cemmap
centre for microdata methods and practice

Solving dynamic discrete choice models using smoothing and sieve methods

Dennis Kristensen
Patrick K. Mogensen
Jong Myun Moon
Bertel Schjerning

The Institute for Fiscal Studies
Department of Economics,
UCL

cemmap working paper CWP15/19

E·S·R·C ECONOMIC & SOCIAL RESEARCH COUNCIL
An ESRC Research Centre

Solving Dynamic Discrete Choice Models Using Smoothing and Sieve Methods*

Dennis Kristensen† Patrick K. Mogensen‡ Jong Myun Moon§
Bertel Schjerning¶

March 29, 2019

Abstract

We propose to combine smoothing, simulations and sieve approximations to solve for either the integrated or expected value function in a general class of dynamic discrete choice (DDC) models. We use importance sampling to approximate the Bellman operators defining the two functions. The random Bellman operators, and therefore also the corresponding solutions, are generally non-smooth which is undesirable. To circumvent this issue, we introduce a smoothed version of the random Bellman operator and solve for the corresponding smoothed value function using sieve methods. We show that one can avoid using sieves by generalizing and adapting the "self-approximating" method of Rust (1997b) to our setting. We provide an asymptotic theory for the approximate solutions and show that they converge with $\sqrt{N}$-rate, where N is number of Monte Carlo draws, towards Gaussian processes. We examine their performance in practice through a set of numerical experiments and find that both methods perform well with the sieve method being particularly attractive in terms of computational speed and accuracy.

Keywords: Dynamic discrete choice; numerical solution; Monte Carlo; sieves.

*We would like to thank John Rust, Victor Aguirregabiria, Lars Nesheim, Aureo de Paula and many other people for helpful comments and suggestions. Kristensen gratefully acknowledges financial support from the ERC (through starting grant No 312474 and advanced grant No GEM 740369). Schjerning gratefully acknowledges the financial support from the Independent Research Fund Denmark (grant no. DFF – 4182-00052) and the URBAN research project financed by the Innovation Fund Denmark (IFD).

†Department of Economics, University College London, Gower Street, London, United Kingdom. E-mail: d.kristensen@ucl.ac.uk. Website: https://sites.google.com/site/econkristensen.

‡Department of Economics, University of Copenhagen, Øster Farimagsgade 5, Building 35, DK-1353 Copenhagen K, Denmark. E-mail: Patrick.Kofod.Mogensen@econ.ku.dk. Webpage: http://www.economics.ku.dk/staff/phd_kopi/?pure=en/persons/374766.

§PIMCO

¶Department of Economics, University of Copenhagen, Øster Farimagsgade 5, Building 35, DK-1353 Copenhagen K, Denmark. E-mail: Bertel.Schjerning@econ.ku.dk. Webpage: http://bschjerning.com/.

1 Introduction

Discrete Decision Processes (DDPs) are widely used in economics to model forward-looking discrete decisions. For their implementation, researchers are required to solve the model which generally cannot be done in closed form. Instead, a number of methods have been proposed for solving the model numerically; see, e.g., Rust (2008) for an overview. We propose two novel methods for approximating the solutions to a general class of Markovian DDP models in terms of either the so-called integrated or expected value function. Our framework allows for both continuous and discrete state variables, non-separable utility functions and unrestricted dynamics. As such, we cover most relevant models used in empirical work. The proposed implementation of model and estimators are found to be computationally very efficient, and at the same time providing precise results with small approximation errors due to the use of simulations and sieve methods.

Our first proposal proceeds in three steps: First, we develop smoothed simulated versions of the Bellman operators that returns the integrated and expected value functions as fixed points. Next, we approximate the unknown value function by a sieve, that is, a parametric function class, thereby turning the problem into a finite-dimensional one. Finally, we solve for the parameters entering the chosen sieve using projection-based methods. When the chosen sieve is linear in the parameters, the approximate solution can be computed using an iterative procedure where each step is on closed form.

As an alternative to the above sieve-based method, we also adapt and generalize the so-called "self-approximating" method proposed in Rust (1997b) to our setting: We design the importance sampler used in the simulated Bellman operators so that the corresponding expected and integrated value functions can be solved for directly without the use of sieves. In comparison with the sieve approach, the self-approximating solution method has the advantage that it will not suffer from any biases due to function approximations. But at the same time, the simulated Bellman operator used in its implementation will generally have a larger variance compared to the one that can be used for the sieve-method. This larger variance also translates into a larger simulation bias of the self-approximating solution due to the non-linear nature of the problem. Thus, neither method strictly dominates the other.

Our two procedures, the sieve-based and self-approximating one, differ from existing proposals in three important aspects: First, we solve for either the integrated or expected value function instead of the value function itself. This reduces the dimensionality of the problem since we integrate out any i.i.d. shocks appearing in the model before solving it. Moreover, while the value function is non-differentiable, the integrated and expected value functions are generally smooth which make them easier to approximate using sieve methods. Second, we allow for a general class of importance samplers in the simulation of the Bellman operator; these can be designed to reduce variances and biases due to simulations. Third, we smooth the simulated Bellman operator by replacing the max-function appearing in the Bellman operator by a smoothed version where the degree of smoothing is controlled by a parameter akin to the bandwidth in kernel smoothing methods. This is similar to the logit-smoothed accept-reject simulator of probit models as proposed by McFadden (1989); see also Fermanian and Salanie (2004), Kristensen and Shin (2012) and Iskhakov, Jørgensen, Rust and Schjerning (2017). The

smoothing turns the problem of solving for the integrated and expected value functions into differentiable ones. In particular, the exact solutions to the smoothed simulated Bellman equations become smooth as functions of state variables and any underlying structural parameters. This in turn means that standard sieves, such as polynomials, will approximate the exact solutions well and that we can control the error rate due to function approximation. Moreover, if used in estimation, standard numerical solvers can be employed in computing estimators of the structural parameters. The smoothing entails an additional bias but this can be controlled for by suitable choice of aforementioned smoothing parameter.

The smoothing device also facilitates the theoretical analysis of the approximate value functions since it allows us to use a functional Taylor expansion of it. This expansion is then used to analyze the leading numerical error terms of the approximate value functions due to simulations, smoothing and function approximations. In particular, under regularity conditions, we show that the approximate value function will converge weakly towards a Gaussian process which is the first result of its kind to our knowledge. These results allow researchers to, for example, build confidence intervals around the approximate value function and should be useful when analyzing the impact of value function approximation when used in estimation of structural parameters. The results may also be potentially helpful in designing selection rules for number of basis functions and the smoothing parameter.

A numerical study investigates the performance of the solution methods in practice. We implement the proposed methods for the engine replacement model of Rust (1987) and investigate how smoothing, number of basis functions and number of simulations affect the approximation errors. We also investigate how the procedures are affected by the dimensionality of the problem and how derivative-based solvers affect computation times. We find that the sieve method generally performs best of the two methods: It is computationally faster and in most situations provides a better approximation in terms of bias and variance. Moreover, the sieve method is found to also work well in higher dimensions with its bias and variance being fairly stable as we increase the the number of state variables of the model. In contrast, variances of the self-approximating method increase dramatically as the number of state variables increases and so appears to be less robust. Finally, the errors due to simulations and function approximation behave according to theory and are found to vanish at the expected rates.

Our proposed methods share similarities with the ones developed in, amongst others, Arcidiacono, Bayer, Bugni and James (2013), Keane and Wolpin (1994), Munos and Szepesvari (2008), Norets (2012) Rust (1997b) and Pal and Stachurski (2013) who also use simulations and/or sieve methods to solve DDP's. However, their methods solve for the value function while ours solve for either the integrated or expected value function which are more well-behaved (smooth) objects and therefore easier to approximate. Moreover, in contrast to the cited papers, we employ importance sampling and smoothing in our implementation which comes with the aforementioned computational advantages. From a theory perspective, we provide a more complete asymptotic analysis of the approximate integrated and expected value functions. On the other hand, Munos and Szepesvari (2008) and Rust (1997b) provide an analysis of the computational complexity of solving for the value function and so the theories of this paper and these studies complement each other.

The remains of the paper are organized as follows: Section 2 introduces a general class of DDP's and their "smoothed" versions. In Section 3, we develop our smoothed simulated versions of the Bellman operators that the integrated and expected value functions are fixed points to. We then show how to (approximately) solve these simulated Bellman equations in Section 4. An asymptotic theory of the approximate value function is presented in Section 5, while the results of the numerical experiments are found in Section 6. All proofs and lemmas have been relegated to Appendix B and C, respectively.

2 Model

We consider the following DDP where a single agent at time $t \geq 1$ solves

$$d_t = \arg\max_{d \in \mathcal{D}} \{u(S_t, d) + \beta E\left[\nu(S_{t+1})|S_t, d_t = d\right]\}, \tag{2.1}$$

where $\mathcal{D} = \{1, ..., D\}$ is the set of alternatives, $u(S_t, d)$ is the per-period utility, $0 < \beta < 1$ is the discount factor, S_t is a set of state variables that follows a controlled Markov process with transition kernel $F_S\left(S_t|S_{t-1}, d_{t-1}\right)$ and the so-called value function ν solves the following fixed-point problem,

$$\nu(S_t) = \max_{d \in \mathcal{D}} \{u(S_t, d_t) + \beta E\left[\nu(S_{t+1})|S_t, d_t = d\right]\}. \tag{2.2}$$

In many of the empirical specifications, S_t contains an i.i.d. component, $S_t = (Z_t, \varepsilon_t) \in \mathcal{Z} \times \mathcal{E} \subseteq \mathbb{R}^{d_z} \times \mathbb{R}^{d_\varepsilon}$ where Z_t and ε_t satisfy the following conditional independence condition,

$$F_S\left(Z_t, \varepsilon_t|Z_{t-1}, \varepsilon_{t-1}, d_{t-1}\right) = F_\varepsilon\left(\varepsilon_t|Z_t\right) F_Z\left(Z_t|Z_{t-1}, d_{t-1}\right).$$

If no i.i.d. component is present, we can always choose $\varepsilon_t = \emptyset$ to be an empty variable so that $S_t = Z_t$. Throughout, we will assume that $\mathcal{Z}$ is a compact set. This is done to simplify the theoretical analysis since it, for example, implies that all relevant functions will lie in the space $\mathbb{B}\left(\mathcal{Z}\right)$ of bounded functions on $\mathcal{Z}$ equipped with the sup-norm, $\|v\|_\infty = \sup_{z \in \mathcal{Z}} |v\left(z\right)|$. We allow for both countable and continuously distributed state variables. However, many of the ideas and results only have real bite in the continuous case.

In the current formulation, the model is characterized in terms of $\nu(s)$. However, it is possible to rewrite the models in terms of either the so-called *integrated value function* or the *expected value function* and solve for these instead. These are defined as

$$v(Z_t) = E\left[\nu(Z_t, \varepsilon_t)|Z_t\right] = \int_{\mathcal{E}} \nu\left(Z_t, e\right) dF_\varepsilon\left(e|Z_t\right),$$

and

$$V(Z_t, d_t) = E\left[\nu(Z_{t+1}, \varepsilon_{t+1})|Z_t, \varepsilon_t, d_t\right] = E[v(Z_{t+1})|Z_t, d_t] = \int_{\mathcal{E}} v(z') dF_Z\left(z'|Z_t, d_t\right),$$

respectively, where we have used the conditional independence assumption. Eq. (2.1) can now be written as $d_t = \arg\max_{d \in \mathcal{D}} \{u(Z_t, \varepsilon_t, d) + \beta V\left(Z_t, d\right)\}$. Similarly, conditional choice

probabilities, which are needed for counterfactuals and for estimation, take the form

$$P\left(d_t = d|Z_t = z\right) = M_{u,d}(\beta V(z')|z),\ M_{u,d}(r|z) = \frac{\partial M_u(r|z)}{\partial r\left(d\right)},$$

where $M_u(r|z)$ is a generalized version of the so-called social surplus function defined as, for any $r = (r(1), ..., r(D))$,

$$M_u(r|z) = \int_{\mathcal{E}} \max_{d\in\mathcal{D}} \left\{u(z, e, d) + r(d)\right\} dF_\varepsilon(de|z). \tag{2.3}$$

The conditional choice probabilities again take as input the expected value function. Thus, for any implementation of the above class of DDP's, such as the computation of estimators and/or counterfactual analysis, we need to be able to compute either V directly, or first v and then $V\left(Z_t, d_t\right) = E[v(Z_{t+1})|Z_t, d_t]$. Except for a few special cases, analytical expressions of v and V are not available and so numerical approximations have to be employed.

We will here develop numerical methods for solving for either v or V instead of ν for the following reasons: First, ν is a function of $s = (z, \varepsilon)$ while V and v are functions of z alone and therefore their approximations are lower-dimensional problems. Second, ν is non-differentiable due to the max-function in (2.2); in contrast, $v(z)$ and $V(z, d)$ are both smooth functions of z if $F_\varepsilon(e|z)$ and $F_Z\left(z'|z, d\right)$ are. If there is no i.i.d. component in the model, $\varepsilon_t = \emptyset$, then $\nu\left(s\right) = \nu\left(z\right) = v(z)$, while $V(z, d)$ remains smooth even in this case. The functions v and V each solves their own fixed-point problem: Taking conditional expectations on both sides of eq. (2.2), V can be expressed as the solution to

$$V(z, d) = \Gamma(V)(z, d), \tag{2.4}$$

where, with M_u defined in eq. (2.3),

$$\begin{aligned}
\Gamma(V)(z, d) &= E\left[\max_{d'\in\mathcal{D}} \left\{u(Z_{t+1}, \varepsilon_{t+1}, d') + \beta V(Z_{t+1}, d')\right\} |Z_t = z, d_t = d\right] \\
&= \int_{\mathcal{Z}} \int_{\mathcal{E}} \max_{d'\in\mathcal{D}} \left\{u(z', e, d') + \beta V(z', d')\right\} dF_\varepsilon(de|z') dF_Z(dz'|z, d) \\
&= \int_{\mathcal{Z}} M_u(\beta V(z')|z') dF_Z(dz'|z, d).
\end{aligned}$$

Here and in the following, we let $V\left(z\right) = \left(V(z, 1), ..., V(z, D)\right)'$ denote the $D \times 1$-vector of expected value function and similar for other objects. With this notation, we can represent the fixed-point problem on vector form, $V\left(z\right) = \Gamma(V)(z)$, where

$$\Gamma(V)(z) = \int_{\mathcal{Z}} M_u(\beta V(z')|z') dF_Z(dz'|z). \tag{2.5}$$

Next, to derive the fixed-point problem that v solves, again take conditional expectations on both sides of eq. (2.2) but now only condition on Z_t to obtain

$$v(z) = M_u(\beta V(z)|z). \tag{2.6}$$

Combining this with eq. (2.4),

$$v(z) = M_u\left(\beta \int_{\mathcal{Z}} M_u(\beta V(z')|z') dF_Z(dz'|z)\middle| z\right) = \bar{\Gamma}(v)(z). \tag{2.7}$$

where

$$\bar{\Gamma}(v)(z) = M_u\left(\beta \int_{\mathcal{Z}} v(z')\, dF_Z(dz'|z)\middle| z\right).$$

Under great generality, Γ and $\bar{\Gamma}$ are contraction mappings and so $V \in \mathbb{B}(\mathcal{Z})^D$ and $v \in \mathbb{B}(\mathcal{Z})$ are well-defined and unique but closed-form expressions of are only available in a few special cases.

Example 1. Consider the special case where $u(Z_{t+1}, \varepsilon_{t+1}, d) = \bar{u}(Z_{t+1}, d) + \lambda\varepsilon_{t+1}(d)$ for some scale parameter $\lambda > 0$ and $F_\varepsilon(e|z) = F_\varepsilon(e)$ in which case

$$M_u(r|z) = \int_{\mathcal{E}} \max_{d\in\mathcal{D}} \{\bar{u}(z,d) + \lambda e + r(d)\}\, dF_\varepsilon(e) = G_\lambda(\bar{u}(z) + r),$$

where $G_\lambda(r) = \int_{\mathcal{E}} \max_{d\in\mathcal{D}} \{\lambda e + r(d)\}\, dF_\varepsilon(e)$. Thus,

$$\bar{\Gamma}(v)(z) = G_\lambda\left(\bar{u}(z) + \beta \int_{\mathcal{Z}} v(z')\, dF_Z(z'|z)\right).$$

Finally, if $\varepsilon_t(1), ..., \varepsilon_t(D)$ are mutually independent and each elements follows a suitably normalized extreme value distribution, we obtain

$$\int_{\mathbb{R}^D} \max_{d\in\mathcal{D}} \{r(d) + \lambda e(d)\} dF_\varepsilon(de) = \lambda \log\left[\sum_{d\in\mathcal{D}} \exp\left(\frac{r(d)}{\lambda}\right)\right] =: G_\lambda(r), \tag{2.8}$$

where $r = (r(1), ..., r(D))$, c.f. Rust, Traub and Wozniakowski (2002).

3 Simulated Bellman operators

As a first step towards a computationally feasible method for solving for either v or V, we introduce simulated versions of their two Bellman operators, Γ and $\bar{\Gamma}$. We first develop simulated versions of the operators and then introduce their smoothed counterparts. To allow for added flexibility and precision in the implementation and to cover as special case a modified version of Rust's self-approximating solution method, we employ importance sampling: Let $\Phi_Z(z'|z,d)$ and $\Phi_\varepsilon(e|z)$ be conditional importance sampling distribution functions as chosen by the researcher. These have to be chosen such that $F_Z(\cdot|z,d)$ and $F_\varepsilon(\cdot|z)$ are absolutely continuous w.r.t. $\Phi_Z(\cdot|z,d)$ and $\Phi_\varepsilon(\cdot|z)$, respectively, with Radon-Nikodym derivatives $w_z(\cdot|z,d) \geq 0$ and $w_\varepsilon(\cdot|z) \geq 0$ so that

$$\frac{dF_Z(z'|z,d)}{d\Phi_Z(z'|z,d)} = w_Z(z'|z,d), \quad \frac{dF_\varepsilon(e|z)}{d\Phi_\varepsilon(e|z)} = w_\varepsilon(e|z), \tag{3.1}$$

We will throughout assume that eq. (3.1) is satisfied. In the leading case $dF_Z = f_Z d\mu_Z$ and $d\Phi_Z = \phi_Z d\mu_Z$ for some measure μ_Z in which case $w_Z = f_Z/\phi_Z$ and similar for the sampling of ε_t. The above covers the case where $Z_{t+1}|Z_t, d_t$ has a continuous distribution (in which case μ_Z

is the Lesbesque measure), a discrete distribution (in which case μ_Z is the counting measure) and the mixed case. With discrete finite support, we could in principle compute the exact Bellman equation and its corresponding solution and so would not need to resort to numerical methods. But if the discrete support is large this may still be computationally very demanding and so even in this case the numerical methods developed below may be computationally attractive, c.f. Arcidiacono, Bayer, Bugni and James (2013).

Given the chosen importance sampler, we can rewrite $\Gamma(V)(z,d)$ as

$$\Gamma(V)(z,d) = \int_{\mathcal{Z}} \int_{\mathcal{E}} \max_{d' \in \mathcal{D}} \left\{ u(s',d') + \beta V(z',d') \right\} w\left(s'|z,d\right) d\Phi(ds'|z,d)$$

where $s' = (z', e')$ and

$$w\left(s'|z,d\right) = w_\varepsilon\left(e'|z'\right) w_Z\left(z'|z,d\right), \; \Phi\left(s'|z,d\right) = \Phi_\varepsilon(e'|z')\Phi_Z\left(z'|z,d\right).$$

For a given choice of V, we can then approximate this integral by Monte Carlo methods: First generate $N \geq 1$ i.i.d. draws, $Z_i\left(z,d\right) \sim \Phi_Z\left(\cdot|z,d\right)$ and $\varepsilon_i\left(z,d\right) \sim \Phi_\varepsilon(\cdot|Z_i\left(z,d\right))$, $i = 1, ..., N$, and then compute

$$\Gamma_N(V)(z,d) = \sum_{i=1}^{N} \max_{d' \in \mathcal{D}} \left\{ u\left(S_i\left(z,d\right), d'\right) + \beta V\left(Z_i\left(z,d\right), d'\right) \right\} w_{N,i}\left(z,d\right), \tag{3.2}$$

where $S_i\left(z,d\right) = \left(Z_i\left(z,d\right), \varepsilon_i\left(z,d\right)\right)$ and

$$w_{N,i}\left(z,d\right) = \frac{w\left(S_i\left(z,d\right)|z,d\right)}{\sum_{i=1}^{N} w\left(S_i\left(z,d\right)|z,d\right)}. \tag{3.3}$$

Note here that we normalize the importance weights so that $\sum_{i=1}^{N} w_{N,i}\left(z,d\right) = 1$. This is done to ensure that Γ_N is a contraction mapping on $\mathbb{B}\left(\mathcal{Z}\right)^D$. Similarly, we approximate $\bar{\Gamma}\left(v\right)$ by

$$\bar{\Gamma}_N(v)(z) = \sum_{j=1}^{N} \max_{d' \in \mathcal{D}} \left\{ u(z, \varepsilon_j\left(z,d'\right), d') + \beta \sum_{i=1}^{N} v\left(Z_i\left(z,d'\right)\right) w_{Z,N,i}\left(z,d'\right) \right\} w_{\varepsilon,N,j}\left(z,d'\right), \tag{3.4}$$

where again we normalize the weights to ensure $\bar{\Gamma}_N$ is a contraction on $\mathbb{B}\left(\mathcal{Z}\right)$,

$$w_{Z,N,i}\left(z,d\right) = \frac{w_Z\left(Z_i\left(z,d\right)|z,d\right)}{\sum_{i=1}^{N} w_Z\left(Z_i\left(z,d\right)|z,d\right)}, \; w_{\varepsilon,N,i}\left(z,d\right) = \frac{w_\varepsilon\left(\varepsilon_i\left(z,d\right)|z\right)}{\sum_{i=1}^{N} w_\varepsilon\left(\varepsilon_i\left(z\right)|z\right)}.$$

When $\varepsilon_t = \emptyset$, the simulated Bellman operator $\bar{\Gamma}_N$ includes as special cases the ones considered in Rust (1997b) (who chooses Φ_Z as the uniform distribution uniform on $\mathcal{Z}$) and Pal and Stachurski (2013) (who chooses $\Phi_Z = F_Z$).

Example 1 (continued). In the additive model, we can compute the integral w.r.t. ε_t analytically, c.f. eq. (2.8) and so the the simulated Bellman operator simplifies to

$$\bar{\Gamma}_N(v)(z) = G_\lambda\left(\bar{u}(z) + \beta \sum_{i=1}^{N} v\left(Z_i\left(z\right)\right) w_{Z,N,i}\left(z\right)\right). \tag{3.5}$$

Importantly, the max-function has been replaced by its smoothed version $G_\lambda(\cdot)$.

