Large-Scale Markov Decision Problems with KL Control Cost and its Application to Crowdsourcing

Yasin Abbasi-Yadkori YASIN.ABBASIYADKORI@QUT.EDU.AU
Queensland University of Technology

Peter L. Bartlett BARTLETT@CS.BERKELEY.EDU
University of California, Berkeley and Queensland University of Technology

Xi Chen XICHEN@NYU.EDU
Stern School of Business, New York University

Alan Malek MALEK@EECS.BERKELEY.EDU
University of California, Berkeley

Abstract

We study average and total cost Markov decision problems with large state spaces. Since the computational and statistical cost of finding the optimal policy scales with the size of the state space, we focus on searching for near-optimality in a low-dimensional family of policies. In particular, we show that for problems with a Kullback-Leibler divergence cost function, we can recast policy optimization as a convex optimization and solve it approximately using a stochastic subgradient algorithm. This method scales in complexity with the family of policies but not the state space. We show that the performance of the resulting policy is close to the best in the low-dimensional family. We demonstrate the efficacy of our approach by optimizing a policy for budget allocation in crowd labeling, an important crowdsourcing application.

1. Introduction

This paper studies Markov decision problems (MDPs) with expected average cost and total cost criteria. Given a transition matrix and loss function, the objective is to find a near-optimal policy in a reasonable amount of time. Let $\mathcal{X}$ be the state space with finite cardinality $X \in \mathbb{N}$, $\mathcal{A}$ be the action space with finite cardinality $A \in \mathbb{N}$, and $\ell(x, a)$ be the loss of taking an action $a \in \mathcal{A}$ at the state $x \in \mathcal{X}$. The average cost MDP policy design problem is to find the optimal state-to-action mapping (a.k.a. policy) π that minimizes the average cost $\lim_{t\to\infty} \frac{1}{t}\mathbb{E}\left[\sum_{s=1}^{t} \ell\left(x_s^\pi, \pi(x_s^\pi)\right)\right]$, where x_s^π is the state at time s under policy π. We will write the transition matrix as $P \in \mathbb{R}^{(X\times A)\times X}$ and use the shorthand $P_{(x,a),:}$ for the (x, a) row of P, i.e. the distribution of the state subsequent to x under action a. The total cost case is similar and the explicit formulation is given in the next section.

Finding the optimal policy is equivalent to solving the following linear program (Manne, 1960),

$$\max_{\lambda\in\mathbb{R}, h\in\mathbb{R}^X} \lambda\,, \qquad \text{s.t.} \quad \forall x \in \mathcal{X},\ (Lh)(x) \le \lambda + h(x)\,,$$

where L is the Bellman operator defined by

$$(Lh)(x) = \min_{a\in\mathcal{A}} \left(\ell(x, a) + P_{(x,a),:}h\right). \tag{1}$$

The scalar λ is the average loss and the vector h is called the differential value function. An optimal action achieves the minimum on the right-hand side of (1).

In general, exactly solving an MDP is computationally expensive (it is known to be P-complete (Papadimitriou & Tsitsiklis, 1987)). The widely used policy iteration and value iteration approaches require at least $O(X^2A)$ per-iteration complexity which precludes their use for problems with large state spaces. For large problems, it is reasonable to constrain the search to a low-dimensional family of policies. In particular, we follow the popular approach (Schweitzer & Seidmann, 1985; de Farias & Van Roy, 2003; Sutton & Barto, 1998) of using a linear approximation for the value function: for some fixed feature ma-

Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s).

trix $\Psi \in \mathbb{R}^{X \times d}$ with d much smaller than X, we will approximate $h(x)$ by $(\Psi w)(x)$ where w is a d-dimensional parameter vector.

This linear restriction has been famously applied to the LP formulation (Schweitzer & Seidmann, 1985) to obtain

$$\max_{\lambda \in \mathbb{R}, w \in \mathbb{R}^d} \lambda\,, \text{ s.t. } \forall x \in \mathcal{X},\ (L\Psi w)(x) \leq \lambda + (\Psi w)(x)\,.$$

Unfortunately, the problem is still difficult to solve because of the large number of constraints.

We can encapsulate this optimization problem by defining the Bellman error as the gap in the Bellman optimality condition, i.e. $\|Lh - h - \lambda\|_1$. Then, for some distribution u, solving the following optimization problem is a good proxy for the LP:

$$\min_w u^\top (L\Psi w - \Psi w)\,, \tag{2}$$

Appendix A explicitly relates optimal values of (2) to values of the LP; basically, solutions to (2) are nearly optimal in the low dimensional class of policies

$$\left\{ \pi \,:\, \exists w, \forall x,\ \pi(x) = \operatorname*{argmin}_{a \in \mathcal{A}} \left(\ell(x,a) + P_{(x,a),:}\Psi w \right) \right\}.$$

Unfortunately, (2) is not convex (the Bellman operator is a minimum over linear functions). Our contribution is to identify an important class of MDPs which allow us to transform (2) into a convex optimization and therefore solve it efficiently. Specifically, we consider the class of linearly solvable MDPs introduced by Todorov (2006; 2009a). Given some finite state space $\mathcal{X}$, the action space $\mathcal{P}$ is the space of state transition matrices, i.e. all $X \times X$ matrices with rows that are probability distributions over the state space. Given some function $q : \mathcal{X} \to [0, Q]$ and some fixed transition matrix $P_0 \in \mathcal{P}$, the KL loss function is defined as

$$\ell(x, P) = q(x) + \sum_{x' \in \mathcal{X}} P(x, x') \log \frac{P(x, x')}{P_0(x, x')}\,. \tag{3}$$

Note that if there is an $x' \in \mathcal{X}$ with $P(x, x') > 0$ and $P_0(x, x') = 0$, then $\ell(x, P) = \infty$. Thus, the decision maker is forced to only play distributions that are absolutely continuous with P_0. The P_0 distribution can be thought of as representing some passive dynamics or base policy. For many problems (e.g., the crowdsourcing application in Section 4), the transition matrix P can be transformed into a distribution μ over the action space such that taking an action according to μ in state x results in a transition to a state distributed according to $P(x, \cdot)$. The KL cost tends to keep explored policies in the vicinity of P_0. This is useful in the setting when we already have a good policy or heuristic that we would like to optimize further or when there is uncertainty in the model specification and we do not want to deviate too much.

