Error Propagation for Approximate Policy and Value Iteration

Authored by:

R?mi Munos
Csaba Szepesv?ri
Amir-massoud Farahmand

Abstract

We address the question of how the approximation error/Bellman residual at each iteration of the Approximate Policy/Value Iteration algorithms influences the quality of the resulted policy. We quantify the performance loss as the Lp norm of the approximation error/Bellman residual at each iteration. Moreover, we show that the performance loss depends on the expectation of the squared Radon-Nikodym derivative of a certain distribution rather than its supremum – as opposed to what has been suggested by the previous results. Also our results indicate that the contribution of the approximation/Bellman error to the performance loss is more prominent in the later iterations of API/AVI, and the effect of an error term in the earlier iterations decays exponentially fast.

1 Paper Body

The exact solution for the reinforcement learning (RL) and planning problems with large state space is difficult or impossible to obtain, so one usually has to aim for approximate solutions. Approximate Policy Iteration (API) and Approximate Value Iteration (AVI) are two classes of iterative algorithms to solve RL/Planning problems with large state spaces. They try to approximately find the fixedpoint solution of the Bellman optimality operator. AVI starts from an initial value function V0 (or Q0), and iteratively applies an approximation of T ? , the Bellman optimality operator, (or T ? for the policy evaluation problem) to the previous estimate, i.e., Vk+1 ? T ? Vk . In general, Vk+1 is not equal to T ? Vk because (1) we do not have direct access to the Bellman operator but only some samples from it, and (2) the function space in which V belongs is not representative enough. Thus there would be an approximation error ?k = T ? Vk ? Vk+1 between the result of the exact VI and AVI. Some examples of AVI-based approaches are tree-based Fitted Q-Iteration of Ernst et al. [1], multilayer perceptron-based Fitted Q-Iteration of Riedmiller [2], and regularized Fitted Q-Iteration of Farahmand et al. [3]. See the work of Munos and Szepesv?ari [4] for more information about AVI. ? Csaba Szepesv?ari is on leave from MTA SZTAKI. We would like to acknowledge the insightful comments by the reviewers. This work was partly supported by AICML, AITF, NSERC, and PASCAL2 under no 216886.

