

Policy-Tree Descendant Algorithm for Solving Consumption–Saving Problems with a Safety Net[†]

Wending Liu[‡] and Fedor Iskhakov[§]

October, 2026

Abstract

This paper develops a new algorithm for solving life-cycle consumption–saving problems that feature social insurance in the form of an exogenous consumption floor. Despite the proliferation of this setup in the literature, the fact that such safety-net programs render the dynamic optimization problem non-convex and the optimal consumption policy discontinuous is generally overlooked. Applicable to a class of piecewise-affine consumption–saving models, the policy-tree descendant (PTD) algorithm developed in this paper accurately tracks the discontinuity points of the policy function and computes it using a graph-theoretic representation without the need to discretize the state space. In a series of simulation exercises, we compare the performance of the PTD algorithm to the existing endogenous grid methods for non-convex dynamic optimization problems. Being grid-free, PTD pays its computational cost per simulated path rather than per parameter configuration. Apart from the cost of numerical integration, PTD’s computational cost does not grow with the dimension of the exogenous state space. In our experiments the accuracy of the PTD solver matches the endogenous grid benchmarks, while in the empirically relevant regime of heterogeneous agents—few simulated paths per parameter configuration—the PTD methods can be more than two orders of magnitude faster.

Keywords: dynamic programming, life-cycle model, safety net, consumption floor, non-convex dynamic optimization, policy-tree descendant algorithm

JEL codes: C61, C63, D15, E21

1 Introduction

The life-cycle model of intertemporal consumption–saving decisions is a workhorse in macroeconomics, labor, and public economics, and many of its applications extend it with a social insurance program. The simplest such extension is a consumption floor,¹ under which the government makes a means-tested transfer whenever the agent’s wealth would otherwise fall below a

[†]The authors thank the participants of Econometric Society Dynamic Structural Econometrics summer schools and conferences, Petr Sedláček, Juergen Meinecke, and John Stachurski for helpful comments and discussion. Iskhakov acknowledges financial support from the Australian Research Council (FT180100632 and DP250104551). The code for the PTD algorithm is available at https://github.com/fediskhakov/ptd_paper.

[‡]Australian National University Wending.Liu@anu.edu.au

[§]Stony Brook University and Deakin University fedor.iskhakov@stonybrook.edu

¹We use “consumption floor”, “safety net”, and “social insurance” interchangeably in this paper.

predefined level, as in [Hubbard et al. \(1995\)](#); [Scholz et al. \(2006\)](#); [De Nardi et al. \(2009, 2010\)](#); [Li et al. \(2016\)](#); [Fan et al. \(2024\)](#); [Capatina and Keane \(2026\)](#). Next-period cash on hand is then the maximum of pre-transfer resources—the gross return on saving plus income—and the government-guaranteed floor. Simple as it is, this safety net changes the nature of the dynamic choice problem. A poor agent faces two competing local optima: cut consumption and save out of the transfer region, or consume everything and rely on the floor next period. The problem becomes non-convex with multiple stationary points, and the first-order conditions are no longer sufficient to characterize the optimal policy.

A common numerical approach to non-convex dynamic problems is grid search over a finitely discretized consumption choice, embedded in value function iteration or backward induction.² The price is an inevitable loss of accuracy from discretizing a continuous choice. The endogenous grid method (EGM) of [Carroll \(2006\)](#), based on inverting the Euler equation, delivers both accuracy and speed, but does not apply directly to non-convex problems. [Fella \(2014\)](#) adapted EGM to a model with switching costs, and [Iskhakov et al. \(2017\)](#) developed the DC-EGM generalization for simultaneous continuous–discrete choices. In both setups the optimal consumption function is discontinuous at the points where the agent is indifferent between regimes, and the value function—continuous and increasing—exhibits “upward kinks” there ([Clausen and Strub, 2020](#)), inherited through the link between its derivative and the consumption policy. With a safety net, these kinks and discontinuities arise from switching between relying and not relying on the consumption floor next period—another manifestation of the same non-convexity.

DC-EGM, developed by [Iskhakov et al. \(2017\)](#), and its multidimensional extension by [Druedahl and Jørgensen \(2017\)](#) use the Euler equation to recover state–choice pairs satisfying the first-order conditions, as in the original EGM. In non-convex problems, however, this yields a *correspondence* rather than a function, in both policy and value space. An additional upper-envelope step is then required to recover the value function by locating the intersections of the overlapping branches of the value correspondence. These intersections are exactly the kinks of the value function and the discontinuities of the policy function. [Iskhakov et al. \(2017\)](#) and [Druedahl and Jørgensen \(2017\)](#) operationalize this step differently, and [Dobrescu and Shanker \(2022\)](#) offer further improvements, but in every case the kink locations are only approximated, and whether all of them are found at all depends on implementation details.

This paper addresses an important source of inaccuracy in DC-EGM-style solvers: we locate the kinks in the value function and the discontinuities in the policy function by root finding. We focus on the deterministic setup, which is the hardest in this respect, because kinks and

²[Skiba \(1978\)](#) and [Dechert and Nishimura \(1983\)](#) provide early analytical characterizations of non-convex optimal growth problems.

discontinuities multiply and accumulate over backward induction (Iskhakov et al., 2017). We develop a new *policy-tree descendant* (PTD) algorithm based on an observation that the continuous segments of the policy function have a recursive structure across time periods. The natural domain of this observation is the class of discrete-time models whose optimal policies are piecewise affine. Under mild regularity conditions, this class includes models with HARA preferences paired with a nonnegative-saving constraint and transitions that are continuous and piecewise affine in savings, such as consumption floors, piecewise-linear taxes, and tapered transfers.³ For this class we prove that the optimal policy consists of finitely many affine segments and that PTD computes every segment in closed form and every kink location to root-finding precision—avoiding discretization entirely, with the consumption floor model as the running illustration.

In essence, the PTD algorithm is a backward induction procedure that represents the optimal policy function as a collection of continuous affine sections we call *policy segments*, and keeps track of the points of discontinuity that separate these segments. Further, we show that, viewed from the terminal period T , the optimal policy can be represented by a tree graph: each interior policy segment in period t descends from a parent segment in period $t + 1$, while corner segments enter as additional candidates. The PTD algorithm traverses the policy tree from the terminal period (root) to the initial period (leaves) and builds the policy segments of each period t from the segments of period $t + 1$. The coefficients of the affine policy segments are computed in closed form through the inverse Euler equation, similarly to EGM. Locating the kinks requires root finding, whose tolerance controls the numerical precision of the switching points on the covered domain. Because the procedure is grid-free, solving these one-dimensional equations leaves PTD competitive in speed with existing DC-EGM implementations, whose accuracy is in turn tied to the number of (endogenous) grid points used to approximate the policy function.

We extend PTD to stochastic models in two ways. Following Cai and Judd (2023), the first builds a certainty-equivalent version of PTD that approximates the stochastic problem with a deterministic one and re-solves the agent’s problem at every simulated period, allowing for stochastic transitions; we call this solver PTD-SCEQ. The approximation is coarse, but computationally very cheap. The second, based on the one-step rollout of Bertsekas (2025), allows stochastic transitions one period ahead and treats the remaining future as deterministic; we call it PTD-OSA. It permits a behavioral response to next-period risk and is intended to improve accuracy, at a higher computational cost.

³Under uncertainty affine consumption policies only arise in special cases, including quadratic certainty equivalence, CARA utility with purely additive income risk and nonstochastic returns, and CRRA utility with purely multiplicative return risk and no labor-income risk (Carroll and Kimball, 1996).

Numerical exercises across several specifications of the consumption–saving model with a safety net show that in deterministic problems PTD is more accurate than DC-EGM at comparable speed, and much faster than the DC-EGM configurations that come closest to its accuracy. In the stochastic models, most contests have no resolved winner against both alternatives. In the model with a three-dimensional state space (Hubbard et al., 1995), both PTD methods yield welfare gains over G2EGM at some initial cash levels. The PTD solvers pay their cost per simulated path rather than per calibration: with few paths per parameter configuration—the relevant regime in heterogeneous-agent applications—they can be more than two orders of magnitude faster. Apart from the cost of numerical integration, their cost does not grow with the dimension of the exogenous state space.

The rest of the paper is organized as follows. In Section 2 we describe the life-cycle model with consumption floor that serves as the illustration model throughout the paper. In Section 3 we characterize the class of models with piecewise-affine policies, develop the PTD solver for its deterministic members, and examine its numerical performance against the DC-EGM algorithm of Iskhakov et al. (2017). Section 4 extends the PTD solver to stochastic models. Section 5 provides discussion and concludes.

2 Life-Cycle Model with Consumption Floor

In this section we lay out the non-convex dynamic consumption–saving problem in the presence of a government-guaranteed consumption floor. Time is discrete and the agent has a finite horizon of T periods. At the beginning of time period $t \in \{1, \dots, T\}$ the agent has cash on hand $X_t \geq 0$ and observes an exogenous Markov state $S_t \in \mathbb{R}^k$. Subject to the credit constraint, she chooses consumption $C_t \in [0, X_t]$ and holds nonnegative end-of-period assets $A_t = X_t - C_t \geq 0$. The per-period utility of consumption is given by the CRRA utility function

$$u(c) = \frac{c^{1-\gamma} - 1}{1-\gamma}, \quad \gamma > 0, \gamma \neq 1; \quad u(c) = \log(c), \quad \gamma = 1, \quad (1)$$

where γ is the coefficient of relative risk aversion. We use the extended-real limit at zero, so $u(0) = -\infty$ when $\gamma \geq 1$.

Let Y_{t+1} be the net income received at the beginning of period $t+1$. Given the end-of-period assets A_t and the gross interest rate $R > 0$, the agent’s cash on hand at the beginning of period $t+1$ before the government transfer is $Z_{t+1} = RA_t + Y_{t+1}$. This is the amount entering the means-tested government transfer program. The government guarantees a minimum consumption level $\underline{x} > 0$ to the agent, so that the cash on hand at the beginning of period $t+1$ after the government

transfer is

$$X_{t+1} = \max\{Z_{t+1}, \underline{x}\} = \max\{RA_t + Y_{t+1}, \underline{x}\}. \quad (2)$$

Uncertainty enters through the i.i.d. innovations $\{\varepsilon_t\}_{t=2}^T$ taking values in $\mathbb{R}^n$. At the beginning of period $t + 1$ the innovation ε_{t+1} is realized, and the state moves according to

$$S_{t+1} = g_t(S_t, \varepsilon_{t+1}), \quad Y_{t+1} = h_{t+1}(S_{t+1}), \quad (3)$$

where the transition map $g_t : \mathbb{R}^k \times \mathbb{R}^n \rightarrow \mathbb{R}^k$ advances the exogenous Markov state, and the income map $h_{t+1} : \mathbb{R}^k \rightarrow \mathbb{R}$ converts the new state into net income.

For any initial date $t \in \{1, \dots, T\}$, a Markov consumption policy $\pi = (\pi_t, \dots, \pi_T)$ is admissible if, for every $\tau = t, \dots, T$, consumption $C_\tau = \pi_\tau(X_\tau, S_\tau)$ depends only on the observed state (X_τ, S_τ) and satisfies $0 \leq C_\tau \leq X_\tau$. We denote this set of admissible policies by Π_t . In the simple case of no bequest motives or any other additional terminal utility, the period- T policy is $C_T = X_T$ and the terminal value is $V_T(x, s) = u(x)$.

With time-separable preferences and a discount factor $\beta \in (0, 1)$, the continuation utility for a policy $\pi \in \Pi_t$ and the value function of the problem are given by

$$V_t^\pi(x, s) = \mathbb{E}^\pi \left[\sum_{\tau=t}^T \beta^{\tau-t} u(C_\tau) \mid X_t = x, S_t = s \right], \quad V_t(x, s) = \sup_{\pi \in \Pi_t} V_t^\pi(x, s). \quad (4)$$

According to standard dynamic programming theory the value function $V_t(x, s)$ satisfies the Bellman equation⁴

$$\begin{aligned} V_t(x_t, s_t) &= \max_{c_t} \left\{ u(c_t) + \beta \mathbb{E}[V_{t+1}(X_{t+1}, S_{t+1}) \mid S_t = s_t] \right\}, \\ \text{s.t.} \quad &0 \leq c_t \leq x_t, \\ X_{t+1} &= \max\{R(x_t - c_t) + h_{t+1}(S_{t+1}), \underline{x}\}, \\ S_{t+1} &= g_t(s_t, \varepsilon_{t+1}), \\ V_T(x_T, s_T) &= u(x_T) \quad \text{for all } t \in \{1, \dots, T-1\}, (x_T, s_T) \in \mathbb{R}_+ \times \mathbb{R}^k. \end{aligned} \quad (5)$$

We consider three special cases of (5) in this paper. For each example below we build the corresponding PTD and EGM solvers and an independent simulator: given a group of agents and a realization of the innovation series $\{\varepsilon_t\}_{t=2}^T$ for each agent, the simulator generates the consumption–saving paths using the transition rules (2).

⁴In (5) the date- t arguments are observed when the action is chosen, while the date- $(t + 1)$ quantities inside the conditional expectation are random.

Example 1 (i.i.d.). $n = k = 1$ with $s_t = \varepsilon_t = \log Y_t$, $g_t(s, \varepsilon) = \varepsilon$, and $h_t(s) = \exp(s)$: income is i.i.d. lognormal and normalized to mean one,

$$\log Y_t \sim \mathcal{N}(\mu, \sigma^2), \quad \mu = -\frac{\sigma^2}{2}, \quad (6)$$

so $\mathbb{E}[Y_t] = 1$. This is the income process of the discrete-continuous dynamic-choice benchmark of [Iskhakov et al. \(2017\)](#).

Example 2 (AR1). $n = k = 1$ with $s_t = \ell_t = \log Y_t$, $g_t(\ell, \varepsilon) = \rho\ell + \sigma_\eta\varepsilon$, and $h_t(\ell) = \exp(\ell)$:

$$\ell_{t+1} = \rho\ell_t + \sigma_\eta\varepsilon_{t+1}, \quad \varepsilon_{t+1} \sim \mathcal{N}(0, 1), \quad Y_t = \exp(\ell_t), \quad (7)$$

with $0 < \rho < 1$. Here σ_η is the innovation standard deviation. This is the workhorse persistent-income specification of the heterogeneous-agent literature with social insurance, as suggested by [Krusell and Smith \(1998\)](#).

Example 3 (HSZ). $n = k = 2$ with $s_t = [e_t, m_t]'$ and independent standard-normal innovations $\varepsilon_t = [\varepsilon_t^e, \varepsilon_t^m]'$. Both components are stationary AR(1) processes in logs,

$$\begin{aligned} e_{t+1} &= \mu_e + \rho_e(e_t - \mu_e) + \sigma_e\varepsilon_{t+1}^e, & \varepsilon_{t+1}^e &\sim \mathcal{N}(0, 1), \\ m_{t+1} &= \mu_m + \rho_m(m_t - \mu_m) + \sigma_m\varepsilon_{t+1}^m, & \varepsilon_{t+1}^m &\sim \mathcal{N}(0, 1), \\ y_t &= \exp(e_t) - \exp(m_t), \end{aligned} \quad (8)$$

with $0 < \rho_e < 1$ and $0 < \rho_m < 1$. The two innovations ε_{t+1}^e and ε_{t+1}^m are independent of each other and over time. Here μ_e and μ_m are the stationary means; σ_e and σ_m are the innovation standard deviations of the two components. We interpret e_t as log earnings and m_t as log medical expenditure, so net income is their level difference and can be negative. This is a stylized version of the earnings and medical-expenditure processes in [Hubbard, Skinner and Zeldes \(1995\)](#), and we label the example HSZ after the authors' initials.

3 Policy-Tree Descendant Algorithm for Deterministic Models

In this section we focus on the deterministic model, where non-convexity manifests itself in the accumulation and multiplication of kinks and discontinuities in the course of backward induction. As [Iskhakov et al. \(2017\)](#) explain, in the absence of stochastic elements in the intertemporal budget constraint it becomes possible to predict the exact level of wealth in period t that brings the agent to a kink point in period $t + 1$, so every regime switch at $t + 1$ is reflected back into

period t , and recursively into all earlier periods. This reflection mechanism is also what gives the discontinuous policy function its tree structure, and it is the foundation of the PTD algorithm. The next section extends the algorithm to stochastic problems.

