Strong Convergence and Dynamic Economic Models

Robert L. Bray∗

r-bray@kellogg.northwestern.edu

Kellogg School of Management at Northwestern University,
Operations Department

April 5, 2018

Abstract

Morton and Wecker (1977) showed that the value iteration algorithm solves a Markov decision process’s policy function faster than its value function when the limiting Markov chain is ergodic. I identify an error in their proof and provide a new proof of this classic result. I use this result to speed up the estimation of Markov decision processes and the solution of Markov perfect equilibria.

Keywords: Markov decision process; Markov perfect equilibrium; strong convergence; dynamic discrete choice; nested fixed point; nested pseudo-likelihood.

1 Introduction

I present a simple refinement that speeds up the estimation of ergodic Markov decision processes. I exploit three facts:

1. The empirical likelihood of a dynamic model depends only on the dynamic program’s policy function, not its value function.

2. The policy function depends only on the value function’s relative differences, not its absolute level.

∗I would like to thank Victor Aguirregabiria, Achal Bassambo, Thomas Bray, Seyed Emadi, Soheil Ghili, Arvind Magesan, John Rust, Dennis Zhang, and three anonymous referees for their helpful comments.
3. The value function’s relative differences converge faster than its level under repeated Bellman contractions when the underlying stochastic process is ergodic. This is called strong convergence.

Operations researchers discovered this strong convergence property decades ago. But their proof is incorrect. I provide a new proof of the relative value function’s faster convergence rate.


2 Markov Decision Process

I consider a dynamic program with discrete time periods, an infinite planning horizon, a finite state space, and an uncountable action space. Taking a specific action in a specific state yields a specific utility. The goal is to determine the actions that yield the maximum expected discounted utility. I will define the constituent parts of the Markov decision process with Judd’s (1998) notation.

1. \( \iota \) is the length-\( m \) vector of ones.

2. \( \delta_i \) is the length-\( m \) unit vector indicating the \( i \)th position.

3. \( a \subset \mathbb{R}^\ell \) is the compact action space.

4. \( \mathbb{x} \equiv \{x_1, \cdots, x_m\} \) is the state space.

5. \( \beta \in [0, 1) \) is the discount factor.

6. \( U : \mathbb{x} \rightarrow a \) is a generic policy function that maps states to actions. Policy \( U \) specifies taking action \( U(x_i) \) in state \( x_i \).

7. \( \mathbb{U} \equiv a^{\mathbb{x}} \) is the set of permissible policy functions: if \( U \in \mathbb{U} \) and \( x_i \in \mathbb{x} \) then \( U(x_i) \in a \).
8. $\pi : \mathbb{U} \rightarrow \mathbb{R}^m$ is a continuous and uniformly bounded function that maps policy functions to length-$n$ flow utility vectors: the $i$th element of $\pi(U)$ is the flow utility received from taking action $U(x_i)$ in state $x_i$.

9. $Q : \mathbb{U} \rightarrow \mathbb{R}^{m \times m}$ is a continuous function that maps policy functions to $m \times m$ stochastic matrices: the $ij$th element of $Q(U)$ is the probability of transitioning from state $x_i$ to state $x_j$ under action $U(x_i)$.

10. $V \in \mathbb{R}^m$ is a vector that characterizes a generic value function: the $i$th element of $V$ denotes the expected discounted flow utility from state $x_i$.

11. $T_U : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is the Bellman contraction operator:

   (a) $T_U V \equiv \pi(U) + \beta Q(U)V$ characterizes the value of following policy $U$ this period, given value function $V$ next period;

   (b) $T_U^n V \equiv T_U(T_U^{n-1}V) = \left(\sum_{t=0}^{n-1} \beta^t Q(U)^t \pi(U)\right) + \beta^n Q(U)^n V$ characterizes the value of following policy $U$ for $n$ periods, given value function $V$ thereafter; and

   (c) $T_U^\infty \equiv \lim_{n \rightarrow \infty} T_U^n V = \sum_{t=0}^{\infty} \beta^t Q(U)^t \pi(U) = (I - \beta Q(U))^{-1} \pi(U)$ characterizes the value of following policy $U$ forever.

12. $\mathcal{U} : \mathbb{R}^m \rightarrow \mathbb{U}$ is the policy update function: $\mathcal{U}(V) \equiv \arg \max_{U \in \mathbb{U}} U' T_U V$.

13. $T : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is the value iteration operator:

   (a) $TV \equiv T_{\mathcal{U}(V)} V$ characterizes the maximum value this period, given value function $V$ next period; and

   (b) $T^n V \equiv T(T^{n-1}V)$ characterizes the maximum value this period, given value function $V$ in $n$ periods.

14. $V^* \in \mathbb{R}^m$ is the optimal value function, implicitly defined as the unique fixed-point solution to Bellman’s equation: $V^* \equiv TV^*$. The $i$th element of $V^*$ is the maximum expected discounted flow utility from state $x_i$.

---

1 To simplify the notation, I assume that each value function corresponds to a unique optimal policy. Note that maximizing the sum of the value function maximizes each element of the value function.
15. $U^* \in \mathbb{U}$ is the optimal policy function, defined as the (assumed unique) policy corresponding to the optimal value function: $U^* \equiv \mathcal{U}(V^*)$.

16. $\| \cdot \|_1$ is the $\ell_1$ norm: $\| x \|_1 \equiv \sum_{i=1}^{m} |\delta'_i x|$.

17. $\| \cdot \|$ is the $\ell_\infty$ norm: $\| x \|_\infty \equiv \max_{i=1}^{m} |\delta'_i x|$.

18. $\| \cdot \|$ is the matrix norm induced by the $\ell_\infty$ norm: $\| A \| \equiv \sup \{ \| Ax \| : \| x \| = 1 \} = \max_{i=1}^{m} \| \delta'_i A \|_1$.

19. Cond$(A) \equiv \| A \| \| A^{-1} \|$ is the condition number of matrix $A$ (see Judd [1998], p. 67).

20. $\lambda(Q)$ is the second-largest eigenvalue modulus of matrix $Q$.

21. $\psi(Q) \equiv \lim_{n \to \infty} \delta'_1 Q^n$ is the stationary distribution associated with stochastic matrix $Q$.

22. $\Delta \equiv I - \iota \delta'_1$ is the $m \times m$ difference operator. Pre-multiplying a length-$m$ vector by $\Delta$ subtracts the first element from every element: $\Delta[x_1, \cdots, x_m]' = [x_1 - x_1, \cdots, x_m - x_1]'$.

3 Relative Value Iteration

3.1 Algorithm

The following proposition establishes that all of the policy-related information resides in the relative value function, $\Delta V$.

**Proposition 1.** (White [1963].) **Differencing the value function does not affect the corresponding policy function:** $\mathcal{U}(\Delta V) = \mathcal{U}(V)$.

This result is intuitive: Changing the value function from $V$ to $\Delta V$ is equivalent to reducing next period’s flow utility by $\delta'_1 V$, regardless of next period’s state and action. Since this utility loss is independent of this period’s action, it doesn’t affect this period’s action.

The relative value iteration algorithm exploits Proposition 1. It proceeds as follows:

1. Initialize $n := 0 \in \mathbb{R}$ and $V_0 := 0 \in \mathbb{R}^m$.

2. Increment $n$.

3. Set $V_n := \Delta T V_{n-1} = (\Delta T)^n V_0 = \Delta T^n V_0$.

---

2 The equivalence of $(\Delta T)^n V_0$ and $\Delta T^n V_0$ follows from Lemma 4 in the appendix.
4. If $\|V_n - V_{n-1}\| \geq \epsilon(1 - \beta)/(2\beta)$ go to 2; otherwise go to 5.

5. Return $U(V_n)$.

The $\epsilon$ in step 4 specifies the convergence tolerance. It’s usually impossible to calculate $U^*$ exactly, so we must make do with an $\epsilon$-optimal policy—a policy that yields within $\epsilon$ of the optimal value when followed forever. The following proposition establishes that relative value iteration indeed yields an $\epsilon$-optimal policy.

**Proposition 2. (Bray, 2018).** The relative value iteration algorithm always returns an $\epsilon$-optimal policy after a finite number of iterations.

### 3.2 Strong Convergence

The only difference between the relative value iteration algorithm and the traditional value iteration algorithm is the presence of $\Delta$ in step 3: whereas traditional value iteration sets $V_n := T^0V_0$, relative value iteration sets $V_n := \Delta T^n V_0$. This subtle change can reduce solution times, as the following proposition indicates.

**Proposition 3. (Morton and Wecker, 1977).** Whereas $\|T^n V - V^*\|$ is $O(\beta^n)$ as $n \to \infty$, $\|\Delta T^n V - V^*\|$ is $O(\beta^n \gamma^n)$ for all $\gamma > \lambda(Q(U^*))$. Thus, if the Markov chain is ergodic under policy $U^*$ then $\lambda(Q(U^*)) < 1$ and the relative value iteration algorithm converges strictly faster than the traditional value iteration algorithm.

