NOTE **Communicated by Andrew Barto and Michael Jordan**

Using Expectation-Maximization for Reinforcement Learning

Peter Dayan
Department of Brain and Cognitive Sciences, Center for Biological and Computational Learning, Massachusetts Institute of Technology, Cambridge, MA 02139 USA

Geoffrey E. Hinton
Department of Computer Science, University of Toronto, Toronto M5S 1A4, Canada

We discuss Hinton's (1989) relative payoff procedure (RPP), a static reinforcement learning algorithm whose foundation is not stochastic gradient ascent. We show circumstances under which applying the RPP is guaranteed to increase the mean return, even though it can make large changes in the values of the parameters. The proof is based on a mapping between the RPP and a form of the expectation-maximization procedure of Dempster, Laird, and Rubin (1977).

1 Introduction

Consider a stochastic learning automaton (e.g., Narendra & Thatachar 1989) whose actions y are drawn from a set $\mathcal{Y}$. This could, for instance, be the set of 2^n choices over n separate binary decisions. If the automaton maintains a probability distribution $p(y|\theta)$ over these possible actions, where θ is a set of parameters, then its task is to learn values of the parameters θ that maximize the expected payoff:

$$\rho(\theta) = \langle \mathcal{E}[r \mid y] \rangle_{p(y|\theta)} = \sum_{\mathcal{Y}} p(y \mid \theta) \mathcal{E}[r \mid y]. \tag{1.1}$$

Here, $\mathcal{E}[r|y]$ is the expected reward for performing action y.

Apart from random search, simulated annealing, and genetic algorithms, almost all the methods with which we are familiar for attempting to choose appropriate θ in such domains are local in the sense that they make small steps, usually in a direction that bears some relation to the gradient. Examples include the Keifer-Wolfowitz procedure (Wasan 1969), the A_{RP} algorithm (Barto & Anandan 1985), and the REINFORCE framework (Williams 1992). There are two reasons to make small steps. One is that there is a noisy estimation problem. Typically the automaton will emit single actions $y^1, y^2, \ldots$ according to $p(y|\theta)$, will receive single samples of the reward from the distributions $p(r|y^m)$, and will have to average the gradient over these noisy values. The other reason is that θ might have a complicated effect on which actions are chosen, or the relationship between actions and rewards

might be obscure. For instance, if $\mathcal{Y}$ is the set of 2^n choices over n separate binary actions $a_1, \ldots, a_n$, and $\theta = \{p_1, \ldots, p_n\}$ is the collection of probabilities of choosing $a_i = 1$, so

$$p(y = \{a_1..a_n\} \mid \theta) = \prod_{i=1}^{n} p_i^{a_i}(1 - p_i)^{1-a_i}, \tag{1.2}$$

then the average reward $\rho(\theta)$ depends in a complicated manner on the collection of p_i. Gradient ascent in θ would seem the only option because taking large steps might lead to decreases in the average reward.

The sampling problem is indeed present, and we will evade it by assuming a large batch size. However, we show that in circumstances such as the n binary action task, it is possible to make large, well-founded changes to the parameters without explicitly estimating the curvature of the space of expected payoffs, by a mapping onto a maximum likelihood probability density estimation problem. In effect, we maximize reward by solving a sequence of probability matching problems, where θ is chosen at each step to match as best it can a fictitious distribution determined by the average rewards experienced on the previous step. Although there can be large changes in θ from one step to the next, we are guaranteed that the average reward is monotonically increasing. The guarantee comes for exactly the same reason as in the expectation-maximization (EM) algorithm (Baum et al. 1970; Dempster et al. 1977) and, as with EM, there can be local optima. The relative payoff procedure (RPP) (Hinton 1989) is a particular reinforcement learning algorithm for the n binary action task with positive r, which makes large moves in the p_i. Our proof demonstrates that the RPP is well founded.

2 Theory

The RPP operates to improve the parameters $p_1, \ldots, p_n$ of equation 1.2 in a synchronous manner based on substantial sampling. It suggests updating the probability of choosing $a_i = 1$ to

$$p_i' = \frac{\langle a_i \mathcal{E}[r \mid y] \rangle_{p(y|\theta)}}{\langle \mathcal{E}[r \mid y] \rangle_{p(y|\theta)}}, \tag{2.1}$$

which is the ratio of the mean reward that accrues when $a_i = 1$ to the net mean reward. If all the rewards r are positive, then $0 \leq p_i' \leq 1$. This note proves that when using the RPP, the expected reinforcement increases; that is,

$$\langle \mathcal{E}[r \mid y] \rangle_{p(y|\theta')} \geq \langle \mathcal{E}[r \mid y] \rangle_{p(y|\theta)} \qquad \text{where} \qquad \theta' = \{p_1', \ldots, p_n'\}.$$

