Bayesian Policy Learning with Trans-Dimensional MCMC

Anonymous Author(s)
Affiliation
Address
email

Abstract

A recently proposed formulation of the stochastic planning and control problem as one of parameter estimation for suitable artificial statistical models has led to the adoption of inference algorithms for this notoriously hard problem. At the algorithmic level, the focus has been on developing Expectation-Maximization (EM) algorithms. In this paper, we begin by making the crucial observation that the stochastic control problem can be reinterpreted as one of trans-dimensional inference. With this new understanding, we are able to propose a novel reversible jump Markov chain Monte Carlo (MCMC) algorithm that is more efficient than its EM counterparts. Moreover, it enables us to carry out full Bayesian policy search, without the need for gradients and with one single Markov chain. The new approach involves sampling directly from a distribution that is proportional to the reward and, consequently, performs better than classic simulations methods in situations where the reward is a rare event.

1 Introduction

Continuous state-space Markov Decision Processes (MDPs) are notoriously difficult to solve. Except for a few rare cases, including linear Gaussian models with quadratic cost, there is no closed-form solution and approximations are required [4]. A large number of methods have been proposed in the literature relying on value function approximation and policy search; including [3, 11, 17, 20, 22]. In this paper, we follow the policy learning approach because of its promise in continuous domains, such as robotics and motor control [7, 15, 18]. Our work is strongly motivated by a recent formulation of stochastic planning and control problem as one of inference and learning with infinite dimensional mixture models. This line of work appears to have been initiated in [5], where the authors used EM as an alternative to standard stochastic gradient algorithms to maximize an expected cost. This algorithm has been applied to operational space control with immediate rewards [19]. In [2], a planning problem under uncertainty was solved using a Viterbi algorithm. This was later extended in [25]. In these works, the number of time steps to reach the goal was fixed and the plans were not optimal in expected reward. An important step toward surmounting these limitations was taken in [24, 23]. In these works, the standard discounted reward control problem was expressed in terms of an infinite mixture of MDPs. To make the problem tractable, the authors proposed using the estimated posterior horizon time to truncate the mixture.

Here, we make the observation that, in the probabilistic approach to stochastic control, the objective function can be written as the expectation of a positive function with respect to a trans-dimensional probability distribution; that is, a probability distribution defined on an union of subspaces of different dimensions. By reinterpreting this function as a (artificial) marginal likelihood, it is easy to see that it can also be maximized using an EM-type algorithm in the spirit of [5]. However, the observation that we are dealing with a trans-dimensional distribution enables us to go beyond EM. We believe it creates many opportunities for exploiting a large body of sophisticated inference algo-

rithms in the decision-making context. In particular, it enables us to formulate a full Bayesian policy learning alternative to the EM algorithm.

In Bayesian policy learning, we set a prior distribution on the set of policy parameters and derive an artificial posterior distribution which is proportional to the prior times the expected return. In the simpler context of myopic Bayesian experimental design, a similar method was developed in [12] and applied successfully to high-dimensional problems [13]. Our method can be interpreted as a trans-dimensional extension of [12]. We sample from the resulting artificial posterior distribution using a single trans-dimensional MCMC algorithm, which only involves a simple modification of the MCMC algorithm developed to implement EM.

Although, the Bayesian policy search approach can benefit from gradient information, it does not require gradients. Moreover, since the target is proportional to the expected reward, the simulation is guided to areas of high reward automatically. This property results in an immediate reduction in variance in policy search.

