Computers & Industrial Engineering 60 (2011) 719–743

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Approximate dynamic programming for an inventory problem: Empirical comparison ☆

Tatpong Katanyukul [a,*], William S. Duff [a], Edwin K.P. Chong [b]

[a] Mechanical Engineering Department, College of Engineering, Colorado State University, United States
[b] Electrical and Computer Engineering Department, College of Engineering, Colorado State University, United States

ARTICLE INFO

Article history:
Received 4 December 2009
Received in revised form 22 November 2010
Accepted 16 January 2011
Available online 22 January 2011

Keywords:
Approximate dynamic programming
Inventory control
Reinforcement learning
Simulation
Heterogeneity
AR(1)/GARCH(1,1)

ABSTRACT

This study investigates the application of learning-based and simulation-based Approximate Dynamic Programming (ADP) approaches to an inventory problem under the Generalized Autoregressive Conditional Heteroscedasticity (GARCH) model. Specifically, we explore the robustness of a learning-based ADP method, Sarsa, with a GARCH(1,1) demand model, and provide empirical comparison between Sarsa and two simulation-based ADP methods: Rollout and Hindsight Optimization (HO). Our findings assuage a concern regarding the effect of GARCH(1,1) latent state variables on learning-based ADP and provide practical strategies to design an appropriate ADP method for inventory problems. In addition, we expose a relationship between ADP parameters and conservative behavior. Our empirical results are based on a variety of problem settings, including demand correlations, demand variances, and cost structures.

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.

Recently, Zhang (2007) analyzed data obtained from the M3 Census Bureau and established the appropriateness of the GARCH(1,1) model in inventory demand. He also showed that the inventory costs may increase significantly when the GARCH(1,1) model is not accounted for and analytically developed an order-upto-level policy for a problem without a setup cost. However, his solution is highly problem specific—a change in the structure of the problem requires reanalysis of the problem. For example, a setup cost is common in practice, but it is not included in Zhang (2007). Inclusion of a setup cost into a problem makes a cost function highly nonlinear and renders an analytical approach very difficult. Furthermore, an analytical approach is time consuming and requires highly specialized skill and effort, as discussed in Kleinau and Thonemann (2004) and Jiang and Sheng (2009). Inventory problems appear in various forms and their forms often change over time. Therefore an analytical approach is not suitable for practical inventory problems. Many articles, e.g., Silver (1981), Lee and Billington (1992) and Bertsimas and Thiele (2006) to name a few, addressed the need for an efficient flexible inventory solution simple to implement in practice.

Approximate Dynamic Programming (ADP) is a method to solve a practical Markov decision problems. It does not rely on an analytical treatment of the problem or hard-to-obtain information, e.g., transition probabilities. Therefore ADP has received extensive attention in many control applications, as discussed by Werbos (2004). ADP uses an approximation technique, which mostly is attained by either a learning-based scheme or a simulation-based scheme. A learning-based ADP method uses a learning technique, e.g., temporal difference learning, to approximate state-action costs. A simulation-based ADP method uses simulation, e.g., Monte Carlo simulation, to approximate state-action costs. Most previous studies of an ADP application to inventory management focused on learning-based ADP; see, e.g., Van Roy, Bertsekas, Lee, and Tsitsiklis (1997), Giannoccaro and Pontrandolfo (2002), Shervais, Shannon, and Lendaris (2003), Kim, Jun, Baek, Smith, and Kim (2005), Kwon, Kim, Jun, and Lee (2008), Kim, Kwon, and Baek (2008), Chaharsooghi, Heydari, and Zegordi (2008) and Jiang and Sheng (2009). Only Choi, Realff, and Lee (2006) investigated simulation-based ADP. Nonetheless, these two schemes are not mutually exclusive. For example, Kim et al. (2008) used simulation to evaluate consequences of actions not taken for accelerating a learning process in their learning-based ADP.

Although all these studies showed satisfying results in their domain problems, to the best of our knowledge, there is no previous

☆ This manuscript was processed by area editor Mohamad Y. Jaber.

* Corresponding author.
E-mail addresses: tatpong@gmail.com (T. Katanyukul), bill@engr.colostate.edu (W.S. Duff), Edwin.Chong@colostate.edu (E.K.P. Chong).

0360-8352/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2011.01.007

work investigating an ADP application to a problem with the GARCH(1,1) model. GARCH(1,1) introduces two latent state variables. The latent state variables will be inadvertently left out if the GARCH(1,1) model is not accounted for. This has posed a challenge to the model-free property of a learning-based ADP method.

When a model of the problem is available, a simulation-based ADP method is another alternative. A simulation-based ADP method is an approach intermediate between an analytical approach and a learning-based ADP method. An analytical approach requires a model of the problem and a development of an analytical solution. A learning-based ADP method requires minimum knowledge of the problem. A simulation-based ADP method requires a model of the problem and, rather than relying purely on analysis, it uses simulation to provide information for assisting action selection. Two simulation-based ADP methods are investigated here: Rollout and Hindsight Optimization (HO). Though it has been used in other applications, Rollout had been investigated for inventory problems in only one study. HO, as discussed by Chong, Givan, and Chang (2000), had not been studied for an inventory problem previously. Also, how these simulation-based ADP methods perform compared to a learning-based ADP method had not been investigated previously. Our study is intended to fill in this space. It investigates the application of ADP to an inventory problem with demand of AR1/GARCH(1,1) and provides empirical comparison among Sarsa, Rollout, and HO.

