Abstract number: 007-0058

Adaptive Critics for Airline Revenue Management

Abhijit Gosavi
Department of Industrial and Systems Engineering
University at Buffalo, SUNY
317 Bell Hall, Buffalo, NY 14260
agosavi@buffalo.edu

POMS 18th Annual Conference
Dallas, Texas, U.S.A.
May 4 to May 7, 2007.

Abstract

We present an approximate dynamic programming (DP) technique, called adaptive critic, for solving an airline revenue management problem. The revenue management problem is cast as a semi-Markov decision problem (semi-MDP). Werbos [25] first suggested the use of policy iteration in approximate DP, and the adaptive critic essentially falls within this class of algorithms. Barto et al [1] wrote a paper that provided a step-size-based technique for this algorithm. Recently, Konda and Borkar [15] analyzed a modified version for its convergence properties. All three of these works focus on the MDP. We combine notions from these works to solve the associated semi-MDP and prove its convergence. We provide computational evidence to demonstrate the efficacy of our method. In addition, we extend our technique to handle Markowitz risk, which should appeal to risk-sensitive airline managers seeking to avoid bankruptcy.

1 Introduction

For solving complex Markov decision problems (MDPs), Adaptive critics (AC) form a useful class of algorithms. For solving MDPs via policy iteration, Werbos [25] developed a framework now well-known as *H*euristic *D*ynamic *P*rogramming (HDP). A powerful feature of this framework is that it relies on a derivative-based mechanism that handily allows the functions of actors and critics to be approximated via neural networks. HDP has led to the birth of several important algorithms, e.g., dual heuristic programming and action-dependent HDP [24, 19]. Another development was that of Barto et al. [1], which combined learning aspects within the AC framework. A third development is that of Konda and Borkar [15], who showed that in a two-time-scale framework most AC algorithms converge with probability 1. Notably, there is also a significant amount of literature on reinforcement learning (RL) [22, 10] or neuro-dynamic (approximate dynamic) programming [3] that addresses AC algorithms.

In this paper, we extend the policy-iteration-based AC algorithm to semi-MDPs (SMDPs) and test it on the airline revenue management (ARM) problem. The ARM problem under realistic considerations, such as cancelations and overbooking, is intractable via classical dynamic programming (DP) because it is difficult to generate the exact transition probability matrices involved. Our work here is motivated by a need to solve the ARM problem in a near-exact fashion. We also study a value-iteration-based approximate DP (ADP) algorithm for the SMDP. We then prove the convergence of both algorithms.

We then extend the AC algorithm to handle risk. In particular, we look at variance, which as a measure of risk was originally presented in Markowitz [17]. Risk-sensitive DP is a topic of much interest currently [5, 4]. The problem we study is solvable via a quadratic programming approach [9], but not exactly by dynamic programming (DP); see however [12] for a related criterion. It turns out that the two-time-scale framework [6] is well-suited for the development of our algorithm. We are still working on the numerical results with the *risk-sensitive* algorithm on the ARM problem, and we will present those at the conference.

The rest of this paper is organized as follows. Section 2 presents the SMDP model and then describes the ARM problem in some detail. It also describes the new AC algorithm and a value-iteration-based algorithm, along with the convergence proofs. Section 3 presents the risk-sensitive versions of the AC algorithm. Section 4 presents the numerical results with the ARM problem and small MDPs for the risk-sensitive case. Section 5 concludes the paper with a discussion on large-scale approximation and comments on possible future work.

2 SMDP modeling for ARM

2.1 SMDP

We first present the SMDP model and begin with some notation and definitions. Let $\mathcal{S}$ denote the finite set of states in the SMDP, $\mathcal{A}(i)$ the finite set of actions permitted in state i, and $\mu(i)$ the action chosen in state i when policy $\hat{\mu}$ is pursued, where $\cup_{i\in\mathcal{S}}\mathcal{A}(i) = \mathcal{A}$. Further let $r(.,.,.) : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to \Re$ denote the one-step immediate reward, $t(.,.,.) : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to \Re$ denote the time spent in one transition, and $p(.,.,.) : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to [0,1]$ denote the associated transition probability. Then the *expected* immediate reward earned in state i when action a is chosen in it can be expressed as: $\bar{r}(i,a) = \sum_{j=1}^{|S|} p(i,a,j)r(i,a,j)$ and the expected time of the associated transition: $\bar{t}(i,a) = \sum_{j=1}^{|S|} p(i,a,j)t(i,a,j)$. In our notation, $\vec{x}$ will denote a column vector whose ith element is $x(i)$.