2 Model formulation

We consider the following class of discrete-time Markov decision processes (MDPs):

$$\begin{aligned} X_1 &\sim \mu(\cdot) \\ X_n|\left(X_{n-1}=x, A_{n-1}=a\right) &\sim f_a\left(\cdot|\, x\right) \\ R_n|\left(X_n=x, A_n=a\right) &\sim g_a\left(\cdot|\, x\right) \\ A_n|\left(X_n=x, \theta\right) &\sim \pi_\theta\left(\cdot|\, x\right), \end{aligned}$$

where $n = 1, 2, \ldots$ is a discrete-time index, $\mu(\cdot)$ is the initial state distribution, $\{X_n\}$ is the $\mathcal{X}-$valued state process, $\{A_n\}$ is the $\mathcal{A}-$valued action process, $\{R_n\}$ is a positive real-valued reward process, f_a denotes the transition density, g_a the reward density and π_θ is a randomized policy. If we have a deterministic policy then $\pi_\theta\left(a|\, x\right) = \delta_{\varphi_\theta(x)}\left(a\right)$. In this case, the transition model $f_a\left(\cdot|\, x\right)$ assumes the parametrization $f_\theta\left(\cdot|\, x\right)$. The reward model could also be parameterized as $g_\theta\left(\cdot|\, x\right)$.

We are interested in maximizing the expected future returns with respect to the parameters of the policy θ:

$$V_\mu^\pi\left(\theta\right) = \mathbb{E}\left[\sum_{n=1}^{\infty}\gamma^n R_n\right] = \mathbb{E}_{\mu(x_1)\pi_\theta(a_1|x_1)g_{a_1}(r_1|x_1)f_{a_1}(x_2|x_1)\cdots}\left[\sum_{n=1}^{\infty}\gamma^n R_n\right],$$

where $0 < \gamma < 1$ is a discount factor. As shown in [24], it is possible re-write this objective of optimizing an infinite horizon discounted reward MDP (where the reward happens at each step) as one of optimizing an infinite mixture of finite horizon MDPs (where the reward only happens at the last time step).

In particular, we note that by introducing the trans-dimensional probability distribution on the union of spaces $\biguplus\{k\}\times\mathcal{X}^k\times\mathcal{A}^k\times\mathbb{R}^+$ given by

$$p_\theta\left(k, x_{1:k}, a_{1:k}, r_k\right) = \left(1-\gamma\right)^{-1}\gamma^k\mu\left(x_1\right)g_{a_k}\left(r_k|\, x_k\right)\prod_{n=2}^{k}f_{a_{n-1}}\left(x_n|\, x_{n-1}\right)\prod_{n=1}^{k}\pi_\theta\left(a_n|\, x_n\right), \tag{1}$$

we can easily rewrite $V_\mu^\pi\left(\theta\right)$ as an infinite mixture model of finite horizon MDPs, with the reward happening at the last horizon step; namely at k. Specifically, for a randomized policy, we obtain the

following mixture model characterization:

$$\begin{aligned} V_{\mu}^{\pi}(\theta) &= \int \gamma^1 r_1 g_{a_1}(r_1|x_1)\mu(x_1)\pi_\theta(a_1|x_1)dx_1 da_1 dr_1 \\ &\quad + \int \gamma^2 r_2 g_{a_2}(r_2|x_2)\mu(x_1)\pi_\theta(a_1|x_1)\pi_\theta(a_2|x_2)f_{a_1}(x_2|x_1)dx_{1:2}da_{1:2}dr_2 + \cdots \\ &= \sum_{k=1}^{\infty}\int \gamma^k r_k g_{a_k}(r_k|x_k)\mu(x_1)\prod_{n=2}^{k} f_a(x_n|x_{n-1})\prod_{n=1}^{k}\pi_\theta(a_n|x_n)dx_{1:k}da_{1:k}dr_k \\ &= \sum_{k=1}^{\infty}\int (1-\gamma) r_k p_\theta\left(k, x_{1:k}, a_{1:k}, r_k\right) dx_{1:k}da_{1:k}dr_k = (1-\gamma)\,\mathbb{E}_{p_\theta}\left[r_k\right]. \end{aligned} \tag{2}$$