3.1 Constructing the policy tree

Consider the simplest deterministic version of the consumption–saving problem with a safety net in [Section 2](#), where the income sequence $\{y_t\}_{t=2}^T$ is fixed and known to the agent from the start, and the gross return R is also fixed. Given consumption c_t , next-period cash on hand evolves as

$$x_{t+1} = \max\{R(x_t - c_t) + y_{t+1}, \underline{x}\}.$$

The Bellman equation is given by

$$\begin{aligned} v_t(x_t) &= \max_{0 \leq c_t \leq x_t} \left\{ u(c_t) + \beta v_{t+1} \left(\max [R(x_t - c_t) + y_{t+1}, \underline{x}] \right) \right\}, \\ v_T(x_T) &= u(x_T) \quad \text{for all } t \in \{1, \dots, T-1\}, \quad x_T \geq 0. \end{aligned} \tag{9}$$

We use the lowercase v for the deterministic value function, which serves as a building block for the extensions to the stochastic model in the next section.

Following the backward induction algorithm to compute the optimal policy and value functions starting from the terminal period T , we note that, in the absence of additional terminal utility, the agent simply consumes all her cash on hand and thus $c_T(x_T) \equiv c_T^0(x_T) = x_T$ for all $x_T \in \mathbb{R}_+$. Hereafter we use the superscript “0” to denote the policy of consuming all available wealth. The policy function and value function at $t = T$ are illustrated in [Fig. 1](#).

In the next-to-last period $T - 1$, the agent’s problem is more complicated. The policy $c_{T-1}^0(x_{T-1}) = x_{T-1}$ is a feasible choice, but for larger values of x_{T-1} it may be optimal to save some of the wealth for the terminal period and stay above the safety net $\underline{x}$. This interior solution is characterized by the first-order condition taking the form of the Euler equation

$$u'(c_{T-1}^1(x_{T-1})) = \beta R u'(c_T^0(z_T)), \tag{10}$$

where $z_T = R(x_{T-1} - c_{T-1}^1) + y_T$, and we use superscript “1” to denote the policy $c_{T-1}^1 \equiv M c_T^0$ that is derived from the policy c_T^0 using the Euler equation (10). The latter defines the *descendant operator* M that maps c_T^0 to c_{T-1}^1 .⁵ The descendant operator M is a piecewise implementation of the Coleman–Reffett operator ([Sargent and Stachurski, 2026](#), Section 8.3.3.3). Since c_{T-1}^1 is

⁵In most typical cases when the marginal utility is invertible as with EGM, the operator M can be implemented in closed form; we give a formal definition in the next subsection.

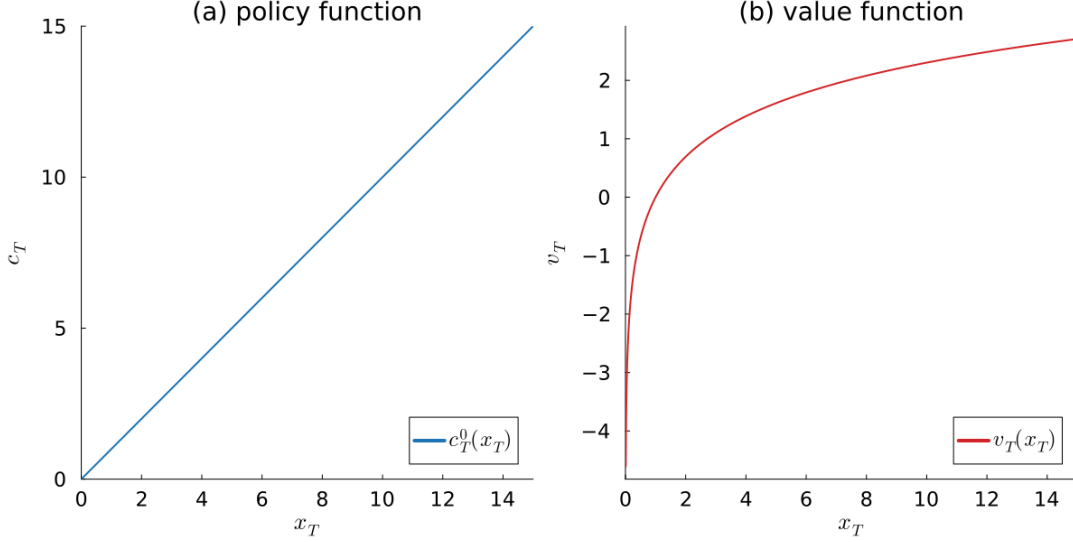


Figure 1: Policy and value functions at T

Notes: Panel (a) shows the terminal policy $c_T^0(x_T) = x_T$ and panel (b) the terminal value $v_T(x_T) = u(x_T)$. Illustrations use the parameterization with $u(x) = \log(x)$, $T = 50$, $\beta = 0.98$, $\gamma = 1$, $R = 1.1$, $\underline{x} = 3$, $y_t = 1$ for all $t \in \{2, \dots, T\}$.

derived purely from the Euler equation and pinned down by c_T^0 , we call c_{T-1}^1 the “child policy” of the “parent policy” c_T^0 .

Panel (a) in Fig. 2 illustrates the two available policies c_{T-1}^0 and c_{T-1}^1 , which we refer to as *policy segments*. To determine the optimal consumption for all values of wealth, the agent has to determine the “switching point” between the two policy segments. This amounts to comparing the two segments of the value correspondence that are supported by the two policy segments, as shown in panel (b) in Fig. 2. The *value segments* derived from the two policy segments are

$$\begin{aligned} v_{T-1}^0(x) &= u(c_{T-1}^0(x)) + \beta v_T(\max\{y_T, \underline{x}\}), \\ v_{T-1}^1(x) &= u(c_{T-1}^1(x)) + \beta v_T(R(x - c_{T-1}^1(x)) + y_T), \end{aligned}$$

with the switching point given by x that satisfies $v_{T-1}^0(x) = v_{T-1}^1(x)$. This point becomes the kink point of the value function as shown in panel (d) in Fig. 2, and also determines the *domains* where each of the policy segments is optimal. The resulting policy function c_{T-1}^* shown in panel (c) in Fig. 2 is the collection of the two refined segments c_{T-1}^{*0} and c_{T-1}^{*1} , whose domains partition $\mathbb{R}_+$.⁶

Fig. 2 also reveals an interesting feature of the consumption–saving model with social insurance: there is an interval of wealth where the choice of positive savings (interior solution) is dominated by the corner solution of consuming all wealth and relying on the consumption floor

⁶At their shared boundary policy segments by definition yield the same value. Our numerical implementation enforces the save-more rule, which selects the lower consumption; in the resulting collection this is the segment to the right. This implies that the resulting consumption policy function is right-continuous.

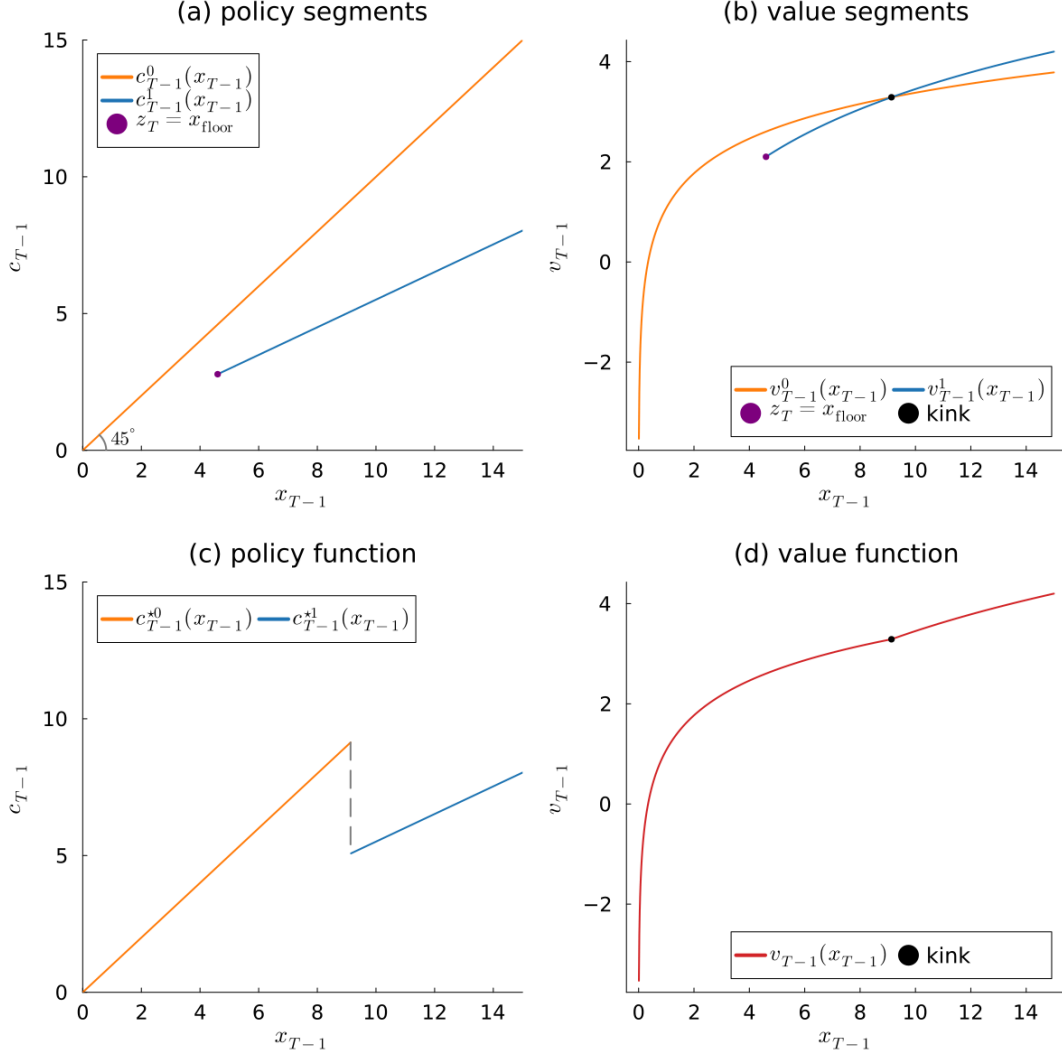


Figure 2: Policy and value segments and functions at $T - 1$

Notes: Panel (a) shows the two policy segments c_{T-1}^0 and c_{T-1}^1 , panel (b) shows the two value segments v_{T-1}^0 and v_{T-1}^1 , panel (c) shows the optimal policy function c_{T-1}^* , and panel (d) shows the optimal value function v_{T-1} . The model is parameterized as in Fig. 1.

in the next period (corner solution), signifying the existence of the “poverty trap”. The typical kink at the 45-degree line in the policy function that is present in consumption–saving models with credit constraints is replaced here by a discontinuity, see panel (c).

The solution to the agent’s problem at $T - 2$ follows the same logic. The policy segment c_{T-2}^0 of consuming the entire cash on hand is feasible as before, and the interior solution may be derived from either of the two policy segments at $T - 1$ using the Euler equation. This exhausts the possibilities, and thus the optimal policy at $T - 2$ is contained within the three policy segments c_{T-2}^0 , c_{T-2}^1 , and c_{T-2}^2 , two of which are descendants of the two refined segments at $T - 1$. The descendance structure of policy segments is illustrated in Fig. 3: at each period, the operator M produces descendants, and the default $c^0(\cdot)$ segment is added to the collection.

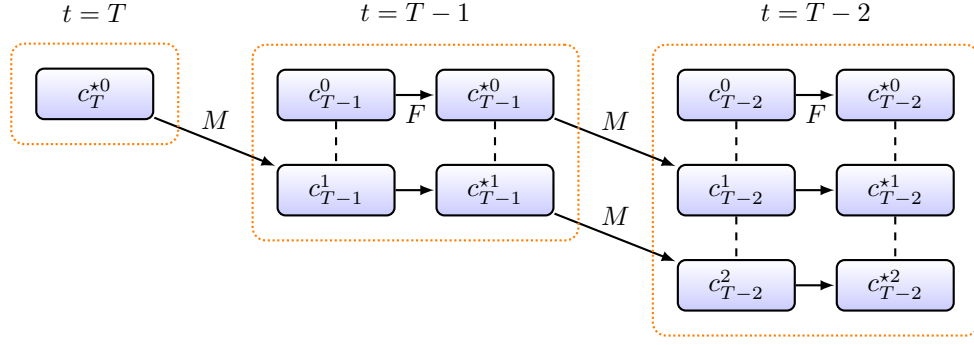


Figure 3: Policy tree in the last three periods

Notes: The diagram shows the descendance structure of the policy segments in the last three periods.

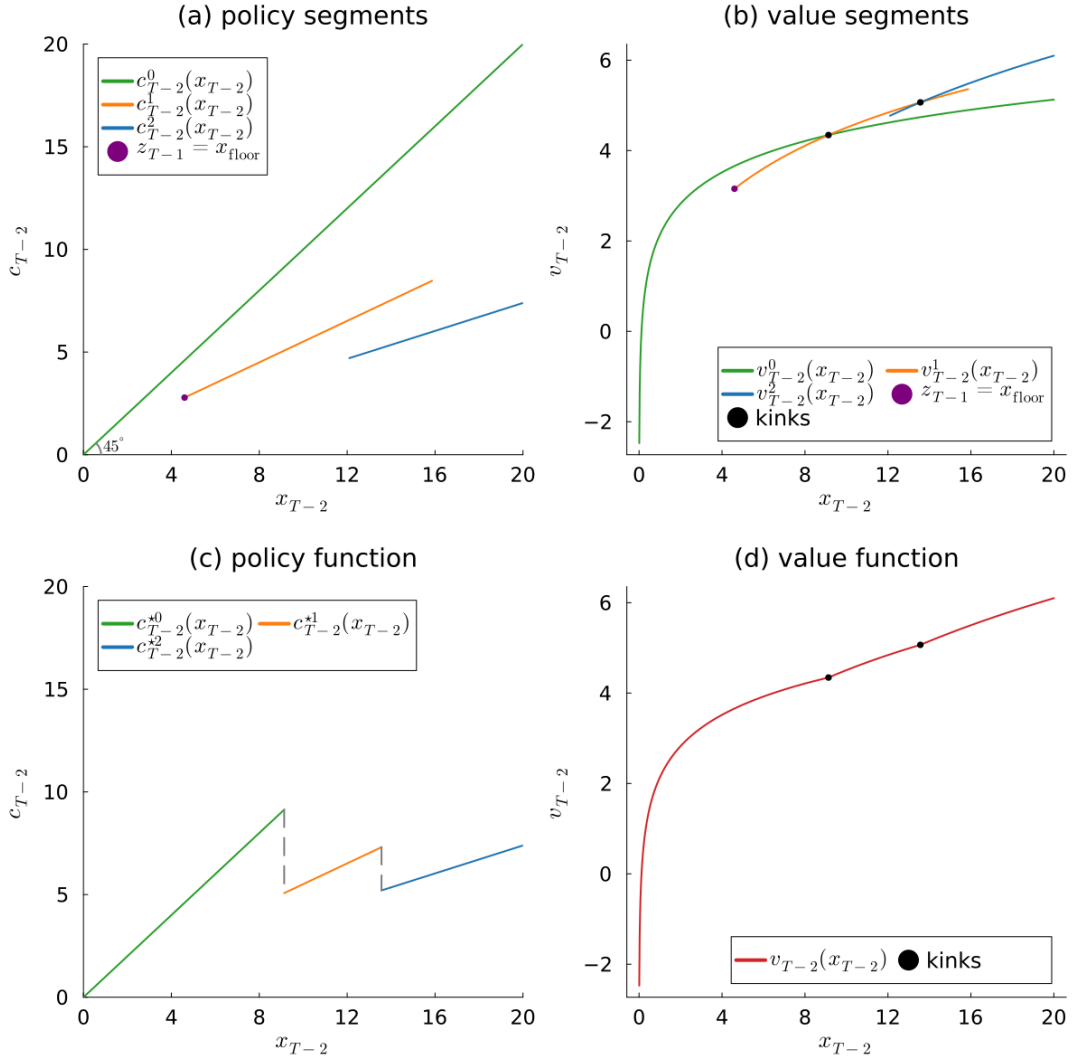


Figure 4: Policy and value functions at $T - 2$

Notes: Panel (a) shows the three policy segments c_{T-2}^0 , c_{T-2}^1 , and c_{T-2}^2 , panel (b) shows the three value segments v_{T-2}^0 , v_{T-2}^1 , and v_{T-2}^2 , panel (c) shows the optimal policy function c_{T-2}^* , and panel (d) shows the optimal value function v_{T-2} . The model is parameterized as in Fig. 1.

