Online solution of the average cost Kullback-Leibler optimization problem

Joris Bierkens
Radboud University Nijmegen
j.bierkens@science.ru.nl

Bert Kappen
Radboud University Nijmegen
b.kappen@science.ru.nl

Abstract

We introduce a stochastic approximation method for the solution of a Kullback-Leibler optimization problem, which is a generalization of Z-learning introduced by [Todorov, 2007]. A KL-optimization problem is Markov decision process with a finite state space and continuous control space. Because the control cost has a special form involving the Kullback-Leibler divergence, it can be shown that the problem may be solved essentially by finding the largest eigenvector and eigenvalue of a non-negative matrix. The stochastic algorithm presented in this paper may be used to solve this problem. It allows for a sound theoretical analysis and can be shown to be comparable to the power method in terms of convergence speed. It may be used as the basis of a reinforcement learning style algorithm for Markov decision problems.

1 Introduction

A few years ago, Todorov [Todorov, 2007] introduced a novel method in stochastic optimal control, to which we will refer as Kullback-Leibler optimization, where the optimal control may be determined essentially by computing the dominant eigenvector (i.e. the eigenvector corresponding to the largest eigenvalue, also called the Perron-Frobenius eigenvector) of some non-negative matrix. The computation of the dominant eigenvector is relevant in other areas of mathematical analysis and optimization, see e.g. [Meyer, 2000], Chapter 8.

It is the goal of this paper to introduce a novel method to compute the dominant eigenvector. The well-known power method (see e.g. [Horn, Johnson, 1990]) provides a convenient and powerful way to compute this dominant eigenvector. For our purposes it has the disadvantage that we need to have access to the entire non-negative matrix. In problems where the state space is large, this may be problematic. Also a requirement for the power method is the knowledge of all the entries of the matrix at initialization time of the algorithm, whereas we wish to think of these entries as being obtained by observing a stochastic process.

To avoid the mentioned problems we introduce a stochastic approximation algorithm for the computation of the Perron-Frobenius eigenvector and corresponding eigenvalue. In Section 2 we discuss the motivation for this algorithm, that we briefly sketched above. In Section 3, which may be read independently of the preceding section, the stochastic approximation algorithm is described and some theoretical results on convergence of the algorithm are provided. In Section 4 we show for a particular example its convergence properties and compare it to the power method in terms of numerical performance. Finally, in Section 5 we discuss some interesting questions that remain open, in particular the possible extension to online control and reinforcement learning.

2 Motivation: Kullback-Leibler optimization

In this section we introduce the problem setting, which we will refer to as *Kullback-Leibler optimization* or *KL-optimization*. For a more detailed introduction, see [Todorov, 2007].

Let $t = 0, 1, 2, \ldots$ denote time. Consider a Markov chain $(x_t)_{t=0}^{\infty}$ on a finite state space $S = \{1, \ldots, n\}$ with transition probabilities $q = (q_{ij})$ called the *uncontrolled dynamics*, where $q_{ij} = \mathbb{P}^q(x_{t+1} = j | x_t = i)$. For simplicity we consider here the stationary case. Suppose for every jump of the Markov chain, at every state i in S a cost $c(i)$ is incurred. We wish to change the transition probabilities in such a way as to minimize for example the total incurred cost (assuming there exist absorbing states where no further costs are incurred) or the average cost per stage. For deviating from the transition probabilities control costs are incurred equal to the *Kullback-Leibler divergence* or *relative entropy* of the probability distribution $\mathbb{P}^p$ on the space of sample paths under the changed transition probabilities p with respect to the probability distribution $\mathbb{P}^q$ corresponding to the uncontrolled dynamics q. This is equivalent to saying that at time t a control cost of

$$\mathrm{KL}(\mathbb{P}^p(x_{t+1}|x_t)||\mathbb{P}^q(x_{t+1}|x_t)) = \sum_{j=1}^{n} \mathbb{P}^p(x_{t+1} = j|x_t) \ln\left(\frac{\mathbb{P}^p(x_{t+1} = j|x_t)}{\mathbb{P}^q(x_{t+1} = j|x_t)}\right)$$

is incurred, in addition to the cost per stage $c(x_t)$.