The proof rests on the following observation: Given a current value of θ, if one could arrange that

$$\alpha \mathcal{E}[r \mid y] = \frac{p(y \mid \theta')}{p(y \mid \theta)} \tag{2.2}$$

for some α, then θ' would lead to higher average returns than θ. We prove this formally below, but the intuition is that if

$$\mathcal{E}[r \mid y_1] > \mathcal{E}[r \mid y_2] \qquad \text{then} \qquad \frac{p(y_1 \mid \theta')}{p(y_2 \mid \theta')} > \frac{p(y_1 \mid \theta)}{p(y_2 \mid \theta)}$$

so θ' will put more weight on y_1 than θ does. We therefore pick θ' so that $p(y|\theta')$ matches the distribution $\alpha \mathcal{E}[r|y]p(y|\theta)$ as much as possible (using a Kullback-Leibler penalty). Note that this target distribution moves with θ. Matching just $\beta \mathcal{E}[r|y]$, something that animals can be observed to do under some circumstances (Gallistel 1990), does not result in maximizing average rewards (Sabes & Jordan 1995).

If the rewards r are stochastic, then our method (and the RPP) does not eliminate the need for repeated sampling to work out the mean return. We assume knowledge of $\mathcal{E}[r|y]$. Defining the distribution in equation 2.2 correctly requires $\mathcal{E}[r|y] > 0$. Since maximizing $\rho(\theta)$ and $\rho(\theta) + \omega$ has the same consequences for θ, we can add arbitrary constants to the rewards so that they are all positive. However, this can affect the rate of convergence.

We now show how an improvement in the expected reinforcement can be guaranteed:

$$\begin{aligned}
\log \frac{\rho(\theta')}{\rho(\theta)} &= \log \sum_{y \in \mathcal{Y}} p(y \mid \theta') \frac{\mathcal{E}[r \mid y]}{\rho(\theta)} \\
&= \log \sum_{y \in \mathcal{Y}} \left[\frac{p(y \mid \theta) \mathcal{E}[r \mid y]}{\rho(\theta)} \right] \frac{p(y \mid \theta')}{p(y \mid \theta)} \qquad (2.3) \\
&\geq \sum_{y \in \mathcal{Y}} \left[\frac{p(y \mid \theta) \mathcal{E}[r \mid y]}{\rho(\theta)} \right] \log \frac{p(y \mid \theta')}{p(y \mid \theta)}, \text{ by Jensen's inequality} \\
&= \frac{1}{\rho(\theta)} \left[Q(\theta, \theta') - Q(\theta, \theta) \right]. \qquad (2.4)
\end{aligned}$$

where

$$Q(\theta, \theta') = \sum_{y \in \mathcal{Y}} p(y \mid \theta) \mathcal{E}[r \mid y] \log p(y \mid \theta'),$$

so if $Q(\theta, \theta') \geq Q(\theta, \theta)$, then $\rho(\theta') \geq \rho(\theta)$. The normalization step in equation 2.3 creates the matching distribution from equation 2.2. Given θ, if θ' is chosen to maximize $Q(\theta, \theta')$, then we are guaranteed that $Q(\theta, \theta') \geq Q(\theta, \theta)$ and therefore that the average reward is nondecreasing.

In the RPP, the new probability p_i' for choosing $a_i = 1$ is given by

$$p_i' = \frac{\langle a_i \mathcal{E}[r \mid y] \rangle_{p(y|\theta)}}{\langle \mathcal{E}[r \mid y] \rangle_{p(y|\theta)}}, \tag{2.5}$$

so it is the fraction of the average reward that arrives when $a_i = 1$. Using equation 1.2,

$$\frac{\partial Q(\theta, \theta')}{\partial p_i'} = \frac{1}{p_i'(1 - p_i')} \times \left[\sum_{y \in \mathcal{Y}: a_i = 1} p(y \mid \theta) \mathcal{E}[r \mid y] - p_i' \sum_{y \in \mathcal{Y}} p(y \mid \theta) \mathcal{E}[r \mid y] \right].$$

So, if

$$p_i' = \frac{\sum_{y \in \mathcal{Y}: a_i = 1} p(y \mid \theta) \mathcal{E}[r \mid y]}{\sum_{y \in \mathcal{Y}} p(y \mid \theta) \mathcal{E}[r \mid y]}, \qquad \text{then} \qquad \frac{\partial Q(\theta, \theta')}{\partial p_i'} = 0,$$

and it is readily seen that $Q(\theta, \theta')$ is maximized. But this condition is just that of equation 2.5. Therefore the RPP is monotonic in the average return.

