UNDER REVIEW

RESCIENCE C

Replication in Computational Neuroscience (Python)

[Re]Exploration in Model-based Reinforcement Learning by Empirically Estimating Learning Progress

Augustin Chartouny[1], **Jean-Pierre Nadal**[2,3] **and Mehdi Khamassi**[1]

[1]Institute of Intelligent Systems and Robotics, Centre National de la Recherche Scientifique, UMR 7222, Sorbonne Université, Paris, France – [2]Centre d'Analyse et de Mathématique Sociales, CAMS, UMR 8557 CNRS-EHESS, École des Hautes Études en Sciences Sociales, Paris, France – [3]Laboratoire de Physique de l'ENS, LPENS, UMR 8023, CNRS - ENS Paris - PSL University - SU - Université Paris Cité, Paris, France

Edited by
(Editor)

Reviewed by
(Reviewer 1)
(Reviewer 2)

Received
–

Published
–

DOI
–

The goal of a model-based reinforcement learning agent is to maximize external returns by understanding the structure of the environment it explores. However, simple reinforcement learning agents may struggle with tasks where external rewards are scarce. Taking inspiration from developmental sciences which show that infants spontaneously explore their environment in the absence of extrinsic rewards, intrinsically motivated reinforcement learning investigates how agents can explore their environments with little extrinsic motivation. In 2012, Lopes and colleagues proposed two new intrinsically motivated models in model-based reinforcement learning [1]. Building upon two optimistic-in-the-face-of-uncertainty agents, they demonstrate theoretical convergence properties for one of their agents, and provide three experiments to show that their models are more versatile than state-of-the-art models. However, due to missing information in their protocol, we only managed to partially reproduce the results of the original article. In our reproduction article, we show the results obtained for several alternative ways to interpret the text of the original article, and discuss what this implies in terms of performance of the different agents. For each tested variant, we performed parameter optimization in order to give the best chances to replicate the figures of the original article.

1 Introduction

In Lopes and colleagues' article, the authors present two new model-based reinforcement learning agents which use learning progress to motivate their exploration. Three experiments compared the performance of these two new agents with three other agents in a discrete environment and we try to reproduce the results of these experiments in this reproduction article. Lopes and colleagues did not provide their code and we did not find any existing code online. Nevertheless, one of the authors (Pierre-Yves Oudeyer) kindly responded to our questions via e-mail to help resolve some of the ambiguities we encountered when trying to interpret the methods described in the original article. An outline of our hypotheses can be found in Table 2. In Appendix C, we extend our results to other environments with a slight change in the reward function, similarly to the partial implementation of Bureau and Sebag in 2014 [2].

2 Methods

2.1 Reproduction of the environment

The environment used in Lopes and colleagues' article is a 5×5 discrete grid. We numbered the states from 0 to 24, starting from the top-left corner and counting from left to

Correspondence should be addressed to Augustin Chartouny (augustin.chartouny@isir.upmc.fr)
The authors have declared that no competing interests exists.
Code is available at https://github.com/AugustinChrtn/Reproduction.

right and from top to bottom. The goal of the agent is to reach and exploit the reward of 1 located in state 14, starting from the initial state 0 (see Figure 1). At each time-step, the agent chooses an action between $A = 5$ actions : *up*, *down*, *left*, *right* or *stay*. The task is episodic with 30 steps per episode and 100 trials in total per agent. At the end of a trial, the agent is moved back to the initial position 0. When the agent reaches the states 6, 7, 8 or 12, it gets a punishment of -0.1. The environment's transitions are stochastic: the actions of the agent have uncertain outcomes, even more so when the agent moves from the cells 1, 3, 11, 13. For every state, the most probable outcome of any action is the deterministic case. For example, if the action is *right*, the agent is most likely to end up directly to the right of the current cell (given that there is no wall on the right).

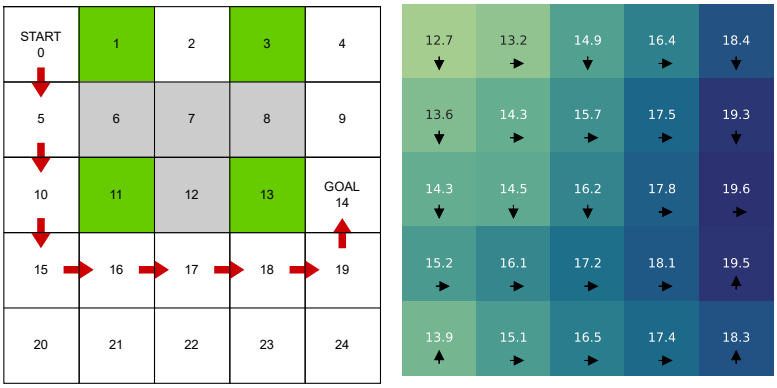

Figure 1. (Left) Representation of the environment, adapted from the first figure of Lopes et al. [1]. The green (light) states have uncertain transitions, the grey (dark) states incur a negative reward. The red arrows indicate the optimal path to the goal state. (Right) Implementation of the environment in Python, with the state values and optimal policies computed by value iteration.

In the original article, the authors claim that the shortest path is sub-optimal. However, all the paths to the goal state with the actions *down* and *right* only are the shortest paths. Excluding all the shortest paths means that the agent has to avoid the states with punishments or uncertain transitions, as shown by the red arrows of Figure 1-Left. The agent should then exploit the reward by staying in the goal state, either with the action *right* or the action *stay*.

Transition probabilities — According to the original article, the transition probabilities for a given (state, action) couple are sampled from a Dirichlet distribution with parameter $\alpha = 0.1$ for the uncertain states and $\alpha = 1$ for the other cells. However, it seems as though the authors may have swapped these two values. In a Dirichlet distribution, the bigger the α, the closer the probabilities. Thus, a Dirichlet distribution with $\alpha = 1$ provides transition probabilities that are closer on average than the one with $\alpha = 0.1$, meaning that the result of the action of the agent is more uncertain. This change of parameters matches the choice made by Bureau and Sebag [2] in their reproduction of the environment in which they swapped α values.
Pierre-Yves Oudeyer indicated by e-mail that the agents can only land on states that are one cell away from their current state (one cell up, down, left, right or the cell they are in). For example, starting from state 20 (bottom-left corner), the agent can only reach state 15, 20 or 21. Accordingly, we computed the transition probabilities for a state in three steps. First, we determined all the states which could be reached from this state. Second, we sampled a Dirichlet distribution of 0.1 or 1 on these states for each action. Finally, we swapped two probabilities to guarantee that the deterministic state would be the most probable outcome.

Optimal path statistics — The original article claims that shorter paths are sub-optimal and that the agent needs to avoid cells with a negative reward or uncertain transitions. Nevertheless, with a discounting factor $\gamma = 0.95$, only 1.7% of 5000 randomly generated environments displayed the optimal path, indicated by red arrows in Figure 1. In their partial reproduction of the environment, Bureau & Sebag [2] used a negative reward of -1 instead of -0.1. With this negative reward, 24.7% of the 5000 environments tested displayed the optimal path. We provide simulations in these environments in Appendix C. The heat-map in Figure 1 shows the environment with negative rewards of -0.1.