Similarly, for a deterministic policy, the representation (2) also holds for the trans-dimensional probability distribution defined on $\biguplus \{k\} \times \mathcal{X}^k \times \mathbb{R}^+$ given by

$$p_\theta\left(k, x_{1:k}, r_k\right) = (1-\gamma)^{-1}\gamma^k \mu\left(x_1\right) g_\theta\left(r_k|\, x_k\right)\prod_{n=2}^{k} f_\theta\left(x_n|\, x_{n-1}\right). \tag{3}$$

The representation (2) was used in [6] to compute the value function through MCMC for a fixed θ. In [24], this representation is exploited to maximize $V_{\mu}^{\pi}(\theta)$ using the EM algorithm which, applied to this problem, proceeds as follows at iteration i

$$\theta_i = \underset{\theta\in\Theta}{\arg\max}\, Q\left(\theta_{i-1}, \theta\right)$$

where

$$Q\left(\theta_{i-1}, \theta\right) = \mathbb{E}_{\widetilde{p}_{\theta_{i-1}}}\left[\log\left(R_K . p_\theta\left(K, X_{1:K}, A_{1:K}, R_K\right)\right)\right],$$

$$\widetilde{p}_\theta\left(k, x_{1:k}, a_{1:k-1}, r_k\right) = \frac{r_k p_\theta\left(k, x_{1:k}, a_{1:k}, r_k\right)}{\mathbb{E}_{p_\theta}\left[R_K\right]}.$$

Unlike [24], we are interested in problems with potentially nonlinear and non-Gaussian properties. In these situations, the Q function cannot be calculated exactly and we need to simulate from $\widetilde{p}_\theta\left(k, x_{1:k}, a_{1:k-1}, r_k\right)$ in order to obtain Monte Carlo estimates of the Q function. *The good news is that $\widetilde{p}_\theta\left(k, x_{1:k}, a_{1:k-1}, r_k\right)$ is proportional to the reward. Consequently, the samples will be drawn where there is high utility.* This is a wonderful feature in situations where the reward is a rare event, which is often the case in high dimensional control settings.

At this stage, we could proceed as in [24] and derive forward-backward algorithms for the E step. We have in fact done this using the smoothing algorithms proposed in [10]. However, we will focus the discussion on a different approach based on trans-dimensional simulation. As shown in the experiments, the latter does considerably better.

Finally, we remark that for a deterministic policy, we can introduce the trans-dimensional distribution:

$$\widetilde{p}_\theta\left(k, x_{1:k}, r_k\right) = \frac{r_k p_\theta\left(k, x_{1:k}, r_k\right)}{\mathbb{E}_{p_\theta}\left[R_K\right]}.$$

In addition, and for ease of presentation only, we focus the discussion on deterministic policies and reward functions $g_\theta\left(r_n|\, x_n\right) = \delta_{r(x_n)}\left(r_n\right)$; the extension of our algorithms to the randomized case is straightforward.

3 Bayesian policy exploration

EM algorithms result in point estimates of θ. Moreover, they are not guaranteed to find the global optimum of the expected return. They are particularly sensitive to initialization and might get trapped in a severe local maximum. Moreover, in the general state space setting that we are considering, the particle smoothers in the E step can be very expensive computationally.

To address these concerns, we propose an alternative full Bayesian approach. In the simpler context of experimental design, this approach was successfully developed in [12], [13]. The idea consists

of introducing a vague prior distribution $p(\theta)$ on the parameters of the policy θ. We then define the new artificial probability distribution defined on $\Theta \times \biguplus \{k\} \times \mathcal{X}^k \times \mathbb{R}^+ \mathcal{X}$ by

$$\overline{p}(\theta, k, x_{1:k}) \propto r(x_k)\, p_\theta(k, x_{1:k})\, p(\theta).$$

By construction, this target distribution admits the following marginal in θ

$$\overline{p}(\theta) \propto V_\mu^\pi(\theta)\, p(\theta)$$

and we can select an improper prior distribution $p(\theta) \propto 1$ if $\int_\Theta V_\mu^\pi(\theta)\, d\theta < \infty$.

