Reinforcement Learning: Approximate Dynamic Programming

Decision Making Under Uncertainty, Chapter 10

Christos Dimitrakakis

Chalmers

November 21, 2013

Definition 1 ($\boldsymbol{u}$-greedy policy and value function)

$$\pi^*_{\boldsymbol{u}} \in \arg\max_{\pi} \mathscr{L}_\pi \boldsymbol{u}, \qquad \boldsymbol{v}^*_{\boldsymbol{u}} = \mathscr{L} \boldsymbol{u}, \tag{1.1}$$

where $\pi : \mathcal{S} \to \mathfrak{D}(\mathcal{A})$ maps from states to action distributions.

Parametric value function estimation

$$\mathcal{V}_\Theta = \{\boldsymbol{v}_\theta \mid \theta \in \Theta\}, \qquad \theta^* \in \arg\min_{\theta \in \Theta} \|\boldsymbol{v}_\theta - \boldsymbol{u}\|_\phi \tag{1.2}$$

where $\| \cdot \|_\phi \triangleq \int_{\mathcal{S}} | \cdot | \, \mathrm{d}\phi$.

Parametric policy estimation

$$\Pi_\Theta = \{\pi_\theta \mid \theta \in \Theta\}, \qquad \theta^* \in \arg\min_{\theta \in \Theta} \|\pi_\theta - \pi^*_{\boldsymbol{u}}\|_\phi \tag{1.3}$$

where $\pi^*_{\boldsymbol{u}} = \arg\max_{\pi \in \Pi} \mathscr{L}_\pi \boldsymbol{u}$

Theorem 2

Consider a finite MDP μ with discount factor $\gamma < 1$ and a vector $\boldsymbol{u} \in \mathcal{V}$ such that $\left\| \boldsymbol{u} - V^*_\mu \right\|_\infty = \epsilon$. If π is the $\boldsymbol{u}$-greedy policy then

$$\left\| V^\pi_\mu - V^*_\mu \right\|_\infty \leq \frac{2\gamma\epsilon}{1-\gamma}.$$

In addition, $\exists \epsilon_0 > 0$ s.t. if $\epsilon < \epsilon_0$, then π is optimal.

Feature mapping $f : \mathcal{S} \times \mathcal{A} \to \mathcal{X}$.

For $\mathcal{X} \subset \mathbb{R}^n$, the feature mapping can be written in vector form:

$$f(s,a) = \begin{bmatrix} f_1(s,a) \\ \dots \\ f_n(s,a) \end{bmatrix} \tag{1.4}$$

Example 3 (Radial Basis Functions)

Let d be a metric on $\mathcal{S} \times \mathcal{A}$ and $\{(s_i, a_i) \mid i = 1, \dots, n\}$. Then we define each element of f as:

$$f_i(s,a) \triangleq \exp\{-d[(s,a),(s_i,a_i)]\}. \tag{1.5}$$

These function are sometimes called *kernels*.

Example 4 (Tilings)

Let $\mathscr{G} = \{X_1, \dots, X_n\}$ be a partition of $\mathcal{S} \times \mathcal{A}$ of size n. Then:

$$f_i(s,a) \triangleq \mathbb{I}\{(s,a) \in X_i\}. \tag{1.6}$$

Approximate policy ieration

Algorithm 1 Generic approximate policy iteration algorithm

input Initial value function $\boldsymbol{v}_0$, approximate Bellman operator $\hat{\mathscr{L}}$, approximate value estimator $\hat{V}$.

for $k = 1, \ldots$ **do**

$\pi_k = \arg\min_{\pi \in \hat{\Pi}} \left\| \hat{\mathscr{L}}_\pi \boldsymbol{v}_{k-1} - \mathscr{L} \boldsymbol{v}_{k-1} \right\|$ `// policy improvement`

