Approximate Dynamic Programming Using Design and Analysis of Computer Experiments

Ying Chen

Department of Management Science and Engineering, School of Management, Harbin Institute of Technology, Harbin, China, yingchen@hit.edu.cn

Feng Liu

Department of Anesthesia, Massachusetts General Hospital, Harvard Medical School, Boston, MA, 02114, U.S.A., fliu0@mgh.harvard.edu

Jay Rosenberger

Department of Industrial, Manufacturing, and Systems Engineering, The University of Texas at Arlington, U.S.A., jrosenbe@uta.edu

Victoria C. P. Chen

Department of Industrial, Manufacturing, and Systems Engineering, The University of Texas at Arlington, U.S.A., vchen@uta.edu

Asama Kulvanitchaiyanunt

Coraline Co., Ltd, Bangkok, Thailand, asama.kulvanitchaiyanunt@mavs.uta.edu

Yuan Zhou

Department of Industrial, Manufacturing, and Systems Engineering, The University of Texas at Arlington, U.S.A., yuan.zhou@uta.edu

*Corresponding author. Phone number: (+86)-451-86414009, Email: yingchen@hit.edu.cn

Department of Management Science and Engineering, School of Management, Harbin Institute of Technology, Harbin, China, 150000.

Abstract

In this paper, an approximate dynamic programming (ADP) method using design and analysis of computer experiments for infinite-horizon Markov decision problems is presented, aiming to mitigate the "curse of dimensionality" in continuous state and decision spaces. In this algorithm, quasi Monte Carlo sampling methods and adaptive value function approximation are incorporated within a sampling-based fitted value iteration algorithm. In addition, we use "translation" of the value function as a basis to develop a new stopping rule, which may stop the ADP algorithm earlier than the traditional L_∞-norm stopping rule and reduce computational effort. Wait-and-see, mean-value and greedy policies, are used as benchmarks to demonstrate the performance of selected ADP policies. Numerical experiments are conducted on three infinite-horizon inventory problems with different dimensions. The results demonstrate that the proposed ADP algorithm is well-suited to solve high-dimensional, infinite-horizon stochastic dynamic programs over continuous state and decision spaces, and the proposed stopping rule is able to select a high-quality ADP policy with fewer value iterations.

Key words: infinite-horizon Markov decision problems, approximate dynamic programming, design and analysis of computer experiments, stopping rule

1. Introduction

Dynamic programming (DP) was first introduced by Bellman (1957) as an optimization approach to solve a system that evolves through a series of consecutive stages or time periods. Value iteration as a popular method, offers a simple approach to obtaining optimal value functions and policies for finite-state discounted dynamic programs. Nevertheless, due to the curse of dimensionality, the traditional approach still has limitations in solving high-dimensional problems, especially over continuous state spaces. Dating back to Bellman and Dreyfus (1959), approximate value iteration (AVI) was proposed by approximating each value function using a linear combination of predefined basis functions, and in many cases, an approximation to the optimal value function is sufficient to achieve a good control performance such as with a 3-dimensonal elevator control problem (Crites and Barto 1996). As a special case of AVI, sampling-based fitted value iteration (FVI) algorithms (Chen et al. 1999; Jung and Uthmann 2004; Ernst et al. 2005; Riedmiller 2005) have been developed and used in approximate dynamic programming (ADP) and reinforcement learning (RL) communities. However, none of this research explains why sampling-based ADP/RL algorithms can be expected to work well due to approximation errors. Specifically, Tsitsiklis and Van Roy (1996) gave several simple counterexamples to illustrate that sampling-based FVI is unstable, and the value function diverges. Recognizing this issue, Munos and Szepesvari (2008) studied the interaction of value iteration and function approximation methods theoretically for discounted, large (possibly infinite) state space, finite-action Markov decision processes (MDP) when only a generative model of the environment is available. They provided finite-time performance bounds for sampling-based FVI that hold with high probability, theoretically. After that, Antos et al. (2008) studied a variant of fitted Q-iteration (Ernst et al. 2005) and proved a first finite-time bound for value function based algorithms for continuous state and action problems. These two theoretical works provide a solid foundation for sample-based FVI algorithms. However, as Munos and Szepesvari (2008) explained, FVI may suffer from the curse of dimensionality except in some specific conditions (sparsity, extreme smoothness, etc). Hence, adaptive methods are needed for more general problems. It is noted that not only FVI, but other ADP/RL algorithms also struggle to overcome the curse of dimensionality, especially with problems over continuous state and decision spaces. For example, Ernst et al. (2009) considered only two state variables and one decision variable in a continuous space, and Wei et al. (2015) only used two state variables and two decision variables in a continuous space. Some exceptions include the work of Peng et al. (2016), which aimed to solve high-dimensional, infinite-horizon MDPs over continuous state spaces. However, their algorithm requires substantial computation, since it needs millions of tuples for value function training. In addition, optimization was conducted in a discrete environment.

Given the limitations of the existing ADP/RL algorithms, an infinite-horizon approach is presented in this study based on the finite-horizon work of Chen et al. (1999), which was the first ADP approach to use statistical methods in order to mitigate the curse of dimensionality in a continuous state and decision space. In this work, we refer to this approach as design and analysis of computer experiments (DACE, Sacks et al. 1989, Chen et al. 2006).

The DACE ADP algorithm is appropriate for studying complex computer models, such as vehicle safety simulations or air quality models (Chen et al. 2006). Specifically, computer experiments involve controlled runs of a computer program from which the experimenter gains information about a complex system (the control of runs comes from an experimental design), therefore, design of experiments is used to sample the input space of the computer model, optimization is conducted according to constraints for each sampled state to achieve the output and corresponding decisions. Then, statistical modeling techniques are used to estimate the relationship between the input and output for function approximation. For ADP, the relevant computer model is the stagewise value function optimization problem, where the input space corresponds to the state space, and the output corresponds to the value of the value function. Hence, the relationship modeled is the value function. As noted, this DACE ADP algorithm is also a variant of sample-based ADP approaches. So far, the DACE ADP algorithm has been employed to solve a 9-dimensional inventory problem (Chen et al. 1999), a 20-dimensional wastewater treatment system problem (Tsai et al. 2004), a 30-dimensional water reservoir problem (Cervellera et al. 2006), and a 524-dimensional nonstationary ground-level ozone pollution problem (Yang et al. 2009). However, all of these problems are over a finite horizon. Therefore, with the help of solid theoretical performance bounds developed in Munos and Szepesvari (2008) and Antos et al. (2008) and the achieved success of the DACE based finite-horizon ADP algorithm in high-dimensional problems mentioned above, in this study, we extend the DACE algorithm to the infinite-horizon case under stationary conditions, and focus on solving high-dimensional, infinite-horizon problems over continuous state and decision spaces.

1.1. Contribution

When the transition probability, cost function and the process governing the exogenous information are stationary, the convergence of the value function is the essence of the infinite horizon problems. However, for a problem over a continuous state and decision space, sampling all the states is impossible, but by increasing the number of samples and the richness of the function space simultaneously, the resulting algorithm can solve a wide class of MDPs (Munos and Szepesvri 2008). From the statistical perspective, DACE attempts to overcome the curse of dimensionality in the state space by using a space-filling method to sample the state space with a limited set of representative points and through a statistical modeling method to develop a model that cover the state space. In this process, the decision variables are calculated through computer experiments as mentioned above, so, we do not need to enumerate the decision space as the other ADP/RL algorithms such as in a Q-iterated algorithm (Ernst et al. 2005), and the curse of dimensionality over the decision space can be avoided.

As suggested by Munos and Szepesvri (2008), an adaptive version of sampling-based FVI is needed to solve high-dimensional, infinite-horizon, problems over continuous state spaces since there is no prior knowledge how many states are needed to represent a state space. Considering this issue, in this study, we aim to develop an adaptive infinite-horizon, ADP algorithm on the basis of the DACE algorithm proposed in Chen et al. (1999) and an adaptive value function approximation (AVFA) method (Fan et al. 2013). Additionally, as stated in Ernst et al. (2005), for some supervised learning algorithms that employ statistical modeling tools for the value function approximation, there is no guarantee that the sequence of approximate value functions actually converges using the Bellman error bound (Bellman 1957), which is referred to as the L_∞-norm stopping rule in this study. Multivariate adaptive regression splines (MARS), as a supervised learning algorithm, have been employed in finite-horizon cases (e.g. Chen et al. 1999 and Yang et al. 2009). However, whether the L_∞-norm stopping rule is able to identify a good solution is investigated when using MARS. Furthermore, we propose a new stopping rule, referred to as the 45-degree line stopping rule that is less restrictive than the traditional L_∞-norm stopping rule but nonetheless identifies an optimally equivalent value function (OEVF), which can be used in the MDP to make the same decisions as an optimal value function. Consequently, the 45-degree line stopping rule may terminate the ADP algorithm earlier than the L_∞-norm stopping rule and yield an optimal policy to the MDP. Although this stopping rule is applied to the DACE ADP algorithm in this paper, it could also be used in many other ADP/RL algorithms for infinite-horizon problems to save the computational effort.

To demonstrate the performance of the proposed DACE ADP algorithm and stopping rule on the infinite-horizon cases, numerical experiments are conducted on an inventory problem with three different scales. As stated in Salas and Powell (2018), providing optimal policies as benchmarks are surprisingly rare in the ADP and RL communities. In the literature, a proposed ADP algorithm is usually compared to other existing ADP/RL algorithms, such as in Jiang and Powell (2015) and in Wei et al. (2015). However, comparing each ADP/RL algorithm requires adapting it to the continuous state and decision spaces, which is not straightforward. Instead, from a stochastic programming perspective, we use the mean-value (MV) policy to provide a good solution (Birge and Louveaux 1997) as a benchmark by using the average value of the uncertainty. Similarly, we consider the wait-and-see (WS) policy that gives an optimal solution, since perfect future information is assumed to be known before making decisions in the MDP. In the literature, the MV policy is similar to the lookahead policy with deterministic rolling horizon procedure (Powell 2011), and the WS policy has been used to generate a lower bound for the cost of a multiclass queuing application (Brown and Haugh 2017). In addition, Sarikprueck et al. (2017) built optimality bounds for an electric vehicle charging station control system using WS and MV policies. Hence, in this study, for the first time, we make use of these two policies to build bounds for the selected ADP policies. Thus, the contributions of this research are:

(1). A new infinite-horizon DACE ADP algorithm to solve high-dimensional, MDPs over a continuous state and decision space is developed.

(2). A theoretical foundation for the 45-degree line stopping rule is described.

(3). Three inventory problems are simulated to demonstrate the performance of selected ADP policies versus the baseline greedy policy.

(4). Optimality bounds based on the WS and MV policies from stochastic programming are determined to evaluate the quality of selected ADP policies.

