

Cost Function Approximation Methods in Dynamic Programming and Extensions

H. Yu* D. P. Bertsekas**

*Department of Computer Science
University of Helsinki

** Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology

INFORMS, Seattle, 2007

Overview of Cost Approximation Methods

Projected Bellman Equation and TD Methods

LS Algorithms using Equation/Mapping Approximation

Extension I: LS Q-Learning for Optimal Stopping Problems

Background

Least Squares Q-Learning Algorithm

Variants with Reduced Computation

Extension II: Solving Linear Equations

Key Ideas

Single Step and Multistep Algorithms

Related Ideas: Basis/Feature Generation

Outline

Overview of Cost Approximation Methods

Projected Bellman Equation and TD Methods

LS Algorithms using Equation/Mapping Approximation

Extension I: LS Q-Learning for Optimal Stopping Problems

Background

Least Squares Q-Learning Algorithm

Variants with Reduced Computation

Extension II: Solving Linear Equations

Key Ideas

Single Step and Multistep Algorithms

Related Ideas: Basis/Feature Generation

Cost Approximation Methods/Single Policy

Bellman Equation for Single Policy in Finite-State MDP:

$$J = g + \alpha PJ, \quad \alpha \in (0, 1]$$

We approximate J by Φr

$$J \approx \Phi r$$

Columns of Φ are basis functions

Algorithms:

- ▶ Direct approach
- ▶ Equation error methods
- ▶ TD methods

Alternatives: Q-learning, including for multiple policies; approximate LP

Projected Bellman Equation and TD Methods

Bellman Equation:

$$J = TJ = g + \alpha PJ, \quad \alpha \in (0, 1]$$

TD(λ) with Function Approximation: approximate the solution by solving *projected Bellman equation*

$$\Phi r = \Pi T^{(\lambda)}(\Phi r), \quad T^{(\lambda)} = (1 - \lambda) \sum_{k=0}^{\infty} \lambda^k T^{k+1}, \quad \lambda \in [0, 1]$$

- ▶ Weighted Euclidean projection

$$\Pi J = \arg \min_{r \in \mathbb{R}^s} \|J - \Phi r\|_{\xi}$$

(ξ_1, \dots, ξ_n): invariant distribution of P

- ▶ Contraction and error bounds

Algorithms: Recursive TD vs. LS

Both recursive TD and LS algorithms solve, in the limit,

$$\Phi r = \Pi T^{(\lambda)}(\Phi r)$$

using a single sample trajectory (i_0, i_1, \dots) (or multiple ones)

Recursive TD(λ) (Sutton '88):

- ▶ Use each sample state only once

$$r_{t+1} = r_t + \gamma_t \mathbf{z}_t (\mathbf{g}_t + \alpha \phi(i_{t+1})' r_t - \phi(i_t)' r_t)$$

- ▶ Stochastic approximation, small stepsize for averaging out noise
- ▶ Analyzed by Tsitsiklis and Van Roy '97, '99, and by others

Methods with Least Squares Forms:

- ▶ Reuse the samples at every iteration
- ▶ Implicitly approximate $\Pi T^{(\lambda)}$, restricted on the approximation subspace, with increasing accuracy

LS Algorithms: Equation/Mapping Approximation

Least Squares Temporal Difference (LSTD)

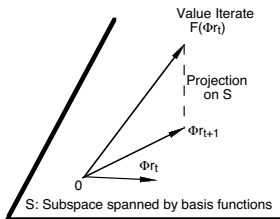
- ▶ Proposed by Bradtke and Barto '96, Boyan '99
- ▶ Solve approximate projected Bellman equation

$$\Phi \tilde{r}_{t+1} = \hat{\Pi}_t \hat{T}_t^{(\lambda)}(\Phi \tilde{r}_{t+1})$$

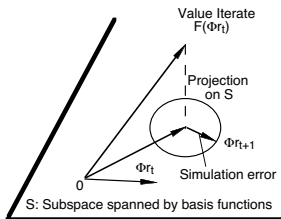
Least Squares Policy Evaluation (LSPE)

- ▶ Proposed by Bertsekas and Ioffe '96 and analyzed by several others
- ▶ Approximate projected value iteration

$$\Phi r_{t+1} = \hat{\Pi}_t \hat{T}_t^{(\lambda)}(\Phi r_t) = \Pi T^{(\lambda)}(\Phi r_t) + \epsilon_t$$



