RAAM: The Benefits of Robustness in Approximating Aggregated MDPs in Reinforcement Learning

Authored by:

Marek Petrik
Dharmashankar Subramanian

Abstract

We describe how to use robust Markov decision processes for value function approximation with state aggregation. The robustness serves to reduce the sensitivity to the approximation error of sub-optimal policies in comparison to classical methods such as fitted value iteration. This results in reducing the bounds on the gamma-discounted infinite horizon performance loss by a factor of 1/(1-gamma) while preserving polynomial-time computational complexity. Our experimental results show that using the robust representation can significantly improve the solution quality with minimal additional computational cost.

1 Paper Body

State aggregation is one of the simplest approximate methods for reinforcement learning with very large state spaces; it is a special case of linear value function approximation with binary features. The main advantages of using aggregation in comparison with other value function approximation methods are its simplicity, flexibility, and the ease of interpretability (Bean et al., 1987; Bertsekas and Castanon, 1989; Van Roy, 2005). Informally, value function approximation methods compute an approximately-optimal policy ? ? by computing an approximate value function v? as an intermediate step. The quality of the solution can be measured by its performance loss: ?(? ?) ? ?(? ?) where ? ? is the optimal policy, and ?(?) is the ?-discounted infinite-horizon return of the policy, averaged over (any) given initial state distribution. The tight upper bound guarantees on the performance loss? tighter for state-aggregation than for general linear value function approximation?are (Van Roy, 2005), ?(? ?) ? ?(? ?) ? 4 ? (v ?)/(1 ? ?)2

(1.1)

where (v ?)?defined formally in Section 4?is the smallest approximation error for the optimal value function v ? . It is important that the error is with

respect to the optimal value function which can have special structural properties, such as convexity in inventory management problems (Porteus, 2002). Because the bound in (1.1) is tight, the performance loss may grow with the discount factor as fast as ?/(1??)2 while the total return for any policy only grows as 1/(1??). Therefore, for ? sufficiently close to 1, the policy ? ? computed through state aggregation may be no better than a random policy even when the approximation error of the optimal policy is small. This large performance loss is caused by large errors in approximating sub-optimal value functions (Van Roy, 2005). In this paper, we show that it is possible to guarantee much smaller performance loss by using a robust model of the approximation errors through a new algorithm we call RAAM (robust approximation for aggregated MDPs). Informally, we use robustness to reduce the approximated return of policies with large approximation errors to make it less likely that such policies will be selected. 1

The performance loss of the RAAM can be bounded as: ?(? ?) ? ?(? ?) ? 2 (v ?)/(1 ? ?) .

(1.2)

As the main contribution of the paper?described in Section 3?we incorporate the desired robustness into the aggregation model by assuming bounded worst-case state importance weights. The state importance weights determine the relative importance of the approximation errors among the states. RAAM formulates the robust optimization over the importance weights as a robust Markov decision process (RMDP). RMDPs extend MDPs to allow uncertain transition probabilities and rewards and preserve most of the favorable MDP properties (Iyengar, 2005; Nilim and Ghaoui, 2005; Le Tallec, 2007; Wiesemann et al., 2013). RMDPs can be solved in polynomial time and the solution methods are practical (Kaufman and Schaefer, 2013; Hansen et al., 2013). To minimize the overhead of RAAM in comparison with standard aggregation, we describe a new linear-time algorithm for the Bellman update in Section 3.1 for RMDPs with robust sets constrained by the L1 norm. Another contribution of this paper?described in Section 4?is the analysis of RAAM performance loss and the impact of the choice of robust uncertainty sets. We focus on constructing aggregate RMPDs with rectangular uncertainty sets (Iyengar, 2005; Wiesemann et al., 2013) and show that it is possible to use MDP structural properties to reduce RAAM performance loss guarantees compared to (1.2). The experimental results in Section 5 empirically illustrate settings in which RAAM outperforms standard state aggregation methods. In particular, RAAM is more robust to sub-optimal policies with a large approximation error. The results also show that the computational overhead of using the robust formulation is very small.

