An Alternating Maximization Approach to Approximate Dynamic Programming*

Click here for the latest version

Adam Dearing† Yichun Song‡

March 19, 2023

Abstract

We propose a new unconstrained maximization approach to approximate the value and policy function in the single-agent dynamic programming context. Our method extends Howard's policy iteration into approximate dynamic programming via successively optimizing over the value or policy function in an unconstrained maximization problem. Our algorithm is flexible, allowing researchers to choose which part to approximate while solving the other part exactly. This flexibility enables researchers to utilize the characteristics of a specific problem. When polynomial approximations are adopted, the objective function is concave across various models, and we show that our algorithm is hill-climbing with a convergence guarantee under concavity. Additionally, a lower and upper bound for a weighted average value function can be obtained following a slight adjustment of our method. Simulations on a bus engine replacement problem and a stochastic growth model demonstrate that our algorithm provides significant computational gains, especially with a large state space and without high-speed computing devices.

Keywords: Dynamic programming, Approximation, Concave Maximization.

JEL classification codes: C61, C63.

*All errors are our own.

†Cornell University and NBER. SC Johnson Graduate School of Management, SC Johnson College of Business, 326 Sage Hall, 114 E Ave., Ithaca NY 14853, United States. Email: aed237@cornell.edu.

‡Department of Economics, The Ohio State University, Columbus OH 43210-1120. Email: song.1399@osu.edu.

1 Introduction and Literature Review

Dynamic programming furnishes researchers in economics and business with a convenient framework for solving forward-looking agents' problems. In the infinite horizon, Bellman optimality applies, and the problem is solved by finding a fixed point in the space of the value function V or the policy function p (Hotz and Miller, 1993). However, the so-called "curse of dimensionality" arising from large state spaces can render dynamic programming practically intractable, especially for those who lack access to high-speed computing techniques.

Many studies have contributed to the computational issue of the time-consuming fixed point iteration. To solve the exact problem, for instance, Howard's policy iteration in Ljungqvist and Sargent (2004), Hotz and Miller (1993) or Aguirregabiria and Mira (2002) has been widely adopted for structural models, where the fixed point is decomposed into an iterative algorithm over V and p alternately. Bray (2019) proves that the fixed point iteration should be done using the difference in V, which leads to a faster computational speed and numerical stability. Ma and Stachurski (2021) also considers the transformations of value function to boost the computational efficiency. Rust (1997) and Bray (2022) jointly depict the restricted alleviation of the curse of dimensionality through the random grid.

Instead of pursuing an exact solution, an approximation using parameterized function classes can relieve the computational difficulty. [1] A common way is approximating the functions via a linear combination of bases. For instance, the value function, V, could be approximated by calculating a parameter vector γ over a pre-selected set of bases Ψ, so that $V \approx \Psi\gamma$.

Approximate dynamic programming remains an active field of research, and there have been several advances in recent years. Cai and Judd (2014) provide a comprehensive tutorial on shape-preserving dynamic programming. Brumm and Scheidegger (2017) use an adaptive sparse grid to approximate the solution, illustrated by an application to an IRBC model. Kristensen et al. (2021) targets a smoothed version of integrated or expected value function, adopts both polynomial approximation and the soft-max operator, and illustrates the behavior of those methods. Moreover, researchers have explored many other methods for approximation, like artificial neural network (Norets, 2012), decision trees in (Boutilier et al., 2000) and Fuzzy rule-based systems (Jouffe, 1998).

1. See, e.g., Powell (2009) for a brief review of approximate dynamic programming.

Approximate dynamic programming has proliferated into empirical works of economics, environmental and operation research, such as large-scale fleet management in Simão et al. (2009), long-term costs in economics and dynamic climate change in Cai and Lontzek (2019), costumer-service capacities allocation in Schütz and Kolisch (2012), dynamic portfolio optimization in Cai et al. (2020), and large-scale oligopoly models in industrial organization like Farias et al. (2012) and Fan and Yang (2022). Empirical studies designate that approximate dynamic programming is a powerful tool for tackling real-life problems.

In this study, we introduce a new method into the approximate dynamic programming literature. Our method computes the value and policy function as the solution to an unconstrained concave optimization problem. The critical insight behind our method is that the constrained linear programming formation to approximate V of Schweitzer and Seidmann (1985) and de Farias and Van Roy (2003) can be reformulated into an unconstrained optimization problem, so the state-wise inequality constraints in de Farias and Van Roy (2003) are no longer needed. Also, we show that by introducing polynomial approximation and smoothing techniques, our method leads to a lower computational burden than other commonly-used methods.

We format the fixed point iteration as an alternating optimization over V and p one after another, and we show that each iteration can be expressed as an unconstrained concave optimization when V and p are solved exactly, or under some proper approximation techniques, like the additive polynomial series and smoothing-maximum techniques. Our algorithm is a hill climber on an upper-bounded function, which guarantees convergence. Under low-dimensional approximations, the objective function value corresponding to the solution delivers a lower bound for the weighted summation of the value function. Moreover, our method is flexible since one can choose which step to approximate and which to solve exactly by exploiting the problem-specific structure. For instance, we can take advantage of the analytical conditional choice probability (CCP) of dynamic discrete choice with Logit error still while approximating the V.

Our method provides an extension of Howard's policy iteration to approximate dynamic programming. It is a true extension – when it is applied with full flexibility in both V and p (i.e., the approximation error goes to zero), our algorithm is equivalent to Howard's policy iteration. Our algorithm still preserves the key concepts of Howard's policy iteration when using approximations: we alternate between optimizing over V and p, and the iteration gives monotone convergence. With

approximation, our method delivers a lower bound to the weighted average of the value function with state-relevance weights pre-specified by the researcher. Additionally, we show that an upper bound can be obtained through a similar – but distinct – unconstrained convex minimization problem.

We illustrate our method through two types of dynamic programming problems – the classic dynamic discrete choice (Rust, 1987) and the Macro-type stochastic growth baseline model with CRRA utility function (Brock and Mirman, 1972), which are both compatible with Howard's formation. We conduct a rich set of Monte Carlo simulations for these models. For V-approximation, keeping the state space the same, our method outperforms all other existing methods in terms of the elapsed time. In the stochastic growth model, when the policy function p does not have an analytical form, our policy approximation provides a reasonable computational improvement, especially for those without a high-end parallel computing toolbox.

The rest of this paper is organized as follows. Section 2 provides an overview of our approximation method with a detailed description of the two examples we focus on. Section 3 derives properties and illustrates how the unconstrained concave alternating algorithm is constructed. 3 also discusses the error bound from polynomial approximation and smoothing techniques. Section 4 emphasizes some other issues regarding the actual implementation, as well as procedures for constructing upper bounds. Section 5 has a large set of Monte Carlo simulations displayed, with section 5.1 on the dynamic discrete choice and section 5.2 on the stochastic growth model, demonstrating the behavior of our estimator in both cases. Section 6 is the conclusion and future research directions. Appendices contain the proofs and simulations omitted in the main text.

2 Description of Method

Consider a standard infinite-horizon dynamic programming model, with state $x_t \in \mathcal{X} \subset \mathbb{R}^d$, $\mathcal{X}$ compact[2]. For now, we assume that the state space is finite: $\mathcal{X} = \{1, 2, \ldots, |\mathcal{X}|\}$. Actions are chosen from a compact set, $a_t \in \mathcal{A}$. The flow utility or payoff is $u(a_t, x_t) \in \mathbb{R}$ is bounded (e.g., continuous on the compact set $\mathcal{A} \times \mathcal{X}$). A policy function, $p(x)$, designates actions for each state, $p(x_t) = a_t$. Markov transitions are governed by a conditional distribution $f(x_{t+1} \mid x_t)$, and the discount factor is $\beta \in (0, 1)$.

2. Extension to finite-horizon problems is trivial, by making time a state variable and adding a terminal state once time exceeds the maximum number of periods.

The value associated with state x under policy p, $V_p(x)$, is the expected discounted reward from following policy p when the initial state is x:

$$V_p(x) = E\left[\sum_{t=0}^{\infty} \beta^t u(p(x_t), x_t) \mid x_0 = x\right].$$

This leads to the classic Bellman operators. Under a finite state space, let U_p denote a vector of flow utilities under policy p and the matrix F_p of expected transitions under policy p. Let $T(p, V)$ be the usual Bellman operator for an infinite horizon dynamic programming model, with policy, p, and value function, V. Then,

$$T(p, V) = U_p + \beta F_p V \qquad T(V) = \max_{p \in \mathcal{A}^{|\mathcal{X}|}} \quad T(p, V), \tag{1}$$

where the maximum is taken component-wise. Equation (1) incorporates a general class of dynamic problems, which contains and not limited to, the two typical examples reviewed below.

Example 1. Single-agent Dynamic Discrete Choice with Logit error. *Conditional Choice Probability (CCP) representation of the single-agent dynamic discrete choice model, discussed in Hotz and Miller (1993) and Aguirregabiria and Mira (2002), is compatible with the setting in (1). With discrete actions: $a_t \in \mathcal{A} = \{0, 1, 2, \cdots, |\mathcal{A}| - 1\}$, we assume that true underlying data generating process leads to an ergodic Markov chain and a unique stationary distribution $\pi(x)$. As in the standard setting, the flow payoff is additively separable in the unobservables and equals to $u_t(a_t, x_t) = \bar{u}_t(a_t, x_t) + \varepsilon_t(a_t)$. By assuming a Logit distribution for the error term $\varepsilon_t(a_t)$, the standard Bellman operator can be written as,*

$$T(p, V) = \underbrace{\sum_{a \in \mathcal{A}} P(a)\{U(a) + e(a, P)\}}_{U_p} + \beta \Big[\underbrace{\sum_{a \in \mathcal{A}} P(a) * F_a}_{F_p}\Big] V \tag{2}$$

where $P(a)$ is the column vector that stack of $p(a, x)$ for all the states. Similar notation applies to $U(a)$ and $e(a, P)$. As shown in Aguirregabiria and Mira (2002), Logit error allows an analytical solution to the conditional expectation of $\varepsilon(a)$ such that $e(a, P) =$ Euler's constant $- \ln P(a, x)$[3].

Example 2. Stochastic Growth Model. *In contrast to the discrete choice, models in Macroe-*

3. An analytical solution also exists if the error terms are independently and identically distributed with a normal distribution – see Aguirregabiria and Magesan (2013).

conomics embrace continuous choices. Consider the stochastic growth model stemmed from Brock and Mirman (1972). The state space is $x_t = (k_t, z_t)'$ *and a social planner maximizes*

$$E\Big[\sum_{t=0}^{\infty} \beta^t u(c_t)\Big], \quad c_t + k_{t+1} = (1-\delta)k_t + Az_t k_t^{\nu}, \quad k_{t+1} \geq 0 \tag{3}$$

The choice of current consumption c_t *and future capital stock* k_{t+1} *is based on the information of capital stock* k_t *and the current realization of productivity shocks* z_t*.* z_t *is a time-invariant Markov process with transition matrix* $Q_{z'|z}$*, which does not depend on the current capital stock* k_t*.* δ *is the capital depreciation rate.* A *is the fixed part of the productivity shock, and the production function follows a Cobb-Douglas function with* $0 < \nu < 1$*.* $u(c_t)$ *is a concave CRRA utility function where* $u(c_t) = c_t^{1-\lambda}/(1-\lambda)$*,* $\lambda \neq 1$*,* $u(c_t) = \ln(c_t)$*,* $\lambda = 1$*. Using Bellman principle, the dynamic system can be written as*

$$V(k,z) = \max_{c \geq 0} \left\{ u(c) + \beta \int V(Azk^{\nu} + (1-\delta)k - c, z')Q_{z'|z}(z)dz' \right\} \tag{4}$$

Usually this problem does not have an analytical solution unless $\delta = 1$ *and* $\lambda = 1$[4]*. Following the notation in equation (1), with* c *as policy function and the discretization of both* k *and* z *into* m_k *and* m_z *points, equation (4) becomes,*

$$T(c,V) = \underbrace{U(c)}_{U_p} + \beta \underbrace{\left[F_V \otimes F_{z'|z}\right]}_{F_p} V \tag{5}$$

for a stacked value function V *with dimensions* $(m_k \times m_z) \times 1$*.* F_V *is a* $m_k \times m_k$ *matrix,* $F_V = (F_{V,1}, F_{V,2}, \cdots, F_{V,m_k})'$*, where* $F_{V,i}$ *is a* m_k*-by-1 vector, indicating the position of* k_i' *after consuming* c_i*.* $F_{z'|z}$ *is the transition matrix of* z *after the discretization. Then* F_p *becomes a* $(m_k \times m_z)$ *by* $(m_k \times m_z)$ *squared matrix.*

In this study, we only focus on the well-defined dynamic problems that have both $T(p,V)$ and $T(V)$ in (1) are contraction mappings with unique fixed point. The value function associated with policy p, V_p, is the unique solution to $V_p = T(p, V_p)$. Formally speaking,

Assumption 1. $\forall\ p \in \mathcal{P}$*, there is* $T(p,V) = V$ *if and only if* $V = V_p$*.*

4. In this case, the dynamic system can be analytically solved and the optimal policy can be represented by $k_{t+1} = \beta\nu A z_t k_t^{\nu}$.