1

API is another iterative algorithm to find an approximate solution to the fixed point of the Bellman optimality operator. It starts from a policy ?0 , and then approximately evaluates that policy ?0 , i.e., it finds a Q0 that satisfies T ?0 Q0 ? Q0 . Afterwards, it performs a policy improvement step, which is to calculate the greedy policy with respect to (w.r.t.) the most recent action-value function, to get a new policy ?1 , i.e., ?1 (?) = arg maxa?A Q0 (?, a). The policy iteration algorithm continues by approximately evaluating the newly obtained policy ?1 to get Q1 and repeating the whole process again, generating a sequence of policies and their corresponding approximate action-value functions Q0 ? ?1 ? Q1 ? ?2 ? ? ? ? . Same as AVI, we may encounter a difference between the approximate solution Qk (T ?k Qk ? Qk) and the true value of the policy Q?k , which is the solution of the fixed-point equation T ?k Q?k = Q?k . Two convenient ways to describe this error is either by the Bellman residual of Qk (?k = Qk ? T ?k Qk) or the policy evaluation approximation error (?k = Qk ? Q?k). API is a popular approach in RL literature. One well-known algorithm is LSPI of Lagoudakis and Parr [5] that combines Least-Squares Temporal Difference (LSTD) algorithm (Bradtke and Barto [6]) with a policy improvement step. Another API method is to use the Bellman Residual Minimization (BRM) and its variants for policy evaluation and iteratively apply the policy improvement step (Antos et al. [7], Maillard et al. [8]). Both LSPI and BRM have many extensions: Farahmand et al. [9] introduced a nonparametric extension of LSPI and BRM and formulated them as an optimization problem in a reproducing kernel Hilbert space and analyzed its statistical behavior. Kolter and Ng [10] formulated an l1 regularization extension of LSTD. See Xu et al. [11] and Jung and Polani [12] for other examples of kernel-based extension of LSTD/LSPI, and Taylor and Parr [13] for a unified framework. Also see the proto-value function-based approach of Mahadevan and Maggioni [14] and iLSTD of Geramifard et al. [15]. A crucial question in the applicability of API/AVI, which is the main topic of this work, is to understand how either the approximation error or the Bellman residual at each iteration of API or AVI affects the quality of the resulted policy. Suppose we run API/AVI for K iterations to obtain a policy ?K . Does the knowledge that all ?k s are small (maybe because we have had a lot of samples and used powerful function approximators) imply that V ?K is close to the optimal value function V ? too? If so, how does the errors occurred at a certain iteration k propagate through iterations of API/AVI and affect the final performance loss? There have already been some results that partially address this question. As an example, Proposition 6.2 of Bertsekas and Tsitsiklis [16] shows that for API applied to a finite MDP, we have 2? ?k lim supk?? kV ? ? V ?k k? ? (1??) ? Vk k? where ? is the discount facto. 2 lim supk?? kV Similarly for AVI, if the approximation errors are uniformly bounded (kT ? Vk ? Vk+1 k? ? ?), we 2? have lim supk?? kV ? ? V ?k k? ? (1??) 2 ? (Munos [17]). Nevertheless, most of these results are pessimistic in several ways. One reason is that they are expressed as the supremum norm of the approximation errors kV ?k ? Vk k? or the Bellman error kQk ? T ?k Qk k? . Compared to Lp norms, the supremum norm is conservative. It is quite possible that the result of a learning algorithm has a small Lp norm but a very large L? norm. Therefore, it is desirable to have a result expressed in Lp norm of the approximation/Bellman residual ?k . In the past couple of years, there have been attempts to extend L? norm results to Lp ones [18, 17, 7]. As a typical example, we quote the following from Antos et al. [7]: Proposition 1 (Error Propagation for API ? [7]). Let p ? 1 be a real and K be a positive integer. Then, for any sequence of functions {Q(k) } ? B(X ? A; Qmax)(0 ? k ¡ K), the space of Qmax bounded measurable functions, and their corresponding Bellman residuals ?k = Qk ? T ? Qk , the following inequalities hold:

K 2? 1/p p ?1 R kQ? ? Q?K kp,? ? C max k? k + ? , k max p,? (1 ? ?)2 ?,? 0?k¡K where Rmax is an upper bound on the magnitude of the expected reward function and

X

d (?P ?1 ? ? ? P ?m)

. C?,? = (1 ? ?)2 m? m?1 sup

d? ?1 ,...,?m ? m?1

This result indeed uses Lp norm of the Bellman residuals and is an improvement over results like Bertsekas and Tsitsiklis [16, Proposition 6.2], but still is pessimistic in some ways and does 2

not answer several important questions. For instance, this result implies that the uniform-over-alliterations upper bound max0?k¡K k?k kp,? is the quantity that determines the performance loss. One may wonder if this condition is really necessary, and ask whether it is better to put more emphasis on earlier/later iterations? Or another question is whether the appearance of terms in the form of ?1 ?m —— d(?P d????P) ——? is intrinsic to the difficulty of the problem or can be relaxed. The goal of this work is to answer these questions and to provide tighter upper bounds on the performance loss of API/AVI algorithms. These bounds help one understand what factors contribute to the difficulty of a learning problem. We base our analysis on the work of Munos [17], Antos et al. [7], Munos [18] and provide upper bounds on the performance loss in the form of kV ? ? V ?k k1,? (the expected loss weighted according to the evaluation probability distribution ? ? this is defined in Section 2) for API (Section 3) and AVI (Section 4). This performance loss depends on a certain function of ?-weighted L2 norms of ?k s, in which ? is the data sampling distribution, and C?,? (K) that depends on the MDP, two probability distributions ? and ?, and the number of iterations K. In addition to relating the performance loss to Lp norm of the Bellman residual/approximation error, this work has three main contributions that to our knowledge have not been considered before: (1) We show that the performance loss depends on the expectation of the squared Radon-Nikodym derivative of a certain distribution, to be specified in Section

3, rather than its supremum. The difference between this expectation and the supremum can be considerable. For instance, for a finite state space with N states, the ratio can be of order O(N 1/2). (2) The contribution of the Bellman/approximation error to the performance loss is more prominent in later iterations of API/AVI. and the effect of an error term in early iterations decays exponentially fast. (3) There are certain structures in the definition of concentrability coefficients that have not been explored before. We thoroughly discuss these qualitative/structural improvements in Section 5.

