A Fast and Convergent Hybrid of Value Function Iteration and the Endogenous Grid Method for Discrete-Time Income Fluctuation Problems[*]

Yuichiro Waki[‡]

Aoyama Gakuin University and the University of Queensland

August 11, 2025

Abstract

This paper proposes a fast and convergent solution method for discrete-time income fluctuation problems with a single asset, combining the strengths of value function iteration (VFI) and the endogenous grid method (EGM). I apply EGM logic to the VFI algorithm using piecewise-linear interpolation to obtain the entire policy function analytically at each VFI step: given a concave, piecewise-linear next-period value function, the policy functions for consumption and savings are also shown to be piecewise linear, with kinks located on an endogenous grid. As a result, the method is fast — eliminating the need for numerical optimization; accurate — avoiding policy interpolation errors; and stable — due to guaranteed convergence. Moreover, policy function iteration can be easily applied to accelerate

[*]This paper was previously circulated as "A Globally Convergent Endogenous Grid Method for Discrete-Time Income Fluctuation Problems" and"Fitted Value Iteration with Piecewise Linear Interpolation for Efficient Solution of Income Fluctuation Problems and Equilibrium Computation in Heterogeneous-Agent Models." I am especially grateful to Masao Fukui and Akira Toda for their insightful comments. I also thank Toni Braun and Karen Kopecky for helpful suggestions, and seminar participants at Meiji University for their comments. I am grateful for financial support from JSPS KAKENHI Grant-in-Aid for Scientific Research (A) Grant Number 22H00058.

[†]Associate Professor, Department of Economics, Aoyama Gakuin University. Email: `ywakiecon@gmail.com`. Address: 4-4-25 Shibuya, Shibuya-ku, Tokyo, Japan.

[‡]Honorary Senior Fellow, the School of Economics, the University of Queensland, Australia.

convergence, as it reduces to solving a sparse linear system of equations — solvable instantly even with a large grid. This makes the convergence speed comparable to the standard EGM and the continuous-time method, while retaining stability. A simple modification substantially improves accuracy by reducing the Euler equation error to the level of the standard EGM. The method also computes the stationary equilibrium in an incomplete-market general equilibrium model easily and nearly as fast as its continuous-time counterpart.

Keywords: Value function iteration; Piecewise linear interpolation; Endogenous grid method; Income fluctuation problem; Heterogeneous-agents models; Numerical solution

1 Introduction

In many macroeconomic models, households are assumed to solve income fluctuation (or consumption-savings) problems. These problems generally do not permit closed-form solutions, and need to be solved numerically. When structurally estimating model parameters or when searching for optimal policy within a parametric class or when solving for general equilibria in these models, one must solve them repeatedly for different values of parameters or prices. Hence, it is important to have a solution method with both sufficient speed and guaranteed convergence.

Existing solution methods for discrete-time income fluctuation problems achieve either speed or guaranteed convergence, but not both. The standard endogenous grid method (EGM) proposed by Carroll (2006) is known to be fast, but its convergence is not guaranteed and may therefore fail, depending on the initial condition. In contrast, the value function iteration (VFI) method is guaranteed to converge due to a contractive property, but is known to be slow. VFI may also suffer from non-convergence when errors that arise from optimization and from value function approximation do not dissipate during iteration.

In this paper, I propose a new method that is applicable to income fluctuation problems with a single asset by applying the idea of EGM to VFI with piecewise linear (affine) interpolation. The proposed method achieves speed and accuracy that are comparable to those of EGM, while retaining guaranteed convergence as VFI. Its speed is comparable to the continuous-time method, which is also known to be fast. Specifically, I demonstrate the following: (1) conditional on a concave piecewise linear value function estimate, the policy function at each optimization step can be derived analytically; (2) VFI is globally convergent because there is no approximation or interpolation error; (3) VFI is fast, and policy function iteration (Howard's improvement algorithm) further accelerates convergence and is easily implementable; (4) a simple modification of the proposed algorithm substantially improves accuracy; and (5) a stationary equilibrium in a heterogeneous-agents incomplete-market general equilibrium model can be computed at a speed comparable to that of continuous-time method.

The main result of this paper is that when the next-period value function is a concave piecewise linear function, the policy function can be obtained analytically at each step of fitted value iteration.

The algorithm extends the idea of the standard EGM. For the interior of a linear segment of the next-period value function, one can identify an interval of current cash-on-hand for which the decision-maker finds it optimal to be in that segment. Within such an interval, consumption is constant because the Euler equation implies that the marginal utility of current consumption must equal the slope of the value function, which is constant. In contrast, savings change one for one with current cash-on-hand. Similarly, for each kink point of the next-period value function, there exists an interval of current cash-on-hand over which the decision-maker finds it optimal to be at the kink in the next period. In such intervals, savings are constant, whereas consumption changes one for one with current cash-on-hand.

These subintervals partition the domain of current cash-on-hand, and their endpoints are determined endogenously, as in the EGM. Consequently, the policy functions for consumption and savings are piecewise linear, with kinks on an endogenous grid. Their values on this grid are computed analytically from the Euler equation and the value function. Once values on the endogenous grid are known, the entire functions are fully characterized, and piecewise linear interpolation yields exact values at any point, including those on an exogenous grid. This is in contrast to the standard EGM: it computes the policy function on an endogenous grid but interpolation or extrapolation introduces approximation error off the grid. Moreover, I show that there is no need for numerical interpolation when evaluating the analytical policy functions for consumption and savings.

Obtaining an analytical policy function is useful for four reasons. First, the numerical Bellman operator is a contraction mapping and VFI is guaranteed to converge globally. Because the policy function is obtained analytically at each step, there is no optimization error arising from interpolation or approximation, which would otherwise cause instability in VFI. This guaranteed convergence is an advantage of the present paper's method over the standard EGM.

Second, evaluating the policy and value functions at a large number of points is computationally inexpensive — far less costly than solving optimization problems repeatedly at many points in the state space. This is beneficial, especially because piecewise linear interpolation often requires a large number of grid points to achieve a given accuracy.

Third, the algorithm is fast. The availability of an analytical policy function accelerates VFI, and both standard and modified policy function iteration (PFI) can be used to further accelerate convergence. Even without PFI, the convergence speed of the proposed method is comparable to the standard EGM. With PFI, it is comparable to the continuous-time method. PFI algorithms reduce either to solving a sparse linear system of equations or performing repeated sparse matrix multiplications for a finite number of times — both of which can be executed instantly even with a large grid size due to sparsity. This is a consequence of using a piecewise linear value function: evaluating a piecewise linear function at an arbitrary point involves finding the two adjacent grid points and taking a weighted average. To compute the continuation utility from a given next-period asset level, one constructs a vector of weights (which can be interpreted as transition probabilities) and multiplies it with the value vector.

Collecting these weights across grid points forms a sparse transition probability matrix, facilitating PFI. One caveat is that PFI algorithms do not preserve concavity, and thus concavification of the resulting value function is necessary before the next optimization step. With concavification, PFI is shown to converge.

Fourth, a simple modification of the algorithm substantially improves accuracy. The proposed method uses piecewise linear interpolation for the value function, which is known to exhibit large errors where curvature is large. The Euler equation error can be significantly reduced by the following modification: compute the policy function using the PFI algorithm, and then apply a single step of the standard EGM or two steps, if necessary. This modification reduces the Euler equation error to a level comparable to that of the standard EGM.

Fifth, the method is also useful for solving heterogeneous-agents incomplete-market general equilibrium models. The sparse transition probability matrix used in PFI allows for instantaneous computation of the stationary asset distribution for given prices. Combining this with an outer loop for prices to clear markets, fast computation of a stationary equilibrium is possible. In a simple Huggett (1993) economy, the time needed to find an equilibrium is comparable to that of a continuous-time method (Achdou et al., 2021). Although the dynamics under aggregate shocks are beyond this paper's scope, the framework is compatible with the sequence-space Jacobian method (Auclert et al., 2021), and thus it is applicable to the present paper's setting.