The remainder of this paper is summarized as follows. Section 2 describes background on infinite-horizon ADP models and MARS; Section 3 introduces the DACE ADP algorithm for the infinite-horizon cases; Section 4 presents the new stopping criteria; Section 5 describes the policy evaluation methods; Section 6 shows the simulation results of applying the proposed methodology to three inventory problems; Section 7 gives conclusions and future research areas.

2. Background

2.1. Infinite-horizon ADP model

DP problems can be classified as either finite-horizon or infinite-horizon. The infinite-horizon formulation is more natural, since a specific finite time horizon is not easily specified in many real-world applications, and more importantly, it offers greater simplicity because stationary problems with infinite time horizon will lead to optimal stationary strategies (Bertsekas 2017). Infinite-horizon ADP is modeled using the following recursive formulation (Bellman 1957).

$$V(x) = \min_{u \in \Gamma} E\{c(x,u,\xi) + \gamma V(f(x,u,\xi))\}, \quad \forall x \in \mathcal{X}. \tag{1}$$

In Eq. (1), x is a vector of state variables within a real domain $\mathcal{X} \subseteq \mathbb{R}^n$, u is a vector of decision variables within a set of feasible decisions $\Gamma \subseteq \mathbb{R}^m$, ξ is a vector of stochastic variables, f is a *state transition function*, $\gamma \in (0,1)$ is a *discount factor*, c is a *cost function*, and V is the *future value function* (FVF). As described in Section 1, finding an FVF exactly is intractable for medium-sized problems. Consequently, ADP attempts to find an *approximate FVF* (aFVF, $\hat{V}$) using the following formulation.

$$\hat{V}(x) \approx \tilde{V}(x) = \min_{u \in \Gamma} E\{c(x,u,\xi) + \gamma \hat{V}(f(x,u,\xi))\}, \quad \forall x \in \mathcal{X}. \tag{2}$$

Multiple methods can be used to construct an aFVF. In this research, we use a statistical modeling technique-MARS, which is described below.

2.2. Multivariate adaptive regression splines

In previous DACE based research, MARS, introduced by Friedman (1991), has been utilized as a statistical regression tool, since it yields an adaptive continuous approximation model that does not impose any structural assumptions on the input and output data in the state space. That is, the use of MARS uncovers the structure of the value function. The MARS model, which essentially is a linear statistical model, constructs the relation from a set of coefficients and basis functions that are entirely "driven" from the regression data. The model yields accurate predictions from a forward stepwise algorithm to select model terms and a backward procedure to prune the model terms. One of the objectives of the forward stepwise algorithm is to select variables and appropriate knots simultaneously, since the MARS approximation bends to model curvature at "knot" locations. After completing the selection of basis functions, smoothness is applied to obtain a certain degree of continuity. The number of basis functions determines the flexibility and computational effort of MARS. The set of eligible knots are chosen to coincide with input levels, which are represented in the data.

The MARS backward algorithm is used to eliminate over-fitting, but this algorithm can often be omitted in order to save computational time due to the extremely low error variability in most DACE applications (Martinez et al. 2015). Instead of using the backward algorithm, an automatic stopping rule proposed by Tsai and Chen (2005) is incorporated in the forward algorithm to seek more robust models with fewer high-order interaction terms. In this study, a quintic version of MARS developed in Martinez et al. (2015) is used to model aFVFs. For detailed information of this version of MARS, please refer to Martinez et al. (2015) and Martinez (2013).

3. Infinite-horizon DACE ADP algorithm

Consider the following DACE ADP algorithm to approximately solve infinite-horizon MDPs over continuous state and decision spaces. This algorithm includes an inner data loop and an outer DP loop. The *data loop* samples the state space sequentially to build an aFVF, and the *DP loop*, provides initial aFVFs for the data loop and checks for convergence. The DP loop in this study is based on the concept of the value iteration algorithm. In order to obtain an aFVF, different stopping criteria should be used for the data loop and the DP loop. A description of the algorithm is shown below:

Step 0: Initialization:

(a). Input a discount factor γ, a state transition function f, and a cost function, c.

(b). Choose the training data set $X^{\text{Train}} \subset \mathcal{X}$ and testing data set $X^{\text{Test}} \subset \mathcal{X}$ generated by low-discrepancy sequences.

(c). Set the iteration counter to $k \leftarrow 0$, set the initial aFVF $\hat{V}_0 = 0$, and the set of evaluated state variables $X \leftarrow \emptyset$.

Step 1: Iteration of DACE ADP algorithm:

(a). Set $k \leftarrow k + 1$.

(b). For each state variable $x \in X^{\text{Train}} - X$, solve $\tilde{V}_k(x) = \min_{u\in\Gamma} E\{c(x,u,\xi) + \gamma\hat{V}_{k-1}(f(x,u,\xi))\}$ and set $X \leftarrow X \cup \{x\}$;

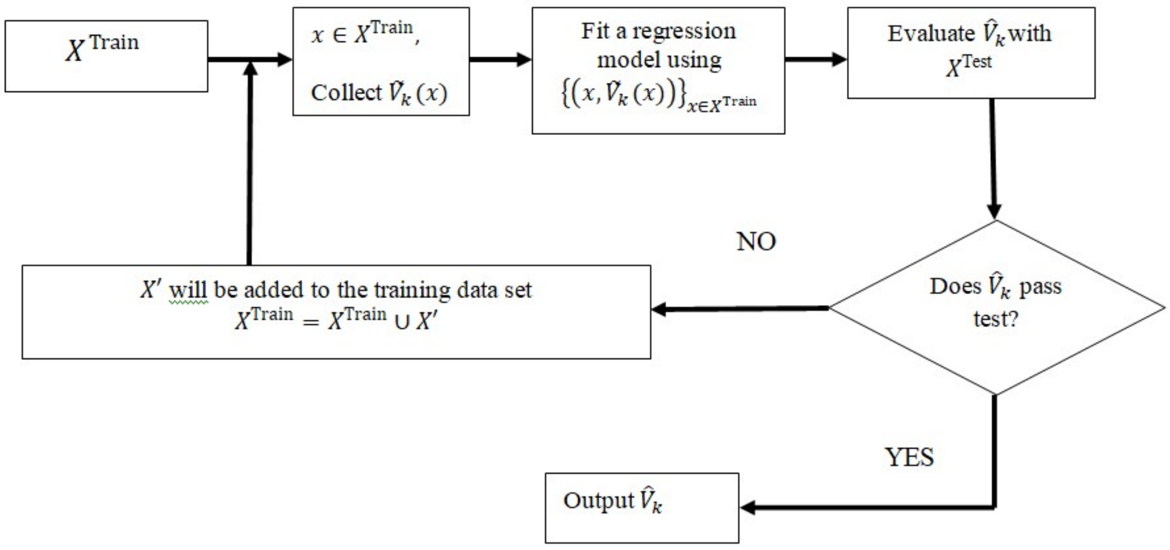
(c). Fit a regression model using the data $\{(x, \tilde{V}_k(x))\}_{x\in X^{\text{Train}}}$ to obtain $\hat{V}_k$;

(d). If $\hat{V}_k$ fails the stopping criteria for the data loop on the data $\{(x, \hat{V}_k(x))\}_{x\in X^{\text{Test}}}$, add a new set of state variables X' generated by low-discrepancy sequences to the training data set $X^{\text{Train}} \leftarrow X^{\text{Train}} \cup X'$ and go to Step 1 (b);

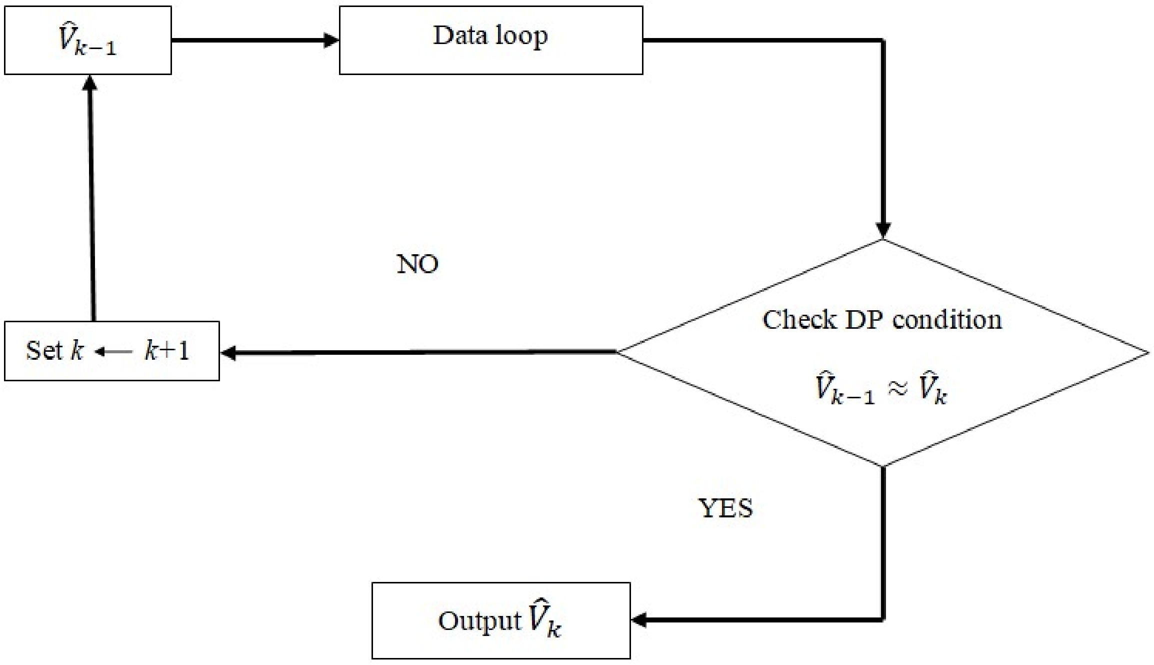
(e) If $\hat{V}_k(x) \approx \hat{V}_{k-1}(x)$, for each $x \in X^{\text{Test}}$, output $\hat{V}_k$; otherwise, $X \leftarrow \emptyset$ and go to Step 1(a).

Figure 1. DACE ADP algorithm for infinite horizon MDPs

In Fig. 1, Steps 1(a)-(e) are the DP loop, and Steps 1(b)-(d) represent the data loop. Flow charts of the data and DP loops are given in Fig. 1(a) and 1(b), respectively. It is noted that X^{Test} determines the quality of the value function since all stopping criteria are performed on X^{Test}. In this study, following Fan et al. (2013), Cervellera et al. (2013 and 2014), low-discrepancy sequences (Kuipers and Neiderreiter 2005), which are also called quasi-random sequences due to their common use as a replacement of uniformly distributed random numbers, are used to sample the training data and testing data sets. This algorithm is a general method to approximately solve infinite-horizon MDPs from the statistical perspective when the transition function, the cost function, and the process governing the exogenous information are stationary.