2

Preliminaries

In this section, we briefly overview robust Markov decision processes (RMDPs). RMDPs generalize MDPs to allow for uncertain transition probabilities and rewards. Our definition of RMDPs is inspired by stochastic zero-sum games to generalize previous results to allow for uncertainty in both the rewards and transition probabilities (Filar and Vrieze, 1997; Iyengar, 2005). Formally, an RMDP

is a tuple (S, A, B, P, r, ?), where S is a finite set of states, ? ? 4S is the initial distribution, As is a set of actions that can be taken in state s ? S, and Bs is a set of robust outcomes for s ? S that represent the uncertainty in transitions and rewards. From a game-theoretic perspective, Bs can be seen as the actions of the adversary. For any a ? As , b ? Bs , the transition probabilities are Pa,b : S ? 4S and the reward is ra,b : S ? R. The rewards depend only on the starting state and are independent of the target state1 . The basic solution concepts of RMDPs are very similar to regular MDPs with the exception that the solution also includes the policy of the adversary. We consider the set of randomized stationary policies ?R = {?s ? 4As }s?S as candidate solutions and use ?D for deterministic policies. Two main practical models of the uncertainty in Bs have been considered: s-rectangular and s, arectangular sets (Le Tallec, 2007; Wiesemann et al., 2013). In s-rectangular uncertainty models, the realization of the uncertainty depends only on the state and is independent on the action; the corresponding set of nature?s policies is: ?S = {?s ? 4Bs }s?S . In s, a-rectangular models, the realization of the uncertainty can also depend on the action: ?SA = {?s,a ? 4Bs }s?S,a?As . We Q will also consider restricted sets on the adversary?s policies: ?Q S = {?s ? Qs }s?S and ?SA = {?s,a ? Qs }s,a?S?As , for some Qs ? 4Bs . Next, we briefly overview the basic properties of robust MDPs; please refer to (Iyengar, 2005; Nilim and Ghaoui, 2005; Le Tallec, 2007; Wiesemann et al., 2013) for more details. The transitions and rewards for any stationary policies ? and ? are defined as: X X P?,? (s, s0) = Pa,b (s, s0) ?s,a ?s,b , r?,? (s) = ra,b (s) ?s,a ?s,b . a,b?As ?Bs

a,b?As ?Bs

1 Rewards that depend on the target state can be readily reduced to independent ones by taking the appropriate expectation.

2

It will be convenient to use P?,? to denote the transition matrix and r?,? and ? as vectors over states. We will also use I to denote an identity matrix and 1, 0 to denote vectors of ones and zeros respectively with appropriate dimensions. Using this notation, with a s, a-rectangular model, the objective in the RMDP is to maximize the ?-discounted infinite horizon robust return ? as: ?? = sup ?? (?) = sup ???R

inf ?(?, ?) = sup

???R ???SA

inf

? X

?T (? P?,?)t r?,? .

(RBST)

???R ???SA t=0

The negative superscript denotes the fact that this is the robust return. The value function for a policy ? pair ? and ? is denoted by v?,? and the optimal robust value function is v?? . Similarly to regular MDPs, the optimal robust value function must satisfy the robust Bellman optimality equation:

X X v?? (s) = max min ?s,a ?s,a,b ra,b (s) + ? Pa,b (s, s0) v?? (s0) . (2.1) ???R ???Q SA