The third and fifth results build on those of Rendahl (2022), which show that approximating the savings policy function with a sparse transition probability matrix is key to accelerating computation of stationary equilibria in discrete-time heterogeneous-agent models. Rendahl's approximation is based on Young's (2010) "lottery" or "histogram" method and is agnostic as to how the policy function is obtained.[1] While he notes that Young's method is exact when the value function is piecewise linear, error bounds for more general value functions are not established. This paper complements Rendahl's in three respects. First, it derives an analytical policy function for piecewise linear value

[1]In Rendahl's code, the optimization step in the Aiyagari (1994) model is solved by finding the root of the Euler equation at each asset level in the exogenous grid.

functions, enabling fast computation of the policy. Second, because it uses piecewise linear value functions, PFI using a sparse transition probability matrix is exact and thus free from approximation error. Third, it achieves stability through concavification of the value function.

There are some limitations in the present paper's algorithm. First, because it relies on the first-order condition to solve the optimization problem, its validity is restricted to concave problems. For non-concave problems, the Euler equation is generally not sufficient, and the algorithm may not correctly solve the optimization step. Second, the analytical policy function is only available when the endogenous state variable is one-dimensional. Identifying linear segments of the value function is straightforward in one dimension but difficult in higher dimensions. For concave problems with multidimensional state variables (e.g., two-asset income fluctuation problems), the lower approximation method of Fukushima and Waki (2013) remains useful. That method computes a probability transition matrix over grid points, as in the present paper, reducing PFI to sparse linear algebra. However, solving the optimization problems numerically to obtain the policy function becomes much more costly when grid sizes are large.

This paper is organized as follows. Section 2 sets up a two-period consumption and savings model and proves a key proposition regarding properties of the policy function. Section 3 describes the VFI algorithm using a deterministic neoclassical growth model. Section 4 describes the PFI algorithm, and Section 5 compares its performance with the standard EGM and with a continuous-time approach. In Section 6, I compare the Euler equation error from the proposed method and the standard EGM, and introduce a simple modification that reduces the error. Section 7 describes the extension to stochastic environments with Markovian shocks and discusses how the method can be applied to a heterogeneous-agent general equilibrium model. Finally, Section 8 extends the method to environments with endogenous labor supply, and Section 9 concludes.

2 Preliminary

Consider a piecewise linear function on a compact interval $\mathcal{K} = [k_{min}, k_{max}] \subset \mathbb{R}_+$. It is characterized by a pair consisting of a finite, increasing grid $k_{grid} = \{k_1, k_2, \ldots, k_{I_k}\}$ with $k_1 = k_{min}$ and $k_{I_k} = k_{max}$, and the function's values on the grid, $\{v_1, v_2, \ldots, v_{I_k}\}$. Henceforth, I will take as exogenously given the grid k_{grid}, and the function can be summarized by a finite vector, $\mathbf{v} = (v_1, v_2, \ldots, v_{I_k})^\top$, which is a column vector. Let $\hat{V}(.; \mathbf{v})$ denote the piecewise linear function on $\mathcal{K}$ given $\mathbf{v}$ such that

$$\hat{V}(k; \mathbf{v}) = \frac{k_{i+1} - k}{k_{i+1} - k_i} v_i + \frac{k - k_i}{k_{i+1} - k_i} v_{i+1} \quad \text{if } k \in [k_i, k_{i+1}] \text{ for some } i.$$

Below I focus on a concave piecewise linear function, and thus assume that $(v_{i+1} - v_i)/(k_{i+1} - k_i)$ is non-increasing in i.

Consider the following consumption-saving problem:

$$m(y; \mathbf{v}) := \max_{c,k'} \{u(c) + \beta \hat{V}(k'; \mathbf{v})\}$$

subject to the budget constraint

$$c + k' \leq y,$$

and the constraint on next-period capital, $k' \in \mathcal{K}$. Here, y denotes the cash-on-hand and k' the saving. The parameter $\beta \in (0, 1)$ is the preference discount factor and $\hat{V}$ represents the value from savings. The function u is a strictly increasing, strictly concave C^1 function and is assumed to satisfy the condition that u' has an analytical inverse, e.g., $u(c) = \ln c$ or $u(c) = c^{1-\sigma}/(1 - \sigma)$ for $\sigma > 0$ with $\sigma \neq 1$.

The above problem, which I denote by $\mathcal{P}(y; \mathbf{v})$, is indexed with cash-on-hand, y, and a value vector, $\mathbf{v}$. Let $\mathcal{Y}$ be the set of y's such that the constraint set is non-empty. If the domain of u is $(0, \infty)$, then $\mathcal{Y} = \{y \in \mathbb{R}_+ | y > k_{min}\}$. If u is well-defined at $c = 0$, then $\mathcal{Y} = \{y \in \mathbb{R}_+ | y \geq k_{min}\}$.

Of interest are a solution and the maximized value, $m(y; \mathbf{v})$, to the problem $\mathcal{P}(y; \mathbf{v})$ for $y \in \mathcal{Y}$. Its solution is the *policy function* for consumption and saving, which I denote by

$g_c(y;\mathbf{v})$ and $g_{k'}(y;\mathbf{v})$, respectively. Although the problem features kinks in the objective function and bound constraints, its solution and value are fully characterized by the following proposition.

Proposition 1 *Suppose* $\mathbf{v}$ *is such that* $\hat{V}(.;\mathbf{v})$ *is a concave function. Then, (1) the set* $\mathcal{Y}$ *can be partitioned into* $(2I_k - 1)$ *subintervals, and* g_c *and* $g_{k'}$ *are piecewise linear functions that are linear on each subinterval; and (2) for each* y*, there exists a probability vector* $\mathbf{p}(y;\mathbf{v}) \in \mathbb{R}_+^{I_k}$ *such that at most two elements in* $\mathbf{p}(y;\mathbf{v})$ *are strictly positive, and that the maximized value,* $m(y;\mathbf{v})$*, satisfies*

$$m(y;\mathbf{v}) = u(g_c(y;\mathbf{v})) + \beta \mathbf{p}(y;\mathbf{v})^\top \mathbf{v}.$$

In the following proof, instead of characterizing the policy function for a given y, I derive a condition under which the optimal saving equals a given level in the whole interval $\mathcal{K}$. This idea is derived from that of the endogenous grid method (EGM) in Carroll (2006), but there is a key difference: EGM only considers the values of k' on a finite, exogenous grid, whereas I consider all values in $\mathcal{K}$. This treatment is possible because of the piecewise linearity of the function $\hat{V}$, as I describe below, and it results in a global characterization of the policy function.

Proof.

[Part 1] Suppose first that $\hat{V}(,;\mathbf{v})$ is strictly increasing so that $(v_{i+1}-v_i)/(k_{i+1}-k_i)$ is strictly positive.

What are the levels of current cash-on-hand such that the planner finds it optimal to choose $k' \in (k_i, k_{i+1})$ for some $i = 1, \ldots, I_k - 1$? Because k' is neither at a kink nor at a bound, and because v is concave, the first order condition should be satisfied:

$$u'(c) = \lambda_i := \beta \frac{v_{i+1} - v_i}{k_{i+1} - k_i} > 0.$$

The right-most expression is strictly positive, because v_i is strictly increasing in i. Hence, the optimal consumption is given by $c = c_i^* := (u')^{-1}(\lambda_i)$. Therefore, if the current cash-on-hand, y is in the open interval,

$$\mathcal{I}_i := \left(c_i^* + k_i, c_i^* + k_{i+1}\right),$$

the planner's optimal consumption and saving choices are given by $(c, k') = (c_i^*, y - c_i^*)$. In sum, the policy function satisfies

$$g_c(y; \mathbf{v}) = c_i^*, g_{k'}(y; \mathbf{v}) = y - c_i^*$$

for all $y \in \mathcal{I}_i$ and for $i = 1, \ldots, I_k - 1$.

When is it optimal to choose a grid point, $k' = k_i$? First consider the boundary, $k' = k_1$ and $k' = k_{I_k}$. For $k' = k_1$ to be optimal, the first order condition requires

$$u'(c) \geq \lambda_1 := \beta \frac{v_2 - v_1}{k_2 - k_1}.$$

Hence, $c \leq c_1^*$. Therefore, if the current cash-on-hand, y, is in the interval

$$\mathcal{J}_1 := [k_1, c_1^* + k_1] \cap \mathcal{Y},$$

the planner's optimal choice for consumption and saving is given by $c = y - k_1$ and $k' = k_1$. For $k' = k_N$ to be optimal, the first order condition is

$$u'(c) \leq \lambda_{I_k - 1} := \beta \frac{v_{I_k} - v_{I_k - 1}}{k_{I_k} - k_{I_k - 1}},$$

implying $c \geq c_{I_k - 1}^*$. Therefore, if the current cash-on-hand is in the interval

$$\mathcal{J}_{I_k} := [c_{I_k - 1}^* + k_{I_k}, \infty)$$

the planner finds it optimal to choose $c = y - k_{I_k}$ and $k' = k_{I_k}$.