X^{Train}
$x \in X^{\text{Train}}$, Collect $\hat{V}_k(x)$
Fit a regression model using $\{(x, \hat{V}_k(x))\}_{x \in X^{\text{Train}}}$
Evaluate $\hat{V}_k$ with X^{Test}
Does $\hat{V}_k$ pass test?
NO
X' will be added to the training data set $X^{\text{Train}} = X^{\text{Train}} \cup X'$
YES
Output $\hat{V}_k$

(a)



(b)

Figure 2. Algorithm to approximately solve infinite horizon MDPs using the DACE ADP algorithm: (a) inner data loop, (b) outer DP loop

Remark: Bellman (1957) showed that the value iteration algorithm converges to an optimal future value function for DP problems with discrete state and decision spaces by showing that an iteration of the algorithm is a contraction mapping. By contrast, there is no similar proof of convergence to an optimal value function over a continuous state and decision space. However, Munos and Szepesvri (2008) developed finite-time bounds for sample-based FVI algorithms to solve infinite (continuous) state-space, discounted-reward MDPs theoretically. They further demonstrated that sampling-based FVI algorithms behave well by using a sufficiently large number of samples for a wide class of MDPs. Moreover, arbitrarily good performance results can be achieved with high probability. In this proposed DACE ADP algorithm, the settings of the data loop and testing data set can ensure a large training data set, which suggests that a good value function is likely generated.

4. Stopping Criteria

Under stationary conditions, there is a unique function $V(x)$ that solves the Bellman equation (1). In the literature, the L_∞-norm (Bellman 1957) is a traditional stopping rule used to identify the convergence of the value function. Specifically, the L_∞-norm stopping rule proposed by Bellman (1957) and used frequently in the literature (e.g., Wei et al. 2016 and Parr et al. 2007) terminates the value iteration algorithm when the following condition is met

$$\max_{x\in X^{\text{Test}}}\left|\hat{V}_k(x)-\hat{V}_{k-1}(x)\right| < \vartheta(1-\gamma)/2\gamma \ , \tag{3}$$

The value iteration algorithm stops when the maximum difference in the aFVF of any state is lower than the setting of the right-hand side in Eq. (3), where γ is the discount factor, and ϑ is a specified error tolerance. However, this stopping rule can be onerous as stated by Ernest et al. (2005) when using supervised learning algorithms to approximate the value function, so in the remainder of this section, a 45-degree line stopping rule is developed. In the remainder of this section, the motivation of this stopping rule is presented first, and then its usage to terminate the DP loop is described. Finally, the fact that as the L_∞-norm converges, the criteria of the 45-degree line stopping rule also converge is proven.

4.1. Motivation of 45-degree line stopping rule

In a real-time MDP, optimal decisions u^* are determined by solving the following optimization problem:

$$u^* \in \arg\min_{u\in\Gamma} E\{c(x,u,\xi)+\gamma V(f(x,u,\xi))\}.$$

However, there may be multiple methods to determine optimal decisions u^*. In this section, optimal equivalent value functions and the translation of the value function are defined. Then, the theoretical relationship between them is proven.

Definition 1 (Optimally Equivalent Value Function). A function $V'(x)$ is an *optimally equivalent value function (OEVF)* if

$$\operatorname{argmin}_{u\in\Gamma} E\{c(x,u,\xi)+\gamma V'(f(x,u,\xi))\} \subseteq \operatorname{argmin}_{u\in\Gamma} E\{c(x,u,\xi)+\gamma V(f(x,u,\xi))\}, \ \forall x \in \mathcal{X}.$$

Note that when used in a real-time MDP, using decisions based upon an OEVF will yield an optimal policy.

Definition 2 (Translation of a Function). $V'(x)$ is a *translation* of $V(x)$ if $V'(x) = V(x) + \theta$, $\forall x \in \mathcal{X}$, where θ is a constant.

Lemma 1. *If $V'(x)$ is a translation of $V(x)$, then $V'(x)$ is an OEVF.*

Proof Since $V'(x)$ is a translation of $V(x)$, *i.e.*, $V'(x) = V(x) + \theta$.

$$V'(x) = V(x) + \theta = \min_{u\in\Gamma} E\{c(x,u,\xi)+\gamma V(f(x,u,\xi))\} + \theta, \quad \forall x \in \mathcal{X}. \tag{4}$$

Since θ is a constant, a decision vector $u^* \in \Gamma$ that optimizes Eq. (4) also optimizes Eq. (1). Thus, by Definition 1, $V'(x)$ is an OEVF.

Proposition 2. *Consider a constant $\theta \in \mathbb{R}$, and a function $V'\colon \mathcal{X} \to \mathbb{R}$ such that*

$$V'(x) = \min_{u\in\Gamma} E\{c(x,u,\xi)+\gamma V'(f(x,u,\xi))\} + \theta, \quad \forall x \in \mathcal{X}. \tag{5}$$

Then, $V'(x) = V(x) + \theta/_{1-\gamma}$, $\forall x \in \mathcal{X}$, where $V(x)\colon \mathcal{X} \to \mathbb{R}$ satisfies the Eq. (1).

The proof of Proposition 2 is given in Appendix A.

The value function V is unique, but there are infinitely many translations of V. Consequently, relaxing the stopping criteria to allow for translations of V has the potential to stop the DACE ADP algorithm earlier than requiring the algorithm to continue until it finds the value function. Nonetheless, a translation of V is an OEVF, which also yields an optimal policy when used in a real-time MDP. Therefore, even if the traditional L_∞-norm stopping rule is unmet, an OEVF may be identified, so the DP loop can be terminated. Consider the following corollary:

Corollary 3: *If* $\hat{V}_k(x) = \tilde{V}_k(x) = \hat{V}_{k-1}(x) + \theta, \ \forall x \in \mathcal{X}$, *where* θ *is a constant, then* $\hat{V}_k$ *is an OEVF.*

Proof By Proposition 2, $\hat{V}_k(x) = V(x) + {}^{\theta}/_{1-\gamma}$, $\forall x \in \mathcal{X}$. By Definition 2, $\hat{V}_k$ is a translation of of V. By Lemma 1, $\hat{V}_k$ is an OEVF.

Based on Corollary 3, in the proposed algorithm shown in Fig. 1, Step 1(e) may be changed to:

If $\hat{V}_k(x) \approx \hat{V}_{k-1}(x) + \theta$, for each $x \in X^{\text{Test}}$, output $\hat{V}_k$; otherwise, $X \leftarrow \emptyset$ and go to Step 1(a).

4.2. Description of 45-degree line stopping rule

Based on Corollary 3, a new stopping rule by using a simple linear regression between two consecutive value functions is proposed. Consider the simple linear regression model shown below:

$$Y_k = \beta_0^k + \beta_1^k X_{k-1} + \epsilon_x^k, \ \forall x \in X^{\text{Test}}, \tag{6}$$

where X_{k-1} is the output of $\hat{V}$ from iteration k - 1 after inputting the X^{Test} , Y_k is the output of $\hat{V}$ from iteration *k* after inputting the X^{Test}, β_0^k is the *y* axis intercept of the regression line at iteration *k*, β_1^k is the slope of the regression line at iteration *k*, and ϵ_x^k is a random error term at iteration *k*. Eq. (6) is a simple linear regression model to generate the best line between the two variables in which the slope and intercept are estimated. As shown in Fig. 3, the best line (fitted line) is the one that minimizes the distances of the points from the line. The fitted regression line shows the relationship between predictor and response. If the slope of the line is 45 degrees, and the line passes through the origin of the coordinate system, the value of the predictor coincides with the response completely, indicating that the value function is determined.

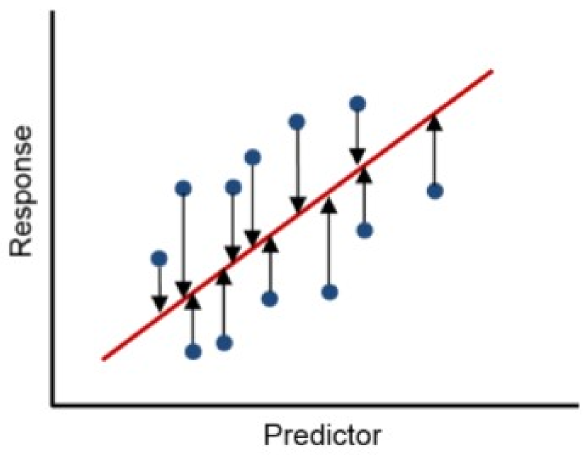


Figure 3. Example of functional relation

In the new stopping rule, the fitted regression line is represented in Eq. (7):

$$\hat{Y}_k = b_0^k + b_1^k X_{k-1}\,, \quad \forall x \in X^{\text{Test}} \tag{7}$$

where $\hat{Y}_k$ estimates Y_k, and b_0^k and b_1^kestimate the intercept and the slope at iteration *k*, respectively. According to Corollary 3, Y_k is an OEVF, if

$$Y_k = X_{k-1} + \theta^k, \qquad \forall x \in X^{\text{Test}} \tag{8}$$

In Eq. (7) and Eq. (8), b_0^k and θ^k are both constants. Therefore, if b_1^k equals 1, and $\hat{Y}_k$ is able to estimate Y_k perfectly, then Y_k is an OEVF. The coefficient of determination, R_k^2, is used to represent how well $\hat{Y}_k$ estimates Y_k. Ideally, R_k^2 equals 1, so the stopping conditions for this rule at the *k*th iteration are given by

$$b_1^k \approx 1 \quad \text{and} \quad R_k^2 \approx 1 \tag{9}$$

4.3. Convergence of 45-degree line stopping rule

In this section, we prove that if the L_∞-norm converges to zero, then under mild conditions, b_1 and R^2 converge to one, which suggests that the 45-degree line stopping rule is no more restrictive than the traditional L_∞-norm stropping rule.

Proposition 4: *Consider* $\delta > 0$ *such that* $\max_{x\in X^{\text{Test}}}\left|\hat{V}_k(x) - \hat{V}_{k-1}(x)\right| < \delta$, *then,*

$$\left|b_1^k - 1\right| < \frac{2\delta \sum_{x\in X^{\text{Test}}} |\hat{V}_{k-1}(x)|/\left|X^{\text{Test}}\right|}{\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2/\left|X^{\text{Test}}\right| - \left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)/\left|X^{\text{Test}}\right|\right)^2} \text{ , } x \in X^{\text{Test}} \tag{10}$$

$$1 - \frac{\delta^2\left(\frac{\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2}{\left|X^{\text{Test}}\right|} + 2\left|\frac{\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)}{\left|X^{\text{Test}}\right|}\right|\left(\frac{\sum_{x\in X^{\text{Test}}} |\hat{V}_{k-1}(x)|}{\left|X^{\text{Test}}\right|}\right)\right)}{\left(\frac{\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2}{\left|X^{\text{Test}}\right|} - \left(\frac{\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)}{\left|X^{\text{Test}}\right|}\right)^2\right)\left(\frac{\sum_{x\in X^{\text{Test}}} \hat{V}_{k}(x)^2}{\left|X^{\text{Test}}\right|} - \left(\frac{\sum_{x\in X^{\text{Test}}} \hat{V}_{k}(x)}{\left|X^{\text{Test}}\right|}\right)^2\right)} < R_k^2 \leq 1 \text{ , } x \in X^{\text{Test}} \tag{11}$$

The proof of **Proposition 4** is provided in Appendix B.

