Stochastic First-Order Method and Online Markov Decision Process

Mengdi Wang

ORFE@Princeton

Princeton IOS
March 18, 2016

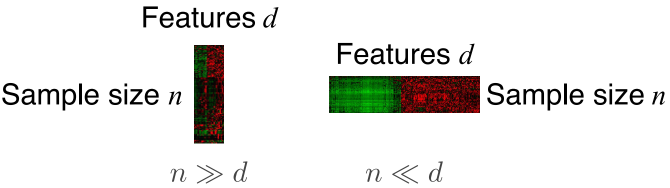
Outline

Motivation

- Machine learning is optimization

$$\min_{x \in \Re^d} \frac{1}{n} \sum_{i=1}^{n} \ell(x; A_i, b_i) + \rho(x)$$

- When $d \gg n$, need sparsity/low-rank regularization to achieve statistical consistency

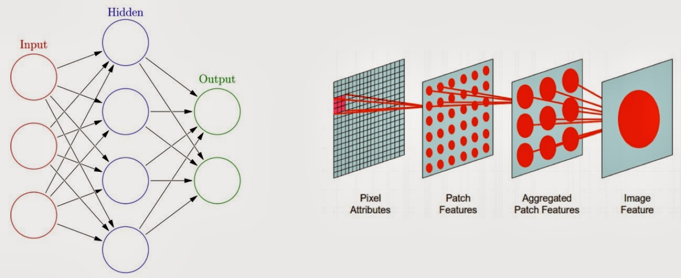


Motivation

- Machine learning is optimization

$$\min_{x \in \Re^d} \frac{1}{n} \sum_{i=1}^{n} \ell(x; A_i, b_i) + \rho(x)$$

- The objective could be highly non-convex, e.g., deep learning



Motivation

- Machine learning is optimization

$$\min_{x \in \Re^d} \frac{1}{n} \sum_{i=1}^{n} \ell(x; A_i, b_i) + \rho(x)$$

- Streaming data setting:

$$\min_{x \in \Re^d} \mathbf{E}_{A,b} \left[\ell(x; A, b) \right] + \rho(x)$$

- Online learning, empirical risk minimization, online principal component analysis, online MDP

Why Stochastic Gradient Descent?

- In both settings (batch and online), a practical algorithm needs to update using partial information (a small subset of all data)
- We have no other choice.

Stochastic first-order methods

The classical problem: $\min_x \mathbf{E}\left[f(x,\xi)\right]$

- Statistical learning
- Online learning
- Incremental algorithms
- Distributed algorithms
- Primal-dual algorithms (Mirror-Prox)
- Optimal first-order algorithms

The classical method: $x_{k+1} = x_k - \alpha \nabla f(x_k, \xi_k)$

stochastic gradient descent $\approx$online gradient
$\approx$ stochastic proximal $\approx$ stochastic primal-dual $\subset$ stochastic approximation

The classical result

Optimal error bounds given k samples:

- $\mathbf{E}\left[F(x_k) - F^*\right] = \mathcal{O}(1/\sqrt{k})$ for convex minimization
- $\mathbf{E}\left[F(x_k) - F^*\right] = \mathcal{O}(1/k)$ for strongly convex minimization

A Simplest Example

Consider the mean estimation problem

$$x^* = \mathbf{E}[\xi] = \mathrm{argmin}_x \mathbf{E}\left[\|x-\xi\|^2\right]$$

- When $\alpha_k = 1/k$, the stochastic gradient method is

$$x_{k+1} = x_k - \alpha_k(x_k - \xi_k) = (1-\frac{1}{k})x_k + \frac{1}{k}\xi_k = \frac{1}{k}\sum_{t=1}^{k}\xi_t$$

- The stochastic gradient iteration computes the empirical mean of $\xi_1, \ldots, \xi_k$
- By strong law of large numbers and central limit theorem, we have

$$x_k \xrightarrow{a.s.} x^*, \qquad \mathbf{E}\left[\|x_k - x^*\|^2\right] = \mathcal{O}(1/k), \qquad \mathit{Regret} = \mathcal{O}(\log k).$$