The value function associated with the optimal policy, V^*, satisfies $V^* = T(V^*)$, with corresponding optimal policy, $p^* = \arg\max_p T(p, V^*)$. The maximum again are taken component-wise. Note that there is $V^* = V_{p^*}$.

Because the Bellman operators are contraction mappings in V, standard fixed point iteration can solve for V_p and V^*. For V^*, policy iteration can also be used with a computational improvement when the maximization over V is computationally burdensome. However, both procedures can be computationally impractical when the state space is large.

These issues for computation have driven researchers to explore approximation methods for the value or policy functions. One method for $V-$approximation conditional on the p that is particularly relevant to our work is the linear programming approach, first proposed by Schweitzer and Seidmann (1985) and then developed by de Farias and Van Roy (2003). de Farias and Van Roy (2003) approximately solves V through a linear program with the number of constraints equal to $|\mathcal{A}| \times |\mathcal{X}|$.

de Farias and Van Roy (2003)'s approach is appealing and has been successful in applications, as illustrated in Section 1, but it still possesses a few significant limitations. First, as pointed out by de Farias and Van Roy (2003), the size of the linear programming still grows exponentially with the state space, even after the number of variables is reduced dramatically by polynomial approximation. Second, the approximated results only provide a lower bound on V^*. Third, it does not provide guidance on how an approximate p should be chosen.

To overcome these drawbacks, we propose a new approach for the approximation of V and p, which extends Howard's policy iteration to approximate dynamic programming. Specifically, let $m(p, V) = \min_x\{T(p, V)(x) - V(x)\}$. We show that, the usual (V^*, p^*) in a discounted reward setting can be formulated as the solution to an unconstrained optimization problem:

$$\max_{p,V} \quad w'T(p, V) + \beta(1-\beta)^{-1}m(p, V) \tag{6}$$

where $w = (w_1, w_2, \cdots, w_{|\mathcal{X}|})'$ are the weights put in each state with $\sum_x w(x) = 1$ and $w(x) \geq 0$ for each x. This problem is concave in V for any p, and is concave in p wherever T is concave in p, as proved in Section 3. Thus, the objective function can be maximized via bi-concave optimization, alternating between two problems:

Value problem (fixing a policy).

$$\max_{V} \quad w'T(p,V) + \beta(1-\beta)^{-1}m(p,V) \tag{7}$$

Policy problem (fixing a value function).

$$\max_{p} \quad w'T(p,V) + \beta(1-\beta)^{-1}m(p,V) \tag{8}$$

Therefore, we compute (V,p) as solutions to the unconstrained maximization problem in (6) by alternating between value problem (7) and policy problem (8). With full flexibility on V and p, alternating between these two problems is identical to Howard's policy iteration. When using approximation methods like polynomial, alternating between *approximated* V or p functions gives a reasonable solution. A fixed point of this alternating procedure provides (i) an approximately-optimal policy function; (ii) a lower bound of the weighted average $w'V^*$ through the objective function value. Section 4 also discusses procedures for generating upper bounds on $w'V_p$ and $w'V^*$, which differ from the procedure used to construct the lower bound here.

3 Analysis and Proofs

3.1 Derivation of the Unconstrained Optimization

Define $H(p,V) \equiv T(p,V) - V$. The following Proposition 1 provides some useful properties of $H(\cdot)$ that would be applied in future proofs.

Proposition 1. *$H(p,V)$ has the following properties:*

1. *$H(p,V_p) = 0$, and $H(p,V) \neq 0$ for all $V \neq V_p$.*
2. *$H(\cdot)$ is concave in V.*
3. *If $T(\cdot)$ is concave in p on a set $\mathcal{P} \times \mathcal{V} \subset \mathcal{A} \times \mathbb{R}^{|\mathcal{X}|}$, then $H(\cdot)$ is concave in p on $\mathcal{P} \times \mathcal{V}$.*
4. *$H(\cdot)$ is* inverse-antitone *in V: if $H(p,V_1) \leq H(p,V_2)$, then $V_1 \geq V_2$, where the inequalities are applied element-wise.*
5. *$\nabla_V H(p,V) = -(I_{|\mathcal{X}|} - \beta F_p)$, where F_p represents the unconditional transition matrix associated with policy p.*

6. $H(p, V + k\iota) = H(p, V) - (1 - \beta)k\iota$ *for all* $k \in \mathbb{R}$, *where* ι *is the vector of ones.*

Proof. See Appendix A. □

The properties of $H(\cdot)$ are essential for our optimization problem. Our unconstrained optimization then follows from the successive lemmas and propositions.

Lemma 1. *If* $H(p, V) \geq 0$, *then* $V \leq T(p, V) \leq V_p$. *Similarly, if* $H(p, V) \leq 0$, *then* $V_p \leq T(p, V) \leq V$.

Proof. We start by proving the first statement. Assume $H(p, V) \geq 0$. Then, $T(p, V) \geq V$ follows immediately from the definition of $H(p, V)$. Additionally, Proposition 1.1 and 1.4 imply that $V \leq V_p$. Since $T(\cdot)$ is isotone, or in other words, if $V_1 \leq V_2$, then $T(p, V_1) \leq T(p, V_2)$, this implies $T(p, V) \leq T(p, V_p) = V_p$. This proves the first statement. The proof of second statement comes through a similar logic. □

Proposition 2. *The optimal policy and value function satisfy*

$$\begin{aligned}(p^*, V^*) = \underset{(p,V)\in\mathcal{A}\times\mathbb{R}^{|\mathcal{X}|}}{arg\ max} \quad & w'T(p, V) \\ s.t. \quad & H(p, V) \geq 0.\end{aligned} \tag{9}$$

Proof. First, note that $H(p^*, V^*) = 0$, so (p^*, V^*) is a feasible solution to (2). Second, by Lemma 1, we have $T(p, V) \leq V_p$ in the feasible region or when $H(p, V) \geq 0$. And $V_p \leq V^*$ for all p since only the optimal policy p^* leads to the highest value V. We then have $T(p, V) \leq V^*$ in the feasible regions. Therefore, for all feasible pairs (p, V), $w'T(p, V) \leq w'V^*$, with equality at (p^*, V^*). □

Problem in (2) is *not* a concave maximization problem in (p, V). However, if we consider alternating between maximization over p and maximization over V individually, we obtain the standard Howard policy iteration algorithm.

To see why, first consider holding p fixed and maximizing over V. That is, the problem

$$\begin{aligned}\max_{V} \quad & w'T(p, V) \\ s.t. \quad & H(p, V) \geq 0.\end{aligned} \tag{10}$$

Problem in (10) is a linear programming because both $T(\cdot)$ and $H(\cdot)$ are linear in V, and the solution to this linear programming is V_p – making it equivalent to the value step in Howard's policy iteration. (10) has been well-studied in the dynamic programming literature, and it leads naturally to an approximated version as in de Farias and Van Roy (2003).

Then, consider fixing V and maximizing over p:

$$\begin{aligned} \max_{p} \quad & w'T(p,V) \\ s.t. \quad & H(p,V) \geq 0. \end{aligned} \tag{11}$$

On any convex set, $\mathscr{P}$, such that $T(\cdot, V)$ is concave in p, (11) is a concave optimization problem over the convex set $\mathscr{P} \cap \{p : H(p,V) \geq 0\}$. Its solution is $\arg\max_p T(p,V)$, where the maximum is taken component-wise over the state space. So, (11) is equivalent to the policy improvement step in Howard's policy iteration. Furthermore, note that for a well-defined problem, there is $V_{p'}$ corresponding to p'. Choosing $V_{p'}$ for some p' makes (11) feasible because $H(p', V_{p'}) = 0$.

Together this shows that (2) can be solved by alternating between maximizing over V and maximizing over p, since that procedure is equivalent to Howard's policy iterations. As mentioned in Section 2, the main obstacle for both (10) and (11) is the large number of constraints for large state spaces. Even when the value and policy functions are approximated, the issue with the number of constraints remains.

This study shows that we can circumvent the many-constraint problem by replacing constrained maximization with an equivalent unconstrained maximization. To show this, we recall the scalar-valued function $m(p,V) \equiv \min_x H(p,V)(x) = \min_x\{T(p,V)(x) - V(x)\}$. Lemma 2 contains the properties regarding $m(p,V)$.

Lemma 2. *(i) $m(p,V)$ is concave in V, and it is also concave in p on any set, $\mathscr{P} \times \mathcal{V}$, where $T(p,V)$ is concave in p; (ii) $H(p, V + (1-\beta)^{-1} m(p,V)\iota) \geq 0$; and (iii) $V + (1-\beta)^{-1} m(p,V)\iota \leq T(p,V) + \beta(1-\beta)^{-1} m(p,V)\iota \leq V_p$.*

Proof. Lemma 2.(i) follows from (a) the minimum function is concave, isotone; and (b) $H(p,V)$ is linear in V, and it is and concave in p on any $\mathscr{P} \times \mathcal{V}$ where $T(p,V)$ is concave in p (Proposition 1.3). Thus, the composite $m(p,V) = \min_x H(p,V)(x)$ is concave in V everywhere and concave in p on $\mathscr{P} \times \mathcal{V}$. Lemma 2.(ii) follows from the fact that $H(p, V + k\iota) = H(p,V) - (1-\beta)k\iota$ for scalar

κ (Proposition 1.6). This implies that

$$\begin{aligned} H(p, V + (1-\beta)^{-1} m(p,V)\iota) &= H(p,V) - m(p,V)\iota \\ &= H(p,V) - \min_x H(p,V)(x)\iota \ \geq 0. \end{aligned}$$

Lemma 2.(iii) follows from combining the second property, along with Proposition 1.4 and Lemma 1, since $T(p, V + (1-\beta)^{-1} m(p,V)\iota) = T_p(V) + \beta(1-\beta)^{-1} m(p,V)\iota$. □

We then use Lemma 2 for a normalization to the objective function of (2) that renders the constraints in the linear programming (10) and (11) unnecessary. The objective function for this unconstrained optimization is,

$$L(p,V) \equiv w'T(p,V) + \beta(1-\beta)^{-1} m(p,V). \tag{12}$$

Proposition 3. *1. The optimal policy p^* and value function V^* satisfy*

$$(p^*, V^*) \in \underset{(p,V)\in\mathcal{A}\times\mathbb{R}^{|\mathcal{X}|}}{arg\ max} \quad L(p,V);$$

2. $L(p^*, V^*) = w'V$;

3. $L(\cdot)$ is concave in V, and concave in p on any set $\mathcal{P}\times\mathcal{V}$, when $T(p,V)$ is concave in p.

Proof. For Proposition 3.2: follows from $T(p^*, V^*) = V^*$ and $m(p^*, V^*) = 0$. For Proposition 3.3: follows from Lemma 2.(i). For Proposition 3.1: with Proposition 3.2 already proven, it suffices to show that $L(p,V) \leq w'V^*$ for all (p,V). We have

$$\begin{aligned} L(p,V) &= w'T(p,V) + \beta(1-\beta)^{-1} m(p,V) \\ &= w'T(p,V) + \beta(1-\beta)^{-1} m(p,V) w'\iota \\ &= w'\left(T(p,V) + \beta(1-\beta)^{-1} m(p,V)\iota\right) \\ &= w'T\left(p, V + (1-\beta)^{-1} m(p,V)\iota\right), \end{aligned}$$

where the second equality follows from $w'\iota = 1$. It then follows from Lemma 2.(ii) and Lemma 1 that

$$L(p,V) \leq w'V_p \leq w'V^* \tag{13}$$

for each (p, V). The second inequality arises because $V_p \leq V^*$ for all p and $w(x) \geq 0$ for each x. $\square$

In Proposition 3, we see how the previous constrained maximization problem in Proposition 2 is replaced with an unconstrained maximization problem. Like the constrained problem, this problem is not jointly concave in (p, V), but concave in V alone and in p alone wherever $T(p, V)$ is concave in p alone. Thus, we consider solving the problem via alternating maximization over two unconstrained concave optimizations like Howard's policy iteration, leading to the algorithm below.

Algorithm 1. *A (local) maximum of $L(p, V)$ can be found through the following steps.*

1. *Choose an initial approximate policy, p^0.*
2. *For $h \geq 1$, update: $V^h = arg\ max_V\ L(p^{h-1}, V)$.*
3. *Update: $p^h = arg\ max_p\ L(p, V^h)$. $h = h + 1$.*
4. *Iterate steps (2) and (3) until convergence, or until $h = \bar{h}$ steps.*

In Algorithm 1 step 2, we fix a policy p and maximize $L(p, V)$ over V. The solution to this is V_p, which can be seen by combining inequality (13) and $L(p, V_p) = w'V_p$ through the fixed-point definition of V_p. In step 3, take V as given and maximize $L(p, V)$ over p, which is equivalent to computing $\arg\max_p T(p, V)$ separately for each state x since (12) is monotonically increase with $T(p, V)(x)$. Thus, our Algorithm 1 achieves each step and inherits properties in Howard's policy iteration. Specifically, Algorithm 1 is an ascent algorithm on the objective function. The increasing sequence of objective function values has a limit by the monotone convergence theorem, since the objective function is bounded above by $w'V^*$. Thus, Algorithm 1 comes with a type of convergence guarantee, which is uncommon in the approximate dynamic programming literature.