I provide the first valid proof of Proposition 3 in the appendix. Morton and Wecker (1977) articulated the result, but their proof is erroneous. Their second theorem requires the set $\{Q(U(T^n V)) : n \geq N\}$ to be “strongly ergodic of order $\lambda$” for some finite $N$; their seventh theorem invokes this result, but with $N = \infty$, which is not allowed.

My proof follows a different line of attack:

1. I use the envelope theorem to establish that

\[
\|\Delta T^{n+2t} V - \Delta T^{n+t} V\| \leq \beta^t \|\Delta \prod_{s=0}^{t-1} Q(U(T^{s+n} V))\| + \epsilon
\]

\[
\leq \beta^t \|\Delta Q(U^*)\| + \beta^t \|\Delta Q(U^*) - \Delta \prod_{s=0}^{t-1} Q(U(T^{s+n} V))\| + \epsilon,
\]

More precisely, policy $U$ is $\epsilon$-optimal if $\|V^* - T^n U\| < \epsilon$ (Puterman, 2005).
for sufficiently large \( n \).

2. I use the binomial theorem to establish that 
\[
\| \Delta Q(U^*)^t - \Delta \prod_{s=0}^{t-1} Q(\mathcal{W}(T^{s+n}V)) \| \leq \epsilon,
\]
for sufficiently large \( n \).

3. I use the Jordan normal form of \( Q(U^*) \) to establish that 
\[
\| \Delta Q(U^*)^t \| \leq (\lambda(Q(U^*)) + \epsilon)^t,
\]
for sufficiently large \( t \).

4. I combine these inequalities to establish that 
\[
\frac{\| \Delta T^{n+2t}V - \Delta T^{n+t}V \|}{\| \Delta T^{n+t}V - \Delta T^nV \|} \leq \beta^t(\lambda(Q(U^*)) + \epsilon)^t,
\]
for sufficiently large \( n \) and \( t \), which implies the result.

While its proof is intricate, the intuition underlying Proposition 3 is straightforward: the relative value function converges faster because it depends on fewer utilities. Whereas the total value function depends on all payoffs not discounted to irrelevance, the relative value function depends only on the payoffs received before the state variables revert back to their limiting distribution; the payoffs received thereafter contribute to the value of each state evenly and thus wash out upon differencing. Accordingly, the relative value function converges not at the rate of discounting, but at the rate of discounting times the rate at which the state variables revert to their stationary distribution. And the rate at which the state variables revert to their stationary distribution is \( \lambda(Q(U^*)) \).

I will illustrate with Bray et al.’s (2018) empirical inventory model. In the model, a supermarket manager regulates a given product’s stock levels by placing daily inventory orders at a supplying distribution center. The dynamic program has three state variables: the store’s inventory level, the distribution center’s inventory level, and tomorrow’s expected demand. Each decision period lasts one day, so the discount factor is \( \beta = 0.9997 \) (which corresponds to an annual discount factor of \( 0.9997^{365} = 0.896 \)). Table 1 reports the quantiles of Bray et al.’s (2018) state transition matrix spectral sub-radii. The median spectral sub-radius is \( \lambda(Q(U^*)) = 0.9775 \); the corresponding product is a 250ml bottle of Lulu brand cashew milk. I will focus henceforth on this median product.

The convergence rate of the value function under value iteration depends on one factor: the rate at which future utilities are discounted, \( \beta \). In general, scaling the value function error by \( \epsilon \)
Table 1

Bray et al. (2018) estimate 246 grocery store inventory dynamic programs. I calculate each dynamic program’s state transition matrix spectral sub-radius and tabulate their three quartiles (0.25, 0.5, and 0.75), two extreme deciles (0.1 and 0.9), and minimum and maximum (0 and 1), by product group. For example, the minimum $\lambda(Q(U^*))$ across detergents is 0.9018, and the median $\lambda(Q(U^*))$ across all products is 0.9775. Although some of the statistics round up to one, all of the spectral sub-radii are less than one.

<table>
<thead>
<tr>
<th>Product Group</th>
<th>0</th>
<th>0.1</th>
<th>0.25</th>
<th>0.5</th>
<th>0.75</th>
<th>0.9</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detergent</td>
<td>0.9018</td>
<td>0.9343</td>
<td>0.9694</td>
<td>0.9814</td>
<td>0.9878</td>
<td>0.9961</td>
<td>1.0000</td>
</tr>
<tr>
<td>Drinks</td>
<td>0.9361</td>
<td>0.9579</td>
<td>0.9693</td>
<td>0.9879</td>
<td>0.9944</td>
<td>0.9973</td>
<td>0.9998</td>
</tr>
<tr>
<td>Oil/Vinegar</td>
<td>0.9165</td>
<td>0.9204</td>
<td>0.9305</td>
<td>0.9593</td>
<td>0.9887</td>
<td>0.9928</td>
<td>0.9964</td>
</tr>
<tr>
<td>Oral Care</td>
<td>0.9198</td>
<td>0.9225</td>
<td>0.9353</td>
<td>0.9648</td>
<td>0.9725</td>
<td>0.9991</td>
<td>1.0000</td>
</tr>
<tr>
<td>Shampoo</td>
<td>0.8996</td>
<td>0.9133</td>
<td>0.9467</td>
<td>0.9738</td>
<td>0.9820</td>
<td>0.9984</td>
<td>0.9994</td>
</tr>
<tr>
<td>Tissues</td>
<td>0.9381</td>
<td>0.9383</td>
<td>0.9408</td>
<td>0.9473</td>
<td>0.9562</td>
<td>0.9762</td>
<td>0.9864</td>
</tr>
<tr>
<td>Toilet Paper</td>
<td>0.9411</td>
<td>0.9512</td>
<td>0.9569</td>
<td>0.9702</td>
<td>0.9945</td>
<td>0.9992</td>
<td>0.9999</td>
</tr>
<tr>
<td>Total</td>
<td>0.8996</td>
<td>0.9313</td>
<td>0.9573</td>
<td>0.9775</td>
<td>0.9898</td>
<td>0.9973</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

requires approximately $\log(\epsilon)/\log(\beta)$ value iteration steps. Thus, scaling the value function error by $10^{-3}$ requires roughly $\log(10^{-3})/\log(0.9997) \approx 23,000$ Bellman contractions. These Bellman contractions compute the expected value of the next $23,000/365 \approx 63$ years’ worth of utilities.

The convergence rate of the relative value function under relative value iteration depends on two factors: the rate at which future utilities are discounted, $\beta$, and the rate at which the state variables regress to their stationary distribution, $\lambda(Q(U^*))$. In general, scaling the relative value function error by $\epsilon$ requires approximately $\log(\epsilon)/\log(\beta \lambda(Q(U^*)))$ relative value iteration steps. Thus, scaling the relative value function error by $10^{-3}$ requires roughly $\log(10^{-3})/\log(0.9997 \cdot 0.9775) \approx 300$ Bellman contractions. These Bellman contractions compute the expected value of the next $300/365 \approx 0.82$ years’ worth of utilities.

We can disregard utilities received thereafter because they are (essentially) independent of the current state variables. Simply put, the system “forgets” the current state after 0.82 years, making all subsequent utilities moot. The Markov chain has such a short memory because, on average, the store turns over its inventory every 10.6 days and the distribution center turns over its inventory every 14.4 days. Figure 1 illustrates this short memory. It plots the distribution of the store’s day-1 inventory from two initial conditions: in the first, the three state variables equal their first quartiles on day 0; in the second, the three state variables equal their third quartiles on day 0. The inventory distributions are different before day 32 but are basically the same by day 256; thus,
both initial conditions have basically the same day-256 expected utility, so the difference between the initial conditions' valuations stems almost entirely from the first 256 utilities. And factoring these 256 utilities requires only 256 Bellman contractions.

4 Relative Policy Iteration

4.1 Algorithm

As the strong convergence analog of traditional value iteration is relative value iteration, the strong convergence analog of traditional policy iteration is relative policy iteration. The relative policy iteration algorithm proceeds as follows:

1. Initialize $n := 0 \in \mathbb{R}$ and $V_0 := 0 \in \mathbb{R}^m$.

2. Increment $n$.

3. Set $U_n := \mathcal{U}(V_{n-1})$.

4. Set $V_n := \Delta T^\infty U_n$.

5. If $\|V_n - V_{n-1}\| \geq \epsilon(1 - \beta)/(2\beta)$ go to 2; otherwise go to 6.

6. Return $\mathcal{U}(V_n)$.

The following proposition establishes that this algorithm yields an $\epsilon$-optimal policy.

**Proposition 4.** (Bray, 2018). The relative policy iteration algorithm always returns an $\epsilon$-optimal policy after a finite number of iterations.