3
s0 ?S
a,b?As ?Bs
RAAM: Robust Approximation for Aggregated MDPs

This section describes how RAAM uses transition samples to compute an approximately optimal policy. We also describe a linear-time algorithm for computing value function updates for the robust MDPs constructed by RAAM. Algorithm 1: RAAM: Robust Approximation for Aggregated MDPs // ? - samples, w - weights, ? - aggregation, ? - robustness Input: ?, w, ?, ? Output: ? ? ? approximately optimal policy // Compute RMDP parameters 1 S ? {?(? s) : (? s, s?0 , a ?, r) ? ?} ? {?(? s0) : (? s, s?0 , a ?, r?) ? ?} ; // States 2 forall the s ? S do 3 As ? {? a : (? s, s?0 , a ?, r) ? ?, s = ?(? s)} ; // Actions 4 Bs ? {? s : (? s, s?0 , a ?, r) ? ?, s = ?(? s)} ; // Outcomes 5 end // Compute RMDP transition probabilities and rewards 0 6 forall the s, s ? S ? S do 7 forall the a, b ? As ? Bs do 8 ?0 ? {(? s0 , r?) : (? s, s?0 , a ?, r?) ? ?, ?(? s) = s, a = a ?, b = s?} ; P 1 0 9 Pa,b (s, s) ? —?0 — s?0 ,???0 1s0 =?(?s0) ; P 10 ra,b (s) ? ?,?r??0 r?/—?0 — ; 11 end 12 end // Construct robust sets based on state weights and L1 bounds ws B k1 ? ?}; 13 Qs ? {? ? 4 s : k? ? T 1 w—B s
14
15 16
?Q SA ? {?s,a ? Qs }s,a?S?As ; // Solve RMDP ? Solve (2.1) to get ? ? ?the optimal RMDP policy?and let ? ?s?,a = ??(? s),a ; return ? ?;

Algorithm 1 depicts a simplified implementation of RAAM. In general, we use s? to distinguish the un-aggregated MDP states from the states in the aggregated RMDP. The main input to the algorithm consists of transition samples ? = {(? si , s?0i , a ?i , ri)}i?I which represent transitions from a state s?i 0 to the state s?i given reward ri and taking an action ai ; the transitions need to be sampled according to the transition probabilities conditioned on the state and an action. The aggregation function ? : S? ? S, which maps every MDP state from S? to an aggregate RMDP state, is also assumed to be given. Finally, the state weights w ? 4S and the robustness ? are tunable parameters. We use the L1 norm to bound the uncertainty. The representation uses ? to continuously trade off between fixed importance weights for ? = 0 and complete robustness ? = 2. We analyze 3
1
a1
a1
s?1
0
s?2
0 s?1
0
s?2
s1
s?3
s1

s2
1 a2
s?1
Figure 1: An example MDP.
0
0
a2
s2
s?2
Figure 2: Aggregated RMDP.

the effect of this parameter in Section 4. However, simply setting w to be uniform and ? = 2 will provide sufficiently strong theoretical guarantees and generally works well in practice. Finally, we assume s, a-rectangular uncertainty sets for the sake of reducing the computational complexity; better approximations could be obtained by using s-rectangular sets, but this makes no difference for deterministic policies. Next, we show an example that demonstrates how the robust MDP is constructed from the aggregation. We will also use this example to show the tightness of our bounds on the performance loss. Example 3.1. The original MDP problem is shown in Fig. 1. The round white nodes represent the states, while the black nodes represent state-action pairs. All transitions are deterministic, with the number next to the transition representing the corresponding reward. Two shaded regions marked with s1 and s2 denote the aggregate states. Fig. 2 depicts the corresponding aggregated robust MDP constructed by RAAM. The rectangular nodes in this picture represent the robust outcome. 3.1

Reducing Computational Complexity

Solving an RMDP is in general more difficult than solving a regular MDP. Most RMDP algorithms are based on value or policy iteration, but in general involve repeated solutions of linear or convex programs (Kaufman and Schaefer, 2013). Even though the worst-case time complexity of these algorithms is polynomial, they may be impractical because they require repeatedly solving (2.1) for every state, action, and iteration. Each of these computations may require solving a linear program. The optimization over ?SA when computing the value function update for solving Line 15 of Algorithm 1 requires solving the following linear program for each s and a. min

?s,a ?4Bs

s.t.