Corollary 5: *If for any* $\varphi' > 0$, $\exists\, K_1, K_2$ *and* K_3 and $\sigma > 0$, *and* $V^u > 0$, *such that*

Condition 1. $\max_{x\in X^{\text{Test}}} |\, \hat{V}_{k-1}(x) \,| < V^u, \forall k = K_1, K_1 + 1, \ldots$

Condition 2. $\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2/\left|X^{\text{Test}}\right| - \left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)/\left|X^{\text{Test}}\right|\right)^2 > \sigma$, $\forall k = K_2, K_2 + 1, \ldots$

Condition 3. $\max_{x\in X^{\text{Test}}}\left|\hat{V}_k(x) - \hat{V}_{k-1}(x)\right| < \varphi'$, $\forall k = K_3, K_3 + 1, \ldots$

then $\lim_{k\to\infty} b_1^k = 1$.

Proof Consider $\varphi_1 > 0$, let $\varphi' = \frac{\sigma\varphi_1}{2V^u} > 0$, by **Condition 3**, $\exists\, K_3$, such that

$$\max_{x\in X^{\text{Test}}}\left|\hat{V}_k(x) - \hat{V}_{k-1}(x)\right| < \frac{\sigma\varphi_1}{2V^u}, \ \forall k = K_3, K_3 + 1, \ldots \tag{12}$$

by **Proposition 4**,

$$\left|b_1^k - 1\right| < \frac{2\frac{\sigma\varphi}{2V^u} \sum_{x\in X^{\text{Test}}} |\hat{V}_{k-1}(x)|/\left|X^{Test}\right|}{\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2/\left|X^{\text{Test}}\right| - \left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)/\left|X^{\text{Test}}\right|\right)^2}, \ \forall k = K_3, K_3 + 1 \ldots \tag{13}$$

Let $K_4 = \max\,(K_1, K_2, K_3)$, then by **Condition 1** and **Condition 2**, $\left|b_1^k - 1\right| < \varphi_1, \forall k = K_4, K_4 + 1 \ldots$

Corollary 6: *If for any* $\varphi' > 0$, $\exists\, K_1, K_2$ *and* K_3 and $\sigma > 0$, *and* $V^u > 0$, *such that*

Condition 1. $\max_{x \in X^{\text{Test}}} | \hat{V}_{k-1}(x) | < V^u, \forall k = K_1, K_1 + 1, \ldots$

Condition 2. $\sum_{x \in X^{\text{Test}}} \hat{V}_{k-1}(x)^2 / |X^{\text{Test}}| - \left(\sum_{x \in X^{\text{Test}}} \hat{V}_{k-1}(x) / |X^{\text{Test}}|\right)^2 > \sigma,\ \forall k = K_2, K_2 + 1, \ldots$

Condition 3. $\max_{x \in X^{\text{Test}}} \left|\hat{V}_k(x) - \hat{V}_{k-1}(x)\right| < \varphi',\ \forall k = K_3, K_3 + 1, \ldots$

then $\lim_{k \to \infty} R_k^2 = 1$.

Proof Consider $\varphi_2 > 0$, let $\varphi' = \frac{\sigma\sqrt{\varphi_2}}{\sqrt{3}V^u}$, by **Condition 3**, $\exists\, K_3$, such that

$$\max_{x \in X^{\text{Test}}} \left|\hat{V}_k(x) - \hat{V}_{k-1}(x)\right| < \frac{\sigma\sqrt{\varphi_2}}{\sqrt{3}V^u},\ \forall k = K_3, K_3 + 1 \ldots, \tag{14}$$

by **Proposition 4**,

$$R_k^2 > 1 - \frac{\left(\frac{\sigma\sqrt{\varphi_2}}{\sqrt{3}V^u}\right)^2 \left(\frac{\sum_{x \in X^{\text{Test}}} \hat{V}_{k-1}(x)^2}{|X^{\text{Test}}|} + 2\left|\frac{\sum_{x \in X^{\text{Test}}} \hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right| \left(\frac{\sum_{x \in X^{\text{Test}}} |\hat{V}_{k-1}(x)|}{|X^{\text{Test}}|}\right)\right)}{\left(\frac{\sum_{x \in X^{\text{Test}}} \hat{V}_{k-1}(x)^2}{|X^{\text{Test}}|} - \left(\frac{\sum_{x \in X^{\text{Test}}} \hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right)^2\right)\left(\frac{\sum_{x \in X^{\text{Test}}} \hat{V}_k(x)^2}{|X^{\text{Test}}|} - \left(\frac{\sum_{x \in X^{\text{Test}}} \hat{V}_k(x)}{|X^{\text{Test}}|}\right)^2\right)},\ \forall k = K_3, K_3 + 1 \ldots \tag{15}$$

Let $K_4 = \max(K_1, K_2, K_3)$, then by **Condition 1** and **Condition 2,** then, $R_k^2 > 1 - \varphi_2, \forall k = K_4, K_4 + 1 \ldots$ By **Proposition 4**, $1 \geq R_k^2 > 1 - \varphi_2,\ \forall k = K_4, K_4 + 1 \ldots$

In Corollaries 5 and 6, Condition 1 indicates the value function is bounded, Condition 2 denotes the value function is not flat, and Condition 3 suggests that the value function converges. All these are the standard conditions for the contraction mapping theorem of the classical value iteration algorithm and are regarded as mild conditions in this study. Therefore, under mild conditions, according Corollaries 5 and 6, we are able to conclude that when L_∞-norm converges to 0, b_1 and R^2 converge to 1. However, by **Proposition 2**, even when R^2 and b_1 are both 1, we cannot conclude L_∞-norm converges to 0. Hence, we conclude that L_∞-norm stopping rule is more restrictive than the 45-degree line stopping rule, which denotes that 45-degree line rule may stop the DACE ADP algorithm earlier than the L_∞-norm rule. Moreover, we note that the 45-degree line stopping rule is a general stopping rule that is applicable to numerous other ADP/RL algorithms in the literature and may save computational effort with them as well.

5. Policy Evaluation

In order to evaluate the selected ADP policy, consider a simulation environment with the following process:

```
Step 0: Initialization:

    (a). Input a discount factor $\gamma$, a state transition function $f$, a cost function, $c$, an aFVF $\hat{V}$, and let $T$ be the number of stages in a finite time horizon.

    (b). Use Sobol's sequence to initialize a set of $N$ state vectors, $X^N$, and $\forall x \in X^N$, set the simulated cost $\tilde{c}(x) \leftarrow 0$.

    (c). Set the evaluated state vectors $X \leftarrow \emptyset$.

Step 1: Iteration of simulation:

    (a). Consider state variable $x \in X^N - X$, set the time period counter to $t \leftarrow 1$.

    (b). Find $u^* \in \arg\min_{u \in \Gamma} E\{c(x,u,\xi) + \gamma\hat{V}(f(x,u,\xi))\}$ .

    (c). Sample a random vector $\zeta$ and update the estimated total cost $\tilde{c}(x) \leftarrow \tilde{c}(x) + \gamma^{t-1}c(x,u^*,\zeta)$ and the state vector $x \leftarrow f(x,u^*,\zeta)$.

    (d). If $t < T$, set $t \leftarrow t + 1$ and go to Step 1(b).

    (e). Set $X \leftarrow X \cup \{x\}$. If $X \subset X^N$, go to Step 1(a); otherwise output the data $\{(x,\tilde{c}(x))\}_{x \in X^N}$.
```

One benchmark used in this and other research (e.g., Kossmann and Stocker 2000, Long et al. 2018) is a greedy policy, which determines actions based upon a myopic optimization process given by:

$$\min_{u \in \Gamma} E\{c(x,u,\xi)\}$$

Observe that the greedy policy optimizes each period without considering the impact of the decisions on future states. Therefore, to evaluate the greedy policy, Step 1(b) in above simulation process is given by

$$\text{Find } u^* \in \arg\min_{u\in\Gamma} E\{c(x,u,\xi)\} \ .$$

Moreover, this research uses the MV and WS policies as optimality bounds in order to demonstrate the quality of the selected ADP policy, which has not previously been done in the ADP/RL literature. The MV policy makes decisions by solving a deterministic model in which all random variables are replaced by their means. Hence, using the MV policy, Step 1(b) in the above simulation process is given by the following two steps:

(b) (i). Determine the expected value of $\xi_\tau, \bar{\xi}_\tau, \quad \forall\tau = t \ldots T+t-1,$

(b) (ii). Find $u_t^* \in \arg\min_{u_t\in\Gamma} \sum_{\tau=t}^{T+t-1} \gamma^{\tau-1} c(x_\tau, u_\tau, \bar{\xi}_\tau)$ s.t. $x_t = x$, $x_{\tau+1} = f(x_\tau, u_\tau, \bar{\xi}_\tau)$, $u_\tau \in \Gamma, \forall\tau = t \ldots T+t-1.$

The value of the stochastic solution (VSS) describes the loss of ignoring uncertainty in a given problem. Specifically, VSS is calculated by taking the difference between $\tilde{c}(x)$ using the MV policy and $\tilde{c}(x)$ using an optimal policy for each $x \in X^N$.

As mentioned above, the assumption for the WS policy is the future states are perfectly known, therefore, Steps 1(a)-1(d) in above process are changed to:

Step 1:

(a). Consider state variable $x \in X^N - X$, set a sample counter to $m \leftarrow 1$.

(b)(i). Sample a random vector $\zeta_\tau, \quad \forall\tau = 1 \ldots T,$

(b)(ii). Find $u_1^*, u_2^*, \dots u_T^* \in \arg\min_{u_1,u_2,\dots,u_T \in \Gamma} \sum_{\tau=1}^{T} \gamma^{\tau-1} c(x_\tau, u_\tau, \zeta_\tau)$ s.t. $x_1 = x$, $x_{\tau+1} = f(x_\tau, u_\tau, \zeta_\tau), \forall \tau = 1 \dots T$.

(c). Calculate the estimated total cost $\tilde{c}(x) \leftarrow \tilde{c}(x) + \sum_{t=1}^{T} \gamma^{t-1} c(x_t, u_t^*, \zeta_t)/M$.

(d). If $m < M$, set $m \leftarrow m + 1$ and go to Step 1(b).