$\boldsymbol{v}_k = \arg\min_{\boldsymbol{v} \in \hat{\mathcal{V}}} \| \boldsymbol{v} - V_\mu^{\pi_k} \|$ `// policy evaluation`

end for

Theoretical gurantees

Assumption 1

Consider a discounted problem with discount factor γ and iterates $\boldsymbol{v}_k, \pi_k$ such that:

$$\left\| \boldsymbol{v}_k - V^{\pi_k} \right\|_\infty \leq \epsilon, \qquad \forall k \tag{2.1}$$

$$\left\| \mathscr{L}_{\pi_{k+1}} \boldsymbol{v}_k - \mathscr{L} \boldsymbol{v}_k \right\|_\infty \leq \delta, \qquad \forall k \tag{2.2}$$

Theorem 5 ([6], proposition 6.2)

Under Assumption 1

$$\limsup_{k \to \infty} \| V^{\pi_k} - V^* \|_\infty \leq \frac{\delta + 2\gamma\epsilon}{(1-\gamma)^2}. \tag{2.3}$$

Lookahead policies

Single-step lookahead

$$\pi_{\boldsymbol{q}}(a \mid i) > 0 \qquad \text{iff } a \in \arg\max_{a' \in \mathcal{A}} q(i, a') \tag{2.4}$$

$$q(i, a) \triangleq r_\mu(i, a) + \gamma \sum_{j \in \mathcal{S}} P_\mu(j \mid i, a)\boldsymbol{u}(j). \tag{2.5}$$

T-step lookahead

$$\pi(i; \boldsymbol{q}_T) = \arg\max_{a \in \mathcal{A}} \boldsymbol{q}_T(i, a), \tag{2.6}$$

where $\boldsymbol{u}_k$ is recursively defined as:

$$q_k(i, a) = r_\mu(i, a) + \gamma \sum_{j \in \mathcal{S}} P_\mu(j \mid i, a)\boldsymbol{u}_{k-1}(j) \tag{2.7}$$

$$\boldsymbol{u}_k(i) = \max\{q_k(i, a) \mid a \in \mathcal{A}\} \tag{2.8}$$

and $\boldsymbol{u}_0 = \boldsymbol{u}$

Rollout policies

Rollout estimate of the q-factor

$$q(i, a) = \frac{1}{K_i} \sum_{k=1}^{K_i} \sum_{t=0}^{T_k - 1} r(s_{t,k}, a_{t,k}),$$

where $s_{t,k}, a_{t,k} \sim \mathbb{P}^{\pi}_{\mu}(\cdot \mid s_0 = i, a_0 = a)$, and $T_k \sim \mathcal{G}eom(1 - \gamma)$.

Rollout policy estimation.

Given a set of samples $q(i, a)$ for $i \in \hat{S}$, we estimate

$$\min_{\theta} \left\| \pi_{\theta} - \pi^*_{q} \right\|_{\phi},$$

for some ϕ on $\hat{S}$.

Generalised linear model using features (or kernel)

Feature mapping $f : \mathcal{S} \to \mathbb{R}^n$, parameters $\boldsymbol{\theta} \in \mathbb{R}^n$.

$$v_{\boldsymbol{\theta}}(s) = \sum_{i=1}^{n} \theta_i f_i(s) \tag{2.9}$$

Fitting a value function.

$$c(\boldsymbol{\theta}) = \sum_{s \in \hat{S}} c_s(\boldsymbol{\theta}), \qquad c_s(\boldsymbol{\theta}) = \phi(s) \left\| v_{\boldsymbol{\theta}}(s) - v(s) \right\|_p^\kappa . \tag{2.10}$$

Example 6

The case $p = 2$, $\kappa = 2$

$$\theta'_j = \theta_j - 2\alpha\phi(s)[v_{\boldsymbol{\theta}}(s) - v(s)]f_j(s). \tag{2.11}$$

Generalised linear model using features (or kernel).