Now, let

$$\alpha_{\hat{\mu}}(i) \equiv \lim_{k\to\infty} \frac{E_{\hat{\mu}}\left[\sum_{s=1}^{k} \bar{r}(x_s,\mu(x_s))|x_1 = i\right]}{k} \text{ and}$$

$$\beta_{\hat{\mu}}(i) \equiv \lim_{k\to\infty} \frac{E_{\hat{\mu}}\left[\sum_{s=1}^{k} \bar{t}(x_s,\mu(x_s))|x_1 = i\right]}{k}.$$

Then for irreducible and recurrent Markov chains, from Theorem 7.5 [18] (pg 160), the long-run average reward of a policy $\hat{\mu}$ in an SMDP starting at state i is $\rho_{\hat{\mu}}(i) = \frac{\alpha_{\hat{\mu}}(i)}{\beta_{\hat{\mu}}(i)}$. For irreducible and aperiodic Markov chains, $\rho(.)$ is independent of the starting state. The risk-neutral optimal policy is one that is associated with the maximum ρ.

2.2 ARM

The field of ARM revolves around maximizing the net revenues obtained by selling airline seats. Although it is essentially a problem of pricing, the pricing control is exercised via capacity control. In other words, the set of allowable prices is fixed beforehand, so that each price is said to belong to a "price class" or simply "class," and limits are set to the number of seats available for each class. The ARM problem is to find the optimal values for these limits. This is also called the seat-allocation problem.

Typically, there are significant differences between the demands of the customers, and some customers demand more expensive tickets in return for special features, such as cheaper cancelation fees or fewer stopovers. Airline companies prepare forecasts of the number of passengers expected for each price class. Some passengers cancel tickets, however, and hence airlines "overbook" planes to reduce the probability of flying with empty seats. The additional question that arises then is: by how much should the company overbook the plane? Customer arrivals during the booking period is a random process, and so is the cancelation process. When the plane takes off, if a confirmed passenger does not get a ticket, the airline has to pay a penalty. Selling too many seats in the lower fare classes does not make business sense, while setting very low booking limits for the lower fare classes usually ensures that the plane is not full at takeoff.

The EMSR (Expected marginal seat revenue) heuristic [16, 2] is widely used in the industry for solving this problem. There is now a vast literature on the ARM problem; we do not cite all the seminal works but refer the reader to a textbook [23]. The ARM problem can be set up as an infinite horizon SMDP by making the system state transition to the start of the booking horizon when the plane takes off. This approach has been used in [13, 11], which used ADP techniques based on temporal differences and modified policy iteration.

The SMDP state can be modeled approximately with the following $(2n+2)$-tuple: $\langle c, t, s_1, s_2, \ldots, s_n, \psi_1, \psi_2, \ldots, \psi_n \rangle$ where c denotes the class of the current customer, t denotes the time remaining for the flight departure, n denotes the number of classes, s_i denotes the number of seats sold in the ith class, and ψ_i is in itself an s_i-tuple that contains the times of arrival (in the booking horizon) of the passengers in the ith class. We will use a scheme to map this state space into a more tractable state space in our numerical computations. The action space is $(Accept, Reject)$. We now present the AC algorithm developed to solve the ARM problem.

2.3 AC algorithm and a benchmark

We present the steps in the main algorithm and then a benchmark based on value iteration. Both will use the vanishing discount approach. We will refer to the main algorithm as SM-DCAC (*S*emi-*M*arkov *D*iscounted *C*ost *A*ctor *C*ritic) algorithm. SM-DCAC is a form of modified policy iteration in which the policy is evaluated with only one iteration. See [15] for more on this issue.

2.3.1 Steps in SM-DCAC

Step 1. For all l, where $l \in \mathcal{S}$, and $u \in \mathcal{A}(l)$, set $J(l) \leftarrow 0$ and $Q(l,u) \leftarrow 0$. Set k, the number of state changes, to 0. Set ρ, the estimate of the average reward per transition, to 0, and $\bar{\gamma}$ to a value close to 0, e.g., 0.01. Set q_0 to a large, computer-permissible, positive value. Run the algorithm for $k_{\max}$ iterations, where $k_{\max}$ is chosen to be a sufficiently large number. Start system simulation at any arbitrary state.
Step 2. Let the current state be i. Select action a with a probability of $\exp(Q(i,a))/\sum_{b\in\mathcal{A}(i)}\exp(Q(i,b))$. (This is called the Gibbs softmax method).

