Intelligent Inventory Control: Is Bootstrapping Worth Implementing?

Tatpong Katanyukul[1], Edwin K. P. Chong[2], William S. Duff[3]

[1] Embedded System Research Group and Department of Computer Engineering, Faculty of Engineering, Khon Kaen University, Khon Kaen, Thailand
tatpong@kku.ac.th
[2] Department of Electrical and Computer Engineering, College of Engineering, Colorado State University, Fort Collins, CO, USA
edwin.chong@colostate.edu
[3] Department of Mechanical Engineering,College of Engineering, Colorado State University, Fort Collins, CO, USA
bill@engr.colostate.edu

Abstract. The common belief is that using Reinforcement Learning methods (RL) with bootstrapping gives better results than without. However, inclusion of bootstrapping increases the complexity of the RL implementation and requires significant effort. This study investigates whether inclusion of bootstrapping is worth the effort when applying RL to inventory problems. Specifically, we investigate bootstrapping of the temporal difference learning method by using eligibility trace. In addition, we develop a new bootstrapping extension to the Residual Gradient method to supplement our investigation. The results show questionable benefit of bootstrapping when applied to inventory problems. Significance tests could not confirm that bootstrapping had statistically significantly reduced costs of inventory controlled by a RL agent. Our empirical results are based on a variety of problem settings, including demand correlations, demand variances, and cost structures.

Keywords. approximate dynamic programming, inventory control, reinforcement learning, bootstrapping, eligibility trace, intelligent agent

1 Introduction

Inventory management is one of the major business activities. A well managed inventory can help a business stay competitive by keeping its cash flow at a controllable level. A stochastic multiperiod inventory problem—one of the most common inventory problems—can be modeled as a Markov Decision Problem (MDP). Approximate Dynamic Programming (ADP) is a method to solve practical Markov decision problems. Most previous studies on ADP applied to inventory management focused on learning scheme, which is also known as Reinforcement Learning (RL). Due to its effectiveness and its link to mammal learning processes, temporal difference learning, or sometimes called “one-step temporal-difference learning”, TD(0), is one of the most widely studied RL methods.

Eligibility trace has been used to bootstrap the learning process of TD(0). The integration of the eligibility trace into the temporal difference learning leads to the TD(λ) method. The success of TD(λ) in many applications has lead to a general belief (see, e.g., Prestwich et al. (2008)) that using TD(λ) with λ between 0 and 1 will give better results than TD(0). However, the effort required to implement TD(λ) is considerably higher than to implement TD(0). The value and benefit of using TD(λ) compared to TD(0) have never been investigated especially for inventory management.

Our study investigates the application of Sarsa(λ), a widely studied implementation of the TD(λ) method. To supplement the investigation, our study develops the Direct Credit Back method (DCB)—the bootstrapped version of the Residual Gradient method (RG). Finally, we evaluate both bootstrapping methods—Sarsa(λ) and DCB—and confer to their non-bootstrapped counterparts, Sarsa and RG. The underlying methods are evaluated with various structures of inventory problems. Two other inventory control methods—Look-Ahead and Rollout—are also included in our study.

The findings here provide a practical approach to apply an ADP method for an inventory problem. The results reveal questionable benefits of bootstrapping. The understanding exposed here will help promote efficient inventory management and aid in the transfer of intelligent system research into practice.

2 Literature Review

ADP has been introduced recently into inventory management research. Kim et al. (2005), Kim et al. (2008), Kwon et al. (2008), and Jiang and Sheng (2009) implemented TD(0) with a look-up table as a cost-to-go approximation—a crucial component of most RL methods. A look-up table is simple to implement and works well with TD(0). However, a look-up table suffers from a scalability issue: its memory requirement grows exponentially as the dimension of the problem space grows.

Some studies, e.g., Van Roy et al. (1997) and Shervais et al. (2003), apply TD(0) with nonlinear function approximation, such as artificial neural network, to mitigate a scalability issue. However, applying TD(0) with nonlinear function approximation requires a high level of expertise in both application and techniques and it might result in divergence leading to instability of the control. Baird (1995) proposed Residual Gradient method (RG) to be used with nonlinear function approximation. However, RG is reported to underperform TD(0) in overall.