If we could sample from $\overline{p}(\theta)$, then the generated samples $\left\{\theta^{(i)}\right\}$ would concentrate themselves in regions where $V_\mu^\pi(\theta)$ is large. We cannot sample from $\overline{p}(\theta)$ directly but we can developed a trans-dimensional MCMC algorithm which will generate asymptotically samples from $\overline{p}(\theta, k, x_{1:k})$, hence samples from $\overline{p}(\theta)$.

Our algorithm proceeds as follows. Assume the current state of the Markov chain targeting $\overline{p}(\theta, k, x_{1:k})$ is $(\theta, k, x_{1:k})$. We propose first to update the components $(k, x_{1:k})$ conditional upon θ using a combination of birth, death and update moves using the reversible jump MCMC algorithm [8, 9, 21]. Then we propose to update θ conditional upon the current value of $(k, x_{1:k})$. This can be achieved using a simple Metropolis-Hastings algorithm or a more sophisticated dynamic Monte Carlo schemes. For example, if gradient information is available, one could adopt Langevin diffusions and the hybrid Monte Carlo algorithm [1, 14]. The overall algorithm is depicted in Figure 1. The details of the reversible jump algorithm are presented in the following section.

1. Initialization: set $(k^{(0)}, x_{1:k^{(0)}}^{(0)}, \theta^{(0)})$.
2. For $i = 0$ to $N - 1$
   - Sample $u \sim \mathcal{U}_{[0,1]}$.
   - If $(u \leq b_k)$
     - then carry out a "birth" move: Increase the horizon length of the MDP, say $k^{(i)} = k^{(i-1)} + 1$ and insert a new state.
     - else if $(u \leq b_k + d_k)$ then carry out a "death" move: decrease the horizon length of the MDP, say $k^{(i)} = k^{(i-1)} - 1$ and an existing state.
     - else generate samples $x_{1:k^{(i-1)}}^{(i)}$ of the MDP states.

     End If.
   - Sample the policy parameters $\theta^{(i)}$ conditional on the samples $(x_{1:k^{(i)}}^{(i)}, k^{(i)})$.

Figure 1: Generic reversible jump MCMC for Bayesian policy learning.

We note that the samples of the states and horizon generated by this Markov process will also be distributed according to the trans-dimensional distribution $\widetilde{p}_\theta(k, x_{1:k})$; this is indeed the output of the reversible jump algorithm for a given θ. Hence, they can be easily adopted to generate a Monte Carlo estimate of $Q(\theta_{i-1}, \theta)$. This allows us to side-step the need for expensive smoothing algorithms in the E step. The trans-dimensional simulation approach has the advantage that the samples will concentrate automatically in regions of high reward. Moreover, unlike in the smoothing counterparts [24], it is no longer necessary to truncate the time domain.

4 Trans-Dimensional Markov chain Monte Carlo

We present a simple reversible jump method composed of two reversible moves (birth and death) and several update moves. Assume the current state of the Markov chain targeting $\widetilde{p}_\theta(k, x_{1:k})$

is $(k, x_{1:k})$. With probability b_k, we propose a birth move; that is we sample a location uniformly in the interval $\{1, ..., k+1\}$, i.e. $J \sim \mathcal{U}\{1, ..., k+1\}$, and propose the candidate $(k+1, x_{1:j-1}, x^*, x_{j+1:k})$ where $X^* \sim q_\theta(\cdot | x_{j-1:j+1})$. This candidate is accepted with probability $A_{birth} = \min\{1, \alpha_{birth}\}$ where we have for $j \in \{2, ..., k-1\}$

$$\begin{aligned}\alpha_{birth} &= \frac{\widetilde{p}_\theta(k+1, x_{1:j-1}, x^*, x_{j+1:k})\, d_{k+1}}{\widetilde{p}_\theta(k, x_{1:k})\, b_k q_\theta(x^* | x_{j-1:j+1})} \\ &= \frac{\gamma f_\theta(x^* | x_{j-1})\, f_\theta(x_{j+1} | x^*)\, d_{k+1}}{f_\theta(x_j | x_{j-1})\, b_k q_\theta(x^* | x_{j-1:j+1})},\end{aligned}$$