One practical issue that needs to be addressed for Algorithm 1 is that the maximization over V only identifies the value function up to a location normalization. For $V + k\iota$, the value of the objective function (12) does not change with k. Any $V_p + k\iota$ would be a solution, and the arg max contains an infinite set. We can avoid this issue by imposing a simple normalization: $V(x_0) = 0$ for some x_0. A convenient choice is often $x_0 = 0$, since the state space can usually be normalized to include the origin. This normalization reduces the problem to a search over the rest of V and has a unique solution. It is still a concave maximization problem. This normalization would also

be convenient when we consider sieve or polynomial approximation[5]. We then obtain the level by computing $T(p,V)+\beta(1-\beta)^{-1}m(p,V)\iota$.

3.2 Polynomial Approximation and Smoothing Algorithm

However, Algorithm 1 still suffers from the well-known curse of dimensionality of state space in dynamic programming when solving the exact optimization. An unconstrained concave optimization like (12) can be computationally hard to solve when the number of states grows extremely large. Here we show that our Algorithm 1 can accommodate large state space by polynomial approximation and smoothing algorithms, which have gained popularity in recent years (Kristensen et al., 2021). By incorporating these techniques, while still retaining a good approximation, our algorithm reveals its computational gain over many existing methods.

The size of the (12) can be further reduced from $|\mathcal{X}|$ by choosing a finite-dimensional function space, or the sieve in econometrics, that may have dimension $K \ll |\mathcal{X}|$. Specifically, consider function spaces that could provide a good approximation for V and p as $\mathcal{V}_{K_v}=\{V_K(\cdot\,;\gamma):\mathcal{X}\mapsto\mathbb{R}|\gamma\in\Upsilon_{\gamma_{K_v}}\}$ and $\mathcal{P}_{K_p}=\{p_K(\cdot\,;\alpha):\mathcal{X}\mapsto\mathbb{R}^{|\mathcal{A}|}|\alpha\in\Upsilon_{\alpha_{K_p}}\}$. $\Upsilon_{\gamma_{K_v}}$ and $\Upsilon_{\alpha_{K_p}}$ are parameter sets with $K_v\leq|\mathcal{X}|$ and $K_p\leq|\mathcal{X}|$. Kristensen et al. (2021) proposes that a smoothed version of V or p function, for instance, the integrated value function, would help justify using well-developed nonparametric techniques. Fortunately, due to the nature of Howard's policy iteration and the definition of $T(p,V)$, we are likely to have smoothed functions suitable for such approximation.

A leading example among those well-developed sieve estimator refer to a linear combination of basis functions. That is, the K-dimensional function space becomes, $\mathcal{V}_{K_v}=\{\Psi_K(x)\gamma:\gamma\in\mathbb{R}^{K_v}\}$ and $\mathcal{P}_{K_p}=\{\Phi_K(x)\alpha:\alpha\in\mathbb{R}^{|\mathcal{A}|\times K_p}\}$. In other words, V and p are approximated by $\sum_k^{K_v}\gamma_k\psi_k(x)$ and $\sum_k^{K_p}\alpha_k\phi_k(x)$. The basis function Ψ and Φ can be chosen from, for instance, B-spline or Chebyshev polynomials – those methods have demonstrated their ability in function approximation. Regularization can be easily incorporated into our setting[6]. However, nonlinear function spaces like neural nets may lead to the failure of concavity, so they may not be recommended.

5. Omitting a constant from the basis functions for approximation is the same as imposing the normalization $V(0)=0$. Interestingly, the normalized problem is equivalent to maximization over the relative value function considered in Bray (2019). Bray (2019) shows that all policy-relevant information in the value function is contained in the relative value function.

6. For example, let $||\cdot||_p$ denote the p-norm of a vector. Then, we can define

$$\tilde{L}(\alpha,\gamma)\equiv w'T(\Phi\alpha,\Psi\gamma)+\beta(1-\beta)^{-1}m_\sigma(\Phi\alpha,\Psi\gamma)+\mu_1||\alpha||_p+\mu_2||\gamma||_p,$$

where μ_1 and μ_2 are hyper-parameters. From here, the alternating procedure can be straightforwardly adapted.

Another potential issue with (12) is the non-smoothness introduced through the min function in $m(p, V)$. Smoothness can be approximately restored by replacing $m(p, V)$ with

$$m_\sigma(p, V) \equiv -\sigma \ln \left(\sum_x \exp\{-\sigma^{-1} H(p, V)(x)\} \right) \tag{14}$$

(14) yields smoothness at the cost of a smoothing error. Since $\min_x f(x) = -\max_x\{-f(x)\}$, applying the error bounds in Gao and Pavel (2018) gives,

$$\max_x f(x) \leq \sigma \ln \left(\sum_x \exp\{\sigma^{-1} f(x)\} \right) \leq \max_x f(x) + \sigma \ln(|\mathcal{X}|) \tag{15}$$

From (15) we can see the smoothing error tends to increase when σ increase or the number of states increases and can be made arbitrarily small since $m_\sigma(\cdot) \to m(\cdot)$ uniformly as $\sigma \to 0$. However, Polak et al. (2003) points out that the smoothing approximation becomes ill-conditioned as the approximation accuracy increases, which may lead to numerical difficulties and slower convergence for the solver.

Combining the two approximation techniques, we rewrite the exact optimization (12) using its approximated counterparts. The smoothed version $\tilde{L}(\Phi\alpha, \Psi\gamma)$ replaces $L(p, V)$ in Algorithm 1 such that,

$$\tilde{L}(\Phi\alpha, \Psi\gamma) = w'T(\Phi\alpha, \Psi\gamma) + \beta(1-\beta)^{-1} m_\sigma(\Phi\alpha, \Psi\gamma). \tag{16}$$

By polynomial approximation, the size of the problem (16) for value iteration is reduced to K. The gain of approximating is higher in our case since we have already eliminated the large set of linear constraints. Note that (12) is encompassed as a special case of (16). If $\sigma = 0$ and both Ψ and Φ are $|\mathcal{X}|$-dim identity matrix, $L(p, V)$ is the same as $\tilde{L}(\Phi\alpha, \Psi\gamma)$. Moreover, we define $\bar{L}(\Phi\alpha, \Psi\gamma)$ as the case where we allow polynomial approximation but not σ-smoothing,

$$\bar{L}(\Phi\alpha, \Psi\gamma) = w'T(\Phi\alpha, \Psi\gamma) + \beta(1-\beta)^{-1} m(\Phi\alpha, \Psi\gamma). \tag{17}$$

Algorithm 2, illustrates the alternating steps with $\tilde{L}(\Phi\alpha, \Psi\gamma)$ in (16),

Algorithm 2. *Find a (local) maximum of $\tilde{L}(\Phi\alpha, \Psi\gamma)$ through the following steps.*

1. *Choose an initial approximate policy, $\tilde{p}^0$, with corresponding parameter, α^0.*
2. *For $h \geq 1$, update: $\gamma^h = arg\ max_\gamma\ \tilde{L}(\Phi\alpha^{h-1}, \Psi\gamma)$.*

3. *Update:* $\alpha^h = arg\ max_\alpha\ \tilde{L}(\Phi\alpha, \Psi\gamma^h)$. $h = h + 1$.

4. *Iterate steps (2) and (3) until convergence, or* $h = \bar{h}$ *steps.*

Next, Lemma 3 and Proposition 4 study the error bound for the objective function $\tilde{L}(\Phi\alpha, \Psi\gamma)$ and the properties of the solution $\tilde{V}_p$. We focus on the approximation issue regarding V, where we expect a similar conclusion of p to follow directly. Suppose for a given policy p, the maximizer for $L(p, V)$ is V_p, for $\bar{L}(p, V)$ is $\bar{V}_p$ and for $\tilde{L}(p, V)$ is $\tilde{V}_p$. Then Lemma 3 provides an error bound of objective function value on V_p and $\tilde{V}_p$.

Lemma 3. *Assume that the error from polynomial approximation* $||\bar{L}(p, \bar{V}_p) - L(p, V_p)|| = O_p(\rho_K)$ *with* $\rho_K \to 0$ *as* $K \to \infty$. *The error bound for the objective function values between* $L(p, V_p)$ *and* $\tilde{L}(p, \tilde{V}_p)$ *equals,*

$$||\tilde{L}(p, \tilde{V}_p) - L(p, V_p)|| = O(\sigma) + O_p(\rho_k) \tag{18}$$

$\sigma \geq 0$. *Or in other words, it is stochastically bounded by* σ *and the polynomial approximating error.*

Proof. By the triangular inequality,

$$||\tilde{L}(p, \tilde{V}_p) - L(p, V_p)|| \leq ||\tilde{L}(p, \tilde{V}_p) - \bar{L}(p, \bar{V}_p)|| + ||\bar{L}(p, \bar{V}_p) - L(p, V_p)|| \tag{19}$$

The second part of (19) equals to $O_p(\rho_K)$ by assumption directly. In the case with finite states, $\rho_K = 0$ if $K \geq |\mathcal{X}|$. To get the first part, write the min in $m(p, V)$ as maximization, and obtain

$$L(p, V) = w'T(p, V) + \beta(1-\beta)^{-1} \min_x H(p, V)(x) = w'T(p, V) - \beta(1-\beta)^{-1} \max_x \{-H(p, V)(x)\}$$

Similar expression applies for $\bar{L}(p, \bar{V}_p)$ and $\tilde{L}(p, \tilde{V}_p)$. Then by definition of $\bar{V}_p$ and inequality (15), $\tilde{L}(p, \tilde{V}_p) \leq \bar{L}(p, \bar{V}_p)$. Furthermore, there is,

$$\begin{aligned}
\tilde{L}(p, \tilde{V}_p) &= w'T(p, \tilde{V}_p) - \beta(1-\beta)^{-1}\left\{\sigma \ln\left(\sum_x \exp\{-\sigma^{-1}H(p, \tilde{V}_p)(x)\}\right)\right\} \\
&\geq w'T(p, \bar{V}_p) - \beta(1-\beta)^{-1}\left\{\sigma \ln\left(\sum_x \exp\{-\sigma^{-1}H(p, \bar{V}_p)(x)\}\right)\right\} \\
&\geq w'T(p, \bar{V}_p) - \beta(1-\beta)^{-1}\left\{\max_x\{-H(p, \bar{V}_p)(x)\} + \sigma \ln|\mathcal{X}|\right\} \\
&= w'T(p, \bar{V}_p) - \beta(1-\beta)^{-1}\left\{\max_x\{-H(p, \bar{V}_p)(x)\}\right\} - \beta(1-\beta)^{-1}\sigma \ln|\mathcal{X}|
\end{aligned} \tag{20}$$

which gives $\tilde{L}(p, \tilde{V}_p) \geq \bar{L}(p, \bar{V}_p) - \beta(1-\beta)^{-1}\sigma \ln |\mathcal{X}|$ and then $||\tilde{L}(p, \tilde{V}_p) - \bar{L}(p, \bar{V}_p)|| = O(\sigma)$. □

Proposition 4. *Here are several properties regarding the smooth approximation –*

1. *Under polynomial approximation $\Phi\alpha$ or $\Psi\gamma$ and m_σ in (14), $\tilde{L}(\cdot)$ is concave in V and concave in p on any set $\mathcal{P} \times \mathcal{V}$ where $T(p, V)$ is concave in p.*

2. *Denote a projector Γ that transforms the value function V^{h-1} to the next iteration through the objective function $\bar{L}(\cdot)$ in (17). A fixed point of $\bar{V}$ then is defined as the solution to $V = \Gamma(V)$. Assume that (i) there is unique fixed point $\bar{V}$ for Γ; (ii) for each $V^{h-1} \in \mathcal{V}$, $\bar{L}(p, V^{h-1})$ is uniquely maximized at $\bar{p}$ and $\bar{p}$ is an interior point of $\mathcal{P}$; (iii) for each $\bar{p}$ that can be obtained from $V^{h-1} \in \mathcal{V}$, $\bar{L}(\bar{p}, V^h)$ is uniquely maximized at $\bar{V}^h$ and $\bar{V}^h$ is an interior point of $\mathcal{V}$; (iv) $T(p, V)$ is concave in p. Then as $\sigma \to 0$, the fixed point solution $\tilde{V}_\sigma$ of Algorithm 2 converges to $\bar{V}$.*

Proof. See Appendix A. □

Proposition 4.1 emphasizes that the concavity of the optimization problems preserves in (16). Proposition 4.2 gives an asymptotic results of the convergence of $\tilde{V}_\sigma$ to the non-smoothed polynomial approximation $\bar{V}$ as $\sigma \to 0$. Note that we abstract away from the error coming from polynomial approximation itself, which is the focus of study in de Farias and Van Roy (2003) and Kristensen et al. (2021).

In practice, however, using both the approximation to V and p in Algorithm 2 may trigger the accumulation of approximating errors in each step and reduce the accuracy. Moreover, lower computational costs and even analytical solutions may be available for a specific problem. The best routine is a customized choice conditional on the problem at hand. For instance, in a dynamic discrete choice with a Logit error term (Aguirregabiria and Mira, 2002), we have an analytical solution of the conditional choice probability given the current iteration of V^h.