If $F_\varepsilon(e|z)$ and $F_Z(z'|z,d)$ are smooth functions w.r.t. z then $\Gamma(V)(z)$ and $\bar{\Gamma}(v)(z)$ will be smooth functions of z as well. In contrast, the general versions of $\Gamma_N(V)(z,d)$ and $\bar{\Gamma}_N(v)(z)$ are non-smooth due to the presence of the max-function in their definitions which does not get smoothed for finite N. This in turn implies that their corresponding fixed points, $V_N(z) = \Gamma_N(V_N)(z)$ and $v_N(z) = \bar{\Gamma}_N(v_N)(z)$, will be non-differentiable w.r.t. the state variables, z, and w.r.t. any underlying structural parameters in the model. This is an unattractive feature for several reasons: First, estimation and counterfactuals will be non-smooth problems. Second, the theoretical analysis of V_N and v_N becomes more complicated.

To resolve this issue, we take inspiration from the additive model in Example 1 and propose to smooth the simulated Bellman operators by replacing the "hard" max-function appearing in eqs. (3.2) and (3.4) by the smoothed version $G_\lambda(r)$ defined in eq. (2.8). We now interpret $\lambda > 0$ as a smoothing parameter that plays a role similar to that of the bandwidth in kernel regression estimation. Elementary calculations show

$$0 \leq G_\lambda(r) - \max_{d\in\mathcal{D}} r(d) \leq \lambda \log D, \tag{3.6}$$

so that $G_\lambda(r) \to \max_{d\in D} r(d)$, as $\lambda \to 0$, uniformly in $r \in \mathbb{R}^D$. Substituting in $G_\lambda(r)$ yields the following smoothed simulated operators,

$$\Gamma_{N,\lambda}(V)(z,d) = \sum_{i=1}^{N} G_\lambda\left(u\left(S_i(z,d)\right) + \beta V\left(Z_i(z,d)\right)\right) w_{N,i}(z,d), \tag{3.7}$$

$$\bar{\Gamma}_{N,\lambda}(v)(z) = \sum_{j=1}^{N} G_\lambda\left(u(z,\varepsilon_j(z)) + \beta \sum_{i=1}^{N} v\left(Z_i(z)\right) w_{Z,N,i}(z)\right) w_{\varepsilon,N,j}(z), \tag{3.8}$$

where $u(z,\varepsilon_j(z)) = (u(z,\varepsilon_j(z,1),1),...,u(z,\varepsilon_j(z,D),D))$ and $w_{Z,N,i}(z)$ defined similarly. These will be employed below to obtain smooth approximations to v and V, respectively. The use of $G_\lambda(r)$ in place of $\max_{d\in\mathcal{D}} r(d)$ generates an additional bias in the approximate solutions of order $O_p(\lambda)$ as we shall see in the theory section, but this can be controlled for by suitable choice of λ.

In Example 1, we saw that $G_\lambda(r)$ appears in the Bellman operators as a direct implication of the model specification in which case no smoothing bias will be present. This can be generalized to the following class of models: Suppose that $\varepsilon_t = \left(\varepsilon_t^{(1)}, \varepsilon_t^{(-1)}\right)$ $\varepsilon_t^{(1)} = \left(\varepsilon_t^{(1)}(1),...,\varepsilon_t^{(1)}(D)\right)$ enters the per-period utility additively so that the model of interest takes the form

$$d_t = \arg\max_{d\in\mathcal{D}} \left\{u(Z_t, \varepsilon_t^{(-1)}, d) + \lambda \varepsilon_t^{(1)}(d) + \beta V(Z_t)\right\}, \tag{3.9}$$

where as in Example 1 $\lambda > 0$ is scale parameter that determines the impact of $\varepsilon_t^{(1)}(d)$ on the utility. By the same arguments as in Example 1, we find that the expected value function in this case solves $\Gamma_\lambda(V) = V$ where

$$\Gamma_\lambda(V)(z) = \int_{\mathcal{Z}} \int_{\mathcal{E}} G_\lambda\left(u(z',e')) + \beta V(z')\right) dF_{\varepsilon^{(-1)}}(e'|z) dF_Z(z'|z,d),$$

and $\Gamma_{N,\lambda}$ in eq. (3.7) is clearly an unbiased simulated version of Γ_λ. Similarly, $\bar{\Gamma}_{N,\lambda}$ is an unbiased estimator of $\bar{\Gamma}_\lambda$. To summarize, if the original model of interest contains an additive extreme value term, which is the case in many empirical papers, G_λ appears as part of the model and so no smoothing bias will be present in our proposed simulated Bellman operators.

The above shows that the smoothing device corresponds to adding structural shocks to the DDP of interest. In earlier work on solving DDPs, researchers have in some case done the opposite and removed structural errors in order to facilitate the numerical solution of the model; see Lumsdaine, Stock and Wise (1992) for one example of this. This was, however, done in the context of discrete state variables with a small number of support points in which case removing continuous structural errors meant that the Bellman operators could be evaluated analytically. In our case, the state variables are either continuous or have a very large discrete support in which case simulations are required in the first place to evaluate the Bellman operator. Once simulations are introduced, there is little computational gains from removing shocks from the model and instead introduction of smoothing facilitates solving and analyzing the corresponding solution.

4 Approximate value functions

The simulated Bellman operators $\Gamma_{N,\lambda}(V)$ and $\bar{\Gamma}_{N,\lambda}(v)$ in eqs. (3.7)-(3.8) are contraction mappings on $\mathbb{B}(\mathcal{Z})^D$ and $\mathbb{B}(\mathcal{Z})$, respectively, and so they have unique fixed points defined as

$$V_{N,\lambda} = \Gamma_{N,\lambda}(V_{N,\lambda}),\ v_{N,\lambda} = \bar{\Gamma}_{N,\lambda}(v_{N,\lambda}). \tag{4.1}$$

However, solving these two simulated Bellman equations are not generally feasible since these are infinite-dimensional problems. We here present two ways to reduce the problems to a finite-dimensional ones. The first method is a generalized version of the so-called self-approximating method proposed in Rust (1997b) while the second one uses projection-based methods as advocated by Pal and Stachurski, 2013.

4.1 Self-approximating method

Rust (1997b) proposed to turn the infinite-dimensional problems in eq. (4.1) into a finite-dimensional ones by choosing the importance sampling to be based on marginal, instead of conditional distributions. In our generalized version this corresponds to restricting $\Phi_Z(z'|z,d) = \Phi_Z(z')$ for some marginal distribution $\Phi_Z(\cdot)$ so that the draws $Z_i \sim \Phi_z(\cdot)$ and $\varepsilon_i \sim \Phi_\varepsilon(\cdot|Z_i)$, $i = 1, ..., N$ no longer depend on (z,d). In this case, the fixed-point problems in eq. (4.1) reduce to the following two sets of N nonlinear equations,

$$V_{N,\lambda,k} = \sum_{i=1}^{N} G_\lambda\left(u(S_i) + \beta V_{N,\lambda,i}\right) w_{N,i}(Z_k), \tag{4.2}$$

$$v_{N,\lambda,k} = \sum_{j=1}^{N} G_\lambda\left(u(Z_k,\varepsilon_j) + \beta \sum_{i=1}^{N} v_{N,\lambda,i} w_{z,N,i}(Z_k)\right) w_{\varepsilon,N,j}(Z_k), \tag{4.3}$$

for $k = 1, ..., N$, w.r.t. $\{V_{N,\lambda,k} : k = 1, ..., N\}$ and $\{v_{N,\lambda,k} : k = 1, ..., N\}$, respectively. Here, $V_{N,\lambda,k} = V_{N,\lambda}(Z_k)$ and $v_{N,\lambda,k} = v_{N,\lambda}(Z_k)$, $k = 1, ..., N$. Each of the two sets of equations have a unique solution due to the contracting property of $\Gamma_{N,\lambda}$ and $\bar{\Gamma}_{N,\lambda}$. Once, for example, eq. (4.2) has been solved, the approximate expected value functions can be evaluated at any other value z by

$$V_{N,\lambda}(z) = \sum_{i=1}^{N} G_\lambda \left(u(S_i) + \beta V_{N,\lambda,i} \right) w_{N,i}(z).$$

Note that $V_{N,\lambda}(z)$ is a smooth function even if $\lambda = 0$ as long as $w_{N,i}(z)$ is smooth and so smoothing is not needed for this property to hold when marginal samplers are employed. However, without smoothing, the set of equations (4.2) become non-smooth w.r.t. the variables $\{V_{N,\lambda,k} : k = 1, ..., N\}$ and so cannot be solved using derivative-based methods. Thus, the numerical implementation of the self-approximating method still benefits from smoothing.

In addition to smoothing, the above self-approximating method differs from Rust's original proposal in two other ways: First, while Rust (1997b) solved for the value function $\nu(z, \varepsilon)$, we here solve for either $V(z)$ or $v(z)$. As explained earlier, the latter are the more relevant ones in most applications. Moreover, our formulation allows for the following generalized version of the simulated Bellman equations for $v_{N,\lambda}$,

$$v_{N,\lambda,k} = \sum_{j=1}^{\tilde{N}} G_\lambda \left(u(Z_k, \varepsilon_j) + \beta \sum_{i=1}^{N} v_{N,\lambda,i} w_{z,N,i}(Z_k) \right) w_{\varepsilon,N,j}(Z_k), \tag{4.4}$$

where we allow for different number of draws from $\Phi_\varepsilon(\tilde{N})$ and $\Phi_Z(N)$. In particular, we can choose $\tilde{N}$ as large as we wish (thereby decreasing the variance of the problem) without increasing the number of variables that need to be solved for (N). A similar generalization of the simulated Bellman equations for $V_{N,\lambda}$ is possible. Second, we here only require that the state dynamics together with the chosen importance sampler satisfy (3.1); in contrast, Rust (1997b) assumed that S_t was continuously distributed with compact support and chose as importance sampler the uniform distribution with same support. Thus, our version allows for a broader class of models and samplers.

The self-approximating method may not always work well: First, finding a marginal distribution $\Phi_Z(\cdot)$ so that (3.1) holds can be difficult in some models. For example, in many specifications with continuous dynamics, the transition density $f_Z(z'|z, d)$ of Z_t will have singularities, e.g., $\lim_{z' \to z} f_Z(z'|z, d) = +\infty$, in which case there does not exist a marginal density $\phi_Z(z')$ so that $w_Z(z'|z, d) = f_Z(z'|z, d) / \phi_Z(z')$ is well-defined. And if $w_Z(z'|z, d)$ is not well-defined, the corresponding importance sampler is inconsistent. And even if (3.1) does hold, the use of marginal samplers instead of conditional ones will generally lead to a larger variance of the solutions since the "marginal" draws $Z_1, ..., Z_N$ do not adapt to the changing shape of $F_Z(\cdot|z, d)$ as a function of z. In particular, many of the draws may fall outside of the support of $F_Z(\cdot|z, d)$ and so are "wasted" in which case a large N is required to achieve a reasonable approximation; see Section 6 for an example of this. This issue tends to become more severe in higher-dimensions (when d_z is large) since the volume of the support shrinks, and so the self-approximating method will generally suffer from a built-in curse-of-dimensionality; see Section 6 for an example of this. This curse-of-dimensionality does not appear in the subclass of models

that Rust (1997b) focused on where it was assumed that $S_t|S_{t-1}, d_{t-1}$ has support $[0,1]^{\dim(S_t)}$ for all values of S_{t-1}, d_{t-1}.

Finally, given that $V_{N,\lambda}$ and $v_{N,\lambda}$ are solutions to non-linear equations, a large variance in the simulated Bellman operator translates into a large bias as is well-known from non-linear GMM estimators. This can be controlled for by choosing N large. But large N means that numerically solving either (4.2) or (4.3) becomes computationally very costly. These issues motivate us to pursue a sieve-based solution strategy.

4.2 Sieve-based method

We now return to the general versions of the simulated Bellman operators and so again allow for conditional importance samplers. Let $\bar{\mathcal{V}} \subseteq \mathbb{B}(\mathcal{Z})$ be a suitable function space that $v_{N,\lambda}$ defined in (4.1) is known to lie in; see below for more details on this. We then choose a finite-dimensional function space (commonly called a sieve in the econometrics literature) $\bar{\mathcal{V}}_K = \{v_K(\cdot;\alpha) : \mathcal{Z} \mapsto \mathbb{R} | \alpha \in \mathcal{A}_K\} \subseteq \bar{\mathcal{V}}$, where $\mathcal{A}_K \subseteq \mathbb{R}^K$ is a parameter set with $K < \infty$, that provides a good approximation to functions in $\bar{\mathcal{V}}$. Similarly, we let $\mathcal{V} \subseteq \mathbb{B}(\mathcal{Z})^D$ be a space of D-dimensional vector functions that the solution $V_{N,\lambda}$ to (4.1) lie in and $\mathcal{V}_K = \left\{V_K(\cdot;\alpha) : \mathcal{Z} \mapsto \mathbb{R}^D | \alpha \in \mathcal{A}_K\right\} \subseteq \mathcal{V}$ be our sieve for this space. Let

$$\bar{\Pi}_K(v) = \arg\min_{v' \in \bar{\mathcal{V}}_K} \|v - v'\|_{\bar{\mathcal{V}}}, \; \Pi_K(V) = \arg\min_{V' \in \mathcal{V}_K} \|V - V'\|_{\mathcal{V}}, \tag{4.5}$$

be the corresponding projections for given (pseudo-) norms $\|\cdot\|_{\bar{\mathcal{V}}}$ and $\|\cdot\|_{\mathcal{V}}$ as chosen by us as well. We then approximate $V_{N,\lambda}$ and $v_{N,\lambda}$ by the solutions to the projected Bellman equations,

$$\hat{v}_{N,\lambda} = \arg\min_{v \in \bar{\mathcal{V}}_K} \left\|v - \bar{\Pi}_K \bar{\Gamma}_{N,\lambda}(v)\right\|_{\bar{\mathcal{V}}}, \; \hat{V}_{N,\lambda} = \arg\min_{V \in \mathcal{V}_K} \|V - \Pi_K \Gamma_{N,\lambda}(V)\|_{\mathcal{V}}. \tag{4.6}$$

These are finite-dimensional problems of size K. When K is small relative to N, which will generally be the case, the above problems are computationally much more tractable compared to the corresponding self-approximating ones. Note here that these projection-based approximations are different from the least-squares approximations that would solve $\min_{V \in \mathcal{V}_K} \|V - \Gamma_{N,\lambda}(V)\|_{\mathcal{V}}$ and $\min_{v \in \bar{\mathcal{V}}_K} \left\|v - \bar{\Gamma}_{N,\lambda}(v)\right\|_{\bar{\mathcal{V}}}$, respectively. In particular, by suitable choice of the projection operators, $\Pi_K \Gamma_{N,\lambda}$ and $\bar{\Pi}_K \bar{\Gamma}_{N,\lambda}$ will be contraction mappings w.r.t. $\|\cdot\|_\infty$ guaranteeing that $\hat{V}_{N,\lambda}$ and $\hat{v}_{N,\lambda}$ exist and are unique. The following discussion focuses on the integrated value function approximation since it carries over with only minor modifications to the one of the expected value function. We discuss their numerical implementation in further detail in the subsection below.

The projection operator $\bar{\Pi}_K$ can be thought of as a function approximator with the approximation error being $v - \bar{\Pi}_K(v)$ for a given function v. Roughly speaking, the projection-based method approximates $v_{N,\lambda}$ by $\hat{v}_{N,\lambda} = \bar{\Pi}_K(v_{N,\lambda})$ which incurs an additional sieve approximation error, $v_{N,\lambda} - \bar{\Pi}_K(v_{N,\lambda})$. The smoothness of $v_{N,\lambda}$ here proves helpful since many well-known sieves are able to provide good approximations of smooth functions using a low-dimensional space (K is small). Due to these features, our proposed projection-based solutions will generally suffer from quite small additional biases relative to the exact simulated solution. This is in contrast to existing projection-based solution methods, such as the one in Pal and Stachurski

(2013), that aim at approximating the value function $\nu(s)$ which is non-differentiable.

The smoothness of $v_{N,\lambda}$ here help guiding us in choosing the sieve: It allows us to restrict $\bar{\mathcal{V}}$ to a suitable smoothness class and then import existing approximation methods for smooth functions as developed in the literature on numerical methods and nonparametric econometrics. A leading example is the class of linear function approximations where the finite-dimensional function space takes the form of $\bar{\mathcal{V}}_K = \left\{\alpha' B_K(z) : \alpha \in \mathbb{R}^K\right\}$ for a set of basis functions $B_K(z) = \{b_k(z) : k = 1, ..., K\}$. The basis functions can be chosen as, for example, Chebyshev interpolation and B-splines that are able to approximate smooth functions well. However, other non-linear function space are possible such as wavelets, artificial neural networks and shrinkage-type function approximators such as LASSO, where the additional constraints are imposed on α; we refer to Chen, 2007 for a general overview of different function approximators and constrained sieve estimators. We also allow for flexibility in terms of the chosen norms $\|\cdot\|_{\bar{\mathcal{V}}}$ with a leading example being $\|v\|_{\bar{\mathcal{V}}} = \sum_{i=1}^{M} v^2(z_i)$ for a set of design points $z_1, ..., z_M \in \mathcal{Z}$. Very often the $M \geq 1$ design points will be chosen in conjunction with the sieve.

The above procedure does not suffer from any of the above mentioned issues of the self-approximating method: We can use conditional importance samplers freely which can be designed to control the variance of the simulated Bellman operators; and the dimension of the problem remains K irrespectively of the number of draws N. The main drawback is that unique solutions to eqs. (4.6) do not necessarily exist for a given choice of N and K. A sufficient condition for this to hold is that $\bar{\Pi}_K$ is a non-expansive operator w.r.t $\|\cdot\|_\infty$, $\left\|\bar{\Pi}_K(v_1) - \bar{\Pi}_K(v_2)\right\|_\infty \leq \|v_1 - v_2\|_\infty$ since this translates into $\bar{\Pi}_K \Gamma_{N,\lambda}$ being a contraction mapping. However, while, by definition, $\bar{\Pi}_K$ is non-expansive w.r.t. $\|\cdot\|_{\mathcal{V}}$, it is not necessarily non-expansiveness w.r.t $\|\cdot\|_\infty$. Pal and Stachurski (2013) provide some examples of projections that are non-expansive w.r.t. $\|\cdot\|_\infty$, but these are unfortunately computationally expensive to use in general. But $\bar{\Pi}_K$ will generally be close to non-expansive w.r.t $\|\cdot\|_\infty$ asymptotically as $K \to \infty$ for a wide range of sieves and pseudo-norms since

$$\left\|\bar{\Pi}_K\right\|_{op,\infty} := \sup_{v \in \bar{\mathcal{V}}, \|v\|=1} \left\|\bar{\Pi}_K(v)\right\|_\infty \leq \sup_{v \in \bar{\mathcal{V}}, \|v\|=1} \left\|\bar{\Pi}_K(v) - v\right\|_\infty + 1,$$

where the first term in the last expression will go to zero in great generality as $K \to \infty$ for many popular sieves (see next subsection for details). Given that $\bar{\Gamma}_{N,\lambda}$ is a contraction with Lipschitz coefficient $\beta < 1$, this in turn implies that $\bar{\Pi}_K \bar{\Gamma}_{N,\lambda}$ will be a contraction mapping for all K large enough. This will be used in our asymptotic analysis of the algorithm. The above argument is not completely satisfactory in practice since it does not say how large K should be chosen to ensure non-expansiveness of $\bar{\Pi}_K \bar{\Gamma}_{N,\lambda}$. But in our numerical experiments we generally did not experience any such issues.

4.3 Numerical implementation of the two methods

We here discuss in more detail the numerical implementation of the self-approximating and projection-based methods. As before, we here focus on solving for the integrated value function since most results and arguments for this case carries over with very minor modifications to the expected value function. First, the researcher has to choose the importance sampling distribu-

tions, the smoothing parameter λ and, in the case of the projection-based method, the function approximation method. Second, given these choices, either eq. (4.3) or (4.6) has to be solved for.

4.3.1 Importance sampler

The choices of Φ_Z and Φ_ε determine the variance of the $\bar{\Gamma}_{N,\lambda}$ and should ideally be tailored to minimize it. In the case of projection-based methods, where we can choose Φ_Z and Φ_ε as conditional distributions, we can rely on the already existing theory for efficient importance sampling for how to do so; see Chapter 3 in Robert and Casella, 2013 for an introduction. In our numerical experiments, we did not experiment with different choices and throughout set $\Phi_Z = F_Z$ and $\Phi_\varepsilon = F_\varepsilon$.

In the case of the self-approximating method, the choice of Φ_Z is restricted to the class of marginal distributions. Generally, this entails a large variance of the corresponding simulated Bellman operators. It will hold in great generality that (Z_t, d_t) has a stationary distribution, say, $F_Z^*(z,d)$. In this case, a suitable choice would be the marginal of this, $\Phi_Z(z) = \sum_{d\in\mathcal{D}} F_Z^*(z,d)$. However, the stationary distribution depends on the value function and so is rarely available on closed form; so this strategy requires an initial solution and exploration of the model. Alternatively, one can try to construct a good approximation of the stationary approximation through a mixture Markov model on the form $\Phi_Z(z') = \sum_{d\in\mathcal{D}} \int \omega_d(z) F_Z(z'|z,d)\, d\mu_Z(z)$ for a set of pre-specified mixture weights $\omega_d(z) \geq 0$. In the numerical experiments, we follow Rust (1997b) and choose $\Phi_Z(z)$ as the uniform distribution on $\mathcal{Z}$ which we conjecture is far from optimal in many cases, and so more research in this direction is needed.

4.3.2 Smoothing

The use of $G_\lambda(r)$ in place of $\max_{d\in\mathcal{D}} r(d)$ generally generates an additional bias in the corresponding integrated value function of order $O(\lambda)$. At the same time, the variance of $v_{N,\lambda}$ is an increasing function of λ. Thus, ideally we would like to choose λ to balance these two effects. A natural criterion would be to minimize the so-called integrated mean-square-error, $\lambda^* = \arg\min_{\lambda\geq 0} E\left[\int_{\mathcal{Z}} \|v_{N,\lambda}(z) - v(z)\|^2 dF_Z(z)\right]$, where $F_Z(z)$ is a suitably chosen distribution such as the stationary one of Z_t. Since $v(z)$ is unknown and we cannot evaluate the expectations, λ^* cannot be solved for but cross-validation methods can be used instead. This could in principle be done along the same lines as bandwidth selection for smoothed empirical cdfs, see Bowman, Hall and Prvan (1998). However, this is computationally somewhat burdensome. Moreover, in our numerical experiments we found that the quality of the approximate value function was quite insensitive to the choice of λ and so in practice we recommend using very little smoothing such as $\lambda = 0.01$.