The only difference between relative policy iteration and traditional policy iteration is the presence of $\Delta$ in step 4, whereas traditional policy iteration sets $V_n := T^\infty U_n$, relative policy iteration sets $V_n := \Delta T^\infty U_n$. Again, this $\Delta$ expedites the computation (provided that the underlying Markov chain is ergodic). But whereas the $\Delta$ in relative value iteration reduces the algorithm iteration count without changing the algorithm iteration difficulty, the $\Delta$ in relative policy iteration reduces the algorithm iteration difficulty without changing the algorithm iteration count. It’s easier to implement a relative policy iteration step than a traditional policy iteration step because it’s easier to evaluate $\Delta T^\infty U_n$ than it is to evaluate $T^\infty U_n$. There are three ways to calculate $T^\infty U_n$ and $\Delta T^\infty U_n$, and the $\Delta$ operator helps with each.
I plot the distribution of store inventories in Bray et al.’s (2018) Lulu brand cashew milk dynamic program. Specifically, I depict the day $t$ distribution for $t \in \{0, 1, 2, 4, 8, 16, 32, 64, 128, 256\}$, given that all day-0 state variables equal their first quartiles (for the left panels) or their third quartiles (for the right panels). The distributions are essentially the same by day 256.
4.2 Iterative Policy Evaluation

The first way to calculate $T_U^\infty$ and $\Delta T_U^\infty$ is to evaluate $T_U^n V$ and $\Delta T_U^n V$ for large $n$. The following proposition establishes that $\Delta T_U^n V$ converges to $T_U^\infty$ faster than $T_U^n V$ converges to $T_U^\infty$ (when the underlying Markov chain is ergodic).

**Proposition 5.** (Morton, 1971). Whereas $\|T_U^n V - T_U^\infty\|$ is $O(\beta^n)$ as $n \to \infty$, $\|\Delta T_U^n V - \Delta T_U^\infty\|$ is $O(\beta^n \gamma^n)$ for all $\gamma > \lambda(Q(U))$. Thus, if the Markov chain is ergodic under policy $U$ then $\lambda(Q(U)) < 1$ and relative policy iteration’s policy evaluation step converges strictly faster than traditional policy iteration’s policy evaluation step.

4.3 Forward Simulation

The second way to calculate $T_U^\infty$ and $\Delta T_U^\infty$ is to simulate the future utilities received under policy $U$. Define $\hat{\sigma}_s^n(x_i)$ as the average discounted utility received by $s$ independent sample paths simulated from state $x_i$ for $n$ periods under policy $U$. In other words, set $\hat{\sigma}_s^n(x_i) = s^{-1} \sum_{j=0}^{s-1} \sum_{k=0}^{n-1} \beta^k \delta^{l(j,k)}_i \pi(U)$, where $l(j,0) = i$ and $l(j,k)$ is a multinoulli random variable with probability simplex $\delta^{l(j,k-1)}_i Q(U)$. And define $\hat{\sigma}_s^n = [\hat{\sigma}_s^n(x_1), \ldots, \hat{\sigma}_s^n(x_m)]^T$ as a vector of such simulation estimates.

Like all estimators, the mean square error of $\delta'_i \hat{\sigma}_s^n = \hat{\sigma}_s^n(x_i)$ decomposes into bias and variance components:

$$\text{MSE}(\delta'_i \hat{\sigma}_s^n) \equiv E((\delta'_i \hat{\sigma}_s^n - \delta'_i T_U^\infty)^2),$$

$$= \text{Bias}(\delta'_i \hat{\sigma}_s^n)^2 + \text{Var}(\delta'_i \hat{\sigma}_s^n),$$

where $\text{Bias}(\delta'_i \hat{\sigma}_s^n) \equiv E(\delta'_i \hat{\sigma}_s^n) - \delta'_i T_U^\infty,$

and $\text{Var}(\delta'_i \hat{\sigma}_s^n) \equiv E((\delta'_i \hat{\sigma}_s^n - E(\delta'_i \hat{\sigma}_s^n))^2).$

Equivalent expressions hold for $\delta'_i \Delta \hat{\sigma}_s^n = \hat{\sigma}_s^n(x_i) - \hat{\sigma}_s^n(x_1)$. But this estimator’s bias falls faster with the length of the simulation horizon, as the following proposition establishes.

**Proposition 6.** Whereas $\text{Bias}(\delta'_i \hat{\sigma}_s^n)$ is $O(\beta^n)$ as $n \to \infty$, $\text{Bias}(\delta'_i \Delta \hat{\sigma}_s^n)$ is $O(\beta^n \gamma^n)$ for all $\gamma > \lambda(Q(U))$. Thus, if the Markov chain is ergodic under policy $U$ then $\lambda(Q(U)) < 1$ and the bias in the relative value function estimate vanishes strictly faster than the bias in the total value function.
estimate.

This proposition establishes that we don’t have to simulate as far into the future as we previously thought. To avoid undue bias in \( \hat{\sigma}_s^n \), economists have generally set \( n \) large enough so that \( \beta^n < \epsilon \), which guarantees that “the increment to the value function is sufficiently small due to discounting” (Arcidiacono and Ellickson 2011, p. 383). But bias in \( \hat{\sigma}_s^n \) doesn’t matter; only bias in \( \Delta \hat{\sigma}_s^n \) matters. And guaranteeing negligible \( \Delta \hat{\sigma}_s^n \) bias only requires \( n \) to be large enough to satisfy \( (\beta \lambda(Q(U)))^n < \epsilon \). Shorting the simulation horizon in this fashion speeds up the computation by a factor of \( \log \left( \frac{\beta \lambda(Q(U))}{\log(\beta)} \right) = 1 + \log \left( \frac{\lambda(Q(U))}{\log(\beta)} \right) \).

And there’s an additional benefit: truncating the simulation horizon makes the estimator not only faster but also more accurate, as the following propositions imply.

**Proposition 7.** There exists \( b < 1 \) such that if \( \beta \in [b, 1) \), \( \lambda(Q(U)) < 1 \), and \( \psi(Q(U))^\prime T_U^\infty \neq 0 \) then

\[
\lim_{n \to \infty} \beta^{-2n} (\text{Bias}(\delta^n_s) - \text{Bias}(\delta^n_s)) = -(1 - \beta^2)(\psi(Q(U))^\prime T_U^\infty)^2 < 0
\]

and

\[
\lim_{n \to \infty} \beta^{-2n} (\text{Var}(\delta^n_s) - \text{Var}(\delta^n_s)) = s^{-1} \pi(U)^\prime (\text{diag}(\psi(Q(U))) - \psi(Q(U)))\psi(Q(U))^\prime
\]

\[
(2I - \Delta Q(U)/\beta)^{-1} - I) \pi(U) > 0.
\]

In this case, there exists \( S > 0 \) such that for all \( s > S \),

\[
\lim_{n \to \infty} \beta^{-2n} (\text{MSE}(\delta^n_s) - \text{MSE}(\delta^n_s)) < 0,
\]

and the limiting accuracy of the \( \hat{\sigma}_s^n \) estimator increases with the length of the simulation horizon.

**Proposition 8.** There exists \( b < 1 \) such that if \( \beta \in [b, 1) \) and \( \lambda(Q(U)) < 1 \) then

\[
\lim_{n \to \infty} \beta^{-2n} (\text{Bias}(\delta^n_s) - \text{Bias}(\delta^n_s)) = 0,
\]

and

\[
\lim_{n \to \infty} \beta^{-2n} (\text{Var}(\delta^n_s) - \text{Var}(\delta^n_s)) = 2s^{-1} \pi(U)^\prime (\text{diag}(\psi(Q(U))) - \psi(Q(U)))\psi(Q(U))^\prime
\]

\[
(2I - \Delta Q(U)/\beta)^{-1} - I) \pi(U) > 0.
\]

In this case

\[
\lim_{n \to \infty} \beta^{-2n} (\text{MSE}(\delta^n_s) - \text{MSE}(\delta^n_s)) > 0,
\]

11
and the limiting accuracy of the $\Delta \hat{\sigma}_s^n$ estimator decreases with the length of the simulation horizon.

Proposition 7 states that increasing $n$ decreases $\delta'_i \hat{\sigma}_s^n$’s squared bias by an order of $\beta^{2n}$ but increases $\delta'_i \hat{\sigma}_s^n$’s variance by an order of $\beta^{2n}/s$. Thus, for sufficiently large $s$, lengthening the simulation horizon decreases the estimator’s mean squared error.

Proposition 8 states that this is a mirage. The variance does increase by an order of $\beta^{2n}/s$. But the squared bias that matters—the squared bias in the relative differences—does not decrease by an order of $\beta^{2n}$; it decreases by an order of $\beta^{2n} \lambda(Q(U))^{2n}$, as Proposition 6 establishes. Thus, increasing $n$ beyond $\log(\epsilon)/\log (\beta \lambda(Q(U)))$ yields a variance increase without a commensurate bias decrease: simulating too far into the future yields an estimator that’s both slower and noisier.