Example 1 Continue. *As for the policy p, we have the concavity of p regarding the $T(p, V)$ since $\partial^2[-p(a,x)\ln(p(a,x))]/\partial p^2(a,x) = -1/p(a,x) < 0$. But we do not need to solve (16) for p since we have analytical solution such that,*

$$p_a^h = \frac{\exp\{u(a,x;\theta) + \beta \int_{\mathcal{X}} f(x'|a,x)\tilde{V}^h(x')dx\}}{\sum_{a\in\mathcal{A}} \exp\{u(a,x;\theta) + \beta \int_{\mathcal{X}} f(x'|a,x)\tilde{V}^h(x')dx\}} \tag{21}$$

Analytically solving an exact problem is equivalent to obtaining the policy updates in Algorithm 1, or in Algorithm 2 but with Φ a $|\mathcal{X}|$-dim identity matrix. Thus, all the desired properties of algorithms can still be established. Additionally, solving (16) directly with $\Phi\alpha$ is problematic since $0 < p(a,x) < 1$ needs to be satisfied for all $a \in |\mathcal{A}|$ and $x \in |\mathcal{X}|$ in each iteration for the objective function to be well-defined.

In this standard case, our method shares some common attributes with Feng et al. (2021), which proposes an iterative method over the policy functions through the choice-specific value function v_a. However, our alternating algorithm may lead to a faster convergence rate than theirs.

4 Discussions and Extensions

4.1 The Computational Gains From Polynomial Approximation

As discussed in Section 3.2, by polynomial approximation, the dimension of optimization drops from $|\mathcal{X}|$ to K, offering a computational gain especially when $K \ll |\mathcal{X}|$. In practice, this gain is achieved primarily by calculating the time-consuming matrix multiplication outside the loop when searching for parameters γ.

Example 1 Continue. *For the V-approximation in Example 1, we may re-write (16) as*

$$\begin{aligned}\tilde{L}(p,\Psi\gamma) &= w'T(p,\Psi\gamma) + \beta(1-\beta)^{-1}m_\sigma(H(p,V)) \\ &= w'U_p + (w'\beta F_p\Psi)\gamma + \beta(1-\beta)^{-1}m_\sigma\Big(U_p + [\beta F_p - I_{|\mathcal{X}|}]\Psi\gamma\Big)\end{aligned} \tag{22}$$

from the definition of $T(p,V)$ in (1). $I_{|\mathcal{X}|}$ is the identity matrix of size $|\mathcal{X}|$. For (22), $w'\beta F_p\Psi$, U_p and $[\beta F_p - I_{|\mathcal{X}|}]\Psi$ does not depend on γ and can be calculated outside the optimization over γ, which reduces the dimension of computation to K and greatly improves the computational efficiency.

Example 2 Continue. *The CRRA utility, however, has a large derivative near 0, which may be hard to capture by a low-dimensional polynomial approximation[7]. This problem deteriorates the accuracy when V is approximated. A complicated high-dimensional polynomial helps, yet it would slow the computational speed.*

For the stochastic growth model, the policy function is less restricted since it is no longer bounded

7. Judd (1998), Chapter 6, pages 228-230, also documents this as an example to show that one needs to be careful when the function approximated takes a CRRA shape.

above by 1 as the CCPs. Moreover, we only need $k_{t+1} \geq 0$ *and* $c_t \geq 0$*, which can be incorporated using the shape restrictions in Section 4.2 below.*

As mentioned in Section 3.1, the optimal policy can be solved state-by-state given the value function. In other words, the policy iteration consists of many small maximizations, with the number of problems equal to the number of discretized states. Thus, the parallel computation can be directly adopted. In contrast, our approximation solves a single concave optimization with a larger dimension[8]*. Which method delivers better performance may depend on the number of states and the availability of parallel computing technologies, further illustrated in Section 5.2.*

4.2 Shape Restrictions

The behavior of polynomial approximation and other nonparametric techniques could be boosted through the inclusion of shape restrictions. Shape restrictions like monotonicity, concavity and other constraints derived from economic theories have been widely applied in economics. Shape restrictions help to eliminate anomalous behavior and obtain precise estimates from flexible methods[9]. In this study, to improve the numerical behavior, we may resort to the simple non-negativity constraints of $\Phi\alpha$ in Example 2.

Example 2 Continue. *For instance, in (3), individual's decision is restricted by a non-negativity constraint* $k_{t+1} \geq 0$*. This non-negativity constraint needs to be satisfied in the solution and translates into the first set of shape restrictions in the model.*

Furthermore, the usual CRRA utility is undefined under negative consumption, leading to $k_{t+1} - (1-\delta)k_t - Az_t k_t^\theta \geq 0$ *as a constraint that needs to be satisfied in each step. For any optimization solver, no information is derived when reaching an undefined point, hindering the descending procedure even in unconstrained concave optimization*[10]*.*

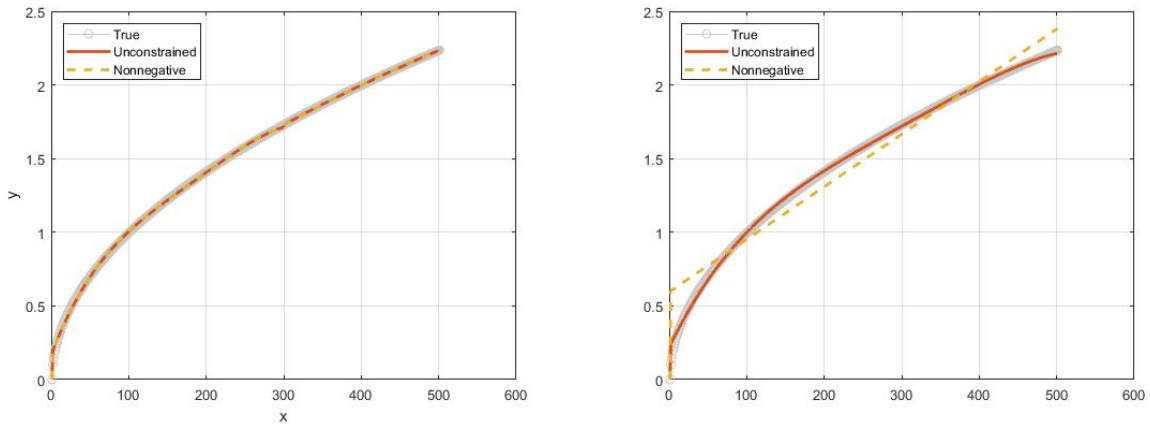
To alleviate this problem, in practice, a possible modification comes from treating the consumption c *as the policy and accounting for the nonnegative consumption by requiring* $\alpha \geq 0$*. This extra con-*

8. This "larger" dimensional is compared with the uni-variate state-by-state optimization. If we choose appropriate techniques for approximation, it would still be a small to moderate problem.

9. For instance, Blundell et al. (2012) uses the nonlinear Slutsky conditions from consumer theory and the shape-restricted nonparametric estimation yields a well-behaved demand function for gasoline in the US. Chetverikov and Wilhelm (2017), on the other hand, investigates the possibility of improving the NPIV estimator's performance by imposing shape restrictions. They demonstrate that the monotonicity constraint possesses a strong regularization effect and consistently enhances the performance of the nonparametric estimator.

10. A possible refinement would have multiple starting values, but it drags down the overall computational efficiency.

straint is usually compatible with step-wise approximations like B-spline, but may not be applicable if the entire interval is approximated by, for instance, Chebyshev polynomials. Figure 1 illustrates a simple approximation of function $y = \sqrt{x}$.



(a) Approximated y, using degree-4 B-spline with knots (0,0.5,1)'

(b) Approximated y, degree-4 Chebyshev polynomials.

Figure 1: Simple illustration on non-negativity restriction.

4.3 Upper Bound for $w'V^*$

For the counterfactual analysis on dynamic programming, one may be interested to know the weighted average welfare or $w'V^*$. We have already constructed a lower bound: upon convergence, Algorithm 2 delivers approximate value and policy functions and also delivers a lower bound on $w'V^*$ through the value of $\tilde{L}(\Phi\alpha^*, \Psi\gamma^*)$ from Proposition 3, where (α^*, γ^*) is the convergence point. One can see this from the proof of Proposition 3 and inequality (15), since $L(p, V) \leq w'V_p \leq w'V^*$ and $\tilde{L}(p, V) \leq L(p, V)$ for $\forall(p, V)$.

With the lower bound obtained, an upper bound on $w'V_p$ can be directly derived from the solution to the approximate problem (23),

$$\min_{V} \quad \hat{L}(p, V) = w'T(p, V) + \beta(1-\beta)^{-1}\left\{\max_{x} H(p, V)\right\} \tag{23}$$

The minimization problem in (23) is almost identical to that of the value function step in Algorithm 2, except that (i) it is a minimization problem; and (ii) the max operator in $L(p, V)$ is replaced with the max operator.

Proposition 5. *The minimization problem in (23) is convex, and the minimum objective function value gives an upper bound for the weighted value $w'V_p$.*

Proof. See Appendix A. □

Proposition 5 shows that the solution to (23) provides an upper bound to the value function associated with a given policy. We can simply fix the policy at its convergence point $\tilde{p}$ in the main maximization procedure, then perform the minimization in (23).

If only V is to be approximated, then we can obtain an upper bound to $w'V^*$ directly. In this case, we replace the fixed policy in (23) with the policy associated with V, so that it changes with V.

$$\min_{V} \quad \hat{L}(p(V), V) = w'T(p(V), V) + \beta(1-\beta)^{-1}\left\{\max_{x} H(p(V), V)\right\} \tag{24}$$

Proposition 6. *The minimization problem in (24) is convex, and the minimum objective function value gives an upper bound for the weighted value $w'V^*$.*

Proof. First we define $\bar{T}(V) \equiv T(p(V), V)$ and $\bar{H}(V) \equiv \bar{T}(V) - V$. Both $\bar{T}(\cdot)$ and $\bar{H}(\cdot)$ are convex, which immediately yields convexity of the minimization problem because the maximum of convex functions is convex. Additionally, $\bar{H}(\cdot)$ is inverse-antitone, so $\bar{H}(V) \leq 0$ implies $V^* \leq \bar{T}(V) \leq V$. From here, we proceed similarly to the proof of Proposition 5, obtaining $\hat{L}(p(V), V) \geq w'V^*$ and $\hat{L}(p(V^*), V^*) = w'V^*$. □

5 Simulations

In this section, we demonstrate the performance of our algorithm by a series of simulations on both Examples 1 and 2. First, we carry on the simulation of the bus-engine replacement problem. We employ user-supplied analytical gradients for the Quasi-Newton method in all the simulations in our Algorithm 2.

5.1 Example 1 Continue

5.1.1 Baseline Model

For a bus-engine replacement problem, an agent has two discrete actions – replacing the engine or not, or $a \in \{1, 0\}$. If no replacement, or $a = 0$, the univariate state variable (the accumulated

mileage) x involves following: with probability 0.0937, $x_{t+1} = x_t$; and with probability 0.9063, $x_{t+1} = x_t + \delta_t$, and δ_t follows a $Beta(2,5)$ distribution re-scaled to interval $[0, 15]$. With $a = 1$, transition of x_t still holds but with $x_t = 0$. We then discretize the mileage into 251 discrete states, $\{0, 1, 2, \cdots, 250\}$. $\beta = 0.975$. In the simulations below, without loss of generality, we assume the transition of x is known. The stationary distribution of x_i is denoted as π_i or $\pi(x_i)$ and is defined by simultaneous equations,

$$\pi(x_j) = \int_{\mathcal{X}} \left[\sum_{a \in \mathcal{A}} p(a, x_i) f(x_j | a, x_i) \right] \pi(x_i) d\, x_i$$

The flow utility encountered in each period depends on the action a. $u_{a=0,t} = u_{0,t} = \theta_0 + \theta_1 \times (0.001)x_t + \varepsilon_0$, $u_{a=1,t} = u_{1,t} = \varepsilon_1$, with ε a Logit error. The true $\theta^* = (11.7257, -2.4569)'$. Here we approximate both p and V through polynomial approximation and smoothing techniques[11].

Figure 2 demonstrates the overall performance when the sample size $N = 1000$ and we start the iteration with an initial $\alpha^0 = \left[\sum_{i=1}^N \Phi_i \Phi_i'\right]^{-1} \sum_{i=1}^N \Phi_i a_i$[12]. $\sigma = 0.001$ and $w(x_i) = \pi(x_i)$. Figure 2(a) and (b) illustrate that in this basic case, solving (16) gives us a good approximation to the true V and p. Besides, the objective function value $\tilde{L}(\Phi\alpha, \Psi\gamma)$ indeed monotonically increases across alternating iterations, as shown in Figure 2(c).