The expected value of perfect information (EVPI) is the loss of objective value due to the presence of uncertainty (Birge and Louveaux 1997), which is defined as the difference between the simulated results of the ADP policy and the WS policy in this study.

Therefore, in this study, after evaluating the identified ADP policy in a simulated environment, in order to demonstrate its quality, it is compared with the greedy, the MV and the WS policies.

6. Computational experiments of inventory problem

In this research, a three-stage nine-dimensional stochastic inventory problem from the literature (e.g., Chen et al. 1999 and Chen 1999) is modified to have an infinite number of time periods and different dimensions. To demonstrate the effectiveness of the proposed infinite-horizon DACE ADP algorithm and stopping rule, numerical experiments are conducted on the inventory problem with different dimensions. Compared to cases with continuous state and decision spaces in the literature (e.g., Wei et al. 2015 and Ernst et al. 2009), these problems are relatively large.

6.1. Overview of stochastic inventory problem

As described in Chen et al. (1999), the stochastic inventory problem considers order quantities for items (G) over two forecast periods given the inventory level and demand forecast for each item. The state variables include:

- The inventory level of item i at the beginning of time period t : $I_t^{(i)}$.

- The forecast of the demand of item *i* in the current time period *t* determined at the beginning of time period *t* : $D_{(t,t)}^{(i)}$.
- The forecast of the demand of item *i* in next time period *t* + 1 determined at the beginning of time period *t* : $D_{(t,t+1)}^{(i)}$.

The state vector at the beginning of a stage is represented by

$$x_t = \left(I_t^{(1)}, D_{\mathrm{t,t}}^{(1)}, D_{\mathrm{t,t+1}}^{(1)}, I_t^{(2)}, D_{\mathrm{t,t}}^{(2)}, D_{\mathrm{t,t+1}}^{(2)}, \dots, I_t^{(G)}, D_{\mathrm{t,t}}^{(G)}, D_{t,t+1}^{(G)}\right)^T. \tag{16}$$

The decision variables are the amounts of item *i* ordered in period *t*, and the decision vector at the beginning of stage is represented by $u_t = \left(u_t^{(1)}, u_t^{(2)}, u_t^{(3)}\right)$. The state transition functions are modeled using a multiplicative MMFE (see Chen et al. 1999 for details). The constraints on the decision variables (amounts ordered) and the state variables (inventory levels) are placed in the form of capacity constraints. The transition functions are given by

$$I_{t+1}^{(i)} = I_t^{(i)} + u_t^i - (D_{(t,t)}^{(i)} \cdot \varepsilon_{(t,t)}^{(i)}) \quad , \quad \forall i \in G \tag{17}$$

$$D_{(t+1,t+1)}^{(i)} = \left(D_{(t,t+1)}^{(i)} \cdot \varepsilon_{(t,t+1)}^{(i)}\right) \quad , \quad \forall i \in G \tag{18}$$

$$D_{(t+1,t+2)}^{(i)} = \left(\mu_{t+2}^{(i)} \cdot \varepsilon_{(t,t+2)}^{(i)}\right) \quad , \quad \forall i \in G \tag{19}$$

where $\mu_t^{(i)}$ is the mean demand for item *i* in period *t*, and $\varepsilon_{(t,t+p)}^{(i)}$ is the change in forecast for the time *t* + *p* from the forecast made in period *t*. The objective function is a cost function involving inventory holding costs and backorder costs. The cost function is V-shaped and is represented as

$$c_t(x_t, u_t, \xi_t) = \sum_{i=1}^{G} h_i \left[I_{t+1}^{(i)}\right]_+ + \pi_i \left[-I_{t+1}^{(i)}\right]_+ \quad , \quad \forall t = 1, \dots \tag{20}$$

where h_i is a constant holding cost parameter for item i, and π_i is a constant backorder cost parameter for item i. For optimization purposes, a smoothed version of the cost function has been used (see Chen et al. 1999 for details).

6.2. Experimental settings

Experiments on the DACE ADP algorithm for the infinite-horizon MDPs are conducted in MATLAB 2016b on a Dell computer with a Xeon, 8-core, 3.6 GHz CPU. In order to compute the expected value of each sampled point in the state space, eight scenarios corresponding to eight realizations of stochastic variables are used (Chen et al. 1999). The discount factor used in this research is 0.9 since the duration between time periods is assumed to be a month. In the experiments, MARS approximates the value function, and the stopping rules described in Section 4 select the ADP policies. After determining the approximate value function, it is simulated as described in Section 5, where stochastic variables are generated by the MMFE.

The discount factor γ is less than 1, so as long as the cost $c(x, u, \xi)$, stabilizes, or grows at a rate less than $1/\gamma$, over time, then the simulated cost $\tilde{c}$ converges. In this research, numerical experiments are conducted to determine a value of T for which the simulated cost appears to stabilize. Specifically, 10 initial states are randomly selected and run 40, 50, 60, 70, 80 and 120 time periods using the greedy policy and then their simulated costs $\tilde{c}$, which are shown in Table 2, are compared. The $\tilde{c}$ of these 10 states appears to stabilize at 70 periods, so for the remainder of the experiments, 70 time periods are simulated.

In the experiments, the inventory problems with three different scales are conducted, respectively, with the following settings. The initial training data set, X^{Train}, includes 500, 1000 and 1500 state vectors generated by a Sobol sequence (Sobol, 1967) for 6-dim, 9-dim and 12-dim cases, separately. For these three cases, each iteration of the data loop adds 50 more state vectors, X', which are also generated by a

Sobol sequence. To avoid over fitting, in practice, we sample the testing data set, X^{Test}, including 250 state vectors created by a Halton sequence (Halton 1960). The stopping rule for data loop in Step 1(d) in Fig. 1 is whether the testing R'^2 from testing data set is greater than 0.8, and the difference of R'^2 between two consecutive data loop is less than 0.05. In statistical modeling, high R'^2 usually indicates a high-quality estimation, but high R^2 might imply the model over-fits the data. Hence, the value 0.8 is used in this study.

Table 2. Simulated cost used for time period determination for 10 initial states ($)

Scenario	40 periods	50 periods	60 periods	70 periods	80 periods	120 periods
1	232.5	244.7	247.6	248.4	248.7	248.8
2	225.5	228.5	229.5	230.3	230.4	230.4
3	37.8	42.9	44.0	44.3	44.3	44.4
4	1875.1	1890.1	1894.7	1896.0	1896.2	1896.3
5	180.1	184.0	184.6	184.8	184.9	184.9
6	220.9	224.7	226.2	226.5	226.5	226.7
7	154.9	156.0	156.8	157.0	157.0	157.1
8	235.9	239.3	240.5	241.1	241.4	241.5
9	128.4	132.2	133.8	134.1	134.1	134.2
10	122.9	126.6	129.8	130.6	130.8	130.9

6.3. Computational results

6.3.1. Stopping criteria and aFVF selection

At the beginning of the computational experiments, 6000 DP iterations of the DACE ADP algorithm are executed on the 9-dimensional inventory problem, spending almost 10.5 days to investigate the ADP policies. A plot of the L_∞-norm over the 6000 iterations is shown in Fig. 4, which clearly indicates that the value function is not converging using the L_∞-norm stopping rule. By contrast, a plot of b_1 from the 45-degree line for the 6000 DP iterations is given in Fig. 5. As observed, the b_1 value starts to level off at early DP loop iterations.

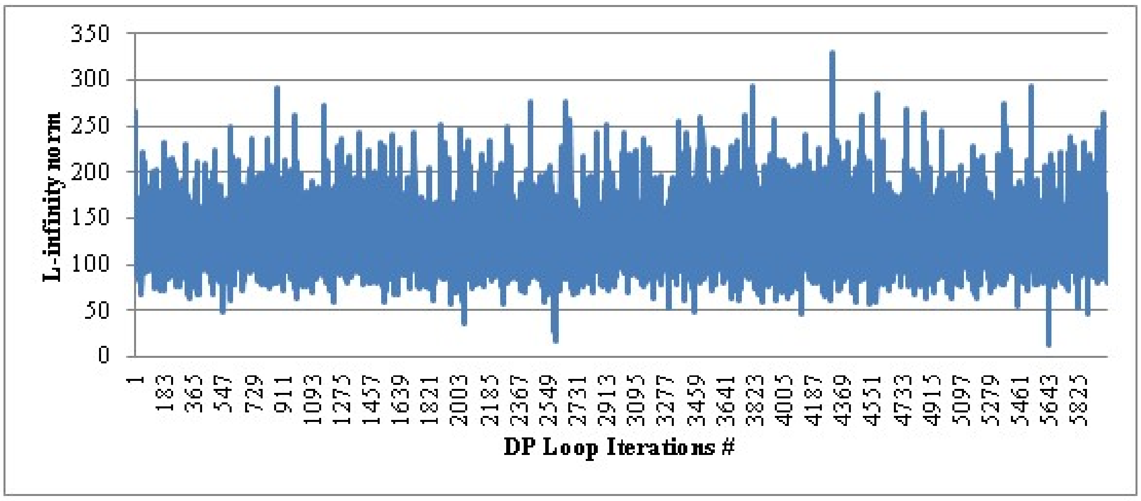


Figure 4. Plot of L_∞-norm value over 6000 DP iterations

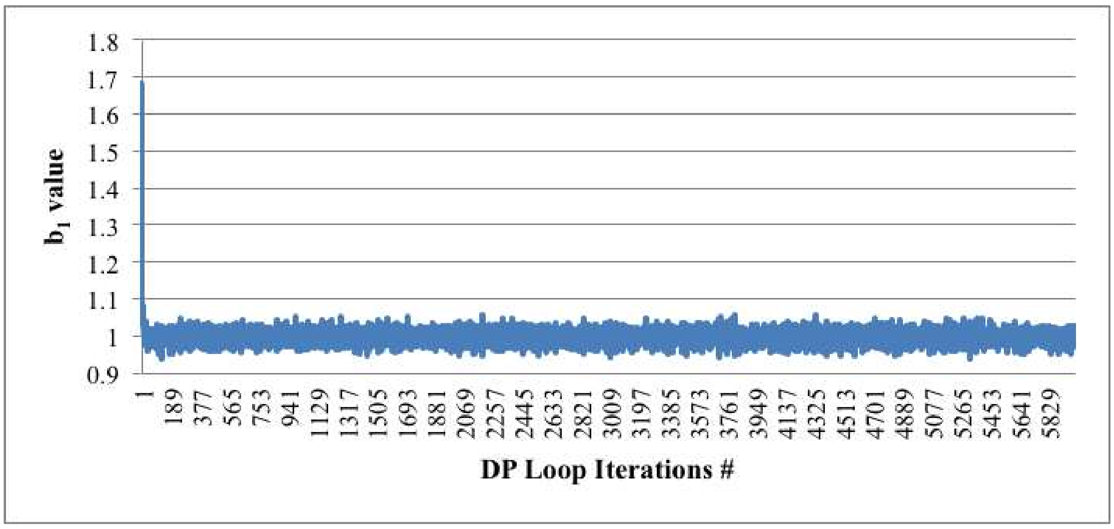


Figure 5. Plot of b_1 slope value from the 6000 DP loop iterations

Numerical experiments for 6, 9 and 12 dimensional inventory problems yield plots of the L_∞-norm and 45-degree line stopping rule shown in Fig. 6 and Fig. 7. Observe in Fig. 6 that the L_∞-norm rule exhibits no clear patterns, which indicates no aFVF can clearly be selected. However, in comparison, Fig. 7 displays very promising patterns using the 45-degree line stopping rule. In Fig. 7(a), the b_1 value of the 6-dimensional inventory problem starts to level off at the 13th DP loop iteration, and the R^2 value is 0.986 at this iteration. Therefore, the 13th aFVF is selected as an OEVF with around 17 minutes of computational time. In Fig. 7(b) on the 9-dimensional problem, the b_1 and R^2 values level off at 1 at the 15th DP loop iteration, but the b_1 value has a subsequent downward spike, so the 20th aFVF is selected as an OEVF, which requires only about 50 minutes to compute. In Fig. 7(c), the b_1 value starts to level off at the 13th iteration and tends to stabilize around 1 afterwards, and the change in the value of R^2 stabilizes close to 1 after 3 DP iterations. Thus, the 13th aFVF is selected as an OEVF with about 91 minutes computational time for the 12-dimensional problem. To determine the quality of these selected aFVFs, they are simulated in the next section.