2.2 Measures of performance

Since all the agents are model-based, one way to compute their level of exploration is to estimate their capacity to understand the transitions and rewards of the environment. To do so, Lopes and colleagues calculate the optimal policy $\pi^*_{\hat{T}}$ in the estimated environment and compare it to the optimal policy in the true environment π^*_T using value iteration. Every 50 steps, we computed $V_{T,R}(s_I; \pi^*_{\hat{T}}) - V_{T,R}(s_I; \pi^*_T)$, where $V_{T,R}$ is the value obtained after value iteration, where T, R represent transitions and rewards respectively and $s_I = 0$ the initial state. In our simulations, since the role of the sampled rewards is not indicated in the article, we used $V_{T,R}(s_I; \pi^*_{\hat{T},\hat{R}}) - V_{T,R}(s_I; \pi^*_{T,R})$.
A limitation of this measure is that a random agent's performance would converge ultimately to the best performance, despite the fact that it takes random actions. To resolve this, we also computed a similar measure of performance every 50 time steps, but on the agent policy (instead of the optimal policy): $V_{T,R}(s_I; \pi_A) - V_{T,R}(s_I; \pi^*_{T,R})$ where π_A is the policy of the agent. We present the results with this measure in Appendix B. This helps us appraise the agents' performance through the lens of a set of different measures.

2.3 Reproduction of the agents

The five agents outlined in the article are model-based: their goal is to learn the rewards and the transitions for every (state, action) couple. An important insight that Pierre-Yves Oudeyer provided to us via e-mail is that the agents know the states before exploring their environment. For every agent, we set uninformative reward and transitions at start. The rewards are set to 0 and the transitions are uniform over all the states (the probability to reach any state is $\frac{1}{25}$).
When an agent reaches a new state, it gets the reward associated with this state. However, the reward computed by the agent depends on the state they are in and the action they take $\hat{R} : S \times A \to \mathbb{R}$. $\hat{R}(s, a)$ is the average reward of the agent when it takes the action a in s. To estimate the transition probabilities, most of the agents use a sampling approach. Let $n(s, a)$ be the number of times the agent chose action a in state s and $n(s, a, s')$ the number of times it reached the state s' by taking the action a in state s. The transition model $\hat{T}(s, a)$ estimated by the agent is

$$\hat{T}(s, a, s') = \frac{\hat{n}(s, a, s')}{\hat{n}(s, a)}. \tag{1}$$

The Bayesian Exploration Bonus (BEB) agent uses a different approach (see Equation 4).

Exploiting the model — At each step and for every visited (state, action) couple, the agents compute the Q-values through value iteration. The Q-values are changed only for visited (state, action) so that optimistic initializations are efficient. If the Q-values changed by less than 10^{-3} between two iterations, the agents consider that the value iteration algorithm converged. Let R_{VI} be the value of the reward used in the value iteration algorithm. For all (s, a), the Q-values are

$$Q(s,a) = R_{VI}(s,a) + \gamma \sum_{\hat{T}(s,a,s')>0} \hat{T}(s,a,s') \max_{a'} Q(s',a'). \tag{2}$$

Model-based ϵ-greedy — The ϵ-greedy agent takes a random action with the probability ϵ or chooses the action associated with the maximal Q-value with the probability $1-\epsilon$. Lopes and colleagues indicate that the initial Q-values of ϵ-greedy are optimistic. As 1 is the maximal reward of the environment and $\gamma < 1$, we set the initial Q-values to $\frac{1}{1-\gamma} = 20$ for $\gamma = 0.95$. For ϵ-greedy, the reward in the value iteration algorithm $R_{VI}(s,a)$ is equal to $\hat{R}(s,a)$.

R-max — R-max is optimistic in the face of reward uncertainty and splits the (state, action) couples between known and unknown [3]. For each (state, action) couple (s,a), there is a fixed number of passages $m(s,a)$ before which the state is unknown and the reward used in value iteration is the maximal reward of the environment $R_{max} = 1$. After $m(s,a)$, the value of the reward in the value iteration process is the sampled reward $\hat{R}(s,a)$.

$$R_{VI}(s,a) = \begin{cases} R_{max} \text{ if } n(s,a) < m(s,a). \\ \hat{R}(s,a) \text{ if } n(s,a) \geq m(s,a), \end{cases} \tag{3}$$

Since R-max splits the (state, action) couples between known and unknown, we could update the Q-values only for known couples, when a new couple is added to the known list. In addition, the agent could stop updating the reward and transition tables of a known state. Such a method would speed up the computation time in a stationary environment [4]. However, Lopes and colleagues also measure the capacity of the agents to adapt to non-stationary environments, and using this inference process would be detrimental to R-max performance in such a case. This is one of the reasons why we chose the same inference process for all the agents, with $\hat{R}$, $\hat{T}$ and Q being updated at each step.

Lopes and colleagues indicate that the number of passages $m(s,a)$ is informative and depends on the transition noise of (s,a). The simplest way to provide an informative number of minimal passages $m(s,a)$ to the model is to choose two values of m, depending on the Dirichlet distribution used for the transition uncertainty. A more informative approach would be to use a function to calibrate the counts more precisely for each (s,a), for example with a function of the entropy of the transitions. However, the article presents no such function and we did not find conclusive results with this method. Thus we used the first method, with two values of m depending on the state stochasticity. All but the four uncertain states have the same m with m_u being the minimal number of passages for the uncertain states, where the subscript u stands for uncertain. The agents may need more passages to acquire a good model of uncertain states and we found that in general, m_u would be greater m, although we explored all possibilities in the parameter fitting presented in Appendix A. We set the initial Q-values of R-max to $\frac{1}{1-\gamma}$.

Bayesian Exploration Bonus (BEB) — Bayesian Exploration Bonus (BEB) is a Bayesian agent optimistic in the face of reward uncertainty [5]. The transitions are represented by a Dirichlet distribution $b = \{\alpha(s,a,s')\}$ which accounts for the uncertainty on the transitions. At every step, the transition model is

$$\hat{T}(s,a,s'|b) = \frac{\alpha(s,a,s')}{\sum_{s'} \alpha(s,a,s')}. \tag{4}$$

This Dirichlet prior facilitates the update of the Bayesian distribution at each new observation. When the agent takes the action a in state s and reaches state s', the distribution

is updated according to $\alpha(s,a,s') \leftarrow \alpha(s,a,s') + 1$. In practice, the evolution of the estimated transitions is close to the previous sampling models where α plays a similar role to that of n, while bringing uncertainty to the transitions.
Rather than classifying (s,a) between known and unknown couples like R-max does, BEB uses an exploration bonus which decreases linearly with the number of passages $n(s,a)$

$$R_{VI}(s,a) = \hat{R}(s,a) + \frac{\beta}{1 + \sum_{s'} \alpha(s,a,s')}, \tag{5}$$

where β is a parameter of exploration (the higher the β, the more the agent explores). If we note $\alpha_0(s,a)$ the sum of the initial priors on the transitions for (s,a), the denominator of Equation 5 becomes $1 + \alpha_0(s,a) + n(s,a)$. Lopes and colleagues indicate that the prior of BEB is informative. However, Kolter and Ng [5] only used the transitions of the environment as an informative prior. As the transitions estimated by the agents are sampled from the priors, this would mean that BEB could have a very good model of the transitions of the environment from the start and not learn much (see subsection 3.2 for an example).
To provide informative but small prior knowledge to the BEB agent, we chose to provide it the real transitions of the environment but with a fixed prior coefficient of 2. This is consistent with the values between 1 and 5 found in Kolter and Ng [5]. In addition, we set all the prior $\alpha(s,a,s')$ to an arbitrary small positive value of 10^{-5} so that Dirichlet distributions are still defined in non-stationary experiments. We set the initial Q-values of BEB to the upper bound $\frac{1+\beta}{1-\gamma}$.
Like R-max, BEB can use known and unknown (state, action) couples [5]. After a given number of passages $m(s,a)$, the agent can stop updating the transition and reward tables. Not updating the transitions and rewards after a given number of passages would improve the computational cost of the simulations but would be detrimental for BEB in non-stationary tasks. As for R-max, we chose to update the transition, reward and q-values tables at every time step.