Feature mapping $f : \mathcal{S} \to \mathbb{R}^n$, parameters $\boldsymbol{\theta} \in \mathbb{R}^n$.

$$\pi_{\boldsymbol{\theta}}(a \mid s) = \frac{g(s, a)}{h(s)}, \qquad g(s, a) = \sum_{i=1}^{n} \theta_i f_i(s, a), \quad h(s) = \sum_{b \in \mathcal{A}} g(s, b) \tag{2.12}$$

Fitting a policy through a cost function.

$$c(\boldsymbol{\theta}) = \sum_{s \in \hat{S}} c_s(\boldsymbol{\theta}), \qquad c_s(\boldsymbol{\theta}) = \phi(s) \left\| \pi_{\boldsymbol{\theta}}(\cdot \mid s) - \pi(\cdot \mid s) \right\|_p^\kappa . \tag{2.13}$$

The case $p = 1$, $\kappa = 1$.

$$\theta'_j = \theta_j - \alpha \phi(s) \left(\pi_{\boldsymbol{\theta}}(a \mid s) \sum_{b \in \mathcal{A}} f_j(s, b) - f_j(s, a) \right) . \tag{2.14}$$

Algorithm 2 Rollout Sampling Approximate Policy Iteration.

for $k = 1, \ldots$ **do**
  Select a set of representative states $\hat{S}_k$
  for $n = 1, \ldots$ **do**
    Select a state $s_n \in \hat{S}_k$ maximising $U_n(s)$ and perform a rollout.
    If $\hat{a}^*(s_n)$ is optimal w.p. $1 - \delta$, put s_n in $\hat{S}_k(\delta)$ and remove it from $\hat{S}_k$.
  end for
  Calculate $\boldsymbol{q}_k \approx Q^{\pi_k}$ from the rollouts.
  Train a classifier $\pi_{\boldsymbol{\theta}_{k+1}}$ on the set of states $\hat{S}_k(\delta)$ with actions $\hat{a}^*(s)$.
end for

Least square value estimation

Projection.

Setting $\boldsymbol{v} = \boldsymbol{\Phi}\boldsymbol{\theta}$ where $\boldsymbol{\Phi}$ is a feature matrix and $\boldsymbol{\theta}$ is a parameter vector we have

$$\boldsymbol{\Phi}\boldsymbol{\theta} = \boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi}\boldsymbol{\Phi}\boldsymbol{\theta} \tag{2.15}$$

$$\boldsymbol{\theta} = \left[(\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})\boldsymbol{\Phi}\right]^{-1} \boldsymbol{r} \tag{2.16}$$

Replacing the inverse with the pseudo-inverse, with $\boldsymbol{A} = (\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})\boldsymbol{\Phi}$

$$\tilde{\boldsymbol{A}}^{-1} \triangleq \boldsymbol{A}^\top \left(\boldsymbol{A}\boldsymbol{A}^\top\right)^{-1},$$

Empirical constructions.

Given a set of data points $\{(s_i, a_i, r_i, s_i') \mid i = 1, \ldots, n\}$, which may not be consecutive, we define:

1. $\boldsymbol{r} = (r_i)_i$.
2. $\boldsymbol{\Phi}_i = f(s_i, a_i)$, $\boldsymbol{\Phi} = (\boldsymbol{\Phi}_i)_i$.
3. $\boldsymbol{P}_{\mu,\pi} = \boldsymbol{P}_\mu \boldsymbol{P}_\pi$, $\boldsymbol{P}_{\mu,\pi}(i,j) = \mathbb{I}\{j = i + 1\}$

Algorithm 3 LSTDQ - Least Squares Temporal Differences on $\boldsymbol{q}$-factors

input data $D = \{(s_i, a_i, r_i, s_i') \mid i = 1, \ldots, n\}$, feature mapping f, policy π

$\boldsymbol{\theta} = (\boldsymbol{\Phi}(\widetilde{\boldsymbol{I} - \gamma} \boldsymbol{P}_{\mu,\pi}))^{-1} \boldsymbol{r}$

Algorithm 4 LSPI - Least Squares Policy Iteration

input data $D = \{(s_i, a_i, r_i, s_i') \mid i = 1, \ldots, n\}$, feature mapping f

Set π_0 arbitrarily.

for $k = 1, \ldots$ **do**

  $\boldsymbol{\theta}_k = LSTDQ(D, f, \pi_{k-1})$.