For other kinks, the first-order condition requires

$$\lambda_i \leq u'(c) \leq \lambda_{i-1}$$

for setting $k' = k_i$, $i = 2, 3, \ldots, I_k - 1$, to be optimal. It is when the current cash-on-hand is in the interval

$$\mathcal{J}_i := [c^*_{i-1} + k_i, c^*_i + k_i],$$

and the optimal choice of the planner satisfies $c = y - k_i$ and $k' = k_i$.

In sum, the policy function satisfies

$$g_c(y; \mathbf{v}) = y - k_i, g_{k'}(y; \mathbf{v}) = k_i$$

for all $y \in \mathcal{J}_i$ and for $i = 1, \ldots, I_k$.

Note that I have partitioned $\mathcal{Y}$ into $2I_k - 1$ subintervals as $\mathcal{Y} = \mathcal{J}_1 \cup \mathcal{I}_1 \cup \mathcal{J}_2 \cup \cdots \cup \mathcal{I}_{I_k - 1} \cup \mathcal{J}_{I_k}$, and shown that the policy function is linear in each subinterval.

Suppose now that $\hat{V}(, ; \mathbf{v})$ is not strictly increasing. Then, due to concavity, there exists $\bar{i} \geq 1$ such that $(v_{i+1} - v_i)/(k_{i+1} - k_i) \leq 0$ for all $i \geq \bar{i}$ and that $(v_{i+1} - v_i)/(k_{i+1} - k_i) > 0$ for all $i < \bar{i}$. I.e., $\hat{V}(k_{\bar{i}}; \mathbf{v}) \geq \hat{V}(k; \mathbf{v})$ for all $k \geq k_{\bar{i}}$. Because u is strictly increasing, it is never optimal to choose k' that is greater than $k_{\bar{i}}$. Hence, $k_{\bar{i}}$ can be treated effectively as an upper bound of the household's saving choice. I.e., the above proof goes through once one defines $\mathcal{J}_{\bar{i}} = [c^*_{\bar{i}-1} + k_{\bar{i}}, \infty)$ and $\mathcal{I}_i = \mathcal{J}_{i+1} = \emptyset$ for all $i > \bar{i}$.

[Part 2] For $y \in \mathcal{J}_i$, let $\mathbf{p}(y; \mathbf{v})$ be a probability vector such that its i-th element is one and other elements are zero. Then,

$$\hat{V}(g_{k'}(y; \mathbf{v}); \mathbf{v}) = \hat{V}(k_i; \mathbf{v}) = v_i = \mathbf{p}(y; \mathbf{v})^\top \mathbf{v}. \tag{1}$$

For $y \in \mathcal{I}_i$, let $\mathbf{p}(y; \mathbf{v})$ be a probability vector such that its i-th and $(i+1)$-th elements are $(k_{i+1} - g_{k'}(y; \mathbf{v}))/(k_{i+1} - k_i)$ and $(g_{k'}(y; \mathbf{v}) - k_i)/(k_{i+1} - k_i)$, respectively, and other elements are zero. Then,

$$\hat{V}(g_{k'}(y;\mathbf{v});\mathbf{v}) = \frac{k_{i+1} - g_{k'}(y;\mathbf{v})}{k_{i+1} - k_i} v_i + \frac{g_{k'}(y;\mathbf{v}) - k_i}{k_{i+1} - k_i} v_{i+1} = \mathbf{p}(y;\mathbf{v})^\top \mathbf{v}, \tag{2}$$

and Part 2 follows. ■

The policy functions for consumption and savings are depicted in Figure 2. They are piecewise linear. The consumption (savings) policy function is constant on $\mathcal{I}_i$ ($\mathcal{J}_i$) intervals but has a slope of one on $\mathcal{J}_i$ ($\mathcal{I}_i$, respectively) intervals.

A big advantage is that these policy functions are inexpensive to evaluate at any point $y \in \mathcal{Y}$, because evaluation of these functions do not require numerical interpolation, which is computationally costly. Suppose one figures out that y is in $\mathcal{I}_i$. It immediately follows that the consumption policy function at y equals c_i^*, defined in the above proof of Proposition 1. Because the cash-on-hand is y, the savings policy function must be equal to the remaining resources, $y - c_i^*$. Suppose instead that y is found to be in $\mathcal{J}_i$. Then the savings policy function must be equal to k_i, and it follows that the consumption policy function must be equal to the remaining resources, $y - k_i$. Both the continuation utility, $\hat{V}(g_{k'}(y;\mathbf{v});\mathbf{v})$, and the probability vector, $\mathbf{p}(y;\mathbf{v})$, can be computed at any y, using equation (1) if $y \in J_i$ for some i and equation (2) if $y \in I_i$ for some i. The most computationally expensive part is to find, for each $y \in y_{grid}$, a subinterval that contains y.

Proposition 1 leads to the following algorithm to evaluate the policy and the value at multiple points $y_{grid} = \{y_1, y_2, \ldots, y_{I_y}\}$ in $\mathcal{Y}$.

Algorithm 1 (Extended EGM algorithm) *1. Compute endogenous grid points, i.e., the endpoints of all subintervals, and the policy function g_c and $g_{k'}$ on them.*

2. For each $y \in y_{grid}$,

(a) Find a subinterval that contains y, and evaluate g_c, $g_{k'}$, $\mathbf{p}$, and $\hat{V}(g_{k'}(.))$ at y.

(b) Compute $m(y)$ using g_c and either $\mathbf{p}$ or $\hat{V}(g_{k'}(.))$.

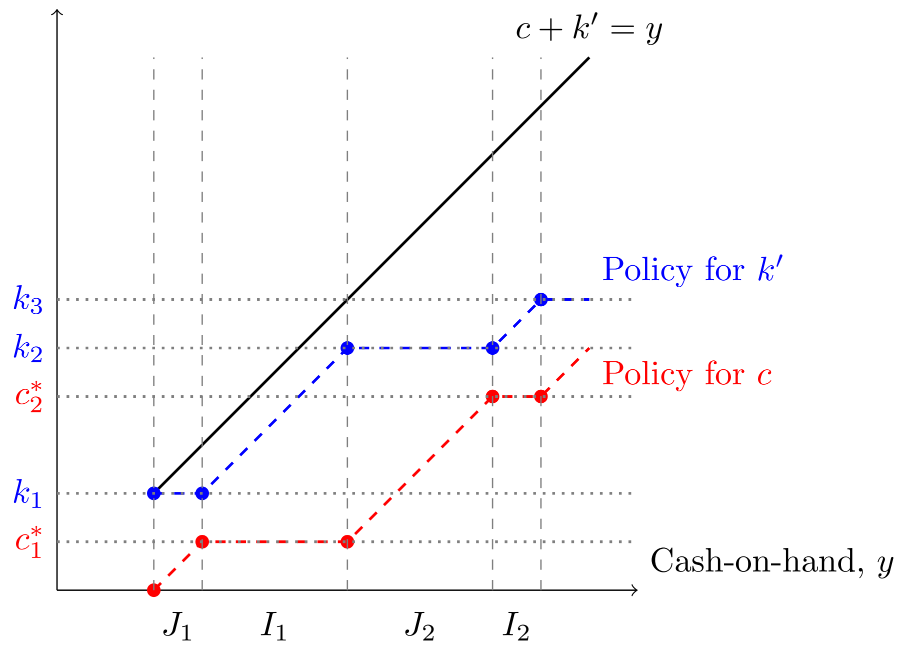


Figure 1: Policy functions for consumption and savings

To simplify the notation, I have omitted above the dependence of functions on $\mathbf{v}$. A detailed explanation about how to implement Step 1 is given in Appendix A.

Given a concave piecewise linear value function, the above algorithm solves the problem exactly — there is no error associated with approximation, interpolation, or extrapolation. As in the standard endogenous grid method, it is assumed that the analytical inverse of the marginal utility function is available, and obtaining $\{c_i^*\}$ — and hence the endogenous grid points — is computationally inexpensive. The algorithm does not need numerical interpolation when evaluating the policy functions and the maximized value on y_{grid}.