Learning progress — Lopes and colleagues built two new models based on R-max and BEB. Their models use learning progress instead of a count-based approach to explore towards the reduction of transition uncertainty. Their models evaluate how much they know about the transitions with a leave one out cross-validation. Let $D_{s,a}$ be all the states reached by taking action a in s, with $|D_{s,a}| = n(s,a)$, the leave one out cross validation $CV(s,a)$ is

$$CV(D_{s,a}, s, a) = -\frac{1}{|D_{s,a}|} \sum_{s' \in D_{s,a}} \ln \hat{T}^{-s'}(s,a,s'), \tag{6}$$

with $\hat{T}^{-s'}(s,a,s')$ being the transition model obtained from $D_{s,a}$, minus one occurrence of s'. With a sampling approach, if the agent has been to state s' only once, the transition model which excludes this one time gives a zero probability to reach the state s' and the cross validation diverges. Thus, we chose to set a small prior of 10^{-2} to compute the learning progress, so that the minimal transition probability used in the cross-validation is always strictly positive.
Rather than directly considering the value of the cross-validation, the learning progress is defined as the variation between two cross-validations. With this method, learning progress models do not remain motivated by transitions with a strong but expected transition entropy. The value of the learning progress $\zeta(s,a)$ is the difference between two cross-validations $k = 10$ passages apart, summed with a variance parameter to add stability and bring theoretical properties presented in the original article.

$$\begin{cases} \zeta(s,a) = CV(D^{-k}_{s,a}, s, a) - CV(D_{s,a}, s, a) + \alpha\sqrt{v(s,a)}, \\ v(s,a) = \frac{1}{|D_{s,a}|} \sum_{s' \in D_{s,a}} [CV(D_{s,a}) + \ln \hat{T}^{-s'}(s,a,s')]^2, \end{cases} \tag{7}$$

where α adjusts the importance of the variance.
The cross-validation on the arrival states can be higher than the cross-validation computed $k = 10$ steps before, for example if the agent has found a new arrival state within the ten new occurrences. In such cases and with a low value of α, the learning progress can be negative, although it has to be strictly positive in Equation 9. There could be several ways the authors might have dealt with this issue. The simplest method would be to put a small positive minimal threshold for the learning progress, so that it remains strictly positive. Another idea would be to use absolute values for the differences between the two cross validations, as it may allow for the detection of task changes. A positive and high difference between cross-validations illustrates that the agent is learning from the new experiences and thus needs more passages to refine its model (as was mentioned by Lopes and colleagues). On the contrary, when the difference between the cross-validations is negative and high in absolute value, this indicates that the model of the agent has significantly worsened and that there might have been a task change. With the use of absolute values, the agent would remain motivated to sample new transitions, whereas this case would be neglected otherwise.
Though we would have preferred using absolute value, we instead chose to use a small minimal threshold for the learning progress so as not to modify the equations from the original article. We set the minimal value of the learning progress to 10^{-3}.

The authors do not indicate the initial value of the learning progress before $k = 10$ passages in a (state, action) couple. We chose to set a high initial learning progress value, so that the agents are motivated to explore the unknown (state, action) couples before focusing on the ones whose learning progress is maximal. Since, we did not find a theoretical upper bound of the learning progress and we did not know if using one would be useful, we chose to set the initial learning progress value to an arbitrary high value. In practice and with our sets of parameters, we found out that 10 was (almost) never reached and set it as the initial learning progress value.

The learning progress ζ used in ζ-R-max and ζ-EB replaces the count n in R-max and BEB, with β and m having similar roles as before. Changing the R-max agent to the ζ-R-max agent is rather straightforward by modifying the reward function used in the value iteration. However, we found no fully satisfying way to implement the learning progress version of BEB. A first approach would be to use priors. With small uniform priors, and the transition probabilities sampled from a Dirichlet distribution, the probabilities may be small enough to be considered computationally equivalent to 0, which would make the cross-validation diverge. On the other hand, a strong uninformative prior could slow down the learning process because false prior information would weigh in the transition probability computation. Thus, the approach that we chose was to use a non-Bayesian agent. The name ζ-EB used in the original article instead of ζ-BEB, may indicate that the agent is not Bayesian, although we found no explicit proof in the original article. The version of ζ-EB we chose is close to ζ-R-max but with a different reward function:

$$\text{For } \zeta\text{-R-max,} \qquad R_{VI}(s,a) = \begin{cases} \hat{R}(s,a) & \text{if } \zeta(s,a) < m, \\ R_{max} & \text{if } \zeta(s,a) \geq m. \end{cases} \tag{8}$$

$$\text{For } \zeta\text{-EB,} \qquad R_{VI}(s,a) = \hat{R}(s,a) + \frac{\beta}{1 + \frac{1}{\sqrt{\zeta(s,a)}}}. \tag{9}$$

Parameter fitting — Some parameter values are missing from the original article, including the discount factor γ. By comparing the starting value policy errors in the figures from the article, around -12 for each figure, and our simulations, we found that the discounting factor may be $\gamma = 0.95$. The specific parameters we needed to infer for each agent were

- ϵ for ϵ-greedy,
- R_{max}, m, and m_u for R-max,
- β and the prior coefficient for BEB,
- R_{max}, α, m, k and the prior for learning progress for ζ-R-max,
- α, β, k and the prior for learning progress for ζ-EB.

Out of all these parameters, we only found the values $k = 10$ and $R_{max} = 1$ in Lopes and colleagues' article. We set the value of the prior coefficient for the agent BEB to 2 and the uniform prior of learning progress to 10^{-2}. We found all the other parameters by grid search parameter fitting. We provide heat-maps of performance depending on parameter couples for all the models in Appendix A. The corresponding one dimensional graphs are provided online.

3 Reproduction of the experiments

We used the same set of values for all the reproduction figures, with $\gamma = 0.95$. We found the values of the parameters through parameter fitting on the stationary environment of the first experiment.

Agent	Parameters
ϵ-greedy	$\epsilon = 0.3$
R-max	$R_{max} = 1$, $m = 8$, $m_u = 12$
ζ-R-max	$R_{max} = 1$, $m = 2$, $k = 10$, $\alpha = 0.3$, prior $= 0.01$
BEB	$\beta = 7$, coeff prior $= 2$
ζ-EB	$\beta = 3$, $k = 10$, $\alpha = 0.1$, prior $= 0.01$

Table 1. Values of the parameters we used in our simulations.

3.1 Experiment 1. Stationary environment

The graphs show the mean performance and the standard error of the mean over 20 trials in the environment of Figure 1. In the original article, the goal of this experiment is to demonstrate that ζ-R-max and ζ-EB converge almost as fast as R-max and BEB respectively, although they do not have prior information on the environment transitions.