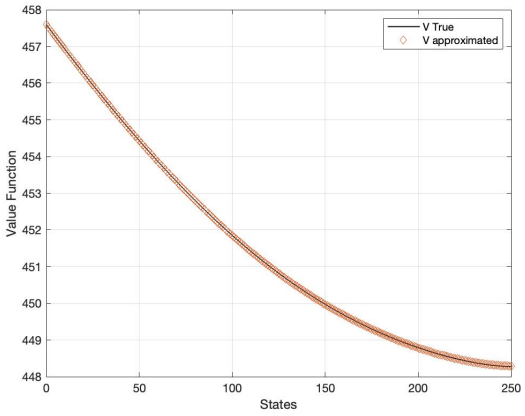
Moreover, Figure 2(d) depicts the lower and upper bounds from equations (16) and (23) by having the weights $w_1 = (1, 0, \cdots, 0)'$, $w_2 = (0, 1, \cdots, 0)'$, $\cdots$, $w_{251} = (0, 0, \cdots, 1)'$, successively. Or in other words, we put all the weights on each state one after another while ignoring others. Our method is successful in providing an lower and upper bound of $w'V^*$ for each state.

5.1.2 Choice of w and σ

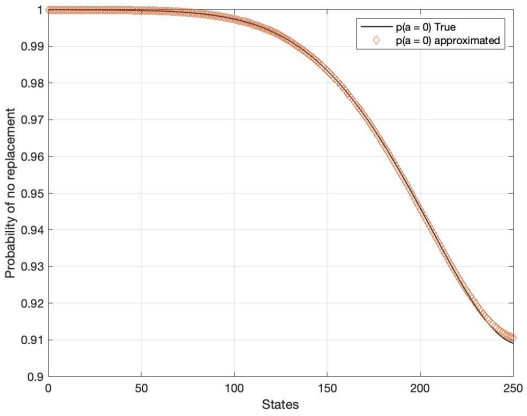
As discussed in previous sections, w is the state-relevance weight. When there is full flexibility for V and p, all choices of w are equivalent for the solutions. However, when approximation binds, different choices of w will yield different results. With limited dimensions of approximation, as indicated by de Farias (2002), we may not anticipate a uniformly good fit throughout all the states.

11. The linear function in B-spline may not generate a value strictly between 0 and 1, contradicting the assumption of ε. We have to trim the estimated probability to accommodate this, especially in the first few iterations. We use a flexible construction of degree-4 B-spline, with basic nodes $(0, 0.5, 1)'$. When implementing, those basic nodes are repeated three times, or we use $(0, 0, 0, 0.5, 0.5, 0.5, 1, 1, 1)'$.

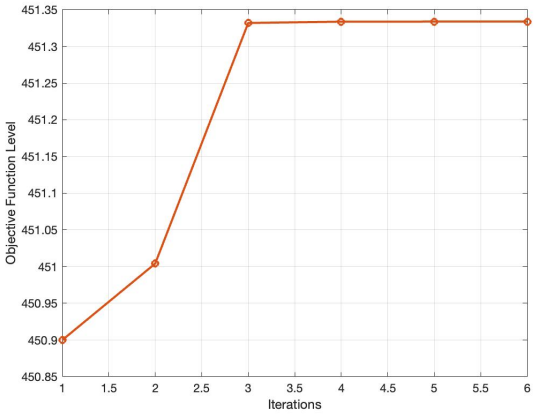
12. The finite sample only comes in when determining the starting value α^0. Usually, running OLS on the binary choice variable a_i gives a poor initial prediction of choice probabilities.



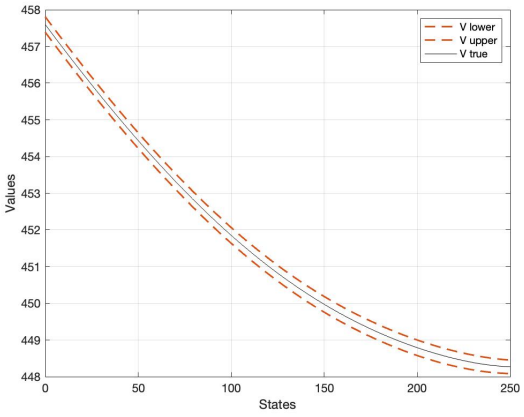
(a) Approximated V



(b) Linearly approximated policy function



(c) $\tilde{L}(\alpha, \gamma)$ at each iteration, with initial value 355.8597.



(d) Upper and Lower bounds for each state

Figure 2: The approximation of value function and choice probabilities in the baseline model of Section 5.1.

Therefore, we face the challenges of balancing the approximating accuracy across different state areas.

Indeed, there would be some instances where the problem at hand can guide the choice of w. An example may come from cases where people only care about specific states. However, if we want a good approximation in general across states, two sets of weights naturally arise. The first could be the original stationary distribution $\pi(x)$, which emphasizes the frequently visited states. The second, on the contrary, would assign $1/|\mathcal{X}|$ to all the states. These arise from the heuristic discussion in de Farias (2002). The approximated method may yield poor results for the states with lower V, unless those states are emphasized by w. Simultaneously, Theorem 1 in de Farias and Van Roy (2003) implies that the frequently-visited states must be highlighted to have the resulting policy function close to the optimal. Appendix B. provides simulation experiments regarding the choice of w in different cases, serving as a heuristic indication of what is happening. We would stick to the two sets of w mentioned above in the following simulations.

Moreover, σ in equation (14) governs the accuracy of the smooth approximation of the non-differentiable max operator. A larger σ leads to inadequate approximation towards the min function, while a σ that is too small may render too less smoothness, hindering the performance of derivative-based algorithms. Appendix B. also contains the trade-off inside the choice of σ, indicating that for stability, we would expect σ to be around 0.001 or 0.0001.

5.1.3 Comparisons of Different Methods for Example 1

In this section, we perform further simulations and a comprehensive comparison across different methods. Table 1 summarizes the settings[13]. The broad comparison covers the fixed point iteration (Rust, 1987), Howard's policy iteration (Aguirregabiria and Mira, 2002), approximation through linear programming (de Farias and Van Roy, 2003), and our Algorithm 2 with either the CCPs approximated by linear polynomials, or by analytical solution in (21).

Each simulation is constituted by $R = 100$ repetitions. The weights for approximation method equals the stationary distribution of states, or $w(x_i) = \pi(x_i)$. The starting value of γ for our method is generated by $V^0(x_i) = (1-\beta)^{-1} \min_{x_j \in \mathcal{X}} \{\ln \sum_{a \in \mathcal{A}} \exp(u(a, x_j))\}, \forall x_i \in \mathcal{X}$, and $\gamma^0 = (\Psi'\Psi)^{-1}\Psi' V^0$. V^0 is also the starting value for fixed point iteration (i) in Table 1. Ψ are polynomial

13. Programs are implemented in a laptop with Apple M1 Max, ten cores, Ram 64GB, at Matlab R2021b. For simulations of Example 1, we do not use parallel computation even with multiple cores.

basis. The sample size N is fixed at 2000. Stopping criterion is $||p^{h+1} - p^h||_\infty <= 0.000001$ for all methods.

Table 2 shows the simulation results when the discount factor $\beta = 0.975$. The accuracy of the approximation, or the error in Table 2 regarding the true values (V^*, p^*) is evaluated through $\max_{r \in R} \sum_{x \in |\mathcal{X}|} \pi_i \times |V - V^*|$ and $\max_{r \in R} \sum_{x \in |\mathcal{X}|} \pi_i \times |p - p^*|$. Our approximation method outperforms all other methods in the simulation, especially when the state expands and the analytical CCPs are utilized. For instance, in Model 4 of Table 1, our method (v) takes around 1/3 of the average elapsed time of (iii) and 1/2 of (i), while still keeping a low approximation error. In contrast, linear policy approximation or (vi) performs poorly from the start, as expected from the reasons discussed in Section 4.1.

Bray (2019) pinpoints that with $\beta \to 1$, the speed of convergence for V iteration would decrease, which is partially demonstrated in Table 3 using both of the approximations. While in Table 3, all the performances of (i) (iii) and (v) exacerbate, our approximation still cut down around 3/4 to 1/2 of the elapsed time in Model 4, demonstrating a large computational gain.

5.1.4 High Dimensional State Space

Next, we evaluate the performance of our approximation in a high-dimensional case. Now the state variable $\boldsymbol{x}_t = (x_{t,1}, x_{t,2})'$ where $x_{t,1} \in \{0, 1, 2, \cdots, 249\}$, and $x_{t,2} \in \{0, 1, 2, \cdots, 249\}$. Total number of states accumulates to $250^2 = 62500$. We stick to Kristensen et al. (2021) where the flow utility u_i is additive and transitions are multiplied across $\boldsymbol{x}_t$[14]. Specifically, $u_{0,t} = \theta_0 + \theta_1 \times (0.001) x_{t,1} + \theta_2 \times (0.001) x_{t,2} + \varepsilon_0$ and $u_{1,t} = \varepsilon_1$ with $\theta^* = (11.45, -2.5, -1)'$.

For an iterative method, the performance depends on the starting values of p. We initialize $p^0(a, x)$ using an MLE-Logit estimated value. We regard the MLE-Logit-based $p^0(a, x)$ as a poor starting value since $N = 5000$ is small compared to 62500 states. The error in p^0 is on average. Stopping criterion is $||p^{h+1} - p^h||_\infty <= 0.00001$ for all methods. For polynomial approximation, we use tensor-product based degree-2 B-spline[15]. The number of repetitions $R = 10$.

14. If no engine replacement or $a = 0$, the transition of $x_{t,1}$ follows: with probability 0.0937, $x_{t+1,1} = x_{t,1}$; and with probability 0.9063, $x_{t+1,1} = x_{t,1} + \delta_t$, and δ_t follows a $Beta(2, 2)$ distribution re-scaled to interval $[0, 10]$. The transition of $x_{t,2}$ follows: with probability 0.0010, $x_{t+1,2} = x_{t,2}$; and with probability 0.999, $x_{t+1,2} = x_{t,2} + \delta_{t,2}$, and $\delta_{t,2}$ follows a $Beta(2, 5)$ distribution re-scaled to interval $[0, 15]$. If $a = 1$, then the transition stays the same with $x_t = 0$.

15. B-spline is implemented with knots $(0, 0.5, 1)'$, and there are 36 bases in total. We no longer use degree-4 polynomials mainly because some of the linear programmings in (iii) experience a tough time converging, which could bias performance evaluation.

Name	Explanation
(i) Fixed Point	Following Kristensen et al. (2021), the fixed point iteration on the integrated value function. We solve for V that satisfies $$V(x) = \ln\left(\sum_{a\in\mathcal{A}} \exp\left\{u(a,x) + \beta \int_{x'} f(x'|a,x)V(x')dx'\right\}\right)$$ via fixed point iteration, which is different from Rust (1987).
(ii) Howard's	Following Aguirregabiria and Mira (2002). Given the CCP $p^h(a,x)$, i) solve $V^{h+1}(x) = A_p\theta + b_p$ where $A_p = (I_{\lvert\mathcal{X}\rvert} - \beta F^U)^{-1}[\sum_{a\in\{0,1\}} p^h(a,x)H_a]$; $b_p = (I_{\lvert\mathcal{X}\rvert} - \beta F^U)^{-1}[\sum_{a\in\{0,1\}} p^h(a,x) * e^h(a,p)]$; $F^U = \sum_{a\in\{0,1\}} p^h(a,x)F_a$; and H_a is the stack of $h(a,x)$ with $u(a,x;\theta) = h(a,x)'\theta$. ii) obtain the new $p^{h+1}(a,x)$ from $V^{h+1}(x)$ using (21) and return to i) until convergence.
(iii) LP Approximation	$\tilde{V} = \Psi\gamma$. Given $\tilde{V}$, p updated from Logit formula as in (21).
(vi) Algorithm 2 – $\tilde{p} = \Phi\alpha$	$\tilde{V} = \Psi\gamma$, $\tilde{p} = \Phi\alpha$. The basis Ψ is degree-4 B-spline, and Φ is degree-2 B-spline.[1]
(v) Algorithm 2 – Analytical p	$\tilde{V} = \Psi\gamma$. Given $\tilde{V}$, p updated from Logit formula as in (21).

Models	Specification
Model 1	For $\beta = 0.975$ or $\beta = 0.990$, $\theta^* = [11.7257, -2.4569]'$. States $= \{0, 1, 2, \cdots, 250\}$.
Model 2	For $\beta = 0.975, \Rightarrow \theta^* = [17, -2]'$. For $\beta = 0.990 \Rightarrow \theta^* = [17, -1.5]'$. States $= \{0, 1, 2, \cdots, 499\}$.
Model 3	For $\beta = 0.975, \Rightarrow \theta^* = [20, -1]'$. For $\beta = 0.990, \Rightarrow \theta^* = [20, -0.7]'$. States $= \{0, 0.25, 0.5, 0.75, \cdots, 799\}$.
Model 4	For $\beta = 0.975, \Rightarrow \theta^* = [25, -0.8]'$. For $\beta = 0.990, \Rightarrow \theta^* = [25, -0.5]'$. States $= \{0, 0.1, 0.2, 0.3, \cdots, 999\}$.

[1] Here we use a B-spline with knots $(0, 0.5, 1)'$ as Ψ and Φ. The V for Example 1 takes a regular shape and can be approximated easily.

Table 1: Methods entering into the comparison for Example 1.