In addition to extend RL method with scalable function approximation, bootstrapping is used to speed up the learning process of TD(0) and leads to a more general method, denoted TD(λ). Successes of applying TD(λ)—the most famous work is Tesauro (1994)—lead to common belief of the benefit of bootstrapping. However, due to complexity of implementation and extra computational costs, there is only work of Prestwich et al. (2008) related to bootstrapping and inventory management. Prestwich et al. (2008) studied the viability of combining Sarsa(λ) and Noisy Genetic Algorithm by using the Cultural Algorithm, introduced by Reynolds (1994). They claimed the viability of their approach for partially observable Markov decision problems.

Among previous authors applying ADP to inventory problems, many authors have used TD(0) or methods related to TD(0), but none[1] has investigated the effectiveness of bootstrapping. Leng et al. (2009) also mentioned that the mechanism of Eligibility Trace in ADP with function approximation has not been sufficiently investigated.

Inspired by the development of eligibility trace to bootstrap TD(0), we develop an extension to RG based on bootstrapping, called "Direct Credit Back" (DCB). The new method DCB provides a contribution in its own right as well as supplements the investigation of an effect of bootstrapping in applying RL to inventory management.

3 Background

Sarsa. Sarsa, as discussed by Sutton and Barto (1998), uses an approximate state-action cost $Q(s_t, a_t)$, often called the Q-value, to determine an action and updates $Q(s_t, a_t)$ based on TD(0). The equations for Sarsa are $Q(s,a) \leftarrow Q(s,a) + \beta\, \psi(s,a)$ where $Q(s, a)$ approximates the state-action cost of state s and action a; β is a Sarsa parameter, called the learning rate; the temporal difference $\psi(s,a) = c(s, a) + \alpha\, Q(s', a') - Q(s, a)$ when $c(s, a)$ is a single period cost and action a_t is determined by a chosen policy based on state s_t and the current Q-value.

Eligibility Trace. TD(0) uses newly observed information, $c(s_t, a_t)$, to update a Q-value of the most recent state-action. To utilize trajectory information, Eligibility Trace (ET) is developed. ET[2] updates its values, such that when (s, a) is visited, eligibility variable $e(s, a) \leftarrow 1$, $e(s, \hat{a}) \leftarrow 0$ for each action $\hat{a} \neq a$, and $e(\hat{s}, b) \leftarrow \alpha\, \lambda\, e(\hat{s}, b)$ for each state $\hat{s} \neq s$ and each action $b \in A(\hat{s})$ where α is a discount factor; λ is an eligibility factor; and $A(\hat{s})$ is a feasible action set for the given state $\hat{s}$. Sarsa(λ) is an implementation of TD(λ), whose the update equation is $Q(s, a) \leftarrow Q(s, a) + \beta\, \psi(s, a)\, e(s, a)$ for all s and a.

Residual Gradient Method. Sarsa was originally designed to be implemented using a look-up table and later was extended for use with an approximation function. Although there are some success, associated risk of instability is commonly known, see Baird (1995), Barreto and Anderson (2008), and Maei et al. (2009). The Residual Gradient method (RG), introduced by Baird (1995), is designed to be used with various approximation functions, including ones belonging to a non-linear family. RG is developed to minimize the approximation error[3] $\xi(\theta) = \sum_{t = 1,\ldots,T} \xi_t(\theta)$ where $\xi(\theta)$ is the total approximation error and $\xi_t(\theta) = \{C(s_t) - Q(s_t\,|\theta)\}^2/2$ is the approximation error of period t. Therefore, the parameter values can be determined by gradient descend method: $\theta \leftarrow \theta - \beta \cdot \{ C(s_t) - Q(s_t|\theta) \}\ \nabla_\theta \{ C(s_t) - Q(s_t\,|\theta) \}$. In the development of Sarsa, partial differentiation is applied, then the real state cost $C(s_t)$ is approximated by $c(s_t) + \alpha\, Q(s_{t+1})$. In development of RG, the approximation is applied before differentiation. The RG update equation is $\theta \leftarrow \theta + \beta \cdot \psi(s_t) \cdot \{ \alpha\, \nabla_\theta Q(s_{t+1}|\theta)$

[1] Prestwich et al. (2008) used Sarsa(λ) only as a competing method with hill-climbing method to determine the parameter values of Sarsa(λ), but have not investigated the effectiveness of bootstrapping.

[2] ET presented here is based on Singh and Sutton (1996)'s replacing trace.