4.4 System of Equations

The third way to calculate $T_{U_n}^\infty$ and $\Delta T_{U_n}^\infty$ is to solve linear equations $(I - \beta Q(U_n))T_{U_n}^\infty = \pi(U_n)$ and $(I - \beta \Delta Q(U_n))(\Delta T_{U_n}^\infty) = \Delta \pi(U_n)$. The latter system remains well-conditioned as the discount factor goes to one, but the former system does not. Thus, we can compute $\Delta T_{U_n}^\infty$ to a higher degree of precision than we can calculate $T_{U_n}^\infty$ when discounting is negligible. The following proposition establishes this fact.

**Proposition 9.** If the Markov chain is ergodic under policy $U$ then:

1. $\text{Cond}(I - \beta Q(U))$ is $O\left(\frac{1}{1-\beta}\right)$ as $\beta \to 1$. This indicates that policy iteration’s policy evaluation equations become ill-conditioned as the discount factor approaches unity.

2. $\text{Cond}(I - \beta \Delta Q(U))$ is $O(1)$ as $\beta \to 1$. This indicates that relative policy iteration’s policy evaluation equations remain well-conditioned as the discount factor approaches unity.

5 Application: Estimating Markov Decision Processes

We can leverage §3 and §4’s strong convergence results when estimating Markov decision processes. For example, we can position Rust’s (1987) celebrated dynamic discrete choice problem in §2’s framework by replacing its finite action space with an infinite continuum of choice probabilities.

---

4 Many thanks to an anonymous reviewer for pointing this out.

5 Rust (1994), Aguirregabiria and Mira (2010), and Arcidiacono and Ellickson (2011) survey the literature on estimating Markov decision processes.
Most dynamic program estimators use some version of value iteration or policy iteration; we can accelerate these estimators by refitting them with relative value iteration and relative policy iteration.

### 5.1 Nested Fixed Point

The nested fixed point (NFXP) estimator embeds an optimization problem inside an optimization problem. The outer problem searches across potential structural parameter estimates, and the inner problem solves the dynamic program associated with a given set of structural parameters. Rust (2000, p. 18) recommends using both value iteration and policy iteration to solve the nested dynamic programs. He maintains that "one has to compute the fixed point \([V = TV^*]\) in order to evaluate the likelihood function." But this is incorrect: the estimator’s empirical likelihood function only depends on the policy function, which only depends on the value function’s relative differences. Hence, we can replace Rust’s (2000) value iteration and policy iteration steps with faster relative value and relative policy iteration steps.

Chen (2017) reports that this change “vastly reduce[s] the computation burden” of estimating her acid-rain-mitigation dynamic programs with NFXP. For example, solving each of her dynamic programs requires 10,701 minutes with value iteration, 661 minutes with relative value iteration, and 126 minutes with endogenous value iteration (see Table 2). Endogenous value iteration is a generalization of relative value iteration; it enjoys even stronger rates of convergence when the most persistent state variable is exogenous (see Bray, 2018).

### 5.2 Conditional Choice Probability Estimators

Hotz and Miller’s (1993) infinite-horizon estimator, Aguirregabiria and Mira’s (2002, 2007) nested pseudo-likelihood (NPL) estimators, Pakes et al.’s (2007) “simple” estimator, and Pesendorfer and Schmidt-Dengler’s (2008) least-squares estimator all follow the conditional choice probability (CCP) approach: (i) pre-estimate the agent’s policy function in reduced form; (ii) choose a set of model primitives; (iii) apply a policy iteration step to the pre-estimated policy function under the given model primitives; and (iv) use the updated policy function to evaluate the empirical likelihood (or moment conditions) associated with the given model primitives. We can streamline these CCP estimators by replacing their policy iteration steps with relative policy iteration steps.
Table 2

I reprint these figures from Chen’s (2017) job market paper. They report the number of minutes required to solve each electric utility’s acid-rain-prevention dynamic program with traditional value iteration, relative value iteration, and Bray’s (2018) endogenous value iteration (which is a generalization of relative value iteration).

<table>
<thead>
<tr>
<th>Traditional</th>
<th>Relative</th>
<th>Endogenous</th>
</tr>
</thead>
<tbody>
<tr>
<td>American Electric Power</td>
<td>404</td>
<td>27</td>
</tr>
<tr>
<td>Atlantic City Electric</td>
<td>303</td>
<td>9</td>
</tr>
<tr>
<td>Carolina Power and Light</td>
<td>410</td>
<td>24</td>
</tr>
<tr>
<td>Central Hudson Gas and Electric</td>
<td>301</td>
<td>9</td>
</tr>
<tr>
<td>Central Illinois Light</td>
<td>300</td>
<td>12</td>
</tr>
<tr>
<td>Central Operating</td>
<td>301</td>
<td>19</td>
</tr>
<tr>
<td>Cincinnati Gas and Electric</td>
<td>360</td>
<td>20</td>
</tr>
<tr>
<td>Dairyland Power Coop</td>
<td>304</td>
<td>30</td>
</tr>
<tr>
<td>Dayton Power and Light</td>
<td>366</td>
<td>9</td>
</tr>
<tr>
<td>Detroit Edison</td>
<td>407</td>
<td>19</td>
</tr>
<tr>
<td>Duke Energy</td>
<td>414</td>
<td>41</td>
</tr>
<tr>
<td>Empire District Electric</td>
<td>277</td>
<td>10</td>
</tr>
</tbody>
</table>

For example, suppose the flow utility vector is linear in the structural parameters: \( \pi(U) = M(U)\theta \), where \( M(U) \) is an \( m \times k \) matrix of reward statistics and \( \theta \) is a length-\( k \) vector of primitives to estimate. Aguirregabiria and Mira (2010, p. 51) explain that in this case the CCP bottleneck is calculating the \( m \times k \) matrix \( \tilde{T}_U^\infty \) that satisfies \( \tilde{T}_U^\infty = M(U) + \beta Q(U) \tilde{T}_U^\infty \). The authors recommend computing \( \tilde{T}_U^\infty \) “by successive approximations, iterating [the fixed point equation] which is a contraction mapping”; in other words, they suggest approximating \( \tilde{T}_U^\infty \) with \( \tilde{T}_U^n \theta \) evaluated under large \( n \), where \( \tilde{T}_U^n M \equiv \tilde{T}_U(T_U^{n-1} M) \) and \( \tilde{T}_U M \equiv M(U) + \beta Q(U) M \). Since \( \tilde{T}_U^\infty \theta = T_U^\infty \) and \( (\tilde{T}_U^n \theta) = T_U^n \), Proposition 5 indicates that Aguirregabiria and Mira’s (2010) iterative scheme converges at linear rate \( \beta \). In contrast, Proposition 5 indicates that \( \Delta \tilde{T}_U^n \theta \) converges to \( \Delta T_U^\infty \) at linear rate \( \beta \lambda(Q(U)) \). And Proposition 1 establishes that \( \Delta \tilde{T}_U^\infty \) is all we need.

Leveraging strong convergence in this fashion can shorten estimation times by an order of magnitude (an online supplement demonstrates this with Monte Carlo simulations).

5.3 Simulation Estimators

Hotz et al.’s (1994) and Bajari et al.’s (2007) infinite-horizon simulation estimators are similar to §5.2’s CCP estimators, except they calculate policy valuations via forward simulation. Currently these estimators simulate “far enough into the future so that the discounting renders future terms
past this point irrelevant” (Arcidiacono and Ellickson 2011, p. 381). Therefore, it is generally accepted that the “main drawback of this particular [estimation scheme] is that when $\beta$ is close to 1, many periods must be included to ensure the properties of the estimator are not unduly affected by the finite-horizon approximation” (Hotz et al. 1994, p. 277). However, §4.3’s results indicate that we can obviate much of this work. We don’t have to simulate the process until the flow utilities are discounted to oblivion; we only have to simulate the process until the state space scrambles.

Proposition 8 implies that truncating the simulation horizon should make Hotz et al.’s (1994) and Bajari et al.’s (2007) estimators not only faster but also more accurate. Simulating the utilities received after the state variables reach their stationary distribution increases the estimators’ variance (since the random draws have positive standard deviation) without decreasing the estimators’ bias (since the random draws have basically zero mean). Needlessly increasing the simulation horizon needlessly increases the simulation error.