The most costly part is Step 2(a), where one needs to find, for each $y \in y_{grid}$, a subinterval that contains y. This can be done using off-the-shelf methods (e.g., Matlab's `discretize`), but one can improve efficiency by exploiting the monotonicity of y_{grid}: if one has found that y_i is in between two endogenous grid points, then y_{i+1}, which is greater than y_i, cannot lie between lower endogenous grid points. Hence, one does not need to examine all subintervals when searching.

Instead of repeating Steps 2(a) and 2(b) for different y's, one may repeat Steps

2(a) for different y's, and then obtain a vector of values, $\mathbf{m} = (m(y_1), \ldots, m(y_{I_y}))^\top$, by computing

$$\mathbf{m} = \mathbf{u}(\mathbf{v}) + \beta \mathbf{P}(\mathbf{v})\mathbf{v}$$

where

$$\mathbf{u}(\mathbf{v}) = \begin{bmatrix} u(g_c(y_1; \mathbf{v})) \\ \vdots \\ u(g_c(y_{I_y}; \mathbf{v})) \end{bmatrix} \quad \text{and} \quad \mathbf{P}(\mathbf{v}) = \begin{bmatrix} \mathbf{p}(y_1; \mathbf{v})^\top \\ \mathbf{p}(y_2; \mathbf{v})^\top \\ \vdots \\ \mathbf{p}(y_{I_y}; \mathbf{v})^\top \end{bmatrix}. \tag{3}$$

The matrix $\mathbf{P}(\mathbf{v})$ is sparse, because each row has either one or two non-zero elements. This matrix-based evaluation is especially useful in the context of policy function iteration. I will come back to this issue in Section 4. For the fitted value function iteration algorithm in the next section, repeating Steps 2(a) and 2(b) for different y's is more efficient.

3 Fitted value function iteration (VFI) algorithm

Fitted value iteration algorithms are widely used to numerically solve continuous-state dynamic programming problems. These algorithms discretize the state space, update a value function estimate on a finite set of grid points at each step, and use these data to construct an approximate function over the entire state space via interpolation or approximation methods. Various such methods have been proposed and their pros and cons have been extensively studied in the literature (see, e.g., Judd (1998) and Cai and Judd (2014)).

Proposition 1 is particularly useful in fitted value iteration in which (1) piecewise linear interpolation is used for the value function and (2) the true value function is known to be a concave function of a one-dimensional endogenous state variable.

VFI with piecewise linear interpolation possesses several desirable properties. First, piecewise linear interpolation preserves monotonicity and concavity, making it a useful approximation technique when the true value function is known to be both monotonic and concave. Second, it is a non-expansive approximation method; hence, fitted value iteration with piecewise linear interpolation — provided that each optimization step is solved accurately — is guaranteed to converge to a unique fixed point because the numerical Bellman operator is a contraction mapping (Stachurski, 2008).[2] This stability contrasts with that of polynomial approximation methods, which may fail to converge.

However, VFI with piecewise linear interpolation has not been widely recommended in practice, it is believed to be computationally costly: the kinks introduced by linear interpolation preclude the use of optimization methods that require differentiable or smooth objective functions.[3] This observation has led researchers to search for approximation methods that yield smooth approximations of the value function (see, e.g., Judd and Solnick (1994) and Cai and Judd (2012)).

Proposition 1 overcomes this bottleneck by providing an analytical solution and eliminating the need for numerical optimization to solve optimization problems, under the condition that the piecewise-linear value function is concave, while retaining the aforementioned desirable properties. This condition is satisfied in many economic applications where the true value function is concave and the Bellman operator preserves concavity. Examples include one-sector neoclassical growth models and typical income fluctuation problems with a single asset.

In these models, VFI using a piecewise linear interpolation solves a problem in Proposition 1, and Proposition 1 provides a complete characterization of the policy function at each optimization step. Hence, there is no error from approximation, interpolation, or extrapolation. The numerical Bellman operator is a contraction mapping, and VFI is guaranteed to converge to its fixed point, which provides a good approximation of the

[2]The convergence result is also shown elsewhere in the literature. See, e.g., Santos and Vigo-Aguiar (1998) and Judd and Solnick (1994).

[3]For example, Judd (1998) notes: "The problem with linear interpolation is that it makes the maximization step less efficient. The kinks in a linear interpolant will generally produce kinks in the objective of the maximization step, forcing us to use slower optimization algorithms. $\cdots$ Using linear interpolation is a costly way of preserving shape" (Ch.12, p.439). More recently, Cai and Judd (2014) write: "If one uses piecewise linear approximations, then one needs to use many nodes to construct a good approximation, and the optimization problems in Algorithm 1 [a fitted value iteration algorithm] have nondifferentiable objective functions, a feature that rules out the use of fast Newton-type solvers. The alternatives, such as bisection, will be much slower. Also, the piecewise linear approximation approach only works [as a shape-preserving method] for one-dimensional problems."

true value function as long as the grid is sufficiently fine.

Note that piecewise linearity of the analytical solution makes it inexpensive to evaluate both policy and value functions on a very fine grid. This feature is important for two reasons. First, it accelerates VFI. Second, it is inexpensive to improve accuracy, because piecewise linear interpolation often requires a fine grid to obtain an accurate solution. As shown in Judd and Solnick (1994), although piecewise linear interpolation is more efficient than simple discretization, it requires a much larger grid size than Schumaker's (1983) shape-preserving interpolation to achieve the same level of accuracy.

Compared to nonlinear interpolation methods, piecewise linear interpolation offers two additional advantages, discussed in the next two sections. First, policy function iteration algorithms can accelerate convergence. The reason is that, with piecewise linear value function, evaluation of the continuation utility amounts to a sparse matrix multiplication — multiplying the value vector by a sparse probability transition matrix, $\mathbf{P}(\mathbf{v})$, from the left. In contrast, it is more expensive to repeatedly evaluate interpolants for other interpolation methods. Second, the same transition matrix enables fast computation of a stationary equilibrium in heterogeneous-agents macroeconomic models.

3.1 Deterministic Ramsey model

Consider a standard, deterministic neoclassical growth model in which the planner's problem that is, in a recursive form, given as follows. For all $k \in \mathcal{K} := [k_{min}, k_{max}]$,

$$v(k) = \max_{c,k'}\{u(c) + \beta v(k')\}$$

subject to the resource constraint,

$$c + k' \leq F(k),$$

and the constraint on the next period capital, $k' \in \mathcal{K}$.

The preference discount factor, β, is positive and strictly less than one. The function F is assumed to be a strictly concave function satisfying $F(0) = 0$, $F'(k) > 0$,

$\lim_{k\downarrow 0} F'(k) = \infty$, and $\lim_{k\downarrow 0} F'(k) = 1 - \delta$. Later in the numerical exercise, I use a Cobb-Douglas production function, $F(k) = Ak^{\alpha} + (1-\delta)k$, with $0 < \alpha < 1$ and $0 < \delta < 1$.

I now make some assumptions so that the constraint set is non-empty for all $k \in \mathcal{K}$. First, I assume that the lower bound for capital is strictly positive, $k_{min} > 0$. Second, the lower bound is not too large in that $F(k_{min}) - k_{min} > 0$. These assumptions guarantees that, for any $k \in \mathcal{K}$, a policy that sets $c = F(k) - k_{min}$ and $k' = k_{min}$ is feasible.

The true value function, v, is then bounded below by a concave function. To construct a lower bound for v, consider a sub-optimal policy that always saves k_{min}. Define the function $v_{LB}(k)$ as the value from following this policy. Then $v \geq v_{LB} \geq v_{LB}(k_{min}) > -\infty$, and v_{LB} is concave. If setting $c = F(k) - k$ is feasible for all $k \in \mathcal{K}$, then one can instead use $v_{LB}(k) = u(F(k) - k)/(1-\beta)$. In the numerical exercises, I use this v_{LB} as the initial condition for VFI and PFI algorithms.

The true value function is bounded above by a constant. Define $\overline{v} \in \mathbb{R}$ by

$$\overline{v} = \frac{u(F(k_{max}) - k_{min})}{1-\beta} < \infty.$$

Then, because the resource constraint requires $c \leq F(k) - k' \leq F(k_{max}) - k_{min}$ for any k, we have $v(k) \leq \overline{v}$.