[3] To be concise, the content here is presented based on state costs. The development based on state-action costs can be conducted in a similar manner.

$-\ \nabla_\theta Q(s_t|\theta)\}$ where $\psi(s_t) = c(s_t) + \alpha Q(s_{t+1}|\theta) - Q(s_t|\theta)$.

4 Direct Credit Back

Baird (1995) claims that RG always converges. However, it has been criticized for delivering an inferior solution compared to TD(0) by Maei et al. (2009). To improve the solution obtained from RG, we develop a Direct Credit Back (DCB) method based on bootstrapping. The DCB method uses the newly observed datum $c(s_t)$ to update the approximate costs of the most recent state as well as other prior states.

The temporal difference error of the approximate cost of state s_t is

$$\psi(s_t) = c(s_t) + \alpha Q(s_{t+1}|\theta) - Q(s_t|\theta). \tag{1}$$

After the period cost $c(s_t)$ is observed, $Q(s_t|\theta)$ can be approximated by $c(s_t) + \alpha Q(s_{t+1}|\theta)$. Therefore, the temporal difference error of the approximate cost of state s_{t-1} after $c(s_t)$ is observed is defined as

$$\psi(s_{t-1}|c(s_t)) = c(s_{t-1}) + \alpha\ \{\ c(s_t) + \alpha\ Q(s_{t+1}|\theta)\ \} - Q(s_{t-1}|\theta). \tag{2}$$

Equation (2) can be rearranged as shown in Equation (3),

$$\psi(s_{t-1}|c(s_t)) = c(s_{t-1}) + \alpha\ \{\ \psi(s_t|c(s_t)) + Q(s_t|\theta)\ \} - Q(s_{t-1}|\theta) \tag{3}$$

where $\psi(s_t|c(s_t)) = \psi(s_t)$.
The update equations for other prior states can be obtained in a similar manner. After $c(s_t)$ is observed, the temporal difference error of approximate cost of state s_{t-i}, for $i = 1, 2, ..., t-1$, is shown in Equation (4):

$$\psi(s_{t-i}|c(s_t)) = c(s_{t-i}) + \alpha\ \{\ \psi(s_{t-i+1}|c(s_t)) + Q(s_{t-i+1}|\theta)\ \} - Q(s_{t-i}|\theta). \tag{4}$$

Minimizing the squared temporal difference errors of all prior states is equivalent to minimizing $\sum_{i=0,\ ...,\ t-1} \psi^2(\ s_{t-i}\ |\ c(s_t)\ )$. The DCB update equation can be obtained by the gradient descent method. The update can be truncated to only a specific number of prior approximate state costs,

$$\theta \leftarrow \theta - \beta \sum_{i=0,\ldots,\min\{t-1,\ N\}} \psi(s_{t-i}|c(s_t)) \cdot \{\ \alpha^{i+1} \cdot \nabla_\theta\ Q(s_{t+1}|\theta) - \nabla_\theta\ Q(s_{t-i}|\theta)\ \} \tag{5}$$

where $N \in \{0, 1, 2, ...\}$ is the number of periods crediting back. The update equation for approximate state-action costs can be obtained in a similar manner. When the parameter $N = 0$, the DCB method reduces to RG.

5 Experiments

Our study uses computer simulations to conduct numerical experiments. The inventory problem investigated here is a periodic review single-echelon problem with non-zero leadtime and a setup cost.

The demand is modeled as AR1/GARCH(1,1). Therefore, a state is composed of the previous demand D_{t-1}, the previous demand error ε_{t-1}, the variance of the previous demand error σ_{t-1}^2, the on-site inventory x_t, and the in-transit inventory $B^{(t)}$ whose length depends on a leadtime L. The state space of this problem is $\{0, I^+\} \times R \times R \times I \times \{0, I^+\}^L$ for D_{t-1}, ε_{t-1}, σ_{t-1}^2, x_t, and $B^{(t)}$, respectively. An action $u_t \in \{0, I^+\}$ is a replenishment order. State transitions are specified by (1) $D_t = a_0 + a_1 D_{t-1} + \varepsilon_t$, (2) $\varepsilon_t = e_t \sigma_t$, (3) $\sigma_t^2 = \nu_0 + \nu_1 \varepsilon_{t-1}^2 + \nu_2 \sigma_{t-1}^2$, (4) $x_{t+1} = x_t + B^{(t)}{}_1 - D_t$, and (5) $B^{(t+1)} = [B^{(t+1)}{}_1 \cdots B^{(t+1)}{}_{L-1}\, B^{(t+1)}{}_L]^T = [B^{(t)}{}_2 \cdots B^{(t)}{}_L\, u_t]^T$ where a_0 and a_1 are AR1 model parameters; ν_0, ν_1, and ν_2 are GARCH(1,1) parameters; e_t is white noise distributed according to $N(0, 1)$; and u_t is the replenishment order.