I will demonstrate with a Monte Carlo simulation study. I generate 600 Rust-style dynamic discrete choice programs, each with 1,000 discrete states and three discrete actions per state. Taking action $a$ in state $x$ in period $t$ yields flow utility $u_1(a, x)\theta_1 + u_2(a, x)\theta_2 + e_t(a)$, where $e_t(a)$ is an independent standard Gumbel random variable that realizes in period $t$. The probability of transitioning from state $x$ to state $x'$, given action $a$, is $q(x'|x,a)$. I set probability vector $[q(x_1|x,a), \cdots, q(x_{1,000}|x,a)]$ to an independent symmetric Dirichlet random variable with concentration parameter one; I set scalars $u_1(a, x)$, $u_2(a, x)$, $\theta_1$, and $\theta_2$ to independent standard normal random variables; and I set discount factor $\beta$ to 0.99.

Following convention, I characterize the agent’s optimal policy with CCPs $\{p(a|x)\}$, where $p(a|x)$ is the probability of choosing action $a$ in state $x$, unconditional on the Gumbel shocks. I calculate these CCPs with relative value iteration.

The objective is to reverse engineer $\theta_1$ and $\theta_2$ from $\{u_i(a,x)\}$, $\{p(a|x)\}$, $\{q(x'|x,a)\}$, and $\beta$. I use Hotz et al.’s (1994) estimator, deploying $s$ samples paths from state $x_1$ action $a_1$, $s$ samples paths from state $x_1$ action $a_2$, and $s$ samples paths from state $x_1$ action $a_3$. I then set $\hat{\theta}_1$ and $\hat{\theta}_2$ to the utility parameters that (i) equate $\log(p(a_2|x_1)) - \log(p(a_1|x_1))$ with the simulation-average $a_2$ discounted utility minus the simulation-average $a_1$ discounted utility, and (ii) equate $\log(p(a_3|x_1)) - \log(p(a_1|x_1))$ with the simulation-average $a_3$ discounted utility minus the simulation-average $a_1$ discounted utility.
I estimate 600 dynamic discrete choice problems with Hotz et al.’s (1994) and Bajari et al.’s (2007) simulation estimators. I estimate 200 problems with \( s = 100 \) sample paths, 200 with \( s = 1,000 \) sample paths, and 200 with \( s = 10,000 \) sample paths. I estimate each problem with both the traditional simulation horizon, \( n_1 \equiv \log(\epsilon)/\log(\beta) \), and my shorter simulation horizon, \( n_2 \equiv \log(\epsilon)/\log(\beta \lambda(Q(U))) \), where \( \epsilon \equiv 10^{-6} \) and \( \beta \equiv 0.99 \). I tabulate the geometric means of the estimation time ratios under \( n_1 \) and \( n_2 \) and the geometric means of the estimation error ratios under \( n_1 \) and \( n_2 \). I measure the estimation error with the Euclidean distance between \([\theta_1, \theta_2]\) and \([\hat{\theta}_1, \hat{\theta}_2]\).

<table>
<thead>
<tr>
<th>( s )</th>
<th>100</th>
<th>1,000</th>
<th>10,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Ratio</td>
<td>229.47</td>
<td>235.13</td>
<td>257.22</td>
</tr>
<tr>
<td></td>
<td>(1.90)</td>
<td>(1.83)</td>
<td>(1.84)</td>
</tr>
<tr>
<td>Error Ratio</td>
<td>2.39</td>
<td>2.88</td>
<td>3.36</td>
</tr>
<tr>
<td></td>
<td>(0.28)</td>
<td>(0.28)</td>
<td>(0.33)</td>
</tr>
</tbody>
</table>

discounted utility (see Aguirregabiria and Mira, 2010, p. 53)[6] I estimate 200 dynamic programs with \( s = 100 \), 200 dynamic programs with \( s = 1,000 \), and 200 dynamic programs with \( s = 10,000 \). For each, I use both the traditional simulation horizon, \( n_1 \equiv \log(\epsilon)/\log(\beta) \), and my shorter simulation horizon, \( n_2 \equiv \log(\epsilon)/\log(\beta \lambda(Q(U))) \), where \( \epsilon \equiv 10^{-6} \) and \( Q(U) \) is the state transition matrix implied by \( \{p(a|x)\} \) and \( \{q(x'|x,a)\} \).

For each dynamic program, I calculate the estimation time under horizon \( n_1 \) divided by the estimation time under horizon \( n_2 \), and I calculate the estimation error under horizon \( n_1 \) divided by the estimation error under horizon \( n_2 \). Table ?? reports these ratios’ geometric means. Using the traditional simulation horizon is between 229 and 257 times slower, and between 2.39 and 3.36 times less accurate.

I’ll close with two technical points. First, Zobel and Scherer (2005, p. 133) list several upper bounds that are easier to compute than \( \lambda(Q(U)) \). And second, Arcidiacono and Miller (2011, p. 1834) provide a means to replace an agent’s observed policy with one that is more conducive to estimation. We can use this technique to further shorten the simulation horizon: rather than set the sample path length to \( \log(\epsilon)/\log(\beta \lambda(Q(U))) \), where \( U \) is the observed policy, we can set the sample path length to \( \log(\epsilon)/\log(\beta \lambda(Q(\tilde{U}))) \), where \( \tilde{U} \) is any policy we desire.

---

[6] Thus, my instrumental variables are dummy variables that indicate the initial state-action pair.
6 Application: Calculating Dynamic Equilibria

In the literature on dynamic games, (i) the canonical equilibrium concept is Maskin and Tirole’s (1988a, 1988b) Markov perfect equilibrium, (ii) the canonical application is Ericson and Pakes’s (1995) market entry problem, and (iii) the canonical solution method is Pakes and McGuire’s (1994) Gauss-Jacobi algorithm.

Pakes and McGuire’s algorithm is just value iteration run in parallel across agents: in iteration $n$, each agent implements a value iteration step, under the assumption that the other agents will follow their iteration $n-1$ policies. To exploit strong convergence, I difference the value functions after each Bellman contraction; doing so transforms the algorithm from a multi-agent version of value iteration to a multi-agent version of relative value iteration. To demonstrate, I solve Doraszelski and Judd’s (2012) discrete-time version of Ericson and Pakes’s (1995) Markov perfect equilibrium with both Pakes and McGuire’s (1994) traditional algorithm (which calculates value functions) and my strong convergence analog (which calculates relative value functions). Table ?? reports the number of Bellman contractions that both these approaches require under various discount factors. As you can see, taking $\beta$ to one breaks Pakes and McGuire’s value-iteration-based algorithm but not my relative-value-iteration-based algorithm.

My algorithm converges even faster when I reformulate the problem to better leverage strong convergence. Porteus (1975) establishes that a dynamic program with flow utility vector $\pi(U)$ and state transition matrix $Q(U)$ has the same optimal policy as a dynamic program with flow utility vector $\tilde{\pi}(U) = (1-q(U))^{-1}\pi(U)$ and state transition matrix $\tilde{Q}(U) = (1-q(U))^{-1}(Q(U)-q(U)/\beta I)$, where $q(U)$ is the smallest diagonal element of $Q(U)$. However, if $q(U) > 0$ then $\lambda(\tilde{Q}(U)) < \lambda(Q(U))$, and the relative value function converges faster under the alternative problem. Table ?? illustrates that Porteus’s (1975) refinement accelerates my algorithm’s convergence rate by another 26%.

---

7 Ericson and Pakes (1995, p. 68) prove that their model is ergodic, which makes it a candidate for strong convergence.
8 Following Borkovsky et al. (2010), I set the number of agents to two.
9 Relative value iteration can accommodate $\beta = 1$ when the Markov chain is ergodic (Morton and Wecker 1977). This property appears to extend to the multi-agent case.
Table 4

I report the number of Bellman contractions required to calculate Doraszelski and Judd’s (2012) discrete-time Markov perfect equilibrium under five discount factors: $\beta \in \{0.9, 0.99, 0.999, 0.9999, 1\}$. I solve the equilibrium with Pakes and McGuire’s (1994) multi-agent value iteration algorithm, an equivalent multi-agent relative value iteration algorithm, and the multi-agent relative value iteration algorithm with Porteus’s (1975) accelerator.

<table>
<thead>
<tr>
<th>0.9</th>
<th>0.99</th>
<th>0.999</th>
<th>0.9999</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value Iteration</td>
<td>246</td>
<td>2,430</td>
<td>24,448</td>
<td>245,845</td>
</tr>
<tr>
<td>Relative Value Iteration</td>
<td>244</td>
<td>639</td>
<td>964</td>
<td>1,001</td>
</tr>
<tr>
<td>Accelerated Relative Value Iteration</td>
<td>221</td>
<td>483</td>
<td>701</td>
<td>725</td>
</tr>
</tbody>
</table>

7 Conclusion

Strong convergence’s raison d’être is the high-frequency problem with low persistence. For example, suppose (i) the period length is one day, and (ii) the system equilibrates within a year. Assumption (i) implies that the per-period discount factor is around $\beta = 0.9998$ (for a per-anum discount factor of $0.9998^{365} = .93$), which implies that value iteration requires around $\log(10^{-6})/\log(0.9998) \approx 69,000$ Bellman contractions. Assumption (ii), in contrast, implies that relative value iteration requires only around 365 Bellman contractions.