for $j = 1$

$$\alpha_{birth} = \frac{\gamma \mu(x^*)\, f_\theta(x_1 | x^*)\, d_{k+1}}{\mu(x_1)\, b_k q_\theta(x^* | x_{1:2})}$$

and $j = k+1$

$$\alpha_{birth} = \frac{\gamma r(x^*)\, f_\theta(x^* | x_k)\, d_{k+1}}{r(x_k)\, b_k q_\theta(x^* | x_{k-1:k})}.$$

With probability d_k, we propose a death move; that is $J \sim \mathcal{U}\{1, ..., k\}$ and we propose the candidate $(k-1, x_{1:j-1}, x_{j+1:k})$ which is accepted with probability $A_{death} = \min\{1, \alpha_{death}\}$ where for $j \in \{2, ..., k-1\}$

$$\begin{aligned}\alpha_{death} &= \frac{\widetilde{p}_\theta(k-1, x_{1:j-1}, x_{j+1:k})\, b_{k+1} q_\theta(x_j | x_{j-1:j+1})}{\widetilde{p}_\theta(k, x_{1:k})\, d_k} \\ &= \frac{f_\theta(x_{j+1} | x_{j-1})\, b_{k+1} q_\theta(x_j | x_{j-1:j+1})}{\gamma f_\theta(x_{j+1} | x_j)\, f_\theta(x_j | x_{j-1})\, d_k},\end{aligned}$$

for $j = 1$

$$\alpha_{death} = \frac{\mu(x_2)\, q_\theta(x_1 | x_{1:2})\, b_{k+1}}{\gamma \mu(x_1)\, f_\theta(x_2 | x_1)\, d_k}$$

and for $j = k$

$$\alpha_{death} = \frac{r(x_{k-1})\, q_\theta(x_k | x_{k-2:k-1})\, b_{k+1}}{\gamma r(x_k)\, f_\theta(x_k | x_{k-1})\, d_k}.$$

Finally with probability $u_k = 1 - b_k - d_k$, we propose a standard (fixed dimensional) move where we update all or a subset of the components $x_{1:k}$ using say Metropolis-Hastings or Gibbs moves. There are many design possibilities for these moves. In general, one should block some of the variables so as to improve the mixing time of the Markov chain. If one adopts a simple one-at-a time Metropolis-Hastings scheme with proposals $q_\theta(x^* | x_{j-1:j+1})$ to update the j-th term, then the candidate is accepted with probability $A_{upd} = \min\{1, \alpha_{upd}\}$ where for $j \in \{2, ..., k-1\}$

$$\begin{aligned}\alpha_{upd} &= \frac{\widetilde{p}_\theta(k, x_{1:j-1}, x^*, x_{j+1:k})\, q_\theta(x_j | x_{j-1}, x^*, x_{j+1})}{\widetilde{p}_\theta(k, x_{1:k})\, q_\theta(x^* | x_{j-1:j+1})} \\ &= \frac{f_\theta(x^* | x_{j-1})\, f_\theta(x_{j+1} | x^*)\, q_\theta(x_j | x_{j-1}, x^*, x_{j+1})}{f_\theta(x_j | x_{j-1})\, f_\theta(x_{j+1} | x_j)\, q_\theta(x^* | x_{j-1:j+1})},\end{aligned}$$

for $j = 1$

$$\alpha_{upd} = \frac{\mu(x^*)\, f_\theta(x_2 | x^*)\, q_\theta(x_1 | x^*, x_2)}{\mu(x_1)\, f_\theta(x_2 | x_1)\, q_\theta(x^* | x_{1:2})}$$

and for $j = k$

$$\alpha_{upd} = \frac{r(x^*)\, f_\theta(x^* | x_{k-1})\, q_\theta(x_k | x^*, x_{k-1})}{r(x_k)\, f_\theta(x_k | x_{k-1})\, q_\theta(x^* | x_{k-1:k})}.$$

Under weak assumptions on the model, the Markov chain $\left\{K^{(i)}, X_{1:K}^{(i)}\right\}$ generated by this transition kernel will be irreducible and aperiodic and hence will generate asymptotically samples from the target distribution $\widetilde{p}_\theta(k, x_{1:k})$.