			Model 1	Model 2	Model 3	Model 4
(i) Fixed Point	Time	Average	0.1368	0.2727	2.4858	10.5319
		Min	0.1295	0.2648	2.3472	10.3986
		Max	0.1749	0.3031	2.6857	11.4077
(ii) Howard's	Time	Average	0.0236	0.0443	2.5077	50.7084
		Min	0.0083	0.0222	1.3011	29.3116
		Max	0.0956	0.1353	3.7583	83.6843
(iii) LP Approximation	Time	Average	0.0338	0.0715	1.6296	15.8182
		Min	0.0266	0.0360	0.9552	7.7739
		Max	0.0442	0.1287	2.9934	26.6545
	Error	in V	0.0044	0.0026	0.0020	0.0032
		in p	0.0001	0.0000	0.0001	0.0001
(vi) Algorithm 2 – $\tilde{p} = \Phi\alpha$	Time	Average	20.8008	-	-	-
		Min	1.7769	-	-	-
		Max	112.3378	-	-	-
	Error	in V	7.6418	-	-	-
		in p	1.1636	-	-	-
(v) Algorithm 2 – Analytical p	Time	Average	0.0270	0.0714	0.4535	4.4568
		Min	0.0195	0.0471	0.2876	2.6713
		Max	0.0538	0.1213	0.7649	7.8929
	Error	in V	0.0028	0.0036	0.0047	0.0050
		in p	0.0001	0.0001	0.0001	0.0001

Table 2: Elapsed time in seconds and approximation error. $\beta = 0.975$.

			Model 1	Model 2	Model 3	Model 4
(i) Fixed Point	Time	Average	0.3627	0.6697	5.9294	24.6214
		Min	0.3277	0.6442	5.2974	24.3667
		Max	0.4061	0.8789	7.1130	26.2521
(ii) Howard's	Time	Average	0.0331	0.0436	2.6276	57.9553
		Min	0.0084	0.0218	1.3140	29.5132
		Max	0.1391	0.1474	4.1821	83.1976
(iii) LP Approximation	Time	Average	0.0390	0.0765	1.7722	46.5776
		Min	0.0264	0.0440	0.9071	8.6889
		Max	0.0571	0.1431	3.5683	151.4353
	Error	in V	0.0036	0.0023	0.0029	0.0073
		in p	0.0001	0.0000	0.0000	0.0002
(vi) Algorithm 2 – $\tilde{p} = \Phi\alpha$	Time	Average	10.8485	-	-	-
		Min	0.9352	-	-	-
		Max	104.3327	-	-	-
	Error	in V	24.2000	-	-	-
		in p	0.9553	-	-	-
(v) Algorithm 2 – Analytical p	Time	Average	0.0333	0.0853	0.5645	10.1143
		Min	0.0265	0.0475	0.2582	2.1327
		Max	0.0663	0.4522	3.5136	63.6861
	Error	in V	0.0040	0.0053	0.0079	0.0096
		in p	0.0001	0.0002	0.0002	0.0002

Table 3: Elapsed time in seconds and approximation error. β =0.990.

			$w_i = 1/\lvert\mathcal{X}\rvert$	$w_i = \pi_i$
(i) Fixed Point	Time	Average	6886.1	
(ii) Howard's	Time	Average	12547.99	
		Min	12035.9	
		Max	14178.8	
(iii) LP Approximation[1]	Time	Average	2399.92	2491.68
		Min	1441	1833.1
		Max	3696.5	2876.2
	Error	in V	0.0103	0.0083
		in p	0.0004	0.0004
(v) Algorithm 2 – Analytical p	Time	Average	1491.7	1832.53
		Min	931.9	1408.2
		Max	1957.8	2342.4
	Error	in V	0.0105	0.0075
		in p	0.0004	0.0004

[1] For this specific example, we have transition matrices $F_{a=0}$ and $F_{a=1}$ which can both be stored as sparse matrix in Matlab. Computational time would further reduced when those sparsity structure is utilized. For instance, with $w_i = 1/|\mathcal{X}|$, the averaged elapsed time for (v) is 266.7 and (iii) is 825.3.

Table 4: Elapsed time in seconds and approximation error for $|\mathcal{X}| = 62500$. $\beta = 0.975$.

Table 4 contains the simulated results. Exact Howard's step suffers from a costly computation when inverting the system of V. At the same time, due to the analytical structure of the integrated value function, Fixed Point iteration itself is not that problematic. Comparing (iii) and (v), as expected, we show that, by replacing the constrained linear programming with unconstrained nonlinear optimization in (16), our Algorithm 2 provides a steady improvement in elapsed times. Although the starting $p^0(a, x)$ is not close to the true p, good approximations are obtained in both (iii) and (v).

5.2 Example 2 Continue

This part contains the simulations of the stochastic growth model in Example 2. We generate different states by varying the grids for k_t and z_t. Table 5 summarizes the models and methods we use here. We test the cases of Algorithm 2 with either V or p approximated, while linear programming only applies to V-approximation. Following the regular calibration, we assume we

know the key parameters before tackling the model. The true parameters are: $(A^*, \nu^*, \delta^*, \lambda^*, \beta^*)' = (0.5, 0.5, 0.01, 0.5, 0.975)'$.

For each case, the starting value of V^0 is generated by $V^0(k, z) = \bar{k} \times [\alpha/(1 - \alpha\beta)] \times (\sqrt{k} + \sqrt{z})$[16]. $\bar{k}$ is the average of the grid points of k, and the starting value p^0 is the optimal policy hinged on V^0. We have equal weights across states[17]. Table 6 displays the simulation results. Unlike the discrete case in Section 5.1.3, here (V^0, p^0) does not change with a random-generated sample – instead, they are fixed across different repetitions. Hence, there are small variations in elapsed time, rendering the inclusion of time-spread unnecessary. Thus, we carry on $R = 10$ total repetitions. When calculating exact p, we use parallel computing across ten cores to speed up.

Table 6 illustrates that our approximation method also provides computational gains in the stochastic growth model. The errors of approximation are calculated via $\max_{r \in R}(1/|\mathcal{X}|) \times \sum_{x \in |\mathcal{X}|} |V - V^*|$ and $\max_{r \in R}(1/|\mathcal{X}|) \times \sum_{x \in |\mathcal{X}|} |p - p^*|$. Having an approximating $\tilde{V}$ in stochastic growth reduces the elapsed time. The computational gain from (ii) to (iii) mainly comes from eliminating all the constraints – for example, in Model 3, given p^h, the elapsed time for solving $\tilde{V}^{h+1}$ is around 140 seconds, around 35% of the time for linear programming[18]. However, as discussed in Section 4.2, the infinite derivative of V near 0 would be hard to capture by a finite-degree polynomial, which is reflected by the non-negligible approximation error of V in both (ii) and (iii) for Model 3, for instance. Hence, we did not refer to V approximation for Model 4. For method (vi), even when the component-wise calculation of p is paralleled, our unconstrained concave optimization still outperforms Howard's policy iteration while still keeping a precise approximation to both V and p.

Besides the computational gain in the elapsed time, our method would be robust to linear programming due to its concavity nature. The advantage of concave or convex optimization leads to a good approximate solution even when the exact optimizer γ may be hard to obtain from some p. To put it differently, stopping after a fixed $\bar{h}$-step of (16) would not jeopardize our Algorithm 2. However, the simplex method of linear programming does not guarantee an approximation close to the true solution after $\bar{h}$ steps. This robustness helps the algorithm to proceed when the current p

16. The concavity of our method renders a good starting value unnecessary, though the elapsed time would depend on where we begin.

17. In this case, the stationary distribution $\pi(x_i)$ would only have positive values for a few states, rendering inadequate emphasis for the overall approximation.

18. The overall elapsed time does not include such an improvement because (iii) takes more iterations to converge in this case.

makes it hard to solve for $\tilde{V}$[19].

In p approximation, we obtain a continuous function over a discrete state space $x_t = (k_t, z_t)$. $\tilde{p} = \Phi\alpha$ needs not exactly falls into a grid point. To accommodate the discretized states, we interpolate the value following (5.2) below. For a current state k_t[20], a continuously-approximated policy $\tilde{p}$ gives $k_{t+1} = Ak_t^{\nu} + (1-\delta)k_t - c_t$. Then $V(k_{t+1})$ on the discrete grid is interpolated as,

$$V(k_{t+1}) = \begin{cases} \dfrac{\overline{x}_{k_{t+1}} - k_{t+1}}{\overline{x}_{k_{t+1}} - \underline{x}_{k_{t+1}}} \times V(\underline{x}_{k_{t+1}}) + \dfrac{k_{t+1} - \underline{x}_{k_{t+1}}}{\overline{x}_{k_{t+1}} - \underline{x}_{k_{t+1}}} \times V(\overline{x}_{k_{t+1}}), & \text{if } \overline{x}_{k_{t+1}} \neq \underline{x}_{k_{t+1}} \\ V(\underline{x}_{k_{t+1}}), & \text{if } \overline{x}_{k_{t+1}} = \underline{x}_{k_{t+1}} \end{cases}$$

where $\overline{x}_{k_{t+1}}$ is the smallest point that is larger than k_{t+1} where V are evaluated and $\underline{x}_{k_{t+1}}$ is the largest point that is smaller than k_{t+1}.

6 Conclusion

This study develops an alternating unconstrained concave optimization approach for dynamic programming approximation to alleviate the computational burden. We start from the linear programming in de Farias and Van Roy (2003) and demonstrate that the constraints in linear programming can be eliminated after a re-construction of the objective function. We show that our alternating steps are hill-climbing under some proper restrictions with guaranteed convergence to a fixed-point solution. Moreover, our approximation can be combined with the polynomial approximation and smoothing log-sum-exp techniques (Kristensen et al., 2021) to provide further computational gains. Our method is compatible with a vast range of dynamic programming, which we illustrate with two common examples – discrete-choice bus engine replacement problem and stochastic growth model. In both cases, Monte Carlo simulations demonstrate our method's superior performance in computational efficiency.

Several important issues are remained open and waiting for further investigations. For instance, we can extend our algorithm into the case with continuous state space $\mathcal{X}$. For instance, we define

19. In simulations, we encounter the cases with high-dimensional states when (iii) in Table 1 and (ii) in Table 5 cannot move on since intermediate linear programming does not converge, which are not included in Table 6. This convergence problem would be largely alleviated when Algorithm 2 is in place.

20. Remember that z_t evolves independently from p_t or k_t.

Name	Explanation
(i) Howard's[1]	Following the textbook example in Ljungqvist and Sargent (2004). Given the current $p^h = c^h(k,z)$, V^{h+1} is calculated by directly inverting the system such that, $V^{h+1} = (I_{\|\mathcal{X}\|} - \beta F_p)^{-1} U_p$. State k are fully discreticized. p^{h+1} is then solved by maximizing over those discrete grid points component-wise. Iterating over p^h and V^h until convergence.
(ii) LP Approximation	$\tilde{V} = \Psi\gamma$. We use degree-4 multivariate B-spline as basis Ψ?[2]
(iii) Algorithm 2 with $\tilde{V}$	$\tilde{V} = \Psi\gamma$. With the given $\tilde{V}$, p is optimized component-wise. Same polynomial basis Ψ as (ii).
(iv) Algorithm 2 with $\tilde{p}$	$\tilde{p} = \Phi\alpha$. The basis Φ is degree-2 B-spline[3]. And with $\tilde{p}$, calculate the implied V exactly as $V = (I_{\|\mathcal{X}\|} - \beta F_p)^{-1} U_p$. Shape restrictions discussed in Section 4.2 are imposed.

Models	Specification
Model 1	$z_t = \{0.5, 1\}$. $k_t = \{0, 0.02, 0.04, \cdots, 15\}$. Number of states = 1502.
Model 2	$z_t = \{0.5, 0.6, \cdots, 1\}$. $k_t = \{0, 0.01, 0.02, \cdots, 15\}$. Number of states = 9006.
Model 3	$z_t = \{0.5, 0.55, 0.6, \cdots, 1.5\}$. $k_t = \{0, 0.01, 0.02, \cdots, 15\}$. Number of states = 31521.
Model 4	$z_t = \{0.5, 0.53, 0.56, \cdots, 1.5\}$. $k_t = \{0, 0.01, 0.02, \cdots, 18\}$. We have high-dimensional case with the number of states = 61234.

[1] We do not test or record the elapsed time from fixed point iteration for Example 2 because unlike (i) in Table 1, we do not have that analytical integrated value function. Instead, for each V^h, we need to calculate p numerically, which is extremely time-consuming.

[2] Here we use a B-spline with knots $(0, 0.01, 0.1, 0.25, 0.5, 0.75, 0.9, 0.95, 1)'$ as Ψ. In contrast with the discrete choice, the irregular infinite derivative of CRRA induce higher challenge in V approximation, as discussed in Section 4.2.

[3] Here we use a B-spline with knots Φ. Policy p, on the other hand, can be approximated well with lower-dimensional polynomials.

Table 5: Methods entering into the comparison for stochastic growth model.