Let B be the space of bounded, continuous functions on $\mathcal{K}$, and endow it with the sup norm, $||\cdot||$. Let $\mathbb{T} : B \to B$ be a Bellman operator such that, for any function $w \in B$ and for any $k \in \mathcal{K}$,

$$\mathbb{T}w(k) = \max_{c,k'} u(c) + \beta w(k_i)$$

subject to $c + k' \leq F(k)$, $k' \in \mathcal{K}$. Then $\mathbb{T}$ is a contraction mapping on $(B, ||\cdot||)$, and thus has a unique fixed point in B, and it coincides with the value function, v. Moreover, by setting $v_0 := v_{LB}$ and recursively define $v_{n+1} = \mathbb{T}v_n$, then $\{v_n\}$ monotonically converges to v from below.

It is straightforward to show that $\mathbb{T}$ is monotonicity- and concavity-preserving, and that v is monotone and concave.

3.2 Illustration of the VFI algorithm

The true Bellman operator cannot be exactly implemented on a computer because it requires solving the maximization problem for a continuum of values for k. Fitted value iteration methods solve the maximization problem only on a finite grid and either approximate or interpolate the resulting values over the whole domain to use it in the next optimization step.

Note that, once the cash-on-hand y is defined as $y = F(k)$, the structure of the problem is identical to that in Section 2 if the interpolant is a piecewise linear, concave function. This observation leads to the following VFI algorithm:

Algorithm 2 (Fitted value iteration algorithm) *1. Initialization:*

(a) Fix the value tolerance level, $tol_v > 0$, and the capital grid, $k_{grid} = \{k_1, k_2, \ldots, k_{I_k}\}$ with $k_1 = k_{min}$ and $k_{I_k} = k_{max}$.

(b) Pre-compute $y_{grid} = \{y_1, y_2, \ldots, y_{I_k}\}$, with $y_i = F(k_i)$ for $i = 1, \ldots, I_k$.

(c) Set an initial value vector, $\mathbf{v}_0$, using a concave value function from a feasible policy, e.g., $v_{0,i} = v_{LB}(k_i)$ for $i = 1, \ldots, I_k$.

2. VFI: for $n = 0, \ldots,$ taking $\mathbf{v_n}$ as given,

(a) Compute the policy function on y_{grid}, by solving the $\mathcal{P}(y; \mathbf{v}_n)$-problem using the algorithm 1 for all $y \in y_{grid}$.

(b) Update the value function as $v_{n+1,i} = u(g_c(y_i; \mathbf{v}_n)) + \beta \mathbf{p}(y_i; \mathbf{v}_n)^\top \mathbf{v}_n$ for all $i = 1, \ldots, I_k$.

(c) Stop if $||\mathbf{v}_{n+1} - \mathbf{v}_n|| < tol_v$. Go back to 2.1 otherwise.

Effectively the above algorithm updates the value vector as

$$\mathbf{v}_{n+1} = \mathbf{u}(\mathbf{v}_n) + \beta \mathbf{P}(\mathbf{v}_n)\mathbf{v}_n,$$

where $\mathbf{u}(\mathbf{v}_n)$ and $\mathbf{P}(\mathbf{v}_n)$ are defined in equation (3), until the convergence criterion is

Table 1: Computational time: VFI

	$I_k = 10,000$	$I_k = 1,000$
Total time (seconds)	3.999×10^{-2}	5.790×10^{-3}
Number of steps	73	72
Time per iteration step (seconds)	5.478×10^{-4}	8.041×10^{-5}

satisfied. The algorithm produces a monotone sequence, i.e., $\mathbf{v}_{n+1} \geq \mathbf{v}_n$ for all n. Because the sequence is bounded above by $\overline{v}$, it converges.

The above algorithm is standard except that the analytical solution is used at the optimization step. Judd and Solnick's (1994) Theorems 8 and 9 show that the numerical Bellman operator preserves monotonicity and concavity.[4] Hence, starting from a concave initial value function, $\hat{V}(.;\mathbf{v}_n)$ is strictly increasing and concave for all n, justifying the use of Proposition 1 at the optimization step. Judd and Solnick's (1994) Theorem 4 establishes that it is a contraction mapping, and thus the above algorithm is guaranteed to converge. Error bounds established in Judd and Solnick (1994) and Stachurski (2008) are also applicable.

Now let us evaluate the algorithm's performance in terms of computational time. As a baseline, I use the following parameterization: $\beta = 1/1.05$, $\alpha = 0.3$, $\delta = 0.05$, $A = (1/\beta - 1 + \delta)/\alpha$, and $u(c) = -c^{-1}$, i.e., $u(c) = c^{1-\sigma}/(1-\sigma)$ with $\sigma = 2$. The technology parameter A is chosen so that the steady-state capital is unity. I consider a sufficiently wide range for capital and set $k_{min} = 0.001$ and $k_{max} = 2$. Below I report the results for equally-spaced grids with 1,000 points and 10,000 points. I set the convergence criterion for value function to a tolerance of 10^{-6}.

The code is written in Matlab 2024b and is executed on MacBook Air with Apple M3 chip and 24GB memory. For each grid size, I calculate the average convergence time from repeated computation for 1000 times.

When 10,000 grid points are used, it takes on average about 0.04 seconds until convergence. In each computation, value iteration takes 73 steps until convergence, implying

[4] Its concavity-preserving property depends not only on the algorithm but also on the structure of the optimization problem. A sufficient condition is strict concavity of u and F.

that each step takes about 0.55 milliseconds. When 1,000 grid points are used instead, the average computation time is about 5.8 milliseconds. The number of value iteration steps is 72, implying that each step takes 0.08 milliseconds.

Unlike the total computational time, which depends on the initial condition as well as the discount factor, the average computational time per iteration step is a measure that is independent of both. Even with 10,000 grid points, the time per iteration step is only about 0.8 milliseconds; with 1,000 grid points, it reduces to about 0.11 milliseconds.

One advantage the above VFI algorithm has over the standard EGM is that global convergence is guaranteed. The standard EGM is a fixed-point iteration that keeps updating only an estimate of the derivative of the value function on an exogenous grid until convergence. Because it does not need to compute the value function itself, its convergence is fast when it does. Its weakness is that the convergence is not guaranteed and a good initial condition is often necessary to achieve convergence. The standard EGM uses interpolation, and sometimes extrapolation, to evaluate the policy and the value functions on an exogenous grid, which is subject to errors, because the true shapes of these functions are unknown. In the method proposed here, the true policy function is piecewise linear with kinks on endogenous grid points, implying that piecewise linear interpolation is exact. Moreover, numerical interpolation is unnecessary, making the whole algorithm more efficient.[5]

There are some other advantages for the proposed method. First, as I show in the next section, significant speed-up is possible with policy function iteration, or Howard's improvement algorithm. My method obtains a sparse probability transition matrix over a grid, **P**, and the policy function iteration amounts to a large yet sparse linear alge-

[5]Barillas and Fernandez-Villaverde (2007) combine the standard EGM with VFI to demonstrate that the combined method improves speed of the standard VFI. The method in the present paper can be thought of as a refinement of theirs. Their method updates the value function estimate in each iteration step, and the derivative estimate on an exogenous grid is updated using the two-sided average of the slopes of the linearly interpolated value function. Both policy and value functions are first computed on an endogenous grid using the standard EGM, and then interpolated/extrapolated to evaluate on an exogenous grid. Hence, their method solves the optimization problem only approximately on the exogenous grid, and VFI is not, in theory, guaranteed to converge. In contrast, the method proposed in the present paper solves the optimization problem (conditional on the piecewise linear value function) exactly, and convergence is guaranteed. The method proposed here is also more efficient, because there is no need for numerical interpolation. Still, their method generates a good approximation of the exact solution and, hence, is likely to converge in practice.

bra. Second, the same probability transition matrix is useful in computing a stationary distribution in heterogeneous-agent models.

4 Policy function iteration (PFI)

One advantage of my method is that policy function iteration (PFI) algorithms can be easily implemented.[6] Specifically, standard PFI reduces to solving a sparse linear system of equations, and modified PFI to repeated sparse matrix multiplications.

Let $\mathbf{u}_n = \mathbf{u}(\mathbf{v}_n)$ and $\mathbf{P}_n = \mathbf{P}(\mathbf{v}_n)$. Recall that $\mathbf{P}_n$ is a sparse probability matrix because each row of $\mathbf{P}_n$ has either one or two non-zero elements.