Projected Value Iteration



Least Squares Policy Evaluation (LSPE)

Convergence Analysis of LSPE

Convergence of LSPE with constant stepsize for $\alpha \in (0, 1]$
(Bertsekas, Borkar, and Nedić '04, Yu and Bertsekas '06):

- ▶ Iteration can be viewed as “deterministic portion + stochastic noise”

$$\Phi r_{t+1} = F(\Phi r) + \epsilon_t, \quad \text{where } F = (1 - \gamma)I + \gamma \Pi T^{(\lambda)}$$

- ▶ The constant stepsize γ plays a role of “damping” the projected value iteration $\Pi T^{(\lambda)}$
- ▶ If the deterministic portion F is a contraction, then noise diminishes asymptotically, and LSPE converges

Proposition

Convergence stepsize ranges for LSPE:

(i) *discounted case* ($\alpha < 1$): $\gamma \in (0, \frac{2}{1+\alpha})$

(ii) *average cost case* ($\alpha = 1$):

$$\lambda \in (0, 1) : \quad \gamma \in (0, 1]$$

$$\lambda = 0, P \text{ aperiodic} : \quad \gamma \in (0, 1], \quad \lambda = 0 : \quad \gamma \in (0, 1)$$

Convergence Analysis of LSPE

Asymptotically Optimal Rate of Convergence (Yu and Bertsekas '06)
shown by a comparison analysis of LSPE/LSTD:

$$\text{LSPE:} \quad \Phi r_{t+1} = (1 - \gamma)\Phi r_t + \gamma \widehat{\Pi}_t \widehat{T}_t^{(\lambda)}(\Phi r_t)$$

$$\text{LSTD:} \quad \Phi \tilde{r}_{t+1} = \widehat{\Pi}_t \widehat{T}_t^{(\lambda)}(\Phi \tilde{r}_{t+1})$$

Proposition

For all stepsize γ in the convergence range of LSPE,

$$t \|\Phi r_t - \Phi \tilde{r}_t\| < \infty, \quad w.p.1.$$

Implications:

- ▶ Empirical phenomenon: r_t “tracks” \tilde{r}_t
- ▶ More precisely: $r_t - \tilde{r}_t \rightarrow 0$ at the rate of $O(t)$, faster than $r_t, \tilde{r}_t \rightarrow r^*$ at the rate of $O(\sqrt{t})$
- ▶ LSTD has the optimal convergence rate comparing to recursive TD (Konda '02), so LSPE shares this property

Outline

Overview of Cost Approximation Methods

Projected Bellman Equation and TD Methods

LS Algorithms using Equation/Mapping Approximation

Extension I: LS Q-Learning for Optimal Stopping Problems

Background

Least Squares Q-Learning Algorithm

Variants with Reduced Computation

Extension II: Solving Linear Equations

Key Ideas

Single Step and Multistep Algorithms

Related Ideas: Basis/Feature Generation

Stopping Problems – Overview

Extension to Optimal Stopping Problems – Main Idea:

- ▶ Apply the same mapping approximation idea to solve the non-linear projected Bellman equation

Basic Optimal Stopping Problem:

- ▶ An irreducible Markov chain with n states and transition matrix P
 Action: stop or continue
 Cost at state i : $c(i)$ if stop; $g(i)$ if continue
 Minimize the expected discounted total cost till stop
- ▶ Bellman equations for $\alpha \in (0, 1)$

$$J^* = \min\{c, g + \alpha P J^*\}, \quad Q^* = g + \alpha P \min\{c, Q^*\}$$

Q-factor: $Q^*(i, a) =$ optimal cost from state i with first action a

Optimal: stop as soon as the state hits the set

$$\mathcal{D} = \{i \mid c(i) \leq Q^*(i)\}$$

Q-Learning with Function Approximation

(Tsitsiklis and Van Roy '99)

Subspace Approximation

$$\tilde{Q} = \Phi r \quad \text{or,} \quad \tilde{Q}(i, r) = \phi(i)' r$$

Weighted Euclidean Projection

$$\Pi Q = \arg \min_{r \in \mathbb{R}^s} \|Q - \Phi r\|_{\xi}, \quad (\xi_1, \dots, \xi_n) : \text{invariant distribution of } P$$