If we write $p_{ij} = \exp(u_j(i))q_{ij}$, in the case of an infinite horizon problem, the corresponding Bellman equation for the value function Φ is

$$\Phi(i) = \min_{(u_1,\ldots,u_n)\in\mathbb{R}^n} \left\{ c(i) + \sum_{j=1}^{n} q_{ij} \exp(u_j)(u_j + \Phi(j)) \right\}.$$

A straightforward computation, as in [Todorov, 2007], yields that the optimal $u_j(i)$ and value function Φ are given by

$$u_j^*(i) = \ln(z_j^*/z_i^*) - c(i), \qquad \Phi(i) = -\ln(z_i^*), \tag{1}$$

with $z^* \in \mathbb{R}^n$ given implicitly by

$$z_i^* = \exp(-c(i)) \sum_{j=1}^{n} q_{ij} z_j^*,$$

which may be written as $z^* = Hz^*$, with $H_{ij} = \exp(-c(i)) \sum_{j=1}^{n} q_{ij}$. This z^* should be normalized so that the value function agrees with the value 0 in the absorbing states. Similarly, in the case where there are no absorbing states, the average cost per stage problem may be solved by finding the solution z of $\lambda^* z^* = Hz^*$, where λ^* is the largest eigenvalue of H.

According to Perron-Frobenius theory of non-negative matrices ([Horn, Johnson, 1990], Theorem 8.4.4), if the uncontrolled Markov chain q is irreducible, there exists a simple eigenvalue λ^* equal to the spectral radius $\rho(H)$, with an eigenvector z^* that has only positive entries. Since λ^* is a simple eigenvalue, $\mathbf{z}^*$ is unique op to multiplication by a positive scalar. These λ^* and $\mathbf{z}^*$, if normalized, are called the called the *Perron-Frobenius* eigenvalue and eigenvector, respectively.

3 Stochastic approximation algorithm and theoretical results

The goal of our method is to find the Perron Frobenius eigenvector and eigenvalue, as explained in the previous section. A straightforward way to find λ^* and z^* is using the *power method*, i.e. by performing the iteration

$$z^{k+1} = \frac{Hz^k}{||Hz^k||}.$$

This assumes that we have access to the full matrix H. Our goal is to relax this assumption, and to find z by iteratively stepping through states of the Markov chain using the uncontrolled dynamics q, and using only the observations of the cost $c(i)$ when we reach state i.

In this section, we let $Q \in \mathbb{R}^n$ be an irreducible, stochastic matrix, i.e. $q_{ij} \geq 0$ for all i, j, $\sum_{j=1}^n q_{ij} = 1$ for all i and for all i, j there exists a k such that $(q^k)_{ij} > 0$. Let R be a diagonal matrix, with only positive entries on the diagonal. In the setting of the previous section, we would have $r_{ii} = \exp(-c(i))$. Let (γ_m) be a sequence of positive *gain factors*, satisfying $\sum_{m=1}^{\infty} \gamma_m = \infty$ and $\sum_{m=1}^{\infty} \gamma_m^2 < \infty$ for theoretical analysis. In practice, often a constant gain factor is used. Throughout, we will write $x(i)$ for the i-th component of a vector $\mathbf{x}$.

3.1 Z-learning

[Todorov, 2007] introduces a stochastic approximation algorithm for our intended task, which we refer to as Z-learning. The stochastic update rule of this algorithm for the vector z is given by

$$z(i) = (1 - \gamma_{m+1})z(i) + \gamma_{m+1} \exp(-c(i)z(j),$$

whenever a jump occurs from state i to state j. This algorithm works in case the spectral radius of H is equal to 1. In the KL-control formulation this corresponds to the case where absorbing states exists. If $\rho(H) \neq 1$, the vector z computed by this update rule actually decreases to zero or increases to infinity. Even if z is normalized, it can be shown not to converge to the right solution if the stationary distribution of the Markov chain is not flat.