2

Background

In this section, we provide a very brief summary of some of the concepts and definitions from the theory of Markov Decision Processes (MDP) and reinforcement learning (RL) and a few other notations. For further information about MDPs and RL the reader is referred to [19, 16, 20, 21]. A finite-action discounted MDP is a 5-tuple (X , A, P, R, ?), where X is a measurable state space, A is a finite set of actions, P is the probability transition kernel, R is the reward kernel, and 0 ? ? ¡ 1 is the discount factor. The transition kernel P is a mapping with domain X ? A evaluated at (x, a) ? X ? A that gives a distribution over X , which we shall denote by P (?—x, a). Likewise, R is a mapping with domain X ? A that gives a distribution of immediate reward over R, which is denoted by R(?—x, a). We denote r(x, a) = E [R(?—x, a)], and assume that its absolute value is bounded by Rmax . A mapping ? : X ? A is called a deterministic Markov stationary policy, or just a policy in short. Following a policy ? in an MDP means that at each time step At = ?(Xt). Upon taking action At at Xt , we receive reward Rt ? R(?—x, a), and the Markov chain evolves according to Xt+1 ? P (?—Xt , At). We denote the probability transition kernel of following a policy ? by P ? , i.e., P ? (dy—x) = P (dy—x, ?(x)).

hP i

? t The value function V ? for a policy ? is defined as V ? (x) , E ? R X = x and the

t 0 t=0

hP i

? t action-value function is defined as Q? (x, a) , E t=0 ? Rt X0 = x, A0 = a . For a discounted MDP, we define the optimal value and action-value functions by V ? (x) = sup? V ? (x) (?x ? X) and Q? (x, a) = sup? Q? (x, a) (?x ? X , ?a ? A). We say that a policy ? ? is optimal ? if it achieves the best values in every state, i.e., if V ? = V ? . We say that a policy ? is greedy w.r.t. an action-value function Q and write ? = ? ? (?; Q), if ?(x) ? arg maxa?A Q(x, a) holds for allR x ? X . Similarly, the policy ? is greedy w.r.t. V , if for all x ? X , ?(x) ? argmaxa?A P (dx0 —x, a)[r(x, a) + ?V (x0)] (If there exist multiple maximizers, some maximizer is chosen in an arbitrary deterministic manner). Greedy policies are important because a greedy policy w.r.t. Q? (or V ?) is an optimal policy. Hence, knowing Q? is sufficient for behaving optimally (cf. Proposition 4.3 of [19]). 3

R We define the Bellman operator for a policy ? as (T ? V)(x) , r(x, ?(x))

+ ? V ? (x0)P (dx0 —x, a) R 0 0 and (T ? Q)(x, a) , r(x, a) + ? Q(x0 , ?(x optimality opn))P (dx —x,R a). Similarly, the Bellman o ? 0 0 erator is defined as (T V)(x) , maxa r(x, a) + ? V (x)P (dx —x, a) and (T ? Q)(x, a) , R r(x, a) + ? maxa0 Q(x0 , a0)P (dx0 —x, a). For a measurable space X , with a ?-algebra ?X , we define M(X) as the set of all probability measures overR ?X . For a probability measure ? ? M(X) and the transition kernel P ? , we define ?P ? (dx0) = P (dx0 —x, ?(x))d?(x). In words, ?(P ?)m ? M(X) is an m-step-ahead probability distribution of states if the starting state distribution is ? and we follow P ? for m steps. In what follows we shall use kV kp,? to denote the Lp (?)-norm of a measurable function V : X ? R: R p p kV kp,? , ?—V —p , X —V (x)—p d?(x). For a function Q : X ? A 7? R, we define kQkp,? , R P 1 p a?A X —Q(x, a)— d?(x). —A—

3

Approximate Policy Iteration

Consider the API procedure and the sequence Q0 ? ?1 ? Q1 ? ?2 ? ? ? ? ? QK?1 ? ?K , where ?k is the greedy policy w.r.t. Qk?1 and Qk is the approximate action-value function for policy ?k . For the sequence {Qk }K?1 k=0 , denote the Bellman Residual (BR) and policy Approximation Error (AE) at each iteration by ?k ?BR Qk , k = Qk ? T

?AE k

?k

= Qk ? Q .