The *fusion operator* F then refines the domains of these policy segments by comparing their associated value segments and combines the refined segments into the optimal policy function. The refined segments are denoted with the superscript “ $\star$ ”. Fig. 3 illustrates the first three steps of backward induction but the same logic applies to all periods, and the resulting structure is a tree of policy segments that is traversed in the PTD algorithm.

Fig. 4 illustrates the operations in period $T - 2$ step-by-step. Panel (a) shows the two policy segments derived from optimal consumption at $T - 1$ using the appropriate Euler equation $u'(c_{T-2}^j(x_{T-2})) = \beta R u'(c_{T-1}^{\star j-1}(z_{T-1}))$ for $j \in \{1, 2\}$; panel (b) shows the value segments computed from the corresponding policy segments. We now observe two crossing points, resulting in two kinks on the value function in panel (d) and two discontinuities in the resulting policy function shown in panel (c). The optimal policy function $c_{T-2}^\star$ is the collection of the three refined segments $c_{T-2}^{\star 0}$, $c_{T-2}^{\star 1}$, and $c_{T-2}^{\star 2}$, whose domains partition $\mathbb{R}_+$.

The fusion operator F can be more complicated in the general case, where some policy segments may be completely dominated by others. We formally define the descendant operator M , the fusion operator F , and describe the full policy descendant algorithm next.

3.2 PTD algorithm for models with piecewise-affine policies

The construction of Section 3.1 used a handful of structural properties of the safety-net model: marginal utility could be inverted in closed form, the intertemporal budget was affine in savings up to the single kink created by the floor, and the constraint boundaries were affine. In this section we state these properties formally and develop the PTD algorithm. The safety-net model (9) serves as a running example, illustrating the definitions and operators. Consider the deterministic finite-horizon problem

$$v_t(x) = \max_{0 < c \leq x} \{u(c) + \beta v_{t+1}(f_t(x - c))\}, \quad x > 0, \quad t < T, \quad (11)$$

where x is a scalar wealth state, consumption is positive and saving is nonnegative, the transition f_t maps end-of-period savings $a = x - c$ into next-period wealth, and the terminal payoff v_T is given. At positive wealth, the safety-net model (9) is the special case of (11) with $f_t(a) = \max\{Ra + y_{t+1}, \underline{x}\}$ and $v_T = u$, so that $c_T^\star(x) = x$. Throughout, $c_t^\star$ denotes the *policy function* at period t .

We first state the structural assumptions under which the Euler equation maps affine parent policies into affine descendants. We assume that the consumption utility is in the hyperbolic absolute risk aversion (HARA) class, which is widely used in theoretical and empirical economics

and includes CRRA utility as a special case. See [Campbell \(2018, Section 1.3\)](#) for an introduction to HARA utility.

Assumption 1 (affine primitives and regularity). The horizon is finite, $0 < \beta < 1$, and problem (11) satisfies:

- (i) the utility function u is strictly increasing, strictly concave and twice continuously differentiable on $\mathbb{R}_{++}$, with affine positive risk tolerance (HARA): $-u'(c)/u''(c) = \tau_0 + \tau_1 c$;
- (ii) each transition f_t is continuous, nondecreasing, and piecewise affine with finitely many branches, with $f_t(0) > 0$;
- (iii) at every date and every positive wealth level, optimal consumption choices exist. On each compact interval of positive wealth, all such choices lie in a common compact subset of $\mathbb{R}_{++}$;
- (iv) the terminal policy c_T^* has finitely many affine pieces with nonnegative slopes and never jumps up, and v_T is continuous with one-sided derivatives $v_T'(x\pm) = u'(c_T^*(x\pm))$.

In [Assumption 1](#), condition (i) makes the deterministic Euler relation affine in current and next-period consumption. Condition (ii) imposes transition regularity. Condition (iii) assumes that optima are attained and optimal consumption is locally bounded away from zero; it rules out a binding nonnegativity constraint on consumption, which, like any binding constraint other than $c \leq x$, would break the link between the marginal value of wealth and marginal utility used below. Condition (iv) requires the terminal period to have the properties that backward induction preserves at every earlier date ([Lemma 3](#)). The Euler equation is a necessary condition at interior optima where both the transition and the continuation value are differentiable. Condition (iii) is the only one that refers to the solution, and it is implied by the others under the Inada condition. Here and below, proofs are collected in [Appendix A](#) unless given in the text.

Lemma 1 (Inada). *If $u'(0+) = +\infty$ and $c_T^*(x) > 0$ for all $x > 0$, then conditions (i), (ii) and (iv) of [Assumption 1](#) imply condition (iii).*

The illustration model satisfies [Assumption 1](#) whenever $\underline{x} > 0$: CRRA utility has $\tau_0 = 0$, $\tau_1 = 1/\gamma$ and $u'(0+) = +\infty$; the transition $f_t(a) = \max\{Ra + y_{t+1}, \underline{x}\}$ is continuous, nondecreasing and piecewise affine with $f_t(0) > 0$; the terminal policy $c_T^*(x) = x$ and value $v_T = u$ satisfy (iv); and (iii) follows from [Lemma 1](#). More generally, the class admits any transition that is continuous and piecewise affine in savings, such as progressive tax brackets and tapered means-tested transfers, as well as bequest motives of the form discussed in [footnote 7](#) (page 15).

For a gross discounted return $\theta > 0$, define the inverse-Euler map $\psi_\theta(c) := (u')^{-1}(\theta u'(c))$, so that $u'(\psi_\theta(c)) = \theta u'(c)$. Integrating the HARA identity $(\log u')' = -1/(\tau_0 + \tau_1 c)$ between c and $\psi_\theta(c)$ gives

$$\log \theta = \log \frac{u'(\psi_\theta(c))}{u'(c)} = - \int_c^{\psi_\theta(c)} \frac{ds}{\tau_0 + \tau_1 s}.$$

Evaluating this integral yields

$$\psi_\theta(c) = \begin{cases} c - \tau_0 \log \theta, & \tau_1 = 0, \\ (c + \tau_0/\tau_1)\theta^{-\tau_1} - \tau_0/\tau_1, & \tau_1 \neq 0. \end{cases} \quad (12)$$

Thus, $\psi_\theta(c) = p_\theta + q_\theta c$ with $q_\theta > 0$; the inverse-Euler map is an affine function of consumption in the next period.

These assumptions and the affine inverse-Euler relation concern deterministic models. Under uncertainty, [Carroll and Kimball \(1996\)](#) show that HARA preferences can generate strictly concave consumption policies. However, the affine inverse-Euler map serves as the computational engine of the deterministic PTD solver, which in turn supports the simulation-based stochastic methods of [Section 4](#). We now introduce the basic building block of PTD: the policy segment.

Definition 1. A *policy segment* c_t^k at period t is a continuous function $c_t^k : D \rightarrow \mathbb{R}_+$, where D is a nondegenerate interval closed relative to $\mathbb{R}_{++}$, with strictly positive consumption on its interior. Its *refinement* c_t^{*k} is the restriction of c_t^k to the closure in D of a maximal nondegenerate interval on which it coincides with the optimal policy. A *policy collection* $\mathbf{c}_t = \{c_t^{*0}, c_t^{*1}, \dots, c_t^{*n}\}$ at period t is a finite ordered set of nonempty affine refinements, whose domains abut,

$$\sup \text{dom}(c_t^{*i}) = \inf \text{dom}(c_t^{*i+1}) \quad \text{for all } i \in \{0, 1, \dots, n-1\},$$

and cover $\mathbb{R}_{++}$, so that at every wealth level in that domain the optimal policy satisfies $c_t^*(x) = c_t^{*k}(x)$ for some $k \in \{0, 1, \dots, n\}$.

By construction the domains of two distinct refinements in a policy collection share at most one point. The policy collection is the data structure that represents the policy function, and it makes the evaluation of both the policy and the value functions immediate. Evaluating $c_t^*(x)$ amounts to locating the single segment of $\mathbf{c}_t$ whose domain contains x and returning its value at x ; at a shared boundary the segment to the right is selected, which makes c_t^* right-continuous, consistently with the save-more tie rule. Evaluating $v_t(x)$ requires no separate object either: the

deterministic problem is solved forward along the optimal path, accumulating

$$v_t(x) = \sum_{j=t}^{T-1} \beta^{j-t} u(c_j) + \beta^{T-t} v_T(x_T), \quad x_{j+1} = f_j(x_j - c_j), \quad (13)$$

where $x_t = x$ and at each step c_j is read off the period- j policy collection. Notice that we never “save” the value function on grids, so no interpolation or extrapolation is needed within the solved domains. This feature distinguishes our algorithm from existing methods for solving non-convex dynamic optimization problems.

By [Assumption 1\(ii\)](#) each transition f_{t-1} is affine on finitely many maximal intervals of savings, its *branches*: on branch i we write $f_{t-1}(a) = R_i a + y_i$ for a in the closed cell A_i . In the illustration model f_{t-1} has two branches, the safety-net branch with $R_i = 0$ and the unconstrained branch with slope R .

Definition 2. Fix a positive-slope branch $f_{t-1}(a) = R_i a + y_i$ on cell A_i . The *child policy segment* c_{t-1}^{k+1} of a nondecreasing policy segment c_t^k satisfies, with $z_t^k(x) := R_i(x - c_{t-1}^{k+1}(x)) + y_i$,

$$\begin{aligned} 0 < c_{t-1}^{k+1}(x) < x, & \quad x - c_{t-1}^{k+1}(x) \in \text{int } A_i, \\ z_t^k(x) \in \text{int dom}(c_t^k), & \quad u'(c_{t-1}^{k+1}(x)) = \beta R_i u'(c_t^k(z_t^k(x))), \end{aligned}$$

on the interior of its domain. Outside this domain, no consumption choice satisfies all four conditions simultaneously. We call c_t^k the *parent policy segment* of c_{t-1}^{k+1} .

The first two conditions of [Definition 2](#) restrict the child policy segment to feasible consumption and to savings for which the chosen branch of the transition is the active one; the domain requirement ensures that the agent’s next-period wealth is in the domain of the parent policy c_t^k ; and the last condition is the Euler equation along the branch. In the illustration model the unconstrained branch is the only positive-slope branch, and on the child’s interior the four conditions specialize to $0 < c_{t-1}^{k+1}(x) < x$, the slack floor condition $z_t^k(x) > \underline{x}$, the parent-domain interior requirement, and the Euler equation with gross return R .

We now introduce the descendant operator M , which constructs a child policy segment from its parent.

Definition 3. The *descendant operator* M maps a policy segment c_t^k at period $t \in \{2, \dots, T\}$ into its child policy segment at period $t - 1$ along the given branch of f_{t-1} , $M(c_t^k) = c_{t-1}^{k+1}$, and returns the empty segment when no nondegenerate child exists.

To implement the descendant operator M , we first show a very useful property of M : within

the class of [Assumption 1](#), descending preserves affinity.

Theorem 1. *Under [Assumption 1](#), for any $t \in \{2, \dots, T\}$ and any positive-slope branch $f_{t-1}(a) = R_i a + y_i$, the descendant operator M maps an affine parent $c_t^k(x') = b_0 + b_1 x'$ with $b_1 \geq 0$ into the affine child*

$$c_{t-1}^{k+1}(x) = d_0 + d_1 x, \quad d_0 = \frac{p + q(b_0 + b_1 y_i)}{1 + q b_1 R_i}, \quad d_1 = \frac{q b_1 R_i}{1 + q b_1 R_i} \in [0, 1), \quad (14)$$

whenever the child is nonempty, where $\psi_{\beta R_i}(c) = p + qc$ is the inverse-Euler map [\(12\)](#) with $\theta = \beta R_i$.

For the illustration model, CRRA marginal utility is $u'(c) = c^{-\gamma}$, including the logarithmic case $\gamma = 1$, so $\psi_{\beta R}$ is proportional, with $p = 0$ and $q = \kappa := (\beta R)^{-1/\gamma}$, and [\(14\)](#) specializes to

$$c_{t-1}^{k+1}(x) = d_0 + d_1 x, \quad d_0 = \frac{\kappa(b_0 + b_1 y_t)}{1 + \kappa b_1 R}, \quad d_1 = \frac{\kappa b_1 R}{1 + \kappa b_1 R}. \quad (15)$$

By [Assumption 1\(iv\)](#) and repeated application of [Theorem 1](#), all policy segments descended from the terminal “roots” are affine functions.⁷ Accordingly, we use a tuple $c_t^k := (x_L, x_R, b_0, b_1)$ to represent an affine policy segment, where $x_L = \inf \text{dom}(c_t^k)$, $x_R = \sup \text{dom}(c_t^k)$, and $c_t^k(x) = b_0 + b_1 x$ for all $x \in \text{dom}(c_t^k)$; this is the representation used in all the algorithms below. Endpoint values are one-sided limits; only feasible endpoint actions are evaluated.

Since the child policy segment is an affine function, we only need to pin down the two endpoints of its domain and its affine coefficients, both available in closed form from [\(14\)](#) and the inverse-Euler map [\(12\)](#). The next-period states x' reachable by the child form an interval: the intersection of the parent’s domain with the image of the branch cell under $a \mapsto R_i a + y_i$ and with the image of the feasibility restrictions, all of which have affine descriptions. In the illustration model, a state x' reachable by the child must lie between the parent’s domain endpoints x_L and x_R , nonnegative saving requires $x' \geq y_t$, and the safety net requires $x' \geq \underline{x}$, so the closure of the reachable parent interval is

$$[x'_L, x'_U] = [\max\{x_L, y_t, \underline{x}\}, x_R]. \quad (16)$$

The operator returns no child when $x'_U \leq x'_L$. Otherwise, the exact inverse state map—for the illustration model

$$\chi(x') = \frac{x' - y_t}{R} + \kappa(b_0 + b_1 x'), \quad (17)$$

⁷ In particular, a bequest term in the final-period utility function as in [De Nardi et al. \(2010\)](#) is admissible when the resulting terminal policy and value satisfy [Assumption 1\(iv\)](#). This holds, for example, whenever $v_T(x) = \max_{0 < c \leq x} \{u(c) + W(x - c)\}$ for a continuous nondecreasing bequest function W with a piecewise-affine maximizer.

and in general $\chi(x') = (x' - y_i)/R_i + \psi_{\beta R_i}(b_0 + b_1 x')$ —is strictly increasing and maps the reachable parent interval's endpoints to the child's domain endpoints $\chi(x'_L)$ and $\chi(x'_U)$, with $\chi(+\infty) = +\infty$. [Algorithm 1](#) collects these steps for the illustration model.

Algorithm 1: Exact descendant operator M (illustration model)

Input : parent policy segment $c_t^k \equiv (x_L, x_R, b_0, b_1)$; β, γ, R, y_t , and $\underline{x}$;
Output: child policy segment c_{t-1}^{k+1} , or the empty segment;

```

1  $\kappa \leftarrow (\beta R)^{-1/\gamma}$ ;
2  $x'_L \leftarrow \max\{x_L, y_t, \underline{x}\}, x'_U \leftarrow x_R$ ; // reachable parent interval
3 if  $x'_U \leq x'_L$  then
4   | return the empty segment;
5 end
6  $d_1 \leftarrow \frac{\kappa b_1 R}{1 + \kappa b_1 R}, d_0 \leftarrow \frac{\kappa(b_0 + b_1 y_t)}{1 + \kappa b_1 R}$ ; // exact inverse-Euler coefficients
7 Define  $\chi(x') \leftarrow \frac{x' - y_t}{R} + \kappa(b_0 + b_1 x')$ ;
8  $\hat{x}_L \leftarrow \chi(x'_L), \hat{x}_R \leftarrow \chi(x'_U)$ , using one-sided endpoint limits and  $\chi(+\infty) = +\infty$ ;
9 return  $c_{t-1}^{k+1} \equiv (\hat{x}_L, \hat{x}_R, d_0, d_1)$ ;
```

Now we show that the child policy segments, complemented by a small model-specific list of *corner candidates*, exhaust the optimal policy. A corner candidate is an affine policy segment obtained by holding saving at zero, $c(x) = x$, or at a nonnegative kink $\bar{a}$ of the transition f_{t-1} , $c(x) = x - \bar{a}$. Under [Assumption 1](#) all of these are affine with nonnegative slopes, and there are finitely many of them.