The beauty of using KL cost, as shown in Todorov (2009a), is that the Bellman operator becomes a linear operator after an exponential transformation $h \mapsto \exp(-h)$. Todorov exploited this fact by solving for the optimal policy directly. We exploit this by linearly parameterizing $\exp(-h) \approx \Phi w$, which allows us to recast (2) as a convex optimization in w. We reduce the number of constraints by sampling and using a stochastic subgradient descent method. Our method can find a near optimal solution in the low-dimensional class while maintaining a computational cost that is independent of the size of the state space.

Note that exploiting the linear solvability of the KL control cost will not scale to large state spaces. Even in the best case, the computational cost of solving such linear problems is $\Omega(X)$. Why is even $\Omega(X)$ too high? For many realistic problems, the state space size grows exponentially; for queueing networks, the state space is exponential in the number of queues, and for budget allocation, in the time horizon. As a specific example, the crowdsourcing application presented in Section 4 has around $20^{500} \approx 3 \times 10^{650}$ states.

For these reasons, it is essential to develop algorithms whose computation is independent of X. Our contribution is to present such a method that provably finds the best policy in a restricted class.

1.1. Related Work

Popular approaches to solve large-scale MDPs include Approximate Dynamic Programming (ADP) and Approximate Linear Programming (ALP). One can find theoretical results on ADP in Bertsekas & Tsitsiklis (1996); Sutton & Barto (1998); Bertsekas (2007) and more recent papers on Temporal-Difference (TD) methods (Sutton et al., 2009b;a; Maei et al., 2009; 2010). ALP methods, first introduced in Schweitzer & Seidmann (1985), were improved and analyzed more thoroughly in a long string of papers including (de Farias & Van Roy, 2003; 2004; 2006; Hauskrecht & Kveton, 2003; Guestrin et al., 2004; Petrik & Zilberstein, 2009; Desai et al., 2012; Veatch, 2013).

As noted by Desai et al. (2012), the prior work on ALP and ADP either requires access to samples from specialized distributions (such as a distribution that depends on the optimal policy) or assumes the ability to solve an LP with as many constraints as states. (See for example, Appendix A in Abbasi-Yadkori et al. (2014) for a more detailed discussion.) Asymptotic behavior of TD methods (Sutton, 1988) in large state and action spaces was studied both in *on-policy* (Tsitsiklis & Van Roy, 1997) and *off-policy* (Sutton et al., 2009b;a; Maei et al., 2009) settings. All these results concern the policy estimation problem, i.e., estimating the value of a fixed policy. The available results for the control problem, i.e., estimating the value of the optimal policy, are more limited (Maei et al., 2010) and prove only convergence to a local optimum of some objective functions.

Todorov (2009b) and Zhong & Todorov (2011a;b) propose computationally efficient algorithms for linearly solvable MDPs with large state spaces. These methods however lack theoretical guarantees. In this paper, we provide strong theoretical guarantees for the log-linear value function approximation.

Abbasi-Yadkori et al. (2014) studied average cost MDPs in the dual space with stationary distributions as variables and showed a convex optimization reduction. They propose computationally efficient algorithms and show that the quality of the solution is close to the best in the comparison class. In this work, we study the problem in the primal space with value functions as variables. In contrast to solving MDPs in the dual space, we encounter additional difficulty arising from the non-convexity of the resulting optimization problem.

1.2. Notation

We use $M(i,:)$ and $M(:,j)$ to denote the ith row and jth column of matrix M, respectively. For some positive vector C, we define scaled norms $\|v\|_{1,c} = \sum_i c_i |v_i|$ and $\|v\|_{\infty,c} = \max_i c_i |v_i|$. The all ones vector in denoted $\mathbf{1}$. Finally, for vectors v and w, the following operations are always element-wise: $v \leq w$, $\max(v,w)$, $\min(v,w)$ and $v \odot w$, where the latter denotes multiplication.

2. MDPs with Total Cost Criterion

The first part of this section defines the KL control cost MDP and demonstrates why it is linearly solvable. The second part describes our contribution: we reduce a high-dimensional KL control cost MDP into a tractable problem on a low dimensional subspace and present an algorithm which does provably well with respect to the best in the subclass of policies. The next section extends this analysis to the average cost case.

Starting from some initial state x_1, the total cost is

$$J_*(x_1) = \min_{P\in\mathcal{P}} \mathbb{E}\left[\sum_{t=1}^{\infty} \ell(x_t, P)\right] . \tag{4}$$

To ensure this is well defined, we assume that the MDP hits a terminating (goal) state z with $q(z) = 0$ and $P_0(z,z) = 1$, i.e. z is an absorbing state. This implies that once the learner reaches z, the only reasonable policy is $P(z,z) = 1$ and thus $J_*(z) = 0$.

In the KL cost setting, the Bellman operator is defined as

$$(LJ)(x) := \min_{P\in\mathcal{P}} \left(\ell(x,P) + \sum_{x'\in\mathcal{X}} P(x,x')J(x')\right) \tag{5}$$

and the optimal value function satisfies $LJ = J$. Given some function J, the minimizer of (5) is called the greedy policy with respect to J, denoted P_J. The KL cost has the property that the greedy action is trivial to compute:

$$\begin{aligned} P_J(x,:) &= \operatorname*{argmin}_{P\in\mathcal{P}} \left\{\ell(x,P) + \sum_{x'\in\mathcal{X}} P(x,x')J(x')\right\} \\ &= \operatorname*{argmin}_{P\in\mathcal{P}} \sum_{x'\in\mathcal{X}} P(x,x')\left(\log \frac{P(x,x')}{P_0(x,x')e^{-J(x')}}\right), \end{aligned}$$

where we use the definition of $\ell(x,P)$ in (3). Using the fact that the KL divergence of P and P is 0, the minimum is

$$P_J(x,x') = \frac{P_0(x,x')e^{-J(x')}}{Z(x)}, \tag{6}$$

where $Z(x) = \sum_{x'} P_0(x,x')e^{-J(x')}$ is the normalizing factor. This allows us to write the Bellman operator as

$$\begin{aligned} (LJ)(x) &= \ell(x,P_J) + \sum_{x'\in\mathcal{X}} P_J(x,x')J(x') \\ &= q(x) + \sum_{x'\in\mathcal{X}} P_J(x,x')\left(\log\frac{P_J(x,x')}{P_0(x,x')} + J(x')\right) \\ &= q(x) + \sum_{x'\in\mathcal{X}} P_J(x,x')\log\frac{1}{Z(x)} \\ &= q(x) - \log Z(x). \end{aligned}$$

This formula was exploited in Todorov (2006) by noting that exponentiating Bellman's optimality equation yields the equations

$$e^{-J(x)} = e^{-(LJ)(x)} = e^{-q(x)}Z(x) = e^{-q(x)}P_0(x,:)e^{-J}.$$

In vector form, we have $e^{-J} = e^{-q}P_0e^{-J}$, which is linear and will yield the optimal value function, J^*. The optimal policy is P_{J^*}.