4.3.3 Function approximation

As mentioned earlier, many approximation architectures are available in the literature. In our numerical experiments we focus on the class of linear function approximators where $\bar{\mathcal{V}}_K = \left\{\alpha' B_K(z) : \alpha \in \mathbb{R}^K\right\}$ for a set of pre-specified basis functions $B_K(z) \in \mathbb{R}^K$. For a given set of

$M \geq 1$ design points in $\mathcal{Z}$, $z_1, ..., z_M$, eq. (4.5) then becomes

$$\bar{\Pi}_K(v)(z) = B_K(z)' \left[\sum_{i=1}^{M} B_K(z_i) B_K(z_i)'\right]^{-1} \sum_{i=1}^{M} B_K(z_i) v(z_i). \tag{4.7}$$

The design points may either be random or deterministic and can be chosen relative to the basis functions to ensure that $\bar{\Pi}_K$ is easy to compute and provides a good approximation for a broad class of functions. The performance of most function approximations will depend on the smoothness of the function of interest. Due to smoothing, $v_{N,\lambda}$ will be $s \geq 1$ times differentiable if u and F_Z are s times differentiable (see Theorem 1). Let $\mathbb{C}_s(\mathcal{Z})$ denote the space of $s \geq 0$ times continuously differentiable functions on the domain $\mathcal{Z}$; we may then choose $\bar{\mathcal{V}}$ as

$$\bar{\mathcal{V}}_{s,r} = \{v \in \mathbb{C}_s(\mathcal{Z}) : \|v\|_{s,\infty} < r\}, \tag{4.8}$$

for some $r < \infty$ where, with $\alpha = (\alpha_1, ..., \alpha_{d_z}) \in \mathbb{N}_0^{d_z}$,

$$\|v\|_{s,\infty} = \sup_{|\alpha| \leq s} \|D^\alpha v\|_\infty, \; D^\alpha v(z) = \frac{\partial^{\alpha_1 + \cdots + \alpha_{d_z}} v(z)}{\partial^{\alpha_1} \cdots \partial^{\alpha_{d_z}}}. \tag{4.9}$$

We can now employ existing results for approximating functions in $\bar{\mathcal{V}}_{s,r}$ to control the error due to projection:

Example 2. Polynomial interpolation using tensor products. Suppose we use Jth order Chebyshev interpolation with $M \geq J$ nodes in each of the d_z dimensions, or a Jth order B-spline interpolation with $M \geq J$ number of nodes in each of the d_z dimensions (see Appendix D for their precise expressions). Let $p_1, ..., p_J$ denote the J polynomials; we then have

$$B_K(z) = \left\{p_{j_1}(z_1) \cdots p_{j_{d_z}}(z_{d_z}) : j_1, ..., j_{d_z} = 1, ..., J\right\},$$

which is of dimension $K = J^{d_z}$. Choosing $J \geq s$, where $s \geq 1$ denotes the number of derivatives of $v(z)$ both interpolation schemes satisfy, for any radius $r < \infty$,

$$\sup_{v \in \bar{\mathcal{V}}_{s,r}} \left\|\bar{\Pi}_K(v) - v\right\|_\infty = O\left(\frac{\log(J)}{J^{s+1}}\right) = O\left(\frac{\log(K)}{K^{(s+1)/d}}\right);$$

see p.14 in Rivlin (1990) for Chebyshev interpolation and Schumaker (2007) for B-splines.

As can be seen from the above example, standard polynomial tensor product approximations suffer from the well-known computational curse of dimensionality: To reach a given level of error tolerance, the total number of basis functions K has to grow exponentially as d_z increases. This issue can be partially resolved by using more advanced function approximation methods:

Example 3. Interpolation with sparse grids. Instead of using tensor-product basis functions to approximate a given function, where the total number of basis function and interpolation points will have to grow exponentially with d_z to control the approximation error, one can instead use so-called Smolyak sparse grids; see, e.g., Judd, Maliar, Maliar and Valero (2014) and Brumm and Scheidegger (2017). Using these, the number of grid points needed to obtain a given

error tolerance are reduced from $O\left(M^{d_Z}\right)$ to $O\left(M\left(\log M\right)^{d_z}\right)$ with only slightly deteriorated accuracy.

Example 4. Variable selection, shape constraints, shrinkage estimators, and machine learning. An alternative way of breaking the curse of dimensionality is to select the basis functions judiciously. This could, for example, be done using standard variable selection methods; one example of this approach can be found in Chen (1999). Alternatively, one can in some cases show that the value functions satisfy certain shape constraints that can then be imposed on , for example, Cai and Judd (2013). Other automated selection methods include shrinkage methods where a penalization term is added to the least-squares criterion. Again this leads to a more sparse representation which is able to break the curse-of-dimensionality. Finally, machine learning algorithms, such as neural networks, can be employed to obtain good approximations of the value function that suffer from only a moderate curse-of-dimensionality; see, for example, Chen and White (1999).

As noted earlier, there is no guarantee that a given function approximator is non-expansive. But this can, in principle, be checked for a given choice. For the least-squares projection, this amounts to solving, for a given choice of basis functions and grid points,

$$\left\|\bar{\Pi}_K\right\|_{op,\infty} = \sup_{v\in\mathbb{R}^M,\|v\|=1}\sup_{z\in\mathcal{Z}}\left|B_K\left(z\right)'\left[\sum_{i=1}^{M}B_K\left(z_i\right)B_K\left(z_i\right)'\right]^{-1}\sum_{i=1}^{M}B_K\left(z_i\right)v_i\right|. \tag{4.10}$$

When M and/or $\dim\mathcal{Z}$ is large this may be computationally demanding and instead one can obtain a lower bound by restricting z to only take values on the chosen set of grid points: With $\mathbf{B}_{K,M}\in\mathbb{R}^{K\times M}$containing the basis functions evaluated at the grid points, we can represent $\bar{\Pi}_K$ when only evaluated at chosen grid points $z_1,....,z_M$ in terms of

$$\mathbf{P}_{K,M} = \mathbf{B}_{K,M}'\left[\mathbf{B}_{K,M}\mathbf{B}_{K,M}'\right]^{-1}\mathbf{B}_{K,M}\in\mathbb{R}^{M\times M}.$$

In particular, it is easily checked that with the supremum in (4.10) being only taken over $z\in\{z_1,...,z_M\}$, $\left\|\bar{\Pi}_K\right\|_{op,\infty} = \left\|\mathbf{P}_{\mathbf{K},\mathbf{M}}\right\|_{op,\infty}$. Furthermore, $\left\|\mathbf{P}_{K,M}\right\|_{op,\infty}\leq 1$ if and only if

$$\max_{i=1,...,M}\sum_{j=1}^{M}\left|p_{ij}\right|\leq 1,$$

where p_{ij} is the (i,j)th element of $\mathbf{P}$, c.f. Lizotte (2011).

4.3.4 Solving for the approximate value functions

Computing the simulated self-approximating solution or the projection-based one can be done using three different numerical algorithms: Successive approximation (SA), Newton-Kantorovich (NK), or a combination of the two. The latter corresponds to the hybrid solution method proposed in Rust (1988). We here discuss the implementation of these algorithms with focus on the sieve-based approximation of v; the implementations of the sieve approximation of V and the self-approximating solutions of either of the two follow along the same lines. The main

difference between solving for V or v is that the latter involves smaller computational burden since it is a scalar function while the former is a D-dimensional vector function.

SA utilizes that (for K chosen large enough), $\bar{\Pi}_K\bar{\Gamma}_{N,\lambda}$, is a contraction mapping which guarantees that the following algorithm will converge towards the solution to (4.2),

$$\hat{v}_{N,\lambda}^{(k)} = \bar{\Pi}_K\bar{\Gamma}_{N,\lambda}(\hat{v}_{N,\lambda}^{(k-1)}), \tag{4.11}$$

for $k = 1, 2, ...$, given some initial guess $\hat{v}_{N,\lambda}^{(0)}$. In the leading case of (4.7), this can be expressed as a sequence of least-squares problems that are easily computed: $\hat{v}_{N,\lambda}^{(k)}(z) = \hat{\alpha}_k' B_K(z)$ where

$$\hat{\alpha}_k = \left[\sum_{i=1}^{M} B_K(z_i) B_K(z_i)'\right]^{-1} \sum_{i=1}^{M} B_K(z_i) \bar{\Gamma}_{N,\lambda}(\hat{\alpha}_{k-1}' B_K)(z_i)' \in \mathbb{R}^K,$$

for $k = 1, 2, ...$, given some initial guess $\hat{\alpha}_0$. In the case where $\varepsilon_t = \emptyset$ or when the model is on the form eq. (3.9) with $\varepsilon_t^{(1}$ being extreme-valued distributed and $\varepsilon_t^{(-1)} = \emptyset$,

$$\bar{\Gamma}_{N,\lambda}(\alpha' B_K)(z_i) = G_\lambda\left(u(z) + \beta\alpha' \sum_{j=1}^{N} B_K(Z_j(z_i)) w_{Z,N,j}(z_i)\right),$$

and so $\sum_{j=1}^{N} B_K(Z_j(z_i)) w_{z,N,j}(z_i)$, $i = 1, ..., M$, only need to be computed once and then recycled in each iteration; in contrast, the simulated averages appearing in $\Gamma_N(\hat{\alpha}_{k-1}' B_K)(z_i)$, $i = 1, ..., M$, have to be recomputed in each step of the SA algorithm. Thus, in this special case, it is faster to (approximately) solve for $v_{N,\lambda}$ instead of $V_{N,\lambda}$. While SA is guaranteed to converge globally when $\bar{\Pi}_K\bar{\Gamma}_{N,\lambda}$ is a contraction, the rate of convergence will be slow with the error vanishing at rate β^k,

$$\left\|\hat{v}_{N,\lambda}^{(k)} - v_{N,\lambda}\right\|_\infty \leq \frac{\beta^k(1+\beta)}{1-\beta}\left\|\hat{v}_{N,\lambda}^{(0)} - v_{N,\lambda}\right\|_\infty. \tag{4.12}$$

To speed up convergence, we therefore follow Rust (1988) and combine SA with NK iterations since NK converges with a quadratic rate once a given guess of the value function is close enough to the fixed point. Moreover, in situations where $\bar{\Pi}_K\bar{\Gamma}_{N,\lambda}$ is expansive, NK is still guaranteed to converge locally. Since both the self-approximating and sieve-based methods solve finite-dimensional problems, the NK algorithm for these are equivalent to Newton's method. First consider the sieve-based method where we focus on the least-squares projection as given in (4.7). We are then seeking $\hat{\alpha}$ solving the following K equations,

$$\bar{S}_{N,K}(\alpha) = 0,$$

with

$$\bar{S}_{N,K}(\alpha) = \alpha - \left[\sum_{i=1}^{M} B_K(z_i) B_K(z_i)'\right]^{-1} \sum_{i=1}^{M} B_K(z_i) \bar{\Gamma}_{N,\lambda}(\alpha' B_K)(z_i).$$

The corresponding derivatives of the left-hand side as a function w.r.t. α can be expressed in

terms of the Hadamard differential of $\bar{\Gamma}_{N,\lambda}$ w.r.t. v,

$$\begin{aligned}\nabla\bar{\Gamma}_{N,\lambda}(v)[m](z) &= \beta\sum_{d\in\mathcal{D}}\sum_{j=1}^{N}\dot{G}_{d,\lambda}\left(u(z,\varepsilon_j(z))+\beta\sum_{i=1}^{N}v(Z_i(z))w_{Z,N,i}(z)\right)\\ &\times \left(\sum_{k=1}^{N}dm(Z_k(z,d))w_{Z,N,k}(z,d)\right)w_{\varepsilon,N,j}(z),\end{aligned}$$

where $m:\mathcal{Z}\mapsto\mathbb{R}$ is the direction and

$$\dot{G}_{d,\lambda}(r)=\frac{\partial G_\lambda(r)}{\partial r(d)}=\frac{\exp\left(\frac{r(d)}{\lambda}\right)}{\sum_{d'\in\mathcal{D}}\exp\left(\frac{r(d')}{\lambda}\right)}.$$

The partial derivatives of $\bar{S}_{N,K}(\alpha)$ then becomes

$$\bar{H}_{N,K}(\alpha)=I_K-\left[\sum_{i=1}^{M}B_K(z_i)B_K(z_i)'\right]^{-1}\sum_{i=1}^{M}B_K(z_i)\nabla\bar{\Gamma}_{N,\lambda}(\alpha'B_K)[B_K](z_i)'\in\mathbb{R}^{K\times K}.$$

With these definitions, the NK algorithm takes the form

$$\hat{\alpha}_k=\hat{\alpha}_{k-1}-\bar{H}_{N,K}^{-1}(\hat{\alpha}_{k-1})\bar{S}_{N,K}(\hat{\alpha}_{k-1}).$$

The NK algorithm for the self-approximating method is on the same form, except that we now solve directly for the value function at the N draws. With slight abuse of notation, let $v_N=\{v_{N,\lambda}(Z_k):k=1,...,N\}$ be the vector of integrated values across the set of draws solving $\bar{S}_N(v_N)=0$ where

$$\bar{S}_{N,k}(v_N)=v_{N,k}-\sum_{j=1}^{N}G_\lambda\left(u(Z_k,\varepsilon_j)+\beta\sum_{i=1}^{N}v_{N,i}w_{z,N,i}(Z_k)\right)w_{\varepsilon,N,j}(Z_k), \tag{4.13}$$

for $k=1,...,N$. The corresponding derivatives is $\bar{H}_N(\alpha)=\left(\bar{H}_{N,1}(\alpha),...,\bar{H}_{N,N}(\alpha)\right)'\in\mathbb{R}^{N\times N}$ where, with $\mathbf{1}_N=(1,...,1)'\in\mathbb{R}^N$,

$$\bar{H}_{N,k}(\alpha)=I_N-\nabla\bar{\Gamma}_{N,\lambda}(v_N)[\mathbf{1}_N](z_k)\in\mathbb{R}^N.$$

Finally, we note that the NK algorithm for the expected value function takes a similar form with the functional differential of $\Gamma_{N,\lambda}$ given by

$$\nabla\Gamma_{N,\lambda}(V)[M](z)=\beta\sum_{d\in\mathcal{D}}\sum_{i=1}^{N}\dot{G}_{d,\lambda}(u(S_i(z,d))+\beta V(Z_i(z,d)))M(Z_i(z,d))w_{N,i}(z,d). \tag{4.14}$$

Comparing the NK algorithm for the self-approximating and the sieve-based method, we note that the former involves inverting a $N\times N$-matrix while the latter a $K\times K$-matrix. As pointed out earlier, the self-approximating method generally needs N to be chosen quite large to achieve a precise simulated version of the Bellman operator, in particular in higher dimensions, and so the NK algorithm for this method may become numerically infeasible in some cases. While the

projection-based method also suffers from a curse of dimensionality, since the number of basis functions, K, has to be quite large in higher dimensions to achieve a reasonable approximation, it is less severe and is implementable for higher-dimensional models. If more advanced function approximation methods are employed, even better performance can be achieved.

5 Theory

We here develop an asymptotic theory for the self-approximating and sieve-based methods. We first establish some important properties of the simulated Bellman operators and their exact solutions, $v_{N,\lambda}$ and $V_{N,\lambda}$. These are then used in the asymptotic analysis of the self-approximating solution method and the sieve-based one. This analysis will rely on two general results for estimated solutions to fixed point problems as stated in Theorems A.1 and A.2 in the appendix. The asymptotic analysis will mostly focus on $V_{N,\lambda}$ and $\hat{V}_{N,\lambda}$ since our results for these easily translate into similar results for the approximate integrated value function. For example, $v_{N,\lambda}(z) = M_{N,\lambda,u}(\beta V_{N,\lambda}(z)|z)$, where

$$M_{N,\lambda,u}(r|z) = \sum_{j=1}^{N} G_\lambda\left(u(z,\varepsilon_j(z)) + r\right) w_{\varepsilon,N,j}(z),$$

and so the asymptotic results for $V_{N,\lambda}$ in conjunction with the functional Delta method can be used to obtain similar results for $v_{N,\lambda}$.

Here, and in the following, we let V_λ be the solution to $V_\lambda = \Gamma_\lambda(V_\lambda)$, where

$$\Gamma_\lambda(V)(z) = \int_{\mathcal{Z}}\int_{\mathcal{E}} G_\lambda\left(u(z',e') + \beta V(z')\right) dF_\varepsilon(de|z) dF_z(dz'|z,d)$$

is the smoothed Bellman operator. In particular, V_0 denotes the solution to the non-smoothed version ($\lambda = 0$), $V_0 = \Gamma(V_0)$. Similarly, v_λ denotes the exact solution to the smoothed version, $\bar{\Gamma}_\lambda$, of $\bar{\Gamma}$. Without loss of generality, we assume that the draws can be written as

$$Z_i(z) = \psi_z(z, U_i) \in \mathcal{Z},\ \varepsilon_i(z) = \psi_\varepsilon(z, U_i) \in \mathcal{E}, \tag{5.1}$$

where $U_i \sim F_U$, $i = 1, ..., N$, for some distribution F_U and some functions ψ_z and ψ_ε. We then impose the following regularity conditions on the model and chosen importance sampler:

Assumption 1. *(i) The supports $\mathcal{Z}$ and $\mathcal{E}$ are bounded, convex sets;* $\sup_{(z,e)\in\mathcal{Z}\times\mathcal{E}} |u(z,e)| < \infty$ *and, for any bounded function $b(z,e)$,*

$$\sup_{(z,e)\in\mathcal{Z}\times\mathcal{E}} \int_{\mathcal{Z}}\int_{\mathcal{E}} |b(z,e)|\, dF_\varepsilon(de|z) dF_z(dz'|z,d) < \infty.$$

(ii) $u(z,e)$, $\psi_z(z,u)$, $\psi_\varepsilon(z,u)$, $w_z(z,u)$, $w_\varepsilon(z,u)$ are Lipschitz uniformly in $(z,e) \in \mathcal{Z}\times\mathcal{E}$.

Assumption 2. *The functions $u(z,e)$, $w_z(z'|z)$, $w_\varepsilon(e|z)$, $\psi_z(z,u)$ and $\psi_\varepsilon(z,u)$ are s times continuously differentiable w.r.t. (z',z) for some $s \geq 0$.*

Assumption 1(i) entails that both the exact and simulated Bellman operators map bounded functions into bounded functions and so ensure that the corresponding fixed points are bounded

functions. Many of the results generalize to the case of unbounded state space but then conditions and arguments would have to be re-written. For example, to allow for unbounded $\mathcal{Z}$, the existence of unique fixed points require additional moment conditions and would be based on a weighted sup-norm, see, e.g., Norets, 2010. Similarly, we will employ results from empirical process theory; without the compactness assumption we would need to use bracketing conditions with weighted norms and impose suitable moment conditions, see Section 2.10.4 in van der Vaart and Wellner, 1996. Assumption 1(ii) is used to show uniform convergence of the simulated Bellman operator; it is automatically satisfied when $\mathcal{Z} \times \mathcal{E}$ is finite.

Assumption 2 impose weak smoothness conditions on the model and the chosen samplers. While the focus is on models with continuous state variables our theory also covers models with discrete state space. If the model of interest has discrete state space, all smoothness conditions, incl. Assumption 1, can be ignored. The only point where the discrete and continuous case differ is in terms of the approximation quality of the projection operator: As soon as M is greater than the cardinality of $\mathcal{Z}$,the approximation error will be zero. We first establish existence and uniqueness of the (generally) infeasible simulated solutions and show that they are smooth; the latter property will allow us to control the approximation error due to the use of projections when the state space is continuous.

Theorem 1. *Under Assumption 1, for all $\lambda \geq 0$ and all $N \geq 1$, the operators $(\Gamma_\lambda, \Gamma_{N,\lambda})$ and $\left(\bar{\Gamma}_\lambda, \bar{\Gamma}_{N,\lambda}\right)$ are almost surely contraction mappings on $\mathbb{B}(\mathcal{Z})^D$ and $\mathbb{B}(\mathcal{Z})$, respectively. In particular, the solutions $V_{N,\lambda}$ and $v_{N,\lambda}$ to the simulated Bellman equations (4.1) exist and are unique. If in addition Assumption 2 holds, then $V_\lambda(z)$, , $V_{N,\lambda}(z)$, $v_\lambda(z)$ and $v_{N,\lambda}(z)$ are $s \geq 0$ times continuously differentiable w.r.t. z for any $\lambda > 0$.*

Next, we show that the bias due to smoothing vanishes with rate λ:

Theorem 2. *Under Assumption 1, the following hold: $\|V_\lambda - V_0\|_\infty = O(\lambda)$ and $\|V_{N,\lambda} - V_N\|_\infty = O_P(\lambda)$ for any given $N \geq 1$.*

These two results show that the smoothing can be controlled for by suitable choice of λ both asymptotically ($N = +\infty$) and for any finite number of simulations ($N < \infty$). Also note that these result hold independently of the smoothness properties of the unsmoothed exact and simulated solutions. In the following, we analyze the error due to simulations, $V_{N,\lambda} - V_\lambda$, uniformly in λ. This error analysis will then be combined with Theorem 2 to obtain the full error that also accounts for the smoothing bias.

5.1 Self-approximating method

In this section, we provide an analysis of $V_{N,\lambda}$ allowing for general importance samplers. As a special case, we obtain an asymptotic theory for the self-approximating solution method (where $\Phi_z(z'|z,d)$ is restricted to be marginal distribution). The general results will then in turn be used in the analysis of the corresponding sieve-based methods in the next section. First, we obtain the convergence rates of the simulated version of the expected value function:

Theorem 3. *Suppose that Assumption 1 holds. Then $V_{N,\lambda}$ solving $\Gamma_{N,\lambda}(V_{N,\lambda}) = V_{N,\lambda}$ satisfies $\sup_{\lambda \in [0,\bar{\lambda}]} \|V_{N,\lambda} - V_\lambda\|_\infty = O_p(1/\sqrt{N})$ for any $\bar{\lambda} > 0$. If furthermore Assumption 2 holds with $s \geq 1$, then $\sup_{\lambda \in (0,\bar{\lambda})} \|\partial V_{N,\lambda}/(\partial z) - \partial V_\lambda/(\partial z)\|_\infty = O_P\left(1/\sqrt{N}\right)$.*

The first part of this theorem covers models with both continuous, discrete and deterministic components and is analogous to results in Rust (1997b) and Pal and Stachurski (2013) who also show $\sqrt{N}$-convergence of their value function approximation. The second part is new and utilizes the smoothness of our problem; this requires Assumption 2 and so rules out discrete components. Importantly, the two convergence results hold uniformly over the smoothing parameter λ and so there is no first-order effect from smoothing. In particular, if λ satisfies $\sqrt{N}\lambda \to 0$, then Theorems 2 and 3 yield $\|V_{N,\lambda} - V\|_\infty = O_p(1/\sqrt{N})$. This is similar to convergence of smoothed empirical cdf where the indicator function is replaced by a smoothed version; this also does not affect the convergence rate as long as the smoothing bias controlled for. However, we conjecture that the higher-order derivatives of $V_{N,\lambda}(z)$ will *not* converge with $O_P\left(1/\sqrt{N}\right)$ because these correspond to ill-posed problems.