(1) (2)

The goal of this section is to study the effect of ?-weighted L2p norm of the Bellman residual K?1 AE K?1 sequence {?BR k }k=0 or the policy evaluation approximation error sequence {?k }k=0 on the per? ?K formance loss kQ ? Q kp,? of the outcome policy ?K . The choice of ? and ? is arbitrary, however, a natural choice for ? is the sampling distribution of the data, which is used by the policy evaluation module. On the other hand, the probability distribution ? reflects the importance of various regions of the state space and is selected by the practitioner. One common choice, though not necessarily the best, is the stationary distribution of the optimal policy. Because of the dynamical nature of MDP, the performance loss kQ? ? Q?K kp,? depends on the difference between the sampling distribution ? and the future-state distribution in the form of ?P ?1 P ?2 ? ? ? . The precise form of this dependence will be formalized in Theorems 3 and 4. Before stating the results, we require to define the following concentrability coefficients. Definition 2 (Expected Concentrability of the Future-State Distribution). Given ?, ? ? M(X), ? ?1 (? is the Lebesgue measure), m ? 0, and an arbitrary sequence of stationary policies {?m }m?1 , let ?P ?1 P ?2 . . . P ?m ? M(X) denote the future-state distribution obtained when the first state is distributed according to ? and then we follow the sequence of policies {?k }m k=1 . Define the following concentrability coefficients that is used in API analysis: ? 2 ?? 21 d ?(P ??)m1 (P ?)m2

cPI1 ,?,? (m1 , m2 ; ?) , ?EX?? ? (X) ?? ,

d? ?

? 2 ?? 21 d ?(P ??)m1 (P ?1)m2 P ?2

cPI2 ,?,? (m1 , m2 ; ?1 , ?2) , ?EX?? ? (X) ?? ,

d? ?

? 2 ?? 12 d ?P ??

, ?EX?? ? (X) ?? ,

d? ?

cPI3 ,?,?

1 For two measures ?1 and ?2 on the same measurable space, we say that ?1 is absolutely continuous with respect to ?2 (or ?2 dominates ?1) and denote ?1 ?2 iff ?2 (A) = 0 ? ?1 (A) = 0.

4

?

with the understanding that if the future-state distribution ?(P ?)m1 (P ?)m2 (or ? ? ?(P ?)m1 (P ?1)m2 P ?2 or ?P ?) is not absolutely continuous w.r.t. ?, then we take cPI1 ,?,? (m1 , m2 ; ?) = ? (similar for others). Also define the following concentrability coefficient that is used in AVI analysis: ? ? 2 ?? 21 d ?(P ?)m1 (P ??)m2

cVI,?,? (m1 , m2 ; ?) , ?EX?? ? (X) ?? ,

d? ?

with the understanding that if the future-state distribution ?(P ?)m1 (P ?)m2 is not absolutely continuous w.r.t. ?, then we take cVI,?,? (m1 , m2 ; ?) = ?. In order to compactly present our results, we define the following notation: ?k =

(1 ? ?)? K?k?1 1 ? ? K+1

0 ? k ¡ K.

Theorem 3 (Error Propagation for API). Let p ? 1 be a real number, K be a positive integer, K?1 Rmax and Qmax ? 1?? . Then for any sequence {Qk }k=0 ? B(X ? A, Qmax) (space of Qmax -bounded K?1 measurable functions defined on X ? A) and the corresponding sequence {?k }k=0 defined in (1) or (2) , we have

1 1 K 2? 2p 2p (? , . . . , ? p ?1 R inf C ; r) + ? (K; r)E . 0 K?1 max (1 ? ?)2 r?[0,1] PI(BR/AE),?,? PK?1 2p where E(?0 , . . . , ?K?1 ; r) = k=0 ?k2r k?k k2p,? . kQ? ? Q?K kp,? ?