Step 3. Simulate action a. Let the next state be j. Let $r(i,a,j)$ be the immediate reward earned in going to j from i under a and $t(i,a,j)$ the time in the same transition. Set $k \leftarrow k+1$ and update J as follows:

$$J(i) \leftarrow (1-\mu)J(i) + \mu\left[r(i,a,j) + \exp(-\bar{\gamma}t(i,a,j))J(j)\right]. \tag{1}$$

Step 4. Update $Q(i,a)$ using a step size, η, decayed slower than μ:

$$Q(i,a) \leftarrow Q(i,a) + \eta\left[r(i,a,j) + \exp(-\bar{\gamma}t(i,a,j))J(j) - J(i)\right]. \tag{2}$$

If $Q(i,a) < -q_0$, set $Q(i,a) = -q_0$. And if $Q(i,a) > q_0$, set $Q(i,a) = q_0$. In general, the step-sizes should satisfy the following rule [6]: $\limsup_{k\to\infty} \eta^k/\mu^k = 0$.
Step 5. If $k < k_{max}$, set $i \leftarrow j$ and then go to Step 2. Otherwise, go to Step 6.
Step 6. For each $l \in \mathcal{S}$, select $d(l) \in \arg\max_{b\in\mathcal{A}(l)} Q(l,b)$. The policy (solution) generated by the algorithm is $\hat{d}$. Stop.

Convergence of SM-DCAC: Let the (i,a)th element of the vector $\vec{Q}$ equal $Q(i,a)$. It is clear from the above description that a policy is associated with a given $\vec{Q}$; the value function associated with $\vec{Q}$ will be denoted by $\vec{J}_{\vec{Q}}$. The following result shows that SM-DCAC will converge to a stable solution.

Theorem 1 *For any q_0, the limits $\lim_{k\to\infty} \vec{J}^k$ and $\lim_{k\to\infty} \vec{Q}^k$ for SM-DCAC exist with probability 1.*

Proof The result follows directly from Lemma 4.10 in [15]. ∎

The following result shows that the solution to which the algorithm will converge will in fact be ϵ-optimal.

Theorem 2 *If $\vec{J}^*_{\bar{\gamma}}$ denotes the optimal value function, then with probability 1, there exists a large enough q_0 such that for any $\epsilon > 0$, $|\vec{J}_{\vec{Q}^\infty} - \vec{J}^*_{\bar{\gamma}}| < \epsilon$.*

Proof The proof follows from mimicking that of Theorem 5.13 in [15]; the only point to be noted is that their proof is for the MDP, and hence γ of their proof needs to be replaced by

$$\gamma_{\max} \equiv \max_{i\in\mathcal{S},a\in\mathcal{A}(i)} \sum_{j\in\mathcal{S}} p(i,a,j)\exp(-\bar{\gamma}t(i,a)). \quad \blacksquare \tag{3}$$

2.3.2 Steps in the value-iteration benchmark

This is based on the algorithm in [10], which is for SMDPs; for continuous-time MDPs, see the algorithm in [8]. We will refer to the algorithm below as SM-DISC, short for *S*emi-*M*arkov *Di*scounted *C*ost algorithm. The steps have the following differences with the steps in Section 2.3.1. The function J will not be needed here.
1. In Step 2, select action a with probability of $1/|\mathcal{A}(i)|$.
2. In Step 3, ignore the update of $J(.)$, and in Step 4, update $Q(i,a)$ as follows:

$$Q(i,a) \leftarrow (1-\mu)Q(i,a) + \mu\left[r(i,a,j) + \exp(-\bar{\gamma}t(i,a,j)) \max_{b\in\mathcal{A}(j)} Q(j,b)\right]. \tag{4}$$

Convergence of SM-DISC: The update in Eqn. (4) belongs the family:

$$Q^{k+1}(i,a) = (1-\mu)Q^k(i,a) + \mu\left[H(\vec{Q}^k)(i,a) + w_j^k(i,a)\right], \text{ where}$$

$$H(\vec{Q}^k)(i,a) = \sum_{j\in\mathcal{S}} p(i,a,j)\left[r(i,a,j) + \exp(-\bar{\gamma}t(i,a,j))\max_{b\in\mathcal{A}(j)} Q^k(j,b)\right] \text{ and} \tag{5}$$