The above result is then in turn used to derive the asymptotic distribution of $V_{N,\lambda}(z)$ uniformly in $(z,\lambda) \in \mathcal{Z} \times \left(0, \bar{\lambda}\right)$. Here, the smoothing proves important since it allows us to generalise the standard arguments used in the analysis of finite-dimensional extremum estimators to our setting. We can expand the "first-order condition", $V_{N,\lambda} - \Gamma_{N,\lambda}(V_{N,\lambda}) = 0$, around $V_\lambda = \Gamma_\lambda(V_\lambda)$ to obtain, with $\nabla\Gamma_{N,\lambda}$ was defined in (4.14).

$$0 \quad = \quad \Gamma_\lambda(V_\lambda) - \Gamma_{N,\lambda}(V_\lambda) + \{I - \nabla\Gamma_{N,\lambda}(V_\lambda)\}[V_{N,\lambda} - V_\lambda] + o_P\left(1/\sqrt{N}\right),$$

Employing empirical process theory, we show that $\sqrt{N}\{\Gamma_\lambda(V_\lambda) - \Gamma_{N,\lambda}(V_\lambda)\} \rightsquigarrow \mathbb{G}$ in $\mathbb{B}\left(\mathcal{Z} \times \left(0, \bar{\lambda}\right)\right)^D$ for a Gaussian process $\mathbb{G}(z,\lambda)$, while the limit of $dV \mapsto \{I - \nabla\Gamma_{N,\lambda}(V_\lambda)\}[dV]$ has a continuous inverse. We conclude:

Theorem 4. *Suppose that Assumption 1 hold together with either $\mathcal{Z}$ being finite or Assumption2 with $s \geq 1$. Then, $\sqrt{N}\{V_{N,\lambda} - V_\lambda\} \rightsquigarrow \mathbb{G}_V$ on $\mathbb{B}\left(\mathcal{Z} \times \left(0, \bar{\lambda}\right)\right)^D$ where $\mathbb{G}_V(z,\lambda) = \{I - \nabla\Gamma_{N,\lambda}(V_\lambda)\}^{-1}[\mathbb{G}](z)$ is a D-dimensional Gaussian process. Here, $\mathbb{G}$ has covariance kernel*

$$\Omega(z_1,\lambda_1,z_2,\lambda_2) = E_U\left[g_i(z_1,\lambda_1)\, g_i(z_2,\lambda_2)'\right],$$

$$g_i(z,\lambda,d) = \{G_\lambda(u(S_i(z,d)) + \beta V_\lambda(Z_i(z,d)))\, w_i(z,d) - \Gamma_\lambda(V_\lambda)\}\, w_i(z,d),$$

The above result implies, for example, that for any given z, $V_{N,\lambda}(z) \sim N\left(V_\lambda(z), \Omega(z,\lambda,z,\lambda)/N\right)$ asymptotically as $N \to \infty$. Thus, it allows us to construct (pointwise or uniform) confidence bands for the expected value function. We expect that the result will also be useful in analyzing the impact of value function approximation when used in estimation. This could be done by combining the above weak convergence result with, e.g., the results for approximate estimators found in Kristensen and Salanie (2017).

We conjecture that a similar convergence result will hold for the non-smoothed value function approximation. However, the proof of such a result would require different tools and stronger assumptions. In particular, the current proof only requires the following empirical process $(z,\lambda) \mapsto \Gamma_{N,\lambda}(V_\lambda)(z)$ to converge weakly. To allow for non-smooth value function approximation ($\lambda = 0$), our proof would have to be modified and we expect we would now require that the empirical process $(V,z) \mapsto \Gamma_N(V)(z)$ converges weakly. For this to hold, we would need to be able to find a suitable function set that the estimated non-smooth solution, V_N, would be

situated in and then verify that the entropy of this function set is well-behaved. Standard choices of function sets are smooth classes, but V_N is non-smooth and so the proof would be much more delicate.

Finally, for a complete analysis that takes into account the smoothing bias, state the following corollary to Theorem 4: For any $\lambda = \lambda_N \to 0$ such that $\lambda\sqrt{N} \to 0$,

$$\sqrt{N}\{V_{N,\lambda} - V_0\} \rightsquigarrow \mathbb{G}_V.$$

5.2 Sieve-based approximation of value functions

We here analyze the asymptotic properties of the sieve-based approximate value function, $\hat{V}_{N,\lambda}$. To this end, we use the following decomposition of the over-all error,

$$\hat{V}_{N,\lambda} - V_\lambda = \left\{\hat{V}_{N,\lambda} - V_{N,\lambda}\right\} + \left\{V_{N,\lambda} - V_\lambda\right\},$$

where the second term converges weakly towards a Gaussian process, c.f. Theorem 4. What remains is to control the first term which is due to the sieve approximation; this is done by imposing the following high-level assumption on the projection operator when applied to the function set $\mathcal{V}_{s,r} = \bar{\mathcal{V}}_{s,r}^D$ where $\bar{\mathcal{V}}_{s,r}$ was defined in (4.8):

Assumption 3. *The projection operator Π_K satisfies $\sup_{V \in \mathcal{V}_{s,r}} \|\Pi_K(V) - V\|_\infty = O_P(\rho_K)$, for some sequence $\rho_K \to 0$, where $s \geq 0$ was given in Assumption 2 and $r \geq \|V_\lambda\|_{s,\infty}$.*

This is a high-level condition that requires the chosen function approximation method to have a uniform error rate over the function class $\mathcal{V}_{s,r}$. It allows for most known function approximations. As a particular example, we showed in Section 4.3 that $\rho_K = \log(K)/K^{(s+1)/d}$ for any value of $r < \infty$ when polynomials are employed. Compared to results on sieve approximations of value functions found elsewhere in the literature, ours is weaker since we are here allowed to restrict attention to the smooth function class $\mathcal{V}_{s,r}$. In contrast, sieve-based approximations developed in other papers, such as Munos and Szepesvari (2008) and Pal and Stachurski (2013), will tend suffer from bigger biases since the underlying value function being approximated is at most Lipschitz. In the case of $\mathcal{Z}$ being discrete, we have $\sup_{V \in \mathcal{V}_{s,r}} \|\Pi_K(V) - V\|_\infty = 0$ for $K > |\mathcal{Z}|$ under great generality and so there will be no asymptotic bias component due to sieve approximations in this case.

Theorem A.1 together with the fact that $\Gamma_{N,\lambda}(V_\lambda) - \Gamma_\lambda(V_\lambda) = O_P\left(1/\sqrt{N}\right)$, c.f. Proof of Theorem 3, now yield the following result:

Theorem 5. *Suppose that Assumptions 1 and 3 hold. Then $\hat{V}_{N,\lambda}$, defined as the solution to $\Pi_K \Gamma_{N,\lambda}(\hat{V}_{N,\lambda}) = \hat{V}_{N,\lambda}$, satisfies $\sup_{\lambda \in (0,\bar{\lambda})} \|\hat{V}_{N,\lambda} - V_\lambda\|_\infty = O_p(1/\sqrt{N}) + O_P(\rho_K)$. Suppose in addition that either $\mathcal{Z}$ is finite or Assumption 2 holds with $s \geq 1$. Then, if $\sqrt{N}\rho_K \to 0$,*

$$\sqrt{N}\{\hat{V}_{N,\lambda} - V_\lambda\} = \sqrt{N}\{V_{N,\lambda} - V_\lambda\} + o_P(1) \rightsquigarrow \mathbb{G}_V.$$

The discussions following Theorems 3 and 4 carry over to the above result. In particular, the rate result still goes through when no smoothing is employed ($\lambda = 0$) but the current proof of the asymptotic distribution result requires smoothing ($\lambda > 0$). Compared to the rate

results for $V_{N,\lambda}$, the projection-based method suffers from an additional error due to the sieve approximation, $O_P(\rho_K)$. This can be interpreted as a bias term, while $O_p(1/\sqrt{N})$ is its variance component which is shared with $V_{N,\lambda}$. The requirement that $\sqrt{N}\rho_K \to 0$ is used to kill the sieve bias term so that $\hat{V}_{N,\lambda}$ is centered around V_λ.

The above result provides a refinement over existing results where a precise rate for the bias is not available; see, e.g., Lemma 5.2 in Pal and Stachurski (2013). This result also shows that there is an inherent computational curse-of-dimensionality built into our projection-based value function approximation when polynomial interpolation is employed: In high-dimensional models, a large number of basis functions are needed which in turn increases the computational effort. In the case of polynomial approximations, the rate condition becomes $\sqrt{N}\log(K)/K^{(s+1)/d} \to 0$ and so, as d increases, we need K to increase faster with N to kill the sieve bias component. However, also note that K as no first-order effect on the variance and so there is no bias-variance trade-off present. In particular, we can let K increase with N as fast as we wish and so our procedure should in principle also work for models with high-dimensional state space. That is, our method does not suffer from any statistical curse-of-dimensionality. However, this requires choosing both N and K large in order to control variance and bias which will increase computation time.

6 Numerical results

In this section we examine the numerical performance of the proposed solution algorithms with focus on how the theoretical results derived in the previous sections translate into practice and how different features of model and implementation affect their performances.

We focus exclusively on approximating $v_0(z)$, and measure the performance of a given approximate solution, say, $\tilde{v}(z)$ in terms of its pointwise bias, variance and mean-square error (MSE) defined as $Bias(z) := E[\tilde{v}(z)] - v_0(z)$, $Var(z) := Var(\tilde{v}(z)) = E\left[(\tilde{v}(z) - E[\tilde{v}(z)])^2\right]$ and $MSE(z) = Bias(z) + Var(z)$, respectively. As overall measures we use uniform bias, variance and MSE, $\|Bias\|_\infty = \sup_z |Bias(z)|$, $\|Var\|_\infty = \sup_z |Var(z)|$ and $\|MSE\|_\infty = \sup_z |MSE(z)|$. Given that the exact solution $v_0(z)$ is unknown, we replace this by a very finely approximated solution computed as the average over 100 sieve approximations, where each approximate solution used $K = 500$ Chebyshev polynomials and $N = 2000$ pseudo-random draws. Each approximate solution was computed by successive approximation until a contraction tolerance of machine precision was reached. The uniform standard deviation across the 100 approximations were less than 0.017 which is tiny compared of the overall range of the integrated value function. We then approximate the point-wise bias and variance of a given method through $S \geq 1$ independent replications it: Let $\tilde{v}_1(z), ..., \tilde{v}_S(z)$ be the solutions obtained across the S replications. We then approximate the mean by $\hat{E}[\tilde{v}(z)] = \frac{1}{S}\sum_{s=1}^{S}\tilde{v}_s(z)$ which in turn is used to obtain the following bias and variance estimates, $\hat{Bias}(z) = \hat{E}[\tilde{v}(z)] - v_0(z)$, and $\hat{Var}(z) = \frac{1}{S}\sum_{s=1}^{S}(\tilde{v}_s(z) - \hat{E}[\tilde{v}(z)])^2$.

To implement the sieve-based method, we need to choose the sieve space used in constructing

Π_K. We here focus on Chebyshev basis functions or B-Splines as discussed in Section 4.3[1].

6.1 A model of optimal replacement

To provide a test bed for comparison of the sieve-based approximation method, we use the well-known engine replacement model by Rust (1987). Rust's model has become the basic framework for modeling dynamic discrete-choice problems and has been extensively used in other studies to evaluate the performance of alternative solution algorithms and estimators. While the model and its solution is well described in many papers, for completeness we briefly describe our variation of it below.

We consider the optimal replacement of a durable asset (such as a bus engine) whose controlled state $Z_t \in \mathbb{R}_+$ is summarized by the accumulated utilization or mileage since last replacement. In each period, the decision maker faces the binary decision $d_t \in \mathcal{D} = \{0, 1\}$ whether to keep ($d_t = 0$) or replace ($d_t = 1$) the durable asset with a fixed replacement cost. If the asset is replaced, accumulated usage regenerates to zero. We assume that the per period change in usage (in absence of the replacement decision) is a mixture of a discrete distribution with a probability mass $\pi > 0$ at zero and a linearly transformed Beta distribution with shape parameters a and b and scale parameter $\sigma_\varepsilon > 0$. Thus,

$$F_Z\left(z'|z,d\right) = \pi I\left\{z' = z\right\} + (1-\pi)F_+\left(z'|z,d\right), \tag{6.1}$$

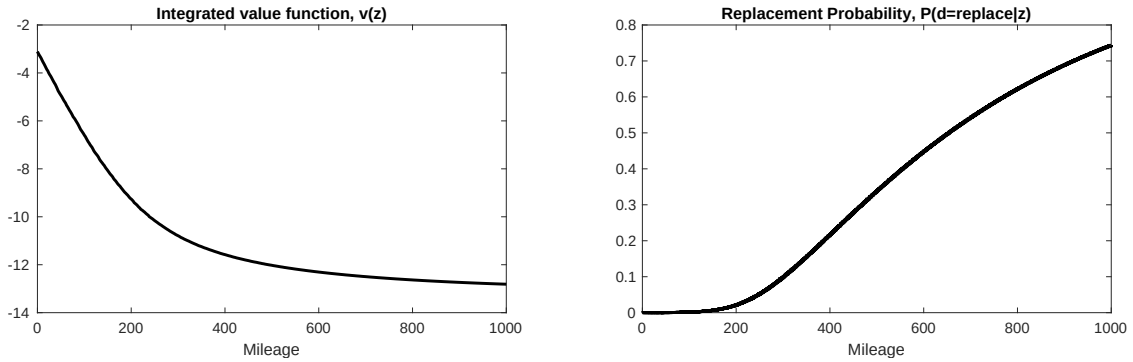
where $F_+\left(z'|z,d\right)$ has density $f_+\left(z'|z,d\right) = \frac{1}{\sigma_\varepsilon} f_\beta\left((z'-z)/\sigma_\varepsilon; a, b\right)$, $\pi > 0$ is the probability of no usage and $f_\beta(x; a, b)$ is the probability density function of the Beta distribution with shape parameters a, b.

We have chosen the scaled Beta distribution because of its flexibility and because it has bounded support $(0, \sigma_\varepsilon)$, i.e. $f_+\left(z'|z,d\right) = 0$ for $z' < z$ or $z' - z > \sigma_\varepsilon$. This is in line with the discretized model in the original formulation in Rust (1987) where monthly mileage were only allowed to take a few discrete values and monthly mileage is naturally bounded above and below (busses never drives backwards and there are limits how far a bus can drive within a month). We introduce probability mass π at $z' = z$ to allow for the possibility that the asset is not used in a given period and thereby can end in the same state with positive probability when $\pi > 0$. As explained below, this feature turns out to be quite important for the applicability of the self-approximating method of Rust (1997).

Let the expected per period operating costs be linear in usage, $c(z) = \theta_c \cdot 0.001 \cdot z$, and let $RC > 0$ denote the replacement cost if installing a new asset. The state and decision dependent per period utility is then given by $u(Z_t, d_t) + \sigma_\varepsilon \varepsilon_t(d_t)$ where $u(z,d) = (RC + c(0))\, I\left\{d = 0\right\} + c\left(z\right) I\left\{d = 1\right\}$ and the utility shocks $\varepsilon_t = (\varepsilon_t(0), \varepsilon_t(1))$ are i.i.d. extreme value and fully independent of Z_t. This specification is a special case of Example 1 with $\lambda = 1$ and the simulated Bellman operator takes the form (3.5) where $G_{\lambda=1}(\cdot)$ appears as part of the model. Thus, there is no smoothing bias present in the baseline model. In Section 6.5, we then investigate the effect of smoothing; this will be done by pretending that we are not able to integrate out ε_t analytically

[1] For more details on the implementation, see appendix D .

Figure 1: Fine Approximation as "Exact" Solution



Notes: Discount factor is $\beta = 0.95$, utility function parameters are $\theta_c = 2$, $RC = 1$, $\lambda = 1$ and transition parameters are $\sigma_\varepsilon = 15$, $a = 2$, $b = 5$ and $\pi = 0.000000001$.

in the baseline model and instead we simulate both Z_t and ε_t and then include our smoothing device in the computation of the simulated Bellman operator.

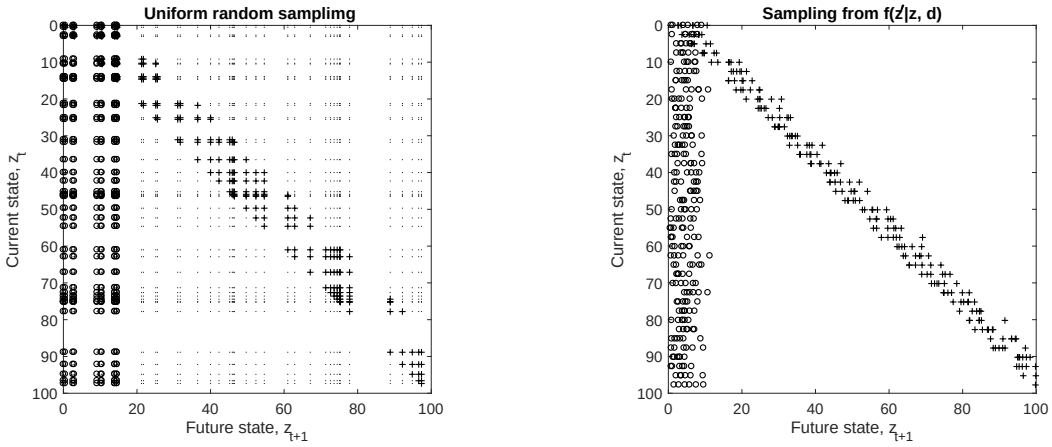
In the numerical illustrations below we use the following set of benchmark parameter values unless otherwise specified: We set replacement cost to $RC = 10$ and the cost function parameter to $\theta_c = 2$ so that RC is 5 times as large as $c(1000)$. This implies a large variation in the probability of replacement over Z_t compared to Rust (1987) and a more curved value function. The parameters indexing the transition density $f_+(z'|z,d)$ are set to $\sigma = 15$, $a = 2$, $b = 5$ and $\pi = 10^{-10}$. This implies a quite sparse transition density, which is similar to the fitted model in Rust (1987). Note that the support of Z_t is unbounded (the positive half line) and therefore the theory does not apply directly, since we throughout assumed bounded support. However, we expect that the theory extends to the unbounded case after suitable modifications, c.f. discussion following Assumptions 1-2. We set $\beta = 0.95$ which is quite small relative to Rust (1987) where $\beta = 0.9999$. The main reason for choosing a smaller discount factor is to avoid overly long computation times for the implementation of successive approximation algorithm whose convergence rate crucially depends on β. c.f. eq. (4.12).

In Figure 1 we plot the corresponding "exact" solution as described earlier. Importantly, since the transition density is an analytic function the value function is also analytic and so well-approximated by polynomial interpolation methods.

6.2 Numerical implementation of simulated Bellman operators

The simulated Bellman operators in (3.7) and (3.8) require the user to choose an importance sampling distribution. For the self-approximating solution method we need to choose a marginal sampler, $d\Phi_Z(z'|z,d) = \phi_Z(z')\,dz'$. We follow Rust (1997b) and choose $\phi_Z(z') = I\{0 < z' < z^{\max}\}$ as a uniform density with support support $[0, z^{\max}]$ for some truncation point $0 < z^{\max} < \infty$ chosen by us. First note that this entails that the simulated Bellman operator used for the self-approximating value function is biased since we do not sample from the full support $\mathcal{Z} = \mathbb{R}_+$; however, this bias can be controlled by choosing $z^{\max}$ large enough. We will explain below why we do not choose $\phi_Z(z')$ as a density with support $\mathbb{R}_+$. Using a uniform sampler, the corresponding Radon-Nikodym derivative takes the form

Figure 2: Random Grids



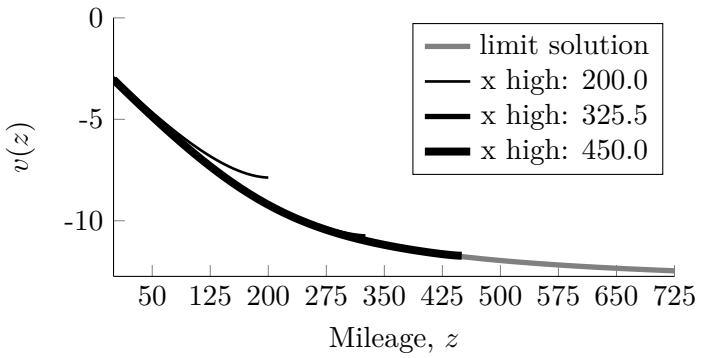
Notes: In the left panel we present the grids used for the self-approximate random Bellman operator. We have uniformly sampled a random grid, $\{Z_1, ..., Z_N\}$ on the interval $[0; 1000]$ with N =400. Dots (.) mark sampled grid points in R^2: $Z_N \times Z_N$, plus (+) mark grid points where $f(z_j|z_i, d = 0) > 0$ and circles (o) mark points where $f(z_j|z_i, d = 1) > 0$. In the right panel, we plot the grid the projected random Bellman operator, where we have sampled directly from the conditional transition density in each of the $M = 400$ uniformly spaced evaluation points. To have equally many grid-points with non-zero transition density we only need $N = 400 * \sigma_\varepsilon / \max(Z_N) = 9$ random grids for each of the $M = 400$ evaluation points. Both figures show only a subset of the state space, $(z, z') \in [0; 100]^2$. Parameters are $\sigma_\varepsilon = 15$, $a = 2$, $b = 5$ and $\pi = 0.0000000001$.

$w_Z(z'|z, d) = \pi\delta(z' - z) + (1 - \pi)f_+(z'|z, d)$ where $\delta(\cdot)$ denotes Dirac's delta function. We approximate this by $\hat{w}_Z(z'|z, d) = \pi I\{z' = z\} + (1 - \pi)f_+(z'|z, d)$ which entails another small approximation error. For the sieve-based version, we simply choose $\Phi_Z(z'|z, d) = F_Z(z'|z, d)$ and so $w_Z(z'|z, d) = 1$.

As explained in Section 4.3, using a marginal importance sampler creates issues since it fails to adapt to the particular shape of the support of $F_Z(z'|z, d)$. In particular, for a given choice of z, many of the draws from $\phi(z')$ will tend to fall outside the support of $f_Z(z'|z, d)$ and so will not contribute. In contrast, when $\phi_Z(z'|z, d) = f_Z(z'|z, d)$, the draws from ϕ_Z will by construction fall within the support of $f_Z(z'|z, d)$. This can be seen in Figure 2 where we have plotted the random draws obtained from the two different importance samplers used for the sieve-based and self-approximating solutions together with the actual support of $f_Z(z'|z, d)$. In the left-hand side panel we have plotted pairs of the uniform draws, (Z_i, Z_j) for $i, j = 1, ..., N$, used for Rust's self-approximating method with $N = 400$ and $z^{max} = 1,000$, while in the right-hand side we have plotted $(z_i, Z_j(z_i, d))$ where z_i are uniform draws and $Z_j(z, d) \sim f_Z(\cdot|z, d)$. In both cases, we have marked the pairs for which the corresponding density, $f_Z(Z_j|Z_i, d)$ and $f_Z(Z_j(z_i, d)|z_i, d)$, respectively, is positive. Clearly, the use of a marginal importance sampling density leads to very poor coverage of the actual support of $f_Z(z'|z, d)$ as z varies while by construction $\Phi_Z(z'|z, d) = F_Z(z'|z, d)$ does an excellent job. This translates into the former simulated Bellman operator exhibiting much larger variance compared to the latter.