Theorem 2. *Under [Assumption 1](#), for any $t \in \{2, \dots, T\}$ and $x > 0$, given all child policy segments generated by the parent policy collection $\mathbf{c}_t$, the optimal policy c_{t-1}^* coincides at x with one of these children or a corner candidate. In the illustration model (9), the only corner candidate that can be optimal is the consume-all segment $c(x) = x$.*

The policy children and corner candidates thus contain all the information needed for the optimal policy function; what remains is to refine their domains.

Definition 4. For a candidate policy segment c_t^k , its *value segment* v_t^k gives the discounted lifetime utility obtained by following c_t^k in period t and the optimal policy thereafter. It is defined on the feasible part of $\text{dom}(c_t^k)$, where $0 < c_t^k(x) \leq x$, by

$$v_t^k(x) := u(c_t^k(x)) + \beta v_{t+1}(f_t(x - c_t^k(x))), \quad (18)$$

where f_t is the transition rule mapping current savings to next-period wealth.

Definition 5. For any $t \in \{1, \dots, T-1\}$, the *fusion operator* F maps a finite set of affine candidate policy segments $\{c_t^k\}_{k=0}^n$ at period t , together with the next-period value function

v_{t+1} , into the policy collection $F(\{c_t^k\}_{k=0}^n; v_{t+1}) = \mathbf{c}_t$. It refines the domains of the policy children and corner candidates to the intervals on which they are optimal.

Before we discuss how to implement the fusion operator F , we establish the properties of the candidates that simplify it.

Theorem 3. *Under [Assumption 1](#), for any $t \in \{1, \dots, T-1\}$, let $a_t^*(x) := x - c_t^*(x)$ denote the selected optimal end-of-period asset, with the larger asset selected when maximizers tie. Then $a_t^*(x)$ is nondecreasing in x .⁸*

Lemma 2 (saving bands). *Under [Assumption 1](#), fix $t \in \{2, \dots, T\}$ and assign to each candidate at $t-1$ a saving band: $\{0\}$ to the consume-all segment, $\{\bar{a}\}$ to the corner candidate at a kink $\bar{a}$, and*

$$S_{k,i} := \{a \in \text{int } A_i : R_i a + y_i \in \text{int dom}(c_t^{*k})\}$$

to the child of c_t^{*k} along branch i . The saving $x - c(x)$ of each candidate lies in its band at every interior point of its domain, and the bands of distinct candidates are disjoint.

Disjoint points and open intervals of the real line are totally ordered. We say that candidate p precedes q in *increasing saving order* if the band of p lies to the left of the band of q ; then $x - p(x) < x - q(x)$ at every interior point of their common domain. The consume-all segment comes first; the remaining candidates are ordered by the transition cell that contains their saving, with each kink corner between the two cells it separates, and within a cell by the position of their parent in $\mathbf{c}_t$.

Corollary 3.1 (No switching back). *Under [Assumption 1](#), as wealth increases, once the optimal policy switches from one policy child or corner candidate to another, it never returns to the former.*

Proof. By [Theorem 3](#) the selected saving is nondecreasing in wealth, and by [Lemma 2](#) it lies in the band of the selected candidate. Since the bands are disjoint, the selected candidates follow increasing saving order and none of them recurs. $\square$

The next property is inherited from the terminal period, and it is the reason why the parent policy carries all the information about marginal continuation values that the descendant operator needs.

⁸Note that this result requires neither the HARA nor the piecewise-affine conditions of [Assumption 1](#), nor concavity of the continuation value: only the strict concavity of u , monotone feasibility, and the fact that the transition depends on the choice through savings alone. DC-EGM likewise relies on the monotonicity of saving. [Iskhakov et al. \(2017, Theorem 4\)](#) establish the corresponding result for each discrete-choice-specific saving function.

Lemma 3 (envelope property). *Under [Assumption 1](#), for every $t \in \{1, \dots, T\}$ the policy function never jumps up, $c_t^*(x+) \leq c_t^*(x-)$, and the value function v_t is continuous on $\mathbb{R}_{++}$ with one-sided derivatives $v_t'(x\pm) = u'(c_t^*(x\pm))$ at every $x > 0$.*

At $t = T$, [Lemma 3](#) is [Assumption 1\(iv\)](#); for $t < T$ it is proved jointly with [Theorems 2](#) and [4](#) by backward induction. Thus the value function has only upward kinks, located at the downward jumps of consumption.

Lemma 4 (single crossing). *Under [Assumption 1](#), let $t \in \{2, \dots, T\}$ and let p precede q in increasing saving order among the candidates at $t - 1$. Then $v_p' = u' \circ p$ and $v_q' = u' \circ q$ on the interiors of their domains, and $\Delta := v_p - v_q$ is strictly decreasing on the interior of their common domain.*

Two value segments therefore intersect at most once, and the candidate that saves more is better to the right of the intersection. At finite or infinite endpoints, value differences are interpreted as inward limits whenever direct evaluation is undefined; at $+\infty$ the limit exists because Δ is monotone.

Single crossing turns the fusion operator into a one-dimensional sweep, summarized in [Algorithm 2](#). The finite domain endpoints of all candidates split $\mathbb{R}_{++}$ into cells. Inside a cell every candidate is defined either throughout or not at all, so a candidate processed later in increasing saving order can only take over a right-hand part of the cell. The upper envelope on a cell is therefore built with a stack: each new candidate is compared with the rightmost piece of the envelope by evaluating Δ at the two ends of that piece, which either discards the new candidate, removes the piece, or locates a single switching point by Brent’s method ([Press et al., 2007](#), Section 9.3). Concatenating the cells and merging adjacent pieces of the same affine policy gives the policy collection. Each candidate enters and leaves the stack at most once per cell, so n candidates require $O(n^2)$ evaluations and root-finding problems. [Algorithm 2](#) constructs the value upper envelope used by [Iskhakov et al. \(2017\)](#) without a grid. [Appendix B](#) illustrates fusion with multiple positive-slope transition branches and compares the resulting policies with independent direct optimization.

Algorithm 2: Fusion operator F

Input : policy children Q and corner candidates K ; continuation value v_{t+1} ; relative tolerance $\varepsilon_{\text{rel}} > 0$;
Output: policy collection $\mathbf{c}_t$;

```
1  $(q_0, \dots, q_n) \leftarrow Q \cup K$  in increasing saving order; // Lemma 2
2  $0 = e_0 < e_1 < \dots < e_{m+1} = +\infty \leftarrow$  the distinct domain endpoints of  $q_0, \dots, q_n$ ;
3  $\mathbf{c}_t \leftarrow []$ ;
4 for  $k = 0, \dots, m$  do
5    $E \leftarrow []$ ; // upper envelope on the cell  $[e_k, e_{k+1}]$ 
6   for each  $q_j$  defined on  $[e_k, e_{k+1}]$ , in increasing saving order do
7     while  $E \neq []$  do
8        $p \leftarrow$  last segment of  $E$ , with domain  $[L, R]$ ;
9       Define  $\Delta(x) \leftarrow v_p(x) - v_{q_j}(x)$  using (18); // strictly decreasing
10      if  $\Delta(R) \geq 0$  then
11        break; //  $q_j$  is not better on  $[L, R]$ 
12      else if  $\Delta(L) \leq 0$  then
13        remove  $p$  from  $E$ ; //  $q_j$  is better on  $[L, R]$ 
14      else
15        find  $\xi \in (L, R)$  with  $\Delta(\xi) = 0$  by Brent's method with tolerance  $\varepsilon_{\text{rel}}$ ;
16        restrict  $p$  to  $[L, \xi]$ ; break;
17      end
18    end
19     $\ell \leftarrow$  right endpoint of the last segment of  $E$ , or  $e_k$  if  $E = []$ ;
20    if  $\ell < e_{k+1}$  then
21      append  $q_j$  restricted to  $[\ell, e_{k+1}]$  to  $E$ ;
22    end
23  end
24  append  $E$  to  $\mathbf{c}_t$ ;
25 end
26 Merge adjacent segments of  $\mathbf{c}_t$  with identical coefficients;
27 return  $\mathbf{c}_t$ ;
```

The PTD algorithm constructs the policy tree by alternately applying the descendant operator M and the fusion operator F , as summarized in Algorithm 3.

Algorithm 3: PTD algorithm for deterministic models

Input : a model in Assumption 1; terminal policy collection and payoff v_T ; relative tolerance $\varepsilon_{\text{rel}} > 0$;
Output: policy collections $(\mathbf{c}_1, \dots, \mathbf{c}_T)$;

```
1  $\mathbf{c}_T \leftarrow$  terminal policy collection; // consume all in the illustration model
2 for  $t = T - 1, T - 2, \dots, 1$  do
3    $Q \leftarrow []$ ; // policy children
4   for each  $p \in \mathbf{c}_{t+1}$  and each positive-slope branch  $i$  of  $f_t$  do
5      $q \leftarrow M(p, i)$ ; // Algorithm 1
6     if  $q \neq \emptyset$  then
7       append  $q$  to  $Q$ ;
8     end
9   end
10   $K \leftarrow$  feasible corner candidates; // Theorem 2
11   $\mathbf{c}_t \leftarrow F(Q, K; v_{t+1}, \varepsilon_{\text{rel}})$ ; // Algorithm 2
12  Recover  $v_t$  using (13);
13 end
14 return  $(\mathbf{c}_1, \dots, \mathbf{c}_T)$ ;
```

Now we consider the size of the policy collections that the PTD algorithm constructs. With

at most B positive-slope transition branches and K corner candidates that can be optimal per period, each parent generates at most B children. By [Corollary 3.1](#), each selected candidate occupies at most one interval, so

$$|\mathbf{c}_{t-1}| \leq B |\mathbf{c}_t| + K. \quad (19)$$

In the illustration model, $B = K = 1$ and a single terminal segment imply $|\mathbf{c}_t| \leq T - t + 1$. With $B \geq 2$ the bound grows geometrically, $|\mathbf{c}_1| \leq B^{T-1} |\mathbf{c}_T| + K(B^{T-1} - 1)/(B - 1)$, although fusion typically discards most dominated children.

Theorem 4 (exactness of PTD). *Under [Assumption 1](#), for every $t \in \{1, \dots, T\}$ the optimal policy function c_t^* is piecewise affine with finitely many segments, and the policy collection $\mathbf{c}_t$ of [Definition 1](#) exists and is unique under the save-more tie rule. [Algorithm 3](#) computes the affine coefficients in closed form and locates every kink and switching point to any prescribed positive root-finding tolerance in finitely many operations.*

Remark. [Theorem 4](#) is stated in exact arithmetic: the affine coefficients are exact, and the root-finding tolerance applies to each switching point given the computed collection of the following period. On the last, unbounded cell [Algorithm 2](#) needs the sign of $\lim_{x \rightarrow \infty} \Delta(x)$. Our implementation instead brackets the switching point by doubling up to a finite wealth level of 10^{16} ; if no switching point is found below it, the solver stops with an error rather than extrapolating the comparison; this did not occur in any of our experiments.

3.3 Performance of PTD solver for the deterministic model

Now we show the accuracy and speed of the PTD algorithm by solving the deterministic dynamic optimization problem [\(9\)](#) and comparing it with DC-EGM. To the best of our knowledge, DC-EGM is among the most accurate and efficient methods for solving non-convex dynamic programming problems. We refer readers to [Iskhakov et al. \(2017\)](#) for the details of DC-EGM. Since both methods construct the policy from the same Euler equation, the Euler-equation residual used in [Iskhakov et al. \(2017\)](#) cannot discriminate between them. Instead, we measure accuracy directly by value and run a “utility contest”: every agent follows each computed policy through all 50 periods to obtain a realized consumption path $\{c_t^{(m)}\}_{t=1}^T$ for each method m , with realized discounted utility $v_m = \sum_{t=1}^T \beta^{t-1} u(c_t^{(m)})$. A method m is a strict winner if every other method’s value is strictly below $v_m - \tau$, where τ is the numerical tolerance.⁹ If several methods

⁹At each draw, we set

$$\tau = 64 \epsilon_{\text{mach}} \max \left\{ 1, \max_m \sum_{t=1}^T |\beta^{t-1} u(c_t^{(m)})| \right\},$$

where $\epsilon_{\text{mach}} = 2^{-52}$ is double-precision machine epsilon.

have values within τ of the highest value, they record a shared tie. Because risk aversion varies across agents, we measure accuracy losses by the consumption-equivalent gap relative to PTD.¹⁰

The contest gives 10,000 agents randomized preferences and initial wealth: a Latin-hypercube sample of the discount factor β , the relative risk aversion coefficient γ , and the initial wealth x_1 on $[0.90, 0.995] \times [0.5, 5] \times [3, 50]$, with the remaining parameters fixed at $T = 50$, $R = 1.1$, $\underline{x} = 3$ and $y_t = 1$ for all $t \in \{2, \dots, T\}$, as in Section 3.1. PTD is grid-free, whereas DC-EGM requires an exogenous end-of-period asset a grid; we use 100 points on $[0, 50]$, 200 points on $[0, 100]$, and 400 points on $[0, 100]$, denoted DC-EGM1, DC-EGM2, and DC-EGM3, respectively. DC-EGM4 uses 400 nodes equally spaced in $\log(1 + a)$ on $[0, 10,000]$, retaining DC-EGM3’s node count while expanding the domain and concentrating nodes near zero.¹¹

The experiments were run with Julia 1.12.7 on a laptop with an Intel Core Ultra 9 275HX CPU. Accuracy is evaluated over all 10,000 draws.¹² For timing, all five methods use the same seeded subset of 1,000 draws from this sample, with one Julia thread and one BLAS thread. Each measurement includes fresh policy construction and simulation of one 50-period path; compilation warmup is excluded. We report the mean and median elapsed time across these draws. PTD locates every switching point by Brent’s method with relative tolerance $\varepsilon_{\text{rel}} = 10^{-12}$.

Table 1 reports the results. PTD wins strictly at 656 draws and shares the highest utility within the numerical tolerance at 9,343 draws. The remaining draw is a strict DC-EGM3 win under this tolerance, with a utility advantage of only 2.15×10^{-12} . The other DC-EGM configurations have no strict wins. Across all draws and dates, including the terminal date, PTD retains 15.38 policy segments on average and at most 50. In both mean and median time, PTD is slightly slower than DC-EGM1 on the coarsest grid but faster than DC-EGM2 on a finer grid, by a factor of about 1.5, and about five and four times faster than DC-EGM3 and DC-EGM4, respectively.

Two numerical mechanisms help explain the DC-EGM losses in this experiment. First, its value upper envelope is constructed by interpolating the value correspondence on grids, so the switch points of the policy are located only up to the grid resolution: in Fig. 5, the DC-EGM policies slightly shift the PTD switching points. Second, when the optimal saving path leaves the asset-grid domain, the policy is extrapolated, and this extrapolation error can contribute to the

¹⁰For each draw, we scale PTD consumption by the same factor λ in every period until the scaled PTD path gives the same utility as the method being evaluated. The reported shortfall is $100 \max\{1 - \lambda, 0\}$ percent; for example, $\lambda = 0.95$ means that the method is equivalent to reducing PTD consumption by 5 percent in every period.

¹¹Our Julia implementation of DC-EGM is adapted from Fedor Iskhakov’s MATLAB code (<https://github.com/fediskhakov/dcegm>) and Florian Oswald’s Julia package (<https://github.com/floswald/DCEGM.jl>).

¹²We use 24 Julia threads for all deterministic and stochastic accuracy experiments, while speed comparisons use one Julia thread and one BLAS thread.

Table 1: Deterministic PTD and DC-EGM performance

Method	Grid of a	Mean time (ms)	Median time (ms)	Strict wins	Ties	Max. CEG (%)
PTD	grid free	12.339	10.822	656	9343	0.00
DC-EGM1	[0,50], 100	8.548	7.096	0	8259	16.26
DC-EGM2	[0,100], 200	19.690	16.558	0	8270	6.93
DC-EGM3	[0,100], 400	61.144	56.035	1	9313	4.54
DC-EGM4	[0,10000], 400 (log space)	51.248	46.893	0	8418	5.70

Notes: Times cover a fresh solve and one path on a single thread after warmup, using a common 1,000-draw subset. Accuracy uses 10,000 draws. Strict wins and ties count the best utilities within numerical tolerance; a tie can include several methods. Max. CEG is the largest consumption-equivalent gap (shortfall relative to PTD), in percent.