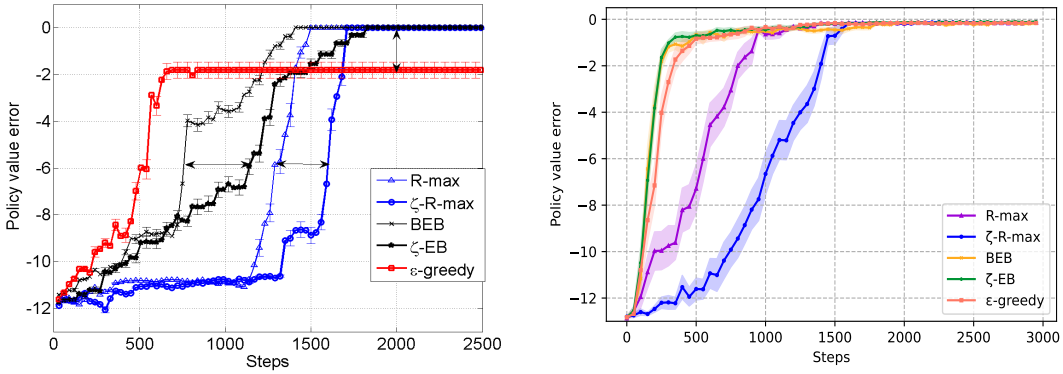

Figure 2. Reproduction of the first experiment of the article. (Left) Picture from the original article [1], reused with the agreement of one of the co-authors, Pierre-Yves Oudeyer. (Right) Picture we generated using the parameters indicated in Table 1.

In our simulations and as in the original article, ζ-R-max converges after R-max. Nevertheless, the agents converge much faster in our simulations than in the original article. In addition, the shape of the policy value error function, the convergence time and the convergence value of ϵ-greedy, BEB and ζ-EB are comparable. One other main difference is that the model-based ϵ-greedy agent converges to the optimal policy.

3.2 Experiment 2. The role of priors

Wrong prior experiment — In the second experiment of the original article, the priors of BEB and R-max are wrongly specified. The goal of this experiment is to show that ζ-R-max and ζ-EB do not need prior information, as opposed to R-max and BEB whose performance depends on good informative priors. The wrong priors are sampled randomly from a uniform distribution between the lowest value and highest value of the previous priors. The passage counts for R-max are taken randomly between m and m_u and the value of the prior of BEB for every triplet (old state,action, new state) is taken randomly between 10^{-5} and the biggest transition probability prior from the first experiment.

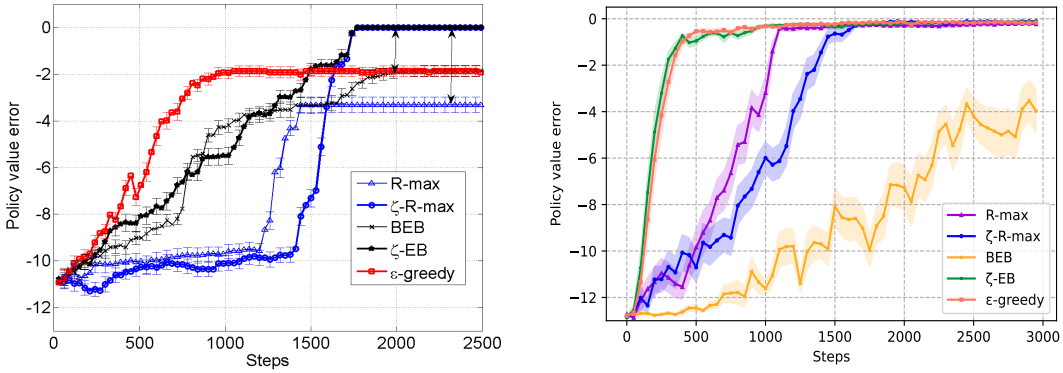

Figure 3. Reproduction of the second experiment of the article, original (left) vs simulated (right). The picture from the original article [1] was reused with the agreement of one of the co-authors, Pierre-Yves Oudeyer. In our simulations, the performance of BEB drops with wrong priors, whereas R-max performs better than in the original article.

This second experiment is partially reproduced as the performance of BEB drops in the wrong prior condition. Nevertheless, the performance of R-max drops less than in the simulations from the original article. This may be because the prior information given to R-max is not informative enough and that the minimal number of passages are sampled between two values that are close, $m = 8$ and $m_u = 12$.

Another difference between our reproduction figure and the one from the original article is that our version of BEB does not reach a plateau within the 3000 trials. This may result from the fact that BEB updates the state and transition tables throughout the whole experiment in our implementation, whereas Lopes and colleagues may have used a version of BEB which stops updating its model of a (state, action) couple after a given number of passages, thus converging faster but to a sub-optimal policy.

Uniform prior experiment — Lopes and colleagues' took informative priors for both BEB and R-max, arguing that R-max would take more time to converge and BEB might not converge. However, using strong priors could worsen the performance of the agents, especially in non-stationary tasks or if they are wrongly specified. In addition, Kolter and Ng [5] showed that BEB can perform well with small uninformative priors. As a consequence, we conducted simulations with uniform priors for R-max and BEB. In Figure 4-Left, we plot R-max with a value of $m = 10$ for all (state, action) couples and BEB with

a small uniform prior of 10^{-3}. R-max and BEB quickly converge to the optimal policy, demonstrating that using an informative prior might not be relevant in this task, when measuring the policy value error.

BEB oracle — In addition, giving strong informative priors to Bayesian agents could be similar to giving them the very transitions of the environment. In Figure 4-Right, we show the performance of BEB with informative prior corresponding to the transitions of the environment, but with different prior coefficients. With high prior coefficients, BEB can become an oracle (*i.e.,* an omniscient agent very rapidly reaching optimal performance).

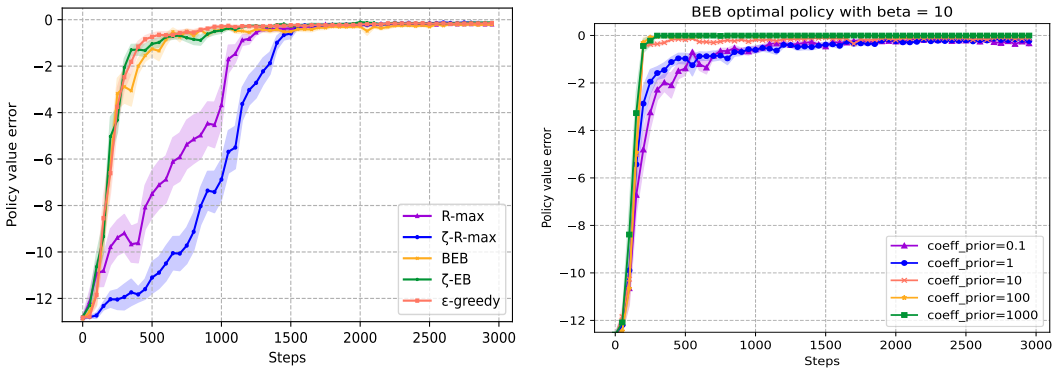

Figure 4. (Left) Performance of the agents without informative prior. The performance of BEB and R-max is close to the one shown Figure 2, demonstrating that the agents may not need informative priors. (Right) Influence of the priors on the performance of the BEB agent. With a strong informative prior, BEB can be an oracle.