T ?s,a zs =

X

X ?s,a,b ra,b (s) + ? Pa,b (s, s0) v(s0) s0 ?S

b?Bs

(3.1)

k?s,a ? qs k1 ? ? .

Here qs = ws /1T w(Bs). While this problem can be solved directly using a linear program solver, we describe a significantly more efficient method in Algorithm 2. Theorem 3.2. Algorithm 2 correctly solves (3.1) in O(—Bs —)

time when the full sort is replaced by a quickselect quantile selection algorithm in Line 4. The proof is technical and is deferred to Appendix B.1. The main idea is to dualize the norm constraint and examine the structure of the optimal solution as a function of the dual variable.

4

Performance Loss Bounds

This section describes new bounds on the performance loss which is the difference between the return of the optimal and approximate policy. The performance loss is a more reliable measure of the error than the error in the value function (Van Roy, 2005). We also analyze the effect of the state weights w and the robustness parameter ? on performance loss. It will be convenient, for the purpose of deriving the error bounds, to treat aggregation as a linear value function approximation (Van Roy, 2005). For that purpose, define a matrix ?(? s, s) = 1s=?(?s) 4

Algorithm 2: Solve (3.1) in Line 15 of Algorithm 1 Input: zs , qs ? sorted such that zs is non-decreasing, indexed as 1 . . . n ? Output: ?s,a ? optimal solution of (3.1) 1 o ? copy(qs) ; i ? n ; ? 2 ? min{1 ? q1 , 2} ; 3 o1 ? + q 1 ; 4 while ¿ 0 ; // Determine the threshold 5 do 6 oi ? oi ? min{, oi } ; 7 ? ? min{, oi } ; 8 i ? i ? 1; 9 end 10 return o ;

? and 1 represents the indicator function. That is, each column corresponds to where s ? S, s? ? S, a single aggregate state with each row entry being either 1 or 0 depending on whether the original state belongs into the aggregate state. In order to simplify the derivation of the bounds, we start by assuming that the RMDP in RAAM is constructed from the full sample of the original MDP; we discuss finite-sample bounds later. ? A, ? P? , r?, ? Therefore, assume that the full regular MDP is M = (S, ?); we are using bars in general ? to denote MDP values, but assume that A = A. We also use ?? to denote the return of a policy in the MDP. The robust outcomes correspond to the original states that compose any s: Bs = ??1 (s). The RMDP transitions and rewards for some ? and ? are computed as:

r?,? = ?T diag ?? r?? ?T = ? ? T ?. (4.1) P?,? = ?T diag ?? P?? ? P Here, ??s? = a?As? ?s,a ?s,a,?s such that ?(? s) = s are state weights induced by ?. There are two types of optimal policies: ? ? ? and ? ? ; ? ? ? is the truly optimal policy, while ? ? is the optimal policy given aggregation constraints requiring the same action for all aggregated states. For any computed policy ? ? , we focus primarily ??(? ?)? ??(? ?). The total loss can on?the performance loss ? ? ? be easily decomposed as ??(? ?)? ??(? ?) = ??(? ?)? ??(?) + ??(?)? ??(? ?) . The error ?(? ? ?)? ??(? ?) is independent of how the value of the aggregation is computed. The following theorem states the main result of the paper. A part of the results uses the concentration coefficient C for a given distribution ? of the MDP (Munos, 2005) which are defined as: P?a (s, s0) ? ? a ? A. ? C?(s0) for all s, s0 ? S, Theorem 4.1. Let ? ? be the solution of Algorithm 1 based on the full sample for ? = 2. Then: ??(? ?) ? ??(? ?) ?

2 (v ?) , 1??

where (v ?) = minv?RS kv ? ? ?vk? and this bound is tight. In addition,

when the concentration coefficient of the original MDP is C with distribution ?, then (v ?) = minv?RS ke(v)k1,? where ? = ?T (? ? + (1 ? ?) ?) and e(v)s = maxs????1 (s) —(I ? ? P???)(? v ? ? ? v)—s?. Before proving Theorem 4.1, it is instrumental to compare it with the performance loss of related reinforcement learning algorithms. When the aggregation is constructed using constant and uniform aggregation weights (as when Algorithm 1 is used with ? = 0), the performance loss of the computed policy ? ? is bounded as (Tsitsiklis and Van Roy, 1996; Gordon, 1995): ??(? ?) ? ??(? ?) ?