  $\pi_k = \pi^*_{\boldsymbol{\Phi}\boldsymbol{\theta}_k}$.

end for

$$V_t^*(s) = \max_{a \in \mathcal{A}} \{r(s, a) + \gamma \mathbb{E}_\mu (V_{t+1}^* \mid s_t = s, a_t = a)\} \tag{3.1}$$

Iterative approximation

$$\hat{V}_t(s) = \max_{a \in \mathcal{A}} \left\{ r(s, a) + \gamma \sum_{s'} P_\mu(s' \mid s, a) \boldsymbol{v}_{t+1}(s') \right\} \tag{3.2}$$

$$\boldsymbol{v}_t = \arg\min \left\{ \left\| \boldsymbol{v} - \hat{V}_t \right\| \;\middle|\; \boldsymbol{v} \in \mathcal{V} \right\} \tag{3.3}$$

Online gradient estimation

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha_t \nabla_{\boldsymbol{\theta}} \left\| \boldsymbol{v}_t - \hat{V}_t \right\| \tag{3.4}$$

Aggregated estimate.

Let $\mathscr{G} = \{S_0, S_1, \ldots, S_n\}$ be a partition of $\mathcal{S}$, with $S_0 = \emptyset$ and $\boldsymbol{\theta} \in \mathbb{R}^n$ and let $f_k(s_t) = \mathbb{I}\{s_t \in S_k\}$. Then the approximate value function is

$$\boldsymbol{v}(s) = \boldsymbol{\theta}(k), \qquad \text{if } s \in S_k,\ k \neq 0. \tag{3.5}$$

Online gradient estimate.

Consider the case $\|\cdot\| = \|\cdot\|_2^2$. For $s_t \in S_k$:

$$\boldsymbol{\theta}_{t+1}(k) = (1 - \alpha)\boldsymbol{\theta}_t(k) + \alpha \max_{a \in \mathcal{A}} r(s_t, a) + \gamma \sum_j P(j \mid s_t, a)\boldsymbol{v}_t(s) \tag{3.6}$$

For $s_t \notin S_k$:

$$\boldsymbol{\theta}_{t+1}(k) = \boldsymbol{\theta}(k). \tag{3.7}$$

Representative states approximation.

Let $\hat{S}$ be a set of n representative states and $\boldsymbol{\theta} \in \mathbb{R}^n$ and a feature mapping f:

$$\sum_{i=1}^{n} f_i(s) = 1, \qquad \forall s \in \mathcal{S}.$$

Representative state update.

For $i \in \hat{S}$:

$$\boldsymbol{\theta}_{t+1}(i) = \max_{a \in \mathcal{A}} \left\{ r(i, a) + \gamma \int \boldsymbol{v}_t(s) \, \mathrm{d}P(s \mid i, a) \right\} \tag{3.8}$$

with

$$\boldsymbol{v}_t(s) = \sum_{i=1}^{n} f_i(s) \boldsymbol{\theta}_t(i). \tag{3.9}$$

Bellman error methods

$$\min_{\boldsymbol{\theta}} \|\boldsymbol{v}_{\boldsymbol{\theta}} - \mathscr{L}\boldsymbol{v}_{\boldsymbol{\theta}}\| \tag{3.10}$$

Gradient update.