3.3 Experiment 3. Non-stationarity

Task change — The third experiment of the article introduces a task change. At step 900, the transitions for one state on the optimal path of the environment are swapped. For example, the transition probabilities of *up* can become the ones for *left* and vice versa. In our code, we guaranteed that the permutation of the five actions had no fixed point (no action keeps the same transition vector). For this experiment, we randomly generated 20 non-stationary environments and ran the simulations 5 times in each environment. Since we ran the experiment on more trials than in the original article, the standard errors of the mean in our simulations seem smaller than in the original plot.

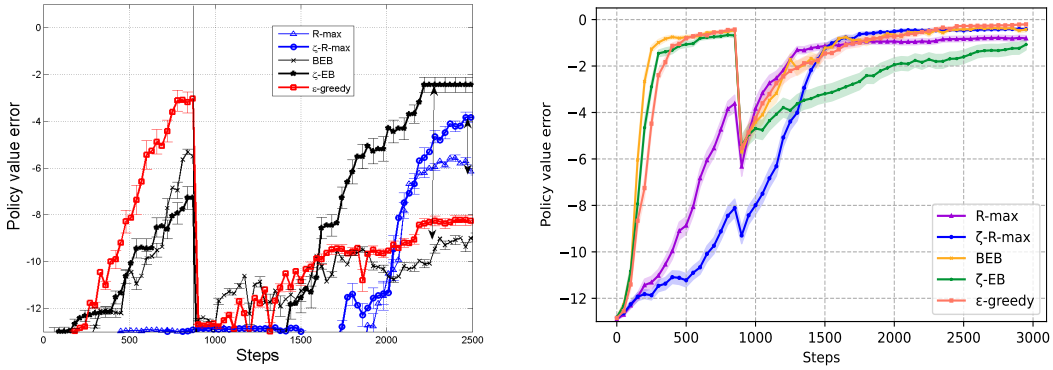

Figure 5. Third experiment of the original article (left) compared to our simulations (right). The picture from the original article [1] was reused with the agreement of one of the co-authors, Pierre-Yves Oudeyer. The policy value error drops more in the original experiment than in our simulations and learning progress agents do not perform better than the other agents.

Following indications from the article, we did not manage to reproduce the sharp drop in policy value error when a single state is changed (Figure 5). In addition, all agents adapt to the task change well, even if ζ-EB performs slightly worse. The drop in performance of the figure from the original article seems surprising as a simple change on the optimal path made the agent perform as if it just discovered the environment and did not know most of its structure.

Stronger task change — A possible interpretation of the original figure is that the authors may have operated a more important task change, for example by changing the transitions on not just one but several states. As a consequence, we present a new version of the third experiment with a stronger task change in Figure 6. In this new version, all the transitions for all the states in the optimal path are swapped without fixed points. The figure shows that imposing transition changes on all the states of the optimal path leads to a better replication of the sharp drop in performance observed in the original article (Figure 5-Left).
Another point we would like to highlight is the importance of the horizon. In the original article, some agents may have not converged after the task change. In Figure 6, we show two horizons of 3000 steps and 6000 steps. With this new horizon, we leave more time for the agents to converge, and at step 6000, ϵ-greedy outperforms all the agents.
In Figure 6-Right and as in the original article, ζ-R-max converges to a better policy than R-max. Nevertheless, we found opposite results for BEB and ζ-EB.

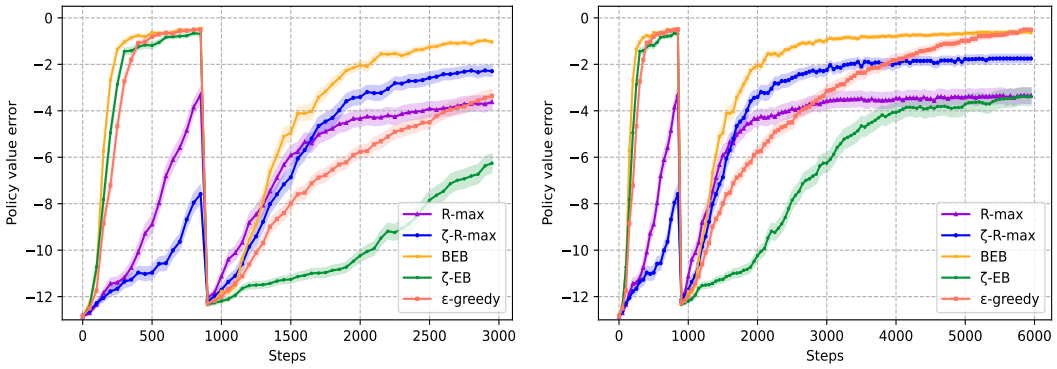

Figure 6. Stronger task change with a time horizon of 3000 steps (left) and 6000 steps (right). BEB and ζ-R-max outperform ζ-EB and R-max respectively. At step 6000, the agent with the best optimal policy is ϵ-greedy.

4 Discussion

Overall, the full reproduction of Lopes and colleagues' article was difficult to attain due to missing information in the original article. We nevertheless managed to generate results that approximately confirm the main conclusions of the article after making a series of modifications. Some general indications are missing from the article, such as the value of γ, the priors of BEB and R-max or an explicit method to implement the learning progress. We made a list of the non-exhaustive problems that were encountered and the hypotheses we made in Table 2.

With the optimal policy value error, the graphs we generated seem too far from the graphs shown in the article. In the stationary experiment, the agents from our simulations converge faster and the ϵ-greedy converges to the optimal policy. In the non-stationary experiment, we did not manage to simultaneously reproduce the sharp drop in general performance and the agents' individual performances. Even if we reproduced this drop with another task change (*i.e.,* changing the transition probabilities on all states of the optimal path rather than on a single state), BEB outperforms ζ-EB, and ϵ-greedy has a good performance. Nevertheless, we found the same qualitative results for the wrong prior experiment with BEB not converging to the optimal policy, and using a more informative prior for R-max could lead to similar results. However, we demonstrated that both models explore well without informative priors, which nuances some of the conclusions of the original article.

Thus, we believe that the authors may instead have used the real policy of the agent (see Appendix B). Since the agent policy is updated at every time-step and until convergence, it is similar to computing the optimal policy but with R_{VI} instead of $\hat{R}$ and with a penalty for ϵ-greedy, the only model without a greedy choice on the Q-values. With the agent policy, the convergences of R-max, ζ-R-max and ϵ-greedy are fairly well reproduced (Appendix B). As in the original article, ϵ-greedy cannot converge to the best policy because ϵ introduces randomness in the decision making. At start, R-max and ζ-R-max have a poor performance because of the value of R_{max}, and the performance of the two agents increases abruptly when they start using $\hat{R}$.

The main difference between our reproduction and the original article is the performance of ζ-EB, which performs worse in our simulation than in the original article. This might have been because we drew incorrect hypotheses or implementations from the descriptions written in the original article, such as assuming that ζ-EB is not Bayesian. We would need further indications from the authors to come up with an efficient ζ-EB.