While this result is striking, it does not directly yield efficiently computable solutions since we would need to solve an eigenvalue problem in X dimensions. Our contribution is to formulate a computationally efficient algorithm guaranteed to return the best policy from a restricted class of policies.

Motivated by the exponential transformation above, consider the class of value functions

$$\mathcal{J} = \{x \mapsto J_w(x) := -\log(\Psi(x,:)w) : w \in \mathcal{W}\} , \tag{7}$$

where $\Psi \in \mathbb{R}^{X \times d}$ is a feature matrix and $\mathcal{W} \subset \mathbb{R}^d$ is some bounded set. This parameterization is different from ADP and ALP which use a linear combination of basis functions. The logarithmic transformation ensures that after the exponentiation the transformed value function is linear. Additionally, to keep the value functions well defined, we make the following positivity assumption on features Ψ and the set $\mathcal{W}$: we assume that there exists a constant $g > 0$ such that for any $w \in \mathcal{W}$ and any state x, $\Psi(x, :)w \geq g$.

In the following section, we solve the problem: given an MDP specified by q and P_0 and a class of value functions $\mathcal{J}$, find parameters $\widehat{w}$ such that[1]

$$J_{P_{J_{\widehat{w}}}}(x_1) \approx \min_{J_w \in \mathcal{J}} J_{P_{J_w}}(x_1) \tag{8}$$

in time polynomial in d and independent of X. We assume that access to arbitrary entries of Ψ, q, P_0, or $e^{-q}P_0\Psi$ takes unit time.

2.1. Recasting as a Convex Optimization

Recall from the introduction that the Bellman error was defined as the gap in the Bellman optimality equation. For the case of total cost, this is $|LJ_w(x) - J_w(x)|$. Also, the introduction and Appendix A argued that keeping the Bellman error small ensured that J_w was a good approximation to the value function. Since the Bellman operator evaluates to $LJ = q - \log Z$, choosing $e^{-J_w(x)} = \Psi(x, :)w$ as our parameterization of J_w from (7) causes the Bellman operator to become $e^{-(LJ_w)(x)} = e^{-q(x)}P_0(x, :)\Psi w$. Thus,

$$e^{-J_w(x)} - e^{-(LJ_w)(x)} = \Psi(x, :)w - e^{-q(x)}P_0(x, :)\Psi w,$$

which is linear in w. This implies that $|e^{-J_w(x)} - e^{-(LJ_w)(x)}|$ is convex in w and can be minimized efficiently. Finally, we argue that minimizing the exponentiated Bellman error will force the Bellman error $|LJ_w(x) - J_w(x)|$ to be small since

$$\begin{aligned} e^{-\max\{u,u'\}}\,|u - u'| &\leq \left|e^{-u} - e^{-u'}\right| \\ &\leq e^{-\min\{u,u'\}}\,|u - u'| \ . \end{aligned} \tag{9}$$

We have now motivated our KL-cost convex optimization formulation. Let $\mathcal{T}$ be the space of trajectories starting at state x_1 and ending at state z and let v be some probability distribution over $\mathcal{T}$. For H some positive constant, we choose w as the minimizer of the cost function

$$\begin{aligned} c(w) := &-\log(\Psi(x_1, :)w) \\ &+ H \sum_{T \in \mathcal{T}} v(T) \sum_{x \in T} \left|\Psi(x, :)w - e^{-q(x)}P_0(x, :)\Psi w\right| , \end{aligned} \tag{10}$$

where the $-\log(\Psi(x_1, :)w)$ term is the approximate value of the policy, $J_w(x_1)$, and the summation is an estimate of the Bellman error of J_w. In other words, we are approximately solving the constrained optimization problem:

$$\min_{w \in \mathcal{W}} J_w(x_1) \quad \text{s.t.} \quad e^{-J_w(x)} - e^{-(LJ_w)(x)} = 0 \ \forall x \in \mathcal{X}$$

by introducing the constraints as a regularization instead. The intuition is that strict feasibility of w is not necessary as it will still produce a valid policy. Instead, we only need the regularization to ensure that $J_w(x_1)$ is a good estimate for $J_{P_{J_w}}$.

2.2. Approximate Minimization Algorithm

In the next section, we give upper bounds on the error of any ϵ-optimal $\widehat{w}$ (i.e. $w \in \mathcal{W}$, $c(\widehat{w}) \leq c(w) + \epsilon$). In principle, the precise method for obtaining an approximate minimizer is not central to the analysis: the novel ideas are in relating the convex objective (10) to the underlying MDP. One caveat is that, for large problems, we cannot sum over $\mathcal{T}$. Thus, we will describe one particular method for handling this difficulty.

An unbiased estimate of the subgradient of (10) is easy to calculate as it decomposes for each $T \in \mathcal{T}$. In fact, it is straightforward to see that if we sample a trajectory $T \sim v$, then the following is an unbiased estimate of the subgradient:

$$\begin{aligned} r(w) = &-\left(\frac{1}{(\Psi(x_1, :)w)}\right)\Psi(x_1, :) \\ &+ H \sum_{x \in T} \Bigg[\text{sign}\left(\Psi(x, :)w - e^{-q(x)}P_0(x, :)\Psi w\right) \\ &\qquad \left(\Psi(x, :) - e^{-q(x)}P_0(x, :)\Psi\right)\Bigg]. \end{aligned} \tag{11}$$

Since we have a tractable subgradient estimation, a natural class of algorithms to minimize $c(w)$ is stochastic subgradient descent methods (see e.g. Kushner & Yin (2003) and references therein) which can obtain an ϵ-optimal solution in $O(\frac{1}{\epsilon^2})$ iterations with high probability under certain mild conditions. The details of the stochastic subgradient method are provided in Figure 1. We note that the recently developed accelerated or distributed stochastic subgradient algorithms (e.g., Sra et al. (2011); Zinkevich et al. (2010); Duchi et al. (2012) and references therein) or different averaging schemes of $\{w_t\}$ to obtain $\widehat{w}_T$ (Shamir & Zhang, 2013) can all be applied to minimize c.