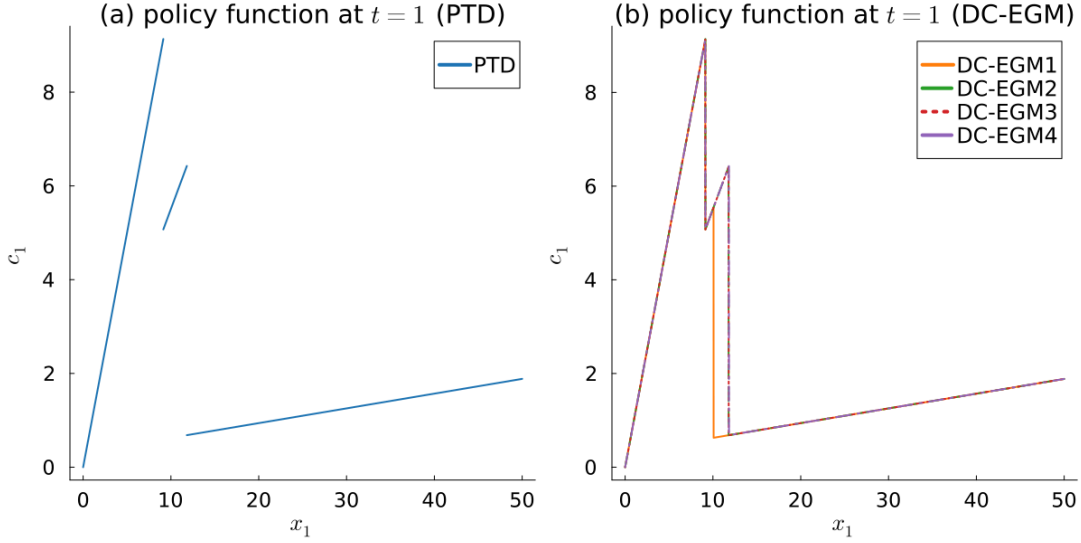


Figure 5: Policy functions derived from PTD and our Julia implementation of DC-EGM

Notes: Period-1 policies are shown for $x_1 \in [0, 50]$, with $\beta = 0.98$, $\gamma = 1$, $R = 1.1$, $\underline{x} = 3$, $T = 50$, and $y_t = 1$ for $t \in \{2, \dots, T\}$. DC-EGM1 through DC-EGM4 use the four asset grids reported in Table 1.

tail losses in Table 1. Both errors can be reduced by expanding, refining, or rearranging the a grid for each specific parameter setting, as the improvement from DC-EGM1 to DC-EGM3 shows. DC-EGM4's larger maximum consumption-equivalent gap than DC-EGM3 (5.70% versus 4.54%) may reflect a trade-off between interpolation and extrapolation error when the same number of nodes is redistributed over a wider domain. But the grid-free PTD algorithm is an off-the-shelf method: it requires no tuning and is essentially never dominated by EGM.

4 Policy-Tree Descendant Algorithm for Stochastic Models

4.1 Simulation-based approach

After describing the PTD algorithm for the deterministic problem, now we can tackle the stochastic problem (5). Before entering the details, we first introduce the distinctive feature of the

stochastic PTD methods: they are *simulation-based* solvers. For algorithms like EGM and VFI, to simulate a consumption-saving path, we need to solve the policy functions of all periods globally on state grids first, and then evaluate them along the path. PTD avoids this process by simulating the state-policy path starting from the observed state directly: at every observed state, the agent solves a deterministic problem anchored there, implements only the current action, observes the next innovation, and repeats at the newly observed state.

The key ingredients of the stochastic PTD methods are the certainty-equivalent (CE) value v^{CE} and policy c^{CE} returned by the deterministic PTD solver: we use the CE value to approximate the maximand $u(c_t) + \beta \mathbb{E}[V_{t+1}(X_{t+1}, S_{t+1}) \mid S_t = s_t]$ of the Bellman equation (5).

At an observed date- t state s , define the nominal future income sequence

$$\bar{y}_{\tau|t}(s) := \mathbb{E}[Y_{\tau} \mid S_t = s], \quad \tau = t+1, \dots, T. \quad (20)$$

Fix the complete vector $\bar{\mathbf{y}}_{t+1:T|t}(s) := [\bar{y}_{t+1|t}(s), \bar{y}_{t+2|t}(s), \dots, \bar{y}_{T|t}(s)]$ and solve the deterministic problem (9) from date t through T with this income sequence. The CE value $v_{\tau|t}^{\text{CE}}(x; s)$ is the deterministic value function v_{τ} on this CE path, and the CE policy $c_{\tau|t}^{\text{CE}}(x; s)$ is the corresponding optimal policy; the subscript $\tau \mid t$ makes the anchoring date t explicit.¹³

We propose two ways to use the CE value to approximate the maximand of the Bellman equation (5). Following the simulated certainty-equivalent approximation (SCEQ) method of Cai and Judd (2023), we can directly use the CE value to approximate the whole maximand, so that the current action is simply the CE policy at the observed state; this leads to PTD-SCEQ in Section 4.2. Alternatively, we can borrow the idea of one-step rollout from Bertsekas (2025): keep the expectation over the next innovation and use the CE value as the approximation of the next-period value; this leads to PTD-OSA in Section 4.3. Other simulation-based stochastic extensions of the deterministic PTD solver are also possible.

4.2 PTD-SCEQ

The SCEQ method of Cai and Judd (2023) repeatedly solves a deterministic model on the CE path anchored at the state actually reached, implements its current action, and then advances the system with a stochastic innovation. We implement their algorithm with the deterministic PTD solver, defined formally as Algorithm 4.

Fig. 6 illustrates a simulated path of PTD-SCEQ: at each observed state, the agent implements the first optimal action of the CE problem anchored there, and the realized innovation

¹³A similar approach is used in the case of stochastic returns, where the CE path is defined by the conditional mean of the return process.

moves her to the next observed state, where she re-solves a new CE problem.

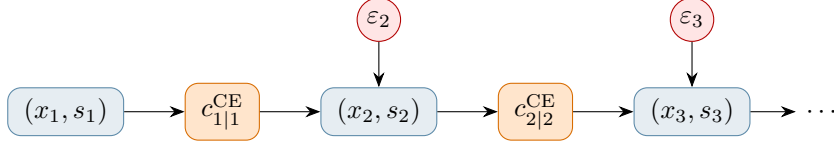


Figure 6: Simulated path of PTD-SCEQ

Notes: Blue rounded boxes are observed states, orange rounded boxes are implemented actions, and red circles are realized innovations. Each action is the first action of the CE problem re-solved at the current observed state.

Algorithm 4: PTD-SCEQ

Input : initial observed state (x_1, s_1) ; innovation sequence $\{\varepsilon_t\}_{t=2}^T$; parameters of (5);
Output: simulated path $\{(x_t, s_t, c_t)\}_{t=1}^T$;

```

1 for  $t = 1, 2, \dots, T - 1$  do
2   construct the nominal income vector  $\bar{y}_{t+1:T|t}(s_t)$  by (20);
3   solve the deterministic problem (9) on this path using Algorithm 3;
4    $c_t \leftarrow c_{t|t}^{\text{CE}}(x_t; s_t)$ ; // implement only the first action
5   observe the innovation  $\varepsilon_{t+1}$ ;
6    $(x_{t+1}, s_{t+1}) \leftarrow$  true transition (2);
7 end
8  $c_T \leftarrow c_T^*(x_T, s_T)$ ; // apply the terminal policy
9 return  $\{(x_t, s_t, c_t)\}_{t=1}^T$ ;

```

PTD-SCEQ avoids stochastic branching: every decision invokes one deterministic policy-tree solve. This feature makes it extremely fast, but it also brings a limitation: conditional-mean substitution discards precautionary saving, so the method may lose accuracy when the income process is highly volatile or the risk aversion coefficient γ is large.

4.3 PTD-OSA

Unlike PTD-SCEQ, which replaces all random variables with their conditional means, PTD-OSA (Algorithm 5)—where OSA denotes *one-step-ahead*—instead retains the distribution of the next innovation and uses the CE values as continuation values from the next period on, following the one-step rollout idea of Bertsekas (2025). At an observed state $(x_t, s_t) = (x, s)$, we approximate the expectation in the maximand of (5) using Gauss–Hermite nodes $\{\varepsilon_{t+1}^j\}_{j=1}^J$ with weights $\{w_j\}_{j=1}^J$ and choose consumption to maximize the approximate action value $\hat{Q}_t^{\text{OSA}}$:

$$\hat{Q}_t^{\text{OSA}}(c; x, s) = u(c) + \beta \sum_{j=1}^J w_j v_{t+1|t+1}^{\text{CE}}(x_{t+1}^j; s_{t+1}^j), \quad 0 \leq c \leq x, \quad (21)$$

where the quantities associated with node j are $s_{t+1}^j = g_t(s, \varepsilon_{t+1}^j)$, $y_{t+1}^j = h_{t+1}(s_{t+1}^j)$, and $x_{t+1}^j = \max\{R(x - c) + y_{t+1}^j, \underline{x}\}$. The same c appears in every summand of (21): it is the date- t

action chosen under uncertainty between the current and next periods only.

Fig. 7 illustrates a simulated PTD-OSA path up to $t = 3$. At each observed state, the agent considers the quadrature nodes for the next innovation, uses the CE values at the corresponding next-period states as continuation values, and implements the maximizer of (21).

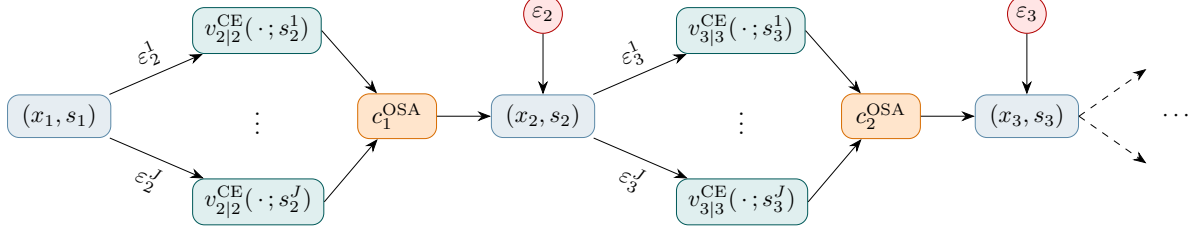


Figure 7: Simulated path of PTD-OSA

Notes: Blue rounded boxes denote observed states, teal rounded boxes denote the CE values at the next-period states associated with the quadrature nodes, orange rounded boxes denote implemented actions, and red circles denote realized innovations. At each observed state, the action maximizes the approximate action value (21); the dashed arrows indicate that the construction repeats at (x_3, s_3) .

Algorithm 5: PTD-OSA

Input : initial observed state (x_1, s_1) ; innovation sequence $\{\varepsilon_t\}_{t=2}^T$; quadrature nodes and weights $\{(\varepsilon^j, w_j)\}_{j=1}^J$; parameters of (5);

Output: simulated path $\{(x_t, s_t, c_t)\}_{t=1}^T$;

- 1 **for** $t = 1, 2, \dots, T - 1$ **do**
- 2 $s_{t+1}^j \leftarrow g_t(s_t, \varepsilon^j)$ for $j = 1, \dots, J$;
- 3 $y_{t+1}^j \leftarrow h_{t+1}(s_{t+1}^j)$ for $j = 1, \dots, J$;
- 4 build the CE value $v_{t+1|t+1}^{CE}(\cdot; s_{t+1}^j)$ at every node using Algorithm 3;
- 5 arrange brackets over $(0, x_t)$ such that every floor indicator and every CE-policy segment in (22) is unchanged within each bracket ; // endpoints: transfer boundaries and CE-policy segment boundaries
- 6 $\mathcal{C}_t \leftarrow \{x_t\} \cup \{\text{bracket endpoints}\}$; // consume-all and boundary candidates
- 7 add to $\mathcal{C}_t$ the unique root (Proposition 1) of (22) in every bracket whose Euler residual changes sign ; // Brent's method; relative tolerance 10^{-12}
- 8 $c_t \leftarrow$ a maximizer of $\widehat{Q}_t^{\text{OSA}}(c; x_t, s_t)$ over $c \in \mathcal{C}_t$; // choose smaller c in a tie
- 9 observe the innovation ε_{t+1} ;
- 10 $(x_{t+1}, s_{t+1}) \leftarrow$ true transition (2);
- 11 **end**
- 12 $c_T \leftarrow c_T^*(x_T, s_T)$; // apply the terminal policy
- 13 **return** $\{(x_t, s_t, c_t)\}_{t=1}^T$;

Proposition 1. Under Assumption 1 with $u'(0+) = +\infty$, every interior maximizer $c_t^{\text{int}} \in (0, x)$ of (21) satisfies the quadrature Euler equation

$$u'(c_t^{\text{int}}) = \beta R \sum_{j=1}^J w_j \mathbb{1}\{R(x - c_t^{\text{int}}) + y_{t+1}^j > \underline{x}\} u'\left(c_{t+1|t+1}^{\text{CE}}(x_{t+1}^j, s_{t+1}^j)\right). \quad (22)$$

On any interval of c on which every floor indicator and every CE-policy segment in (22) are fixed, the equation has at most one root. Hence every maximizer of (21) is $c = x$, an endpoint

of such an interval, or such a root.

Thus, we only need to solve the Euler equation (22), which may admit several solutions because the floor indicators switch, but at most one within each bracket of Algorithm 5. Then we compare the interior solutions and the transfer-boundary candidates with the option of consuming all the wealth. PTD-OSA incorporates one period of income risk into the current decision, thereby partially restoring precautionary saving at the cost of solving up to J CE problems per decision; when income is i.i.d., the CE problems coincide across nodes and their plans can be cached.

4.4 Performance of PTD solver for the stochastic model

Now we examine the performance of the stochastic PTD methods in the three examples of Section 2. The contenders include the standard DC-EGM for one choice and one state variable; we also use the grid operations from the G2EGM algorithm of Druedahl and Jørgensen (2017) for the examples with additional exogenous state variables.¹⁴ All exercises keep $T = 50$, $R = 1.10$, and $\underline{x} = 3$. We again measure accuracy directly by value: recalling (4), a policy π is evaluated by its expected discounted utility from the initial state,

$$V_1^\pi(x_1, s_1) = \mathbb{E}^\pi \left[\sum_{t=1}^T \beta^{t-1} u(C_t) \mid X_1 = x_1, S_1 = s_1 \right],$$

which we estimate by averaging realized discounted utilities over common hidden shock paths.

4.4.1 Example 1, i.i.d.

The i.i.d. contest compares PTD-SCEQ, PTD-OSA, and DC-EGM across 100 randomized configurations: a fixed-seed Latin-hypercube sample of the discount factor β , the relative risk aversion coefficient γ , and the initial wealth x_1 on $[0.90, 0.995] \times [0.5, 5] \times [3, 50]$. Every configuration is run in two variance panels of (6), $\sigma \in \{0.1, 0.5\}$. Within a configuration, the three methods face the same 100 hidden income paths, which are used for the Monte Carlo integration of the expected utility in (4). DC-EGM uses 6,401 log-spaced end-of-period asset points on $[0, 60,000]$: the wide asset range reduces the need for extrapolation at high wealth, while the log spacing keeps the grid dense in the low-asset region where the floor-induced kinks lie. PTD-OSA and DC-EGM share the same five-node Gauss–Hermite rule. Every method solved all 100 configurations in both panels, and all 100 configurations of the AR(1) contest below, without failures

¹⁴Our Julia implementation of G2EGM is adapted from Jeppe Druedahl’s C++ code (<https://github.com/JeppeDruedahl/G2EGM>).

or replacements. For timing, we simulate one path for each agent on a single thread.

Table 2 reports the winning shares and the mean and median single-path times. For single-path simulation, PTD-SCEQ is over 100 times faster than DC-EGM in median runtime, while PTD-OSA is about 60 times faster. However, DC-EGM can amortize its policy-construction cost across many paths for an agent with fixed preferences and may become faster for such workloads.

