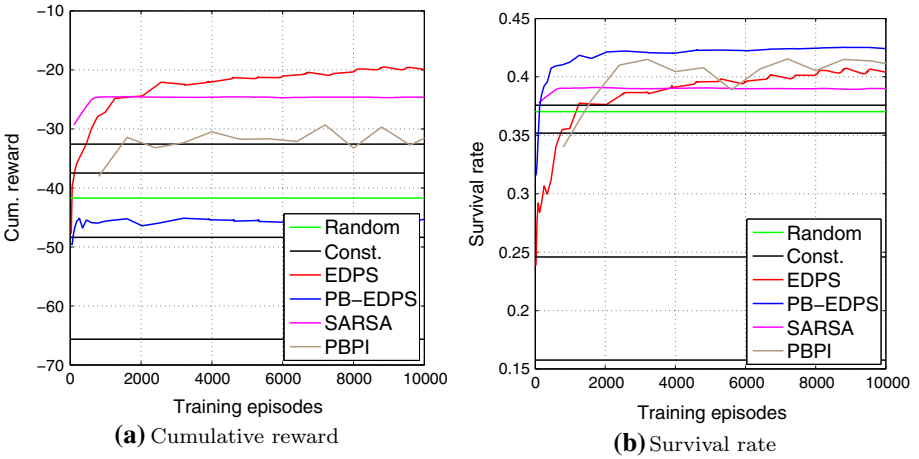


Fig. 5 Illustration of patient status under different treatment policies. On the x axis is the tumor size after 6 months, on the y axis the highest toxicity during the treatment



(a) Cumulative reward **(b)** Survival rate

Fig. 6 Convergence of different method in terms of training episode used

the same experiments were conducted and the same plots were produced as in the previous section. The maximum toxicity level as well as the tumor size C_S at the end of the treatment are plotted in Fig. 5. As before, the PB- EDPS is on the Pareto front, while the other preference-based method (PBPI) is close to the Pareto front. What is striking, however, is that the preference-based methods (PBPI and PB- EDPS) perform as well as the constant and random policies in term of cumulative reward (see Fig. 6a), while they outperform all other methods in terms of survival rate (see Fig. 6b). Moreover, as expected, they achieve faster convergence. This observation suggests that the reward signal cannot truly express the real objective in this test scenario.

7 Related work

The idea of preference-based reinforcement learning was introduced simultaneously and independently in Akrour et al. (2011) and Cheng et al. (2011). While preferences on histories

are taken as a point of departure in both approaches, policy learning is accomplished in different ways. As already mentioned, Cheng et al. (2011) generalize classification-based approximate policy iteration as proposed in Lagoudakis and Parr (2003) and Lazaric et al. (2010). To this end, they train a model that ranks actions given a state, using a preference learning method called *label ranking*. Instead of ranking actions given states, Akrour et al. (2011) learn a preference model on histories, which can then be used for policy optimization. To this end, policies are mapped to real (feature) vectors that represent important properties of the induced histories.[6] Then, a standard learning-to-rank method is applied in this behavioral representation space, in which preferences are expressed by an expert. More specifically, the authors learn a numerical utility function that agrees with the expert's preferences as much as possible. Then, given this utility function, policy search can be formalized as an optimization problem, namely as finding the policy that maximizes expected utility. In a follow-up work (Akrour et al. 2012), the authors extend this approach with the idea of active learning in order to decrease the number of preference queries asked to the expert.

In Wilson et al. (2012), the authors study the problem of learning expert policies via history preference queries to an expert. This setting can be viewed as an inverse reinforcement learning problem, where the behavior of the target policy can be accessed through preferences on pairs of short state histories originating from a common state. The authors propose a Bayesian approach to learn parametric policies, where the goal is to estimate the expert policy based on as few preference queries as possible. Active learning techniques are applied to generate histories that are maximally informative in the learning process.

Evolutionary Direct Policy Search (EDPS) was introduced by Heidrich-Meisner and Igel (2009), who use racing algorithms to control the number of rollouts in each iteration. Our approach can be seen as a generalization of EDPS. Thanks to the preference-based racing algorithm we developed, it does not require access to the policy performances themselves but only to a ranking over them.

The racing setup and the Hoeffding race algorithm were first considered by Maron and Moore (1994) and Maron and Moore (1997) in the context of model selection. For a detailed implementation of the HR algorithm, see Heidrich-Meisner and Igel (2009). This algorithm was improved in Mnih et al. (2008), where the empirical Bernstein bound was used instead of the Hoeffding bound. In this way, the variance information of the mean estimates could be incorporated in the calculation of confidence intervals.