Below we propose a stochastic algorithm that generalizes Z-learning to the case where $\rho(H) \neq 1$, thus extending Z-learning to average cost per stage KL optimization.

3.2 Algorithm

Consider the following algorithm, similar in spirit to the Robbins-Monro algorithm ([Benveniste et al., 1990], [Bertsekas, Tsitsikils, 1996], [Kushner, Yin, 2003]) and the Z-learning algorithm of [Todorov, 2007]. It computes the Perron Frobenius eigenvector and eigenvalue synchronously. A parameter $\alpha > 0$ balances the updates to the eigenvector $\mathbf{z}_m$ and the eigenvalue λ_m and is arbitrary but influences the convergence speed.

1. Initialize by setting
$$\mathbf{z}_0 \leftarrow \mathbf{1}, \quad \lambda_0 \leftarrow 1, \quad x_0 \leftarrow 1.$$
2. Repeat, for $m = 0, 1, 2, \ldots$, the updates
$$\begin{aligned} x_{m+1} &\leftarrow \text{random jump from state } x_m \text{ according to distribution } \mathbb{P}^q(x_{m+1}|x_m), \\ \zeta_{m+1} &\leftarrow \exp(-c(x_m))z_m(x_{m+1})/\lambda_m - z_m(x_m), \\ \mathbf{z}_{m+1} &\leftarrow \mathbf{z}_m, \\ z_{m+1}(x_m) &\leftarrow z_m(x_m) + \gamma_{m+1}\zeta_{m+1}, \\ \lambda_{m+1} &\leftarrow \lambda_m + \alpha\gamma_{m+1} z_m(x_m)\zeta_{m+1} \end{aligned}$$
until convergence.

We see that for every step of the algorithm, to update the current value of $\mathbf{z}$ and λ we just need to know cost incurred, $c(x_m)$ and the information of the vector $\mathbf{z}$ on the current and next position, $z(x_m)$ and $z(x_{m+1})$, respectively. In this sense the algorithm only uses local information, as opposed to the power method.

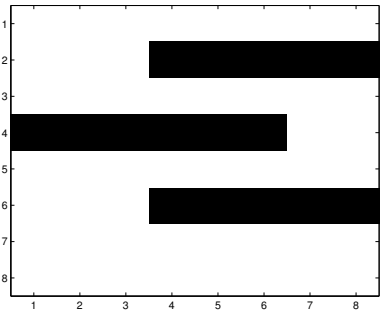
Note that if we would keep λ_{m+1} fixed at the value 1, the algorithm is equal to Z-learning described above.

3.3 Theoretical analysis of the algorithm

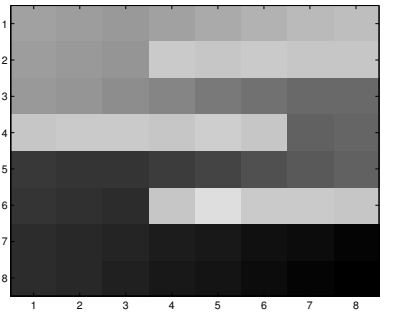
The space here is too limited to present the theoretical analysis in detail. A publication is being prepared that will contain the detailed analysis [Bierkens, Kappen, 2011]. Here is a short overview.