When the norm is

$$\|\boldsymbol{v}_{\boldsymbol{\theta}} - \mathscr{L}\boldsymbol{v}_{\boldsymbol{\theta}}\| = \sum_{s \in \hat{S}} D_{\boldsymbol{\theta}}(s)^2, \qquad D_{\boldsymbol{\theta}}(s) = \boldsymbol{v}_{\boldsymbol{\theta}}(s) - \max_{a \in \mathcal{A}} \int_{\mathcal{S}} \boldsymbol{v}_{\boldsymbol{\theta}}(j) \, \mathrm{d}P(j \mid s, a). \tag{3.11}$$

then the gradient update becomes

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha D_{\boldsymbol{\theta}_t}(s_t) \nabla_{\boldsymbol{\theta}} D_{\boldsymbol{\theta}_t}(s_t) \tag{3.12}$$

$$\nabla_{\boldsymbol{\theta}} D_{\boldsymbol{\theta}_t}(s_t) = \nabla_{\boldsymbol{\theta}} \boldsymbol{v}_{\boldsymbol{\theta}_t}(s_t) - \int_{\mathcal{S}} \nabla_{\boldsymbol{\theta}} \boldsymbol{v}_{\boldsymbol{\theta}_t}(j) \, \mathrm{d}P(j \mid s_t, a_t^*) \tag{3.13}$$

$$a_t^* = \arg\max_{a \in \mathcal{A}} \left\{ r(s_t, a) + \gamma \int_{\mathcal{S}} \boldsymbol{v}_{\boldsymbol{\theta}}(j) \, \mathrm{d}P(j \mid s_t, a) \right\} \tag{3.14}$$

A litany of approximation algorithms

- Fitted Q-iteration [2].
- Fitted value iteration [16].
- Rollout sampling policy iteration [13]
- State aggregation [19, 4]
- Bellman error minimisation [1, 12, 14]
- Least-squares methods [8, 7, 15].

[1] A. Antos, C. Szepesvári, and R. Munos. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71(1):89–129, 2008.

[2] Andrè Antos, Rémi Munos, and Csaba Szepesvari. Fitted q-iteration in continuous action-space mdps. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*. MIT Press, Cambridge, MA, 2008.

[3] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2/3):235–256, 2002.

[4] A. Bernstein. Adaptive state aggregation for reinforcement learning. Master's thesis, Technion – Israel Institute of Technolog01y, 2007.

[5] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2001.

[6] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[7] J.A. Boyan. Technical update: Least-squares temporal difference learning. *Machine Learning*, 49(2):233–246, 2002.

[8] S.J. Bradtke and A.G. Barto. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22(1):33–57, 1996.

[9] Herman Chernoff. Sequential design of experiments. *Annals of Mathematical Statistics*, 30(3):755–770, 1959.

[10] Herman Chernoff. Sequential Models for Clinical Trials. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Vol.4*, pages 805–812. Univ. of Calif Press, 1966.

[11] Morris H. DeGroot. *Optimal Statistical Decisions*. John Wiley & Sons, 1970.

[12] Christos Dimitrakakis. Monte-carlo utility estimates for bayesian reinforcement learning. In *IEEE 52nd Annual Conference on Decision and Control (CDC 2013)*, 2013. arXiv:1303.2506.

[13] Christos Dimitrakakis and Michail G. Lagoudakis. Rollout sampling approximate policy iteration. *Machine Learning*, 72(3):157–171, September 2008. Presented at ECML'08.

[14] Mohammad Ghavamzadeh and Yaakov Engel. Bayesian policy gradient algorithms. In *NIPS 2006*, 2006.

[15] M.G. Lagoudakis and R. Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003.

[16] R. Munos and C. Szepesvári. Finite-time bounds for fitted value iteration. *The Journal of Machine Learning Research*, 9:815–857, 2008.

[17] Marting L. Puterman. *Markov Decision Processes : Discrete Stochastic Dynamic Programming*. John Wiley & Sons, New Jersey, US, 1994.

[18] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

[19] S. Singh, T. Jaakkola, and M.I. Jordan. Reinforcement learning with soft state aggregation. *Advances in neural information processing systems*, pages 361–368, 1995.

[20] Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *ICML 2010*, 2010.

[21] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.