4 ? (v ?) . (1 ? ?)2

This bound holds specifically for aggregation (and approximators that are averagers) and is tight; the performance loss for more general algorithms can be even larger. Note that the difference in the 1/(1 ? ?) factor is very significant when ? ? 1. Van Roy (2005) shows similar bounds as RAAM, but they are weaker and require the invariant distribution ?. In addition, similar performance loss bounds as Theorem 4.1 can be guaranteed by DRADP, but this approach results in general to NPhard computational problems (Petrik, 2012). In fact, the robust aggregation can be seen as a special case of DRADP with rectangular uncertainty sets (Iyengar, 2005). 5

To prove Theorem 4.1 we need the following result showing that for properly chosen robust uncertainty sets, the robust return is a lower bound on the true return. We will use d?? to represent the normalized occupancy frequency for the MDP M and policy ?. Q ? Lemma 4.2. Assume the uncertainty set to be ?Q S or ?SA as constructed in (4.1). Then ? (?) ? ? ??(?) as long as for each ? ? ? we have that d? —Bs ? ?s ? Qs for each s ? S and some ?s .

When ? = 2, the inequality in the theorem also holds for value functions as Proposition B.1 in the appendix shows. Proof. We prove the result for s-rectangular uncertainty sets; the proof for s, a-rectangular sets is analogous. When the policy ? is fixed, solving for the nature?s policy represents a minimization MDP with continuous action constraints that has the following dual linear program formulation (Marecki et al., 2013): ?? (?) =

min

d?{RBs }s?S

dT r?? / (1 ? ?) ?T (I ? ? P??T) d = (1 ? ?) ?T ? ? X ds,b / ds,b0 ? Qs , ?s ? S, ?b ? Bs .

s.t.

(4.2)

b0 ?Bs

Note that the left-hand side of the last constraint corresponds to ?a,b . Now, setting d = d?? shows the desired inequality for ?; this value is feasible inP (4.2) from (B.3) and the objective value is correct from (B.4). The normalization constant is ?s = b0 ?Bs ds,b0 . Proof of Theorem 4.1. Using Lemma 4.2, the performance loss for ? = 2 can be bounded as: 0 ? ??(? ?) ? ??(? ?) ? ??(? ?) ? ?? (? ?) = min(? ?(? ?) ? ??? (?)) ? ??(? ?) ? ?? (? ?) ???

?

For a policy ?, solving ? (?) corresponds to an MDP with the following LP formulation: ??(? ?) ? ?? (? ?) ? min {?T (v ? ? ?v) : ?v ? ? P??? ?v +

r?? } . v

(4.3)

1+? Now, let the minimum = minv kv ? ??vk? be attained at v0 . Then, to show that v1 = v0 ? 1?? 1 is feasible in (4.3), for any k:

? 1 ? v ? ? ?v0 ? 1 (k ? 1) 1 ? v ? ? ?v0 + k 1 ? (1 + k) 1 (k ? 1)? 1 ? ? P??? (v ? ? ?v0 + k 1) ? (1 + k)? 1

(4.4) (4.5)