			Model 1	Model 2	Model 3	Model 4
(i) Howard's	Time	Average	4.48	142.5	2161.4	15176.0
(ii) LP Approximation	Time	Average	2.13	37.74	1378.1	-
	Error	in V	0.0102	0.0147	5.3989	-
		in P	0.0004	0.0008	0.0096	-
(iii) Algorithm 2 with $\tilde{V}$	Time	Average	2.41	63.19	1060.6	-
	Error	in V	0.0044	0.1197	5.2417	-
		in P	0.0073	0.0110	0.0234	-
(vi) Algorithm 2 with $\tilde{p}$	Time	Average	9.48	91.36	941.7	5407.5
	Error	in V	0.0132	0.0121	0.0018	0.0008
		in P	0.0073	0.0110	0.0065	0.0046

Table 6: Elapsed time and approximation error for the stochastic growth model in Table 5.

$\tilde{m}(p, V) = \inf_x H(p, V)(x)$, and the objective function becomes

$$\tilde{L}(\alpha, \gamma) = \int w(x) T(\Phi\alpha, \Psi\gamma)(x) dx + \beta(1-\beta)^{-1} \tilde{m}(\Phi\alpha, \Psi\gamma) \tag{25}$$

where $\int w(x)dx = 1$ and $w(x) \geq 0$ almost everywhere.

(25) is infeasible with only finite computer memory available. For a feasible version of the problem, Kristensen et al. (2021) considers taking a sample of states, $\{x_i\}_{i=1}^N$, from the conditional or unconditional distribution of x. Define $m_N(p, V) \equiv \min_{x_i} H(p, V)(x_i)$. Then, Algorithm 2 can be applied to

$$\tilde{L}_N(\alpha, \gamma) \equiv \frac{1}{N} \sum_{i=1}^{N} T(\Phi\alpha, \Psi\gamma)(x_i) + \beta(1-\beta)^{-1} m_N(\Phi\alpha, \Psi\gamma). \tag{26}$$

Other extensions include endogenizing the choice of basis functions. In either of our two examples, the choice of basis functions or the knots in the B-spline is motivated mainly by the prior knowledge of V's rough shapes. When such information is unavailable, the choice of basis may need further scrutinizing. Other directions may be to find a theoretical-justifiable way to properly choose the state-relevance weights w for a better overall approximation accuracy or incorporate our approximation into the estimation of structural parameters θ[21] or unobserved heterogeneity in Arcidiacono

21. The estimation of the structural parameters is currently beyond the scope of this paper. For dynamic discrete choice, while our approximation is ready to replace the fixed point iteration on NFXP (Rust, 1987), it is hard to incorporate into the computationally-efficient NPL estimator in Aguirregabiria and Mira (2002). A potential reason is that in an NPL iteration, one only needs to invert the system once for different candidates of structural parameters,

and Miller (2011).

significantly reducing the computational gain. On the contrary, one needs to calculate an unconstrained optimization for every candidate for approximation.

References

Aguirregabiria, V., and A. Magesan. 2013. "Euler Equations for the Estimation of Dynamic Discrete Choice Structural Models." *Structural Econometric Models, Advances in Econometrics* 31:3–44.

Aguirregabiria, V., and P. Mira. 2002. "Swapping the Nested Fixed Point Algorithm: A Class of Estimators for Discrete Markov Decision Models." *Econometrica* 70 (4): 1519–1543.

Arcidiacono, P., and R. Miller. 2011. "Conditional Choice Probability Estimation of Dynamic Discrete Choice Models With Unobserved Heterogeneity." *Econometrica* 79 (6): 1823–1867.

Blundell, R., J. Horowitz, and M. Parey. 2012. "Measuring the Price Responsiveness of Gasoline Demand: Economic Shape Restrictions and Nonparametric Demand Estimation." *Quantitative Economics* 3 (1): 29–51.

Boutilier, C., R. Dearden, and M. Goldszmidt. 2000. "Stochastic Dynamic Programming with Factored Representations." *Artificial Intelligence* 121 (1-2): 49–107.

Bray, R. 2019. "Strong convergence and dynamic economic models." *Quantitative Economics* 10 (1): 43–65.

———. 2022. "A Comment on "Using Randomization to Break the Curse of Dimensionality"." *Econometrica* 90 (4): 1915–1929.

Brock, W., and L. Mirman. 1972. "Optimal Economic Growth and Uncertainty: The Discounted Case." *Journal of Economic Theory* 4 (3): 479–513.

Brumm, J., and S. Scheidegger. 2017. "Using Adaptive Sparse Grids to Solve High-Dimensional Dynamic Models." *Econometrica* 85 (5): 1575–1612.

Cai, Y., and K. Judd. 2014. "Chapter 8 - Advances in Numerical Dynamic Programming and New Applications." *Handbook of Computational Economics* 3:479–516.

Cai, Y., K. Judd, and R. Xu. 2020. "Numerical Solution of Dynamic Portfolio Optimization with Transaction Costs." Working Paper, https://arxiv.org/pdf/2003.01809.pdf, March.

Cai, Y., and T. Lontzek. 2019. "The Social Cost of Carbon with Economic and Climate Risks." *Journal of Political Economy* 127 (6).

Chetverikov, D., and D. Wilhelm. 2017. "Nonparametric Instrumental Variable Estimation Under Monotonicity." *Econometrica* 85 (4): 1303–1320.

de Farias, D., and B. Van Roy. 2003. "The Linear Programming Approach to Approximate Dynamic Programming." *Operations Research* 51 (6): 850–865.

de Farias, D. P. 2002. "The Linear Programming Approach to Approximate Dynamic Programming: Theory and Application." Dissertation, http://web.mit.edu/~pucci/www/daniela_thesis.pdf, June.

Fan, Y., and C. Yang. 2022. "Estimating Discrete Games with Many Firms and Many Decisions: An Application to Merger and Product Variety." Working Paper, http://www-personal.umich.edu/~yingfan/Fan_Yang_discrete_games.pdf, October.

Farias, V., D. Saure, and G. Weintraub. 2012. "An Approximate Dynamic Programming Approach to Solving Dynamic Oligopoly Models." *RAND Journal of Economics* 43 (2): 253–282.

Feng, Y., E. Khmelnitskaya, and D. Nekipelov. 2021. "Global Concavity and Optimization in a Class of Dynamic Discrete Choice Models." Working Paper, http://proceedings.mlr.press/v119/feng20b/feng20b-supp.pdf.

Gao, B., and L. Pavel. 2018. "On the Properties of the Softmax Function with Application in Game Theory and Reinforcement Learning." Working Paper, https://arxiv.org/pdf/1704.00805.pdf, August.

Hotz, V., and R. Miller. 1993. "Conditional Choice Probabilities and the Estimation of Dynamic Models." *The Review of Economic Studies* 60 (3): 497–529.

Jouffe, L. 1998. "Fuzzy inference system learning by reinforcement methods." *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 28 (3): 338–355.

Judd, K. 1998. *Numerical Methods in Economics.* 1st ed. Vol. 1. 0262100711. The MIT Press.

Kristensen, D., P. Mogensen, J. Moon, and B. Schjerning. 2021. "Solving Dynamic Discrete Choice Models using Smoothing and Sieve Methods." *Journal of Econometrics* 223 (2): 328–360.

Ljungqvist, L., and T. Sargent. 2004. *Recursive Macroeconomic Theory, 2nd Edition.* Vol. 1. MIT Press Books 026212274x. The MIT Press.

Ma, Q., and J. Stachurski. 2021. "Dynamic Programming Deconstructed: Transformations of the Bellman Equation and Computational Efficiency." *Operations Research* 69 (5): 1349–1650.

Moré, J. 1971. "Global Convergence of Newton-Gauss-Seidel Methods." *SIAM Journal on Numerical Analysis* 8 (2): 325–336.

Newey, W., and D. McFadden. 1994. "Chapter 36 Large sample estimation and hypothesis testing." *Handbook of Econometrics* 4:2111–2245.

Norets, A. 2012. "Estimation of Dynamic Discrete Choice Models Using Artificial Neural Network Approximations." *Econometric Reviews* 31 (1): 84–106.

Polak, E., J. Royset, and R. Womersley. 2003. "Algorithms with Adaptive Smoothing for Finite Minimax Problems." *Journal of Optimization Theory and Applications* 119:459–484.

Powell, W. 2009. "What You Should Know About Approximate Dynamic Programming." *Naval Research Logistics* 56 (3): 239–249.

Rust, J. 1987. "Optimal Replacement of GMC Bus Engines: An Empirical Model of Harold Zurcher." *Econometrica* 55 (5): 999–1033.

———. 1997. "Using randomization to break the curse of dimensionality." *Econometrica* 65 (3): 487–516.

Schütz, H., and R. Kolisch. 2012. "Approximate dynamic programming for capacity allocation in the service industry." *European Journal of Operational Research* 218 (1): 239–250.

Schweitzer, P., and A. Seidmann. 1985. "Generalized polynomial approximations in Markovian decision processes." *Journal of Mathematical Analysis and Applications* 110 (2): 568–582.

Simão, H., J. Day, A. George, T. Gifford, J. Nienow, and W. Powell. 2009. "An Approximate Dynamic Programming Algorithm for Large-Scale Fleet Management: A Case Application." *Transportation Science* 43 (2): 178–197.

Appendix A. Proofs Omitted in the Main Text

Proof of Proposition 1

Proof. Proposition 1.1 is a direct result from Assumption 1 under the definition of $H(p, V)$. Proposition 1.2 and 1.3 are directly from the definition of $H(p, V)$ and $T(p, V)$ in (1). The derivative of $H(p, V)$ to V can be obtained directly from (1), which constitutes Proposition 1.5.

For Proposition 1.4, to show that $H(p, V)$ is inverse-antitone, we refer to the lemma below, which was originally in Moré (1971).

Lemma 4. *(Moré (1971), Lemma 3.3) Let $F : D \subset \mathbb{R}^n \to \mathbb{R}^n$ be convex and continuously differentiable on convex set $D_0 \subset D$. If $F'(x)^{-1} \geq 0$ for x in D_0, then F is inverse isotone on D_0. Conversely, if F is inverse isotone on D_0, then $F'(x)^{-1} \geq 0$ for each x in the interior of D_0.*

Moré (1971)'s proof is for inverse-isotonicity of convex functions. Similar result of inverse-antitonicity for concavity can be obtained directly by flipping the inequality in (3.1) of Moré (1971) and change the assumption from $F'(x)^{-1} \geq 0$ to $F'(x)^{-1} \leq 0$[22]. All other conditions are satisfied for $H(p, V)$. Note that, fixing the current value of p, $dH(p, V)/dV' = -(I_{|\mathcal{X}|} - \beta F_p)$ by Proposition 1.5. With $\beta < 1$, this matrix is diagonally dominant with positive diagonal and non-positive off-diagonal elements – thus, an M-matrix. Moreover, $dH(p, V)/dV'$ is nonsingular with $(dH(p, V)/dV')^{-1} \leq 0$, which is clear from the Neumann series representation: $-(I_{|\mathcal{X}|} - \beta F_p)^{-1} = -\sum_{t=0}^{\infty}(\beta F_p)^t$. Then the conditions for Lemma 4 is satisfied. That is, $H(p, V)$ is inverse-antitone in V.

Proposition 1.6 follows from,

$$\begin{aligned} H(p, V + k\iota) &= T(p, V + k\iota) - (V + k\iota) = U_p + \beta F_p V - V + \beta F_p k\iota - k\iota \\ &= H(p, V) - (1 - \beta)k\iota \end{aligned}$$

The last equation follows since F_p has its row elements summed up to 1 and thus, $F_p\iota = \iota$. □

Proof of Proposition 4

Proof. Proving Proposition 4.1 is direct. Firstly, the parameters α and γ are linear functions in generating V and p. Secondly, Gao and Pavel (2018) has shown that the smoothing in (14) is a

22. By flipping the inequality, (3.1) in Moré (1971) becomes $Fy - Fx \leq F'(x)(y - x)$, or $F'(x)^{-1}[Fy - Fx] \leq y - x$. Since $F'(x)^{-1} \leq 0$, $Fy - Fx \leq 0$ infers $y - x \geq 0$.

convex function with respect to inputs[23]. Therefore, the concavity of V and p regarding $L(\cdot)$ in Proposition 3.2 would not change with the inclusion of polynomial approximation and the log-sum-exp smoothing techniques for minimization.

To prove Proposition 4.2, we want to show that the fixed point $(\tilde{p}, \tilde{V})$ from $\tilde{L}(\cdot)$ would converge to the fixed point $(\bar{p}, \bar{V})$ from exact minimization $\bar{L}(\cdot)$. By the uniform error bound (3), we know that $\tilde{L}(\cdot)$ converges uniformly to $\bar{L}(\cdot)$. Define a projector Γ that transforms V^{h-1} to its next iteration as $\Gamma(V)$ and $\Gamma_\sigma(V)$. The fixed point of value function solved from a σ-smoothing approximation is $\tilde{V}_\sigma$.

We proceed on the proof following two steps. Firstly, we show that as $\sigma \to 0$, we have $\Gamma_\sigma(V) \to \Gamma(V)$ for all $V \in \mathcal{V}$. Secondly, we prove by contradiction, that as $\sigma \to 0$, the fixed point solution $\tilde{V}_\sigma$ converges to the fixed point $\bar{V}$ of the exact problem.