The findings here provide a practical approach to design an ADP method for an inventory problem and an insight into relations of ADP components, performance, and control behavior. Furthermore, the results reaffirm the model-free property of a learning-based ADP method even in the presence of latent state variables introduced by GARCH(1,1). The understanding exposed in this paper will help promote efficient inventory management and aid in the transfer of inventory research into practice.

This section has provided an overview of our study. Section 2 provides a review of the related literature. Section 3 establishes the general background of the three methods. Section 4 explains the problem under investigation, how our empirical study is conducted, how each controller is set up, how experiments are performed, and how the experimental results are evaluated. Section 5 explains what the experimental results indicate. The last section presents conclusions and further discussions.

2. Literature review

ADP has been recently introduced into inventory management research by Van Roy et al. (1997), Godfrey et al. (2001), Pontrandolfo, Gosavi, and Okobaa (2002), Giannoccaro and Pontrandolfo (2002), Shervais et al. (2003), Kim et al. (2005), Choi et al. (2006), Topaloglu and Kunnumkal (2006), Iida and Zipkin (2006), Chaharsooghi et al. (2008), Kim et al. (2008), Kwon et al. (2008) and Jiang and Sheng (2009). Fig. 1 shows a block diagram illustrating ADP methods used in previous studies.

2.1. Simulation-based ADP

Of all these authors, only Choi et al. (2006) investigates the application of simulation-based ADP. Simulation was used to provide reduced state space, reduced action space, and approximate transition probabilities for a dynamic program, which in turn was solved with either value iteration or Rollout. Rollout uses simulation to provide approximate state-action costs. The simulation requires a control method to provide decisions in simulation. Such a control method is called a base policy. For their base policy, Choi et al. used an (s,S) policy whose parameters were obtained from a heuristic search over pre-defined sets. The pre-defined sets of parameters used in Choi et al. (2006) are problem specific and it is unclear how they obtained them. Rollout is also investigated in our study. We use a simple formula based on the well-known Economic Order Quantity (EOQ) equation to determine parameters for the base policy. In addition to Rollout, HO—another simulation-based ADP method—has never been investigated for inventory problems. HO does not require a base policy. Therefore we investigate HO for its own virtues as well as to provide a good measure of how simulation-based ADP performs without the choice of a base policy.

2.2. Learning-based ADP

Authors studying learning-based ADP methods investigated several learning schemes.

- Van Roy et al. (1997) used one-step temporal difference learning (TD0).
- Chaharsooghi et al. (2008) used Q-learning. Q-learning, introduced by Watkins (1989), is an algorithm based on TD0.
- Kim et al. (2005) used an action-value method whose learning scheme was based on a weighted average value of a current approximation and a new observation. Their approach is similar to TD0, but it only approximates a current state-action value without a cost-to-go.
- Kwon et al. (2008) and Jiang and Sheng (2009) used the case-based myopic reinforcement learning (CMRL) method developed by Kwon et al. (2008) CMRL is based on a combination of an action-value method and a case-based reasoning technique. Case-based reasoning is state aggregation with an ability to create a new aggregation when an observed state is over a preset range of any existing aggregation group.
- Kim et al. (2008) proposed and used an asynchronous action-reward learning method. For a fast changing inventory system they assumed that information of action-consequence relations, regardless of state, was sufficient for decision making. Their asynchronous action-reward learning scheme is developed based on characteristics of inventory problems that allows simultaneously multiple action updates. Multiple action updates help accelerate the learning process to enable it to catch up with changes in the system. Instead of only updating an action-reward value for an action taken, approximate values of actions not taken were updated as well. Given an observation of an exogenous variable (e.g., demand), consequences of actions not taken can be calculated and the multiple updates achieved with these computed consequences.
- Shervais et al. (2003) used the dual heuristic programming (DHP) method. DHP, discussed by Werbos (1992), is a learning ADP method that updates a control policy directly using derivatives of the cost function. It should be noted that inclusion of a setup cost, formulated as a mathematical step function, renders this method inapplicable to the problems addressed in our study, because a step function is not differentiable. However, there is a technique to approximate a step function with a sigmoid function. A sigmoid function is differentiable. The application of DHP with an approximate step function has not been investigated and it may be worth further study.
- Giannoccaro and Pontrandolfo (2002) used the SMART algorithm, developed by Das, Gosavi, Mahadevan, and Marchalleck (1999). The SMART algorithm is similar to Q-learning. In Q-learning, every time step is assumed to be equal. Giannoccaro and Pontrandolfo studied an inventory problem whose time response is a function of a current state, a next state and a current action. To handle varied time response, SMART uses a time correction term and its associate procedures to approximate an average state-action value.