The derivation above uses the monotonicity of P??? in (4.5). Then, after multiplying by (I ? ? P???), which is monotone, and rearranging the terms: (I ? ? P???)?(v0 ? k 1) ? (1 + ? ? (1 ? ?)k) 1 + r?? , where (I ? ? P???)v ? = r?? . Letting k = (1 + ?)/(1 ? ?) proves the needed feasibility and (4.4) establishes the bound. The tightness of the bound follows from Example 3.1 with ? 0. The bound on the second inequality follows from bounding the dual gap between the primal feasible solution v1 and an upper bound on a dual optimal solution. To upper-bound the dual solution, define a concentration coefficient for an RMDP similarly to an MDP: P?a,b (s, s0) ? C?(s0) for all s, s0 ? S, a ? As , b ? Bs . By algebraic manipulation, if the original MDP has a concentration coefficient C with a distribution ? then the aggregated RMDP has the same concentration coefficient with a distribution ?T ?. Then, using Lemma 4.3 in (Petrik, 2012), the occupancy frequency (and therefore C the dual value) of the RMDP for any policy is bounded as u ? 1?? ?((1 ? ?) ?T ? + ??T ?). The linear program (4.3) can be formulated as the following penalized optimization problem:

max min ?T (v ? ? ?v) + uT (I ? ? P???)?v ? r?? + , u

v

Note that: ?T (v ? ? ?v) = ?T I ? ? P???

?1

? ? (I ? ? P???)(v ? ? ?v) = d?T ? ? (I ? ? P? ?)(v ? ?v) .

6

The penalized optimization problem can be rewritten, using the fact that r?? = (I ? ? P???) v ? and the feasibility of v1 as: max u

s.t.

2 uT —(I ? ? P???)(? v1 ? v ?)— 1?? C u? ? ((1 ? ?) ?T ? + ? ?T ?) 1??

The theorem then follows by simple algebraic manipulation from the upper bound on u. 4.1

State Importance Weights

In this section, we discuss how to select the state importance weights w and the robustness parameter ?. Note that Lemma 4.2 shows that any choice of w and ? such that the normalized occupancy frequency is within ? of w in terms of the L1 norm, provides the theoretical guarantees of Theorem 4.1. Smaller uncertainty sets under this condition only improve the guarantees. In practice, the values w and ? can be treated as regularization parameters. We show sufficient conditions under which the right choice of w and ? can significantly reduce the performance loss, but these conditions have a more explanatory than predictive character. As it can be seen easily from the proof of Lemma 4.2 and Appendix B.2, the optimal choice for the RAAM weights w to approximate

the return of a policy ? is to use its state occupancy frequency. While the occupancy frequency is rarely known, there exist structural properties, such as the concentration coefficient (Munos, 2005), that can lead to upper bounds on the possible occupancy frequencies. However, the following example shows that simply using an upper bound on the occupancy frequency is not sufficient to reduce the performance loss. Example 4.3. Consider an MDP with 4 states: s1 , . . . , s4 and the aggregation with two states that correspond to {s1 , s2 } and {s3 , s4 }. Let the set of admissible occupancy frequencies be: Q = {d ? 44 : 1/4 ? d(s1) + d(s4) ? 1/2, d ? 1/8}. The set of uncertainties for this bounded set is 4 for i = 1, 3, and j = 2, 4 as follows: ?Q S = {d ? R+ : 1/6 ? d(si) ? 4/5, 1/5 ? d(sj) ? 5/6, d(si) + d(sj) = 1}, which is smaller than ?S . However, Q without the constraint d ? 1/8 results in ?Q S = ?S . As Example 4.3 demonstrates, the concentration coefficient alone does not guarantee an improvement in the policy loss. One possible additional structural assumption is that the occupancy frequencies for the individual states in each aggregate state to be ?correlated? across policies. More formally, the aggregation correlation coefficient D ? R+ must satisfy: ? ?(? s) ? d? (? s) ? ? D ?(? s) ,

(4.6)

? and ? as defined in Theorem 4.1. Using this assumption, we can derive for some ? ? 0, each s? ? S, the following theorem. Consider the uncertainty set Qs = {q : q ? C (?—Bs)/(1T ?(Bs))} then we can show the following theorem. Theorem 4.4. Given an MDP with a concentration coefficient C for ? and a correlation coefficient T D, then for uncertainty set ?Q S and for ? = ? (? ? + (1 ? ?) ?) we have: ??(? ?) ? ??(? ?) ?