(a) If ?k = ?BR for all 0 ? k ¡ K, we have

CPI(BR),?,? (K; r) = (

K?1 X 2(1?r) 1?? 2 ?k) sup 0 2 ?00 ,...,?K k=0

X

0 ? m cPI1 ,?,? (K ? k ? 1, m + 1; ?k+1)+

m?0

cPI1 ,?,? (K ?

k, m; ?k0)

! 2

.

(b) If ?k = ?AE for all 0 ? k ¡ K, we have

CPI(AE),?,? (K; r, s) = (

K?1 X 2(1?r) 1?? 2) sup ?k 0 2 ?00 ,...,?K k=0

X

0 ? m cPI1 ,?,? (K ? k ? 1, m + 1; ?k+1)+

m?0

!2 X

0 ? m cPI2 ,?,? (K ? k ? 1, m; ?k+1 , ?k0) + cPI3 ,?,?

m?1

4

Approximate Value Iteration

Consider the AVI procedure and the sequence V0 ? V1 ? ? ? ? ? VK?1 , in which Vk+1 is the result of approximately applying the Bellman optimality operator on the previous estimate Vk , i.e., Vk+1 ? T ? Vk . Denote the approximation error caused at each iteration by ?k = T ? Vk ? Vk+1 .

(3)

The goal of this section is to analyze AVI procedure and to relate the approximation error sequence ? ?K {?k }K?1 kp,? of the obtained policy ?K , which is the greedy k=0 to the performance loss kV ? V policy w.r.t. VK?1 . 5

.

Theorem 4 (Error Propagation for AVI). Let p ? 1 be a real number, K be a positive integer, and Rmax Vmax ? 1?? . Then for any sequence {Vk }K?1 k=0 ? B(X , Vmax), and the corresponding sequence {?k }K?1 defined in (3), we have k=0

1 1 K 2 2? 2p ? ?K 2p p (K; r)E (?0 , . . . , ?K?1 ; r) + inf C ? Rmax , kV ? V kp,? ? (1 ? ?)2 r?[0,1] VI,?,? 1?? where CVI,?,? (K; r) = (

1?? 2) sup 2 ?0

and E(?0 , . . . , ?K?1 ; r) =

5

K?1 X

2(1?r)

?k

X ?

k=0

? m (cVI,?,? (m, K ? k; ? 0) + cVI,?,? (m + 1, K ? k ? 1; ? 0))? ,

m?0

k=0

PK?1

?2

?

2p

?k2r k?k k2p,? .

Discussion

In this section, we discuss significant improvements of Theorems 3 and 4 over previous results such as [16, 18, 17, 7]. 5.1

Lp norm instead of L? norm

As opposed to most error upper bounds, Theorems 3 and 4 relate kV ? ? V ?K kp,? to the Lp norm of the approximation or Bellman errors k?k k2p,? of iterations in API/AVI. This should be contrasted with the traditional, and more conservative, results such as lim supk?? kV ? ? V ?k k? ? 2? ?k ? Vk k? for API

(Proposition 6.2 of Bertsekas and Tsitsiklis [16]). The (1??)2 lim supk?? kV use of Lp norm not only is a huge improvement over conservative supremum norm, but also allows us to benefit from the vast literature on supervised learning techniques, which usually provides error upper bounds in the form of Lp norms, in the context of RL/Planning problems. This is especially interesting for the case of p = 1 as the performance loss kV ? ? V ?K k1,? is the difference between the expected return of the optimal policy and the resulted policy ?K when the initial state distribution is ?. Convenient enough, the errors appearing in the upper bound are in the form of k?k k2,? which is very common in the supervised learning literature. This type of improvement, however, has been done in the past couple of years [18, 17, 7] - see Proposition 1 in Section 1. 5.2

Expected versus supremum concentrability of the future-state distribution

The concentrability coefficients (Definition 2) reflect the effect of future-state distribution on the performance loss kV ? ? V ?K kp,? . Previously it was thought that the key contributing factor to the performance loss is the supremum of the Radon-Nikodym derivative of these two distributions. This is ? m evident in the definition of C?,? in Proposition 1 where we have terms in the form of —— d(?(Pd?)) ——?