In the context of multi-armed bandits, a slightly different setup was introduced in Even-Dar et al. (2002), where an ϵ-optimal random variable has to be chosen with probability at least $1-\delta$; here, ϵ-optimality of X_i means that $\mu_i + \epsilon \geq \max_{j\in[K]} \mu_j$. An algorithm solving this problem is called (ϵ, δ)-PAC bandit algorithm. The authors propose such an algorithm and prove an upper bound on the expected sample complexity. In this paper, we borrowed their technique and used it in the complexity analysis of PBR.

Recently, a PAC-bandit algorithm which is based on the widely-known UCB index-based muli-armed bandit method of Auer et al. (2002) was introduced in Kalyanakrishnan et al. (2012). In their formalization, an algorithm is an (ϵ, m, δ)-PAC bandit algorithm that selects the m best random variables under the PAC-bandit conditions. According to their definition, a racing algorithm is a $(0, \kappa, \delta)$-PAC algorithm. Instead of a high probability bound for the *expected* sample complexity, the authors manage to prove such a bound for the *worst case* sample complexity. It is an interesting question whether or not the technique used in their proof, which makes use of a specific type of slack variables, can also be applied in our setting.

[6] For example, a policy π can be mapped to the frequency vector of states obtained by following that policy (Abbeel and Ng 2004).

Yue et al. (2012) introduce a multi-armed bandit setup where feedback is provided in the form of pairwise comparisons between options, just like in our approach. However, their decision model is more restrictive than ours. It not only assumes $\gg$ to be a total order, but also requires additional properties such as strong stochastic transitivity and stochastic triangle inequality.

8 Conclusion and future work

By introducing a preference-based extension of evolutionary direct policy search, called PB-EDPS, this paper contributes to the emerging field of preference-based reinforcement learning. Our method, which merely requires qualitative comparisons between sample histories as training information (and even allows for incomparability), is based on a theoretically sound decision-theoretic framework and shows promising results in first experimental studies.

The core of our method is a preference-based version of the Hoeffding race algorithm, for which we could provide theoretical guarantees. Empirically, we have seen that this algorithm is not only more widely applicable than the original value-based version, but may even reduce sample complexity in (numerical) settings where both versions can be used. Therefore, the idea of preference-based racing should not be limited to reinforcement learning; instead, it seems worthwhile to explore it for other applications, too, such as multi-objective optimization with several competing objectives (Coello et al. 2007).

Coming back to our PB-EDPS framework, we hope to achieve further improvements by elaborating on its individual components. For example, the Copeland relation $\gg_C$ is not necessarily an optimal surrogate of the $\gg$ relation on policies, and indeed, voting and decision theory offers a large repertoire of alternative relations that could in principle be used. Moreover, we would like to investigate the use of Bernstein instead of Hoeffding races, since Bernstein-like bounds (which exploit the empirical variance of the estimates) are normally tighter than Hoeffding bounds. A Bernstein bound for two sample U-statistics can be obtained based on Peel et al. (2010), Arcones (1995). However, since the empirical bound requires an estimation of the standard deviation of the kernel (the function $h(\cdot)$ in (9) in the Appendix), and this estimation needs to be accompanied by a confidence bound, this is certainly a non-trivial problem.

Our theoretical analysis so far essentially focused on the racing algorithm, and therefore only covers a single iteration of the evolutionary search process implemented by PB-EDPS. Extending this analysis toward the convergence behavior of the complete search process is another important (and likewise difficult) topic to be addressed in future work.

Another interesting problem in this regard is to minimize the overall complexity of PB-EDPS by balancing the effort invested in a single selection step, i.e., a racing between the current candidates, and the number of generations needed by the evolutionary search process. In fact, one may suspect that finding the top-candidates with an extremely high probability is actually not necessary, since the selection of a less optimal candidate can be eliminated in the following iterations of the evolutionary search process. Lowering the guarantee $1-\delta$ for selection may significantly reduce the complexity of the racing algorithm, while more effort needs to be invested in the search. Finding a good tradeoff by adapting δ in a proper way is an interesting challenge. As an alternative to lowering the guarantee, i.e., playing with δ, one may also think of relaxing the requirement of finding the top-κ set exactly; in fact, for the same reasons just mentioned, it would arguably be enough to find a kind of ϵ-approximation of this set.

The computation of preferences over histories, which can be seen as querying an "oracle" (accepting two histories as input and providing a preference as output), can be costly in practical applications; for example, the oracle might be realized by a computationally complex simulator, or it could even be a human expert. Therefore, based on the response obtained so far, one may think of training a surrogate model to mimic the oracle, i.e., to replace the true oracle by a model for at least some of the queries. Initial work along this line has recently been presented in Weng et al. (2013), Akrour et al. (2013).

Last but not least, further experimental studies are of course needed to better understand the behaviours and performances of PBR and PB-EDPS on both synthetic problems and also challenging real-world domains.