2C D min k(I ? ? P???) (? v ? ? ? v)k1,? . 1 ? ? v?RS

The proof is based on a minor modification of Theorem 4.1 and is deferred until the appendix. Theorem 4.4 improves on Theorem 4.1 by entirely replacing the L? norm by a weighted L1 norm. While the correlation coefficient may not be easy to determine in practice, it may a property to analyze to explain a failure of the method. Finite-sample bounds are beyond the scope of this paper. However, the sampling error is additive and can be based for example on coverage assumptions made for approximate linear programs. In particular, (4.2) represents an approximate linear program and can be bounded as such, as for example done by Petrik et al. (2010).

5

Experimental Results

In this section, we experimentally validate the approximation properties of RAAM with respect to the quality of the solutions and the computational time required. For the purpose of the empirical 7

40 Robust Aggregation, jj ?jj1 ?1:5

10?1

Approximate Linear Programming

0

Time (s)

Mean Return

20

100
Mean Aggregation/LSPI Robust Aggregation, jj ?jj1 ?0:5
?20
?40
?60 0.0
CPLEX Total CPLEX Solver Custom Python Custom C++
10?2
10?3
10?4
0.5
1.0 Extra Reward rq
1.5
10?5 1 10
2.0
102
103
104
Variables

Figure 3: Sensitivity to the reward perturbation for regular aggregation and RAAM.

Figure 4: Time to compute (3.1) for Algorithm 2 versus a CPLEX LP solver.

evaluation we use a modified inverted pendulum problem with a discount factor of 0.99, as described for example in (Lagoudakis and Parr, 2003). For the aggregation, we use a uniform grid of dimension 40 ? 40 and uniform sampling of dimensions 120 ? 120. The ordinary setting is solved easily and reliably by both the standard aggregation and RAAM. To study the robustness with respect to the approximation error of suboptimal policies we add an additional reward ra for the pendulum under a tilted angle (?/2 ? 0.12 ? ? ? ?/2 and ?? ? 0 where ? is the angle and ?? is the action). This reward can be only achieved by a suboptimal policy. Fig. 3 shows the return of the approximate policy as the function of the magnitude of the additional reward for the standard aggregation and RAAM with various values on ?. We omit the confidence ranges, which are small, to enhance image clarity. Note that we assume that once the pendulum goes over ?/2, the reward -1 is accrued until the end of the horizon. This result clearly demonstrates the greater stability and robustness of RAAM for than standard aggregation. The results also illustrate the lack of stability of ALP, which is can be seen as an optimistic version of RAAM. We observed the same behavior for other parameter choices. The main cost of using RAAM compared to ordinary aggregation is the increased computational complexity. Our results show, however, that the computational overhead of RAAM is minimal. Section 5 shows that Algorithm 2 is several orders of magnitude faster than CPLEX 12.3. The value function update for the aggregated inverted pendulum with 1600 states, 3 actions, and about 9 robust outcomes takes 8.7ms for ordinary aggregation, 8.8ms for RAAM with ? = 2, and 9.7ms for RAAM with ? = 1. The guarantees on the improvement for one iteration are the same for both algorithms and all implementations are in C++.

6

Conclusion

RAAM is novel approach to state aggregation which leverages RMDPs. RAAM significantly reduces performance loss guarantees in comparison with standard aggregation while introducing negligible computational overhead. The robust approach has some distinct advantages in comparison with previous methods with improved performance loss guarantees. Our experimental results are encouraging and show that adding robustness can significantly improve the solution quality. Clearly, not all problems will benefit from this approach. However, given the small computational overhead and there is no reason to not try. While we do provide some theoretical justification for choosing w and ?, it is most likely that in practice these can be best treated as regularization parameters. Many improvements on the basic RAAM algorithm are possible. Most notably, the RMDP action set could be based on ?meta-actions? or ?options?. The L1 may be replaced by other polynomial norms or KL divergence. RAAM could be also extended to choose adaptively the most appropriate aggregation for the given samples (Bernstein and Shikim, 2008). Finally, using s-rectangular uncertainty sets may lead to better results. Acknowledgments We thank Ban Kawas for extensive discussions on this topic and the anonymous reviewers for their comments that helped to significantly improve the paper.