Standard PFI computes, at each step n, the value of permanently following the policy function computed in step n. The value on the exogenous grid, $\mathbf{w}_n$, satisfies

$$\mathbf{w}_n = \mathbf{u}_n + \beta \mathbf{P}_n \mathbf{w}_n,$$

i.e., $w_{n,i} = u(g_c(y_i; \mathbf{v}_n)) + \beta \hat{V}(g_{k'}(y_i; \mathbf{v}_n); \mathbf{w}_n)$ for all i. The solution to this equation is obtained by solving a sparse linear system of equations as $\mathbf{w}_n = (\mathbf{I} - \beta \mathbf{P}_n)^{-1} \mathbf{u_n}$.

Modified PFI instead iterates

$$\mathbf{w}_n^{(j+1)} = \mathbf{u}_n + \beta \mathbf{P}_n \mathbf{w}_n^{(j)},$$

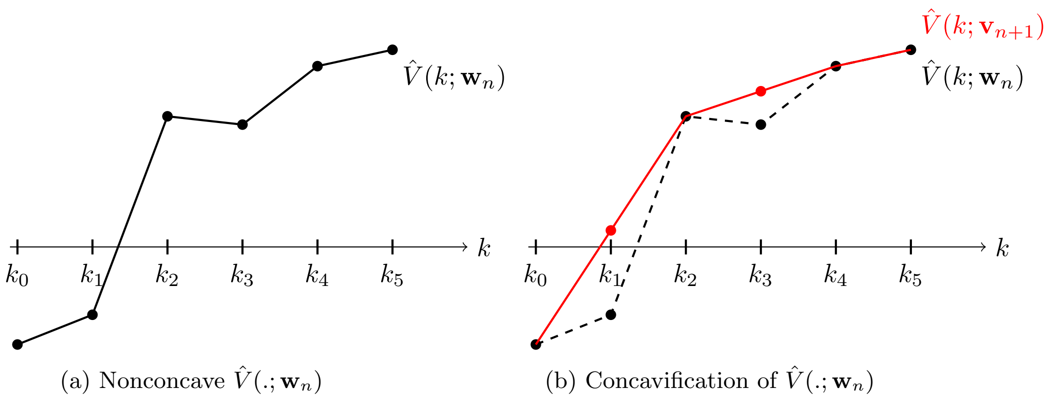
starting from $\mathbf{w}_n^{(0)} = \mathbf{v}_n$, until j reaches a pre-specified integer J. The idea is that $\mathbf{w}_n^{(J+1)}$ is a good approximation of $\mathbf{w}_n$, because $\mathbf{w}_n^{(J+1)} \uparrow \mathbf{w}_n$ as $J \to \infty$. Hence, modified PFI requires repeated sparse matrix multiplications.

The PFI algorithm replaces Step 2(b) in the VFI algorithm with

(b') Compute $\mathbf{w}_n$, and update the value function so that $\hat{V}(., \mathbf{v}_{n+1})$ is the smallest concave function that satisfies $\hat{V}(k; \mathbf{v}_{n+1}) \geq \hat{V}(k; \mathbf{w}_n)$ for all $k \in \mathcal{K}$.

The modified PFI algorithm is identical to the PFI algorithm except it uses $\mathbf{w}_n^{(J+1)}$ in

[6]The concept of policy function iteration originates from Bellman's (1957) approximation in policy space and Howard's (1960) improvement algorithm.



$\hat{V}(k;\mathbf{w}_n)$
k
k_0
k_1
k_2
k_3
k_4
k_5
(a) Nonconcave $\hat{V}(.;\mathbf{w}_n)$
$\hat{V}(k;\mathbf{v}_{n+1})$
$\hat{V}(k;\mathbf{w}_n)$
k
k_0
k_1
k_2
k_3
k_4
k_5
(b) Concavification of $\hat{V}(.;\mathbf{w}_n)$

Figure 2: Illustration of concavification

place of $\mathbf{w}_n$.

Two points warrant elaboration. First, as long as $\hat{V}(k;\mathbf{w}_n)$ is concave for all n, we have $\mathbf{v_{n+1}} = \mathbf{w_n}$, and the above algorithm reduces to the standard PFI algorithm. Its convergence properties can be found in, e.g., Santos and Rust (2004).

Second, in general, the above PFI and modified PFI algorithms do not use directly $\mathbf{w}_n$ or $\mathbf{w}_n^{(J+1)}$ for $\mathbf{v}_{n+1}$, because neither of their piecewise linear interpolants is guaranteed to be concave. Hence, if we were to use $\mathbf{w}_n$ or $\mathbf{w}_n^{(J+1)}$ for $\mathbf{v}_{n+1}$, Proposition 1 is not in general applicable because it requires that the linear interpolant of $\mathbf{v}_{n+1}$ is concave. This problem can be circumvented by choosing $\mathbf{v}_{n+1}$ so that it concavifies these interplants, as in the above algorithm.[7]

How can one find such $\mathbf{v}_{n+1}$? One needs to find an upper concave envelope of the data $\{(k_i, w_{n,i})\}_{i=1}^{I_k}$. Figure 2 illustrates how to do it. If the piecewise linear interpolation of the data is non-concave, the function $\hat{V}(.,\mathbf{v}_{n+1})$ is the smallest, concave piecewise linear function that is greater than the simple piecewise linear interpolation. Therefore, $\mathbf{v}_{n+1}$ can be computed by finding a convex hull of the set of points, $S = \{(k_i, w_{n,i})\}_{i=1}^{I_k} \cup \{(k_1, \min_i w_{n,i} - \varepsilon), (k_{I_k}, \min_i w_{n,i} - \varepsilon)\}$, for some $\varepsilon > 0$, and then traces its upper envelope.[8]

[7]This is effectively the same as using Fukushima and Waki's (2013) lower approximation method in place of the piecewise linear interpolation.

[8]For example, in Matlab, the convhull function returns the set, S^{ext}, of extreme points of conv(S). Then any point in $S^{ext} \setminus \{(k_1, \min_i w_{n,i} - \varepsilon), (k_{I_k}, \min_i w_{n,i} - \varepsilon)\}$ is on the graph of $\hat{V}(.,\mathbf{v}_{n+1})$. Then, by piecewise-linear interpolation, one can obtain $v_{n+1,i} = \hat{V}(k_i;\mathbf{v}_{n+1})$ for any $i = 1, \ldots, I_k$.

Table 2: Computational time comparison: VFI vs. PFI

	$I_k = 10{,}000$			$I_k = 1{,}000$		
	VFI	PFI	MPFI($J = 20$)	VFI	PFI	MPFI
Total time (sec.)	3.999×10^{-2}	1.884×10^{-2}	1.533×10^{-2}	5.790×10^{-3}	1.899×10^{-3}	1.721
Number of steps	73	8	8	72	7	7
Time per step (sec.)	5.478×10^{-4}	2.355×10^{-3}	1.916×10^{-3}	8.041×10^{-5}	2.712×10^{-4}	2.458

Both PFI and MPFI converge due to their monotonicity. For each n, there holds $\mathbf{w}_n \geq \mathbf{w}_n^{(j)} \geq \mathbf{v}_n$ for any $j \geq 0$. The concavification step implies $\mathbf{v}_{n+1} \geq \mathbf{w}_n$, for each n. Hence, $\{\mathbf{v}_n\}$ is a non-decreasing sequence in $\mathbb{R}^{I_k}$. Because the sequence is bounded above by $\overline{v}$, it must converge, and the limit coincides with that of the VFI algorithm.

When 10,000 grid points are used, PFI takes on average 0.019 seconds until convergence. In each computation, PFI takes 8 steps until convergence, implying that each step takes about 2.4 milliseconds. When 1,000 grid points are used instead, the average computation time is 1.9 milliseconds. The number of iteration steps is 7, implying that each step takes 0.27 milliseconds.

PFI achieves a speed gain relative to VFI, by substantially reducing the number of iteration steps. For 10,000 grid points, the total time decreases from 73 to 8. For 1,000 grid points, it decreases from 72 to 7. The computational time per iteration step is longer for PFI than for VFI, because PFI computes the value of following the analytical policy function and concavifies the resulting value function. Yet, the speed gain from a reduction in the number of iteration steps dominates, and the total computational time is cut by 50-70%. Note that the PFI results in Table 2 use a mex function to speed up the concavification step, but PFI remains significantly faster than VFI, even without a compiled implementation.