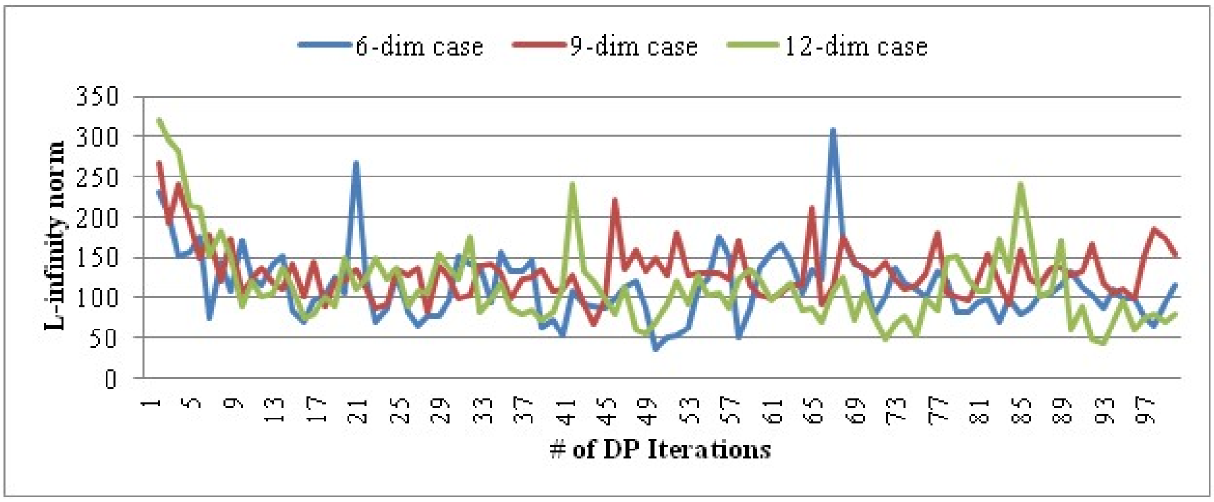
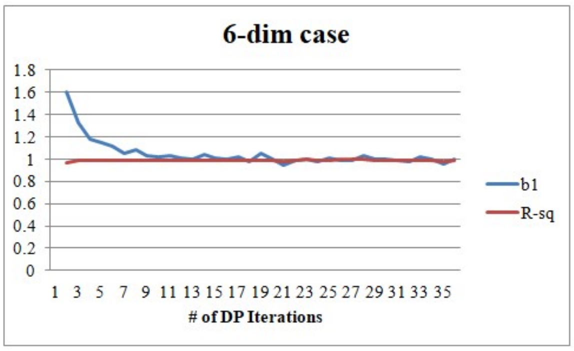
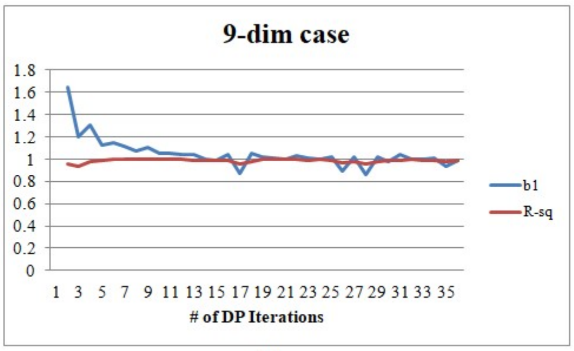


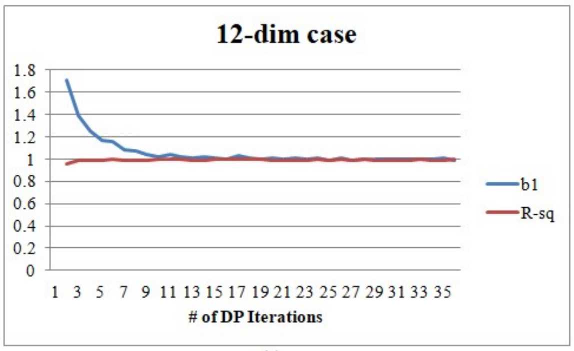
Figure 6. L_∞-norm value evolving patterns of the first 100 DP loop iterations for the 6, 9 and 12-dimensional inventory problems



(a)



(b)



(c)

Figure 7. Plot of b_1 and R^2 at early DP loop iterations of the DACE ADP algorithm for the 6, 9 and 12-dimensional inventory problems

6.3.2. Simulation results

The selected ADP policies from the previous section are simulated with 100 initial state vectors, X^{100} from a Sobol sequence. After collecting the results using the 13th, 20th, and 13th aFVFs for the 6, 9 and 12-dimensional inventory problems, respectively, optimality bounds are determined by simulating the WS and MV policies. In addition, the greedy policy is used as another benchmark. Table 3 shows the average simulated costs using the different policies. To compare the performance of the ADP policy and the other three policies, a paired t-test on the simulation results is conducted, for which each pair has the same initial state and epsilon trajectory. Hence, the simulation results for the two policies are dependent (Kutner et al. 2004). Note that the paired t-test has not previously been used for policy comparison in the literature. Table 4 shows the paired t-test results between the ADP policy and the other three policies ("*p*-value < α value" indicates strong evidence against the null hypothesis that the average costs are the same, so the null hypothesis will be rejected; "*p*-value > α value", indicates weak evidence against the null hypothesis, therefore, it fails to reject the null hypothesis).

Table 3. Average simulated costs using different policies for the 6, 9 and 12-dimesnioal inventory problems

	WS policy	MV policy	Greedy Policy	ADP policy
6-dim case	231.35	302.10	285.79	264.59
9-dim case	270.56	335.6	331.54	297.53
12-dim case	394.62	543.34	494.46	450.89

Table 4. Paired t-test results (*p*-values) between ADP policy using the selected ADP policies and the other policies for the 6, 9 and 12-dimensional inventory problems (α=0.05)

	WS policy VS ADP policy	MV policy VS ADP policy	Greedy policy VS ADP policy
6-dim case	<0.0001	<0.0001	0.0022
9-dim case	<0.0001	<0.0001	<0.0001
12-dim case	<0.0001	<0.0001	0.0171

The WS policy is a lower bound on the optimal cost since the future states are perfectly known in advance. As observed in Table 3, the WS policy has the lowest average simulated cost. On the other hand, the selected ADP policies perform much better than the MV policy and the greedy policy, which strongly demonstrates this infinite-horizon DACE ADP algorithm is able to solve the MDPs over continuous state and decision spaces, and the 45-degree line stopping rule is likely to stop the DP iterations reasonably by selecting high-quality ADP policies.

In addition, using the optimality bounds from stochastic programming, the importance of the building the ADP policy for the MDP is exhibited. Table 5 displays the WS, EVPI upper bounds and VSS lower bounds for the three case studies conducted above. The fourth and sixth columns indicate the EVPI upper bound percentages and VSS lower bound percentages of the simulation results using the ADP policy. From Table 5, observe that all of the EVPI upper bounds are smaller than the corresponding VSS lower bounds, which indicates that using the DACE ADP algorithm to build the model first decreases the effect of stochasticity in the MDP.

Table 5. An upper bound on EVPI and a lower bound on VSS of these three problem instances

Instance	WS	EVPI upper bound	%EVPI upper bound	VSS lower bound	% VSS lower bound
6-dim case	231.35	33.24	12.56	37.51	14.18
9-dim case	270.56	26.97	9.06	34.01	11.43
12-dim case	394.62	56.27	12.48	92.45	20.51

7. Conclusion

Solving high-dimensional, infinite-horizon MDPs over continuous state and decision spaces is still a challenge. In this study, a new algorithm based on DACE concepts from the statistical perspective to solve MDPs over a continuous state and decision space is proposed. Meanwhile, a new stopping rule based on the translation of the value function is developed. After theoretical exploration, the L_∞-norm stopping rule is more restrictive than the 45-degree line stopping rule, which indicates 45-degree line rule is likely to terminate DP iteration earlier than the L_∞-norm rule. Although the 45-degree line stopping rule is developed for the DACE ADP algorithm in this research, this rule may also be applied to numerous other RL/ADP algorithms for infinite-horizon MDPs. Furthermore, the DACE ADP algorithm and stopping rule are successfully applied to an inventory problem with three different scales. WS and MV policies from stochastic programming are used as optimality bounds for the selected ADP policies. The main conclusions from the results of these three inventory problems include: 1) the DACE based ADP algorithm has the capability to find high-quality approximate value functions for medium/high-dimensional MDPs over continuous state and decision spaces from the results shown in Tables 3-5 and Figs. 4-7; 2) the L_∞-norm may not be useful to derive a high-quality or near-optimal ADP policy when using MARS as a supervised learning algorithm to approximate the value function from Figs 4 and 6; 3) the proposed 45-degree line stopping rule facilitates the selection of an OEVF at early DP iterations, which significantly shortens the computational time: according to Fig. 7, the leveling-off patterns of b_1 and R^2 around 1, are observed for all three cases; 4) combining Tables 3-4, the ADP policy performs much better than the MV and greedy policies, statistically: in Table 3, the average simulated costs of ADP policies are much lower than those of the MV and greedy policies, and in Table 4, two-sided paired t-test results indicate the ADP policies do not have the same performance as the MV and greedy policies, statistically; 5) the resulting optimality bounds reveal the benefit of building the model for decision making in the MDPs to decrease the effects of uncertainty in practice since all of the EVPI upper bounds are smaller than the VSS lower bound in Table 5; 6) although the computational time to obtain an OEVF increases when the dimension of the problem increases, it does not increase exponentially (6-dim, 9-dim and 12 dim cases consume 17, 50, 91 minutes, respectively), which is also an essential benefit of this proposed DACE ADP algorithm for the infinite-horizon problems.