8

2 References

Bean, J. J. C., Birge, J. R. J., and Smith, R. R. L. (1987). Aggregation in dynamic programming. Operations Research, 35(2), 215?220. Bernstein, A. and Shikim, N. (2008). Adaptive aggregation for reinforcement learning with efficient exploration: Deterministic domains. In Conference on Learning Theory (COLT). Bertsekas, D. P. D. and Castanon, D. A. (1989). Adaptive aggregation methods for infinite horizon dynamic programming. IEEE Transations on Automatic Control, 34, 589?598. de Farias, D. P. and Van Roy, B. (2003). The linear programming approach to approximate dynamic programming. Operations Research, 51(6), 850?865. Desai, V. V., Farias, V. F., and Moallemi, C. C. (2012). Approximate dynamic programming via a smoothed linear program. Operations Research, 60(3), 655?674. Filar, J. and Vrieze, K. (1997). Competitive Markov Decision Processes. Springer. Gordon, G. J. (1995). Stable function approximation in dynamic programming. In International Conference on Machine Learning, pages 261?268. Carnegie Mellon University. Hansen, T., Miltersen, P., and Zwick, U. (2013). Strategy iteration is strongly polynomial for 2player turn-based stochastic games with a constant discount factor. Journal of the ACM (JACM), 60(1), 1?16. Iyengar, G. N. (2005). Robust dynamic programming. Mathematics of Operations Research, 30(2), 257?280. Kaufman, D. L. and Schaefer, A. J. (2013). Robust modified policy iteration. INFORMS Journal on Computing, 25(3), 396?410. Lagoudakis, M. G. and Parr, R. (2003). Least-squares policy iteration. Journal of Machine Learning

Research, 4, 1107?1149. Le Tallec, Y. (2007). Robust, Risk-Sensitive, and Data-driven Control of Markov Decision Processes. Ph.D. thesis, MIT. Mannor, S., Mebel, O., and Xu, H. (2012). Lightning does not strike twice: Robust MDPs with coupled uncertainty. In International Conference on Machine Learning. Marecki, J., Petrik, M., and Subramanian, D. (2013). Solution methods for constrained Markov decision process with continuous probability modulation. In Uncertainty in Artificial Intelligence (UAI). Munos, R. (2005). Performance bounds in Lp norm for approximate value iteration. In National Conference on Artificial Intelligence (AAAI). Nilim, A. and Ghaoui, L. E. (2005). Robust control of Markov decision processes with uncertain transition matrices. Operations Research, 53(5), 780?798. Petrik, M. (2012). Approximate dynamic programming by minimizing distributionally robust bounds. In International Conference of Machine Learning. Petrik, M. and Zilberstein, S. (2009). Constraint relaxation in approximate linear programs. In International Conference on Machine Learning, New York, New York, USA. ACM Press. Petrik, M., Taylor, G., Parr, R., and Zilberstein, S. (2010). Feature selection using regularization in approximate linear programs for Markov decision processes. In International Conference on Machine Learning. Porteus, E. L. (2002). Foundations of Stochastic Inventory Theory. Stanford Business Books. Puterman, M. L. (2005). Markov decision processes: Discrete stochastic dynamic programming. John Wiley & Sons, Inc. Tsitsiklis, J. N. and Van Roy, B. (1996). An analysis of temporal-difference learning with function approximation. Van Roy, B. (2005). Performance loss bounds for approximate value iteration with state aggregation. Mathematics of Operations Research, 31(2), 234?244. Wiesemann, W., Kuhn, D., and Rustem, B. (2013). Robust Markov decision processes. Mathematics of Operations Research, 38(1), 153?183. 9