Key Fact: DP mapping F is $\|\cdot\|_{\xi}$ -contraction and so is ΠF

$$FQ \stackrel{\text{def}}{=} g + \alpha P \min\{c, Q\}$$

Projected Bellman Equation solvable by using TD:

$$\Phi r^* = \Pi F(\Phi r^*)$$

Suboptimality: stop as soon as the state hits $\{i \mid c(i) \leq \phi(i)' r^*\}$

$$\sum_{i=1}^n \xi(i) (J_{\mu}(i) - J^*(i)) \leq \frac{2}{(1-\alpha)\sqrt{1-\alpha^2}} \|\Pi Q^* - Q^*\|_{\xi}$$

Least Squares Q-Learning

The Algorithm

(i_0, i_1, \dots) unstopped state process, $\gamma \in (0, \frac{2}{1+\alpha})$ constant stepsize

$$r_{t+1} = r_t + \gamma(\hat{r}_{t+1} - r_t)$$

where \hat{r}_{t+1} is the LS solution:

$$\hat{r}_{t+1} = \arg \min_{r \in \mathbb{R}^s} \sum_{k=0}^t \left(\phi(i_k)' r - g(i_k, i_{k+1}) - \alpha \min \{ c(i_{k+1}), \phi(i_{k+1})' r_t \} \right)^2$$

Proposition

For all $\gamma \in \left(0, \frac{2}{1+\alpha}\right)$, $\lim_{t \rightarrow \infty} r_t = r^*$, w.p.1.

Computation and Interpretation

The algorithm can be viewed as approximate projected value iteration

$$\hat{r}_{t+1} = \left(\sum_{k=0}^t \phi(i_k) \phi(i_k)' \right)^{-1} \sum_{k=0}^t \phi(i_k) \left(g(i_k, i_{k+1}) + \alpha \min \{ c(i_{k+1}), \phi(i_{k+1})' r_t \} \right)$$

$$\Phi \hat{r}_{t+1} = \hat{\Pi}_t \hat{F}_t(\Phi r_t) = \hat{\Pi}_t \left(\hat{g}_t + \alpha \tilde{P}_t \min \{ c, \Phi r_t \} \right)$$

$$\Phi r_{t+1} = (1 - \gamma) \Phi r_t + \gamma \hat{\Pi}_t \hat{F}_t(\Phi r_t)$$

- ▶ Convergence follows from mapping approximation and contraction: w.p.1, for all t sufficiently large,
 - $\hat{\Pi}_t \hat{F}_t$ is $\| \cdot \|_{\xi}$ -contraction with modulus $\hat{\alpha} \in (\alpha, 1)$
 - $(1 - \gamma)I + \gamma \hat{\Pi}_t \hat{F}_t$ is $\| \cdot \|_{\xi}$ -contraction for $\gamma \in (0, \frac{2}{1+\alpha})$
- ▶ Computation overhead: require repartitioning past states into stopping or continuation sets

Comparison to an LSTD Analogue

$$\text{LS Q-learning:} \quad \Phi r_{t+1} = (1 - \gamma)\Phi r_t + \gamma \hat{\Pi}_t \hat{F}_t(\Phi r_t) \quad (1)$$

$$\text{LSTD analogue:} \quad \Phi \tilde{r}_{t+1} = \hat{\Pi}_t \hat{F}_t(\Phi \tilde{r}_{t+1}) \quad (2)$$

Eq. (1) is one *single* fixed point iteration for solving Eq. (2).

Yet, they have the *same* convergence rate [two-time scale argument, similar to the comparison analysis of LSPE/LSTD]:

Proposition

$$\text{For all } \gamma \in \left(0, \frac{2}{1 + \alpha}\right), \quad t \|\Phi r_t - \Phi \tilde{r}_t\| < \infty, \quad w.p.1.$$

Variants

A Variant: repartition each sample state at most m times

$$r_{t+1} = \arg \min_{r \in \mathfrak{R}^s} \sum_{k=0}^t \left(\phi(i_k)' r - g(i_k, i_{k+1}) - \alpha \min \{ c(i_{k+1}), \phi(i_{k+1})' r_{k,t} \} \right)^2$$

Intermediate between TD and mapping approximation:

- ▶ $m \rightarrow \infty$: LS Q-learning algorithm
- ▶ $m = 1$: the fixed point Kalman filter (Choi and Van Roy '06), equivalent to TD with scaling,

$$r_{t+1} = r_t + \frac{1}{t+1} B_t^{-1} \phi(i_t) (g(i_t, i_{t+1}) + \alpha \min \{ c(i_{t+1}), \phi(i_{t+1})' r_t \} - \phi(i_t)' r_t)$$

Convergence for $m \geq 1$ can be shown, using o.d.e. analysis (Borkar '06, Borkar and Meyn '01), or an alternative “direct” proof.