To show that $\Gamma_\sigma(V) \to \Gamma(V)$ for all $V \in \mathcal{V}$, we refer to the consistency proof of extreme estimator with a concave objective function as in Newey and McFadden (1994). We need to verify that the assumptions required for Theorem 2.7 in Newey and McFadden (1994) are satisfied. When V^{h-1} is projected into the next iteration, firstly the implied policy function is calculated. For a given V^{h-1}, the policy function solves $\tilde{p} = \arg\max_p \tilde{L}(p, V^{h-1})$ or $\bar{p} = \arg\max_p \bar{L}(p, V^{h-1})$, where V^{h-1} can be considered as a set of parameters determined outside the optimization. If the policies are CCPs that can be calculated analytically like in Example 1, we would have $\tilde{p} = \bar{p}$.

If not, we need to verify that $\tilde{p} \to \bar{p}$ by showing all the assumptions satisfied for Theorem 2.7 in Newey and McFadden (1994). For any V^{h-1}, by assumptions in Proposition 4, $\tilde{L}(p, V^{h-1})$ is a concave function on p. $\tilde{L}(p, V^{h-1}) \to \bar{L}(p, V^{h-1})$ for all $p \in \mathcal{P}$, as suggested by the error bound of log-sum-exp approximation in (15), where for a finite $|\mathcal{X}|$, the approximation error goes to 0 as $\sigma \to 0$. Therefore, all the assumptions for Theorem 2.7 in Newey and McFadden (1994) hold and $\tilde{p} \to \bar{p}$ as $\sigma \to 0$.

With the updated policy at hand, V^h is calculated through optimization $\arg\max_{V^h} \tilde{L}(\tilde{p}, V^h)$ and $\arg\max_{V^h} \bar{L}(\bar{p}, V^h)$. Following the similar logit as above, with $\tilde{p} \to \bar{p}$ and $\sigma \to 0$, we have all the assumtpions for Theorem 2.7 hold and $\tilde{V}^h \to \bar{V}^h$, or $\Gamma_\sigma(V) \to \Gamma(V)$.

For the second part, suppose instead, fixed point $\tilde{V}_\sigma$ converges to $\hat{V}$ that is different from $\bar{V}$ with

23. Following a similar logic, one can show that the corresponding log-sum-exp for minimization is concave.

$||\hat{V} - \bar{V}|| \neq 0$. By the definition of convergence, for a fixed radius $\epsilon_1 > 0$, there exists a σ_1 such that for all $\sigma \leq \sigma_1$, there is $||\tilde{V}_\sigma - \hat{V}|| \leq \epsilon_1/2$. Thus, for those σ, $\tilde{V}_\sigma$ is different from $\bar{V}$. Then by the assumption that $\bar{V}$ is the only fixed point for $\Gamma(V)$, $\tilde{V}_\sigma$ is not a fixed point for Γ with $||\Gamma(\tilde{V}_\sigma) - \tilde{V}_\sigma|| = \epsilon_2 > 0$. Also because $\Gamma_\sigma(V) \to \Gamma(V)$ for all V, there exists a $\sigma_2 > 0$ such that for all $\sigma \leq \sigma_2$, $||\Gamma_\sigma(\tilde{V}_\sigma) - \Gamma(\tilde{V}_\sigma)|| \leq \epsilon_2/2$. Then for any $\sigma <= \min\{\sigma_1, \sigma_2\}$, by the triangular inequality, we have,

$$\begin{aligned} ||\Gamma_\sigma(\tilde{V}_\sigma) - \tilde{V}_\sigma|| &= ||\Gamma(\tilde{V}_\sigma) - \tilde{V}_\sigma + \Gamma_\sigma(\tilde{V}_\sigma) - \Gamma(\tilde{V}_\sigma)|| \\ &\geq ||\Gamma(\tilde{V}_\sigma) - \tilde{V}_\sigma|| - ||\Gamma_\sigma(\tilde{V}_\sigma) - \Gamma(\tilde{V}_\sigma)|| \\ &\geq \epsilon_2 - \frac{\epsilon_2}{2} = \frac{\epsilon_2}{2} > 0 \end{aligned}$$

which is contradicted with the assumption that $\tilde{V}_\sigma$ is a fixed point under the projector Γ_σ. Thus, the fixed point sequence $\tilde{V}_\sigma$ would not converge to a different point from $\bar{V}$. Or in other words, $\tilde{V}_\sigma \to \bar{V}$ as $\sigma \to 0$. □

Proof of Proposition 5

Proof. Proposition 5 is proved by applying the similar logic of the proof for Lemma 2. Firstly, there is,

$$H(p, V + (1-\beta)^{-1}\{\max_x H(p,V)(x)\}\iota) = H(p,V) - \{\max_x H(p,V)(x)\}\iota \leq 0 \tag{A.1}$$

By Lemma 1, we have $V_p \leq T(p,V) \leq V$ whenever $H(p,V) \leq 0$. Combined with the expression of $L(p,V)$ in the proof of Proposition 3(i), we can write $\hat{L}(p,V)$ in (23) as $\hat{L}(p,V) = T(p, V + (1-\beta)^{-1}\{\max_x H(p,V)\}\iota)$. Then consider the polynomial approximation $\Psi\gamma$ of value function with $K < |\mathcal{X}|$ and the smoothing technique in (14) and (15), there is,

$$V_p \leq \hat{L}(p,V) \leq w'T(p,V) + \beta(1-\beta)^{-1}\sigma \ln\left(\sum_x \exp\{\sigma^{-1}H(p,V)\}\right), \text{ for a fixed } V \tag{A.2}$$

And $\min_V \hat{L}(p,V) \leq \min_\gamma \hat{L}(p, \Psi\gamma)$ since $\Psi\gamma$ contains less flexibility in optimization. Thus, having polynomial approximation or smoothing technique would not hurt the upper bound constructed by (23) for V_p. □

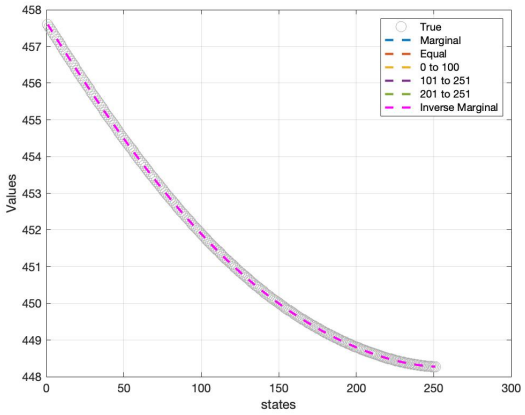
Appendix B. Simulations for Choices of w and σ

Figure B.1 contains two approximations from the different data-generating processes. We fix $\sigma = 0.0001$. Case 1 follows the baseline setting in Section 5.1, while case 2 expands the state space but does not change the flow utility, leading to near-zero stationary probabilities for large states. Specifically case 2 contains states $\{0, 1, 2, \cdots, 499\}$ with the same θ^*.

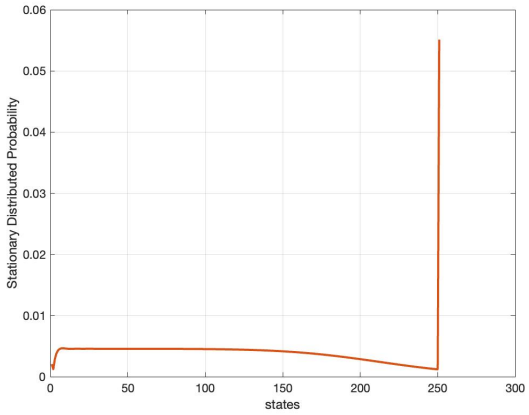
Figure B.1 experiments among different choices of w regarding the approximation quality. The legends in Figure B.1 Panel (c) refers to – (1) Marginal: $w_i = \pi_i$; (2) Equal: $w_i = 1/|\mathcal{X}|$; (3) 0 to 250: $w_i = 1/|\mathcal{X}|$ if state $i \in \{0, 1, 2, \cdots, 250\}$, and $w_i = 0$ otherwise; (4) 251 to 499: $w_i = 1/|\mathcal{X}|$ if state $i \in \{251, 252, \cdots, 499\}$, and $w = 0$ otherwise; (5) 401 to 499: $w_i = 1/|\mathcal{X}|$ if state $i \in \{401, 402, \cdots, 499\}$, and $w = 0$ otherwise; (6) 450 to 499: $w_i = 1/|\mathcal{X}|$ if state $i \in \{450, 451, \cdots, 499\}$, and $w = 0$ otherwise; (7) 481 to 499: $w_i = 1/|\mathcal{X}|$ if state $i \in \{481, 482, \cdots, 499\}$, and $w = 0$ otherwise; (8) Inverse Marginal: w is the inverse of the stationary distribution probability. Similar notations apply to Panel (a) for case 1.

In Figure B.1, (a) and (c) indicate how precise the approximation is towards the value function, while (c) and (d) contain the corresponding stationary distribution of states. Although the approximated $\tilde{V}$ is robust to the w in case 1, having $w_i = \pi_i$ in case 2 may generate bias for V, a direct result from (d) since $\pi_i \approx 0$ for large states. However, this may be harmless in real applications since if $\pi_i \approx 0$ for some large x, we would not observe those x and would not have such discretization. Moreover, (c) in Figure B.1 confirms the claim in de Farias (2002) that those states with $\pi_i \approx 0$ need to be emphasized for a better overall performance of approximation. In our simulation, w_i that only emphasizes the states 401 to 499 still leads to a precise approximation over the entire profile of V.

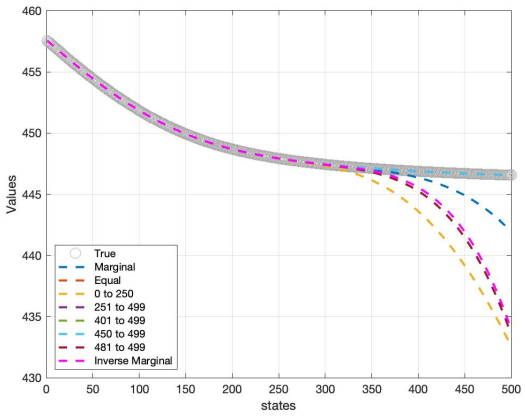
Figure B.2 illustrates the behavior of σ-smoothing, holding $w(x_i) = \pi(x_i)$. Figure B.2 (a) contains the V approximated for case 1, and (b) shows an alternative case 3 with $\theta^* = (20, -1)'$, and state space $\{0, 0.25, 0.5, 1, \cdots, 800\}$. Figure B.2 illustrates that the approximation quality varies with the choice of σ, confirming the conjecture in Section 5.1.2 that a σ that is too large or too small would be problematic. Therefore, in the simulations below, we fix the σ with $\sigma = 0.001$ or $\sigma = 0.0001$.



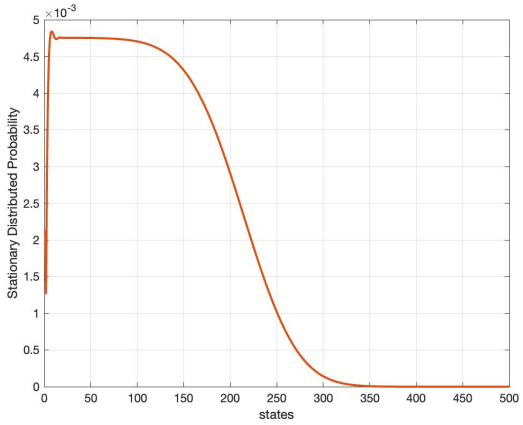
(a) Approximated value function, case 1.



(b) π_i, case 1.

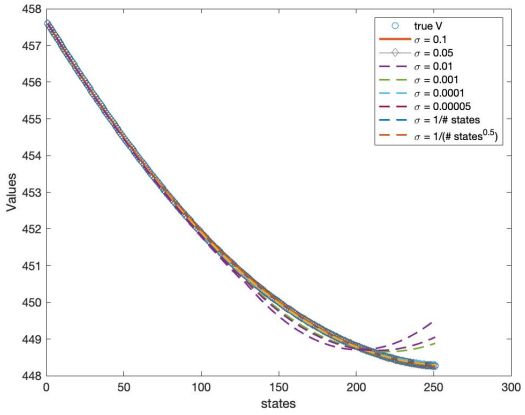


(c) Approximated value function, case 2.

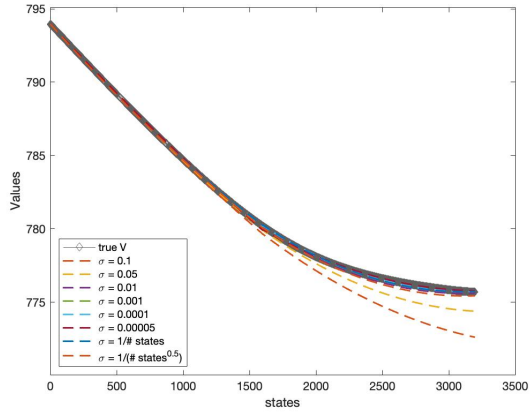


(d) π_i, case 2.

Figure B.1: Simulations for the choice of w, for a simulated sample.



(a) Approximated value function, case 1.



(b) Approximated value function, case 3.

Figure B.2: The value function approximation regarding different σ.