But you can use the technique for any dynamic program, as there’s no downside. Simplicity is not sacrificed because leveraging strong convergence requires only one additional line of code: `value.fn = value.fn - value.fn[1]`. And information is not sacrificed because deriving the value function from the relative value function requires only one additional Bellman contraction, since $V^* = \Delta V^* + (1 - \beta)^{-1}(T\Delta V^* - \Delta V^*)$.\[10\]

Proofs

**Lemma 1.** $\Delta^2 = \Delta$.

*Proof.* $(I - \Delta)^2 = \nu\delta_1'\nu\delta_1' = \nu\delta_1' = I - \Delta$. This implies the result.\[\]

**Lemma 2.** $Q(U)^n(I - \Delta) = (I - \Delta)$ for all $n \in \mathbb{N}$.

\[10\] Lemmas 1 and 2 in the appendix imply $(I - \Delta)(T\Delta V^* - \Delta V^*) = T\Delta V^* - \Delta V^*$. And Proposition 1 implies $T^n\Delta V^* = \sum_{t=0}^{n-1} \beta^t Q(U)^t\pi(U) + \beta^n Q(U)^n\Delta V^*$. With this, Lemma 2 implies $(T^{n+1} - T^n)\Delta V^* = \beta^n Q(U)^n\pi(U) - \beta^n Q(U)^n(I - \beta Q(U))\Delta V^* = \beta^n Q(U)^n T\Delta V^* - \Delta V^* = (I - \Delta)(T\Delta V^* - \Delta V^*) = \Delta V^* + (1 - \beta)^{-1}(T\Delta V^* - \Delta V^*) = \Delta V^* + \sum_{n=0}^{\infty} \beta^n (T\Delta V^* - \Delta V^*) = \Delta V^* + (1 - \beta)^{-1}(T\Delta V^* - \Delta V^*)$. And this implies $V^* = \Delta V^* + \sum_{n=0}^{\infty} (T^{n+1} - T^n)\Delta V^* = \Delta V^* + \sum_{n=0}^{\infty} \beta^n (T\Delta V^* - \Delta V^*) = \Delta V^* + (1 - \beta)^{-1}(T\Delta V^* - \Delta V^*)$.\[10\]
Proof. The rows of a stochastic matrix sum to one. This implies $Q(U)\iota = \iota$, which implies $Q(U)(I - \Delta) = Q(U)\iota \delta_1^t = \iota \delta_1^t = (I - \Delta)$. By induction, this implies the result.

Lemma 3. $\Delta Q(U)\Delta = \Delta Q(U)$.

Proof. Lemmas 1 and 2 imply $\Delta Q(U)(I - \Delta) = \Delta(I - \Delta) = \Delta - \Delta = 0$.

Proposition 1. (White, 1963). Differencing the value function does not affect the corresponding policy function: $\mathcal{U}(\Delta V) = \mathcal{U}(V)$.

Proof. Lemma 2 implies $\mathcal{U}(\Delta V) = \arg \max_{U \in \mathcal{U}} \iota'(\pi(U) + \beta Q(U)\Delta V) = \arg \max_{U \in \mathcal{U}} \iota'(\pi(U) + \beta Q(U)\Delta V + \beta Q(U)(I - \Delta)V) = \arg \max_{U \in \mathcal{U}} \iota'(\pi(U) + \beta Q(U)V)$.

Lemma 4. $\Delta T \Delta = \Delta T$.

Proof. Proposition 1 and Lemma 3 imply $\Delta T \Delta V = \Delta \pi(\mathcal{U}(\Delta V)) + \beta \Delta Q(\mathcal{U}(\Delta V)) \Delta V = \Delta \pi(\mathcal{U}(V)) + \beta \Delta Q(\mathcal{U}(V)) V = \Delta TV$.

Proposition 2. (Bray, 2018). The relative value iteration algorithm always returns an $\epsilon$-optimal policy after a finite number of iterations.

Proof. This is a special case of Bray’s (2018) fourth proposition.

Lemma 5. For all $\epsilon > 0$ there exists $N(\epsilon) > 0$ such that $\|\Delta Q(U^*)^t\| \leq (\lambda(Q(U^*)))^t + \epsilon$ for all $t > N(\epsilon)$.

Proof. This follows from the Jordan normal form of $Q(U^*)$ and the fact that $\iota$ resides both in the null space of $\Delta$ and the eigenspace corresponding to $Q(U^*)$’s largest eigenvalue.

Lemma 6. For all $t > 0$ and $\epsilon > 0$ there exists $N(t, \epsilon) > 0$ such that for all $n > N(t, \epsilon)$

$$\frac{\|\Delta T^{n+2t}V - \Delta T^{n+t}V\|}{\|\Delta T^{n+t}V - \Delta T^n V\|} \leq \beta^t \exists \Delta \prod_{s=0}^{t-1} Q(\mathcal{U}(T^{n+s}V)) + \epsilon.$$
Proof. The envelope theorem implies
\[
\frac{\partial}{\partial V} TV = \frac{\partial}{\partial V} \left( \max_{U \in \mathcal{U}} \pi(U) + \beta Q(U) V \right) = \frac{\partial}{\partial V} (\pi(U) + \beta Q(U) V) \big|_{U = \mathcal{U}(V)} = \beta Q(\mathcal{U}(V)).
\]

By induction this implies \( \frac{\partial}{\partial V} T^{t+n}V = \beta \prod_{s=0}^{t-1} Q(\mathcal{U}(T^sV)) \). This, in turn, implies
\[
T^t T^{n+t}V = T^t T^nV + \beta \prod_{s=0}^{t-1} Q(\mathcal{U}(T^sT^nV)) (T^{n+t}V - T^nV) + o(T^{n+t}V - T^nV).
\]

With this, Lemma 3 implies
\[
\Delta T^t T^{n+t}V = \Delta T^t T^nV + \beta \Delta \prod_{s=0}^{t-1} Q(\mathcal{U}(T^sT^nV)) (\Delta T^{n+t}V - \Delta T^nV) + \Delta o(T^{n+t}V - T^nV).
\]

Since \( \lim_{n \to \infty} T^{n+t}V - T^nV = V^* - V^* = 0 \), this implies the result. \( \square \)

**Lemma 7.** For all \( t > 0 \) and \( \epsilon > 0 \) there exists \( N(t, \epsilon) > 0 \) such that \( \left\| \Delta Q(U^*)^t - \Delta \prod_{s=0}^{t-1} Q(\mathcal{U}(T^sT^nV)) \right\| \leq \epsilon \) for all \( n > N(t, \epsilon) \).

**Proof.** Choose \( \xi > 0 \) small enough so that \( 2t \sum_{i=1}^{t-1} \binom{t-1}{i} \left\| Q(U^*) \right\|^{t-1-i} \xi^{i+1} < \epsilon \). And choose \( N(t, \epsilon) \)

large enough so that \( \left\| Q(\mathcal{U}(T^nV)) - Q(U^*) \right\| \leq \xi \) for all \( n > N(t, \epsilon) \) (doing so is possible because
\[
\lim_{n \to \infty} \mathcal{V}(T^n V) = U^* \text{ and function } Q \text{ is continuous). Now, choosing } n > N(t, \epsilon) \text{ yields}
\]
\[
\left\| \Delta Q(U^*)^t - \Delta \prod_{s=0}^{t-1} Q(\mathcal{V}(T^{s+n} V)) \right\| \leq \|\Delta\| \left\| Q(U^*)^t - \prod_{s=0}^{t-1} Q(\mathcal{V}(T^{s+n} V)) \right\|
\]
\[
= 2 \left\| Q(U^*)^t - \prod_{s=0}^{t-1} Q(\mathcal{V}(T^{s+n} V)) \right\|
\]
\[
= 2 \left\| \prod_{s=0}^{t-1} \sum_{i=0}^{t-1} \left( 1(s \neq i)Q(U^*) + (Q(\mathcal{V}(T^{s+n} V)) - Q(U^*)) \right) \right\|
\]
\[
\leq 2 \sum_{s=0}^{t-1} \prod_{i=0}^{t-1} \left( 1(s \neq i)\|Q(U^*)\| + \|Q(\mathcal{V}(T^{s+n} V)) - Q(U^*)\| \right)
\]
\[
\leq 2 \sum_{s=0}^{t-1} \prod_{i=0}^{t-1} \left( 1(s \neq i)\|Q(U^*)\| + \xi \right)
\]
\[
= 2t\xi \left( \|Q(U^*)\| + \xi \right)^{t-1}
\]
\[
\leq 2t \sum_{i=1}^{t-1} \left( \frac{t-1}{i} \right) \|Q(U^*)\|^{t-1-i}\xi^{i+1}
\]
\[
\leq \epsilon.
\]

\[\square\]

**Proposition 3.** \textbf{(Morton and Wecker, 1977).} Whereas \(\|T^n V - V^*\| \text{ is } O(\beta^n) \text{ as } n \to \infty,\)
\[\|\Delta T^n V - V^*\| \text{ is } O(\beta^n \gamma^n) \text{ for all } \gamma > \lambda(Q(U^*))\). Thus, if the Markov chain is ergodic under policy \(U^* \text{ then } \lambda(Q(U^*)) < 1\) and the relative value iteration algorithm converges strictly faster than the traditional value iteration algorithm.