h i 12 ? m instead of EX?? — d(?(Pd?)) (X)—2 that we have in Definition 2. Nevertheless, it turns out that the key contributing factor that determines the performance loss is the expectation of the squared Radon-Nikodym derivative instead of its supremum. Intuitively this ? m implies that even if for some subset of X 0 ? X the ratio d(?(Pd?)) is large but the probability ?(X 0) is very small, performance loss due to it is still small. This phenomenon has not been suggested by previous results. As an illustration of this difference, consider a Chain Walk with 1000 states with a single policy that 1 drifts toward state 1 of the chain. We start with ?(x) = 201 h for x ? [400, i 600] and zero everywhere ? m

? m

1

else. Then we evaluate both —— d(?(Pd?)) ——? and (EX?? — d(?(Pd?)) —2) 2 for m = 1, 2, . . . when ? is the uniform distribution. The result is shown in Figure 1a. One sees that the ratio is constant in the beginning, but increases when the distribution ?(P ?)m concentrates around state 1, until it reaches steady-state. The growth and the final value of the expectation-based concentrability coefficient is much smaller than that of supremum-based. 6

3

5 4

Infinity norm?based concentrability Expectation?base concentrability

Uniform Exponential

3 2

2

10

L1 error

Concentrability Coefficients

10

1
1.5 1 0.8 0.6 0.5 0.4
10
0.3 0
10
1
500
Step (m)
1000
1500
(a)
0.2 10
20
40
60
80
100
120
140
160
180
Iteration
(b)
h i 12 ? m ? m
and d(?(Pd?)) Figure 1: (a) Comparison of EX?? — d(?(Pd?)) —2
?
?
(b) Comparison of

kQ ? Qk k1 for uniform and exponential data sampling schedule. The total number of samples is the same. [The Y -scale of both plots is logarithmic.]

It is easy to show that if the Chain Walk has N states and the policy same i concentrating h has?the ? d(?(P ?)m) d(?(P)m) 2 1 behavior and ? is uniform, then —— ——? ? N , while (EX?? — —) 2 ? N when d? d? ? m ? ?. The ratio, therefore, would be of order ?(N). This clearly shows the improvement of this new analysis in a simple problem. One may anticipate that this sharper behavior happens in many other problems too. h i d? 2 12 More generally, consider C? = —— d? d? ——? and CL2 = (EX?? — d? —) . For a finite state space ? with N states and ? is the uniform distribution, C? ? N but CL2 ? N . Neglecting all other differences between our results and the previous ones, we get a performance upper bound in the form of kQ? ? Q?K k1,? ? c1 (?)O(N 1/4) supk k?k k2,? , while Proposition 1 implies that kQ? ? Q?K k1,? ? c2 (?)O(N 1/2) supk ——k ——2,? . This difference between O(N 1/4) and O(N 1/2) shows a significant improvement. 5.3

Error decaying property

Theorems 3 and 4 show that the dependence of performance loss kV ? ? V ?K kp,? (or PK?1 2r 2p kQ? ? Q?K kp,?) on {?k }K?1 k=0 ?k k?k k2p,? . This has k=0 is in the form of E(?0 , . . . , ?K?1 ; r) = a very special structure in that the approximation errors at later iterations have more contribution to the final performance loss. This behavior is obscure in previous results such as [17, 7] that the dependence of the final performance loss is expressed as E(?0 , . . . , ?K?1 ; r) = maxk=0,...,K?1 k?k kp,? (see Proposition 1). This property has practical and algorithmic implications too. It says that it is better to put more effort on having a lower Bellman or approximation error at later iterations of API/AVI. This, for instance, can be done by gradually increasing the number of samples throughout iterations, or to use more powerful, and possibly computationally more expensive, function approximators for the later iterations of API/AVI. To illustrate this property, we compare two different sampling schedules on a simple MDP. The MDP is a 100-state, 2-action chain similar to Chain Walk problem in the work of Lagoudakis and Parr [5]. We use AVI with a lookup-table function representation. In the first sampling schedule, every 20 iterations we generate a fixed number of fresh samples by following a uniformly random walk on the chain (this means that we throw away old samples). This is the fixed strategy. In the exponential strategy, we again generate new samples every 20 iterations but the number of samples at the k th iteration is ck ? . The constant c is tuned such that the total number of both sampling strategy is almost the same (we give a slight margin of about 0.1% of samples in favor of the fixed strategy). What we compare is kQ? ? Qk k1,? when ? is the uniform distribution. The result can be seen in Figure 1b. The improvement of the exponential sampling schedule is evident. Of course, one 7