9 Appendix

9.1 U-statistics

The *U-statistic* plays a central role in many practical statistical problems. For an independent sample set $x^{(1)}, \ldots, x^{(n)}$ drawn from the same distribution over Ξ, its general form can be written as

$$U = \frac{1}{\binom{n}{m}} \sum h\left(x^{(i_1)}, \ldots, x^{(i_m)}\right) \tag{9}$$

where the summation is taken over all subsets $\{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}$ of size m, and where the kernel funcion h is of the form $\Xi^m \to [a, b]$ for some $a, b \in \mathbb{R}$. An especially attractive feature of this statistic is that it is an *efficient* (or *minimum variance unbiased*) estimator (Serfling 1980). The U-statistic also generalizes to multiple samples in a natural way. For example, the general form of the *two-sample U-statistic* for two independent i.i.d. samples $x^{(1)}, \ldots, x^{(n)}$ and $y^{(1)}, \ldots, y^{(n')}$ drawn from Ξ and Υ, is

$$U = \frac{1}{\binom{n}{m}\binom{n'}{m'}} \sum h\left(x^{(i_1)}, \ldots, x^{(i_m)}, y^{(i'_1)}, \ldots, y^{(i'_{m'})}\right) \tag{10}$$

where the summation is taken over all subsets $\{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}$ of size m and $\{i'_1, \ldots, i'_{m'}\} \subseteq \{1, \ldots, n'\}$ of size m', and, similarly, the kernel function h is a bounded function of the form $\Xi^m \times \Upsilon^{m'} \to [a, b]$ for some $a, b \in \mathbb{R}$. The general form of the Hoeffding theorem for two-sample U-statistics can be written as follows.

Theorem 10 (Hoeffding 1963, §5b) *For any $\epsilon > 0$, using the notations introduced above,*

$$\mathbf{P}\left(|U - \mathbb{E}[U]| \geq \epsilon\right) \leq 2 \exp\left(\frac{-2k\epsilon^2}{(b-a)^2}\right)$$

where U is defined as in (10) and

$$k = \min\left(\lfloor n/m \rfloor \lfloor n'/m' \rfloor\right)$$

We applied Theorem 10 to Wilcoxon two-sample statistic in Sect. 4.1.

References

Abbeel, P., & Ng, A. (2004). Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21th international conference on machine learning*. New York, NY: ACM.

Akrour, R., Schoenauer, M., & Sebag, M. (2011). Preference-based policy learning. In *Proceedings ECMLP-KDD 2011, European conference on machine learning and principles and practice of knowledge discovery in databases* (pp. 12–27). Berlin: Springer.

Akrour, R., Schoenauer, M., & Sebag, M. (2012). April: Active preference-learning based reinforcement learning. In *Proceedings ECMLPKDD 2012, European conference on machine learning and principles and practice of knowledge discovery in databases* (pp. 116–131). Berlin: Springer.

Akrour, R., Schoenauer, M., & Sebag, M. (2013). Interactive robot education. In *ECML workshop on reinforcement learning with generalized feedback: Beyond numeric rewards*.

Arcones, M. A. (1995). A Bernstein-type inequality for u-statistics and u-processes. *Statistics & Probability Letters*, *22*(3), 239–247.

Audibert, J., Munos, R., & Szepesvári, C. (2007). Tuning bandit algorithms in stochastic environments. In *Proceedings of the algorithmic learning theory* (pp. 150–165).

Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, *47*, 235–256.

Beyer, H., & Schwefel, H. (2002). Evolution strategies-a comprehensive introduction. *Natural computing*, *1*, 3–52.

Cheng, W., Fürnkranz, J., Hüllermeier, E., & Park, S. (2011). Preference-based policy iteration: Leveraging preference learning for reinforcement learning. In *Proceedings ECMLPKDD 2011, European conference on machine learning and principles and practice of knowledge discovery in databases* (pp. 414–429). Berlin: Springer.

Coello, C., Lamont, G., & Van Veldhuizen, D. (2007). *Evolutionary algorithms for solving multi-objective problems*. Berlin: Springer.

Even-Dar, E., Mannor, S., & Mansour, Y. (2002). PAC bounds for multi-armed bandit and markov decision processes. In *Proceedings of the 15th annual conference on computational learning theory* (pp. 255–270). Berlin: Springer.

Fishburn, P. (1982). Nontransitive measurable utility. *Journal of Mathematical Psychology*, *26*, 31–67.

Fudenberg, D., & Tirole, J. (1991). *Game theory*. Cambridge, MA: MIT.

Fürnkranz, J., & Hüllermeier, E. (Eds.). (2011). *Preference learning*. Berlin: Springer.