In Table 2, a point win means that a method attains the strictly highest estimated expected welfare in a contest. A resolved win means that the method’s paired 95 percent Monte Carlo intervals for the utility differences are favorable against both alternatives.

PTD-OSA has the largest point-win share in both panels: 52 percent at baseline and 42 percent at higher variance. PTD-SCEQ has more point wins than DC-EGM at low variance but fewer at high variance, consistent with its omission of the precautionary saving motive.

Table 2: i.i.d. contest outcomes and single-path elapsed time

Method	Point win (%)	Resolved win (%)	Mean time (s)	Median time (s)
Baseline ($\sigma = 0.1$)				
PTD-SCEQ	29	3	0.219	0.209
PTD-OSA	52	4	0.437	0.363
DC-EGM	9	0	21.281	21.057
Higher variance ($\sigma = 0.5$)				
PTD-SCEQ	13	0	0.216	0.213
PTD-OSA	42	3	0.435	0.359
DC-EGM	34	6	21.846	21.679

Notes: Times cover a fresh solve and one path on a single thread after warmup, using 100 agents per panel. Point wins denote the strictly highest estimated expected utility. Resolved wins require favorable paired 95% utility-difference intervals against both alternatives.

The pairwise comparisons in Table 3 provide additional detail. Here, consumption-equivalent gaps are signed, with positive values favoring the first-named method. At baseline, PTD-SCEQ and PTD-OSA have resolved pairwise wins over DC-EGM in 58 and 63 percent of configurations, respectively. At higher variance, these shares fall to 27 and 30 percent. In this panel, PTD-SCEQ’s 5th-percentile consumption-equivalent gap relative to DC-EGM is approximately -0.032% . PTD-OSA’s corresponding 5th-percentile consumption-equivalent gap is approximately -0.012% , consistent with its capturing only part of the precautionary saving motive.

Table 3: i.i.d. pairwise consumption-equivalent gaps (CEGs) and resolution

Method A	Method B	CEG (basis points)			Pairwise resolution (%)		
		P5	Median	P95	A resolved	B resolved	Unresolved
Baseline ($\sigma = 0.1$)							
PTD-SCEQ	PTD-OSA	-0.002	-0.000	0.000	3.00	8.00	89.00
PTD-SCEQ	DC-EGM	-0.002	0.115	0.476	58.00	0.00	42.00
PTD-OSA	DC-EGM	-0.000	0.118	0.476	63.00	0.00	37.00
Higher variance ($\sigma = 0.5$)							
PTD-SCEQ	PTD-OSA	-2.512	-0.026	0.006	0.00	28.00	72.00
PTD-SCEQ	DC-EGM	-3.199	0.000	0.514	27.00	17.00	56.00
PTD-OSA	DC-EGM	-1.235	0.002	0.510	30.00	7.00	63.00

Notes: CEG is the consumption-equivalent gap; positive values favor method A. One basis point equals 0.01%. A or B is resolved when its paired 95% utility-difference interval favors that method.

4.4.2 Example 2, AR(1)

The AR(1) contest keeps the persistent income process (7) with $\rho = 0.9$, $\sigma_\eta = 0.1$, and $Y_1 = 1$, and compares PTD-SCEQ, PTD-OSA, and G2EGM across 100 configurations from fixed-seed Latin-hypercube sampling of (β, γ, x_1) on $[0.90, 0.995] \times [0.5, 3.5] \times [3, 50]$, with 100 hidden income paths per configuration for the Monte Carlo integration of the expected utility in (4). Relative to the i.i.d. contest, the upper bound of γ is 3.5 to limit the cost of G2EGM policy construction at high risk aversion and high patience. The comparator G2EGM uses a post-decision grid with 1,601 log-spaced asset points on $[0, 30,000]$ and 61 log-income nodes, and a $1,201 \times 41$ common grid on cash and log income. Both log-income grids use centered sinh spacing on $[-3, 3]$; the common cash grid uses shifted log spacing on $[3, 33,020]$. PTD-OSA and G2EGM share the same five-node Gauss–Hermite rule for numerical integration over income shocks. For timing, we simulate the same income path on a single thread for five agents with $\gamma \in \{1, 1.5, 2, 2.5, 3\}$, fixing $\beta = 0.98$ and $x_1 = 16$.

Table 4 reports the winning shares and the mean and median single-path times across these five agents. G2EGM has the largest point-win share at 49 percent, followed by PTD-SCEQ at 23 percent and PTD-OSA at 16 percent. For a single-path simulation, G2EGM averages about 200 seconds, roughly ten times the DC-EGM runtime in the baseline i.i.d. case, while both PTD methods remain below one second on average. This reflects PTD’s avoidance of grids over the state space, although higher-dimensional shocks can increase its numerical integration costs.

Table 4: AR(1) contest outcomes and single-path elapsed time

Method	Point win (%)	Resolved win (%)	Mean time (s)	Median time (s)
PTD-SCEQ	23	1	0.097	0.096
PTD-OSA	16	0	0.523	0.495
G2EGM	49	6	201.022	196.948

Notes: Times cover a fresh solve and one path on a single thread after warmup; means and medians are across the five preference settings. Point and resolved wins are defined as in Table 2.

Fig. 8 plots the date-1 consumption policies with log utility and $\beta = 0.98$. All three surfaces retain the discrete regime switches created by the floor; the slices obtained by fixing date-1 income show the resulting discontinuities.

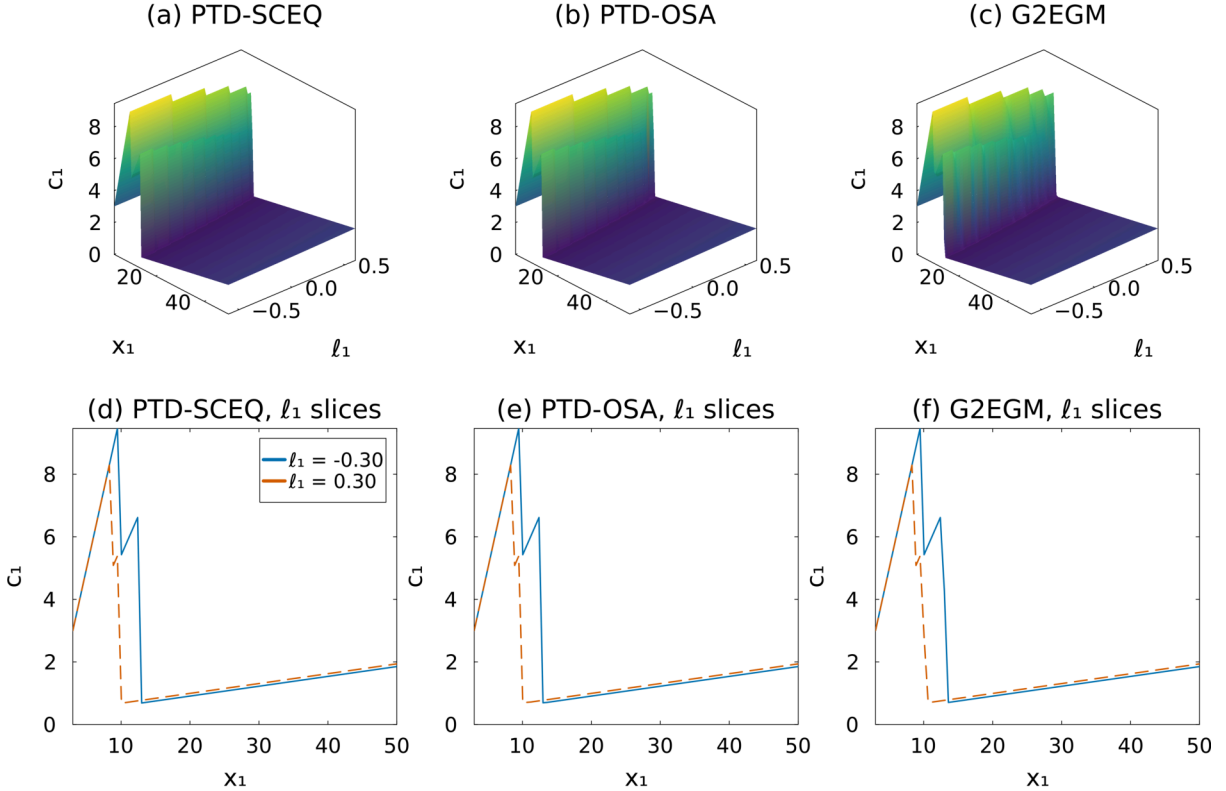


Figure 8: Date-1 consumption policies in the scalar AR(1) example

Notes: x_1 is date-1 cash on hand, ℓ_1 is date-1 log income, and c_1 is date-1 consumption. The bottom row shows policy slices at $\ell_1 \in \{-0.3, 0.3\}$.

Fig. 9 replays five heterogeneous contest configurations near initial cash points 5.5, 10.75, 16, 21.25, and 26.5, with 20 income paths per configuration. It shows agents who remain at the consumption floor as well as agents who continue to save.

Table 5 reports the pairwise consumption-equivalent gaps and resolution shares. G2EGM has resolved pairwise wins over PTD-SCEQ and PTD-OSA in 11 and 6 percent of configurations, respectively. The 5th-percentile consumption-equivalent gaps of PTD-SCEQ and PTD-OSA

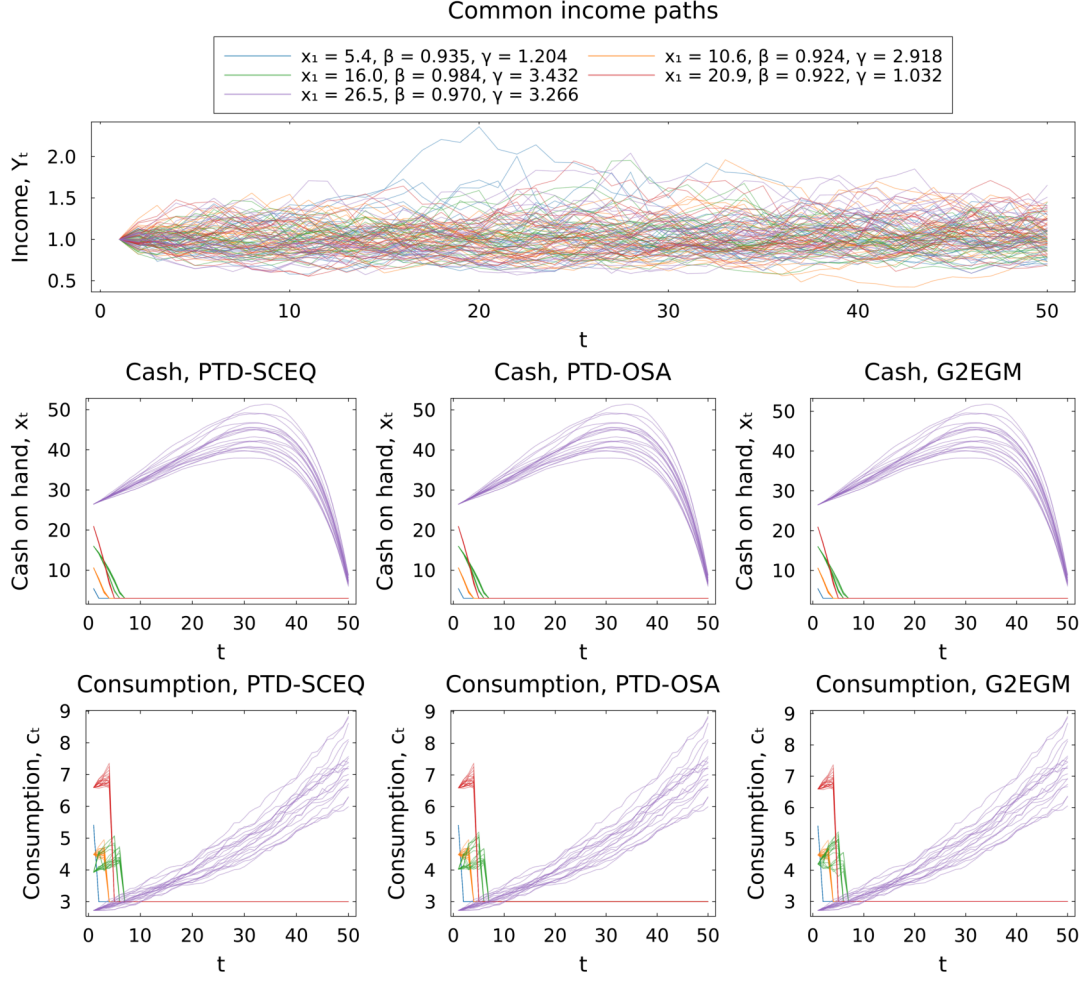


Figure 9: Paths for five AR(1) contest configurations

Notes: The five groups use the contest configurations closest to the stated initial cash points, with preferences varying across groups. Each group uses its first 20 original income paths, shared by all three methods.

relative to G2EGM are approximately -0.0076% and -0.0029% , respectively. The smaller lower-tail shortfall under PTD-OSA is consistent with its capturing part of the precautionary saving motive.

Table 5: AR(1) pairwise consumption-equivalent gaps (CEGs) and resolution

Method A	Method B	CEG (basis points)			Pairwise resolution (%)		
		P5	Median	P95	A resolved	B resolved	Unresolved
AR(1) ($\rho = 0.9$, $\sigma = 0.1$, $Y_1 = 1$)							
PTD-SCEQ	PTD-OSA	-0.650	-0.001	0.010	1.00	14.00	85.00
PTD-SCEQ	G2EGM	-0.761	-0.000	0.263	1.00	11.00	88.00
PTD-OSA	G2EGM	-0.287	-0.000	0.175	2.00	6.00	92.00

Notes: CEG is the consumption-equivalent gap; positive values favor method A. One basis point equals 0.01%. A or B is resolved when its paired 95% utility-difference interval favors that method.

4.4.3 Example 3, HSZ

For the HSZ example in (8) with three continuous state variables, solving the model by EGM is too time-consuming for a heterogeneous agent utility contest. Thus, we use log utility and $\beta = 0.98$. Log earnings and log medical expenditure follow independent AR(1) processes with persistence 0.9, innovation standard deviation 0.1, and unconditional log means $\mu_e = \log 2$ and $\mu_m = \log 1.5$; net income is earnings minus medical expenditure. Initial earnings and medical expenditure are $\exp(e_1) = 2$ and $\exp(m_1) = 1.5$, giving initial net income $y_1 = 0.5$. We use five initial cash points, $\{5.5, 10.75, 16, 21.25, 26.5\}$. For the utility comparison, we evaluate the same 100 shock paths at every starting point.

G2EGM solves the model on three-dimensional tensor grids with 401 asset or cash nodes and 17 nodes in each income/expense dimension, hence $401 \times 17 \times 17 = 115,889$ nodes per age in both the log-spaced end-of-period asset grid spanning $[0, 300]$, and the log-spaced cash-on-hand grid spanning $[3, 379]$. These marginals are much sparser than in the AR(1) case. PTD-OSA and G2EGM share the same 5×5 tensor quadrature rule for numerical integration over earnings and medical-expenditure shocks.

At the baseline grid resolution, the methods sometimes choose different saving paths in Fig. 10.

Table 6 reports the 5th percentile, median, and 95th percentile of consumption-equivalent gaps across the 100 paired paths at each starting cash level, with positive values favoring PTD over G2EGM. At $x_1 = 21.25$, G2EGM produces floor transfers on 26 of 100 paths, compared with none under either PTD method. Both PTD methods have a median consumption-equivalent gap relative to G2EGM of approximately 0.62%, with a 95th percentile of approximately 15.15%.¹⁵ Because the G2EGM outcome at $x_1 = 21.25$ changes this much with small changes in the exogenous grids, we read these gaps as evidence of the sensitivity of grid-based policies near the floor in this example, rather than as a precise measure of the welfare cost of using G2EGM. PTD-SCEQ and PTD-OSA, which use no exogenous grid, agree closely with each other at every starting point.

¹⁵With agent setting and shocks fixed, increasing each exogenous grid from 17 to 18 nodes raises initial G2EGM consumption at $x_1 = 21.25$ from 1.188 to 2.635 and the number of floor-transfer paths from 26 to 73; 19 nodes gives 17 such paths. The 17- and 19-node grids contain the initial earnings and medical states exactly, whereas the 18-node grid requires interpolation across them. This illustrates that increasing the number of grid points need not improve EGM results monotonically in higher-dimensional problems, even if the approximation converges to the true policy as the grid becomes arbitrarily fine.