$w_j^k(i,a) = r(i,a,j) + \exp(-\bar{\gamma}t(i,a,j))\max_{b\in\mathcal{A}(j)} Q^k(j,b) - H(\vec{Q}^k)(i,a)$. The Bellman equation of interest here is:

$$\vec{Q} = H(\vec{Q}). \tag{6}$$

Theorem 3 *If $t(i,a,j) > 0$ for all $i,j \in \mathcal{S}$ and all $a \in \mathcal{A}(i)$, and for $\gamma_{\max} > 0$, the algorithm SM-DISC will converge to an optimal solution, i.e., a solution of the Bellman equation in (6), with probability 1.*

Proof We first show that the transformation H in (5) is contractive. From (5) and by using definition in (3), for any (i,a),

$$\begin{aligned}|H(\vec{Q}_1^k)(i,a) - H(\vec{Q}_2^k)(i,a)| &\leq \gamma_{\max}|\max_{b\in\mathcal{A}(j)} Q_1^k(j,b) - \max_{b\in\mathcal{A}(j)} Q_2^k(j,b)| \\ &\leq \gamma_{\max}||\vec{Q_1}^k - \vec{Q_2}^k||_\infty.\end{aligned}$$

Then it follows that $||H(\vec{Q_1}^k) - H(\vec{Q_2}^k)||_\infty \leq \gamma_{\max}||\vec{Q_1}^k - \vec{Q_2}^k||_\infty$, where $0 < \gamma_{\max} < 1$ when any $t(i,a,j) > 0$, and hence H has a unique fixed point. Hence H is also non-expansive.

Again, from (5) and by using definition in (3), we note that for any (i,a)

$$|H(\vec{Q}^k)(i,a)| \leq \gamma_{\max}|Q^k(i,a)| + \max_{i,a}|\bar{r}(i,a)|$$

which implies that $||H(\vec{Q}^k)||_\infty = \gamma_{\max}||\vec{Q}^k||_\infty + D$ where $D > 0$ and $0 < \gamma_{\max} < 1$ when any $t(i,a,j) > 0$. This, along with standard stochastic approximation conditions common in ADP, from Proposition 4.7 in [3] establishes that $Q^k(,.,)$ will remain bounded with probability 1. Thus the iterate is bounded with probability 1 and H is non-expansive. Then from Theorem 3.1(b) in [7], the algorithm should converge with probability 1 to a fixed point of H. Since H is contractive, it has a unique fixed point, and hence the algorithm converges to that fixed point, which is in fact the optimal solution. ∎

We used the vanishing discount approach for the average-reward case, which implies that the discounted algorithm will converge to average-reward optimality as $\bar{\gamma}$ tends to 0.

3 Risk-sensitive AC algorithms

We now propose a variance-sensitive version of the AC algorithm presented above. We first define: $\gamma_{\hat{\mu}}(i) \equiv \lim_{k\to\infty} \frac{E_{\hat{\mu}}\left[\sum_{s=1}^k \bar{r}^2(x_s,\mu(x_s))|x_1=i\right]}{k}$. Then, from Theorem 1 of [12], the corresponding long-run variance of rewards of the policy $\hat{\mu}$, starting at state i, is: $\sigma^2_{\hat{\mu}}(i) = \frac{\gamma_{\hat{\mu}}(i)}{\beta_{\hat{\mu}}(i)} - \frac{(\alpha_{\hat{\mu}}(i))^2}{\beta_{\hat{\mu}}(i)}$. It can be shown that $\sigma^2(.)$ is independent of the starting state for irreducible and aperiodic Markov chains. The optimal policy for a variance-sensitive (risk-sensitive) problem, according to [9], is one that maximizes $\rho_{\hat{\mu}} - \theta\sigma^2_{\hat{\mu}}$.

3.1 Vanishing discount procedure

There are two approaches to this: Method (I) and Method (II).