Using the ODE method for analysing stochastic algorithms (see [Benveniste et al., 1990], [Bertsekas, Tsitsikils, 1996], [Kushner, Yin, 2003]), it can be shown that as $m \to \infty$, for a rescaled time parameter t, the algorithm will behave as the deterministic ordinary differential equation (ODE)

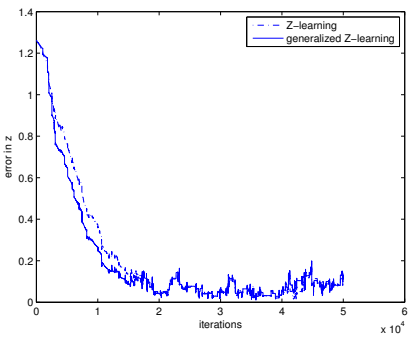
$$\begin{cases} \frac{d}{dt}\mathbf{z}(t) = f(\mathbf{z}(t), \lambda(t)), \\ \frac{d}{dt}\lambda(t) = g(\mathbf{z}(t), \lambda(t)), \end{cases} \tag{2}$$



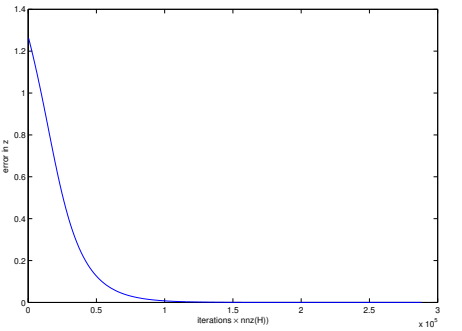
(a) Grid world



(b) Value function



(c) Comparison between original Z-learning algorithm and our generalization



(d) Power method

Figure 1: Numerical experiment

with

$$f(\mathbf{z}, \lambda) = D\left(\tfrac{1}{\lambda} H - I\right)\mathbf{z}, \quad g(\mathbf{z}, \lambda) = \mathbf{z}^T f(\mathbf{z}, \lambda),$$

where $H = RQ$ and $D = \operatorname{diag}(q_1, \ldots, q_n)$, with $(q_1, \ldots, q_n)$ the unique invariant probability distribution of the Markov chain with transition matrix Q. This ODE has the property that $z(t)(i) \geq 0$ for all i, and $\lambda(t) \geq 0$, for all $t \geq 0$, as long as the initial values are non-negative. The ODE has a unique equilibrium $(\mathbf{z}^*, \lambda^*)$, with λ^* and $\mathbf{z}^*$ equal to the Perron-Frobenius eigenvalue and (a rescaled version of) the Perron-Frobenius eigenvector of H.

Using linearization around the equilibrium and non-trivial matrix analysis, it can be shown that the ODE is locally asymptotically stable if and only if the matrix $D(H - \lambda I - \alpha P_{\mathbf{z}^*})$ is strictly stable for where $\alpha > 0$, $P_{\mathbf{z}^*}$ is the orthogonal projection along $\mathbf{z}^*$. For some important special cases this can be shown to be true, including the case where $q_1 = \ldots = q_n = \frac{1}{n}$, and the case where $H = H^T$. It remains to establish stability in full generality.

We may conclude that for the mentioned cases, and we expect more generally, there is a unique stable equilibrium point $(\mathbf{z}^*, \lambda^*)$ of the algorithm described in Section 3.2, equal to the Perron-Frobenius eigenvector and eigenvalue of H.

4 Numerical results

Consider the example of a gridworld (Figure 1 (a)), with some walls. Suppose the uncontrolled dynamics allow to move through the walls, but walking through a wall is very costly, say a cost of 100 per step through a wall is incurred. Where there is no wall, a cost of 1 per step is incurred. There is a single state, in the bottom right, where no costs are incurred. The uncontrolled dynamics q are such that with equal probability we may move left, right, up, down or stay where we are (but

it is impossible to move out of the gridworld). The value function for this problem can be seen in Figure 1 (b). In order to be able to compare our algorithm to the original Z-learning algorithm, the cost vector is normalized in such a way that $\lambda^* = 1$, so that Z-learning converges on the given input.