As our algorithm is based on a stochastic subgradient method, the per-step computation cost is linear in the number of features and the algorithm can be adapted to the online and offline settings.

[1] J_{P_J} is the value function under the policy P_J.

```
Input: Starting state $x_1$, number of rounds $N$, constant $H$, a decreasing sequence of step sizes $(\eta_t)$, a distribution $v$ over trajectories.
Let $\Pi_{\mathcal{W}}$ be the Euclidean projection onto $\mathcal{W}$.
Initialize $w_1 = \mathbf{0}$.
for $t := 1, 2, \dots, N$ do
    Sample trajectory $(x_1, a_1, \dots, z) \sim v$.
    Compute the stochastic subgradient $r_t$ defined by (11).
    Update $w_{t+1} = \Pi_{\mathcal{W}}(w_t - \eta_t r_t)$.
end for
$\widehat{w}_T = \frac{1}{T}\sum_{t=1}^{T} w_t$.
Return policy $P_{J_{\widehat{w}_T}}$ defined by (6), (7).
```

Figure 1. The Stochastic Subgradient Method for Total Cost Markov Decision Processes

2.3. Analysis

Assume that some approximate optimization algorithm (such as a stochastic gradient method) produces a $\widehat{w}$ that is ϵ-optimal, i.e., for any $w \in \mathcal{W}$, $c(\widehat{w}) \le c(w) + \epsilon$. The first step in the analysis is to relate the suboptimality of $\widehat{w}$ to the suboptimality of the policy. Specifically, we bound the gap between the value function under the policies $P_{J_{\widehat{w}}}$ and P_{J_w} for any $w \in \mathcal{W}$ in terms of the Bellman errors. Recall that g is a lower bound on $\Psi(x,:)w$, and Q is an upper bound on q.

Theorem 1. *Assume that $\widehat{w}$ is ϵ-optimal and choose any $H \ge e^{Q - \log g}$. Then, for any $w \in \mathcal{W}$ with $l_w = \min(J_w, LJ_w)$, we have,*

$$\begin{aligned} J_{P_{J_{\widehat{w}}}}(x_1) - J_{P_{J_w}}(x_1) &\le \epsilon \\ &+ \|P_{J_{\widehat{w}}} - v\|_1 \max_{T\in\mathcal{T}} \sum_{x\in T} |J_{\widehat{w}}(x) - LJ_{\widehat{w}}(x)| \\ &+ \sum_{T\in\mathcal{T}} P_{J_w}(T) \sum_{x\in T} |J_w(x) - LJ_w(x)| \\ &+ H \sum_{T\in\mathcal{T}} v(T) \sum_{x\in T} e^{-l_w(x)} |J_w(x) - LJ_w(x)| \,, \end{aligned} \tag{12}$$

where, with an abuse of notation, $P_J(T)$ denotes the probability of trajectory T under transition dynamics P_J.

The full proof is in Appendix B, but below is a rough outline. We first bound the gap between value function J_w and the value function of the greedy policy w.r.t. J_w (i.e., $J_{P_{J_w}}$) in terms of the Bellman error $|J_w(x) - LJ_w(x)|$ at w. This indicates that if the Bellman error at w is small, then we have $J_w \approx J_{P_{J_w}}$. The gap between J_w and $J_{\widehat{w}}$ can be bounded using the fact that $\widehat{w}$ is an ϵ-optimal solution of the convex optimization problem. We thus have control of the gaps between: J_w and $J_{P_{J_w}}$; $J_{\widehat{w}}$ and $J_{P_{J_{\widehat{w}}}}$; and J_w and $J_{\widehat{w}}$. Combining these yields the bound between $J_{P_{J_{\widehat{w}}}}(x_1)$ and $J_{P_{J_w}}(x_1)$ in Theorem 1.

In addition to the optimality gap ϵ, there are three terms in the right hand side of the inequality (12). The first term is related to the Bellman error at $\widehat{w}$. This term becomes small whenever the Bellman error at $\widehat{w}$ is small and/or the sampling distribution v is close to $P_{J_{\widehat{w}}}$. Unfortunately, $P_{J_{\widehat{w}}}$ itself depends on v. Note that such dependencies also exist in the results of de Farias & Van Roy (2003; 2006); Desai et al. (2012). Controlling this distribution mismatch term remains a challenging open problem. The second term in the bound is the expectation of the Bellman error $\sum_{x\in T} |J_w(x) - LJ_w(x)|$ under the distribution over $\mathcal{T}$ determined by the transition dynamics P_{J_w}, while the third term is the expectation of the Bellman error under the distribution v over $\mathcal{T}$. Thus, both the second and third terms are small if one can find a $J_w \in \mathcal{J}$ which leads to a small Bellman error. The size of the Bellman error at w is determined by the value function class $\mathcal{J}$ in (7), which depends on the feature matrix Ψ. This point can be seen more clearly from the following equivalent statement of Theorem 1,

$$\begin{aligned} J_{P_{J_{\widehat{w}}}}(x_1) &\le \inf_{J_w\in\mathcal{J}} \left\{ J_{P_{J_w}}(x_1) + V(J_w) \right\} + \epsilon \\ &\quad + \|P_{J_{\widehat{w}}} - v\|_1 \max_{T\in\mathcal{T}} \sum_{x\in T} |J_{\widehat{w}}(x) - LJ_{\widehat{w}}(x)| \,, \\ V(J_w) &= H \sum_{T\in\mathcal{T}} v(T) \sum_{x\in T} e^{-l_w(x)} |J_w(x) - LJ_w(x)| \\ &\quad + \sum_{T\in\mathcal{T}} P_{J_w}(T) \sum_{x\in T} |J_w(x) - LJ_w(x)| \,. \end{aligned}$$

As we can see, $V : \mathcal{J} \to \mathbb{R}_+$ can be viewed as a "penalty function" which represents how far J_w is from the optimal value function, since $V(J_w) = 0$ if $LJ_w = J_w$ (i.e., the Bellman error is zero). Theorem 1 shows that the value function of the policy corresponding to $\widehat{w}$, i.e., $J_{P_{J_{\widehat{w}}}}(x_1)$, is upper bounded by the best possible greedy policy w.r.t. J_w measured by the sum of the value $J_{P_{J_w}}(x_1)$ and the penalty $V(J_w)$. In practice, it is important to choose a good feature matrix Ψ which exploits some underlying structure of the problem so that for some choice of J_w, $J_{P_{J_w}}(x_1) + V(J_w)$ is small.