This issue is further amplified when we introduce the normalization given in eq. (3.3): Suppose that we had not included a discrete component $\pi I\{z' = z\}$ in the model. Then, with $Z_i \sim U[0, z^{\max}]$, $w_{N,Z,i}(Z_j, d) = f_+(Z_i|Z_j, d) / \sum_{k=1}^N f_+(Z_k|Z_j, d)$. Since $f_+(z'|z, d)$ has bounded support, it often happens that $\sum_{k=1}^N f_+(Z_k|Z_j, d) = 0$ for even large values of N and so the

Figure 3: Truncation bias due to z^{max} being too low.



simulated Bellman operator is not even well-defined. This issue will vanish as $N \to \infty$, but this on the other hand increases the computational burden since the self–approximating method require us to solve for the value function at the N draws. Introducing the discrete component in the model resolves this issue since now $w_{Z,N,i}(Z_j, d) = \hat{w}_Z(Z_i|Z_j, d) / \sum_{k=1}^{N} \hat{w}_Z(Z_k|Z_j, d)$, where $\sum_{k=1}^{N} \hat{w}_Z(Z_k|Z_j, d) > 0$ for all $j = 1, ..., N$ by construction. Thus, $\pi > 0$ functions as a regularization device.

This brings us to the reason why we do not choose $\phi_Z(z')$ as a density with unbounded support to avoid the issue of truncation. In our initial experimentation, we did try out sampling from distributions with unbounded support, but the above numerical issues were even more severe since the draws become even more spread out in this case. So we instead choose $\phi_Z(z')$ to have bounded support. Figure 3 show how the solution depends on $z^{\max}$. The effect of the truncation $z^{\max}$ will be model specific and in practice experimentation is required. If we, for example, simply set $z^{\max} = 1,000,000$, the variance of the simulated Bellman operator becomes very large for a given N due to the issue with undefined sample weights $w_{N,i}(z, d)$ mentioned above. At the same time, choosing $z^{\max}$ too small leads to a large bias. To balance the bias and variance, we ended up using $z^{\max} = 1000$ which all subsequent numerical results for the self-approximating method is based on. Finally, we would like to stress that none of these issues appear for the sieve-based method.

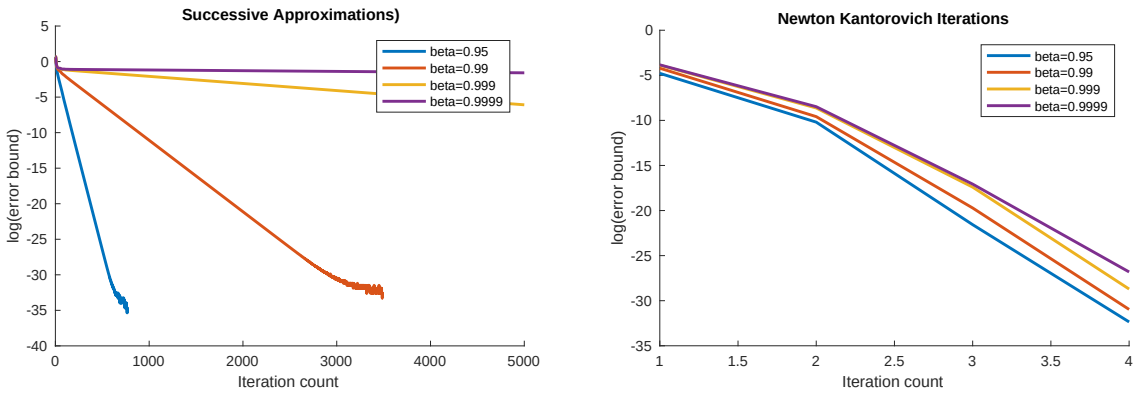
6.3 Convergence properties and computation times

We first investigate the convergence properties of our solution methods: Do the they converge and if so how fast?

Global convergence properties of sieve method

As demonstrated in Theorem 1, the simulated Bellman operators are always contraction mappings and so the self-approximating method is guaranteed to converge using successive approximations. In contrast, $\Pi_K \bar{\Gamma}_{N,\lambda}$ is not necessarily a contraction and so global convergence of the sieve method may fail, c.f. discussion in Section 5.2. A sufficient condition for global convergence is $||\Pi_K||_{op,\infty} < 1/\beta$ and we saw that $||\mathbf{P}_K||_{op,\infty} > 1$ implies $||\Pi_K||_{op,\infty} > 1$. However, even if $||\mathbf{P}_K||_{op,\infty} > 1$, successive approximation may still converge: Across various parameter

Figure 4: Convergence and discount factor



Notes: Discount factor is $\beta \in \{0.95, 0.99, 0.999, 0.9999\}$, utility function parameters are $\theta_c = 2$, $RC = 1$, $\lambda = 1$ and transition parameters are $\sigma_\varepsilon = 15$, $a = 2$, $b = 5$ and $\pi = 0.000000001$.

values of model, choices of sieve spaces and number of simulations, we did not encounter any failure of the sieve method to converge and the resulting approximate solution was well-behaved. This finding held across various initializations of the solution algorithms (initial choice of sieve coefficients). For example, we implemented the sieve method using $M = 64$ evaluation points and using either $K = 1$ or $K = 4$ Chebyshev basis functions. We found that $||\mathbf{P}_1||_{op,\infty} = 1$ while $||\mathbf{P}_4||_{op,\infty} > 1.78$ and so the sieve method was guaranteed to converge for $K = 1$ but not for $K = 4$. Nevertheless, the method of successive approximations did in fact converge to a tolerance of 10^{-12} for both $K = 1$ and $K = 4$.

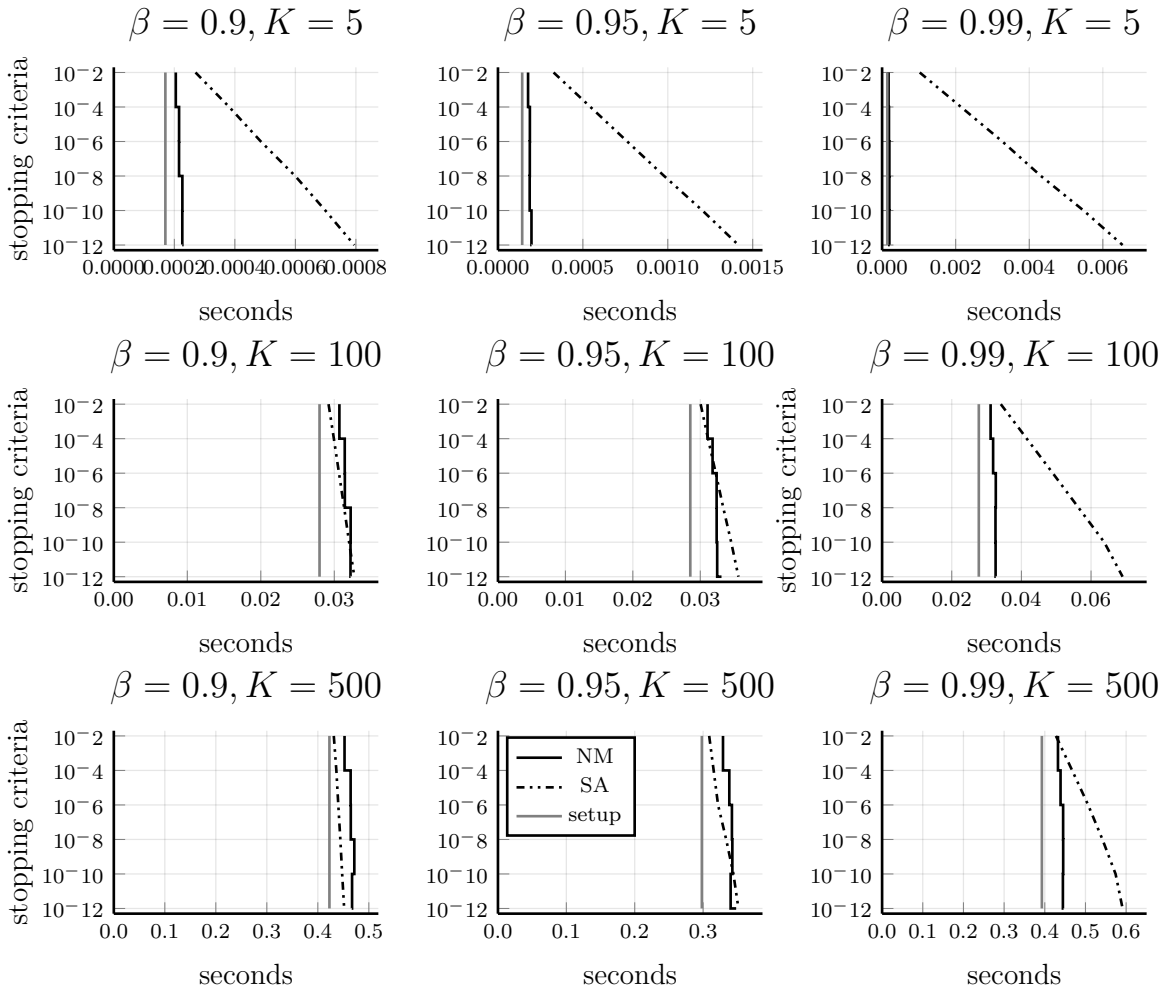
Successive approximation versus Newton-Kantorovich

In Section 4.3 we advocate using a hybrid of successive approximation (SA) and Newton-Kantorovich (NK) where we start with SA to ensure global convergence, and switch to NK iterations once the domain of attraction has been reached since NK generally converges faster. We illustrate this attractive feature of the NK algorighm in Figure 4 where we have plotted the log residual error of the current value function approximation (relative to the "exact" solution) against the iteration count for the SA and NK algorithms, respectively, for four different values of β. As expected, the convergence of the SA algorithm requires a very large number of iterations (> 1000) with computation time increasing in β, where as NK converges after less than 10 iterations and with the value of β having little effect on its performance.

Figure 4 is silent about the over-all computation time of SA relative to NK. Compared to SA, each NK iteration is more expensive since the former only requires computing the simulated Bellman operator evaluated at the value function obtained in the previous step while the latter, in addition, requires computing its functional derivative and inverting a $K \times K$ dimensional matrix for the integrated value function and a $KD \times KD$ dimensional matrix for the expected value function, c.f. Section 4.3. With K large, one could therefore fear that NK would become computationally too expensive.

In Figure 5 we report best of 10 run-times for various levels of K and β and tolerance levels of SA and NK where we also include set-up time (time spent on initial computations before

Figure 5: Run-times (incl setup times) for SA (dotted lines) and NK (drawn lines) algorithms.



starting the actual algorithm). As expected, we find that NK is the faster of the two algorithms when β is relatively large and K is relatively small. With $K = 5$ NK is faster across all levels of β while for $K = 100$ and $K = 500$, SA is faster for moderate values of β. However, as we shall subsequently see, with $K = 5$ the sieve method carries almost no bias and so choosing K larger (such as 100 or 500) is actually unnecessary here and is only included here to illustrate potential issues with NK for models where a large number of sieve terms are needed to obtain a good approximation of the value function. Moreover, in most empirical applications, β is chosen to be larger than 0.99 in which case NK still dominates SA even with $K = 500$.

6.4 Approximation quality

We will now look at how the approximate value function is affected by the number of draws and the chosen projection basis. The goal is to demonstrate the rate results of the theoretical sections, and to compare the two types of basis functions spaces that we described above. We will take a partial approach and first fix N to study the role of K, and then afterwards fix K to study the role of N.

Effect of varying K for projection-based value function approximation

The theory for the projection-based value function approximation informs us that the choice of the basis functions will have a first-order effect on the bias while only a second-order effect on the variance. In particular, we expect $Bias\left(z\right)$, as defined in the beginning of this section, to satisfy $Bias\left(z\right) \cong \Pi_K\left(v_0\right)\left(z\right) - v_0\left(z\right)$, c.f. discussion following Theorem 5, while $Var\left(z\right)$ should be much less affected by K. The actual size of the bias obviously depends on the curvature and smoothness of v_0 and the particular choice of basis functions. But we know that v_0 is an analytic function and with only moderate curvature, c.f. Figure 1 and so expect it to be well-approximated by a small number of polynomial basis functions. This is confirmed by the pointwise bias and standard deviation, $\sqrt{Var\left(z\right)}$, reported in Figure 6:

First, as can be seen in the left-hand side panel of Figure 6, first-order B-splines lead to significantly more point-wise bias than the other two sieve bases, namely second-order B-splines and Chebyshev polynomials. This is accordance with theory since we know that a smooth function is better approximated by higher-order polynomials, c.f. the error rates reported in Example 1 as a function of s. At the same time, second-order B-splines and Chebyshev polynomials exhibit very similar biases for a given K.

The right-hand side panel of Figure 6 shows the point-wise standard deviation across different choices of K for the three different sieve bases. Consistent with the theory, the standard deviation of the value function approximation is not very sensitive to the particular choice of the sieve basis and the number of basis functions uses. That is, the sieve basis mostly affect the bias with only minor impact on the variance.

Finally, we examine how the bias behaves as we further increase K. Figure 7 plots $\left\|Bias\right\|_\infty$ as a function of K. Similar to Figure 6 we see much more rapid convergence when smooth basis functions are used, and with little improvement for K greater than 9. This is not surprising given the reported shape of v_0. The second-order B-splines and Chebyshev basis functions produce very similar fits, even if they are evaluated on different grids and the B-splines have very different properties compared to Chebyshev polynomials. Indeed, the curves are practically overlapping. This is in accordance with the asymptotic theory that predicts that higher-order B-splines and Chebyshev polynomials should lead to similar biases. Moreover, the theory informs us that if v_0 is analytic, and this is the case in this particular implementation, we should expect the bias to vanish with rate $O(K^{-K})$ when using polynomial interpolation. The bias indeed does go to zero very quickly and so the numerical results support the theory.

Simulation errors, rates of convergence and asymptotic normality

We now compare the errors due to simulations and the rates with which these vanish for the two solution methods. For both methods, theory tells us that N should have a first-order effect on the variance of the approximate value function which is supposed to vanish at rate $1/N$, c.f. Theorems 5 and 3. Our asymptotic theory is, on the other hand, silent about the size of simulation bias and the rate with which it should vanish with. However, we can think of both the sieve-based and self-approximating method as a nonlinear GMM-estimator where the simulated Bellman operator defines the sample moments. Importing results for GMM estimators, see, e.g., Newey and Smith (2004), we should expect the simulation bias to be of order $1/N$.

Figure 6: Point-wise bias and standard deviation of solutions for various choices of K using different interpolation schemes, $N = 200$, $S = 200$, $\sigma_\varepsilon = 15$.

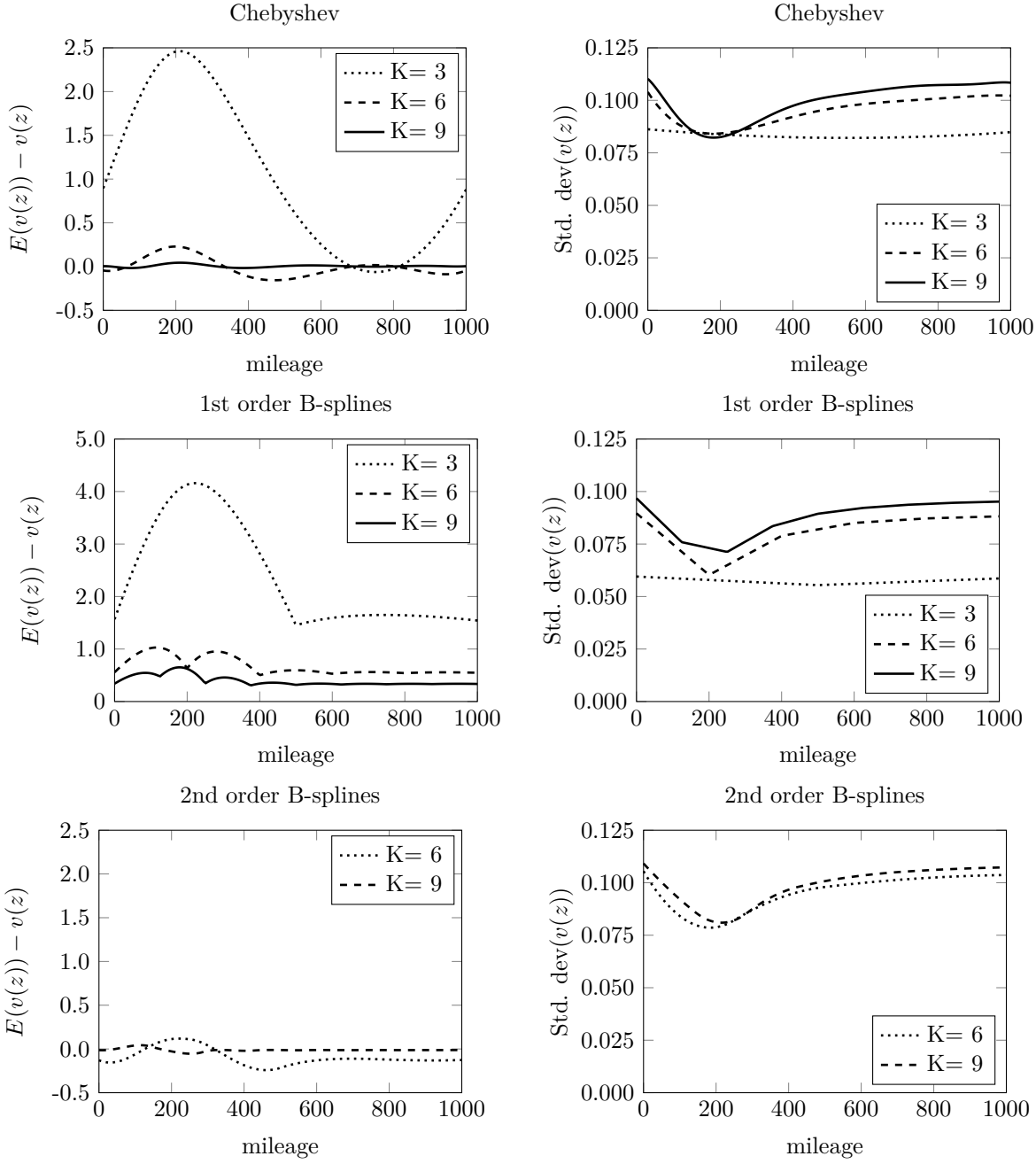
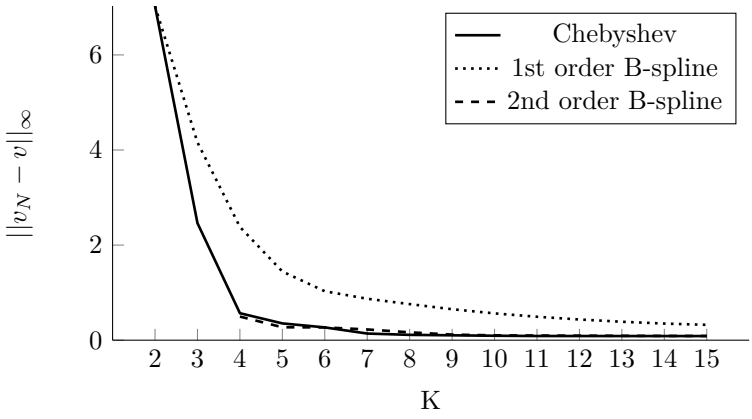


Figure 7: Sup-norm of bias of solutions for various choices of K using various interpolation schemes, $N = 200$, $S = 200$.

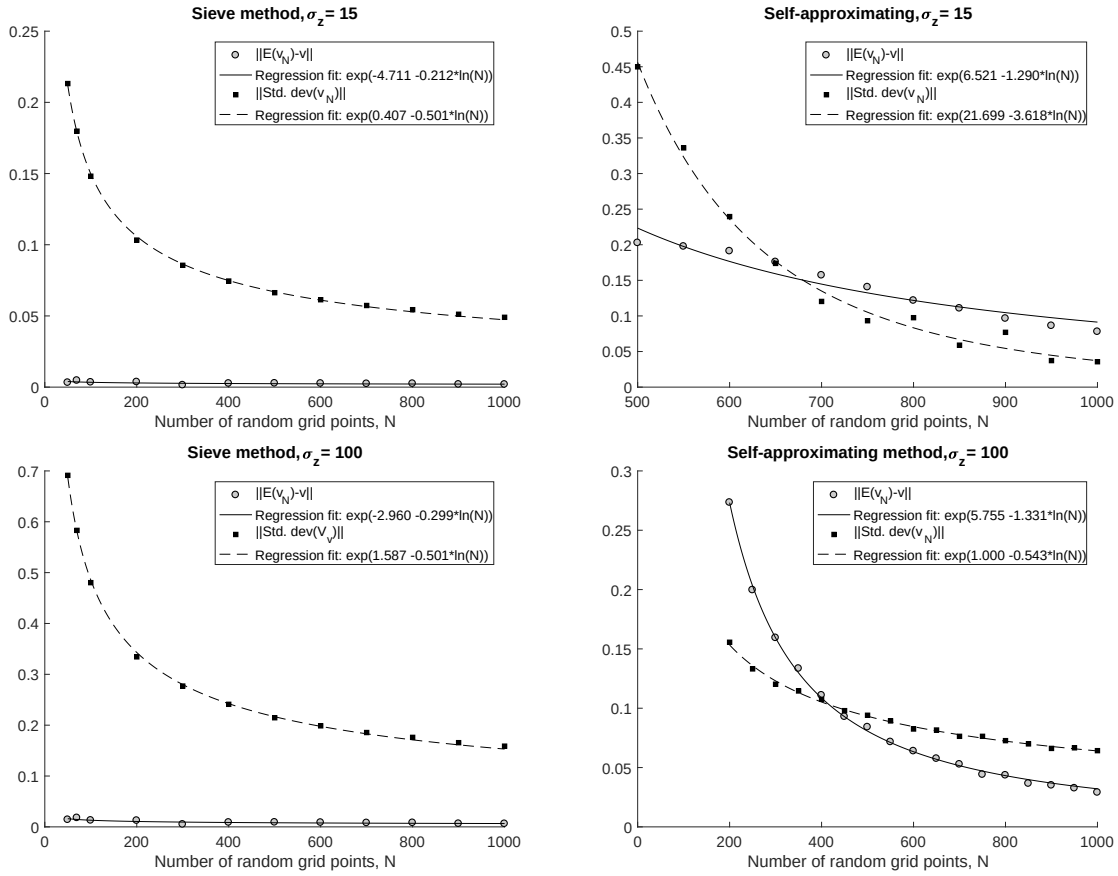


In Figure 8 we investigate this prediction by plotting $\|Bias\|_\infty$ and $\left\|\sqrt{Var}\right\|_\infty$ for the sieve-based method (left panels) and for the self-approximating method (right panels) for two different choices of σ_ε and for across different values of N. To examine the rate with which the simulation bias and variance vanish we estimate the following an exponential regressions by NLS $||\sqrt{Var}||_\infty = \exp(\alpha_{SD} + \rho_{SD} \ln(N))$ and $||Bias||_\infty = \exp(\alpha_{Bias} + \rho_{Bias} \ln(N))$ where $N^{\rho_{SD}}$ and $N^{\rho_{Bias}}$ measures the rate for $||\sqrt{Var}||_\infty$ and $||Bias||_\infty$ respectively. The resulting regression fit estimates are reported in both Figure 8 as well as in Table 1. In Table 1 we present bias and standard deviation for $N = 500$ as well as their rates of convergence both methods; with various values of K for the sieve approximation method.