Method (I): Here we estimate ρ, the average reward per transition. Change the steps in Section 2.3.1 as follows.
1. In Step 1, let ρ be the estimate of average reward per transition, and initialize it to 0.
2. In Step 3, use the following update for $J(i)$.

$$J(i) \leftarrow (1-\mu)J(i) + \mu\left[r(i,a,j) - \theta\left(r(i,a,j)-\rho\right)^2 + \exp(-\bar{\gamma}t(i,a,j))J(j)\right].$$

In Step 3, also update ρ as follows.

$$\rho \leftarrow (1-\mu)\rho + \mu\left[\frac{r(i,a,j)+k\rho}{k+1}\right]. \tag{7}$$

3. In Step 4, update $Q(i,a)$ as follows:

$$Q(i,a) \leftarrow Q(i,a) + \eta\left[r(i,a,j) - \theta\left(r(i,a,j)-\rho\right)^2 + \exp(-\bar{\gamma}t(i,a,j))J(j) - J(i)\right].$$

Method (II): Here we also estimate the expected time of each transition and use it in the calculations. Change the steps in Section 2.3.1 as follows.
1. In Step 1, let τ be the estimate of the expected transition time. Initialize it to a small positive value. Also, initialize ρ to 0.
2. In Step 3, use the following update for $J(i)$.

$$J(i) \leftarrow (1-\mu)J(i) + \mu \times \left[r(i,a,j) - \theta\left(r(i,a,j) - \frac{\rho}{\tau}t(i,a,j)\right)^2 + \exp(-\bar{\gamma}t(i,a,j))J(j)\right].$$

In addition to updating ρ (as shown in Method (I) above), also update τ as follows:

$$\tau \leftarrow (1-\mu)\tau + \mu\left[\frac{t(i,a,j)+k\tau}{k+1}\right]. \tag{8}$$

3. In Step 4, update $Q(i,a)$ as follows:

$$Q(i,a) \leftarrow Q(i,a) + \eta\Big[r(i,a,j) - \theta\left(r(i,a,j) - \frac{\rho}{\tau}t(i,a,j)\right)^2 + \exp(-\bar{\gamma}t(i,a,j))J(j) - J(i)\Big].$$

Convergence of Method (I): We consider the MDP case (with γ as the discounting factor). The result can be extended easily to SMDPs. We also show that there is a bound on the difference between the optimal value function and the value function generated by the algorithm.

Theorem 4 *The limits* $\lim_{k\to\infty}\vec{J}^k$ *and* $\lim_{k\to\infty}\vec{Q}^k$ *for the risk-sensitive algorithm above exist with probability 1 for any* q_0.

Proof The result follows directly from Lemma 4.10 in [15]. ∎

We now present a bound on the value function. It is based on the ideas in [15], but, unfortunately, we cannot make it ϵ-small as done in their elegant proof. The reason is the existence of ρ^*, the average reward of the optimal solution, in our equations. Let ρ_π denote the average reward of the policy $\hat{\pi}$; then we define the following terms:

$$w^*(i,a) = \theta \sum_j p(i,a,j) \left(r(i,a,j) - \rho^*\right)^2,$$

$$w^\pi(i,a) = \theta \sum_j p(i,\pi(i),j) \left(r(i,\pi(i),j) - \rho_\pi\right)^2.$$

which allows to define the Bellman equation of interest here

$$J^*_\gamma(i) = \max \left[\bar{r}(i,a) - \theta w^*(i,a) + \gamma \sum_j p(i,a,j) J^*_\gamma(j) \right] \quad \forall i.$$

The vector J^*_γ will denote the optimal value function. This equation can be solved, provided ρ^* is known, and has a unique solution. Also, if $\pi(i,a)$ denotes the probability of selecting action a in state i when policy $\hat{\pi}$ is pursued, then consider the following equation: $\forall i$

$$J_\pi(i) = \sum_a \pi(i,a) \left[\bar{r}(i,a) - w^\pi(i,a) + \gamma \sum_j p(i,a,j) J_\pi(j) \right].$$

Theorem 5 $||\vec{J}_\pi - \vec{J}^*_\gamma||_\infty \leq K\frac{2}{(1-\gamma)^2}$.

Proof It is clear that $\max_i |J_\pi(i)| \leq \bar{r}(i,a) - w^\pi(i,a) + \gamma \max_j |J_\pi(j)|$, which implies that $||\vec{J}_\pi||_\infty \leq \frac{K}{1-\gamma}$ where $K = \max_{i,a,\hat{\pi}} |\bar{r}(i,a) - w^\pi(i,a)|$. We now define the following transformation on vector $\vec{x}$: $\forall i$,

$$T(x(i)) = \max_a \left[\bar{r}(i,a) - w^*(i,a) + \gamma \sum_j p(i,a,j) x(j) \right].$$

It is easy to extend Lemma 5.11 in [15] to establish that:

$$||\vec{J}_\pi - \vec{J}^*_\gamma||_\infty \leq (1-\gamma)^{-1} ||T\vec{J}_\pi - \vec{J}_\pi||_\infty. \tag{9}$$