The period cost consists of the replenishment cost and the inventory handling cost, $c_{t+1} = K_t\, \delta(u_t) + g_t\, u_t + h_t\, (x_{t+1})^+ + b_t\, (-x_{t+1})^+$ where c_{t+1} is the period cost, whose value will be known at time $t + 1$ (the end of period t); K_t is the setup cost; g_t is the unit replenishment cost; h_t is the unit holding cost; b_t is the unit backlogging cost; $\delta(\cdot)$ is the step function; $(\cdot)^+$ is the positive function defined by $(a)^+ = a\ \delta(a)$; and other variables are as mentioned earlier.

Our study investigated 13 different inventory problem structures (as shown in Table 1) with other parameters held fixed: AR1's $a_0 = 2$, GARCH(1,1)'s $\nu_1 = 0.1$ and $\nu_2 = 0.8$, a discount factor $\alpha = 1$, a leadtime $L = 1$, and the unit replenishment cost $g =$ \$100/unit. Each experiment is initialized at $D_0 = 50$, $\varepsilon_0 = 10$, $\sigma_0^2 = 400$, $x_1 = 10$, and $B^{(1)} = 0$. Each method's performance is evaluated based on its aggregate cost of Periods 13–60. Using an aggregate cost allows ADP methods to have initial learning periods and thus provides better performance evaluation of ADP methods. The experiments run each inventory controller for 50 replications of each of 60 time-unit-indeterminate periods.

Implementation. Sarsa, Sarsa(λ), RG, and DCB methods are used with the Radial Basis Function (RBF) as the approximation function. RBF has three sets of parameters: centers, scales, and weights. RBF centers and scales are set up as Katanyukul et al. (2011). RBF weights are adjustable parameters whose values are determined by the method under investigated. Variables ε_{t-1} and σ_{t-1}^2 are excluded from state parameters for reasons discussed in Katanyukul et al. (2011).

Table 1. A summary of demand and cost variables investigated

	demand variables		cost variables		
	correlation (a_1)	noise variance offset (ν_0)	set up cost (K: \$/transaction)	penalty cost (b: \$/item)	holding cost (h: \$/item)
P1	0.8	70	100	200	1
P2	0	70	100	200	1
P3	-0.8	70	100	200	1
P4	0.8	40	100	200	1
P5	0.8	10	100	200	1
P6	0.8	70	150	100	1
P7	0.8	70	100	100	1
P8	0.8	70	50	100	1
P9	0.8	70	0	100	1

P10	0.8	70	0	200	1
P11	0.8	70	0	50	1
P12	0.8	70	0	50	10
P13	0.8	70	0	50	25

6 Experimental Results

We measure performance of each inventory control by aggregate costs. Figures 1 and 2 show averages and confidence intervals of aggregate costs obtained from different methods under problem scenarios P1–P13 (Table 1). A '*' marks an average aggregate cost and line beside it represents a 90% confidence interval, based on t-test statistics. On the y-axis, labels 'H', 'S', 'SL', 'RG', and 'DCB', and 'Roll' indicate results obtained from the 12-period Look-Ahead, Sarsa, Sarsa(λ), RG, and the DCB method, respectively[4]. The Roll-out method was used with perfect system information: Rollout simulation parameter values match those of the actual problem.

7 Discussion and Conclusions

The average costs obtained from RL methods with bootstrapping are lower than without in most scenarios, but P13 that the average cost obtained from DCB is higher than one from RG method. Figures 3 and 4 illustrate the effect of bootstrapping. The y-axes show the percentage relative cost difference of method A to method B, denoted "% rel. cost diff.", which is $(cost_A/cost_B - 1) \times 100\%$. The x-axes show scenario parameters as indicated: demand correlations (a_1), variances (v_0), set up per unit cost (K/g), penalty cost per unit cost (b/g), and holding cost per penalty cost (h/b). The plots in the first columns show % rel. cost diff. of each RL method, as indicated by the legends, to the 12-period Look-Ahead method (H12). Similarly, the plots in the middle and last columns show % rel. cost diff. to Sarsa (S) and the Residual Gradient method (RG), respectively.