The modified PFI with $J = 20$ slightly outperforms the standard PFI for the model considered here. MPFI is about 20% faster for 10,000 grid points and about 10% for 1,000 grid points.

Using PFI and piecewise linear interpolation of the value function is not new. For example, Santos and Rust (2004) examine convergence properties of PFI. They establish some theoretical results on the rate of convergence and examine the performance of PFI through numerical experiments. Their numerical results are rather disappointing, firstly because they compute the policy function by numerically solving the optimization problems, and secondly because they do not use a sparse transition matrix to reduce PFI to a sparse linear algebra. The latter is pointed out by Rendahl (2022).

Rendahl (2022) demonstrates that, once the policy function for savings is approximated by a probability transition matrix over a discretized state space, its sparsity can be effectively exploited for PFI. He uses Young's (2010) "lottery" or "histogram" method for such approximation, which he notes is exact in evaluating the continuation value when the value function is piecewise linear. Using such approximation, PFI accelerates computation just as in a continuous-time method in Achdou et al. (2021).

My paper complements Rendahl (2022) in three aspects. First, my paper provides an analytical policy function for a piecewise linear value function, whereas his paper is agnostic about how to compute the policy function. The proposed method significantly accelerates the optimization step. Second, because the proposed method uses piecewise linear value functions, it is guaranteed that there is no approximation error from using a probability transition matrix over a finite grid to conduct PFI. This is advantageous because, while such approximation is possible for a general policy function, error bounds for such cases are not established. Third, the proposed method achieves stability through concavification of the value function.

5 Speed comparison with other methods

In this section I compare the above method with two widely-used methods that are known to be fast: the standard endogenous grid method (EGM) and the continuous-time method.

5.1 Comparison with the standard EGM

The standard EGM has been widely used to solve discrete-time income-fluctuation problems. The main reason is its speed.

The biggest advantage of the method proposed in this paper over the standard EGM is that convergence is guaranteed in the former but not in the latter. The standard EGM may fail to converge, depending on the initial condition. Guaranteed convergence is desirable in many applications. When one structurally estimates some model parameters, one needs to solve the model repeatedly with different parameter values along the way, and it is important to have a solution method that is robust to changes in parameter values. The same is true when one solves general equilibrium models for which one repeatedly updates prices until one finds the set of correct prices.

In terms of speed, the standard EGM has an advantage over the VFI algorithm in this paper. The standard EGM is a fixed point iteration, and during iteration it does not compute the value function, which converges only slowly. Instead, it uses the envelope theorem to update an estimate of the derivative of the value function. Therefore, it is expected that, when the EGM converges, it converges faster than the VFI. However, it is not obvious how fast it is relative to the VFI. It is also not clear whether the standard EGM is faster than the PFI algorithm in this paper.

Comparing their speed is not a straightforward task. Because the standard EGM does not compute the value function, one cannot use the same stopping criterion as the VFI algorithm. To make fair comparison, I compute in the VFI algorithm how much the consumption policy function is updated on the exogenous grid in the final iteration step, $\max_{k \in k_{grid}} |g_{c,n}(k) - g_{c,n-1}(k)|$, and use it as the stopping criterion for the standard EGM. The same initial condition is used for the standard EGM as for the VFI, implying that the initial estimate of the value function derivative is given by

$$Dv_{0,i} = \frac{u'\left(F(k_i) - k_i\right)\left(F'(k_i) - 1\right)}{1 - \beta}$$

Table 3: Computational time comparison: VFI, PFI vs. EGM

	EGM	VFI	PFI
Total time (sec.)	3.127×10^{-2}	3.999×10^{-2}	1.884×10^{-2}
Number of steps	72	73	8

for all $i = 1, 2, \ldots, I_k$.[9] I set the grid size $I_k = 10{,}000$. It turns out that the standard EGM takes 72 steps — one step less than VFI — until the consumption policy function satisfies the aforementioned stopping criterion. Appendix B describes the standard EGM algorithm in detail.

Table 3 compares the computational time for the standard EGM, along with that for VFI and PFI. The standard EGM converges more quickly than the VFI method proposed here. This property reflects the fact that the number of endogenous grid points for the standard EGM equals I_k whereas that for the VFI method equals $2I_k$, making it more costly to evaluate the interpolant on the exogenous grid. Although VFI takes 20% longer than the standard EGM, the difference is less than 0.01 seconds and is small. Therefore, the VFI method proposed here delivers comparable computational speed to EGM while offering the advantage of global convergence. Because the PFI method significantly accelerates convergence, its speed is much faster than the standard EGM. Moreover, with a smaller grid size of $I_k = 1,000$, VFI converges faster than EGM: VFI takes 72 steps and 5.790×10^{-3} seconds to converge, whereas EGM takes 73 steps and 5.957×10^{-3} seconds.

Overall, the above comparison suggests that the PFI method proposed in this paper is effective: it is much faster than the standard EGM and its convergence is guaranteed, unlike the standard EGM.

In Section 6, I compare the Euler equation error generated by the method proposed in the present paper and the standard EGM. Although the standard EGM is more accurate, there is a simple modification for the proposed method that makes the Euler equation error comparable to that of the standard EGM.

[9]This is the derivative of the initial value function, $u(F(k) - k)/(1 - \beta)$, in the VFI algorithm.

Table 4: Computational time for the deterministic growth model: discrete-time PFI vs. continuous-time implicit method

	$I_k = 10{,}000$		$I_k = 1{,}000$	
	Discrete-time PFI	Continuous-time	Discrete-time PFI	Continuous-time
Total time (sec.)	1.884×10^{-2}	1.056×10^{-2}	1.899×10^{-3}	2.013×10^{-3}
Number of steps	8	6	7	7
Time per step (sec.)	2.355×10^{-3}	1.751×10^{-3}	2.712×10^{-4}	2.875×10^{-4}

5.2 Comparison with the continuous-time method

Note that the method presented here resembles that of a continuous-time heterogeneous-agents model in Achdou et al. (2021): (1) the policy function is computed using finite differences of the value function, and (2) the PFI is implemented using sparse linear algebra. Because the continuous-time method is often viewed as a fast method, it makes sense to compare the present paper's method with it.

In this section, I compare two methods using the deterministic growth model. The continuous-time benchmark is the implicit method, and I use Ben Moll's code written in Matlab (`http://benjaminmoll.com/wp-content/uploads/2020/06/HJB_NGM_implicit.m`).[10] Table 4 reports the results, both for the grid size of 10,000 and of 1,000, along with the PFI results in the discrete-time model. When running these files, I disable all commands that display variables and that create figures.

When the grid size is 1,000, the proposed method is as fast as the continuous-time counterpart. For the continuous-time method, the number of steps is 7 and the average computational time is 2 milliseconds, implying that each step takes 0.29 milliseconds. Both numbers are almost identical to their corresponding numbers for the discrete-time PFI method here. The discrete-time MPFI is faster, suggesting that the method proposed in this paper performs well for a grid size that is not too large.

With a larger grid size, however, the continuous-time method converges much faster than the discrete-time method. When the grid size is as large as 10,000, the total

[10]His parameterization is identical to mine, except that the technology A is not normalized so that the steady state capital is not equal to one. The lower and upper bounds for capital are set to $0.001k_{ss}$ and $2k_{ss}$, respectively.

computation time for the continuous-time method is about 45% less than that for the discrete-time method. The number of iteration steps is smaller by about 20% for the continuous-time method than for the discrete-time PFI method, and the time per iteration step by about 25%. The latter is partly due to an increased computational burden for concavification in the discrete-time method.

Other than the computational costs from concavification, there is a reason why the method proposed in this paper is slightly slower than the continuous-time counterpart. It is that there is an additional constraint in a continuous-time model: savings is a continuous function of time. In the continuous-time model, if the decision-maker is currently on a grid point k_i, then within a small amount of time, she either moves to a subinterval below, (k_{i-1}, k_i), or moves to a subinterval above, (k_i, k_{i+1}), or stays at k_i. It is simply infeasible for the decision-maker on a grid point to move to nonadjacent subintervals. In contrast, in a discrete-time model the decision-maker does not face such a constraint. Therefore, the decision-maker on each savings grid point may choose to move to nonadjacent subintervals or to another grid point. When solving the income fluctuation problem in a discrete-time model, one needs to take care of these possibilities, thereby increasing the computational costs.