Then, it follows that for every i,

$$\begin{aligned}
|TJ_\pi(i) - J_\pi(i)| &= |\max[\bar{r}(i,a) - w^*(i,a) + \gamma \sum_j p(i,a,j) J^*_\gamma(j) - \vec{J}_\pi(i)]| \\
&\leq |\bar{r}(i,a) - w^*(i,a)| + \gamma |\sum_j p(i,a,j) J^*_\gamma(j)| + |\vec{J}_\pi(i) \\
&\leq K + \frac{\gamma K}{1-\gamma} + \frac{K}{1-\gamma} = K\frac{2}{1-\gamma}.
\end{aligned}$$

It follows that $||T\vec{J}_\pi - \vec{J}_\pi||_\infty \leq K\frac{2}{1-\gamma}$. Then this combined with (9) implies that $||\vec{J}_\pi - \vec{J}^*_\gamma||_\infty \leq K\frac{2}{(1-\gamma)^2}$. ∎

3.2 Undiscounted procedure

We now suggest an approach that does not use a discount factor. It requires in each iteration the estimate of the variance, something that is avoided in the vanishing discount procedure. The changes to the steps in Section 2.3.1 are as follows:
1. In Step 1, initialize each of ϱ and ρ to 0 and τ to a small positive value.
2. In Step 3, define $\psi = \frac{\rho}{\tau} - \theta \frac{[\varrho - \rho^2]}{\tau}$. Then update $J(i)$ as follows:

$$J(i) \leftarrow (1-\mu)J(i) + \mu \left[r(i,a,j) - \theta \left(r(i,a,j) - \rho \right)^2 - \psi t(i,a,j) + J(j) \right].$$

In addition to updates of ρ (Eqn. (7)) and τ (Eqn. (8)), also update ϱ as follows:

$$\varrho \leftarrow (1-\mu)\varrho + \mu \left[\frac{[r(i,a,j)]^2 + k\varrho}{k+1} \right].$$

3. In Step 4, update $Q(i,a)$ using ψ as defined above:

$$Q(i,a) \leftarrow Q(i,a) + \eta \left[r(i,a,j) - \theta \left(r(i,a,j) - \rho \right)^2 - \psi t(i,aj) + J(j) - J(i) \right].$$

By setting $\theta = 0$, one obtains a *S*emi-*M*arkov *A*verage *C*ost *A*ctor *C*ritic (SM-AC^2) algorithm.

4 Numerical results

4.1 ARM

We present one numerical example as a vehicle to compare the performance of SM-DCAC, SM-DISC, and EMSR (the heuristic; for more details on this, see [14]).

- The flight has three fare classes, and the fare structure is $FS = (f_1, f_2, f_3, b)$, where f_i is the fare of the ith class. $f_1 = \$199, f_2 = \$275, f_3 = \$350$. \$400 is the bumping cost. A lower value of i stands for a lower revenue fare class.
- The customer arrival process is Poisson with parameter 1.4 passengers per day, and the booking horizon is 100 days long. Probability that the customer is of class 1 is 0.7, and that he/she is of class 2 is 0.2.
- The capacity of the aircraft is 100.
- Every customer belonging to class i has a probability of p_c^i of canceling the trip and the time of cancelation is uniformly distributed between the time of sale and the time of flight departure. $p_c^1 = 0.1; p_c^2 = 0.2; p_c^3 = 0.3$.
- The cancelation penalties are: Class 1 (lowest fare): \$100, Class 2 (medium fare):\$ 50, and Class 3 (highest fare): \$ 10.
- The function approximation scheme used is described in [11]. We used the following rules for the step sizes: $\mu^k = 0.01\frac{\log(k+1)}{k+1}; \eta^k = \frac{0.01}{k+1}$.

The results are shown in Table 1.

Table 1: Sample problem showing a comparison of the average revenue (dollars per day) using EMSR, SM-DCAC, and SM-DISC.