[4] Results of each method are presented with the best performing set of parameter values.

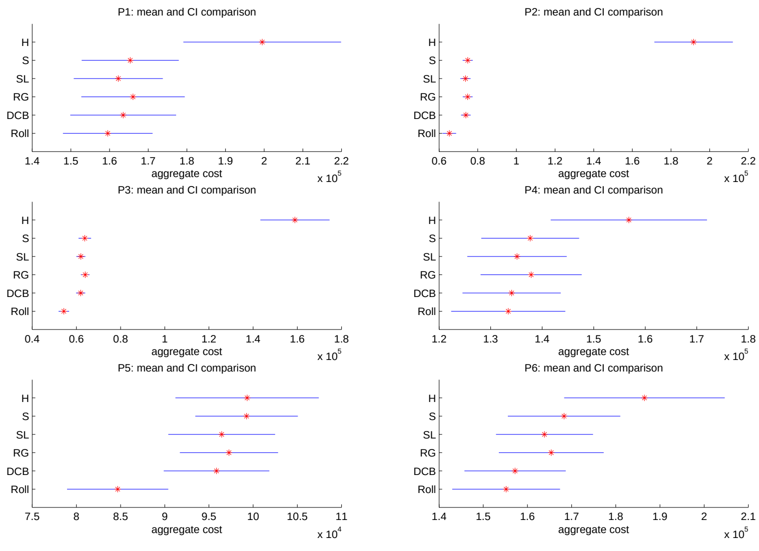

Fig. 1. Means and Confidence Intervals of aggregate costs

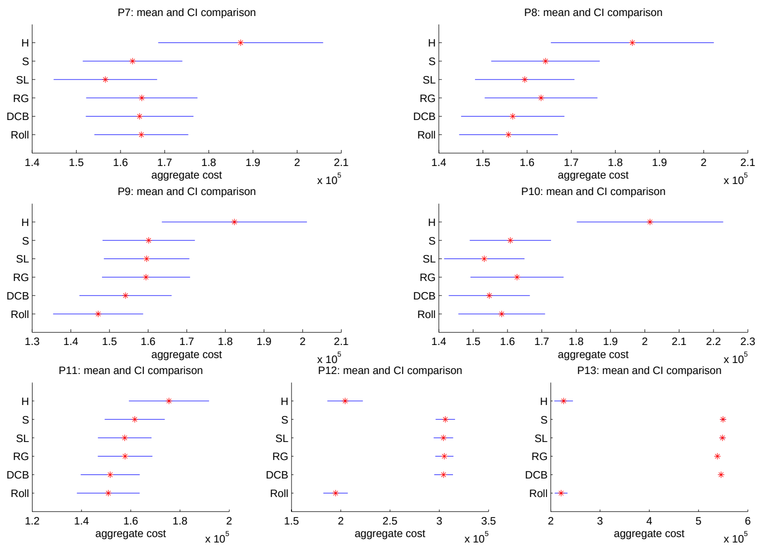

Fig. 2. Means and Confidence Intervals of aggregate costs

Based on percentages of relative cost difference, bootstrapping reduces average cost up to about 5%, regardless of the RL method. It should be noted that scenario P13 that

DCB has higher average cost than RG is when the holding cost ratio (*h/b*) is relatively high. When the holding cost is relatively high, not only that DCB underperforms RG method, all methods—Sarsa, Sarsa(λ), RG, and DCB methods—underperform the 12-period Look-Ahead method (bottom left plot of Figure 4). This implies that RL method does not work well under high *h/b* ratio. High *h/b* ratios make inventory decisions more critical—stocking more inventory has more negative effect. This may cause slower convergence of all RL methods. The study of suitable ADP methods for critical decision problems deserves further investigation.