Interpretation: stochastic gradient method essentially updates a sufficient statistics x_k for estimating $x^* = \mathrm{argmin}_x \mathbf{E}[f(x,\xi)]$

When there are two entangled uncertainties (Wang et al., 2015)

Consider the problem

$$\min_{x \in \mathcal{X}} \left\{ F(x) = (f \circ g)(x) \right\},$$

where

$$f(y) = \mathbf{E}\left[f_v(y)\right], \qquad g(x) = \mathbf{E}\left[g_w(x)\right],$$

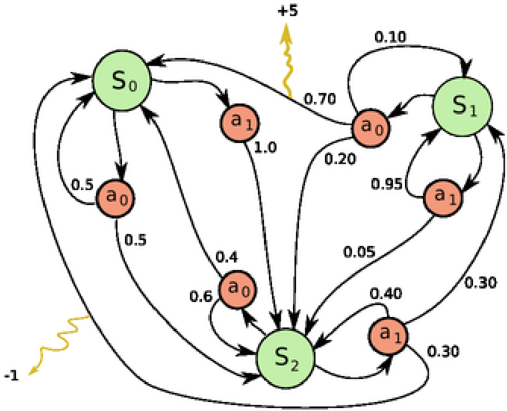
	General Convex	Strongly Convex
Non-Smooth	$\mathcal{O}(k^{-1/4})$	$\mathcal{O}(k^{-2/3})$
Smooth	$\mathcal{O}(k^{-2/7})$	$\mathcal{O}(k^{-4/5})$
$\min_x \mathbb{E}[g(x)]$	$\mathcal{O}(k^{-1/2})$	$\mathcal{O}(k^{-1})$

Figure : Summary of sample complexities.

Outline

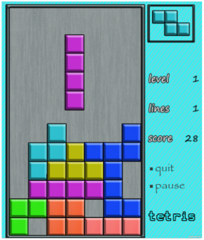
Markov Decision Process

Consider a controllable Markov chain

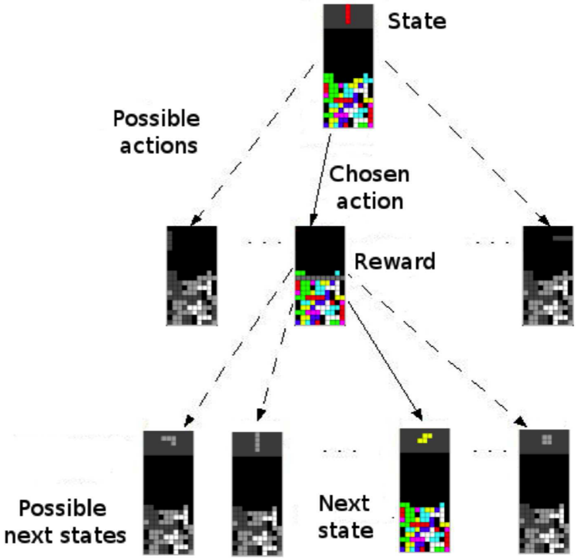


- State space $\mathcal{S} = \{S_1, \ldots, S_n\}$
- Action space $\mathcal{A} = \{a_1, \ldots, a_m\}$
- Transition probability matrix $P_a \in \Re^{n \times n}$ parameterized by actions $a \in \mathcal{A}$.
- Upon a state transition from i to j using action a, incurs a cost g_{ija} with second moment bounded by σ^2.

Applications



Tetris is a MDP



Optimal Policy and Optimal Value Function

Objective of MDP

The Markovian decision problem (MDP) is to find an optimal policy $\mu^*: \mathcal{S} \mapsto \mathcal{A}$ such that the infinite-horizon discounted cost is minimized, regardless of the initial state:

$$\mu^* = \operatorname{argmin}_{\mu: \mathcal{S} \mapsto \mathcal{A}} \mathbf{E}\left[\sum_{k=1}^{\infty} \alpha^k g_{i_k i_{k+1} \mu(i_k)}\right],$$

where $\alpha \in (0, 1)$ is a discount factor, $(i_0, i_1, \ldots)$ are state transitions generated by the Markov chain under policy μ, and the expectation is taken over the entire process.