According to the theory, the variance should vanish with rate $1/N$ for both methods and we therefore expect $\rho_{SD} = -0.5$ so that $||\sqrt{Var}||_\infty$ vanish with $1/\sqrt{N}$. For the projection based method, we see that the rate with which the standard deviation shrinks to zero is indeed close to -0.5 for all values of $K > 1$ and irrespectively of the value of σ_ε. For the self-approximating method we estimate the rate to $1/N^{0.541}$ when $\sigma_\varepsilon = 100$, which is in line with the theory. However, $||\sqrt{Var}||_\infty$ is found to vanish with rate $1/N^{3.6}$ for $\sigma_\varepsilon = 15$. This seems to indicate that the asymptotic theory developed in Theorems 3 and 4 do not provide a very accurate approximation of the performance of the self-approximating method for small and moderate choices of N when the support of $Z_t|Z_{t-1} = z, d_t = 1$ is small ($\sigma_\varepsilon = 15$). We conjecture that the discrepancy between theoretical predictions and numerical results for the self-approximating method is due to the aforementioned issues with the marginal importance sampler discussed in Section 6.2: Many of the draws are not used in the computation of the simulated Bellman operator because they fall outside the support of $Z_t|Z_{t-1} = z$ for a given choice of z. Thus, the effective number of draws is smaller than N and changes as z varies.

For the projection based method, the main source of bias is due to the sieve projection. From Figure 6, we see that, with $N = 200$ and $K = 9$, the sieve-based methods using second-order B-splines or Chebyshev polynomials have virtually no bias, and both Figure 8 as well as in Table 1 also confirms that we practically eliminate by approximate the value function using Chebychev polynomials with $K = 20$. However, there still remains a small simulation bias for that decays with N. For small K, we see that the bias is roughly independent of N. As K increases so

Figure 8: Convergence results



Notes: Discount factor is $\beta = 0.95$, utility function parameters are $\theta_c = 2$, $RC = 10$, $\sigma_\varepsilon = 1$ and transition parameters are $a = 2$, $b = 5$ and $\pi = 0.000000001$. Uniform bias and variance were estimated using 500 evaluation points and $S = 2000$ implementations.

Table 1: Bias, variance, and rates of convergence for various values of K

		Sieve Method				Self-approx.
# of basis functions, K	1	2	5	10	15	method
				σ_z=15		
$\|\|Bias\|\|_\infty$ for $N = 500$	12.743	7.029	0.348	0.016	0.003	0.203
$\|\|\sqrt{Var}\|\|_\infty$ for $N = 500$	0.000	0.020	0.063	0.066	0.066	0.450
Convergence rate for $\|\|Bias\|\|_\infty$	0.000	0.000	0.002	-0.012	-0.163	-1.290
Convergence rate for $\|\|\sqrt{Var}\|\|_\infty$	0.169	-0.500	-0.501	-0.501	-0.501	-3.618
				σ_z=100		
$\|\|Bias\|\|_\infty$ for $N = 500$	22.446	10.937	0.112	0.009	0.009	0.084
$\|\|\sqrt{Var}\|\|_\infty$ for $N = 500$	0.000	0.128	0.218	0.215	0.215	0.094
Convergence rate for $\|\|Bias\|\|_\infty$	0.000	0.000	-0.027	-0.299	-0.300	-1.331
Convergence rate for $\|\|\sqrt{Var}\|\|_\infty$	0.169	-0.500	-0.501	-0.501	-0.501	-0.543

does the dependence on N. However, even for $K = 20$ where we estimate ρ_{Bias} to be 0.21 and 0.30 for $\sigma_\varepsilon = 15$ and $\sigma_\varepsilon = 100$ respectively, the rate of convergence is far from $1/N$. This is probably due to the presence of higher-order bias components that our asymptotic theory does not account for or because there is still some remaining sieve approximation bias left even with $K = 20$.

For the self-approximating method, there is no sieve projection bias but a larger simulation induced bias that decreases with N. We obtain rate estimates of $1/N^{1.7}$ and $1/N^{1.4}$ for the bias when $\sigma_\varepsilon = 15$ and $\sigma_\varepsilon = 100$ respectively; these are slightly faster than expected but not too far from the theoretical predictions of $1/N$. For the self-approximating method, bias constitute more than half of RMSE when $N < 600$ for $\sigma_\varepsilon = 15$ (or $N < 400$ for $\sigma_\varepsilon = 100$), but since $||Bias||_\infty$ decays faster than $||\sqrt{Var}||_\infty$, the simulation bias eventually becomes second order for large N.

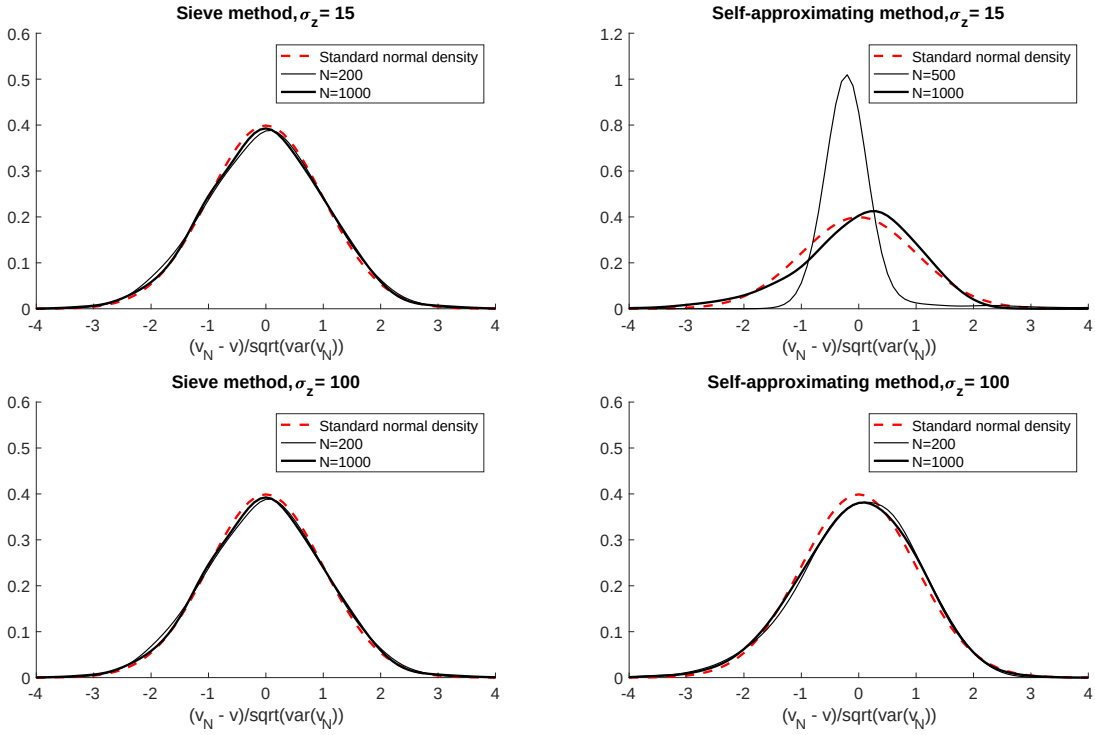
Comparing $||MSE||_\infty$ for $N = 500$ we find that the sieve-based method clearly dominates the self-approximating method when $\sigma_\varepsilon = 15$, whereas the self-approximating method performs best when $\sigma_\varepsilon = 100$. This is not entirely surprising since a large value of σ_ε implies a large conditional support of Z_t in which case the draws of the marginal sampler are more likely to fall within the support, c.f. the discussion in Subsection 6.2. Thus, the over-all error of the self-approximating method will tend to be smaller when σ_ε is large. The opposite is the case for the sieve based method which becomes more precise for smaller value of σ_ε since the variance of the simulated Bellman operator used for this method gets smaller as σ_ε gets smaller. This shows that there is considerably scope for improving the performance of the sieve-based method by more careful design of the sampling method.

Theorems 4 and 5 state that when N is large, the approximate value functions should be normally distributed. We here investigate whether this asymptotic approximation is useful in practice by looking at the pointwise distribution of the approximate solutions obtained through both methods. In Figure 9, we plot the distribution of $\left(\tilde{v}\left(z\right) - E\left[\tilde{v}\left(z\right)\right]\right) / \sqrt{Var\left(\tilde{v}\left(z\right)\right)}$ for $z = 500$, where $\tilde{v}$ denotes a given approximation method, together with the standard normal distribution. It is here important to note we do not center the estimate around $v\left(z\right)$ but instead around $E[\tilde{v}(z)]$; this is due to the sizable bias of the self-approximating method. For the sieve-based method, we see that its normalized distribution is quite close to the standard normal irrespectively of the value of σ_ε. In contrast, the normal distribution is a poor approximation for the self-approximating method when $\sigma_\varepsilon = 15$ when $N = 500$; we expect this is due to the fact that the effective number of draws is quite small and so the asymptotic approximation is poor in this case. As expected the approximation gets better as N and/or σ_ε increases.

6.5 Effect of smoothing

The results reported above did not involve any smoothing bias. We now numerically study the effect of smoothing. This is done by, instead of integrating out the i.i.d. extreme value taste shocks ε_t analytically as we have done so far, using Monte Carlo simulations to evaluate this part of the integral and then introducing the smoothing device to ensure that the simulated Bellman operator remains smooth. While this may appear somewhat artificial, the merit of doing this exercise is that we can use the same "exact" solution as benchmark as used above.

Figure 9: Asymptotic Normality



Notes: Each panel shows kernel density estimates of $(\widehat{v}_N(z) - E[\widehat{v}_N(z)])/\sqrt{var(\hat{v}_N(z))}$ for $z = 500$ based on $S = 2000$ solutions for each sample size N. Discount factor is $\beta = 0.95$, utility function parameters are $\theta_c = 2$, $RC = 10$, $\lambda = 1$ and transition parameters are $\sigma_\varepsilon = 100$, $a = 2$, $b = 5$ and $\pi = 0.000000001$.

In Figure 10 we plot the sup-norm of the mean squared error, $||\text{MSE}||_\infty$ as a function of λ (the smoothing scale parameter) for the sieve-based method using $K = 4$ or 8 Chebyshev polynomials (similar results were obtained for the self-approximating method and so are left out). For $K = 4$, the MSE increases monotonically as a function of λ while for $K = 15$ the bias due to smoothing is non-monotonic in λ. In both cases, at $\lambda = 0$, any remaining biases are due to either sieve-approximation or simulations. Importantly, the bias due to smoothing is negiglible (relative to the other biases) for small and moderate values of λ while the variance is largely unaffected. We have no theory or heuristics for choosing an optimal λ to optimally balance bias and variance due to smoothing but the current numerical results indicate that choosing a quite small λ value works well.

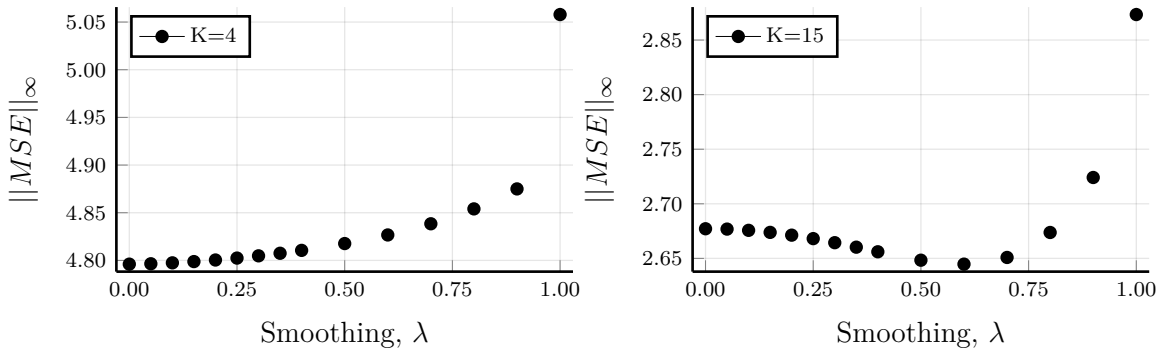
6.6 Performance in the bivariate case

We now examine how the solution methods perform in the bivariate case ($d_z = 2$) in order to see if there is any curse of dimensionality built into the two methods. We do this for two different models as described below.

An Additive DDP

We here follow the approach of Arcidiacono, Bayer, Bugni and James (2013) and Rust (1997a) and build a d_z-dimensional model by adding up d_z independent versions of the univariate model considered so far. That is, we choose the utilities and state dynamics as $\bar{u}(z, d) =$

Figure 10: Sup-norm MSE of solutions to Bellman operators with simulated taste shocks and state transitions for varying levels of smoothing, for $N = 100$.



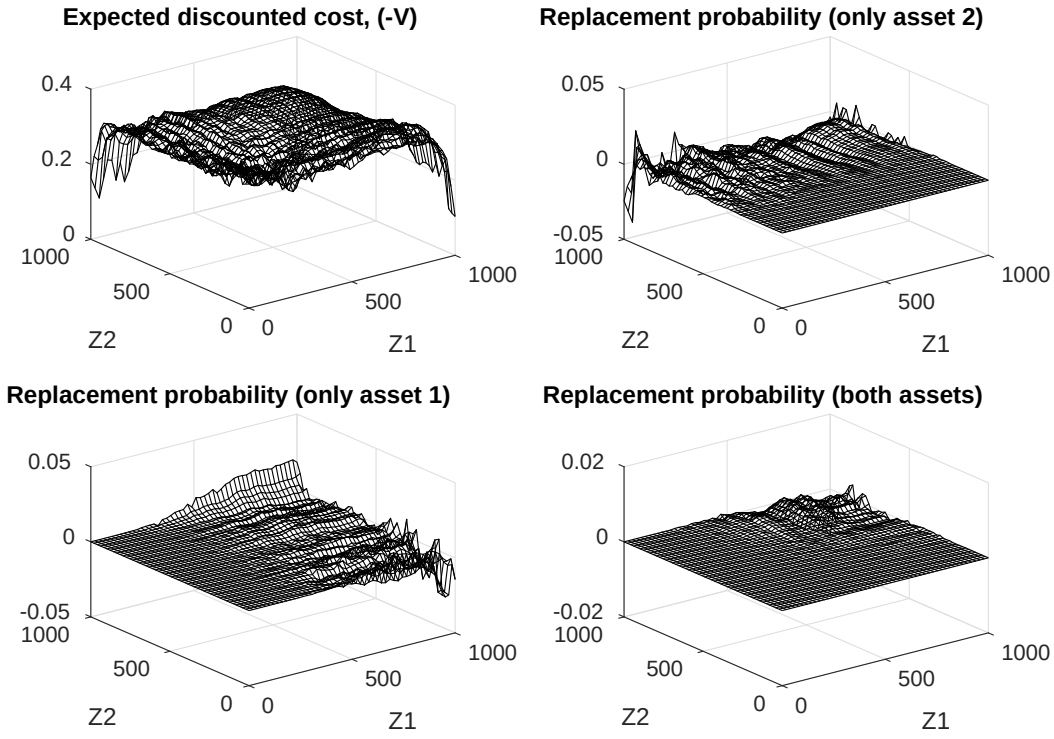
$\sum_{i=1}^{d_z} u(z_i, d_i)$ and $\bar{F}_Z(z'|z,d) = \prod_{i=1}^{d_z} F_Z(z_i'|z_i, d_i)$, where $z = (z_1, ..., z_{d_z})$ and $d = (d_1, ..., d_{d_z})$, with $F_Z(z_i'|z_i, d_i)$ and $u(z_i, d_i)$ denoting the state transition and per-period utility in the univariate case as described in Section 6.1. Note here that $Z_{t,i}$ and $Z_{t,j}$ are fully independent of each other, $i \neq j$ and the number of alternatives are 2^{d_z}. Thus, the model considers the joint replacement decision of d_z assets whose stochastic usages $(Z_{t,1}, ..., Z_{t,d_z})$ are mutually independent. Conveniently, the integrated value function of this multidimensional problem, $\bar{v}(z_1, ..., z_{d_z})$, is simply the sum of the solutions to each of the underlying univariate models, $\bar{v}(z_1, ..., z_{d_z}) = \sum_{i=1}^{d_z} v(z_i)$.where $v(z_i)$ is the solution to the univariate model in Section 6.1. This is a rather simplistic multivariate model but it comes with the major advantage that we can obtain a very accurate approximation of the exact solution by simply adding up the "exact" solution found for the univariate case. With a more complicated multidimensional structure, the computational cost of finding the "exact" solution is much higher. However, when implementing our solution methods we forgo forgo the knowledge of the additive structure of the solution and so treat the above model as a "proper" multivariate problem.

Simulation error

Given the issues with the self-approximating method for small values of $\sigma_\varepsilon = 15$, we here focus exclusively on the case $\sigma_\varepsilon = 100$. To get a sense of the pointwise performance of the self-approximating method, we plot the pointwise bias and standard deviation for this method in Figure 11 together with the pointwise errors of the corresponding replacement (choice) probabilities. The overall shape and level of the value function is quite well captured, and the same is true for the policy. However, the approximation errors tend to get larger out in the tails of the distribution and some of this comes from the fact that the issues with the marginal sampler used for the self-approximating method are amplified here. The problems are especially present in the off-grid evaluations, where we often have very few draws in a given region where we want to evaluate the value function or policies.

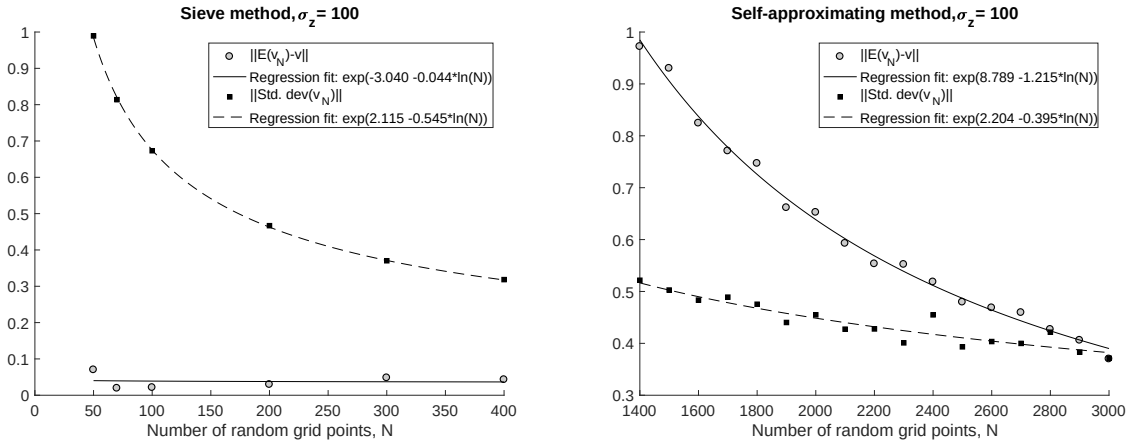
Next, we examine $||Bias||_\infty$ and $||\sqrt{Var}||_\infty$ for both methods as we increase N. These are plotted in Figure 12 where it should be noted that the reported range of N reported on the x-axis of the two figures differ substantially. This is due to the fact that the self–approximating method became numerically unstable for N smaller than 1,400 while no such issues were present

Figure 11: Approximation Error, bivariate DDP



Notes: Discount factor is $\beta = 0.95$, utility function parameters are $\theta_c = 2$, $RC = 10$, $\lambda = 1$ and transition parameters are $\sigma_\varepsilon = 100$, $a = 2$, $b = 5$ and $\pi = 0.000000001$. The "exact" solution was computed by averaging over $S = 100$ solutions, each found using the smoothed random Bellman operator with $N = 3000$ pseudo random draws. Each fixed point was found using a contraction tolerance of machine precision.

Figure 12: Simulation errors for bivariate additive DDP



Notes: Discount factor is $\beta = 0.95$, utility function parameters are $\theta_c = 2$, $RC = 10$, $\lambda = 1$ and and parameters for transition density $f(z'|z,d)$ are $\sigma_\varepsilon = 100$, $a = 2$, $b = 5$ and $\pi = 0.000000001$. Point-wise bias and variance was estimated in 500 evaluation points with $S = 200$ solutions, each found using the Bellman operator with N pseudo random draws. We report the sup norm of the bias and the standard deviation for each N for both methods and NLS regression fits of $||\sqrt{Var}\}||_\infty = \exp(\alpha_{SD} + \rho_{SD}\ln(N)$ and $||Bias||_\infty = \exp(\alpha_{Bias} + \rho_{Bias}\ln(N)$ where $\rho_{SD}$and ρ_{Bias}measures the rate for $||\sqrt{Var}\}||_\infty$ and $||Bias||_\infty$ respectively. For the self-approximating method we used $N = \{1400, 1500, .., 3000\}$ pseudo random draws. For the sieve methods we used much fewer draws.

for the sieve-based method. As in the univariate case, both bias and variance of the two methods vanish as N increases. However, comparing Figures 12 and 8, while the errors of the sieve-based method in the bivariate case is of a similar magnitude as in the univariate case, the errors of the self-approximating method are much larger in the bivariate case. This seems to indicate a certain type of curse-of-dimensionality in this particular application of the self-approximating method. This is caused by the issues with the marginal importance sampler employed for this method.