ρ_{EMSR}	$\rho_{SM-DCAC}$	$\rho_{SM-DISC}$
211.11	220.45	217.10

4.2 Small Risk-Sensitive MDPs

Consider a 2-state MDP in which two actions are allowed in each state. Additionally, all the Markov chains are regular. $\mathbf{P(a)}$ will denote the transition probability matrix associated with action a, and $\mathbf{R(a)}$ will denote the transition reward matrix associated with action a. The element in the ith row and jth column of $\mathbf{P(a)}$ will equal $p(i, a, j)$, and the corresponding element in $\mathbf{R(a)}$ will equal $r(i, a, j)$.

$$\mathbf{P}(1) = \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix}; \mathbf{P}(2) = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix};$$

$$\mathbf{R}(1) = \begin{bmatrix} 6.0 & -5 \\ 7.0 & 12 \end{bmatrix}; \mathbf{R}(2) = \begin{bmatrix} 5.0 & 68 \\ -2 & 12 \end{bmatrix}.$$

Policy	ρ	σ^2
(1,1)	5.828571	30.142041
(1,2)	8.625000	31.284375
(2,1)	**11.04000**	**287.23840**
(2,2)	10.95000	187.54750

Table 2: The average reward and variance (σ^2) of all policies associated are shown above. The risk-neutral-optimal policy has its parameters typed in bold. Note that it also has the highest variance.

We used $\theta = 0.15$. The optimal action for the risk-sensitive problem is 1 in state 1 and 2 in state 2. This can be determined by exhaustive enumeration. See Table 2 that lists the mean and variance via exhaustive enumeration. Method (I) and the undiscounted procedure converged to the optimal solution in a maximum of 100 iterations. The step-size rules used were: $\mu^k = \frac{\log(k+1)}{k+1}; \eta^k = \frac{0.001}{k+1}$.

5 Conclusions and future work

AC algorithms have been restricted to MDPs and risk-neutral objective functions. The main contribution here is to present an AC algorithm for (i) the SMDP model that can solve the ARM problem and (ii) for risk-sensitive objective functions. We showed convergence for the AC algorithm (SM-DCAC), and also a convergence analysis for a value-iteration-based algorithm for the SMDP (SM-DISC). In addition, we also presented an average reward AC algorithm for SMDPs (SM-AC^2). We showed some numerical results with our algorithm using the SM-DCAC and SM-DISC, both of which used a simple function approximation scheme. We wish to expand the scope of the numerical study. In future (i) we intend to test the risk-sensitive algorithms on risk-sensitive versions of the ARM problem, (ii) prove convergence of SM-AC^2, and (iii) use a neural-network-based function-approximation scheme described next. It will be interesting to use this algorithm on the network version of this problem where risk-neutral ADP has already been used and tested [20, 21].

Function approximation for the AC framework can be done using the Bellman error method proposed in Werbos [25, 26]. We will assume that the J-values and Q-values will be stored in artificial neural networks (ANNs), i.e., $J(i, \vec{w_1})$ and $\tilde{Q}(i, a, \vec{w_2})$ will denote the J-value and Q-value, respectively, where $\vec{w_1}$ and $\vec{w_2}$ denote the neural-network weights associated with J and Q respectively. In the context of the algorithm, then, there are two types of errors to minimize.

$$BE_1 = \frac{1}{2}\sum_i \left[J(i) - \tilde{J}(i, \vec{w_1})\right]^2 \text{ and } BE_2 = \frac{1}{2}\sum_{i,a} \left[Q(i,a) - \tilde{Q}(i, a, \vec{w_2})\right]^2.$$

The generalized gradient-descent algorithm within the ANN will be: $w(l) \leftarrow w(l) + \mu\frac{\partial BE}{\partial w}$ for the lth weight. Using calculus, the *single-sample* update in Step 3 for updating the J-values becomes $\forall l$:

$$w_1(l) \leftarrow w_1(l) + \mu\nabla\tilde{J}(i, \vec{w_1}) \times \Big(r(i,a,j) + \exp(-\bar{\gamma}t(i,a,j))\tilde{J}(j, \vec{w_1})\Big).$$

And that for updating the Q-values in Step 4 becomes $\forall l$:

$$w_2(l) \leftarrow w_2(l) + \eta\nabla\tilde{Q}(i, a, \vec{w_2}) \times \Big(r(i,a,j) + \exp(-\bar{\gamma}t(i,a,j))\tilde{J}(j, \vec{w_1}) - \tilde{J}(i, \vec{w_1})\Big).$$

Also, the values of $\nabla\tilde{J}(i, \vec{w_1})$ and $\nabla\tilde{Q}(i, a, \vec{w_2})$ depend on the ANN architecture.

ACKNOWLEDGMENTS: The author thanks Dr. Paul J. Werbos at the National Science Foundation for enlightening discussions on adaptive critics and for pointing out some key references — in particular [27] — that led to this work.

References

[1] A.G. Barto, R.S. Sutton, and C.W. Anderson. Neuronlike elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:835–846, 1983.