Fürnkranz, J., Hüllermeier, E., Cheng, W., & Park, S. (2012). Preference-based reinforcement learning: A formal framework and a policy iteration algorithm. *Machine Learning*, *89*(1–2), 123–156.

Hansen, N., & Kern, S. (2004). Evaluating the CMA evolution strategy on multimodal test functions. In *Parallel problem solving from nature-PPSN VIII* (pp. 282–291). Berlin: Springer.

Heidrich-Meisner, V., & Igel, C. (2008). Variable metric reinforcement learning methods applied to the noisy mountain car problem. *Recent advances in reinforcement learning* (pp. 136–150). Berlin: Springer.

Heidrich-Meisner, V., & Igel, C. (2009). Hoeffding and Bernstein races for selecting policies in evolutionary direct policy search. In *Proceedings of the 26th international conference on machine learning* (pp. 401–408). New York, NY: ACM.

Heidrich-Meisner, V., & Igel, C. (2009). Neuroevolution strategies for episodic reinforcement learning. *Journal of Algorithms*, *64*(4), 152–168.

Hemelrijk, J. (1952). Note on Wilcoxon's two-sample test when ties are present. *The Annals of Mathematical Statistics*, *23*(1), 133–135.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, *58*, 13–30.

Kalyanakrishnan, S., Tewari, A., Auer, P., & Stone, P. (2012). PAC subset selection in stochastic multi-armed bandits. In *Proceedings of the Twenty-ninth International Conference on Machine Learning (ICML 2012)* (pp. 655–662). Omnipress.

Kreweras, G. (1961). Sur une possibilité de rationaliser les intransitivités. In *La décision, CNRS*.

Lagoudakis, M., & Parr, R. (2003). Reinforcement learning as classification: Leveraging modern classifiers. In *Proceedings of the 20th international conference on machine learning* (pp. 424–431). AAAI Press.

Laslier, J. (1997). *Tournament solutions and majority voting*. Berlin: Springer.

Lazaric, A., Ghavamzadeh, M., & Munos, R. (2010). Analysis of a classification-based policy iteration algorithm. In *Proceedings of the 27th international conference on machine learning* (pp. 607–614). Omnipress.

Maron, O., & Moore, A. (1994). Hoeffding races: accelerating model selection search for classification and function approximation. In *Advances in neural information processing systems* (pp. 59–66). Morgan Kaufmann.

Maron, O., & Moore, A. (1997). The racing algorithm: Model selection for lazy learners. *Artificial Intelligence Review*, *5*(1), 193–225.

Mnih, V., Szepesvári, C., & Audibert, J. (2008). Empirical Bernstein stopping. In *Proceedings of the 25th international conference on Machine learning* (pp. 672–679). New York, NY: ACM.

Moulin, H. (1988). *Axioms of cooperative decision making*. Cambridge: Cambridge University Press.

Ostermeier, A., Gawelczyk, A., & Hansen, N. (1994). A derandomized approach to self adaptation of evolution strategies. *Evolutionary Computation*, *2*(4), 369–380.

Peel, T., Anthoine, S., & Ralaivola, L. (2010). Empirical Bernstein inequalities for u-statistics. *Advances in Neural Information Processing Systems*, *23*, 1903–1911.

Prékopa, A., Yoda, K., & Subasi, M. (2011). Uniform quasi-concavity in probabilistic constrained stochastic programming. *Operations Research Letters*, *39*(1), 188–192.

Puterman, M. (1994). *Markov decision processes: Discrete stochastic dynamic programming*. New York: Wiley.

Rummery, G. A., & Niranjan, M. (1994). On-line Q-learning using connectionist systems. Tech. Rep. CUED/F-INFENG/TR 166, Cambridge University, Engineering Department.

Serfling, R. (1980). *Approximation theorems of mathematical statistics* (Vol. 34). Wiley Online Library.

Szepesvári, C. (2010). *Algorithms for reinforcement learning*. Morgan and Claypool.

Weng, P., Busa-Fekete, R., & Hüllermeier, E. (2013). Interactive q-learning with ordinal rewards and unreliable tutor. In *ECML workshop on reinforcement learning with generalized feedback: Beyond numeric rewards*.

Williams, R. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, *8*(3), 229–256.

Wilson, A., Fern, A., & Tadepalli, P. (2012). A bayesian approach for policy learning from trajectory preference queries. *Advances in Neural Information Processing Systems*, *25*, 1142–1150.

Yue, Y., Broder, J., Kleinberg, R., & Joachims, T. (2012). The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, *78*(5), 1538–1556.

Zhao, Y., Kosorok, M., & Zeng, D. (2009). Reinforcement learning design for cancer clinical trials. *Statistics in Medicine*, *28*(26), 3294–3315.