Extensions to undiscounted optimal stopping problems

Outline

Overview of Cost Approximation Methods

Projected Bellman Equation and TD Methods

LS Algorithms using Equation/Mapping Approximation

Extension I: LS Q-Learning for Optimal Stopping Problems

Background

Least Squares Q-Learning Algorithm

Variants with Reduced Computation

Extension II: Solving Linear Equations

Key Ideas

Single Step and Multistep Algorithms

Related Ideas: Basis/Feature Generation

Overview and Key Ideas from ADP

Large Systems of Linear Equations

$$x = T(x) = b + Ax$$

Ideas Inspired by ADP:

- ▶ Subspace approximation by solving *projected equation*

$$x = \Pi T(x) \quad \text{i.e.,} \quad \Phi r = \Pi T(\Phi r) = \Pi b + \Pi A(\Phi r)$$

- ▶ Avoid matrix-vector product by sampling entries of A
- ▶ Weighted sampling with an artificial Markov chain can be used to construct implicitly a projection

Overall Characteristics:

- ▶ Need only low-order calculation
- ▶ Equation/mapping approximation-based LS algorithms applicable
- ▶ Error bounds analogous to those in ADP, if e.g., ΠT is a contraction
- ▶ Multistep methods, such as analogues of $TD(\lambda)$, are as easy to implement using simulation as the single step method

Single Step Version

Solve the projected equation $x = \Pi T(x)$, i.e.,

$$r = \left(\sum_{i=1}^n \xi_i \phi(i) \phi(i)' \right)^{-1} \sum_{i=1}^n \xi_i \phi(i) \left(b_i + \sum_{j=1}^n a_{ij} \phi(j)' r \right)$$

Sampling Schemes

- ▶ generate row and column-index pairs $\{(i_0, j_0), (i_1, j_1), \dots\}$ independently such that $i_t \sim \xi$, $j_t \sim i_t$ -th row of P ; or
- ▶ generate (i_0, i_1, \dots) using a Markov chain P (ξ need not be known)

Desirable Properties of P : $a_{ij} \neq 0 \Rightarrow p_{ij} \neq 0$, no transient states, fast mixing, easy to simulate

Equation Approximation Method (LSTD analogue): solve

$$r_{t+1} = \left(\sum_{k=0}^t \phi(i_k) \phi(i_k)' \right)^{-1} \sum_{k=0}^t \phi(i_k) \left(b_{i_k} + \frac{a_{i_k j_k}}{p_{i_k j_k}} \phi(j_k)' r_{t+1} \right)$$

Projected Jacobi Method (LSPE analogue)

Constructing Markov Chain and Projection

Generalization of DP

If ΠT or its damped version is a contraction, then

- ▶ Error bound analogous to that in DP is available
- ▶ Projected Jacobi method is also applicable

Generalization of Discounted/Undiscounted Cases in DP:

Proposition

Assume $|A| \leq P$ for some irreducible P , and ξ is such that $\xi' P = \xi'$.

- (i) ΠT is contraction with respect to some norm if $\sum_{i,j} |a_{ij}| < 1$ for some i .
- (ii) $\Pi((1 - \gamma)I + \gamma T)$, $\gamma \in (0, 1)$ is $\|\cdot\|_{\xi}$ -contraction if $(I - \Pi A)$ is invertible.