Although the reduction of average aggregate costs by using bootstrapping seems apparent in most cases, the reduction is still within ranges of variation (at 95% confidence level). Therefore, statistically we cannot rule out the possibilities of the variation of results due to stochastic variations in the problems. We have run significance tests to double-check, but the test results could not confirm the cost reduction of bootstrapping at 95% confidence level. Table 2 shows p-values of the significance tests. The higher p-value, i.e., closer to 1, implies that evidence against the null hypothesis is insufficient.

To conclude the point, bootstrapping has shown to be able to reduce average cost up to 5% with possibility that this reduction may be due to variation of the underlying process. Therefore, taking into consideration the additional implementation effort and computation cost required by bootstrapping, bootstrapping in the forms investigated in this study are not recommended for inventory control applications.

It should be noted that our conclusion is drawn based on our experimental results. Investigation of different form of bootstrapping or different problem structure, such as a problem with long leadtime, may reveal a situation where bootstrapping can work more effectively.

Table 2. Significance test results

p-values obtained from t-tests							
Scenario	P1	P2	P3	P4	P5	P6	P7
H_a: $cost_S \neq cost_{S(\lambda)}$	0.76	0.60	0.39	0.75	0.57	0.66	(0.45)
H_a: $cost_{RG} \neq cost_{DCB}$	(0.86)	0.65	0.23	0.64	0.77	(0.39)	(0.97)
Scenario	P8	P9	P10	P11	P12	P13	
H_a: $cost_S \neq cost_{S(\lambda)}$	0.63	0.96	0.44	(0.66)	0.82	(0.69)	
H_a: $cost_{RG} \neq cost_{DCB}$	(0.71)	(0.52)	(0.58)	(0.43)	0.92	0.12	

Remark: p-values in parentheses represent p-values obtained from Wilcoxon ranksum tests. Wilcoxon ranksum test is used when assumption of normality is doubted. The normal assumption is tested by Lilliefor test at 5% significance level.

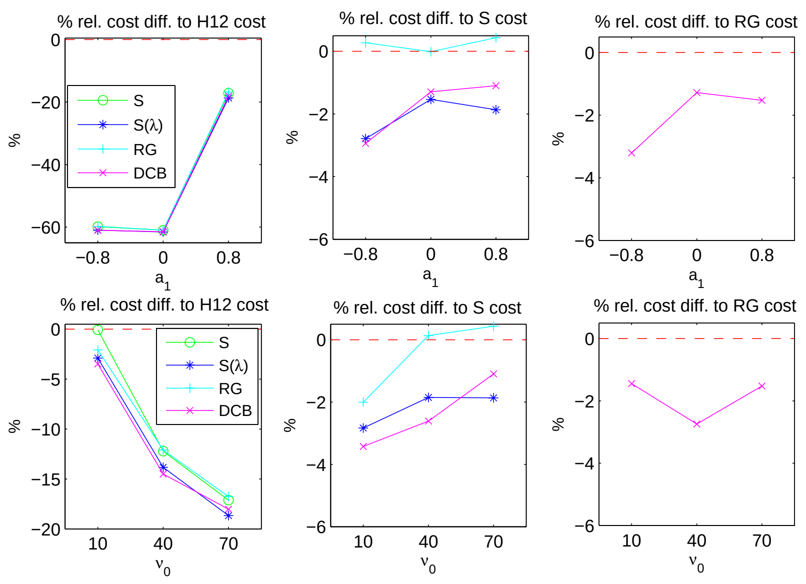

Fig. 3. Relative cost differences of ADP methods on different demand correlations (upper row) and variances (lower row)