[2] P.P. Belobaba. Application of a probabilistic decision model to airline seat inventory control. *Operations Research*, 37:183–197, 1989.

[3] D.P. Bertsekas and J.N. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, Belmont, MA, 1996.

[4] V. Borkar. Q-learning for risk-sensitive control. *Mathematics of operations research*, 27(2):294–311, 2002.

[5] V. Borkar and S. Meyn. Risk sensitive optimal control: existence and synthesis for models with unbounded cost. *Mathematics of Operations Research*, 27(1):192–209, 2002.

[6] V. S. Borkar. Stochastic approximation with two-time scales. *Systems and Control Letters*, 29:291–294, 1997.

[7] V. S. Borkar. Asynchronous stochastic approximation. *SIAM J. Control Optim.*, 36 No 3:840–851, 1998.

[8] S.J. Bradtke and M. Duff. Reinforcement learning methods for continuous-time Markov decision problems. In *Advances in Neural Information Processing Systems 7*. MIT Press, Cambridge, MA, 1995.

[9] J. Filar, L. Kallenberg, and H. Lee. Variance-penalized Markov decision processes. *Mathematics of Operations Research*, 14(1):147–161, 1989.

[10] A. Gosavi. *Simulation-Based Optimization:Parametric Optimization Techniques and Reinforcement Learning*. Kluwer Academic Publishers, Boston, MA, 2003.

[11] A. Gosavi. A reinforcement learning algorithm based on policy iteration for average reward: Empirical results with yield management and convergence analysis. *Machine Learning*, 55(1):5–29, 2004.

[12] A. Gosavi. A risk-sensitive approach to total productive maintenance. *Automatica*, 42:1321–1330, 2006.

[13] A. Gosavi, N. Bandla, and T. K. Das. A reinforcement learning approach to a single leg airline revenue management problem with multiple fare classes and overbooking. *IIE Transactions*, 34:729–752, 2002.

[14] A. Gosavi, E. Ozkaya, and A. Kahraman. Simulation optimization for revenue management of airlines with cancellations and overbooking. *OR Spectrum*, 29:231–38, 2007.

[15] V.R. Konda and V. S. Borkar. Actor-critic type learning algorithms for Markov decision processes. *SIAM Journal on Control and Optimization*, 38(1):94–123, 1999.

[16] K. Littlewood. Forecasting and control of passenger bookings. *In Proceedings of the 12th AGIFORS (Airline Group of the International Federation of Operational Research Societies) Symposium)*, pages 95–117, 1972.

[17] H Markowitz. Portfolio selection. *Journal of Finance*, 7(1):77–91, 1952.

[18] S. M. Ross. *Applied Probability Models with Optimization Applications*. Dover, 1992.

[19] S. Shervais and T. T. Shannon. Improving quasi-optimal inventory and transportation policies using adaptive critic based apprximate dynamic programming. In *Proceedings of 2000 International Conference on Systems, Man, and Cybernetics, Nahsville, TN*, 2000.

[20] V. Singh. A stochastic approximation approach to an airline network revenue management problem. Master's thesis, University of South Florida, Dept. of Industrial and Management Systems Engineering, Tampa, FL, 2002.

[21] V. Singh, T.K. Das, A. Gosavi, and B. Mohan. Algorithms for airline network seat revenue management via reinforcement learning. *Working paper at the University of South Florida, Industrial and Management Systems Engineering Dept, Tampa, FL.*

[22] R. Sutton and A. G. Barto. *Reinforcement Learning.* The MIT Press, Cambridge, Massachusetts, 1998.

[23] K. Talluri and G. van Ryzin. *The Theory and Practice of Revenue Management.* Kluwer Academic, Boston, MA, 2004.

[24] G. Venayagamoorthy, R. Harley, and D. Wunsch. Comparison of heuristic dynamic programming and dual heuristic programming adaptive critics for neuro-control of a turbogenerator. *IEEE Transactions on Neural Networks*, 13 (3):764–773, 2002.

[25] P. J. Werbös. Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research. *IEEE Transactions on Systems, Man., and Cybernetics*, 17:7–20, 1987.

[26] P. J. Werbös. Consistency of HDP applied to a simple reinforcement learning problem. *Neural Networks*, 3:179–189, 1990.

[27] P. J. Werbös. Approximate dynamic programming for real-time control and neural modeling. In *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, pages 493–525. Van Nostrand, NY, 1992.