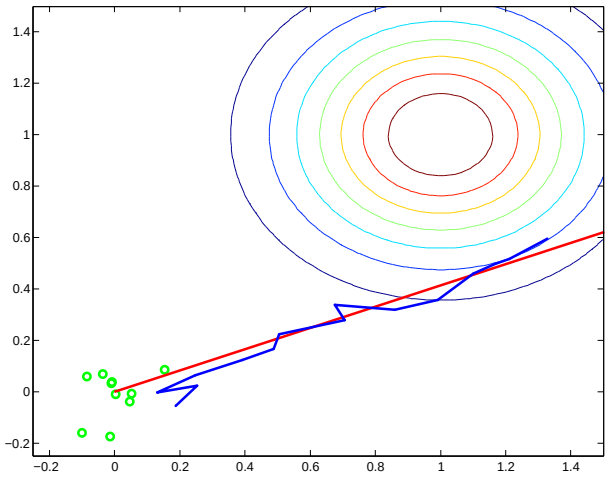

Figure 2: **Example state-space.** This figure shows an illustration of the 2d state-space described in section 5. Ten sample points are shown distributed according to μ, the initial distribution, and the contour plot corresponds to the reward function r. The red line denotes the policy parameterized by an angle θ, while a path sampled according to this policy is shown in blue.

We emphasize that the structure of the distributions $\widetilde{p}_\theta\left(x_{1:k}\middle|\, k\right)$ will not in many applications vary significantly with k and we will often have $\widetilde{p}_\theta\left(x_{1:k}\middle|\, k\right) \approx \widetilde{p}_\theta\left(x_{1:k}\middle|\, k+1\right)$. Hence the probability of having the reversible moves accepted will be reasonable. Standard Bayesian applications of reversible jump MCMC usually do not enjoy this property and it makes it more difficult to design fast mixing algorithms. In this respect, our problem is easier.

5 Experiment

We consider state and action spaces $\mathcal{X} = \mathcal{A} = \mathbb{R}2$ such that each state $x \in \mathcal{X}$ is a 2d position and each action $a \in \mathcal{A}$ is a vector corresponding to a change in position. A new state at time n is given by $X_n = X_{n-1} + A_{n-1} + \nu_{n-1}$ where ν_{n-1} denotes zero-mean Gaussian noise. Finally we will let μ be a normal distribution about the origin, and consider a reward (as in [24]) given by an unnormalized Gaussian about some point m, i.e. $r(x) = \exp(-\frac{1}{2}(x-m)^T\Sigma^{-1}(x-m))$. An illustration of this space can be seen in Figure 2 where $m = (1,1)$.

It is worth noting that the problem described so far is not as simple as it might appear on the surface. Using a reward covariance of $\Sigma = .01I$ the reward is relatively rare in the state-space and especially when starting from an initial parameter of $\theta = \pi/2$. There is very little gradient information to direct the search.

For this experiment, we chose a simple stochastic policy parameterized by $\theta \in [0, \pi/2]$. Under this policy, an action A_n is normally distributed about $nw(\cos\theta, \sin\theta) - x_n$ for some (small) constant step-length w. Intuitively, this ensures that an agent following this policy will advance on a path along the angle θ. For a state-space with initial distribution and reward function as shown in Figure 2, the optimal policy corresponds to $\theta = \pi/4$.

The plots in Figure 3 compare the performance of the reversible jump MCMC algorithms when applied to both Bayesian policy search and optimization with the EM algorithm. The comparison also includes and EM algorithm with a two-filter particle smoother, which is not entirely a straightforward extension of the algorithms presented in [24].