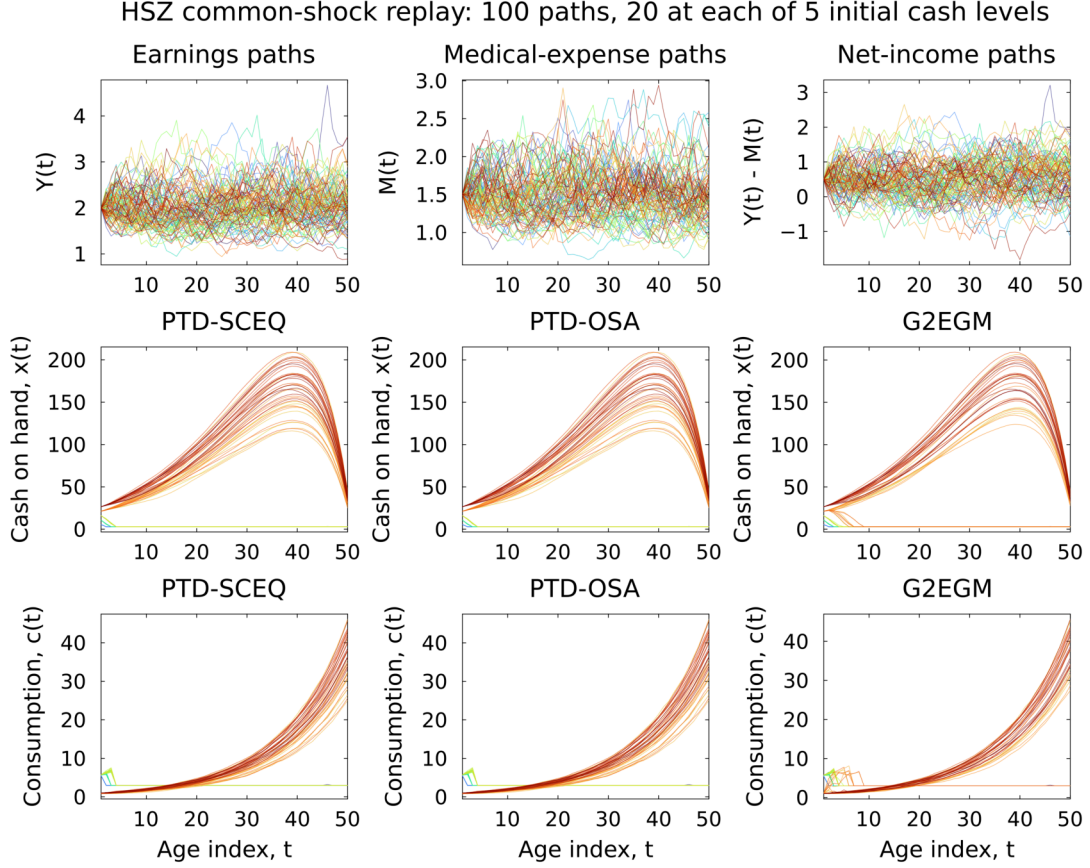


Figure 10: Paths in the HSZ example

Notes: The top row shows the common earnings, medical-expenditure, and net-income paths. The middle and bottom rows show cash and consumption under PTD-SCEQ, PTD-OSA, and G2EGM.

Table 6: HSZ consumption-equivalent gaps (CEGs) and floor transfers

Initial cash	Floor-transfer paths			PTD-SCEQ vs G2EGM CEG (%)			PTD-OSA vs G2EGM CEG (%)		
	PTD-SCEQ	PTD-OSA	G2EGM	P5	Median	P95	P5	Median	P95
5.50	100/100	100/100	100/100	0.000	0.000	0.000	0.000	0.000	0.000
10.75	100/100	100/100	100/100	-0.000	0.000	0.000	-0.000	0.000	0.000
16.00	100/100	100/100	100/100	-0.001	0.000	0.062	-0.001	0.000	0.062
21.25	0/100	0/100	26/100	-0.036	0.620	15.146	-0.039	0.622	15.146
26.50	0/100	0/100	0/100	-0.012	0.001	0.040	-0.010	0.001	0.041

Notes: Floor counts are paths receiving at least one transfer. CEGs scale each G2EGM consumption path to match PTD utility; positive values favor PTD. P5 and P95 are the 5th and 95th percentiles across paired paths.

For timing, we simulate 100 paths at $x_1 = 16$ on a single thread, reusing G2EGM’s policy and PTD’s cached plans across paths. Table 7 reports construction time and the mean and median of construction plus one-path simulation time. G2EGM has a substantial construction cost but simulates subsequent paths much faster, whereas both PTD methods optimize along each simulated path. PTD-SCEQ uses a single certainty-equivalent path at each decision, with

a runtime comparable to the AR(1) case. The longer runtime of PTD-OSA reflects the need to evaluate 25 quadrature continuations for each saving decision, compared with five in the AR(1) case.

Table 7: HSZ construction and single-path times

Method	Construction (s)	Mean time (s)	Median time (s)
PTD-SCEQ	0.000	0.067	0.063
PTD-OSA	0.000	2.092	2.051
G2EGM	636.526	636.526	636.526

Notes: Log utility, $\beta = 0.98$, and $x_1 = 16$. Mean and median times add the fixed construction cost to simulation times across 100 paths sharing the same policy or cached plans. PTD is a simulation-based solver with no separate policy-construction phase. All times are in seconds and exclude warmup.

5 Conclusion

In this paper we develop the policy-tree descendant (PTD) algorithm for a class of discrete-time dynamic models with piecewise-affine optimal policies and use the consumption-saving model with a safety net as an illustration. This class combines HARA preferences with piecewise-affine transitions and a nonnegative-saving constraint, as specified in [Assumption 1](#). The consumption floor makes the problem non-convex: the value function acquires kinks, the consumption policy becomes discontinuous, and these features multiply over backward induction. PTD embraces this structure instead of smoothing it over: it represents the policy function as a tree of affine segments, each produced from its parent in closed form through the inverse Euler equation and refined by locating the single crossing of the competing value segments to a prescribed numerical tolerance.

The deterministic solver is entirely grid-free, and it serves in turn as the engine of two simulation-based stochastic solvers: PTD-SCEQ, which follows the simulated certainty-equivalent approximation of [Cai and Judd \(2023\)](#), and PTD-OSA, which implements the one-step rollout idea of [Bertsekas \(2025\)](#). In the utility contest of [Section 3](#), PTD matches the highest lifetime utility within numerical tolerance in 9,999 of 10,000 randomized configurations at a speed comparable to the coarsest DC-EGM configuration. In the model with i.i.d. income risk, both PTD variants record more resolved pairwise welfare wins than losses against DC-EGM. Most AR(1) contests remain unresolved at the available Monte Carlo precision. In the three-state HSZ example based on [Hubbard et al. \(1995\)](#), PTD also yields welfare gains over G2EGM at some initial wealth levels on the tested grid. PTD-SCEQ is the fastest method in the reported stochastic single-path comparisons. In the HSZ example, PTD-SCEQ and PTD-

OSA are approximately 9,500 and 300 times faster than G2EGM, respectively, for single-path simulation.

One computational tradeoff between PTD and EGM concerns the cost of locating the switching points at which the agent moves between competing local optima. EGM declines to solve any equation—it compares candidate values on a fixed grid, so the switching points are resolved only up to the grid resolution, paths that leave the grid domain are extrapolated, the grid must be tuned to each calibration, and its size grows exponentially with the number of persistent shocks. PTD declines to lay down a grid—it represents the deterministic policy through affine segments and locates switching points using one-dimensional root finding. Its stochastic variants build certainty-equivalent plans as needed during simulation. By avoiding policy grids over exogenous states, PTD can achieve larger speed advantages as the number of exogenous state variables increases, particularly when using PTD-SCEQ.

The practical implication is that relative computational costs depend on the number of paths simulated per distinct DP problem and the number of exogenous state variables. When many paths share a problem, EGM can reuse its policy and amortize its construction cost. PTD can also reuse cached certainty-equivalent plans only when planning inputs coincide. The potential benefit of PTD is therefore strongest when heterogeneity requires many distinct solutions and few paths use each solution. For example, PTD could facilitate estimation of extensions of the structural models of [Hubbard et al. \(1995\)](#) and [De Nardi et al. \(2010\)](#) in which agents’ preferences are drawn from a parametric distribution or agents have different income processes, whereas repeated policy construction for distinct preference and income-process specifications can pose a formidable computational challenge for EGM.

The limitations of PTD map out the directions for future research. The descendant step is available in closed form only within the piecewise-affine class, and the policy tree is a one-dimensional object, so the method as it stands accommodates a single endogenous state. A natural first extension is the retirement model of [Iskhakov et al. \(2017\)](#), which satisfies the monotone-saving condition on which the refinement step rests; the policy tree would then carry one family of segments per discrete alternative. A second is to equip the stochastic solvers, whose accuracy we have established only by simulation, with error bounds or a deeper rollout. The most ambitious extension is to multiple endogenous states, which the endogenous grid framework accommodates ([Druedahl and Jørgensen, 2017](#)): the one-dimensional policy segments would become simplex-based policy cells, and the open challenge is a geometric analog of the monotone-saving ordering that controls their proliferation. We leave these extensions to future work.

Appendix A Mathematical Proofs

Proof of Theorem 1. Let the active branch be $x' = R_i[x - c_{t-1}^{k+1}(x)] + y_i$ with $R_i > 0$, where x' denotes the next-period state. By the definition of $\psi_{\beta R_i}$, the Euler equation in Definition 2 is equivalent to the inverse Euler equation

$$c_{t-1}^{k+1}(x) = \psi_{\beta R_i}(c_t^k(x')) = p + q c_t^k(x'). \quad (\text{A.1})$$

Substituting the affine parent into (A.1) gives

$$(1 + qb_1 R_i) c_{t-1}^{k+1}(x) = p + q(b_0 + b_1 y_i) + qb_1 R_i x.$$

Because $q, R_i > 0$ and $b_1 \geq 0$, the coefficient $1 + qb_1 R_i$ is strictly positive, which yields (14) with $d_1 \in [0, 1)$. All domain restrictions are affine inequalities, so their intersection is an interval. $\square$

Proof of Theorem 2. The proof uses Lemma 3 at the parent date t , which holds at $t = T$ by Assumption 1(iv) and is established for $t < T$ in the proof of Theorem 4 before the present theorem is applied. Fix $x > 0$ and an optimal action $c > 0$, and let $a = x - c$. If the action lies on a corner candidate, the claim holds. Otherwise, $a > 0$ lies in the interior of a positive-slope branch cell, since reducing saving on a flat branch would strictly improve the payoff. Write $z = R_i a + y_i$ and $H(a') = u(x - a') + \beta v_t(R_i a' + y_i)$ for a' near a . By Lemma 3,

$$H'_\pm(a) = -u'(c) + \beta R_i u'(c_t^*(z)), \quad (\text{A.2})$$

and since c_t^* never jumps up, $H'_-(a) \leq H'_+(a)$. Local optimality requires $H'_-(a) \geq 0 \geq H'_+(a)$, so $H'_-(a) = H'_+(a) = 0$: the parent policy is continuous at z and $u'(c) = \beta R_i u'(c_t^*(z))$. Let c_t^{*k} be the refinement whose domain contains $[z - \epsilon, z]$ for some $\epsilon > 0$. The action satisfies the Euler equation of Definition 2 with this parent. For $z' \in (z - \epsilon, z)$ close to z , the inverse state map χ of this parent, which is continuous and strictly increasing, yields child actions that satisfy $0 < c < x$ and $a \in \text{int } A_i$ by continuity. Hence the child is nondegenerate, $x = \chi(z)$ lies in the closure of its domain, and the child prescribes c at x .

In the illustration model the only kink of f_{t-1} is the right end $\bar{a} = (\underline{x} - y_t)/R$ of the flat safety-net branch, where $f_{t-1}(\bar{a}) = f_{t-1}(0)$, so holding saving at $\bar{a}$ is strictly dominated by consuming all wealth. $\square$

Proof of Theorem 3. For $t < T$ write the objective as

$$H_x(a) = u(x - a) + W_t(a), \quad W_t(a) = \beta v_{t+1}(f_t(a)), \quad a \in [0, x).$$

Take $x_2 > x_1$ and optimal choices a_2, a_1 , and suppose $a_2 < a_1$. Both choices are feasible at both states. Indeed,

$$0 \leq a_2 < a_1 < x_1 < x_2.$$

The two cross-consumption choices lie between the original optimal consumptions $x_1 - a_1$ and $x_2 - a_2$, and thus also belong to the open utility domain. Revealed preference implies

$$\begin{aligned} W_t(a_1) - W_t(a_2) &\geq u(x_1 - a_2) - u(x_1 - a_1), \\ W_t(a_1) - W_t(a_2) &\leq u(x_2 - a_2) - u(x_2 - a_1). \end{aligned}$$

With $d = a_1 - a_2 > 0$, strict concavity says that $u(z + d) - u(z)$ is strictly decreasing in z . Since $x_2 - a_1 > x_1 - a_1$, the first right-hand side is strictly greater than the second, a contradiction. $\square$

Proof of Lemma 2. The saving of the consume-all segment is 0 and that of a corner candidate is $\bar{a}$. On the interior of a child's domain, Definition 2 requires $a \in \text{int } A_i$ and $R_i a + y_i \in \text{int dom}(c_t^{*k})$,

so the saving lies in $S_{k,i}$, an open interval. Bands of children along different branches lie in disjoint open cells. Along the same branch, $a \mapsto R_i a + y_i$ is injective and the interiors of distinct parent domains are disjoint, so the bands are disjoint. The points 0 and $\bar{a}$ are endpoints of branch cells, so they belong to no band $S_{k,i} \subseteq \text{int } A_i$, and distinct corner candidates have distinct points. $\square$

Proof of Lemma 4. For a corner candidate saving is fixed, so the continuation term in (18) is constant and $v'_p = u' \circ p$. For the child of $c_t^{\star k}$ along branch i with slope d_1 , next-period wealth z lies in $\text{int dom}(c_t^{\star k})$, where $v'_t(z) = u'(c_t^{\star k}(z))$ by Lemma 3. Differentiating (18) and using the Euler equation in Definition 2 gives

$$v'_p(x) = d_1 u'(p(x)) + (1 - d_1) \beta R_i u'(c_t^{\star k}(z)) = u'(p(x)).$$

Hence $\Delta'(x) = u'(p(x)) - u'(q(x)) < 0$, because $p(x) > q(x)$ by Lemma 2 and u' is strictly decreasing. $\square$

Proof of Theorem 4 and of Lemma 3 for $t < T$. For each t , let P_t be the statement that $c_t^{\star}$ is represented by a policy collection $\mathbf{c}_t$ whose refinements are affine with nonnegative slopes, and that the conclusions of Lemma 3 hold at t . By Assumption 1(iv), P_T holds once adjacent terminal pieces with identical coefficients are merged. We show that P_t implies P_{t-1} for $t \in \{2, \dots, T\}$ and that Algorithm 3 computes $\mathbf{c}_{t-1}$ from $\mathbf{c}_t$. Theorem 2 and Lemma 4 at $t-1$ use only P_t , so their proofs are part of this induction.

Candidates. By Theorem 1, each parent refinement generates along each positive-slope branch at most one child, which is affine with slope in $[0, 1)$ and closed-form coefficients; corner candidates are affine with slope one. There are finitely many candidates and every candidate action is feasible. By Theorem 2, at every $x > 0$ some candidate whose domain contains x prescribes an optimal action. Hence $v_{t-1}(x)$ is the maximum of the value segments defined at x , and $c_{t-1}^{\star}(x)$ is the action of a maximizing candidate, the one that saves most in case of a tie.

Fusion. Fix a cell $[e_k, e_{k+1}]$ of Algorithm 2; each candidate is defined either on the whole cell or on none of its interior. We claim that after each candidate is processed, E lists restrictions of processed candidates in increasing saving order, with abutting domains that cover the cell, and represents their upper envelope under the save-more tie rule. Suppose the claim holds before $q = q_j$ is processed. By Lemma 4, for each processed candidate p the set of points of the cell where $v_q \geq v_p$ is empty or a closed interval that contains the right end of the cell. Hence the set where q weakly beats all processed candidates is a right-hand interval $[\xi, e_{k+1}]$ of the cell, possibly empty. Scanning the envelope from the right, Algorithm 2 removes every piece on which q weakly beats it, and either stops at a piece on which q is worse except possibly at its right end, so that ξ is that end, or locates ξ inside the piece where the two cross. Appending q on $[\xi, e_{k+1}]$ restores the claim. Concatenating the cells therefore yields $c_{t-1}^{\star}$. Merging adjacent segments with identical coefficients makes every refinement maximal, so $\mathbf{c}_{t-1}$ satisfies Definition 1 and is unique.