- ▶ Example of $|A| \leq P$: $Cx = d$, C weakly diagonally dominant, i.e., $\sum_{j \neq i} |c_{ij}| \leq |c_{ii}| \neq 0$
- ▶ Extend to solving non-linear fixed point equations with certain separable structure and contraction/non-expansive property (such as in optimal stopping problems)

Multistep Versions

Solve projected multistep versions of the fixed point equation

$$x = \Pi T^l(x), \quad x = \Pi T^{(\lambda)}(x)$$

$$T^l = A^l x + \sum_{m=0}^{l-1} A^m b, \quad T^{(\lambda)} = (1 - \lambda) \sum_{k=0}^{\infty} \lambda^k T^{k+1}$$

Sampling using a Markov chain gives recursive update formulas

- ▶ Intuition: $i_0^m = (i_0, i_1, \dots, i_m)$ with $i_0 = i$; for any vector v , just as

$$E\{v(i_m) \mid i_0 = i\} = (P^m v)(i), \quad E\{w_m(i_0^m) v(i_m) \mid i_0 = i\} = (A^m v)(i)$$

where $w_m(i_0^m)$ is a properly defined weight

$$w_m(i_0^m) = \frac{a_{i_0 i_1}}{p_{i_0 i_1}} \frac{a_{i_1 i_2}}{p_{i_1 i_2}} \dots \frac{a_{i_{m-1} i_m}}{p_{i_{m-1} i_m}}.$$

Implementation is as easy as in the single step case, because of the product structure of w_m .

Example: Multistep Algorithm Analogous to TD(λ)

Assume $(I - A)$ invertible and $\sigma(A) \leq 1$; solve

$$x = \Pi T^{(\lambda)}(x), \quad \lambda \in (0, 1)$$

- ▶ $\Pi T^{(\lambda)}$ is contraction for λ sufficiently close to 1 for all Π
- ▶ Solution of projected equation approaches Πx^* as $\lambda \rightarrow 1$; however, λ affects the variance as in DP

Update formulas resulting from using temporal difference expressions:

$$\text{LSPE: } r_{t+1} = r_t + \gamma B_t^{-1} (C_t r_t + h_t), \quad \text{LSTD: } \tilde{r}_{t+1} = -C_t^{-1} h_t$$

$$B_t = B_{t-1} + \phi(i_t)\phi(i_t)', \quad C_t = C_{t-1} + z_t \left(\frac{a_{i_t i_{t+1}}}{p_{i_t i_{t+1}}} \phi(i_{t+1}) - \phi(i_t) \right)'$$

$$h_t = h_{t-1} + z_t b_{i_t}, \quad z_t = \lambda \frac{a_{i_{t-1} i_t}}{p_{i_{t-1} i_t}} z_{t-1} + \phi(i_t)$$

- ▶ DP is a special case with $a_{i_t i_{t+1}}/p_{i_t i_{t+1}} = \alpha, \alpha \in (0, 1]$
- ▶ Analogue of recursive TD is also applicable

Generating Basis/Feature by Sampling

Consider using basis functions of the form

$$B^m g$$

- ▶ Reminiscent of Krylov subspace methods
- ▶ In the context of DP, $B^m g$ can be the finite-stage costs of some base policy

Implementation:

- ▶ For the simulation-based algorithms, one unbiased sample of $\phi(i)$ is sufficient
- ▶ So we only need to use an additional sampling procedure, *independent* of the one that generates $(i_0, i_1, \dots, i_t, \dots)$, to generate an unbiased estimate of $(B^m g)(i_t)$

Questions and Discussion

- ▶ Non-linear fixed point equations
to what extent the mapping approximation approach can be applied
- ▶ Linear equations
in the context of applications, how does the proposed method
compare with commonly used numerical methods
- ▶ Approximate dynamic programming
methods of basis selection and exploration

References

For a detailed presentation and analysis see:

- ▶ On convergence (average cost) and rate of convergence of LSPE
H. Yu and D. P. Bertsekas. "Convergence Results for Some Temporal Difference Methods Based on Least Squares," LIDS report 2697, MIT, 2006; revised 2007.
- ▶ On optimal stopping problems
H. Yu and D. P. Bertsekas. "A Least Squares Q-Learning Algorithm for Optimal Stopping Problems," LIDS report 2731, MIT, 2006; revised 2007. A shorter version appears in ECC'07.
- ▶ On solving linear equations
D. P. Bertsekas and H. Yu. "Solution of Large Systems of Equations using Approximate Dynamic Programming Methods," LIDS report 2754, MIT, 2007.

Available from

- ▶ Janey's web site: <http://cs.helsinki.fi/u/hyu/>
- ▶ Dimitri's web site: <http://web.mit.edu/dimitrib/www/home.html>