Figure 1 shows the consequence of employing the RPP. Figure 1a shows the case in which $n = 2$; the two lines and associated points show how p_1 and p_2 change on successive steps using the RPP. The terminal value $p_1 = p_2 = 0$, reached by the left-hand line, is a local optimum. Note that the RPP always changes the parameters in the direction of the gradient of the expected amount of reinforcement (this is generally true) but by a variable amount.

Figure 1b compares the RPP with (a deterministic version of) Williams's (1992) stochastic gradient ascent REINFORCE algorithm for a case with $n = 12$ and rewards $\mathcal{E}[r|y]$ drawn from an exponential distribution. The RPP and REINFORCE were started at the same point; the graph shows the difference between the maximum possible reward and the expected reward after given numbers of iterations. To make a fair comparison between the two algorithms, we chose n small enough that the exact averages in equation 2.1 and (for REINFORCE) the exact gradients,

$$p_i' = \alpha \frac{\partial}{\partial p_i} \langle \mathcal{E}[r \mid y] \rangle_{p(y|\theta)},$$

could be calculated. Figure 1b shows the (consequently smooth) course of learning for various values of the learning rate. We observe that for both algorithms, the expected return never decreases (as guaranteed for the RPP but not REINFORCE), that the course of learning is not completely smooth—with a large plateau in the middle—and that both algorithms get stuck in

local minima. This is a best case for the RPP: only for a small learning rate and consequently slow learning does REINFORCE not get stuck in a worse local minimum. In other cases, there are values of α for which REINFORCE beats the RPP. However, there are no free parameters in the RPP, and it performs well across a variety of such problems.

3 Discussion

The analogy to EM can be made quite precise. EM is a maximum likelihood method for probability density estimation for a collection $\mathcal{X}$ of observed data where underlying point $x \in \mathcal{X}$ there can be a hidden variable $y \in \mathcal{Y}$. The density has the form

$$p(x \mid \theta) = \sum_{y \in \mathcal{Y}} p(y \mid \theta) p(x \mid y, \theta),$$

and we seek to choose θ to maximize

$$\sum_{x \in \mathcal{X}} \log \left[p(x \mid \theta) \right].$$

The E phase of EM calculates the posterior responsibilities $p(y|x, \theta)$ for each $y \in \mathcal{Y}$ for each x:

$$p(y \mid x, \theta) = \frac{p(y \mid \theta) p(x \mid y, \theta)}{\sum_{z \in \mathcal{Y}} p(z \mid \theta) p(x \mid z, \theta)}.$$

In our case, there is no x, but the equivalent of this posterior distribution, which comes from equation 2.2, is

$$\mathcal{P}_y(\theta) \equiv \frac{p(y \mid \theta) \mathcal{E}[r \mid y]}{\sum_{z \in \mathcal{Y}} p(z \mid \theta) r(z)}.$$

The M phase of EM chooses parameters θ' in the light of this posterior distribution to maximize

$$\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(y \mid x, \theta) \log[p(x, y \mid \theta')].$$

In our case this is exactly equivalent to minimizing the Kullback-Leibler divergence

$$KL[\mathcal{P}_y(\theta), p(y \mid \theta')] = -\sum_{y \in \mathcal{Y}} \mathcal{P}_y(\theta) \log \left[\frac{p(y \mid \theta')}{\mathcal{P}_y(\theta)} \right].$$

Up to some terms that do not affect θ', this is $-Q(\theta, \theta')$. The Kullback-Leibler divergence between two distributions is a measure of the distance between them, and therefore minimizing it is a form of probability matching.