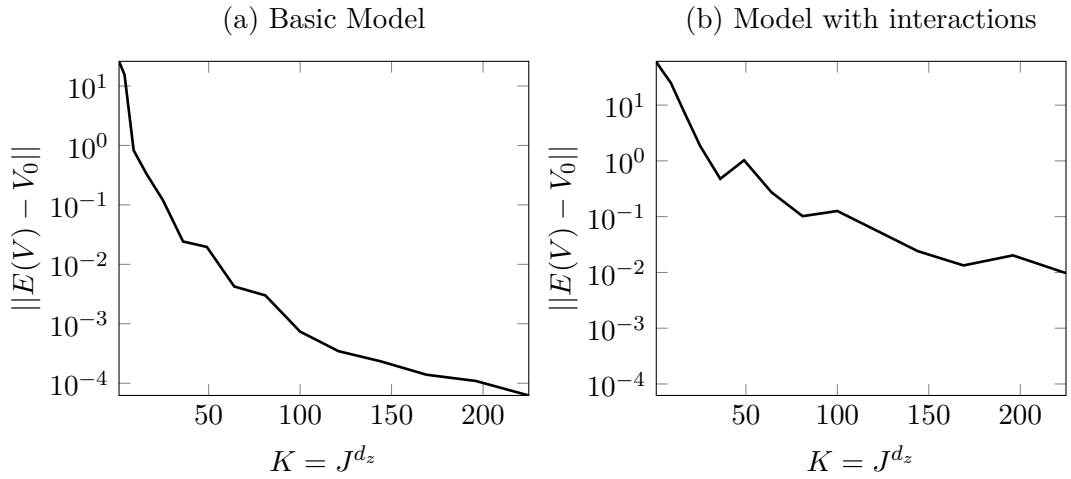
Sieve approximation error

In the implementation of the sieve-based method we use as sieve basis the tensor product of univariate Chebyshev polynomials or B-splines. That is, given, say, J univariate basis functions, say, $p_1, ..., p_J$, we construct our bivariate basis functions as $B_{i,j}(z_1, z_2) = p_i(z_1)\, p_j(z_2)$ for $i, j = 1, ..., J$ yielding a total of $K = J^2$ bivariate basis functions. In particularly, we do not exploit the additive structure of the problem since we are interested in the practical contents of Theorems 5 where no particular sparsity/special structure of the model is assumed to be known.

However, in practive, Chebyshev polynomials very easily pick up the additive structure and effectively sets the coefficients of the cross-product terms to zero. This is illustrated in Table 2 in Appendix D, where we report the coefficients for one particular projection-based bivariate value function estimate using a tensor product of $J = 5$ Chebyshev polynomials. However, this is due to the particular properties of the Chebyshev polynomials and is not enforced by us in the implementation. For example, if we instead use B-splines, the "estimated" coefficients of the cross-product terms were significantly different from zero, c.f. Table 3 in Appendix D.

In the left-hand side panel (a) of Figure 13, we report the uniform bias of the projection-based method with N chosen very large for the additive bivariate model. We find that the bias

Figure 13: Bias of value function in bivariate DDP for varying K .



vanishes as K increases as in the one-dimensional model. However, convergence is now slower in K relative to the univariate case and we require $K = 50$ to obtain a sieve approximation bias of 10^{-2} while $K = 7$ sufficed in the univariate case. This is consistent with theoretical error rates for polynomial interpolation where the rate slows down as the dimension of the problem increases, c.f. Section 4.3.

A non-additive DDP

Solving the additively separable model using sieve methods could be done using relatively few basis functions due to its simple structure. To investigate how it does in more comples, non-additive model, we now consider a slightly more complicated bivariate model where we include a multiplicative interaction term so that maintenance and replacement costs of the two busses interact, $\bar{u}(z, d) = \sum_{i=1}^{2} u(z_i, d_i) - u(z_1, d_1)u(z_2, d_2)/20$. Such a structure could, for example, reflect that capacity constraints make it more costly to simultaneously the engines of both busses at once.

The sieve approximation bias of our solution method for this model is reported in the right-hand side panel (b) in Figure 13. Compared to panel (a) – the additive case – we see that more sieve terms are required in order to reach a specific absolute error level in the model with interactions. In Table 4 in Appendix D the coefficients on the first ten basis functions in each dimension and their interactions are reported. Compared to the Chebyshev-based solution earlier we see quite significant coefficients on the coefficients for the cross-terms. However, the coefficients on the basis functions tend to zero quite quickly as K increase. The sup-norm of the difference in the value function at 40.000 evaluation grids is on the order of 10^{-5} when comparing the solutions with $K = 50^2 = 2500$ and $K = 30^2 = 900$ basis functions, and individual coefficients fall below 10^{-6} for univariate basis functions and cross products beyond the 22nd univariate basis functions, and below 10^{-8} around the 30th basis functions.However, it is important to stress that a large K here only comes with a computational cost while a large K has little effect on the variance of the sieve method. All together, we find that the sieve-based solution method works well also in higher dimensions, in particular when the model

has a particular structure that can be utilized in the solution method.

7 Conclusion

We have proposed two novel solution methods for dynamic discrete choice models. An asymptotic theory provided the leading bias and variance terms of the two methods and their large-sample distributions. A number of numerical experiments showed that the self-approximating method can be somewhat unstable while the sieve-based version is robust. The next step is to develop methods for choosing N, K and λ in a given setting so that the resulting approximate solution is of a good quality. Another area of research is to investigate how the proposed solution methods can be used to estimate dynamic discrete choice models.

References

Arcidiacono, P., P. Bayer, F. A. Bugni and J. James (2013): "Approximating High-dimensional Dynamic Models: Sieve Value Function Iteration," *Advances in Econometrics*, 31, 45–95.

Bowman, A., P. Hall and T. Prvan (1998): "Bandwidth Selection for the Smoothing of Distribution Functions," *Biometrika*, 85, 799–808.

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

Cai, Y. and K. L. Judd (2013): "Shape-preserving dynamic programming," *Mathematical Methods of Operations Research*, 77(3), 407–421.

Chen, V. C. (1999): "Application of orthogonal arrays and MARS to inventory forecasting stochastic dynamic programs," *Computational Statistics & Data Analysis*, 30, 317–341.

Chen, X. (2007): *Handbook of Econometrics, Volume 6, Part B*chap. Large Sample Sieve Estimation of Semi-Nonparametric Models, pp. 5549–5632. Elsevier B.V.

Chen, X. and H. White (1999): "Improved rates and asymptotic normality for nonparametric neural network estimators," *IEEE Transactions on Information Theory*, 45(2), 682–691.

Fermanian, J.-D. and B. Salanie (2004): "A Nonparametric Simulated Maximum Likelihood Estimation Method," *Econometric Theory*, 20, 701–734.

Iskhakov, F., T. H. Jørgensen, J. Rust and B. Schjerning (2017): "The endogenous grid method for discrete-continuous dynamic choice models with (or without) taste shocks," *Quantitative Economics*, 8(2), 317–365.

Judd, K. L., L. Maliar, S. Maliar and R. Valero (2014): "Smolyak method for solving dynamic economic models: Lagrange interpolation, anisotropic grid and adaptive domain," *Journal of Economic Dynamics and Control*, 44, 92–123.

Keane, M. and K. I. Wolpin (1994): "The Solution and Estimation of Discrete Choice Dynamic Programming Models by Simulation and Interpolation: Monte Carlo Evidence," *The Review of Economics and Statistics*, 76, 648–672.

Kristensen, D. and B. Salanie (2017): "Higher Order Properties of Approximate Estimators," *Journal of Econometrics*, 198, 189–208.

Kristensen, D. and Y. Shin (2012): "Estimation of Dynamic Models with Nonparametric Simulated Maximum Likelihood," *Journal of Econometrics*, 167, 76–94.

Lizotte, D. J. (2011): "Convergent fitted value iteration with linear function approximation," in *NIPS'11 Proceedings of the 24th International Conference on Neural Information Processing Systems*, pp. 2537–2545.

Lumsdaine, R., J. Stock and D. Wise (1992): "Three Models of Retirement: Computational Complexity versus Predictive Validity," in *Topics in the Economics of Aging*, ed. by D. Wise, pp. 21–60. NBER: University of Chicago Press.

McFadden, D. (1989): "A Method of Simulated Moments for Estimation of Discrete Response Models Without Numerical Integration," *Econometrica*, 57, 995–1026.

Munos, R. and C. Szepesvari (2008): "Finite-Time Bounds for Fitted Value Iteration," *Journal of Machine Learning Research*, 1, 815–857.

Newey, W. K. and R. J. Smith (2004): "Higher order properties of GMM and generalized empirical likelihood estimators," *Econometrica*, 72, 219–255.

Norets, A. (2010): "Continuity and differentiability of expected value functions in dynamic discrete choice models," *Quantitative economics*, 1(2), 305–322.

Norets, A. (2012): "Estimation of dynamic discrete choice models using artificial neural network approximations," *Econometric Reviews*, 31, 84–106.

Pal, J. and J. Stachurski (2013): "Fitted value function iteration with probability one contractions," *Journal of Economic Dynamics & Control*, 37, 251–264.

Rivlin, T. J. (1990): *Chebyshev Polynomials: From Approximation Theory to Algebra and Number Theory*. Wiley-Interscience, New York.

Robert, C. and G. Casella (2013): *Monte Carlo statistical methods*. Springer Science & Business Media.

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

——— (1988): "Maximum likelihood estimation of discrete control processes," *SIAM Journal on Control and Optimization*, 26(5), 1006–1024.

——— (1997a): "A comparison of policy iteration methods for solving continuous-state, infinite-horizon Markovian decision problems using random, quasi-random, and deterministic discretizations," .

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

——— (2008): "Dynamic programming," *The New Palgrave Dictionary of Economics: Volume 1–8*, pp. 1471–1489.

Rust, J., J. Traub and H. Wozniakowski (2002): "Is There a Curse of Dimensionality for Contraction Fixed Points in the Worst Case?," *Econometrica*, 70, 285–329.

Schumaker, L. L. (2007): *Spline Functions: Basic Theory, Third Edition*. Cambridge University Press.

Tjahjowidodo, T. et al. (2017): "A direct method to solve optimal knots of B-spline curves: An application for non-uniform B-spline curves fitting," *PloS one*, 12(3), e0173857.

van der Vaart, A. W. and J. A. Wellner (1996): *Weak Convergence and Empirical Processes*. Springer.

A Auxiliary Results

We derive a general result for projection-based approximate solutions to functional fixed-points. Let $(\mathcal{X}, \|\cdot\|)$ be a normed vector space and $\Psi : \mathcal{X} \to \mathcal{X}$ be some contraction mapping w.r.t. $\|\cdot\|$ so that there exists a unique solution $x_0 \in \mathcal{X}$ to $x = \Psi(x)$. Let Ψ_N be an approximation to Ψ and let Π_K be a projection operator, $K, N \geq 1$; the following theorem characterizes the (set of) solution(s) to $x = (\Pi_K \Psi_N)(x)$ as $K, N \to \infty$:

Theorem A.1. *Suppose that (i)* $\|\Psi_N(x_0) - \Psi(x_0)\| = O_p(\rho_{\Psi,N})$ *for some* $\rho_{\Psi,N} \to 0$*; (ii) for some* $\beta < 1$, $\|\Psi_N(x) - \Psi_N(y)\| \leq \beta\|x - y\|$ *for all* N *large enough and all* x, y*; (iii)* Π_K *satisfies* $\sup_{\|x - x_0\| < \delta} \|\Pi_K(x) - x\| = O_p(\rho_{\Pi,K})$ *for some* $\rho_{\Pi,K} \to 0$*. Then there exists a unique solution* $\hat{x} \in \mathcal{X}$ *to* $x = (\Pi_K \Psi_N)(x)$ *with probability approaching one (w.p.a.1) satisfying, with* $x_N = \Psi_N(x_N)$*,*

$$\|\hat{x} - x_0\| \leq \|\hat{x} - x_N\| + \|x_N - x_0\| = O_p(\rho_{\Pi,K}) + O_p(\rho_{\Psi,N}).$$

as $N \to \infty$*.*

Proof. We first observe that due to (ii), there exists a unique solution $x_N = \Psi_N(x_N)$ which satisfies

$$\begin{aligned}\|x_N - x_0\| = \|\Psi_N(x_N) - \Psi(x_0)\| &\leq \|\Psi_N(x_N) - \Psi_N(x_0)\| + \|\Psi_N(x_0) - \Psi(x_0)\| \\ &\leq \beta\|x_N - x_0\| + \|\Psi_N(x_0) - \Psi(x_0)\|,\end{aligned}$$

and so $\|x_N - x_0\| \leq \|\Psi_N(x_0) - \Psi(x_0)\| / (1 - \beta) = O_P(\rho_{\Psi,N})$. Next, combining (ii) and (iii), we see that $\Pi_K \Psi_N$ is a contraction mapping w.p.a.1. with Lipschitz coeffient β, and so $\hat{x}$ defined in the theorem exists and is unique w.p.a.1. Moreover, by the same arguments employed in the analysis of x_N together with the fact that $\|x_N - x_0\| < \delta$ w.p.a.1,

$$\|\hat{x} - x_N\| \leq \|\Pi_K \Psi_N(x_N) - \Psi_N(x_N)\| / (1 - \beta) \leq \sup_{\|x - x_0\| < \delta} \|\Pi_K(x) - x\| = O_p(\rho_{\Pi,K}).$$

□

Theorem A.2. *Suppose that in addition to the conditions stated in Theorem A.1, the following ones are satisfied: (i)* $\rho_{\Psi,N}\{\Psi_N(x_0) - \Psi(x_0)\} \rightsquigarrow \mathbb{G}$ *in* $(\mathcal{X}, \|\cdot\|)$ *and (ii)* $\Psi_N(x_0)$ *is almost surely Frechet differentiable at* x_0 *with Frechet differential* $\nabla\Psi_N(x_0)[dm] : \mathcal{X} \mapsto \mathcal{X}$ *such that* $\|\Psi_N(x) - \Psi_N(x_0) - \nabla\Psi_N(x_0)[x - x_0]\| = o_P(\|x - x_0\|)$ *for all* x *in a neighbourhood of* x_0*; (iii) it satisfies* $\|\nabla\Psi_N(x_0) - \nabla\Psi(x_0)\|_{op} = o_p(1)$*, where* $\|\nabla\Psi(x_0)\|_{op} = \sup_{\|dm\|=1} \|\nabla\Psi(x_0)[dm]\|$*. Then* $\{I - \nabla\Psi(x_0)\}[\rho_{\Psi,N}\{x_N - x_0\}] \rightsquigarrow \mathbb{G}$*. If furthermore* $dm \mapsto \{I - \nabla\Psi(x_0)\}[dm]$ *has a continuous inverse, then* $\rho_{\Psi,N}\{x_N - x_0\} \rightsquigarrow \{I - \nabla\Psi(x_0)\}^{-1}[\mathbb{G}]$*.*

Proof. Combining assumption (ii) with Lemma A.1,

$$\begin{aligned}0 &= x_N - \Psi_N(x_N) = x_0 - \Psi_N(x_0) + \{I - \nabla\Psi_N(x_0)\}[x_N - x_0] + o_P(\|x_N - x_0\|) \\ &= \Psi(x_0) - \Psi_N(x_0) + \{I - \nabla\Psi_N(x_0)\}[x_N - x_0] + o_P(\rho_{\Psi,N}),\end{aligned}$$

where, using (iii),

$$\begin{aligned}\left\|\left\{I-\nabla\Psi_N(x_0)\right\}\left[x_N-x_0\right]-\left\{I-\nabla\Psi(x_0)\right\}\left[x_N-x_0\right]\right\| &\leq \left\|\nabla\Psi_N(x_0)-\nabla\Psi(x_0)\right\|_{op}\left\|x_N-x_0\right\| \\ &= o_P(1)\,O_P\left(\rho_{\Psi,N}\right)=o_P\left(\rho_{\Psi,N}\right).\end{aligned}$$

The first part now follows from (i), while the second part follows by the continuous mapping theorem. □

B Proofs

Proof of Theorem 1. By the quasi-linearity of the social surplus function, for any $V_1, V_2 \in \mathbb{B}(\mathcal{Z})^D$,

$$\begin{aligned}\Gamma_N(V_1)(z,d) &= \sum_{i=1}^N G_\lambda\left(u\left(S_i(z)\right)+\beta V_2\left(Z_i(z)\right)+\beta\left[V_1\left(Z_i(z)\right)-V_2\left(Z_i(z)\right)\right]\right)w_{N,i}(z,d) \\ &\leq \sum_{i=1}^N G_\lambda\left(u\left(S_i(z)\right)+\beta V_2\left(Z_i(z)\right)+\beta\left\|V_1-V_2\right\|_\infty \mathbf{1}_D\right)w_{N,i}(z,d) \\ &= \sum_{i=1}^N G_\lambda\left(u\left(S_i(z)\right)+\beta V_2\left(Z_i(z)\right)\right)w_{N,i}(z,d)+\beta\left\|V_1-V_2\right\|_\infty\sum_{i=1}^N w_{N,i}(z,d) \\ &= \Gamma_N(V_2)(z,d)+\beta\left\|V_1-V_2\right\|_\infty,\end{aligned}$$

where $\mathbf{1}_d=(1,....,1)\in\mathbb{R}^D$ and we have used that $\sum_{i=1}^N w_{N,i}(z,d)=1$ by construction. Similarly, for any $v_1, v_2 \in \mathbb{B}(\mathcal{Z})$,

$$\begin{aligned}\bar{\Gamma}_N(v_1)(z) &\leq \sum_{j=1}^N G_\lambda\left(u(z,\varepsilon_j(z))+\beta\sum_{i=1}^N v_2\left(Z_{1,i}(z)\right)\circ w_{z,N,i}(z)+\beta\left\|v_1-v_2\right\|_\infty\mathbf{1}_D\right)w_{\varepsilon,N,j}(z) \\ &= \sum_{j=1}^N G_\lambda\left(u(z,\varepsilon_j(z))+\beta\sum_{i=1}^N v_2\left(Z_{1,i}(z)\right)\circ w_{z,N,i}(z)\right)w_{\varepsilon,N,j}(z)+\beta\left\|v_1-v_2\right\|_\infty \\ &= \bar{\Gamma}_N(v_2)(z)+\beta\left\|v_1-v_2\right\|_\infty.\end{aligned}$$

Next, we prove that $V_{N,\lambda}(z)$ is $s\geq 1$ times continuously differentiable: We know that Γ_N is a contraction mapping on $\mathbb{B}(\mathcal{Z})$. But the set of $s\geq 0$ continuously differentiable functions $\mathbb{C}_s(\mathcal{Z})$ is a closed subset of $\mathbb{B}(\mathcal{Z})$ and so the result will follow if $\Gamma_N\left(\mathbb{C}_s(\mathcal{Z})^D\right)\subseteq\mathbb{C}_s(\mathcal{Z})^D$. But for any $V\in\mathbb{C}_s(\mathcal{Z})^D$, it follows straightforwardly by the chain rule in conjuction with the stated assumptions that $\Gamma_N(V)(z)=\sum_{i=1}^N G_\lambda\left(u\left(S_i(z)\right)+\beta V\left(Z_i(z)\right)\right)w_{N,i}(z)$ is $s\geq 0$ continuously differentiable w.r.t. z.

For later use, we derive an expression of the first-order derivative $\partial V_{N,\lambda}(z)/(\partial z_j)$. To this end, first note that $0=\left\{I-\Gamma_{N,\lambda}\right\}(V_{N,\lambda})(z)$. The implicit function theorem then implies that, assuming that $M\mapsto\left[I-\nabla\Gamma_{N,\lambda}\right]^{-1}(V)[M]$ is well-defined and continuous,

$$\frac{\partial V_{N,\lambda}(z)}{\partial z_j}=\left\{I-\nabla\Gamma_N(V_{N,\lambda})\right\}^{-1}\left[\dot{\Gamma}_{N,j}(V_{N,\lambda})\right](z), \tag{B.1}$$

where, with $\dot{G}_{d,\lambda}(r) = \frac{\partial G_\lambda(r)}{\partial r(d)} = \exp\left(\frac{r(d)}{\lambda}\right) / \sum_{d' \in \mathcal{D}} \exp\left(\frac{r(d')}{\lambda}\right)$,

$$\begin{aligned}\dot{\Gamma}_{N,j}(V)(z) &= \sum_{i=1}^{N} \sum_{d \in \mathcal{D}} \dot{G}_{\lambda,d}\left(u\left(S_i(z)\right) + \beta V\left(Z_i(z)\right)\right) \frac{\partial u\left(S_i(z), d\right)}{\partial z_j} w_{N,i}(z) \\ &\quad + \sum_{i=1}^{N} G_\lambda\left(u\left(S_i(z)\right) + \beta V\left(Z_i(z)\right)\right) \frac{\partial w_{N,i}(z)}{\partial z_j}.\end{aligned}$$

But it is easily checked that $M \mapsto \nabla\Gamma_{N,\lambda}(V)[M]$ is a continuous linear operator with norm $\|\nabla\Gamma_{N,\lambda}\|_{op} \leq \beta$. By the Banach inverse theorem, the inverse of $I - \nabla\Gamma_{N,\lambda}$ is therefore well-defined and continuous. For later use, also observe that

$$\left\|\frac{\partial \hat{V}_R(z)}{\partial z_j}\right\| \leq \left\|\left\{I - \nabla\Gamma_N(\hat{V}_R)\right\}^{-1}\right\| \left\|\dot{\Gamma}_{N,j}(\hat{V}_R)\right\| \leq (1-\beta)^{-1} \left\|\dot{\Gamma}_{N,j}(\hat{V}_R)\right\|,$$

where

$$\begin{aligned}\left\|\dot{\Gamma}_{N,j}(V)\right\| &\leq \sum_{i=1}^{N} \sum_{d \in \mathcal{D}} \dot{G}_{\lambda,d}\left(u\left(S_i(z)\right) + \beta V\left(Z_i(z)\right)\right) w_{N,i}(z) \times \left\|\frac{\partial u(\cdot)}{\partial z_j}\right\|_\infty \\ &\quad + \sum_{i=1}^{N} \left|G_\lambda\left(u\left(S_i(z)\right) + \beta V\left(Z_i(z)\right)\right)\right| \left\|\frac{\partial w_{N,i}(z)}{\partial z_j}\right\| \\ &\leq \left\|\frac{\partial u(\cdot)}{\partial z_j}\right\|_\infty + \sum_{i=1}^{N} \left\|\frac{\partial w_{N,i}(z)}{\partial z_j}\right\| \times \left\{\|u(\cdot)\|_\infty + \beta \|V(\cdot)\|_\infty\right\}\end{aligned}$$

□

Proof of Theorem 2. We only show the result for V_λ; the proof for $V_{N,\lambda}$ is analogous. Applying (3.6), the following holds for any V,

$$\begin{aligned}|\Gamma_\lambda(V)(z,d) - \Gamma(V)(z,d)| &\leq \int_{\mathcal{Z}\times\mathcal{E}} \left|\max_{d \in \mathcal{D}} \left\{u(s',d) + \beta V(z',d')\right\} - G_\lambda\left(u(s') + \beta V(z')\right)\right| dF_s(ds'|z,d) \\ &\leq \sup_{r \in \mathbb{R}^D} \left|G_\lambda(r) - \max_{d \in \mathcal{D}} r(d)\right| \int_{\mathcal{Z}\times\mathcal{E}} dF_s(ds'|z,d) \\ &= \lambda \log D.\end{aligned}$$

The result now follows from Theorem A.1 with $\Pi_K(V) = V$ and $\Psi_N = \Gamma_{\lambda_N}$. □

Proof of Theorem 3. We apply Theorem A.1 with $\Pi_K(V) = V$ and $\Psi_N = \Gamma_N$. We define

$$\bar{u}(U;z) := u\left(\psi_z(z,U), \psi_\varepsilon(z,U)\right), \; \bar{w}(U;z) = w\left(\psi(z,U)|z\right), \bar{V}(U;z,\lambda) = V_\lambda\left(\psi_z(z,U)\right),$$

so that we can write $\Gamma_\lambda(V_\lambda)(z) = E_U\left[G_\lambda\left(\bar{u}(U;z) + \beta\bar{V}(U;z,\lambda)\right)\bar{w}(U;z)\right]$ and

$$\Gamma_{N,\lambda}(V)(z) = \frac{\tilde{\Gamma}_{N,\lambda}(V)(z)}{W_N(z)}. \tag{B.2}$$

where

$$\tilde{\Gamma}_{N,\lambda}(V_\lambda)(z) = \frac{1}{N}\sum_{i=1}^{N} G_\lambda\left(\bar{u}\left(U_i;z\right)+\beta\bar{V}(U_i;z,\lambda)\right)\bar{w}\left(U_i;z\right), \tag{B.3}$$

$$W_N\left(z\right) = \frac{1}{N}\sum_{j=1}^{N}\bar{w}\left(U_i;z\right), \tag{B.4}$$

Due to Assumption 1(ii), it follows from Theorem 2.7.11 of van der Vaart and Wellner (1996) that the bracketing number of

$$\mathcal{W} := \left\{u \mapsto \bar{w}\left(u;z\right)|z \in \mathcal{Z}\right\} \tag{B.5}$$

satisfies $N_{[]}(2\varepsilon C, \mathcal{W}, \|\cdot\|_\infty) \precsim N(\varepsilon, \mathcal{Z}, \|\cdot\|) \precsim \varepsilon^{-d_z}$. Thus, $\mathcal{W}$ is a P_U-Donsker class since $\int_0^\infty \sqrt{\log N_{[]}(\varepsilon, \mathcal{W}, \|\cdot\|_\infty)}d\varepsilon \precsim \int_0^\infty \sqrt{\log N(\varepsilon, \mathcal{Z}, \|\cdot\|)}d\varepsilon < \infty$, and so

$$\sup_{z\in\mathcal{Z}}|W_N\left(z\right)-1| = O_P\left(1/\sqrt{N}\right), \tag{B.6}$$

Next, observe that the function class $\mathcal{G}$ defined in Lemma 1 also is a Donsker class. Given that both $\mathcal{G}$ and $\mathcal{W}$ are bounded classes, $\mathcal{G}\cdot\mathcal{W}$ is also Donsker and so $\sup_{\lambda\in(0,\bar{\lambda})}\left\|\tilde{\Gamma}_{N,\lambda}(V_\lambda)-\Gamma_\lambda(V_\lambda)\right\|_\infty = O_P\left(1/\sqrt{N}\right)$ which combined with (B.6) yield $\sup_{\lambda\in(0,\bar{\lambda})}\|\Gamma_{N,\lambda}(V_\lambda)-\Gamma_\lambda(V_\lambda)\|_\infty = O_P\left(1/\sqrt{N}\right)$. We conclude from Theorem A.1 that $\sup_{\lambda\in(0,\bar{\lambda})}\|V_{N,\lambda}-V_\lambda\|_\infty = O_p(1/\sqrt{N})$.