3. MDPs with Average Cost Criterion

Here, we extend the previous analysis to KL control cost MDPs with an average cost criterion. The average cost of policy P starting at state x_1 is

$$\begin{aligned} \lambda_P &= \lim_{t\to\infty} \frac{1}{t} \mathbb{E}\left[\sum_{s=1}^{t} \ell(x_t, P) \right] \\ &= \lim_{t\to\infty} \frac{1}{t} \sum_{s=1}^{t} P^s(x_1, :)\ell(:, P) = P^\infty(x_1, :)\ell(:, P) \,, \end{aligned}$$

where we have assumed that the Markov chain induced by policy P is irreducible and aperiodic (see, for example, Puterman (1994)), and $P^\infty = \lim_{s\to\infty} P^s$, which is independent of the starting state x. Thus, $\lambda_P \mathbf{1} = P^\infty \ell(:, P)$. Our goal is to find $P_* = \operatorname{argmin}_{P\in\mathcal{P}} \lambda_P$, the optimal policy.

As before, exact methods do not scale to large state spaces and we need to reduce our search to a parameterized class of approximate differential value functions and restrict our search to greedy policies. Fortunately, we can apply much of the total-cost analysis to the average cost case.

The Bellman optimality equation for the average cost problems is $Lh = \lambda\mathbf{1} + h$, where λ is a scalar and h is an X-dimensional vector. If a pair (λ_*, h_*) satisfy this equation, then λ_* is the average loss of the optimal policy and vector h_* is called the differential value function of the optimal policy. The greedy policy with respect to any function $h : \mathcal{X} \to \mathbb{R}$, denoted by P_h, is defined by $P_h(x) = \operatorname{argmin}_{P\in\mathcal{P}}\{\ell(x, P) + \sum_{x'} P(x, x')h(x')\}$.

Similar to the previous section, we consider parameterized approximate differential value functions with some feature matrix $\Psi \in \mathbb{R}^{X\times d}$ and a bounded parameter set $\mathcal{W} \subset \mathbb{R}^d$. Specifically, define the class of approximate differential value functions $\mathcal{H} = \{x \mapsto h_w(x) := -\log(\Psi(x, :)w) : w \in \mathcal{W}\}$.

Our goal in this section is to prove that we can find a $\widehat{w}$ such that $\lambda_{P_{h_{\widehat{w}}}} \approx \min_{h_w\in\mathcal{H}} \lambda_{P_{h_w}}$ for KL control cost MDPs with an algorithm that has complexity polynomial in d and independent of X. We assume the same computational model as before: access to arbitrary entries of Ψ, q, P_0, and $e^{-q}P_0\Psi$ takes unit time.

Let b be an estimate of the optimal average cost. Recall the bound on the exponential of a difference from (9). Applying this bound to $|Lh(x) - (b + h(x))|$, the average cost Bellman error, allows us to argue that if $\left|e^{-q(x)}P_0(x, :)\Psi w - e^{-b}\Psi(x, :)w\right|$ is small, then the Bellman error in state x is also small. This motivates the convex optimization problem

$$\min_{w\in\mathcal{W}} c(w) := v^\top \left|e^{-b}\Psi w - e^{-q}P_0\Psi w\right| \qquad (13)$$
$$\equiv \sum_{T\in\mathcal{T}} v(T) \sum_{x\in T} \left|e^{-b}\Psi(x, :)w - e^{-q(x)}P_0(x, :)\Psi w\right|,$$

where v is a positive vector and $c(w)$ is convex in w. We note that instead of requiring v to be a distribution, we choose v to be an “unnormalized" distribution, i.e., $v = Cu$ where $C > 0$ is a positive constant and u is a distribution. The performance bound of a greedy policy w.r.t. $h_{\widehat{w}}$ for an ϵ-optimal solution $\widehat{w}$ is provided in the next theorem.

Theorem 2. *Let P_{h_w} be the greedy policy with respect to value function h_w. Also, let $\widehat{w}$ be an ϵ-optimal solution and $u_{\widehat{w}} = \max(Lh_{\widehat{w}}, h_{\widehat{w}} + b)$. Then, for any $w \in \mathcal{W}$ with $l_w = \min(Lh_w, h_w + b)$, we have,*

$$\lambda_{P_{h_{\widehat{w}}}} - \lambda_{P_{h_w}} \le \left|b - \lambda_{P_{h_w}}\right| \qquad (14)$$
$$+ \|(e^{-u_{\widehat{w}}} \odot v - \mu_{P_{h_{\widehat{w}}}})\|_1 \|Lh_{\widehat{w}} - h_{\widehat{w}} - b\|_\infty$$
$$+ \|\,(Lh_w - h_w - b)\,\|_{1,(e^{-l_w}\odot v)} + \epsilon\,,$$

where $\odot$ denotes component-wise multiplication of vectors and $\mu_{P_{h_{\widehat{w}}}}$ is the stationary distribution of the transition dynamics $P_{h_{\widehat{w}}}$.