200

may think of more sophisticated sampling schedules but this simple illustration should be sufficient to attract the attention of practitioners to this phenomenon. 5.4

Restricted search over policy space

One interesting feature of our results is that it puts more structure and restriction on the way policies may be selected. Comparing CPI,?,? (K; r) (Theorem 3) and CVI,?,? (K; r) (Theorem 4) with C?,? (Proposition 1) we see that: (1) Each concentrability coefficient in the definition of CPI,?,? (K; r) depends only on a single or two policies (e.g., ?k0 in cPI1 ,?,? (K ? k, m; ?k0)). The same is true for CVI,?,? (K; r). In contrast, the mth term in C?,? has ?1 , . . . , ?m as degrees of freedom, and this number is growing as m ? ?. (2) The operator sup in CPI,?,? and CVI,?,? appears outside the summation. Because of that, we 0 only have K + 1 degrees of freedom ?00 , . . . , ?K to choose from in API and remarkably only a single degree of freedom in AVI. On the other other hand, sup appears inside the summation in the definition of C?,? . One may construct an MDP that this difference in the ordering of sup leads to an arbitrarily large ratio of two different ways of defining the concentrability coefficients. (3) In API, the definitions of concentrability coefficients cPI1 ,?,? , cPI2 ,?,? , and cPI3 ,?,? (Defini? tion 2) imply that if ? = ?? , the stationary distribution induced by an optimal policy ? , then

2 ? ? m2 1 cPI1 ,?,? (m1 , m2 ; ?) = cPI1 ,?,? (?, m2 ; ?) = (EX?? d(? (Pd?))) 2 (similar for the other two coefficients). This special structure is hidden in

the definition of C?,? in Proposition 1, and instead we have an extra m1 degrees of flexibility. Remark 1. For general MDPs, the computation of concentrability coefficients in Definition 2 is difficult, as it is for similar coefficients defined in [18, 17, 7].

6

Conclusion

To analyze an API/AVI algorithm and to study its statistical properties such as consistency or convergence rate, we require to (1) analyze the statistical properties of the algorithm running at each iteration, and (2) study the way the policy approximation/Bellman errors propagate and influence the quality of the resulted policy. The analysis in the first step heavily uses tools from the Statistical Learning Theory (SLT) literature, e.g., Gy?orfi et al. [22]. In some cases, such as AVI, the problem can be cast as a standard regression with the twist that extra care should be taken to the temporal dependency of data in RL scenario. The situation is a bit more complicated for API methods that directly aim for the fixed-point solution (such as LSTD and its variants), but still the same kind of tools from SLT can be used too ? see Antos et al. [7], Maillard et al. [8]. The analysis for the second step is what this work has been about. In our Theorems 3 and 4, we have provided upper bounds that relate the errors at each iteration of API/AVI to the performance loss of the whole procedure. These bounds are qualitatively tighter than the previous results such as those reported by [18, 17, 7], and provide a better understanding of what factors contribute to the difficulty of the problem. In Section 5, we discussed the significance of these new results and the way they improve previous ones. Finally, we should note that there are still some unaddressed issues. Perhaps the most important one is to study the behavior of concentrability coefficients cPI1 ,?,? (m1 , m2 ; ?), cPI2 ,?,? (m1 , m2 ; ?1 , ?2), and cVI,?,? (m1 , m2 ; ?) as a function of m1 , m2 , and of course the transition kernel P of MDP. A better understanding of this question alongside a good understanding of the way each term ?k in E(?0 , . . . , ?K?1 ; r) behaves, help us gain more insight about the error convergence behavior of the RL/Planning algorithms.