To show convergence of the first-order derivatives of the approximate value function, recall that $\partial V_{N,\lambda}/\left(\partial z_j\right)$ satisfies (B.1) while $\partial V_\lambda/\left(\partial z_j\right)$ solves

$$\frac{\partial V_\lambda\left(z\right)}{\partial z_j} = \left\{I-\nabla\Gamma_\lambda(V_\lambda)\right\}^{-1}\left[\dot{\Gamma}_{\lambda,j}(V_\lambda)\right]\left(z\right),$$

,

where

$$\nabla\Gamma_\lambda(V_\lambda)\left[M\right]\left(z\right) = \beta\sum_{d\in\mathcal{D}} E_U\left[\dot{G}_{d,\lambda}\left(\bar{u}\left(U;z\right)+\beta\bar{V}(U;z,\lambda)\right)M\left(U;z,d\right)\bar{w}\left(U;z\right)\right]$$

and

$$\begin{aligned}\dot{\Gamma}_{\lambda,j}(V)\left(z\right) = &\sum_{d\in\mathcal{D}} E_U\left[\dot{G}_{\lambda,d}\left(\bar{u}\left(U;z\right)+\beta\bar{V}(U;z,\lambda)\right)\frac{\partial\bar{u}\left(U;z\right)}{\partial z_j}\bar{w}\left(U;z\right)\right]\\ &+E_U\left[G_\lambda\left(\bar{u}\left(U;z\right)+\beta\bar{V}(U;z,\lambda)\right)\frac{\partial\bar{w}\left(U;z\right)}{\partial z_j}\right].\end{aligned}$$

Here note that $M \mapsto \left\{I-\nabla\Gamma_\lambda(V_\lambda)\right\}[M]$ and $M \mapsto \left\{I-\nabla\Gamma_{N,\lambda}(V_{N,\lambda})\right\}[M]$ are both bounded linear operators with operator norm $1-\beta$. In particular, they are both contraction mappings. Thus, we can again apply A.1 with $\Psi_N\left(M\right) = \left(I-\nabla\Gamma_{N,\lambda}(V_{N,\lambda})\right)[M]$. To show convergence of $\Psi_N\left(\partial V_\lambda/\left(\partial z_j\right)\right)$, first note that $\dot{G}_{\lambda,d}\left(r\right)$ is Lipschitz in r uniformly in λ, c.f. the proof of Lemma

1. This combined with $\mathcal{W}$ being a Donsker class implies

$$\left\|\nabla\Gamma_{N,\lambda}(V_{N,\lambda})\left[\partial V_\lambda/\left(\partial z_j\right)\right]-\nabla\Gamma_{N,\lambda}(V_\lambda)\left[\partial V_\lambda/\left(\partial z_j\right)\right]\right\|_\infty = O_P\left(\left\|V_{N,\lambda}-V_\lambda\right\|_\infty\right) = O_P\left(1/\sqrt{N}\right),$$

while, using arguments similar to the ones used in the first part of the proof in conjunction with Lemma 1,

$$\left\|\nabla\Gamma_{N,\lambda}(V_\lambda)\left[\partial V_\lambda/\left(\partial z_j\right)\right]-\nabla\Gamma_{\lambda}(V_\lambda)\left[\partial V_\lambda/\left(\partial z_j\right)\right]\right\|_\infty = O_P\left(1/\sqrt{N}\right).$$

This completes the proof. □

Proof of Theorem 4. We apply Theorem A.2. First recall that$M \mapsto \{I-\nabla\Gamma_\lambda\}^{-1}(V_\lambda)[M]$ is well-defined and continuous and that $\sqrt{N}\left(\tilde{\Gamma}_{N,\lambda}(V_\lambda)-\Gamma_\lambda(V_\lambda), W_N-1\right) \rightsquigarrow (\mathbb{G}_1,\mathbb{G}_2)$, c.f. proof of Theorem 3. The latter result implies that

$$\begin{aligned}\sqrt{N}\left\{\Gamma_{N,\lambda}(V_\lambda)-\Gamma_\lambda(V_\lambda)\right\} &= \sqrt{N}\left\{\tilde{\Gamma}_{N,\lambda}(V_\lambda)-\Gamma_\lambda(V_\lambda)\right\}-\Gamma_\lambda(V_\lambda)\sqrt{N}\left\{W_N-1\right\}+o_P(1)\\ &\rightsquigarrow \mathbb{G}:=\mathbb{G}_1-\Gamma_\lambda(V_\lambda)\mathbb{G}_2.\end{aligned}$$

It is easily seen that the influence function of $\Gamma_{N,\lambda}(V_\lambda)$ takes the form

$$g(U;z,\lambda)=\left\{G_\lambda\left(\bar{u}(U;z)+\beta\bar{V}(U;z,\lambda)\right)-\Gamma_\lambda(V_\lambda)\right\}\bar{w}(U;z),$$

and so $\mathbb{G}(z,\lambda)$ has covariance kernel

$$\Omega(z_1,\lambda_1,z_2,\lambda_2)=E_U\left[g(U;z_1,\lambda_1)\,g(U;z_2,\lambda_2)'\right]$$

What remains to be shown is the uniform convergence of $M \mapsto \nabla\Gamma_{N,\lambda}(V_\lambda)[M]$ over some suitable function set $\mathcal{M}$ chosen such that $V_{N,\lambda}-V_\lambda \in \mathcal{M}$ w.p.a.1. In the case where $\mathcal{Z}$ is finite, $\mathcal{M}$ is also finite-dimensional and so has finite bracketing number. In the case where $\mathcal{Z}$ is not finite, we instead invoke Assumption 2 which implies that $V_{N,\lambda}$ and V_λ are both smooth, c.f. Theorem 2. Moreover, from Theorem 3, $\sup_{\lambda\in(0,\bar{\lambda})}\|V_{N,\lambda}-\partial V_\lambda\|_\infty = O_P\left(1/\sqrt{N}\right)$ and $\sup_{\lambda\in(0,\bar{\lambda})}\left\|\partial V_{N,\lambda}/\left(\partial z\right)-\partial V_\lambda/\left(\partial z\right)\right\|_\infty = O_P\left(1/\sqrt{N}\right)$. Thus, we can choose $\mathcal{M}=\{M\in \mathbb{C}_1(\mathcal{Z})^D : \|M\|_{1,\infty}<r\}$ for some $r<\infty$, where $\|M\|_{s,\infty}$ was defined in (4.9). By Theorem 2.7.1 in van der Vaart and Wellner (1996), $N_{[]}(\epsilon,\mathcal{M},L_1(P_U))<\infty$ for any given $\epsilon>0$ which combined with the fact that the bracketing numbers of $\dot{\mathcal{G}}_d$ and $\mathcal{W}$ are also finite imply that $\mathcal{F}=\dot{\mathcal{G}}_d\cdot\mathcal{W}\cdot\mathcal{M}$ has finite bracketing number (where we have used that each of the three functions classes are uniformly bounded in U). It now follows from the Glivenko-Cantelli Theorem that $\sup_{M\in\mathcal{M}}\left\|\nabla\tilde{\Gamma}_{N,\lambda}(V_\lambda)[M]-\nabla\Gamma_\lambda(V_\lambda)[M]\right\|_\infty = o_p(1)$ which together with (B.6) yield the desired result. □

Proof of Theorem 5. The rate result is an immediate consequence of Theorem A.1 together with Assumption 3. For the weak convergence result, we use the decomposition (**??**) where, by Theorem A.1, $\left\|\hat{V}_{N,\lambda}-V_{N,\lambda}\right\|_\infty = O_p(\rho_K) = o_P\left(1/\sqrt{N}\right)$ while the second term converges weakly according to Theorem 4. □

C Lemmas

Lemma 1. *Suppose that Assumption 2 hold. Then, for any* $\bar{\lambda} < \infty$,

$$\mathcal{G} \equiv \left\{ G_\lambda \left(u \left(\psi \left(z, \cdot \right) \right) + \beta V_\lambda \left(\psi_z \left(z, \cdot \right) \right) \right) | \left(z, \lambda \right) \in \mathcal{Z} \times \left(0, \bar{\lambda} \right) \right\} \tag{C.1}$$

$$\dot{\mathcal{G}}_d \equiv \left\{ \dot{G}_{\lambda,d} \left(u \left(\psi \left(z, \cdot \right) \right) + \beta V_\lambda \left(\psi_z \left(z, \cdot \right) \right) \right) | \left(z, \lambda \right) \in \mathcal{Z} \times \left(0, \bar{\lambda} \right) \right\}, \tag{C.2}$$

satisfy $N \left(\varepsilon, \mathcal{G}, \|\cdot\|_2 \right) \precsim \varepsilon^{-v}$ *and* $N \left(\varepsilon, \dot{\mathcal{G}}_d, \|\cdot\|_2 \right) \precsim \varepsilon^{-v}$, $d \in \mathcal{D}$, *for some* $v \geq 1$.

Proof. We first analyze the properties of $V_\lambda(z)$: It is easily seen that $\Gamma_\lambda(V)(z)$ is Lipschitz w.r.t z uniformly over $\lambda \in \left(0, \bar{\lambda} \right)$ for any function $V \left(z \right)$ which is Lipschitz. Thus,by the same arguments as used in the proof of Theorem 1, $V_\lambda \left(z \right)$ is Lipshitz in z uniformly over $\lambda \in \left(0, \bar{\lambda} \right)$. Moreover, $\lambda \mapsto \Gamma_\lambda(V)(z) = E_U \left[G_\lambda \left(\bar{u} \left(U; z \right) + \beta V \left(\psi_z \left(z, U \right) \right) \right) \bar{w} \left(U; z \right) \right]$ is continuously differentiable by the dominated convergence theorem which, by the implicit function theorem, in turn implies that $\lambda \mapsto V_\lambda \left(z \right)$ is continuously differentiable with its derivative being on the form

$$\frac{\partial V_\lambda \left(z \right)}{\partial \lambda} = \left\{ I - \nabla \Gamma_\lambda(V_\lambda) \right\}^{-1} \left[\partial_\lambda \Gamma_\lambda(V_\lambda) \right] \left(z \right), \tag{C.3}$$

where

$$\partial_\lambda \Gamma_\lambda(V) \left(z \right) = E \left[\frac{\partial G_\lambda \left(\bar{u} \left(U; z \right) + \beta \bar{V}(U; z, \lambda) \right)}{\partial \lambda} \bar{w} \left(U; z \right) \right],$$

and

$$\frac{\partial G_\lambda \left(r \right)}{\partial \lambda} = \log \left[\sum_{d \in \mathcal{D}} \exp \left(\frac{r \left(d \right)}{\lambda} \right) \right] - \frac{\sum_{d \in \mathcal{D}} \exp \left(\frac{r(d)}{\lambda} \right) r \left(d \right)}{\lambda \sum_{d \in \mathcal{D}} \exp \left(\frac{r(d)}{\lambda} \right)}.$$

Write

$$G_\lambda \left(r \right) = \lambda \log \left[\sum_{d \in \mathcal{D}} \exp \left(\frac{r \left(d \right)}{\lambda} \right) \right] = \max_{d \in D} r \left(d \right) + \lambda \log \left[\sum_{d \in \mathcal{D}} \exp \left(\frac{\bar{r} \left(d \right)}{\lambda} \right) \right],$$

where $\bar{r} \left(d \right) = r \left(d \right) - \max_{d \in \mathcal{D}} r(d) \leq 0$, $d \in \mathcal{D}$, to obtain

$$\frac{\partial G_\lambda \left(r \right)}{\partial \lambda} = \log \left[\sum_{d \in \mathcal{D}} \exp \left(\frac{\bar{r} \left(d \right)}{\lambda} \right) \right] - \frac{\sum_{d \in \mathcal{D}} \exp \left(\frac{\bar{r}(d)}{\lambda} \right) \bar{r} \left(d \right) / \lambda}{\sum_{d \in \mathcal{D}} \exp \left(\frac{\bar{r}(d)}{\lambda} \right)}.$$

Since $1 \leq \sum_{d \in \mathcal{D}} \exp \left(\frac{\bar{r}(d)}{\lambda} \right) \leq D$ and $-De^{-1} \leq \sum_{d \in \mathcal{D}} \exp \left(\frac{\bar{r}(d)}{\lambda} \right) \bar{r} \left(d \right) / \lambda \leq 0$ for all $\lambda > 0$ and all $r \in \mathbb{R}^D$, we conclude that $\left| \partial G_\lambda \left(r \right) / \left(\partial \lambda \right) \right| \leq \log \left(D \right) + De^{-1}$ and so is Lispchitz uniformly in $\lambda \in (0, \bar{\lambda})$. This in turn implies that $\sup_{z,\lambda} \left\| \frac{\partial V_\lambda(z)}{\partial \lambda} \right\| < \infty$. This combined with Assumption 1 implies that $G_\lambda(u(\psi \left(z, \cdot \right)) + \beta V_\lambda(\psi_z \left(z, \cdot \right)))|z)$ is Lipschitz w.r.t. (z, λ). It now follows from Theorem 2.7.11 of van der Vaart and Wellner (1996) that the ε-covering number of $\mathcal{G}$ is of order ε^{-v} for some $v \geq 1$.

The proof of the bound of $N \left(\varepsilon, \dot{\mathcal{G}}_d, \|\cdot\|_2 \right)$ is analogous except that we now use that $\dot{G}_{\lambda,d} \left(r \right) \in$

$(0,1)$ so that

$$\begin{aligned} & \left\{\left\{(u,t)\,|\dot{G}_{\lambda,d}(u\left(\psi\left(z,u\right)\right)+\beta V_{\lambda}\left(\psi_{z}\left(z,u\right)\right))<t\right\}\,|\,(z,\lambda)\in\mathcal{Z}\times(0,\bar{\lambda})\right\} \\ = & \left\{\left\{(u,t)\,|\dot{G}_{\lambda,d}^{-1}(t)-u\left(\psi\left(z,u\right)\right)+\beta V_{\lambda}\left(\psi_{z}\left(z,u\right)\right)>0\right\}\,|\,(z,\lambda)\in\mathcal{Z}\times(0,\bar{\lambda})\right\} \end{aligned}$$

and it now follows from Lemmas 2.6.15 and 2.6.18(iii) of van der Vaart and Wellner (1996) that the ε-covering number of $\dot{\mathcal{G}}_d$ is of order ε^{-v} for some $v \geq 1$. □

D Additional numerical details for sieve method

Chebyshev basis functions

Interpolation and approximation by Chebyshev polynomials of the first kind have well-known good properties when approximating functions on bounded intervals. Recall that Chebyshev polynomials are defined on $[-1,1]$. We then choose $0 \leq z^{min} < z^{max} < \infty$ and define the kth basis function as follows for any $z \in \mathbb{R}$,

$$B_{c,k}(z) = \begin{cases} \cos\left((k-1)\arccos(T(z))\right) & |T(z)| \leq 1 \\ (\text{sign}(T(z)))^{k} & |T(z)| > 1 \end{cases}.$$

where

$$T(z) = 2\frac{z - z^{min}}{z^{max} - z^{min}} - 1$$

linearly transforms our z's to the interval $[-1,1]$. In particular, the basis functions are "truncated" and are set to one outside the interval $\left[z^{min}, z^{max}\right]$. This is done to avoid any erratic extrapolation. For interpolation it is natural to use the Chebyshev nodes to minimize the presence of Runge's phenomenon. For approximation, that is $M > K$, it is less clear what to do, though one possibility is to augment the Chebyshev nodes associated with the order $K-1$ Chebyshev polynomial with random numbers distributed uniformly across the interval we're considering.

B-Splines

We use cardinal B(asis)-splines to form our B-spline spaces, so they are represented by a knot vector with equidistant entries $(0, \frac{1}{M+1}, \frac{2}{M+1}, \ldots, \frac{M}{M+1}, 1)$, and the Cox-de Boor recursion

$$\begin{aligned} \bar{B}_{i,0}(z) =& \begin{cases} 1 & \text{if } t_i \leq z < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \\ \bar{B}_{i,k}(z) =& \frac{z - t_i}{t_{i+k} - t_i}\bar{B}_{i,k-1}(z) + \frac{t_{i+k+1} - z}{t_{i+k+1} - t_{i+1}}\bar{B}_{i+1,k-1}(z). \end{aligned}$$

For interpolation purposes we use the so-called Universal (Parameters) Method by Tjahjowidodo et al. (2017). This amounts to choosing the M-grid to consist of the unique maximizers of all

Table 2: Coefficients on tensor product Chebyshev basis functions in the 2D model of engine replacement for $K = J^2 = 25$, $N = 200$.

$J_1 \backslash J_2$	1	2	3	4	5
1	-38.4713	-4.4754	1.6176	-0.256420	-0.064960
2	-4.4754	1.9662e-14	-6.2341e-15	-1.1318e-15	2.5392e-15
3	1.6176	7.2256e-15	5.6179e-14	-2.0548e-14	-3.3049e-15
4	-0.2564	-1.0110e-14	-8.8673e-15	7.5672e-15	-2.9838e-15
5	-0.0649	4.0869e-15	-1.5251e-14	3.4571e-15	1.4218e-15

Table 3: Coefficients on tensor product 2nd order B-Spline basis functions in the 2D model of engine replacement for $K = J^2 = 25$, $N = 200$.

$J_1 \backslash J_2$	1	2	3	4	5
1	-21.9800	-27.1194	-30.0583	-31.1025	-31.506
2	-27.1194	-32.2589	-35.1977	-36.2419	-36.6455
3	-30.0583	-35.1977	-38.1366	-39.1808	-39.5844
4	-31.1025	-36.2419	-39.1808	-40.2250	-40.6285
5	-31.5060	-36.6455	-39.5844	-40.6285	-41.0321

B-splines of degree $k \geq 1$, or any point if $k = 0$ in which case we set it to the first K elements of the knot vector. Since the points given by the Universal Method makes the interpolation quite well-behaved, we also use these points for approximation, and augment them the same way as we described for the Chebyshev Sieve spaces.

The above are defined on the unit interval $[0, 1]$ and so the final basis functions are chosen as

$$B_{c,k}(z) = \begin{cases} \bar{B}_k(T(z)) & 0 \leq T(z) \leq 1 \\ (\text{sign}(T(z))) & otherwise \end{cases}.$$

where now

$$T(z) = \frac{z - z^{min}}{z^{max} - z^{min}}.$$

Table 4: Coefficients on the ten basis functions, and their products, upon convergence with $K = 50^2$.

$J_1 \backslash J_2$	1	2	3	4	5	6	7	8	9	10
1	-52.200	-5.070	2.700	-1.170	0.478	-0.145	-0.006	0.038	-0.024	0.007
2	-5.070	-1.560	0.135	0.225	-0.057	-0.023	0.013	-0.006	0.004	0.001
3	2.700	0.135	-0.176	0.032	0.046	-0.021	-0.004	0.005	-0.002	0.001
4	-1.170	0.225	0.032	-0.087	0.019	0.019	-0.012	0.002	0.001	-0.001
5	0.478	-0.057	0.046	0.019	-0.038	0.010	0.009	-0.008	0.002	0.000
6	-0.145	-0.023	-0.021	0.019	0.010	-0.018	0.005	0.005	-0.004	0.002
7	-0.006	0.013	-0.004	-0.012	0.009	0.005	-0.009	0.003	0.002	-0.003
8	0.038	-0.006	0.005	0.002	-0.008	0.005	0.003	-0.005	0.002	0.001
9	-0.024	0.004	-0.002	0.001	0.002	-0.004	0.002	0.002	-0.003	0.001
10	0.007	0.001	0.001	-0.001	0.0002	0.002	-0.003	0.001	0.001	-0.002