**Proof.** Let \(N_5(\epsilon), N_6(t, \epsilon), \text{ and } N_7(t, \epsilon)\) represent the thresholds characterized in Lemmas 5, 6, and 7. Choose \(\epsilon, t, \text{ and } n\) such that \(\epsilon \in (0, 1 - \beta \lambda(Q(U^*)))\), \(t > N_6(\epsilon/2)\), and \(n > \max \left( N_7(t, \xi(\epsilon)), N_7(t, \xi(\epsilon)) \right)\), \(\xi(\epsilon) = (1 + \beta^t)^{-1} \left( \beta^t(\lambda(Q(U^*)) + \epsilon)^t - \beta^t(\lambda(Q(U^*)) + \epsilon/2)^t \right)\). With this, Lemmas 5, 6, and
imply

\[
\frac{\|\Delta T^{n+2t} V - \Delta T^{n+t} V\|}{\|\Delta T^{n+t} V - \Delta T^n V\|} \leq \beta^t \left\| \Delta \prod_{s=0}^{t-1} Q\left( \mathcal{U} \left( T^{s+n} V \right) \right) \right\| + \xi(t)
\]

\[
\leq \beta^t \left\| \Delta Q(U^*)^t \right\| + \beta^t \left\| \Delta Q(U^*)^t - \Delta \prod_{s=0}^{t-1} Q(\mathcal{U} \left( T^{s+n} V \right) \right\| + \xi(t)
\]

\[
\leq \beta^t (\lambda(Q(U^*)) + \epsilon/2)^t + \beta^t \xi(t) + \xi(t)
\]

\[
= \beta^t (\lambda(Q(U^*)) + \epsilon)^t.
\]

By induction this implies

\[
\frac{\|\Delta T^{n+(i+1)t} V - \Delta T^{n+it} V\|}{\|\Delta T^{n+it} V - \Delta T^n V\|} \leq \beta^it (\lambda(Q(U^*)) + \epsilon)^it,
\]

which implies

\[
\frac{\|V^* - \Delta T^{n+jt} V\|}{\|\Delta T^{n+jt} V - \Delta T^n V\|} = \frac{\left\| \sum_{i=j}^{\infty} \Delta T^{n+(i+1)t} V - \Delta T^{n+it} V \right\|}{\|\Delta T^{n+it} V - \Delta T^n V\|}
\]

\[
\leq \sum_{i=j}^{\infty} \frac{\|\Delta T^{n+(i+1)t} V - \Delta T^{n+it} V\|}{\|\Delta T^{n+it} V - \Delta T^n V\|}
\]

\[
\leq \sum_{i=j}^{\infty} \beta^it (\lambda(Q(U^*)) + \epsilon)^it
\]

\[
= \beta^it (\lambda(Q(U^*)) + \epsilon)^it \frac{1 - \beta (\lambda(Q(U^*)) + \epsilon)}{1 - \beta (\lambda(Q(U^*)) + \epsilon)}
\]

which implies \(\|V^* - \Delta T^{n+jt} V\|\) is \(O(\beta^it (\lambda(Q(U^*)) + \epsilon)^it)\) as \(j \to \infty\), which implies \(\|V^* - \Delta T^n V\|\) is \(O(\beta^n (\lambda(Q(U^*)) + \epsilon)^n)\) as \(n \to \infty\).

Proposition 4. (Bray, 2018). The relative policy iteration algorithm always returns an \(\epsilon\)-optimal policy after a finite number of iterations.

Proof. This is a special case of Bray’s (2018) fourth proposition.

Proposition 5. (Morton, 1971). Whereas \(\|T^n_U V - T^n_U^\infty\|\) is \(O(\beta^n)\) as \(n \to \infty\), \(\|\Delta T^n_U V - \Delta T^n_U^\infty\|\) is \(O(\beta^n \gamma^n)\) for all \(\gamma > \lambda(Q(U))\). Thus, if the Markov chain is ergodic under policy \(U\) then \(\lambda(Q(U)) < 1\) and relative policy iteration’s policy evaluation step converges strictly faster than traditional policy iteration’s policy evaluation step.
Proof. Consider an auxiliary problem in which \( \mathbb{U} = \{U\} \). In this case \( \mathcal{Z}(V) = U \), and thus 
\[ T_U V = T_{\mathcal{Z}(V)} V = TV, \]
and thus \( T_U^i V = T^i V \). With this, Proposition 3 implies the result.

**Proposition 6.** Whereas \( \text{Bias}(\delta_i^t \tilde{\sigma}_s^n) \) is \( O(\beta^n) \) as \( n \to \infty \), \( \text{Bias}(\delta_i^t \Delta \tilde{\sigma}_s^n) \) is \( O(\beta^n \gamma^n) \) for all \( \gamma > \lambda(Q(U)) \). Thus, if the Markov chain is ergodic under policy \( U \) then \( \lambda(Q(U)) < 1 \) and the bias in the relative value function estimate vanishes strictly faster than the bias in the total value function estimate.

**Proof.** Proposition 5 implies the result, since

\[
\text{Bias}(\delta_i^t \tilde{\sigma}_s^n) = \left( \sum_{k=0}^{n-1} \beta^k \mathbb{E} \left( \delta_i^{t(0,k)} \right) \pi(U) \right) - \delta_i^t T_U^\infty
\]

\[
= \left( \sum_{k=0}^{n-1} \beta^k \delta_i^t Q(U)^k \pi(U) \right) - \delta_i^t T_U^\infty
\]

\[
= \delta_i^t \left( T_U^n - T_U^\infty \right),
\]

and, similarly, \( \text{Bias}(\delta_i^t \Delta \tilde{\sigma}_s^n) = \delta_i^t \left( \Delta T_U^n - \Delta T_U^\infty \right) \).

**Proposition 7.** There exists \( b < 1 \) such that if \( \beta \in [b, 1) \), \( \lambda(Q(U)) < 1 \), and \( \psi(Q(U))' T_U^\infty \neq 0 \) then

\[
\lim_{n \to \infty} \beta^{-2n} (\text{Bias}(\delta_i^t \tilde{\sigma}_s^{n+1})^2 - \text{Bias}(\delta_i^t \tilde{\sigma}_s^n)^2) = -(1 - \beta^2)(\psi(Q(U))' T_U^\infty)^2 < 0
\]

and

\[
\lim_{n \to \infty} \beta^{-2n} (\text{Var}(\delta_i^t \tilde{\sigma}_s^{n+1}) - \text{Var}(\delta_i^t \tilde{\sigma}_s^n)) = s^{-1} \pi(U)' \left( \text{diag}(\psi(Q(U))' - \psi(Q(U))) - \psi(Q(U))' \pi(U) \right) (2(I - \Delta Q(U)/\beta)^{-1} - I) \pi(U) > 0.
\]

In this case, there exists \( S > 0 \) such that for all \( s > S \),

\[
\lim_{n \to \infty} \beta^{-2n} (\text{MSE}(\delta_i^t \tilde{\sigma}_s^{n+1}) - \text{MSE}(\delta_i^t \tilde{\sigma}_s^n)) < 0,
\]

and the limiting accuracy of the \( \tilde{\sigma}_s^n \) estimator increases with the length of the simulation horizon.