Definition

Define the optimal cost vector $x^* \in \Re^{|\mathcal{S}|}$ to be

$$x^*(i) = \min_{\mu: \mathcal{S} \mapsto \mathcal{A}} \mathbf{E}\left[\sum_{k=1}^{\infty} \alpha^k g_{i_k i_{k+1} \mu(i_k)} \mid i_0 = i\right].$$

The value $x^*(i)$ is equal to the optimal expected total cost when the initial state is i. The optimal cost vector x^* is often regarded as the *optimal value function* or *optimal cost-to-go*.

Bellman Equation

According to DP theory, the vector x^* is the optimal cost vector if only if it solves the following non-linear fixed-point equation:

$$x^*(i) = \min_{a \in \mathcal{A}} \left\{ \alpha \sum_{j \in \mathcal{S}} P_a(i,j) x^*(j) + \sum_{j \in \mathcal{S}} P_a(i,j) \mathbf{E}\left[g_{ija} \mid i, j, a \right] \right\}, \quad i \in \mathcal{S},$$

A policy μ^* is an optimal policy if and only if it attains the minimization of the Bellman equation.

Remarks

- In the continuous-time analog of MDP, i.e., stochastic optimal control, the Bellman equation is the HJB
- Exact solution methods: value iteration, policy iteration, variational analysis
- What makes things hard:

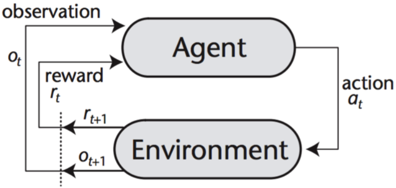
Curse of dimensionality + Modeling Uncertainty

Outline

Black-Box Model

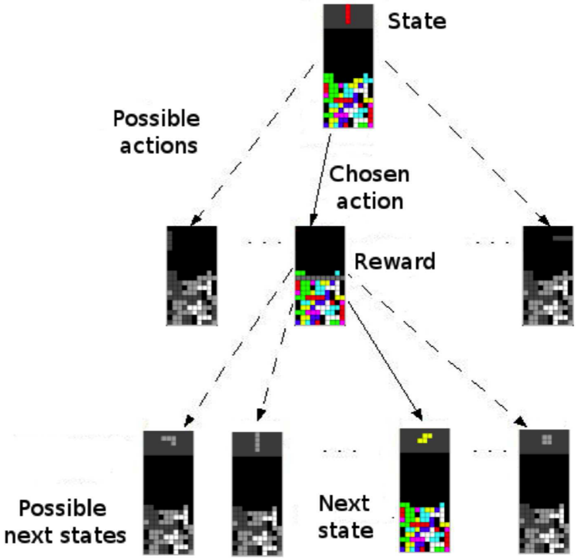
Assumption

Suppose that we do not know about the cost distribution and transition probabilities. Instead, we have a Simulation Oracle $\mathcal{M}$ that takes input (i, a) and generates state transition to j such that the next state j is chosen with probabilities $P_a(i, j)$.



- More examples of learning methods: Q-learning, Temporal difference learning, TD(λ), LSTD, cross-entropy method, actor-critic, active learning, etc ...
- These algorithms are essentially all combinations of DP, sampling, parametric models.
- DP + Online Learning + Feature Approximation $\approx$ Reinforcement Learning

Tetris is a MDP



Bellman Equation as LP

Bellman Equation as LP (Farias and Van Roy, 2003)

The Bellman equation is equivalent to

$$\begin{aligned} &\text{minimize} \;\; -e^T x \\ &\text{subject to} \;\; (I - \alpha P_a)\, x - g_a \leq 0, \qquad a \in \mathcal{A}, \end{aligned}$$

- Exact policy iteration is a form of simplex method and exhibits strongly polynomial performance (Ye 2011)
- Again, curse of dimensionality:
- Variable dimension $= |\mathcal{S}|$.
- Number of constraints $= |\mathcal{S}| \times |\mathcal{A}|$.