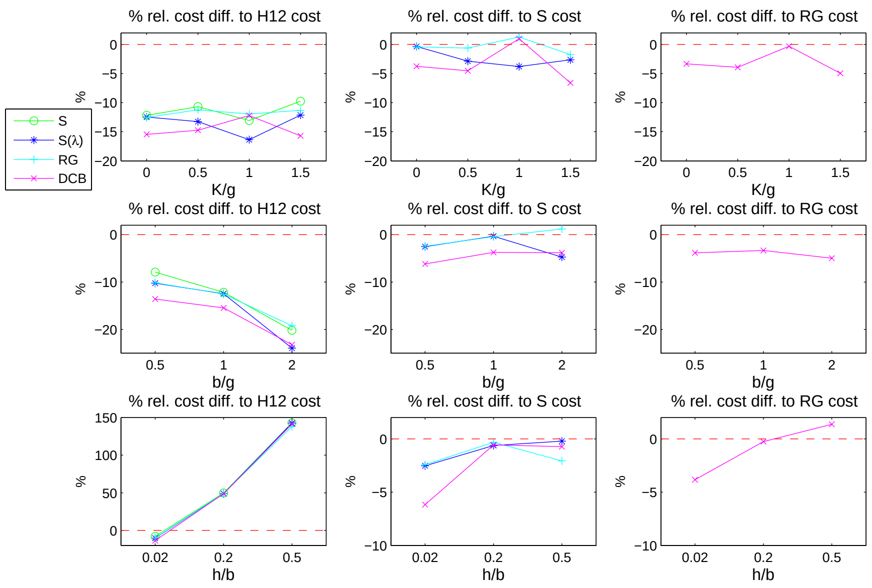

Fig. 4. Relative cost differences of ADP methods on different set up costs (top row), penalty costs (middle row) and holding costs (bottom row)

References

1. L. Baird. Residual Algorithms: Reinforcement Learning with Function Approximation, in Proceedings of the 12th International Conference on Machine Learning, Morgan Kaufmann, 1995, 30–37.
2. A. M. S. Barreto and C. W. Anderson. Restricted gradient-descent algorithm for value-function approximation in reinforcement learning. Artificial Intelligence, 172 (4–5) 2008, 454–482.
3. C. Jiang and Z. Sheng. Case-based reinforcement learning for dynamic inventory control in a multi-agent supply chain system. Expert Systems with Applications, 36 (3) 2009, 6520–6526.
4. T. Katanyukul, W. S. Duff, and E. K. P. Chong. Approximate dynamic programming for an inventory problem: Empirical comparison, Computers & Industrial Engineering, 60 (4) 2011, 719–743.
5. C. O. Kim, J. Jun, J. K. Baek, R. L. Smith, and Y. D. Kim. Adaptive inventory control models for supply chain management, International Journal of Advanced Manufacturing Technology, 26 (9–10) 2005, 1184–1192.
6. C. O. Kim, I. H. Kwon, and J. G. Baek. Asynchronous action-reward learning for nonstationary serial supply chain inventory control, Applied Intelligence 28 (1) (2008), 1–16.
7. I. H. Kwon, C. O. Kim, J. Jun, and J. H. Lee. Case-based myopic reinforcement learning for satisfying target service level in supply chain, Expert Systems with Applications 35 (1–2) 2008, 389–397.
8. J. Leng, L. Jain, and C. Fyfe. Experimental analysis of eligibility traces strategies in temporal difference learning, International Journal of Knowledge Engineering and Soft Data Paradigms, 1 (1) 2009, 26–39.
9. H. R. Maei, Cs. Szepesvari, S. Bhatnagar, D. Precup, D. Silver, and R. S. Sutton. Convergent Temporal-Difference Learning with Arbitrary Smooth Function Approximation, in Advances in Neural Information Processing Systems, MIT Press, Vancouver, BC., 2009.
10. S. D. Prestwich, S. A. Tarim, R. Rossi, and B. Hnich. A Cultural Algorithm for POMDPs from Stochastic Inventory Control, in HM '08: Proceedings of the 5th International Workshop on Hybrid Metaheuristics, Springer-Verlag, Berlin, Heidelberg, 2008, 16–28.
11. R. G. Reynolds. An Introduction to Cultural Algorithms, in Proceedings of the 3rd Annual Conference on Evolutionary Programming, World Scientific Publishing, 1994, 131–139.
12. S. Shervais, T. T. Shannon, and G. G. Lendaris. Intelligent Supply Chain Management Using Adaptive Critic Learning, IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 33 (2) 2003, 235–244.
13. S. P. Singh and R. S. Sutton. Reinforcement Learning with Replacing Eligibility Traces, Machine Learning 22 (1–3) 1996, 123–158.
14. R. S. Sutton and A. G. Barto. Reinforcement Learning, MIT Press, 1998.
15. G. J. Tesauro. TD-Gammon, a self-teaching backgammon program, achieves master level play. Neural Computation, 6 (2) 1994, 215–219.
16. B. Van Roy, D. P. Bertsekas, Y. Lee, and J. N. Tsitsiklis. A Neuro-Dynamic Programming Approach to Retailer Inventory Management, in Proceedings of the IEEE Conference on Decision and Control, 1997.