The first thing of note is the poor performance of the EM approaches with particle smoothing. This comes as no surprise considering the $\mathrm{O}(N^2k_{\max}^2)$ time-complexity involved in computing the importance weights, where N is the number of particles and $k_{\max}$ is the length of the truncated MDP. While there do exist methods [10] for reducing this complexity to $\mathrm{O}(N \log N k_{\max}^2)$, the discrepancy between this and the reversible jump MCMC method suggests that the MCMC approach may be more adapted to this class of problems (and for this reason we have omitted discussion of the particle smoothing method).

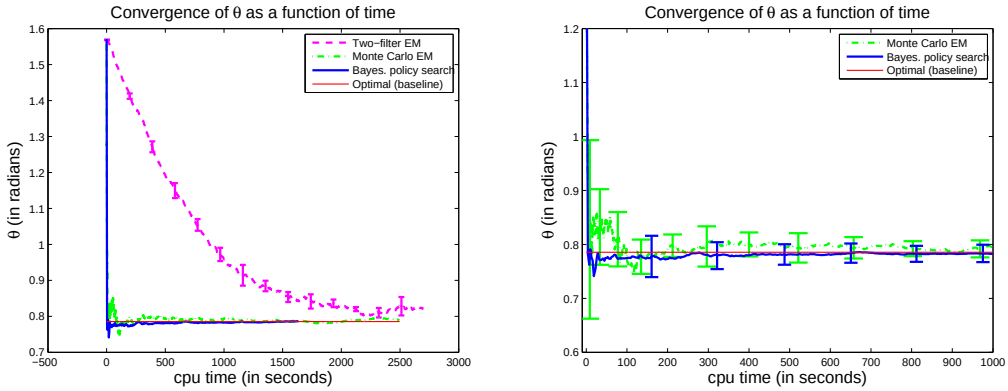

Figure 3: The left plot shows estimates for the policy parameter θ as a function of the cpu time. This data is shown for the three discussed Monte Carlo algorithms as applied to a synthetic example and has been averaged over five runs; error bars are shown for the SMC-based EM algorithm. The bottom figure shows a "zoomed" version of this plot in order to see the reversible-jump EM algorithm and the fully Bayesian algorithm in more detail. In both plots the red line denotes the known optimal policy parameter of $\pi/4$.

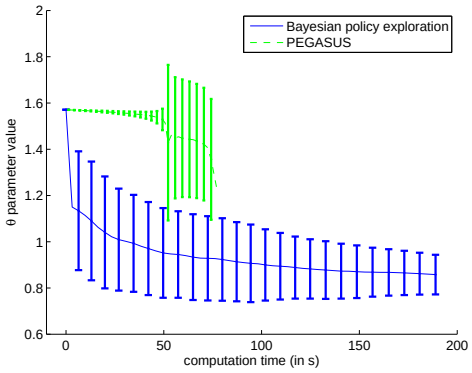

Figure 4: Comparison between Bayesian policy search with the reversible jump MCMC algorithm and policy gradient search with PEGASUS. Because of the extremely low gradient information, and the needed increase in step-size for the gradient updates, the variance across multiple runs increases.

The reversible jump Monte Carlo EM algorithm and the fully Bayesian approach performed comparably on this synthetic example. However, the Bayesian approach exhibited, in general, less in-run variance and less variance between runs. The EM algorithm was found to be more sensitive, and we were forced to increase the number of samples N used in the E-step as the algorithm progressed. This required controlling the learning rate with a smoothing parameter. For higher dimensional and/or larger models it is not inconceivable that this could have an adverse effect on the algorithm's performance.

Finally, we also compared the proposed Bayesian policy exploration method to the PEGASUS [16] approach using policy gradients. As shown in Figure 4, the Bayesian strategy is more efficient in this rare event setting. As the state space increases, we expect this difference to become even more pronounced.