Problem encountered	Attempted solution	Result
No code available	Contacted the authors	No code was provided
What states can the agent reach from a given state?	Contacted the authors	The agents can only access the states next to them
The cross validation diverges if the agents do not know the states	Contacted the authors	The agents know the states from the start
The authors may have swapped $\alpha = 0.1$ and $\alpha = 1$	Checked the values in [2] and provided a theoretical explanation	We swapped back the two values
No indication for the learning progress before $k = 10$	Used an arbitrary high value for the learning progress before $k = 10$	The learning progress is not computed before $k = 10$
The learning progress can be negative	Set a minimal value of 10^{-3}	The learning progress is strictly positive
With a strong informative prior, BEB is an oracle	We gave BEB the transitions of the environment with a small coefficient	BEB still learns the environment structure
Why did the authors use informative priors?	We tried to use uninformative priors for BEB and R-max	The agents attain good policies even with uninformative priors
When should the agents compute the Q-values?	At every time-step and until the convergence of the value iteration algorithm	All agents exploit the environment in the same way
Initial optimistic Q-values are useless if we update the Q-values at each step	We updated the Q-values of the visited states only	The initial optimistic Q-values of $\frac{1}{1-\gamma}$ and $\frac{1+\beta}{1-\gamma}$ are efficient
Are there one or twenty environments?	We used one environment with a punishment of -0.1 and ten for -1	We cannot conclude on what environments they used but covered both hypotheses
Initial values of policy value errors are different between the experiments	Unexplained if they used one environment	They might have used different environments for each experiment
We do not know what informative prior to use for R-max	We took two values depending on whether the state is uncertain or not	Our prior may be less informative than the one they used in the article
The performance of ϵ-greedy is high compared to the one of the article	We used another measure of performance which depends on the agent policy	ϵ-greedy performs worse with this measure
The value of γ is missing	We chose γ with the initial policy value errors	We used $\gamma = 0.95$
Many parameter values are missing from the article	We provided our parameter fitting in Appendix A	The agents are optimized
The task change description might be wrong	We provided figures with a stronger task change	This task change reproduces the drop in performance
Cross validations diverge with sampling methods	We used a small learning progress prior	Cross validations do not diverge anymore
We do not know whether ζ-EB is Bayesian or not	We infered that ζ-EB is not Bayesian	ζ-EB implementation might be wrong

Table 2. What hypotheses did we make?

5 Conclusions

We did not manage to fully reproduce the results presented by Lopes and colleagues. However, by replicating some of their results and providing our code, we hope that others will be able to implement the learning progress agents in more diverse environments and compare them to other intrinsically motivated reinforcement learning agents.

Conflicts of interest

The authors declare no conflict of interest.

Funding

This work was funded by the ANR project ELSA.

Acknowledgements

We would like to thank Pierre-Yves Oudeyer and Michèle Sebag for useful indications on the environment of the original article, Benoît Girard and Olivier Sigaud for fruitful discussions on the topic and Clarissa Montgomery for proofreading the first version of the manuscript.

Copyright

In this replication article, we reused three figures from the original article [1], published in 2012 in Neural Information Processing Systems. According to the journal's Copyright policy, authors do not transfer their Copyright to the journal. Thus, we reused the three figures with the agreement of Pierre-Yves Oudeyer, co-author and co-owner of the Copyright.

References

1. M. Lopes, T. Lang, M. Toussaint, and P.-y. Oudeyer. "Exploration in Model-based Reinforcement Learning by Empirically Estimating Learning Progress." In: **Advances in Neural Information Processing Systems**. Ed. by F. Pereira, C. Burges, L. Bottou, and K. Weinberger. Vol. 25. Curran Associates, Inc., 2012.
2. A. Bureau and M. Sebag. "Bellmanian bandit network." In: **Autonomously Learning Robots, at NIPS 2014**. 2014.
3. R. I. Brafman and M. Tennenholtz. "R-max-a general polynomial time algorithm for near-optimal reinforcement learning." In: **Journal of Machine Learning Research** 3.Oct (2002), pp. 213–231.
4. A. L. Strehl, L. Li, and M. L. Littman. "Reinforcement Learning in Finite MDPs: PAC Analysis." In: **Journal of Machine Learning Research** 10.11 (2009).
5. J. Z. Kolter and A. Y. Ng. "Near-Bayesian exploration in polynomial time." In: **Proceedings of the 26th annual international conference on machine learning**. 2009, pp. 513–520.

A Parameter fitting for the two measures

We led a grid-search parameter fitting for the two measures presented in subsection 2.2 (optimal and agent policy) for the environment with a punishment of -0.1. The data is available on Github.

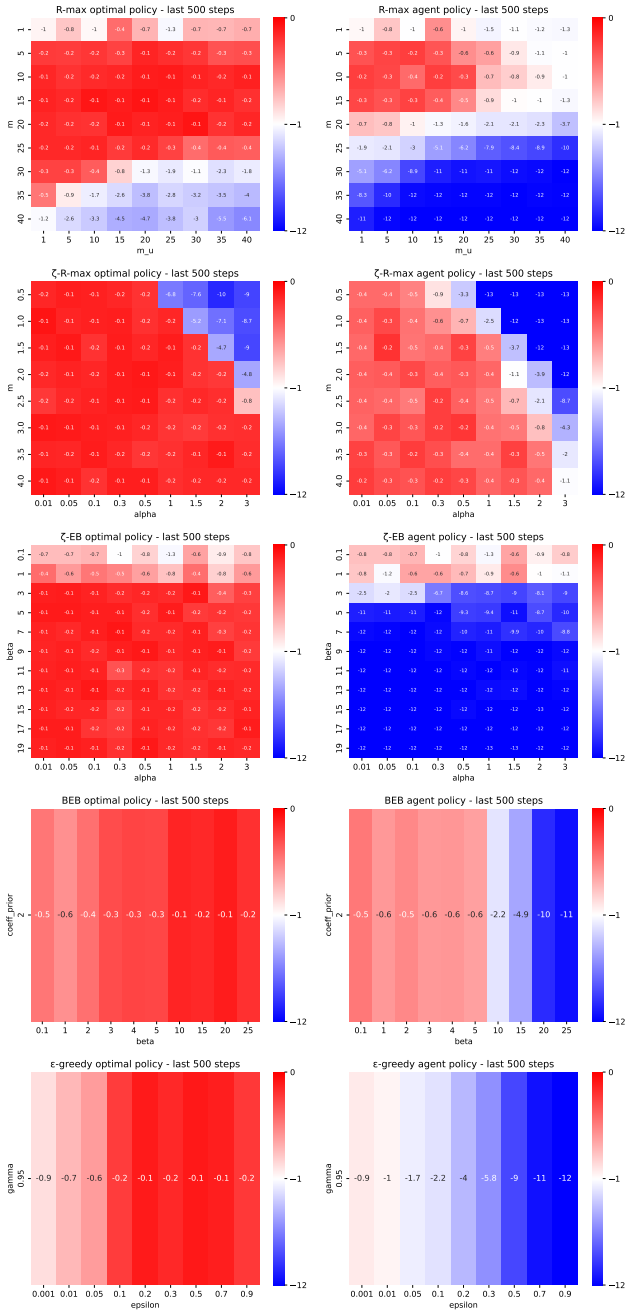

Figure 7. Parameter fitting for all the agents in Lopes and colleagues' environment. The left column corresponds to the optimal policy and the right one to the agent policy. We look at the average value the last 500 steps on 20 trials for each couple or singleton of parameters. From top to bottom: R-max, ζ-R-max, ζ-EB, BEB and ϵ-greedy. The color-scale is the same for all the agents.

Then, when needed, we focused on the red areas of Figure 7 to improve the precision of the parameter fitting (see Figure 8).