Future research will expand in three directions. Firstly, even though the 9- and 12-dimensional versions of the inventory problem are higher than many other problems in the literature, this algorithm should be applied to other MDPs with more dimensions and in different domains. Secondly, more efficient sampling of the state space should be explored, and data loop stopping rule needs to be investigated further. Thirdly, in this paper, aFVFs are not forced to be convex, so the optimization of problems in the proposed DACE ADP algorithm may be challenging. Consequently, methods to ensure that aFVFs are convex (e.g., Martinez et al. 2015) and methods to globally optimize nonconvex aFVFs (e.g., Martinez et al. 2017) should be explored.

Acknowledgments

This research is supported by National Science Foundation grant ECCS-1128871.

Conflict of Interests

The authors declare that they have no conflict of interest.

References

Antos, A., Munos, R., Szepesvari, C. (2008) Fitted Q-iteration in continuous action-space MDPs. In Advances in Neural Information Processing Systems. pp. 9-16.

Bellman RE (1957) Dynamic programming. *Princeton University Press*.

Bellman R, Dreyfus S. (1959) Functional approximations and dynamic programming. Mathematical Tables and Other Aids to Computation. Vol. 13, No. 68, pp. 247-251.

Bertsekas, DP (2017) Dynamic Programming and Optimal Control. Vol. I, 4th Ed. *Athena Scientific*.

Brown DB, Haugh, MB. (2017) Information relaxation bounds for infinite horizon Markov decision processes. Operations Research. Vol. 65, No.5, pp. 1355-1379.

Crites R, Barto A. (1996) Improving elevator performance using reinforcement learning. Advances in Neural Information Processing Systems. In Advances in Neural Information Processing Systems 9.

Long EF, Nohdurft E, Spinler S (2018) Spatial resource allocation for emerging epidemics: a comparison of greedy, myopic, and dynamic policies. Manufacturing & Service Operations Research. Article in Advance, pp. 1-18.

Birge JF, Louveaux F (1997) Introduction to stochastic programming. Springer, New York.

Cervellera C, Chen VCP, Wen A. (2006) Optimization of a large-scale water reservoir network by stochastic dynamic programming with efficient state space discretization. *European Journal of Operational Research*, 171(3): 1139-1151.

Cervellara, C, Gaggero, M, Maccio, D, Marcialis, R. (2013) Quasi-Random Sampling for Approximate Dynamic Programming. The International. Joint Conference on Neural Networks (IJCNN), Aug 4-9, Dallas, TX, USA.

Cervellara, C, Gaggero, M, Maccio D (2014) Low-discrepancy sampling for approximate dynamic programming with local approximators. Computers & Operations Research, 43: 108-115.

Chen VCP, Ruppert D, Shoemaker CA (1999). Applying experimental design and regression splines to high-dimensional continuous-state stochastic dynamic programming. *Operations Research*, 47(1): 38-53.

Chen VCP, Tsui KL, Barton RR, Meckesheimer M (2006) Design, modeling, and applications of computer experiments. *IIE Transactions*, 38(4): 273-291.

Ernst D, Geurts P, Wehenkel L (2005) Tree-based batch mode reinforcement learning. Journal of Machine Learning Research, Vol.(6): 503-556.

Ernst, D., Glavic, M., Capitanescu, F., Wehenkel, L., (2009) Reinforcement learning versus model predictive control: A comparison on a power system problem. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, 39(2): 517-529.

Fan H, Tarun PK, Chen VCP (2013) Adaptive value function approximation for continuous-state stochastic dynamic programming. *Computers & Operations Research*, 40(4): 1076-1084.

Friedman JH (1991) Multivariate adaptive regression splines. *Annals of Statistics*. 19(1): 1-67.

Heath DC, Jackson PL (1994) Modelling the evolution of demand forecasts with application to safety stock analysis in production/distribution systems. *IIE Transactions*. 26(3): 17-30.

Jiang, DR, Powell WB (2015) Optimal hour-ahead bidding in the real-time electricity market with battery storage using approximate dynamic programming. INFORMS Journal on Computing. 27(3): 525-543.

Jung T, Uthmann T (2004) Experiments in value function approximation with sparse support vector regression. In ECML, pp: 180-191.

Kossmann D, Stocker K (2000) Iterative dynamic programming: a new class of query optimization algorithms. *ACM Transaction on Database Systems*. 25(1): 43-82.

Kuipers, L. Niederreiter, H. (2005) Uniform distribution of sequences. John Wiley & Sons, Inc.

Martinez, D. L. (2013). Variants of Multivariate Adaptive Regression Splines (MARS): Convex vs. Non-convex, Piecewise-linear vs. Smooth and Sequential Algorithms. Ph.D. dissertation, The University of Texas at Arlington.

Martinez, D. L., Shih, D. T., Chen, V. C. P., Kim, S. B. (2015). "A Convex Version of Multivariate Adaptive Regression Splines." Computational Statistics and Data Analysis, 81, pp. 89–106.

Martinez, N., Anahideh, H., Rosenberger, J. M., Martinez, D., Chen, V. C. P., Wang, B. P. (2017). Global optimization of non-convex piecewise linear regression splines. 68(3): 563-586.

Munos, R., Szepesvari, C. (2008) Finite-time bounds for fitted value iteration. Journal of Machine Learning Research 9(May), 815-857.

Parr R, Painter-Wakefield C, Li L, Littman M (2007) Analyzing feature generation for value-function approximation. Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR.

Peng X, Berseth G, van de Panne M (2016) Terrain-adaptive locomotion skills using deep reinforcement learning. *ACM Transactions on Graphics* (*TOG*), 35(4): No. 81.

Powell WB (2011) Approximate dynamic programming: solving the curses of dimensionality. John Wiley, New York.

Riedmiller M (2005) Neural fitted Q iteration-first experiences with a data efficient neural reinforecement learning method. In 16th European Conference on Machine Learning, pp: 317-328.

Salas DF, Powell WB. (2018). Benchmarking a scalable approximate dynamic programming algorithm for stochastic control of grid-level energy storage. INFORMS Journal on Computing. Vol. (30): 106-123.

Sacks J, Welch WJ, Mitchell TJ, Wynn HP (1989) Design and Analysis of Computer Experiments. *Statistical Science*, 4(4): 409-423.

Sobol IM (1967) The distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4): 784-802.

Tsai JCC, Chen VCP, Beck MB, Chen J (2004) Stochastic Dynamic Programming Formulation for a Wastewater Treatment Decision-Making Framework. *Annals of Operations Research, Special Issue on Applied Optimization under Uncertainty*, 132: 207-221.

Tsai, JCC, Chen, VCP (2005). "Flexible and Robust Implementations of Multivariate Adaptive Regression Splines within a Wastewater Treatment Stochastic Dynamic Program." *Quality and Reliability Engineering International*, 21, pp. 689–699

Tsitsiklis JN and Van Roy B (1996) Feature-based methods for large scale dynamic programming. *Machine Learning*, 22:59–94.

Wei Q, Liu D, Shi G, Liu Y (2015) Multibattery optimal coordination control for home energy management systems via distributed iterative adaptive dynamic programming. *IEEE Transactions on Industrial Electronics*, 62(7): 4203-4214.

Wei Q, Liu D, Liu H (2016) Value iteration adaptive dynamic programming for optimal control of discrete-time nonlinear systems. *IEEE Transactions on Cybernetics*. 46(3): 840-853.

Yang Z, Chen VCP, Chang ME, Sattler ML, Wen A (2009) A decision-making framework for ozone pollution control. *Operations Research*, 57(2): 484 498.

Appendix A

The proof of Proposition 2 is given below.

Proposition 2. Consider a constant $\theta \in \mathbb{R}$, and a function $V'\colon \mathcal{X} \to \mathbb{R}$ such that

$$V'(x) = min_{u \in \Gamma} E\{c(x,u,\xi) + \gamma V'(f(x,u,\xi))\} + \theta, \qquad \forall x \in \mathcal{X}. \qquad \text{(A1)}$$

Then, $V'(x) = V(x) + {}^{\theta}/_{1-\gamma}$, $\forall x \in \mathcal{X}$, where $V(x)\colon \mathcal{X} \to \mathbb{R}$ satisfies the Eq. (1).

Proof: for proving convenience, we add a notation for x.

$\forall x_0 \in \mathcal{X}$,

$$V'(x_0) = \min_{u \in \Gamma} E\{c(x_0, u_0, \xi_0) + \gamma V'(x_0, u_0, \xi_0)\} + \theta$$

$$= \min_{u_0 \in \Gamma} E_{\xi_0}\left\{c(x_0,u_0,\xi_0) + \gamma\left[\min_{u_1 \in \Gamma} E_{\xi_1}\{c(f(x_0,u_0,\xi_0),u_1,\xi_1) + \gamma V'(f(f(x_0,u_0,\xi_0),u_1,\xi_1))\}\right]\right\} + \gamma\theta + \theta$$

$$= \min_{u_0,u_1 \in \Gamma} E_{\xi_0,\xi_1}\{c(x_0,u_0,\xi_0) + \gamma c(x_1,u_1,\xi_1) + \gamma^2 V'(f(x_1,u_1,\xi_1))\} + \gamma\theta + \theta$$

$$= \min_{u_0,u_1,\ldots,u_t \in \Gamma} E_{\xi_0,\xi_1,\ldots,\xi_t}\{\textstyle\sum_{\hat{t}=0}^{t} \gamma^{\hat{t}} c(x_{\hat{t}},u_{\hat{t}},\xi_{\hat{t}}) + \gamma^{t+1} V'(f(x_t,u_t,\xi_t))\} + \sum_{\hat{t}=0}^{t} \gamma^{\hat{t}}\,\theta.$$

when $t \to \infty$:

$$V'(x_0) = \min_{u_0,u_1,\ldots, \in \Gamma} E_{\xi_0,\xi_1,\ldots}\{\textstyle\sum_{\hat{t}=0}^{\infty} \gamma^{\hat{t}} c(x_{\hat{t}},u_{\hat{t}},\xi_{\hat{t}})\} + {}^{\theta}/_{1-\gamma}, \qquad \forall x_0 \in \mathcal{X} \qquad \text{(A2)}$$

$= V(x_0) + {}^{\theta}/_{1-\gamma}, \ \forall x_0 \in \mathcal{X}$ (A3)

From the Principle of Optimality (Bellman 1957), there is a unique value function $V(x_0)$, then it must follow:

$V(x_0) = V'(x_0) - {}^{\theta}/_{1-\gamma}, \ \forall x_0 \in \mathcal{X}$ (A4)

In general without notation for *x*,

$V'(x) = V(x) - {}^{\theta}/_{1-\gamma}, \ \forall x \in \mathcal{X}$ (A5)

Appendix B

The proof of Proposition 4 is given below.