2 References

[1] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. Journal of Machine Learning Research, 6:503?556, 2005. 8

[2] Martin Riedmiller. Neural fitted Q iteration ? first experiences with a data efficient neural reinforcement learning method. In 16th European Conference on Machine Learning, pages 317?328, 2005. [3] Amir-massoud Farahmand, Mohammad Ghavamzadeh, Csaba Szepesv?ari, and Shie Mannor. Regularized fitted Q-iteration for planning in continuous-space markovian decision problems. In Proceedings of American Control Conference (ACC), pages 725?730, June 2009. [4] R?emi Munos and Csaba Szepesv?ari. Finite-time bounds for fitted value iteration. Journal of Machine Learning Research, 9:815?857,

2008. [5] Michail G. Lagoudakis and Ronald Parr. Least-squares policy iteration. Journal of Machine Learning Research, 4:1107?1149, 2003. [6] Steven J. Bradtke and Andrew G. Barto. Linear least-squares algorithms for temporal difference learning. Machine Learning, 22:33?57, 1996. [7] Andr?as Antos, Csaba Szepesv?ari, and R?emi Munos. Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. Machine Learning, 71:89?129, 2008. [8] Odalric Maillard, R?emi Munos, Alessandro Lazaric, and Mohammad Ghavamzadeh. Finitesample analysis of bellman residual minimization. In Proceedings of the Second Asian Conference on Machine Learning (ACML), 2010. [9] Amir-massoud Farahmand, Mohammad Ghavamzadeh, Csaba Szepesv?ari, and Shie Mannor. Regularized policy iteration. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, Advances in Neural Information Processing Systems 21, pages 441?448. MIT Press, 2009. [10] J. Zico Kolter and Andrew Y. Ng. Regularization and feature selection in least-squares temporal difference learning. In ICML ?09: Proceedings of the 26th Annual International Conference on Machine Learning, pages 521?528, New York, NY, USA, 2009. ACM. [11] Xin Xu, Dewen Hu, and Xicheng Lu. Kernel-based least squares policy iteration for reinforcement learning. IEEE Trans. on Neural Networks, 18:973?992, 2007. [12] Tobias Jung and Daniel Polani. Least squares SVM for least squares TD learning. In In Proc. 17th European Conference on Artificial Intelligence, pages 499?503, 2006. [13] Gavin Taylor and Ronald Parr. Kernelized value function approximation for reinforcement learning. In ICML ?09: Proceedings of the 26th Annual International Conference on Machine Learning, pages 1017?1024, New York, NY, USA, 2009. ACM. [14] Sridhar Mahadevan and Mauro Maggioni. Proto-value functions: A Laplacian framework for learning representation and control in markov decision processes. Journal of Machine Learning Research, 8:2169?2231, 2007. [15] Alborz Geramifard, Michael Bowling, Michael Zinkevich, and Richard S. Sutton. iLSTD: Eligibility traces and convergence analysis. In B. Sch?olkopf, J. Platt, and T. Hoffman, editors, Advances in Neural Information Processing Systems 19, pages 441?448. MIT Press, Cambridge, MA, 2007. [16] Dimitri P. Bertsekas and John N. Tsitsiklis. Neuro-Dynamic Programming (Optimization and Neural Computation Series, 3). Athena Scientific, 1996. [17] R?emi Munos. Performance bounds in lp norm for approximate value iteration. SIAM Journal on Control and Optimization, 2007. [18] R?emi Munos. Error bounds for approximate policy iteration. In ICML 2003: Proceedings of the 20th Annual International Conference on Machine Learning, 2003. [19] Dimitri P. Bertsekas and Steven E. Shreve. Stochastic Optimal Control: The Discrete-Time Case. Academic Press, 1978. [20] Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning). The MIT Press, 1998. [21] Csaba Szepesv?ari. Algorithms for Reinforcement Learning. Morgan Claypool Publishers, 2010. [22] L?aszl?o Gy?orfi, Michael Kohler, Adam Krzy?zak, and Harro Walk. A Distribution-Free Theory of Nonparametric Regression. Springer Verlag, New York, 2002. 9