Similar to Theorem 1, the right hand side of (14) also involves the Bellman error at w (i.e., $Lh_w - h_w - b$) and a distribution mismatch term (i.e., $e^{-u_{\widehat{w}}} \odot v - \mu_{P_{h_{\widehat{w}}}}$). How small the term $\|Lh_w - h_w - b\|_{1,v}$ can be depends on the class of differential value functions $\mathcal{H}$, which further depends on the choice of the feature matrix Ψ. We also note that the term $\left|\lambda_{P_{h_w}} - b\right|$ in (14) characterizes the quality of the estimate b. In practice, one can formulate the problem of estimation of b as a one-dimensional optimization problem, which can be solved by zero-order optimization approaches. The proof of Theorem 2 is provided in Appendix C. To obtain an ϵ-optimal solution $\widehat{w}$, one can apply the stochastic subgraident descent algorithm. We note that instead of sampling a trajectory T as in Figure 1, one only needs to sample a state x from the distribution defined by $v(x)/\|v\|_1$. The detailed algorithm is presented in Appendix C due to space constraints.

4. Applications to Crowdsourcing

So far, we have motivated, presented, and proven error bounds for the total cost and average cost versions of KL control cost MDPs. The rest of the paper is devoted to applying our algorithm to a fundamental problem in crowdsourcing: budget allocation. We are given a set of items and want to learn some binary label for each item. For a unit cost, we can have a member of the crowd return a noisy label. Some labels are harder to identify than others and receive noisier raw labels. The goal of budgeted learning is to optimally query items to maximize the number of correctly inferred labels. Intuitively, some instances are easy and require few raw labels for a consensus while more ambiguous instances will require more raw labels to obtain a good estimate.

This budget allocation problem in crowdsourcing has been studied recently in Chen et al. (2013). We adopt the same model here: the learner can submit labeling tasks to a worker pool with anonymous workers but cannot assign items to particular workers. There are A instances with the latent true labels $Z_i \in \{0, 1\}$ for $1 \le i \le A$. The labeling difficulty for the i-th instance is characterized by its soft label $\theta_i \in [0, 1]$, which is defined as the probability that the i-th instance will be labeled as positive by a random worker. When θ_i is close to 0 or 1, the i-th instance is an easy one; while when θ_i is close to 0.5, the instance is highly ambiguous. It is also assumed that the soft label is consistent with the true label, i.e., $\theta_i \geq 0.5$ iff $Z_i = 1$.

Assume that the budget is B. For each stage $0 \leq t \leq B-1$, an instance i_t from the action set $\mathcal{A} = \{1, \ldots, A\}$ is chosen and the item is queried. The decision maker receives the label before choosing the next item. In a pull marketplace with anonymous workers, it is reasonable to assume that workers provide the label y_{i_t} according to a Bernoulli distribution with the parameter θ_{i_t}, i.e., $\Pr(y_{i_t} = 1) = \theta_{i_t}$. When the budget is exhausted, the decision maker infers the true labels by aggregating all the collected labels. More specifically, let $H^* = \{i : Z_i = 1\} = \{i : \theta_i \geq 0.5\}$ be the set of true positive labels. At the final stage B, the decision maker estimates the positive set H^B and the overall cost is defined using the binary classification error $|H^B \cap (H^*)^c| + |(H^B)^c \cap H^*|$. Our goal is to find an allocation policy that minimizes the binary classification error.

In Chen et al. (2013), the budget allocation problem was formulated as a Bayesian MDP with a total cost criterion. Each θ_i is assumed to be drawn from $\text{Beta}(a_i^0, b_i^0)$. At each stage t, the decision maker chooses an instance i_t from the action set $\mathcal{A}$ and observes its label $y_{i_t} \sim \text{Bernoulli}(\theta_{i_t})$. By the conjugacy between Beta and Bernoulli, the posterior of θ_{i_t} will be updated by

$$\text{Beta}(a_{i_t}^{t+1}, b_{i_t}^{t+1}) = \begin{cases} \text{Beta}(a_{i_t}^t + 1, b_{i_t}^t) & \text{if} \quad y_{i_t} = 1; \\ \text{Beta}(a_{i_t}^t, b_{i_t}^t + 1) & \text{if} \quad y_{i_t} = -1. \end{cases}$$

Therefore, the state at the stage t, x^t, can be characterized by an $A \times 2$ state matrix where each row contains two elements, $\{a_i^t, b_i^t\}$. The state space at stage t is

$$\mathcal{X}^t = \Bigg\{ x^t = \{a_i^t, b_i^t\}_{i=1}^A : a_i^t - a_i^0 \in \mathbb{N}_0, \\ b_i^t - a_i^0 \in \mathbb{N}_0, \sum_{i=1}^A (a_i^t - a_i^0) + (b_i^t - b_i^0) = t \Bigg\},$$

where $\mathbb{N}_0$ denotes the nonnegative integers and we use $\{a_i^t, b_i^t\}_{i=1}^A$ to represent the $A \times 2$ state matrix. Therefore, the state space for the entire budget is $\mathcal{X} = \bigcup_{t=0}^B \mathcal{X}^t$, which grows exponentially with B. Under this model, the state transition probability can be easily calculated as the posterior probability of the next state given the current state x^t and the action i_t; $\Pr(y_{i_t} = 1 | x^t, i_t) = \mathbb{E}(\theta_{i_t} | x^t) = \frac{a_{i_t}^t}{a_{i_t}^t + b_{i_t}^t}$.

The objective is to find a budget allocation policy which minimizes the posterior classification error given the final state x^B. By defining $I(a, b) = \Pr(\theta \geq 0.5 | \theta \sim \text{Beta}(a, b))$ and $h(x) = \min(x, 1-x)$, Proposition 2.1 in Chen et al. (2013) shows that the posterior classification error takes the following form,

$$\mathbb{E}\left(\sum_{i=1}^A (\{i \in H\}\{i \notin H^*\} + \{i \notin H\}\{i \in H^*\}) \Big| x^B \right) \\ = \sum_{i=1}^A h\left(I(a_i^B, b_i^B)\right), \tag{15}$$

where $\{\cdot \in A\}$ is the indicator function of A. Therefore, the loss function as a function of the state can be expressed as

$$q(x) = \begin{cases} 0 & x \in \bigcup_{t=0}^{B-1} \mathcal{X}^t, \\ \sum_{i=1}^A h\left(I(a_i^B, b_i^B)\right) & x \in \mathcal{X}^B. \end{cases} \tag{16}$$