Duality between Value Function and Policy

Dual Problem

Let $\lambda_{i,a} > 0$ be the multiplier associated with the ith row of the primal constraint $\alpha P_a x + g_a \geq x$. The dual problem is

$$\begin{aligned} \text{maximize} \quad & -\sum_{a \in \mathcal{A}} \lambda_a^T g_a \\ \text{subject to} \quad & \sum_{a \in \mathcal{A}} \left(I - \alpha P_a^T \right) \lambda_a = e, \qquad \lambda_a \geq 0, \end{aligned}$$

where the dual variable is high-dimensional $\lambda = (\lambda_a)_{a \in \mathcal{A}} \in \Re^{|\mathcal{S}||\mathcal{A}|}$.

Theorem

The optimal dual solution $\lambda^* = (\lambda^*_{i,a})_{i \in \mathcal{S}, a \in \mathcal{A}}$ is sparse and has exact $|S|$ nonzeros. It satisfies

$$\left(\lambda^*_{i,\mu^*(i)}\right)_{i \in \mathcal{S}} = (I - \alpha P_{\mu^*}^T)^{-1} e,$$

and $\lambda^*_{i,a} = 0$ if $a \neq \mu^*(i)$.

Finding the optimal policy μ^* = Finding the basis of the dual solution λ^*

Stochastic primal-dual (value-policy) algorithm

- **Input:** Simulation Oracle $\mathcal{M}$, $n = |\mathcal{S}|$, $m = |\mathcal{A}|$, $\alpha \in (0,1)$.
- Initialize $x^{(0)}$ and $\lambda = (\lambda_u^{(0)} : u \in \mathcal{A})$ arbitrarily.
- For$k = 1, 2, \ldots, T$
    - Sample i_k uniformly from $\mathcal{S}$ and sample u_k uniformly from $\mathcal{A}$.
    - Sample next state j_k and immediate reward $g_{i_k j_k u_k}$ conditioned on (i_k, u_k) from $\mathcal{M}$.
    - Update the iterates by

$$x^{(k-\frac{1}{2})} = x^{(k-1)} - \gamma_k\Big(- e + m\lambda_{u_k}^{(k-1)} - \alpha mn \left(\lambda_{u_k}^{(k-1)} \cdot e_{i_k}\right) e_{j_k}\Big),$$

$$\lambda_{u_k}^{(k-\frac{1}{2})} = \lambda_{u_k}^{(k-1)} + m\gamma_k\Big(x^{(k-1)} - \alpha n \left(x^{(k-1)} \cdot e_{j_k}\right) e_{i_k} - ng_{i_k j_k u_k} e_{i_k}\Big),$$

$$\lambda_u^{(k-\frac{1}{2})} = \lambda_u^{(k-1)}, \qquad \forall\ u \neq u_k,$$

    - Project the iterates orthogonally to some regularization constraints

$$x^{(k)} = \Pi_X x^{(k-\frac{1}{2})}, \qquad \lambda^{(k)} = \Pi_\Lambda \lambda^{(k-\frac{1}{2})}.$$

- **Ouput:** Averaged dual iterate $\hat{\lambda} = \frac{1}{T}\sum_{k=1}^{T} \lambda^{(k)}$

Dual Variable as a Randomized Policy

Let the randomized policy $\hat{\mu}$ be such that

$$\mathbf{P}(\hat{\mu}(i) = a) = \frac{\hat{\lambda}_{a,i}}{\sum_{i=1}^{n} \hat{\lambda}_{a,i}}.$$

Theorem (Near-Optimality of Randomized Policy (Wang 2016))