Proposition 4: Consider $\delta > 0$ such that $\max_{x\in X^{\text{Test}}}\left|\hat{V}_k(x) - \hat{V}_{k-1}(x)\right| < \delta$, then,

$$\left|b_1^k - 1\right| < \frac{2\delta \sum_{x\in X^{\text{Test}}} |\hat{V}_{k-1}(x)|/|X^{\text{Test}}|}{\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2/|X^{\text{Test}}| - \left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)/|X^{\text{Test}}|\right)^2}, \ x \in X^{\text{Test}} \qquad \text{(B1)}$$

$$1 - \frac{\left|X^{\text{Test}}\right| \delta^2 \left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2 + 2\left|\frac{\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right| \left(\sum_{x\in X^{\text{Test}}} |\hat{V}_{k-1}(x)|\right)\right)}{\left(\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2 - \left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)\right)^2\right)\left(\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_k(x)^2 - \left(\sum_{x\in X^{\text{Test}}} \hat{V}_k(x)\right)^2\right)} < R_k^2 \le 1, x \in X^{\text{Test}} \qquad \text{(B2)}$$

where $\hat{V}(x): X^{\text{Test}} \to \mathbb{R}$ satisfies the Eq. (2).

***Proof for* (B1):**

$\forall x \in X^{\text{Test}}$,

$$b_1^k = \frac{|X^{\text{Test}}| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)\hat{V}_k(x) - \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x) \sum_{x\in X^{\text{Test}}} \hat{V}_k(x)}{|X^{\text{Test}}| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2 - \left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)\right)^2}$$

$$=\frac{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)(\hat{V}_k(x)+\hat{V}_{k-1}(x)-\hat{V}_{k-1}(x))-\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x) \sum_{x\in X^{\text{Test}}}(\hat{V}_k(x)+\hat{V}_{k-1}(x)-\hat{V}_{k-1}(x))}{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2-\left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)\right)^2}$$

$$=\frac{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2-\sum_{x\in X^{\text{Test}}}\left(\hat{V}_{k-1}(x)\right)^2+\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)((\hat{V}_k(x)-\hat{V}_{k-1}(x))-\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x) \sum_{x\in X^{\text{Test}}}(\hat{V}_k(x)-\hat{V}_{k-1}(x))}{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2-\left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)\right)^2}$$

$$=1+\frac{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)((\hat{V}_k(x)-\hat{V}_{k-1}(x))-\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x) \sum_{x\in X^{\text{Test}}}(\hat{V}_k(x)-\hat{V}_{k-1}(x))}{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2-\left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)\right)^2} \qquad \text{(B3)}$$

To obtain the bounds for b_1^k, Eq. (B3) is expressed as:

$\forall x \in X^{\text{Test}}$,

$$b_1^k \leq 1+\frac{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}}\left|\hat{V}_{k-1}(x)\right|\left|\hat{V}_k(x)-\hat{V}_{k-1}(x)\right|+\sum_{x\in X^{\text{Test}}}\left|\hat{V}_{k-1}(x)\right| \sum_{x\in X^{\text{Test}}}\left|\hat{V}_k(x)-\hat{V}_{k-1}(x)\right|}{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2-\left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)\right)^2}$$

$$<1+\frac{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}}\left|\hat{V}_{k-1}(x)\right| \delta+\sum_{x\in X^{\text{Test}}}\left|\hat{V}_{k-1}(x)\right|\left|X^{\text{Test}}\right| \delta}{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2-\left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)\right)^2}$$

$$=1+\frac{2\delta \frac{\sum_{x\in X^{\text{Test}}}\left|\hat{V}_{k-1}(x)\right|}{\left|X^{Test}\right|}}{\frac{\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2}{\left|X^{\text{Test}}\right|}-\left(\frac{\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)}{\left|X^{\text{Test}}\right|}\right)^2}, \qquad \text{(B4)}$$

or

$$b_1^k \geq 1+\frac{-\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}}\left|\hat{V}_{k-1}(x)\right|\left|\hat{V}_k(x)-\hat{V}_{k-1}(x)\right|-\sum_{x\in X^{\text{Test}}}\left|\hat{V}_{k-1}(x)\right| \sum_{x\in X^{\text{Test}}}\left|\hat{V}_k(x)-\hat{V}_{k-1}(x)\right|}{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2-\left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)\right)^2}$$

$$>1+\frac{-\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}}\left|\hat{V}_{k-1}(x)\right| \delta-\sum_{x\in X^{\text{Test}}}\left|\hat{V}_{k-1}(x)\right|\left|X^{\text{Test}}\right| \delta}{\left|X^{\text{Test}}\right| \sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)^2-\left(\sum_{x\in X^{\text{Test}}} \hat{V}_{k-1}(x)\right)^2}$$

$$= 1 - \frac{2\delta\frac{\sum_{x\in X^{\text{Test}}}|\hat{V}_{k-1}(x)|}{|X^{Test}|}}{\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2}{|X^{\text{Test}}|}-\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right)^2} \quad . \tag{B5}$$

Hence, Combining (B4) and (B5) the relationship between b_1^k and δ is

$$\left|b_1^k - 1\right| < \frac{2\delta\sum_{x\in X^{\text{Test}}}|\hat{V}_{k-1}(x)|/|X^{\text{Test}}|}{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2/|X^{\text{Test}}|-(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)/|X^{\text{Test}}|)^2}$$

***Proof for* (B2)**

$$R_k^2 = \frac{(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)\hat{V}_k(x)-|X^{\text{Test}}|\left(\frac{\sum_{x\in X^{Test}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right)\left(\frac{\sum_{x\in X^{Test}}\hat{V}_k(x)}{|X^{\text{Test}}|}\right))^2}{(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2-|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|})^2)(\sum_{x\in X^{\text{Test}}}\hat{V}_k(x)^2-|X^{Test}|(\frac{\sum_{x\in X^{Test}}\hat{V}_k(x)}{|X^{\text{Test}}|})^2)}$$

$$= \frac{(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)(\hat{V}_{k-1}(x)+\Delta_x)-|X^{\text{Test}}|\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right)\left(\frac{\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)}{|X^{\text{Test}}|}\right))^2}{(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2-|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|})^2)(\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)^2-|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)}{|X^{\text{Test}}|})^2)}$$

$$=1 - \frac{(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2)(\sum_{x\in X^{\text{Test}}}{\Delta_x}^2)+2|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|})(\frac{\sum_{x\in X^{\text{Test}}}\Delta_x}{|X^{\text{Test}}|})(\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)\Delta_x)}{(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2-|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|})^2)(\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)^2-|X^{Test}|(\frac{\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)}{|X^{\text{Test}}|})^2)} +$$

$$+\frac{|X^{Test}|(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|})^2(\sum_{x\in X^{\text{Test}}}{\Delta_x}^2)+|X^{\text{Test}}|\left(\frac{\sum_{x\in X^{\text{Test}}}\Delta_x}{|X^{\text{Test}}|}\right)^2\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2+(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)\Delta_x)^2}{\left(\sum_{x\in X^{Test}}\hat{V}_{k-1}(x)^2-|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|})^2\right)\left(\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)^2-|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)}{|X^{\text{Test}}|})^2\right)} \tag{B7}$$

Since $\frac{|X^{Test}|(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|})^2(\sum_{x\in X^{\text{Test}}}{\Delta_x}^2)+|X^{\text{Test}}|\left(\frac{\sum_{x\in X^{\text{Test}}}\Delta_x}{|X^{\text{Test}}|}\right)^2\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2+(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)\Delta_x)^2}{\left(\sum_{x\in X^{Test}}\hat{V}_{k-1}(x)^2-|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|})^2\right)\left(\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)^2-|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)}{|X^{\text{Test}}|})^2\right)} \geq 0$,

hence,

$$R_k^2 \geq 1 - \frac{(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2)(\sum_{x\in X^{\text{Test}}}{\Delta_x}^2)+2|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|})(\frac{\sum_{x\in X^{\text{Test}}}\Delta_x}{|X^{\text{Test}}|})(\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)\Delta_x)}{(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2-|X^{\text{Test}}|(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|})^2)(\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)^2-|X^{Test}|(\frac{\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)}{|X^{\text{Test}}|})^2)}$$

$$> 1 - \frac{\left|X^{\text{Test}}\right|\delta^2\left(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2\right)+2\left|X^{\text{Test}}\right|\delta^2\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right)\left(\sum_{x\in X^{\text{Test}}}|\hat{V}_{k-1}(x)|\right.}{\left(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2-\left|X^{\text{Test}}\right|\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right)^2\right)\left(\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)^2-\left|X^{\text{Test}}\right|\left(\frac{\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)}{|X^{\text{Test}}|}\right)^2\right)}$$

$$= 1 - \frac{\left|X^{\text{Test}}\right|\delta^2\{\left(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2\right)+2\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right)\left(\sum_{x\in X^{\text{Test}}}|\hat{V}_{k-1}(x)|\right\}}{\left(\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2-\left|X^{\text{Test}}\right|\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right)^2\right)\left(\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)^2-\left|X^{Test}\right|\left(\frac{\sum_{x\in X^{\text{Test}}}(\hat{V}_{k-1}(x)+\Delta_x)}{|X^{\text{Test}}|}\right)^2\right)}$$

$$= 1 - \frac{\delta^2\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2}{|X^{\text{Test}}|}+2\left|\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right|\left(\frac{\sum_{x\in X^{\text{Test}}}|\hat{V}_{k-1}(x)|)}{|X^{\text{Test}}|}\right)\right)}{\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2}{|X^{\text{Test}}|}-\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right)^2\right)\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k}(x)^2}{|X^{\text{Test}}|}-\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k}(x)}{|X^{\text{Test}}|}\right)^2\right)} < R_k^2 \leq 1. \qquad \text{(B8)}$$

Since $R_k^2 \leq 1$, therefore,

$$1 - \frac{\delta^2\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2}{|X^{\text{Test}}|}+2\left|\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right|\left(\frac{\sum_{x\in X^{\text{Test}}}|\hat{V}_{k-1}(x)|)}{|X^{\text{Test}}|}\right)\right)}{\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)^2}{|X^{\text{Test}}|}-\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k-1}(x)}{|X^{\text{Test}}|}\right)^2\right)\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k}(x)^2}{|X^{\text{Test}}|}-\left(\frac{\sum_{x\in X^{\text{Test}}}\hat{V}_{k}(x)}{|X^{\text{Test}}|}\right)^2\right)} < R_k^2 \leq 1\,, x \in X^{\text{Test}}.$$