The result of running the stochastic approximation algorithm, with a constant gain of $\gamma = 0.05$ is portrayed in Figure 1 (c), where it is compared to Z-learning. This result may also be compared to the use of the power method in Figure 1 (d). Here the following version of the power method is used, in order to be able to give a fair comparison with our stochastic method.

$$\mathbf{z}_{m+1} = \mathbf{z}_m + \gamma(H\mathbf{z}_m - \mathbf{z}_m).$$

Note that for each iteration, the number of operations is (for sparse H) proportional to the number of non-zero elements in H. In the stochastic method the number of operations per iteration is of order 1. Comparing the graphs in Figure 1 (c) and (d), we see that our generalization of Z-learning does not disappoint in terms of speed of convergence, with respect t Z-learning as well as the power method. In the example it converges even slightly faster than Z-learning (executed with the same random seed), but this should in our understanding be seen as a coincidince.

5 Open questions

Some problems remain open. First of all, the algorithm presented in this paper needs to be investigated in more detail. In particular the stability of the equilibrium point of the ODE (2) remains to be established in full generality. Different variants of the update rules for λ and $\mathbf{z}$ can be imagined, which may result in better stability of the algorithm. For example, the size of the updates to λ depends on the size of $\mathbf{z}$, which currently is not restricted, and may therefore result in instabilities.

On a more general level, an interesting open question is how this algorithm performs as an algorithm that learns its optimal control while performing it at the same time. If the transition probabilities are constantly updated using $p_{ij} = \exp(u_i^*(j))q_{ij}$, with u^* as in (1), and p_{ij} is then taken as the new uncontrolled dynamics q_{ij}, we are effectively solving the problem of minimizing average cost where, as time goes to infinity, the KL-divergence between the original uncontrolled dynamics and the computed dynamics is allowed to approach infinity. This can be seen as a exploration-exploitation algorithm, like Q-learning (see e.g. [Bertsekas, Tsitsikils, 1996]), where in the beginning the environment is explored according to probability distribution q_{ij}, but as the algorithm proceeds it becomes more biased towards the optimal transition probabilities.

In case of a large state space, the stochastic algorithm can be used in case the matrix H is to big too fit into the memory of a computer. It only needs the current observation. We have an extension of the algorithm in mind that performs even when the state space itself is too large to fit in the working memory. Basically this amounts to only updating the components of $\mathbf{z}$ that correspond to states we have already seen. This algorithm needs to be developed and analysed further.

6 Summary

We have introduced a stochastic algorithm which may be used to solve Kullback-Leibler optimization problems online which extends the results of [Todorov, 2007] to average cost per stage optimization. It allows for rigourous stability analysis and performs well numerically. The method can be extended to a reinforcement learning style exploration-exploitation algorithm, which will be the subject of future research.

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement 231495.

References

A. Benveniste, M. Métivier, and P. Priouret. *Adaptive algorithms and stochastic approximations*, volume 22 of *Applications of Mathematics (New York)*. Springer-Verlag, Berlin, 1990. Translated from the French by Stephen S. Wilson.

D.P. Bertsekas and J.N. Tsitsiklis. Neuro-dynamic programming, athena scientific. *Belmont, MA*, 1996.

J. Bierkens and B. Kappen. Z-learning for average cost per stage Kullback-Leibler control. *In preparation*, 2011.

R.A. Horn and C.R. Johnson. *Matrix analysis*. Cambridge University Press, Cambridge, 1990. Corrected reprint of the 1985 original.

H.J. Kushner and G.G. Yin. *Stochastic approximation and recursive algorithms and applications*, volume 35 of *Applications of Mathematics (New York)*. Springer-Verlag, New York, second edition, 2003. Stochastic Modelling and Applied Probability.

C. Meyer. *Matrix analysis and applied linear algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

E. Todorov. Linearly-solvable markov decision problems. *Advances in neural information processing systems*, 19:1369, 2007.