Let $\hat{\mu}$ be generated by Algorithm 1 using T queries to the oracle $\mathcal{M}$, and let $x_{\hat{\mu}}$ be the cost function under policy $\hat{\mu}$, i.e.,

$$x_{\hat{\mu}}(i) = \mathbf{E}\left[\sum_{k=1}^{\infty} \alpha^k g_{i_k i_{k+1} \hat{\mu}(i_k)} \mid i_0 = i\right],$$

where $(i_0, i_1, \ldots)$ are generated by the Markov chain with transition matrix $P_{\hat{\mu}}$. Comparing the cost function of $\hat{\mu}$ and the optimal cost function, the suboptimality of $\hat{\mu}$ satisfies

$$\frac{\mathbf{E}\left[\|x_{\hat{\mu}} - x^*\|_\infty\right]}{\|x^*\|_\infty} \leq \mathcal{O}\left(\frac{|\mathcal{S}|^2 |\mathcal{A}|}{(1-\alpha)^2 \sqrt{T}}\right).$$

This rate is nearly-optimal and non-improvable w.r.t. sample size T.

Recovery of Optimal Policy

Definition

We define the minimal action discrimination constant as the minimal efficiency loss of deviating from the optimal policy μ^* by making a single wrong action. It is given by

$$\bar{d} = \min_{(i,a):\mu^*(i)\neq a} (\alpha P_{a,i} x^* + g_a(i) - x^*(i)).$$

- When there exists a unique optimal policy μ^*, therefore $\bar{d} > 0$.
- A large value of $\bar{d}$ means that it is easy to discriminate optimal state-actions from suboptimal state actions. A small value of $\bar{d}$ means that some suboptimal actions perform similarly to optimal actions.
- The constant $\bar{d}$ measures how hard it is to discriminate suboptimal policies from the optimal policy.

Recovery of Optimal Policy

Recall that we only care about the support of the dual variable.
Idea: Rounding the dual iterate $\hat{\lambda}$ to the nearest extreme point solution.

Theorem (Recovering Optimal Policy By Truncation)

Let $\hat{\mu}_\delta^{Tr}$ be the truncated pure policy such that $\hat{\mu}_\delta^{Tr}(i) = argmax_{a \in \mathcal{A}} \hat{\lambda}_{i,a}$ for all $i \in \mathcal{S}$. Then

$$\mathbf{P}\left(\hat{\mu}_\delta^{Tr} = \mu^*\right) \geq 1 - \mathcal{O}\left(\frac{|\mathcal{S}|^2 |\mathcal{A}|^2 (1+\sigma^2)}{\bar{d}(1-\alpha)^2 \sqrt{T}}\right).$$

Good news: Without accurate knowledge of value functions, we can recover the exact optimal policy with high probability

Remarks: Learning vs. Optimization

- In online MDP, the LP geometry is yet to be fully exploited.
- Difference between statistical goal and and optimization goal.
- Deterministic optimization model sometimes does not work.

An Example: Gap between statistical goal and optimization hardness

Statisticians like the ℓ_0-regularized optimization problem

$$\hat{x}_k = \text{argmin}_{x \in \Re^d} \frac{1}{k} \sum_{i=1}^{k} \ell(x; a_i, b_i) + \lambda_k \|x\|_0$$

The purpose of ℓ_0 regularization is to achieve the optimal statistical error

$$\mathcal{O}\left(\frac{\log d}{k}\right)$$

Hardness of Optimization (Chen and Wang, 2015)

Finding an ϵ-optimal solution with $\epsilon = \frac{d^\delta}{k}$ is strongly NP-hard, for all $\delta \in (0, 1)$.

Hardness of approximation within statistical error.

Remarks: Learning vs. Optimization

- In online MDP, the LP geometry is yet to be fully exploited.
- Difference between learning goal and and optimization goal.
- Deterministic optimization model sometimes does not work.

What should be the goal?

- Solving optimization problem?
- Estimating/approximating the optimal solution of the “true” problem?

Thank you very much!