So far, we have formulated the budget allocation problem in crowdsourcing as an MDP with a total cost criterion with a loss that only depends on the state. Once we define some passive dynamics P_0, we can form a KL control cost MDP. However, since the state space is exponentially large in B, even the nice properties of KL control cost MDPs would not make an exact solution tractable. Therefore, we will apply the stochastic subgradient method from Section 2. Once we obtain $P_{J_{\widehat{w}}}$ based on the learned $\widehat{w}$, it can be easily transformed into a distribution over the action space (i.e., randomized policy mapping from state space to action space). In particular, given the current $A \times 2$ state matrix $x = \{a_i, b_i\}_{i=1}^A$, the next state matrix is $x' \in \{x + (e_i, 0) : i = 1, \ldots, A\} \cup \{x + (0, e_i) : i = 1, \ldots, A\}$, that is, there are $2 \cdot A$ possible next states. Here, e_i denotes the $A \times 1$ column vector with 1 at the i-th entry and 0 at all other entries. Therefore, given $P_{J_{\widehat{w}}}$, we can define the probability of taking an action $i \in \mathcal{A}$ as $p(i|x) = P_{J_{\widehat{w}}}(x, x + (e_i, 0)) + P_{J_{\widehat{w}}}(x, x + (0, e_i))$.

We also note that the stochastic subgradient method in Figure 1 has a per-iteration complexity independent of X. Although X is exponential in B, the state transition matrix is extremely sparse since for each state x there are at most $2 \cdot A$ possible next states. The dominant term in the subgradient computation, $P_0(x, \cdot)\Psi$, has complexity of only $O(Ad)$ and thus the per-iteration complexity is $O(ABd)$ which is independent of X. Also, the computation of the normalization term $Z(x)$ in (6) has complexity of only $O(A)$.

One last point of discussion is how to choose P_0. As mentioned in the introduction, any reasonable policy is also a natural choice for the "passive" dynamics P_0 as the stochastic subgradient method will improve on it. Therefore, we construct P_0 based on the optimistic knowledge gradient policy (Opt-KG) proposed in Chen et al. (2013), which was shown to work quite well in practice. Opt-KG is an index policy that selects the next instance based on the optimistic outcome of the cost. More precisely, for any given state (a, b) of a single instance, the differ-

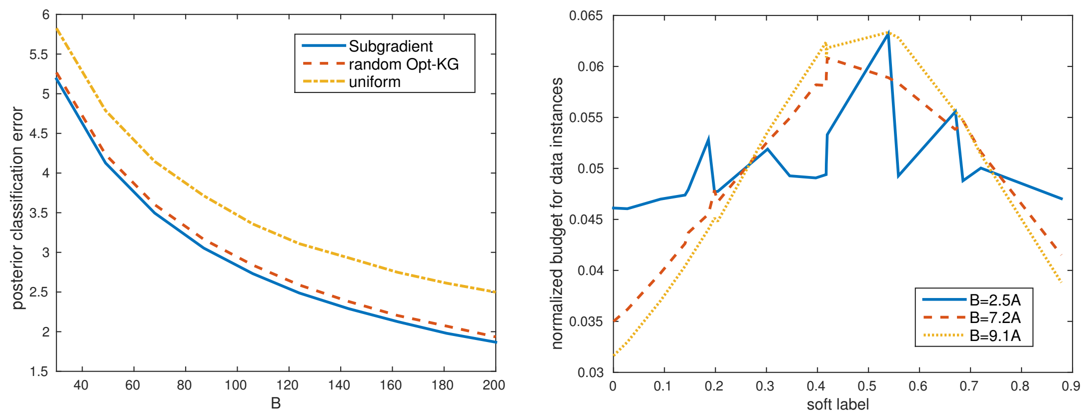

Figure 2. The left plot compares the average posterior classification error of the opt-KG policy, the uniform policy, and our optimized policy for a variety of budgets with $A = 20$ labels to learn. The optimized policy makes improvements for the entire budget range. The right plot indicates the portion of the budget used for each label as a function of the soft label. Soft labels closer to .5 are the hardest, and we see that they receive more budget, especially as B grows. For large B, the policy spends most of the budget on the hard labels, but for small B, the budget is spread more uniformly.

ence/reduction of cost in getting an extra positive or negative label takes the following form: $C_1(a,b) = h(I(a+1,b)) - h(I(a,b)), C_{-1}(a,b) = h(I(a,b+1)) - h(I(a,b))$.

Given the current state $x = \{a_i, b_i\}_{i=1}^A$, Opt-KG associates the i-th instance with a score defined as $C(a_i,b_i) = \min(C_1(a_i,b_i), C_{-1}(a_i,b_i))$, which is based on the minimum cost of obtaining the positive and negative labels. It can be proven that $C(a_i,b_i) < 0$ for all possible a_i and b_i. The randomized Opt-KG policy chooses the instance $i \in \mathcal{A}$ with the probability $p(i|x) = \frac{|C(a_i,b_i)|}{\sum_{i'=1}^{A} |C(a_{i'},b_{i'})|}$. Then, the "passive" dynamics P_0 can be defined based on the randomized Opt-KG policy as for each $i \in \mathcal{A}$,

$$P_0(x, x + (e_i, 0)) = p(i|x) \cdot \frac{a_i}{a_i + b_i},$$

$$P_0(x, x + (0, e_i)) = p(i|x) \cdot \frac{b_i}{a_i + b_i}.$$

In addition, the distribution v over the trajectory space $\mathcal{T}$ can also be constructed based on Opt-KG and one can easily sample the trajectory from v.

4.1. Numerical Testing

We test the proposed stochastic subgradient method on a simulated dataset with (known) soft labels θ_i sampled uniformly from $[0,1]$; this corresponds to $a_i^0 = b_i^0$. To optimize the policy, we need knowledge of θ_i in order to evaluate q; however, the policy, of course, will not need to know θ_i. We use the randomized Opt-KG policy as both the "passive" dynamics P_0 and the trajectory sampling distribution v. The feature matrix is generated using the moments of Beta distributions as follows. For each state $x = \{a_i, b_i\}_{i=1}^A$, let $X_i \sim \text{Beta}(a_i, b_i)$ and define the feature vector $\Psi(x,.) \in \mathbb{R}^{3A+1}$ as the concatenation of the 3-tuples: $\mathbb{E}(X_i) = \frac{a_i}{a_i+b_i}$, $1 - \mathbb{E}(X_i) = \frac{b_i}{a_i+b_i}$, $\mathbb{E}(X_i^2) = \frac{a_i}{a_i+b_i}\frac{a_i+1}{a_i+b_i+1}$ for $1 \le i \le A$ and the single number 1 (for modeling the bias).