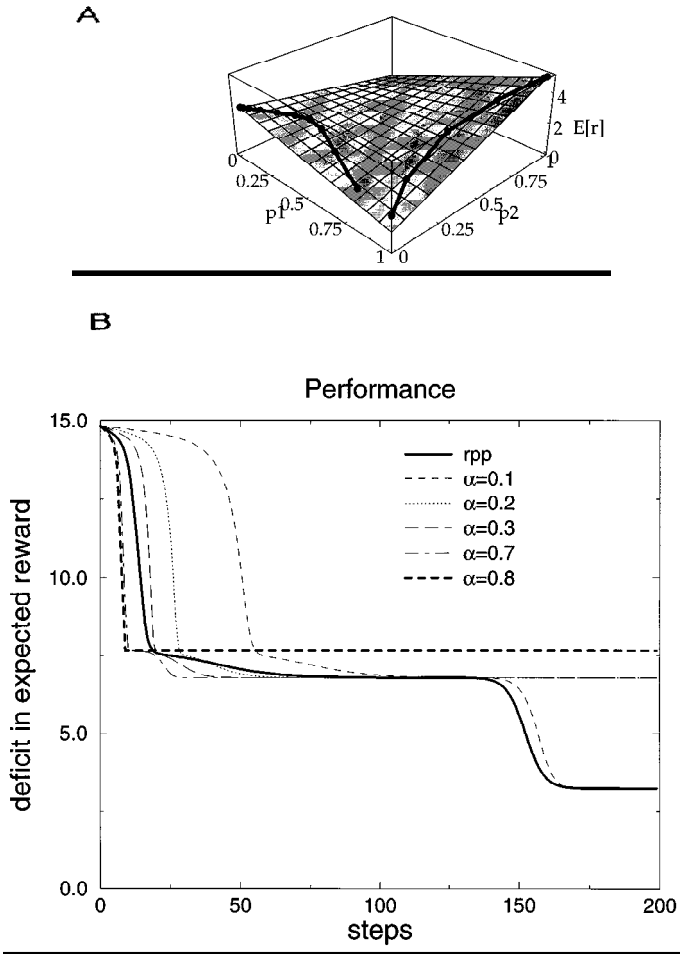

Figure 1: Performance of the RPP. (a) Adaptation of p_1 and p_2 using the RPP from two different start points on the given $\rho(p_1, p_2)$. The points are successive values; the lines are joined for graphical convenience. (b) Comparison of the RPP with Williams's (1992) REINFORCE for a particular problem with $n = 12$. See text for comment and details.

Our result is weak: the requirement for sampling from the distribution is rather restrictive, and we have not proved anything about the actual rate of convergence. The algorithm performs best (Sutton, personal communication) if the differences between the rewards are of the same order of magnitude as the rewards themselves (as a result of the normalization in equation 2.3). It uses multiplicative comparison rather than the subtractive comparison of Sutton (1984), Williams (1992), and Dayan (1990).

The link to the EM algorithm suggests that there may be reinforcement learning algorithms other than the RPP that make large changes to the values of the parameters for which similar guarantees about nondecreasing average rewards can be given. The most interesting extension would be to dynamic programming (Bellman 1957), where techniques for choosing (single-component) actions to optimize return in sequential decision tasks include two algorithms that make large changes on each step: value and policy iteration (Howard 1960). As various people have noted, the latter explicitly involves both estimation (of the value of a policy) and maximization (choosing a new policy in the light of the value of each state under the old one), although its theory is not at all described in terms of density modeling.

Acknowledgments

Support came from the Natural Sciences and Engineering Research Council and the Canadian Institute for Advanced Research (CIAR). GEH is the Nesbitt-Burns Fellow of the CIAR. We are grateful to Philip Sabes and Mike Jordan for the spur and to Mike Jordan and the referees for comments on an earlier version of this paper.

References

Barto, A. G., & Anandan, P. (1985). Pattern recognizing stochastic learning automata. *IEEE Transactions on Systems, Man and Cybernetics*, **15**, 360–374.

Baum, L. E., Petrie, E., Soules, G., & Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Stat.*, **41**, 164–171.

Bellman, R. E. (1957). *Dynamic programming*. Princeton, NJ: Princeton University Press.

Dayan, P. (1990). Reinforcement comparison. In *Proceedings of the 1990 Connectionist Models Summer School*, D. S. Touretzky, J. L. Elman, T. J. Sejnowski, & G. E. Hinton (Eds.), (pp. 45–51). San Mateo, CA: Morgan Kaufmann.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Proceedings of the Royal Statistical Society*, 1–38.

Gallistel, C. R. (1990). *The organization of learning*. Cambridge, MA: MIT Press.

Hinton, G. E. (1989). Connectionist learning procedures. *Artificial Intelligence*, **40**, 185–234.

Howard, R. A. (1960). *Dynamic programming and Markov processes.* Cambridge, MA: MIT Press.

Narendra, K. S., & Thatachar, M. A. L. (1989). *Learning automata: An introduction*. Englewood Cliffs, NJ: Prentice-Hall.

Sabes, P. N., & Jordan, M. I. (1995). Reinforcement learning by probability matching. *Advances in Neural Information Processing Systems, 8*. Cambridge, MA: MIT Press.

Sutton, R. S. (1984). *Temporal credit assignment in reinforcement learning*. Unpublished doctoral dissertation, University of Massachusetts, Amherst.

Wasan, M. T. (1969). *Stochastic approximation*. Cambridge University Press, Cambridge.

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

Received October 2, 1995; accepted May 30, 1996.