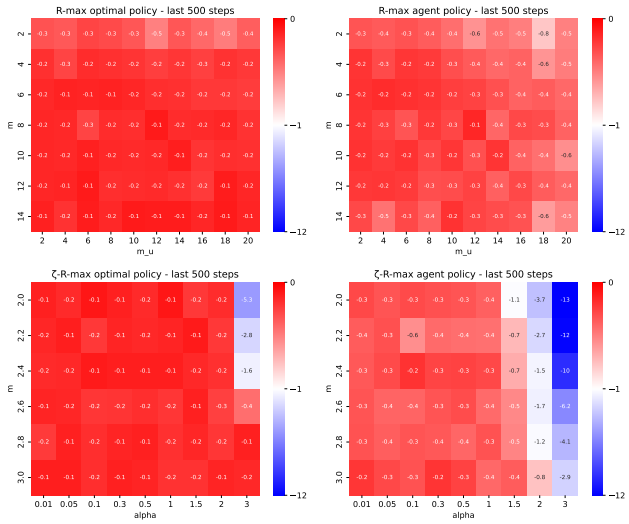

Figure 8. More precise parameter fitting for R-max (top) and ζ-R-max (bottom) using the optimal policy value error (left column) and the agent policy value error (right column).

In the parameter fitting of Figure 7, we see that ζ-EB is difficult to optimize and that there is no good parameter couple which optimizes the performance of the agent on the two measures (the optimal policy and the agent policy). In Figure 9, we tried to elicit a range of parameters for which the ζ-EB model performs well on the real agent policy metric.

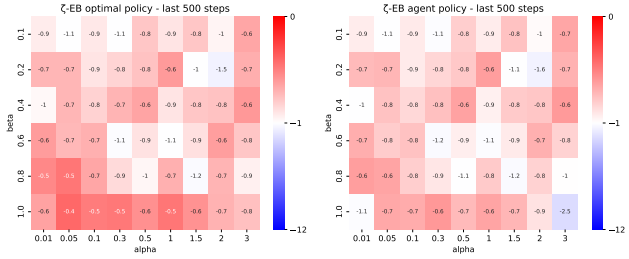

Figure 9. Parameter fitting for the ζ-EB agent, with the optimal (left) or real (right) policy value error. We did not find any parameter couple which made ζ-EB converge to the best policy in both cases.

In Figure 10, we also provide heat-maps when the prior coefficient of BEB is not already set.

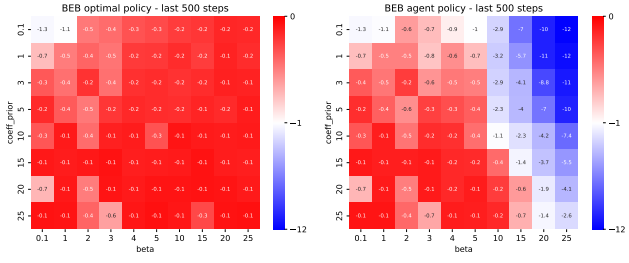

Figure 10. Parameter fitting for BEB with the optimal (left) or agent policy value error (right).

B Performance of the agents with the agent policy

The optimal policy measures the capacity of the agent to sample the environment it explores, without taking into account the actual agent policy. In our reproduction, ϵ-greedy performs much better than in Lopes and colleagues' article, although it sometimes takes random actions (see Figures 2, 3 and 5). To solve this discrepancy with the original results, we looked for a more canonical measure based on the agent policy presented subsection 2.2. In Figure 11, we present the performance of the agents on this new measure, either with the parameters used thus far, or with the best parameters on this new measure, found in Appendix A and available in Table 3. We used the same environments and number of trials as before.

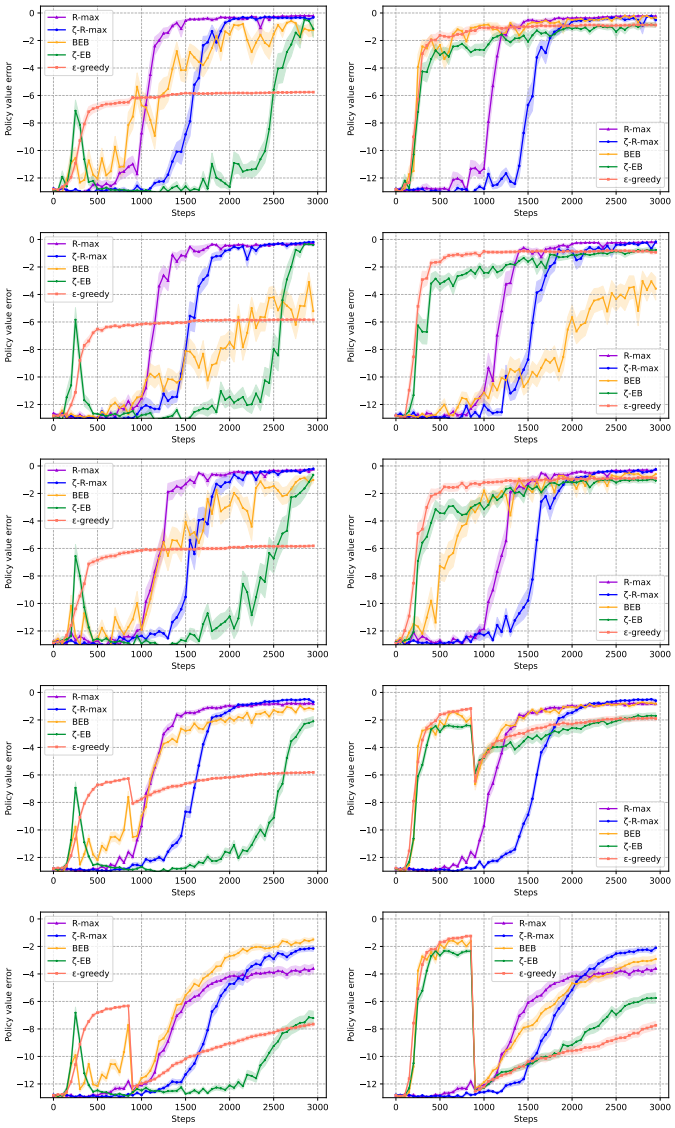

Figure 11. In both columns, the agents are evaluated based on their policy (and not on the best policy they can have with the models they learnt). The left column shows the performance of the agents with the parameters that were optimized on the optimal policy, whereas the right column shows the models with the parameters optimized on the agent policy. From top to bottom: Stationary experiment, wrong prior, uninformative prior, small task change, bigger task change.

Overall, we show that BEB, ζ-EB and ϵ-greedy policies may be sub-optimal in practice,

although they have great performance in the optimal policy value error condition. More particularly, ϵ-greedy cannot converge to the optimal policy most of the time due to the significant value of ϵ. Besides, the exploration bonus models suffer from the high impact of the exploration bonus in their decision-making. As shown in Figure 11, the agent policies often fail to converge to the very best policy in the environment.

In the right column of Figure 11, R-max and ζ-R-max have most of the qualitative properties observed in the original article. They show poor performance at start and converge to a close to optimal policy. With a more informative prior, R-max performance could further drop in the wrong prior experiment. Likewise, ϵ-greedy performance is reproduced in Figure 11-Right. In all the tasks, it cannot converge to the optimal policy because of the random actions.