We set $A = 20$ and vary the budget B from 30 to 200 ($1.5A$ to $10A$). We run the stochastic subgradient method (Figure 1) with a learning rate of $\eta_t \propto 1/\sqrt{t}$, a uniform starting weight $w_1 = \frac{1}{d}\mathbf{1}$, and regularization parameter $H = 7$. To speed up convergence, we use the mini-batch strategy (Dekel et al., 2012) where each stochastic subgradient is estimated by averaging 200 independent stochastic subgradients, and ran the gradient descent for 2500 iterations. Additionally, we threshold $P_J(\cdot,\cdot)$ below by 0.

The left plot in Figure 2 compares the average posterior classification error of policies corresponding to 1) choosing every data instance uniformly (a.k.a. pure exploration), 2) the Opt-KG policy, and 3) the policy obtained from our stochastic subgradient method. For each policy, the posterior classification error was estimated by averaging the loss for 10,000 independent runs. Our algorithm outperforms both policies for every budget level, and generally achieves the same performance as Opt-KP with 10% fewer samples.

The right half of Figure 2 plots the (normalized) number of times each data instance is queried against the soft label of the instance θ_i (which is of course unknown to the policy). The plots are generated by averaging 2500 independent runs. As the budget is increased, proportionally more budget is spent on the harder instances, i.e. those with soft labels close to 0.5. However, even with as few as $B = 2.5A$ queries, the harder labels tend to be queried more often.

Acknowledgement

Xi Chen would like to thank Google Faculty Research Awards to support this work. We gratefully acknowledge the support of the Australian Research Council through the award of an Australian Laureate Fellowship (FL110100281), the NSF through grant CCF-1115788, and Adobe through a Digital Marketing Research Award.

References

Abbasi-Yadkori, Y., Bartlett, P., and Malek, A. Linear programming for large-scale Markov decision problems. In *Proceedings of the International Conference on Machine Learning*, 2014.

Bertsekas, D. P. *Dynamic Programming and Optimal Control*. Athena Scientific, 2007.

Bertsekas, D. P. and Tsitsiklis, J. *Neuro-Dynamic Programming*. Athena scientific optimization and computation series. Athena Scientific, 1996.

Chen, X., Lin, Q., and Zhou, D. Optimistic knowledge gradient for optimal budget allocation in crowdsourcing. In *Proceedings of the International Conference on Machine Learning*, 2013.

de Farias, D. P. and Van Roy, B. The linear programming approach to approximate dynamic programming. *Operations Research*, 51, 2003.

de Farias, D. P. and Van Roy, B. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29, 2004.

de Farias, D. P. and Van Roy, B. A cost-shaping linear program for average-cost approximate dynamic programming with performance guarantees. *Mathematics of Operations Research*, 31, 2006.

Dekel, Ofer, Gilad-Bachrach, Ran, Shamir, Ohad, and Xiao, Lin. Optimal distributed online prediction using mini-batches. *J. Mach. Learn. Res.*, 13(1), 2012.

Desai, V. V., Farias, V. F., and Moallemi, C. C. Approximate dynamic programming via a smoothed linear program. *Operations Research*, 60(3):655–674, 2012.

Duchi, John, Bartlett, Peter L, and Wainwright, Martin. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012.

Guestrin, C., Hauskrecht, M., and Kveton, B. Solving factored mdps with continuous and discrete variables. In *UAI*, 2004.

Hauskrecht, M. and Kveton, B. Linear program approximations to factored continuous-state markov decision processes. In *NIPS*, 2003.

Kushner, H. J. and Yin, G. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, 2003.

Maei, H. R., Szepesvári, Cs., Bhatnagar, S., Precup, D., Silver, D., and Sutton, R. S. Convergent temporal-difference learning with arbitrary smooth function approximation. In *NIPS*, 2009.

Maei, H. R., Szepesvári, Cs., Bhatnagar, S., and Sutton, R. S. Toward off-policy learning control with function approximation. In *ICML*, 2010.

Manne, A. S. Linear programming and sequential decisions. *Management Science*, 6(3):259–267, 1960.

Papadimitriou, C. and Tsitsiklis, J. N. The complexity of markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

Petrik, M. and Zilberstein, S. Constraint relaxation in approximate linear programs. In *ICML*, 2009.

Puterman, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

Schweitzer, P. and Seidmann, A. Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, 110: 568–582, 1985.

Shamir, O. and Zhang, T. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *ICML*, 2013.

Sra, S., Nowozin, S., and Wright, S. J. *Optimization for Machine Learning*. MIT Press, 2011.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. Bradford Book. MIT Press, 1998.

Sutton, R. S., Maei, H. R., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, Cs., and Wiewiora, E. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *ICML*, 2009a.

Sutton, R. S., Szepesvári, Cs., and Maei, H. R. A convergent O(n) algorithm for off-policy temporal-difference learning with linear function approximation. In *NIPS*, 2009b.

Sutton, Richard S. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

Todorov, E. Linearly-solvable markov decision problems. In *NIPS*, 2006.

Todorov, E. Efficient computation of optimal actions. *PNAS*, 106:1478–11483, 2009a.

Todorov, E. Eigenfunction approximation methods for linearly-solvable optimal control problems. In *ADPRL*, 2009b.

Tsitsiklis, John N. and Van Roy, Benjamin. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5): 674–690, 1997.

Veatch, M. H. Approximate linear programming for average cost MDPs. *Mathematics of Operations Research*, 38(3), 2013.

Zhong, M. and Todorov, E. Moving least-squares approximations for linearly-solvable optimal control problems. In *ADPRL*, 2011a.

Zhong, M. and Todorov, E. Aggregation methods for linearly-solvable MDPs. In *World Congress of the International Federation of Automatic Control*, 2011b.

Zinkevich, M., Weimer, M., Smola, A. J., and Li, L. Parallelized stochastic gradient descent. In *NIPS*, 2010.