6 Discussion

We believe that formulating stochastic control as a trans-dimensional inference problem is fruitful. It has led to the development of the first, to the best of our knowledge, trans-dimensional MCMC algorithm for policy search in general non-linear non-Gaussian control problems. Our results, on an illustrative example, showed that this trans-dimensional simulator is more effective than the sim-

ulators based on EM with smoothing in the E step. It is also more effective than classic policy gradient methods, where there is little gradient information. In the near future, we plan to apply our algorithms to several control and planning tasks of interest.

References

[1] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50:5–43, 2003.

[2] H. Attias. Planning by probabilistic inference. In *Uncertainty in Artificial Intelligence*, 2003.

[3] J. Baxter and P. L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

[4] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.

[5] P. Dayan and G. E. Hinton. Using EM for reinforcement learning. *Neural Computation*, 9:271–278, 1997.

[6] A. Doucet and V. B. Tadic. On solving integral equations using Markov chain Monte Carlo methods. Technical Report CUED-F-INFENG 444, Cambridge University Engineering Department, 2004.

[7] T. Field. Policy-gradient learning for motor control. Master's thesis, Victoria University of Wellington, Wellington, 2005.

[8] P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[9] P. J. Green. Trans-dimensional Markov chain Monte Carlo. In *Highly Structured Stochastic Systems*, 2003.

[10] M. Klaas, M. Briers, N. de Freitas, A. Doucet, and S. Maskell. Fast particle smoothing: If i had a million particles. In *International Conference on Machine Learning*, 2006.

[11] G. Lawrence, N. Cowan, and S. Russell. Efficient gradient estimation for motor control learning. In *Uncertainty in Artificial Intelligence*, pages 354–36, 2003.

[12] P. Müller. Simulation based optimal design. *Bayesian Statistics*, 6, 1999.

[13] P. Müller, B. Sansó, and M. De Iorio. Optimal Bayesian design by inhomogeneous Markov chain simulation. *J. American Stat. Assoc.*, 99:788–798, 2004.

[14] R. M. Neal. *Bayesian Learning for Neural Networks*. Lecture Notes in Statistics No. 118, Springer-Verlag, New York, 1996.

[15] A. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang. Inverted autonomous helicopter flight via reinforcement learning. In *International Symposium on Experimental Robotics*, 2004.

[16] A. Y. Ng. *Shaping and Policy Search in Reinforcement Learning*. PhD thesis, University of California, Berkeley, 2003.

[17] A. Y. Ng and M. I. Jordan. PEGASUS: A policy search method for large MDPs and POMDPs. In *Uncertainty in Artificial Intelligence*, 2000.

[18] J. Peters and S. Schaal. Policy gradient methods for robotics. In *IEEE International Conference on Intelligent Robotics Systems*, 2006.

[19] J. Peters and S. Schaal. Reinforcement learning for operational space control. In *International Conference on Robotics and Automation*, 2007.

[20] M. Porta, N. Vlassis, M. T. J. Spaan, and P. Poupart. Point-based value iteration for continuous POMDPs. *Journal of Machine Learning Research*, 7:2329–2367, 2006.

[21] S. Richardson and P. J. Green. On Bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society B*, 59(4):731–792, 1997.

[22] S. Thrun. Monte Carlo POMDPs. In S. Solla, T. Leen, and K.-R. Müller, editors, *Neural Information Processing Systems*, pages 1064–1070. MIT Press, 2000.

[23] M. Toussaint, S. Harmeling, and A. Storkey. Probabilistic inference for solving (PO)MDPs. Technical Report EDI-INF-RR-0934, University of Edinburgh, School of Informatics, 2006.

[24] M. Toussaint and A. Storkey. Probabilistic inference for solving discrete and continuous state Markov decision processes. In *International Conference on Machine Learning*, 2006.

[25] D. Verma and R. P. N. Rao. Planning and acting in uncertain environments using probabilistic inference. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2006.