However, BEB and ζ-EB results are not reproduced well. ζ-EB has the worst performance out of all the agents, which could be because of wrong hypotheses on the agent, for example considering that it is not Bayesian. We observed that the agent explores frequently but does not manage to converge to a good policy. This has been described as a bias towards exploration in Bureau and Sebag [2]. For BEB and ζ-EB, we observe a strong initial rise in performance that we do not observe for the agents of the article which could be because of the use of high initial Q-values (we chose the upper theoretical bound of $\frac{1+\beta}{1-\gamma}$). Nevertheless, such initial Q-values are necessary with the inference process that we chose, with the Q-values of a (state,action) couple not being updated before the first passage. Updating the Q-values from the start would allow for use of any initial Q-values. However, the optimistic initialization of ϵ-greedy would be impossible, although described in Lopes and colleagues' article.

To obtain more similar results to those obtained by Lopes and colleagues, one could try to change the inference process of the different agents. For example, R-max and ζ-R-max could use the known/unknown paradigm, and stop updating their models when the states are known. BEB and ζ-EB could also use the same paradigm with the introduction of a new parameter comparable to m and stop updating the model when the states are known or unknown. To guarantee that ϵ-greedy is optimistic at start, the previous inference process seems to be a good solution. However, these choices are not described in the original article of Lopes and colleagues.

Agent	Parameters changed	Previous values
ϵ-greedy	$\epsilon = 0.01$	$\epsilon = 0.3$
R-max	No change	No change
ζ-R-max	No change	No change
BEB	$\beta = 3$	$\beta = 7$
ζ-EB	$\beta = 1$	$\beta = 3$

Table 3. Change in the parameters to optimize the agent policy from the ones used for the optimal policy

C Performance of the agents on new environments

It was not clear whether the authors generated one environment and evaluated each model twenty times in this environment, or if they generated twenty environments and evaluated each model once per environment. To broaden the extent of our results, we generated 10 environments and evaluated each agent 10 times in each environment. However, generating 10 environments out of 2% of valid environments could result in using similar environments and not provide additional results to the ones we showed so

far. To generate more diverse environments, we use punishments of -1 instead of -0.1, as in Bureau and Sebag [2]. In Figure 12, we show the parameter fitting in these new environments, for the optimal policy and the agent policy described in subsection 2.2.

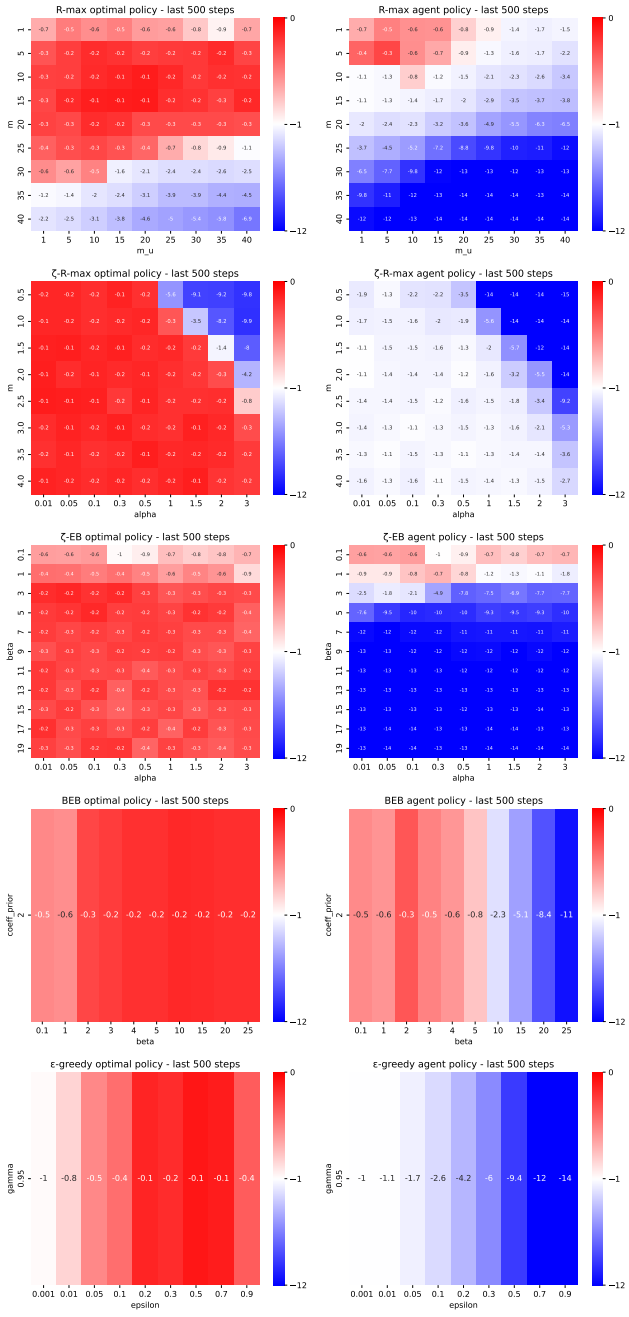

Figure 12. Parameter fitting on the new environments.

The parameter fitting for the optimal policy seems rather similar to the one of Appendix A. Thus and for simplicity, we use the values of Table 1. However, all the models seem to perform worse on the agent policy measure as compared to their performance in the previous environment (see Figure 7). More particularly, the learning progress agents have a worse performance and stability as compared to the non-learning progress agents.
In Figure 13, we show the performance of the five agents on four of the experiments studied so far: in a stationary environment, with wrong priors, with the task change of

the original article and with the stronger task change we presented in Figure 6. There are 100 different environments for non-stationary tasks (10 environments with 10 task changes on each of them) and each agent is simulated a hundred times on each task.

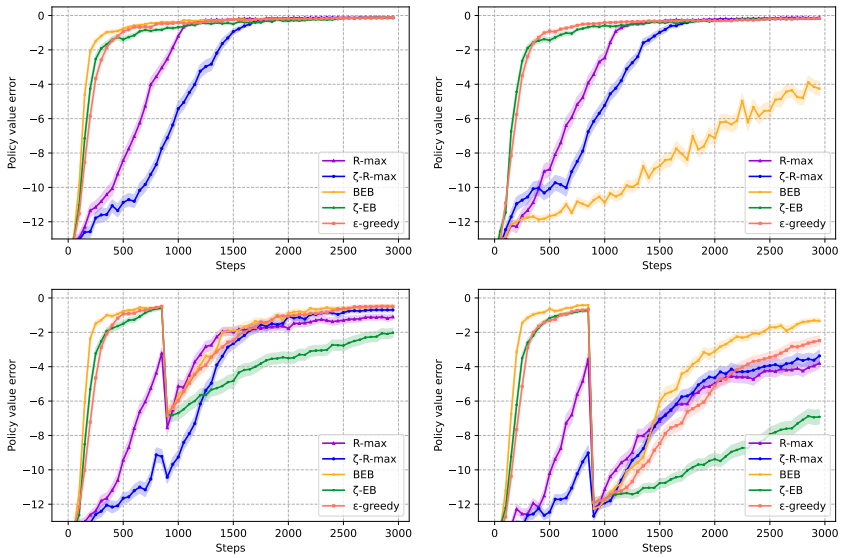

Figure 13. Reproduction of the four main experiments with the new environments and the optimal policy measure. There are 100 simulations for each agent on each task. (Upper row) Stationary environments, with informative priors (left) or wrong ones (right). (Lower row) Non-stationary environments, with a small task change (left) or a stronger task change (right).

Figure 13 illustrates the main qualitative results that we described so far: ϵ-greedy has a very good performance on the optimal policy measure, and ζ-EB performs worse than in the original article.