**Proof.** First, the proof of Proposition 6 establishes that \( \text{Bias}(\delta_i^t \tilde{\sigma}_s^n) = \delta_i^t T_U^n - \delta_i^t T_U^\infty = -\beta^n \delta_i^t Q(U)^n T_U^\infty \),

23


which implies
\[
\lim_{n \to \infty} \beta^{-2n}(\text{Bias}(\hat{\sigma}_s^{n+1})^2 - \text{Bias}(\hat{\sigma}_s^{n})^2) = \lim_{n \to \infty} \beta^2(\delta'_i Q(U)^{n+1} T_U^{\infty})^2 - (\delta'_i Q(U)^n T_U^{\infty})^2 \\
= -(1 - \beta^2)(\psi(U)' T_U^{\infty})^2.
\]

Second, direct manipulation yields
\[
\text{Cov}(\delta_{\ell(n-t)}, \delta_{\ell(n)}) = E(\delta_{\ell(n-t)} \delta'_{\ell(n)}) - E(\delta_{\ell(n-t)}) E(\delta'_{\ell(n)}) \\
= \left( \sum_{j=1}^{m} \sum_{k=1}^{m} \delta'_i Q(U)^{n-t} \delta_j (\delta'_j Q(U)' \delta_k) \delta_j' \delta_k \right) - Q(U)^{n-t'} \delta_i \delta_i' Q(U)^n \\
= \left( \sum_{j=1}^{m} \delta_j \delta'_j Q(U)^{n-t} \delta_j' Q(U)' \delta_k \delta_k' \right) - Q(U)^{n-t'} \delta_i \delta_i' Q(U)^n \\
= \left( \sum_{j=1}^{m} \delta_j \delta'_j Q(U)^{n-t} \delta_j Q(U)' \right) - Q(U)^{n-t'} \delta_i \delta_i' Q(U)^n \\
= \left( \text{diag}(\delta'_i Q(U)^{n-t}) - Q(U)^{n-t'} \delta_i \delta_i' Q(U)^{n-t} \right) Q(U)^t.
\]

Third, Lemma \ref{lem:lemma2} implies
\[
\left( \text{diag}(\delta'_i Q(U)^{n-t}) - Q(U)^{n-t'} \delta_i \delta_i' Q(U)^{n-t} \right) (I - \Delta) \\
= \text{diag}(\delta'_i Q(U)^{n-t}) (I - \Delta) - Q(U)^{n-t'} \delta_i \delta_i' (I - \Delta) \\
= \text{diag}(\delta'_i Q(U)^{n-t}) \delta_1' - Q(U)^{n-t'} \delta_i \delta_i' \delta_1' \\
= Q(U)^{n-t'} \delta_i \delta_1' - Q(U)^{n-t'} \delta_i' \delta_1' \\
= 0.
\]

Fourth, the second and third points imply that
\[
\text{Cov}(\delta_{\ell(n-t)}, \delta_{\ell(n)}) = \left( \text{diag}(\delta'_i Q(U)^{n-t}) - Q(U)^{n-t'} \delta_i \delta_i' Q(U)^{n-t} \right) \Delta Q(U)^t.
\]
Fifth, set $b > \lambda(Q(U))$, so that $\beta > \lambda(Q(U))$. With this, Lemmas 3 and 5 imply that $(I - \Delta Q(U)/\beta)^{-1} = \sum_{t=0}^{\infty} \left( \frac{\Delta Q(U)}{\beta} \right)^t = \sum_{t=0}^{\infty} \Delta Q(U)^t / \beta^t$ exists, which implies

$$\lim_{n \to \infty} \frac{\operatorname{Cov}(\delta_{t(n)}, \delta_{t(n)})}{\beta^t} = \lim_{n \to \infty} \frac{\sum_{t=0}^{n} \left( \operatorname{diag}(\delta_i Q(U)^{n-t}) - Q(U)^{n-t} \delta_i Q(U)^{n-t} \right) \Delta Q(U)^t / \beta^t}{\beta^t} = (\operatorname{diag}(\psi(U)) - \psi(U)\psi(U)) \sum_{t=0}^{\infty} \Delta Q(U)^t / \beta^t = (\operatorname{diag}(\psi(U)) - \psi(U)\psi(U))(I - \Delta Q(U)/\beta)^{-1}.$$

Sixth, this implies

$$\lim_{n \to \infty} \beta^{-2n} \left( \operatorname{Var}(\delta_i^{s+1}) - \operatorname{Var}(\delta_i^{s}) \right) = \lim_{n \to \infty} \beta^{-2n} s^{-1} \left( \operatorname{Var} \left( \sum_{k=0}^{n} \beta^k \delta_{\ell(0,k)} \pi(U) \right) - \operatorname{Var} \left( \sum_{k=0}^{n-1} \beta^k \delta_{\ell(0,k)} \pi(U) \right) \right) = s^{-1} \lim_{n \to \infty} \pi(U) \left( -\operatorname{Cov}(\delta_{t(n)}, \delta_{t(n)}) + 2 \sum_{t=0}^{n} \beta^{-t} \operatorname{Cov}(\delta_{t(n)}, \delta_{t(n-t)}) \right) \pi(U) = s^{-1} \pi(U) \left( \operatorname{diag}(\psi(U)) - \psi(U)\psi(U) \right) \left( 2(I - \Delta Q(U)/\beta)^{-1} - I \right) \pi(U).$$

Finally, I establish that this quantity is positive for sufficiently large $\beta$. Lemmas 3 and 5 imply that $(I - \Delta Q(U)/\beta)^{-1}$ is differentiable in $\beta$ at $\beta = 1$. So it suffices to consider the $\beta = 1$ case. Clearly $\lim_{n \to \infty} \operatorname{Var}(\delta_i^{s+1}) = \infty$ when $\beta = 1$, and thus $\lim_{n \to \infty} \operatorname{Var}(\delta_i^{s+1}) - \operatorname{Var}(\delta_i^{s}) > 0$ when $\beta = 1$.

**Proposition 8.** There exists $b < 1$ such that if $\beta \in [b, 1)$ and $\lambda(Q(U)) < 1$ then

$$\lim_{n \to \infty} \beta^{-2n} (\operatorname{Bias}(\delta_i^{s+1}) - \operatorname{Bias}(\delta_i^{s})) = 0,$$

and

$$\lim_{n \to \infty} \beta^{-2n} \left( \operatorname{Var}(\delta_i^{s+1}) - \operatorname{Var}(\delta_i^{s}) \right) = 2s^{-1} \pi(U) \left( \operatorname{diag}(\psi(Q(U))) - \psi(Q(U))\psi(Q(U)) \right) \left( 2(I - \Delta Q(U)/\beta)^{-1} - I \right) \pi(U) > 0.$$

In this case

$$\lim_{n \to \infty} \beta^{-2n} \left( \operatorname{MSE}(\delta_i^{s+1}) - \operatorname{MSE}(\delta_i^{s}) \right) > 0,$$

25
and the limiting accuracy of the $\Delta \hat{\sigma}_s^n$ estimator decreases with the length of the simulation horizon.

Proof. This follows from Propositions 6 and 7. The variance term increases by a factor of two because $\delta_i' \Delta \hat{\sigma}_s^n$ depends on both the sample paths deployed from state $x_i$ and the sample paths deployed from state $x_1$.

\[ \text{Proposition 9. If the Markov chain is ergodic under policy } U \text{ then:} \]

1. $\text{Cond}(I - \beta Q(U)) = O\left(\frac{1}{1-\beta}\right)$ as $\beta \to 1$. This indicates that policy iteration’s policy evaluation equations become ill-conditioned as the discount factor approaches unity.

2. $\text{Cond}(I - \beta \Delta Q(U)) = O(1)$ as $\beta \to 1$. This indicates that relative policy iteration’s policy evaluation equations remain well-conditioned as the discount factor approaches unity.

Proof. With the $(I - \beta Q(U))^{-1} = \sum_{t=0}^{\infty} \beta^t Q(U)^t$ identity, it’s straightforward to show that (i) $\|I - \beta Q(U)\| = 1 + \beta - 2\beta \min_{i=1}^m \delta_i' Q(U) \delta_i$, (ii) $\left\| (I - \beta Q(U))^{-1} \right\| = (1 - \beta)^{-1}$, and (iii) $\|I - \beta \Delta Q(U)\| \leq 1 + \beta \max_{i=1}^m \| (\delta_i - \delta_1)' Q(U) \|_1$. Bounding $\lim_{\beta \to 1} \left\| (I - \beta \Delta Q(U))^{-1} \right\|$ is more difficult. To do so, choose $\epsilon > 0$ such that $\epsilon \leq (1 - \lambda(Q(U)))/2$. And choose $n > N(\epsilon)$, where $N(\epsilon)$ is defined in Lemma 5. With this, Lemmas 3 and 5 imply

\[ \left\| (I - \beta \Delta Q(U))^{-1} \right\| = \left\| \sum_{t=0}^{\infty} (\beta \Delta Q(U))^t \right\| \]

\[ = \left\| \sum_{t=0}^{\infty} \beta^t \Delta Q(U)^t \right\| \]

\[ \leq \sum_{t=0}^{\infty} \left\| \beta^t \Delta Q(U)^t \right\| \]

\[ \leq \sum_{t=0}^{n} \left\| \beta^t \Delta Q(U)^t \right\| + \sum_{t=n}^{\infty} (\beta \lambda(Q(U)) + \epsilon)^t \]

\[ \leq \sum_{t=0}^{n} \left\| \beta^t \Delta Q(U)^t \right\| + (1 - \beta \lambda(Q(U)) - \epsilon)^{-1} \]

\[ \leq \sum_{t=0}^{n} \left\| \Delta Q(U)^t \right\| + \left( \frac{(1 - \lambda(Q(U)))/2}{(1 - \lambda(Q(U)))} \right)^{-1}, \]

which is independent of $\beta$. \qed
References