Properties at $t-1$. The refinements are affine with nonnegative slopes. By Assumption 1(ii)–(iii) and continuity of v_t , Berge’s maximum theorem applies locally with consumption restricted to a compact subset of $\mathbb{R}_{++}$, so v_{t-1} is continuous. On the interior of each refinement, v_{t-1} coincides with the value segment of a candidate, so $v'_{t-1} = u'(c_{t-1}^{\star})$ there by Lemma 4. Since v_{t-1} is continuous and $c_{t-1}^{\star}$ has finitely many affine pieces, the one-sided derivatives at the joins are the limits $u'(c_{t-1}^{\star}(x \pm))$. Finally, $t-1 \leq T-1$, so the selected saving is nondecreasing by Theorem 3, and $c_{t-1}^{\star}(x+) - c_{t-1}^{\star}(x-) = -(a_{t-1}^{\star}(x+) - a_{t-1}^{\star}(x-)) \leq 0$. This proves P_{t-1} .

Operations. At every date, (19) bounds the number of candidates, hence of cells and root-finding problems, and Brent’s method reaches any positive tolerance in finitely many steps. Since the horizon is finite, the total number of operations is finite. $\square$

Proof of Lemma 1. We show by backward induction that (iii) holds at every date. The induction in the proof of Theorem 4 derives P_{t-1} from P_t and (iii) at $t-1$, so it suffices to derive (iii) at $t-1$ from P_t and (iii) at t .

At T , c_T^* is positive, has finitely many affine pieces with nonnegative slopes, never jumps up and is right-continuous under the save-more rule. On a compact interval $[a, b] \subset \mathbb{R}_{++}$ it is therefore bounded below by the minimum of its values at a and at the finitely many joins in $(a, b]$, which is positive, and above by b .

Suppose P_t and (iii) at t hold, and fix $[a, b] \subset \mathbb{R}_{++}$. Since f_{t-1} is nondecreasing with $f_{t-1}(0) > 0$, it maps $[0, b]$ into a compact interval $I \subset \mathbb{R}_{++}$, on which $c_t^* \geq \delta$ for some $\delta > 0$ by (iii) at t . By Lemma 3, v_t is continuous with one-sided derivatives bounded by $u'(\delta)$ on I , so it is Lipschitz there, and $W(s) := \beta v_t(f_{t-1}(s))$ is Lipschitz on $[0, b]$ with constant $L := \beta \bar{R} u'(\delta)$, where $\bar{R}$ is the largest slope of f_{t-1} . Because $u'(0+) = +\infty$, there is $\epsilon \in (0, a]$ with $u' > L$ on $(0, \epsilon)$. For $x \in [a, b]$ and $0 < c < c' \leq \epsilon$,

$$[u(c') + W(x - c')] - [u(c) + W(x - c)] \geq \int_c^{c'} u'(s) ds - L(c' - c) > 0,$$

so the objective is strictly increasing on $(0, \epsilon]$. It is continuous on the compact interval $[\epsilon, x]$ and therefore attains its maximum there, and every maximizer lies in $[\epsilon, b]$. This is (iii) at $t-1$. $\square$

Proof of Proposition 1. Fix (t, x_t, s_t) and let $c_t^* \in (0, x_t)$ be any OSA maximizer. To simplify notation, define

$$\begin{aligned} z_{t+1}^j(c_t) &:= R(x_t - c_t) + y_{t+1}^j, & x_{t+1}^j(c_t) &:= \max\{z_{t+1}^j(c_t), \underline{x}\}, \\ V_j(\hat{x}_{t+1}) &:= v_{t+1|t+1}^{\text{CE}}(\hat{x}_{t+1}; s_{t+1}^j), & F_j(c_t) &:= V_j(x_{t+1}^j(c_t)). \end{aligned}$$

Thus

$$\hat{Q}_t^{\text{OSA}}(c_t; x_t, s_t) = u(c_t) + \beta \sum_{j=1}^J w_j F_j(c_t). \quad (\text{A.3})$$

Gauss–Hermite weights are strictly positive.

Define

$$\mathcal{J}_+ := \{j : z_{t+1}^j(c_t^*) > \underline{x}\},$$

and write $x_{t+1}^{j,*} := x_{t+1}^j(c_t^*)$. For every $j \in \mathcal{J}_+$, select a CE-optimal plan from $x_{t+1}^{j,*}$ and denote its first consumption and assets by

$$\bar{c}_{t+1}^j := c_{t+1|t+1}^{\text{CE}}(x_{t+1}^{j,*}; s_{t+1}^j) > 0, \quad \bar{a}_{t+1}^j := x_{t+1}^{j,*} - \bar{c}_{t+1}^j.$$

If next-period cash changes locally from $x_{t+1}^{j,*}$ to $\hat{x}_{t+1}$, hold $\bar{a}_{t+1}^j$ and all later choices fixed and change next-period consumption to $\bar{c}_{t+1}^j + \hat{x}_{t+1} - x_{t+1}^{j,*}$. This feasible deviation gives the differentiable lower support

$$\ell_j(\hat{x}_{t+1}) := u(\bar{c}_{t+1}^j + \hat{x}_{t+1} - x_{t+1}^{j,*}) + V_j(x_{t+1}^{j,*}) - u(\bar{c}_{t+1}^j) \leq V_j(\hat{x}_{t+1}). \quad (\text{A.4})$$

Here $V_j(x_{t+1}^{j,*}) - u(\bar{c}_{t+1}^j)$ is the discounted value of the selected plan after date $t+1$. The inequality holds because the left-hand side is the value of this particular feasible plan from $\hat{x}_{t+1}$, whereas $V_j(\hat{x}_{t+1})$ is the maximum over all feasible CE plans. This is the lazy-policy construction used by Clausen and Strub (2020).

For $j \in \mathcal{J}_+$, let $f_j(c_t) := \ell_j(z_{t+1}^j(c_t))$; this is a differentiable lower support of F_j at c_t^* with

$f'_j(c_t^*) = -Ru'(\bar{c}_{t+1}^j)$, since the chosen plan fixes $\bar{c}_{t+1}^j$ and $x_{t+1}^{j,*}$. For $j \notin \mathcal{J}_+$, set

$$f_j(c_t) := V_j(\underline{x}).$$

This constant is a lower support of F_j by monotonicity of V_j , since $x_{t+1}^j(c_t) \geq \underline{x}$ for every c_t , with equality at c_t^* .

Therefore

$$L(c_t) := u(c_t) + \beta \sum_{j=1}^J w_j f_j(c_t)$$

is a differentiable lower support of $\widehat{Q}_t^{\text{OSA}}(c_t; x_t, s_t)$ at c_t^* . Since c_t^* is an interior maximizer, the constant $U(c_t) := \widehat{Q}_t^{\text{OSA}}(c_t^*; x_t, s_t)$ is a differentiable local upper support. The Differentiable Sandwich Lemma (Clausen and Strub, 2020, Lemma 1) implies that the action value is differentiable at c_t^* and

$$0 = L'(c_t^*) = u'(c_t^*) - \beta R \sum_{j \in \mathcal{J}_+} w_j u'(\bar{c}_{t+1}^j). \quad (\text{A.5})$$

Since $\mathcal{J}_+$ is exactly the set selected by the strict indicator in (22), this proves that equation for every interior maximizer.

Now fix an interval of c on which the set $\mathcal{J}_+$ and, for each $j \in \mathcal{J}_+$, the affine CE-policy segment $c_{t+1|t+1}^{\text{CE}}(x'; s_{t+1}^j) = b_0^j + b_1^j x'$ with $b_1^j \geq 0$ (Theorem 1) are fixed. On it, (22) reads $r(c) = 0$ with

$$r(c) := u'(c) - \beta R \sum_{j \in \mathcal{J}_+} w_j u'(b_0^j + b_1^j (R(x_t - c) + y_{t+1}^j)).$$

The first term is strictly decreasing in c , and each argument in the sum is nonincreasing in c , so the sum is nondecreasing; hence r is strictly decreasing and has at most one root. Finally, $c = 0$ is not a maximizer: each V_j is continuous with one-sided derivatives bounded on compact sets by Lemmas 1 and 3, while $u'(0+) = +\infty$, so the objective increases near $c = 0$. Every maximizer is therefore $c = x_t$, an endpoint of such an interval, or an interior point of one, where it satisfies (22) and is its unique root. $\square$

Appendix B Fusion with multiple transition branches

Consider three dates, $u(c) = 1 - 1/c$, $\beta = 0.95$, and $c_3(x) = x$. The subscript of f_t denotes the date: saving $a_t = x_t - c_t$ leads to next-period cash on hand $x_{t+1} = f_t(a_t)$, for $t = 1, 2$, with transitions

$$f_2(a) = \begin{cases} 2 + 0.5a, & 0 \leq a \leq 1, \\ 2.5 + 2(a - 1), & a \geq 1, \end{cases} \quad f_1(a) = \begin{cases} 1 + 0.5a, & 0 \leq a \leq 1.5, \\ 1.75 + 2(a - 1.5), & 1.5 \leq a \leq 4, \\ 6.75 + 1.5(a - 4), & a \geq 4. \end{cases}$$

Both transitions are continuous and increasing, with nonmonotone marginal returns in f_1 . This example illustrates how [Algorithm 2](#) fuses descendants from multiple positive-slope transition branches and corner candidates.

We solve the model for $x_1 \in [0.01, 40]$, with continuation domains that contain all reachable wealth levels. We use a direct optimizer based on Brent’s method from the Julia optimization library `Optim.jl` to verify the accuracy of [Algorithm 2](#). [Table 8](#) reports the resulting errors; [Figure 11](#) compares the policies.

Table 8: Verification of the multiple-branch example

Date	States	Pieces	Max. $ v - v^{\text{ref}} $	Max. $ c - c^{\text{ref}} $
1	820	7	1.78×10^{-15}	5.38×10^{-8}
2	807	3	1.42×10^{-14}	7.11×10^{-15}

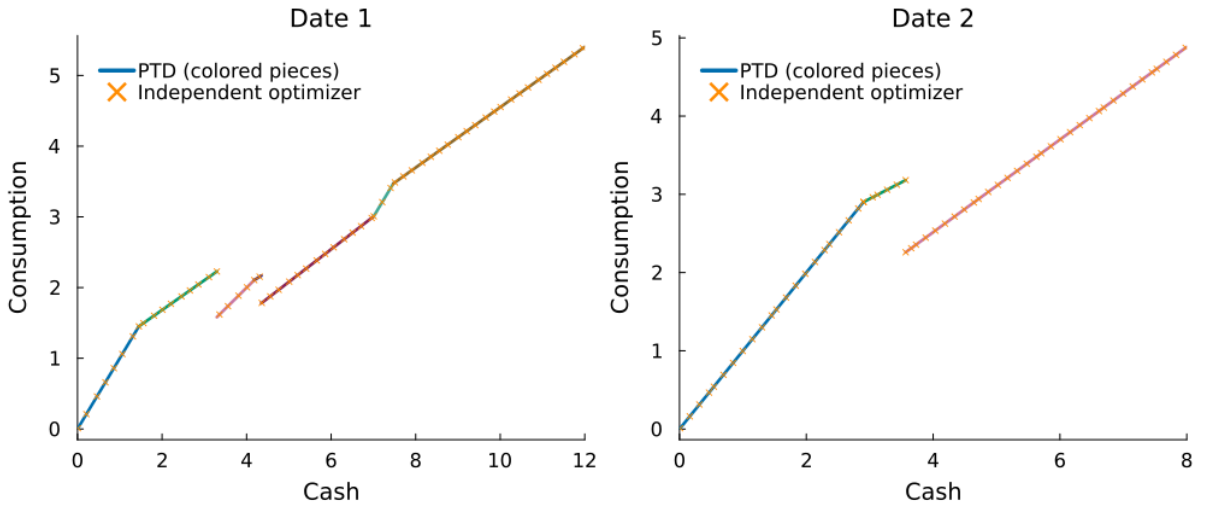


Figure 11: Consumption policies near their switching points. Different colors distinguish adjacent PTD policy pieces; orange crosses show independent direct optimization.

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used Claude (Anthropic) and Codex (OpenAI) in order to improve the language and readability of the manuscript, to assist with code development and proof checking. Afterwards the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

References

- BERTSEKAS, D. P. (2025): *A Course in Reinforcement Learning*: Athena Scientific, 2nd edition.
- CAI, Y., AND K. L. JUDD (2023): “A simple but powerful simulated certainty equivalent approximation method for dynamic stochastic problems,” *Quantitative Economics*, 14 (2), 651–687.
- CAMPBELL, J. Y. (2018): *Financial Decisions and Markets: A Course in Asset Pricing*: Princeton University Press.
- CAPATINA, E., AND M. P. KEANE (2026): “Health shocks, health insurance, human capital, and the dynamics of earnings and health,” *Journal of Political Economy*, 134 (6), 1714–1774.
- CARROLL, C. D. (2006): “The method of endogenous gridpoints for solving dynamic stochastic optimization problems,” *Economics Letters*, 91 (3), 312–320.
- CARROLL, C. D., AND M. S. KIMBALL (1996): “On the concavity of the consumption function,” *Econometrica*, 64 (4), 981–992.
- CLAUSEN, A., AND C. STRUB (2020): “Reverse calculus and nested optimization,” *Journal of Economic Theory*, 187, 105019.
- DE NARDI, M., E. FRENCH, AND J. B. JONES (2010): “Why do the elderly save? The role of medical expenses,” *Journal of Political Economy*, 118 (1), 39–75.
- DE NARDI, M., E. FRENCH, AND J. B. JONES (2009): “Life expectancy and old age savings,” *American Economic Review, Papers and Proceedings*, 99 (2), 110–115.
- DECHERT, W. D., AND K. NISHIMURA (1983): “A complete characterization of optimal growth paths in an aggregated model with a non-concave production function,” *Journal of Economic Theory*, 31 (2), 332–354.

- DOBRESCU, L., AND A. SHANKER (2022): “Fast upper-envelope scan for discrete-continuous dynamic programming,” Working Paper.
- DRUEDAHL, J., AND T. H. JØRGENSEN (2017): “A general endogenous grid method for multi-dimensional models with non-convexities and constraints,” *Journal of Economic Dynamics and Control*, 74, 87–107.
- FAN, X., A. SESHADRI, AND C. TABER (2024): “Estimation of a life-cycle model with human capital, labor supply, and retirement,” *Journal of Political Economy*, 132 (1), 48–95.
- FELLA, G. (2014): “A generalized endogenous grid method for non-smooth and non-concave problems,” *Review of Economic Dynamics*, 17 (2), 329–344.
- HUBBARD, R. G., J. SKINNER, AND S. P. ZELDES (1995): “Precautionary saving and social insurance,” *Journal of Political Economy*, 103 (2), 360–399.
- 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.
- KRUSELL, P., AND A. A. SMITH, JR. (1998): “Income and wealth heterogeneity in the macroeconomy,” *Journal of Political Economy*, 106 (5), 867–896.
- LI, W., H. LIU, F. YANG, AND R. YAO (2016): “Housing over time and over the life cycle: A structural estimation,” *International Economic Review*, 57 (4), 1237–1260.
- PRESS, W. H., S. A. TEUKOLSKY, W. T. VETTERLING, AND B. P. FLANNERY (2007): *Numerical Recipes: The Art of Scientific Computing*: Cambridge University Press.
- SARGENT, T. J., AND J. STACHURSKI (2026): “Dynamic Programming Volume II: General States,” Online book, <https://quantecon.github.io/book-dp2/>.
- SCHOLZ, J. K., A. SESHADRI, AND S. KHITATRAKUN (2006): “Are Americans saving ‘optimally’ for retirement?” *Journal of Political Economy*, 114 (4), 607–643.
- SKIBA, A. K. (1978): “Optimal growth with a convex-concave production function,” *Econometrica*, 46 (3), 527–539.