However, the costs are not very large, because one can exploit continuity and monotonicity of the policy function even in the discrete-time model. Suppose one has found out that an exogenous grid point of cash-on-hand, y_i, belongs to a certain endogenous subinterval, say, J_m. Then, the next grid point, y_{i+1}, which is greater than y_i, never lies in a subinterval below J_m. Hence, one only needs to start checking subintervals from J_m, I_m, J_{m+1}, I_{m+1}, and so on, until finds one that contains y_{i+1}. Further, if y_{i+1} is sufficiently close to y_i, then it is likely that such an interval can be found without examining a large number of subintervals.

6 Improving the Euler equation error

This section compares the Euler equation errors generated by the method proposed in this paper and the standard EGM, and introduces a simple modification that substantially improves the accuracy of the proposed method.

The Euler equation error is defined as

$$\max_{k \in k_{grid}} \left| \frac{(u')^{-1} \left(\beta u' \left(g_c(F(g_{k'}(k))) \right) F'(F(g_{k'}(k))) \right)}{g_c(k)} - 1 \right| .$$

Columns 1 and 2 of Table 5 report the Euler equation error for the method in the present paper and the standard EGM, using the grid size $I_k = 10{,}000$. The Euler equation error is smaller by an order of magnitude for the standard EGM than for the method in the present paper. This outcome is expected, given that the method in the present paper uses piecewise linear interpolation for the value function, which is known to suffer from large error where the curvature of the value function is large. The standard EGM achieves lower error, because it uses a condition not employed in the method in the present paper: the envelope condition. The envelope condition provides an accurate estimate for the derivative of the value function, whereas the method in the present paper effectively uses a finite-difference approximation, which is much less accurate, particularly when the curvature is large.

However, the difference between the consumption policy functions obtained by the two methods is actually small: the maximal difference on the grid is 5.40×10^{-5} in absolute value, and the percentage difference is 0.16%. This suggests that a small modification of the policy function obtained by the method in the present paper can significantly reduce the Euler equation error.[11]

A simple modification yields substantial improvement in accuracy: after obtaining the policy function using the PFI method, apply a single step of the standard EGM (or two steps if necessary). This *hybrid* approach is computationally inexpensive yet retains benefits of both methods: (1) it is as fast as the proposed method because

[11] Consumption policy functions obtained by VFI, PFI, and MPFI are very close to each other. Their maximal differences on the grid are about 3×10^{-7}.

Table 5: Euler equation error

PFI	Standard EGM	PFI + 1-step EGM	PFI + 2-step EGM	PFI + 3-step EGM
6.068×10^{-3}	1.343×10^{-4}	9.652×10^{-4}	1.618×10^{-4}	1.223×10^{-4}

PFI accelerates convergence, (2) its convergence is guaranteed; and (3) it utilizes the envelope condition to reduce the Euler equation error. The third column in Table 5 reports the Euler equation error for this approach. As a result of this modification, the Euler equation error becomes comparable to that of the standard EGM. The error can be further reduced by applying the standard EGM step once OR twice more, as shown in the fourth and fifth column in Table 5.

These findings suggest that the present paper's method can be easily modified so that the Euler equation error is comparable to the standard EGM while maintaining its advantages — speed and guaranteed convergence. There are other well-known remedies that can be applied here, on top of the above modification. The error can be made smaller by using a log-spaced grid, which puts more points near the lower end of the interval $\mathcal{K}$. The error can also be reduced by raising the lower bound, k_{min}.

7 A stochastic model and computing a stationary equilibrium in a heterogeneous-agent model

Another point made in Rendahl (2022) is that the same sparse probability transition matrix can be used to compute a stationary distribution of asset in a heterogeneous-agents model, as Achdou et al. (2021) show for the continuous-time model. Given that the proposed method improves the speed in the optimization step, it is worthwhile to examine how fast a stationary equilibrium in a discrete-time heterogeneous-agents model can be computed using the proposed algorithms, relative to its continuous-time counterpart.

Toward this end, I first extend the proposed algorithms to a stochastic environment using a growth model, and then apply it to a prototypical household problem in a heterogeneous-agent model.

7.1 VFI and PFI algorithms in a stochastic growth model

Imagine that the productivity shock, z, follows a time-homogeneous, first-order Markov chain, with the set of possible values denoted by Z and the transition probability from z to z' by $\pi_{z,z'}$. Let $\mathbf{\Pi}$ be the transition matrix, with its (i,j)-element being π_{z_i,z_j}.

Then, the planning problem is given by:

$$v(k,z) = \max_{c,k'}\{u(c) + \beta \sum_{z' \in Z} \pi_{z,z'} v(k',z')\}$$

subject to the resource constraint,

$$c + k' \leq F(k,z),$$

and the constraint on the next period capital, $k' \in \mathcal{K}$.

Clearly, v is the unique fixed point of the following Bellman operator:

$$\mathbb{T}w(k,z) = \max_{c,k'}\{u(c) + \beta \sum_{z' \in Z} \pi_{z,z'} w(k',z')\}$$

subject to the same set of constraints.

If $w(.,z)$ is a concave piecewise linear function with a common set of grid points, k_{grid}, for all z, then $w(.,z)$ is represented, for each z, by a vector of its values on the grid, $\mathbf{w}(z) = (w_1(z), \ldots, w_{I_k}(z))^\top$. For each z, the conditional expected value, $\sum_{z' \in Z} \pi_{z,z'} w(k',z')$, in the objective function is then also an increasing and concave piecewise linear function with a value vector $\sum_{z'} \pi_{z,z'} \mathbf{w}(z')$. In other words,

$$\sum_{z'} \pi_{z,z'} \hat{V}(k'; \mathbf{w}(z')) = \hat{V}\left(k'; \sum_{z'} \pi_{z,z'} \mathbf{w}(z')\right).$$

Hence, for a given z, Proposition 1 can still be used to characterize the solution to the maximization problem. The exogenous grid for cash-on-hand now depends on the current shock z: $y_{grid}(z) = \{F(k_1, z), F(k_2, z), \ldots, F(k_{I_k}, z)\}$,

Algorithm 3 (Fitted value iteration algorithm (Stochastic case)) *1. Initialization:*

(a) Fix the value tolerance level, $tol_v > 0$, and the capital grid, $k_{grid} = \{k_1, k_2, \ldots, k_{I_k}\}$ with $k_1 = k_{min}$ and $k_{I_k} = k_{max}$.

(b) Set a concave initial value function, e.g., $v_{0,i}(z) = u(F(k_i, z) - k_i)/(1-\beta)$ for $i = 1, \ldots, I_k$ and for $z \in Z$.

(c) Pre-compute $y_{grid}(z) = \{F(k_1, z), F(k_2, z), \ldots, F(k_{I_k}, z)\}$ for $z \in Z$. Let $y_i(z)$ denote $F(k_i, z)$.

2. VFI: for $n = 0, \ldots,$ taking $\mathbf{v_n}$ as given, do the following for each $z \in Z$:

(a) Compute $\mathbf{v_n^e}(z) := \sum_{z'} \pi_{z,z'} \mathbf{v_n}(z')$.

(b) Compute the policy function on $y_{grid}(z)$, by solving the $\mathcal{P}(y; \mathbf{v_n^e}(z))$-problem using the algorithm 1 for all $y \in y_{grid}(z)$.

(c) Update the value function using $v_{n+1,i}(z) = u(g_c(y_i(z); \mathbf{v_n^e}(z))) + \beta \mathbf{p}_n(y_i(z); \mathbf{v_n^e}(z))^\top \mathbf{v_n^e}(z)$.

(d) Stop if $||\mathbf{v_{n+1}} - \mathbf{v_n}|| < tol_v$, where

$$\mathbf{v}_n = \begin{bmatrix} \mathbf{v}_n(z_1) \\ \vdots \\ \mathbf{v}_n(z_{N_z}) \end{bmatrix}.$$

Otherwise go back to Step 2.

The VFI algorithm for a stochastic model is thus straightforward. One may use stacked vector notation:

$$\mathbf{v_{n+1}} = \mathbf{u_n} + \beta \tilde{\mathbf{P}}_\mathbf{n} \mathbf{v_n},$$

where

$$\mathbf{u}_n = \begin{bmatrix} \mathbf{u}_n(z_1) \\ \vdots \\ \mathbf{u}_n(z_{N_z}) \end{bmatrix} \text{ with } \mathbf{u}_n(z) = \begin{bmatrix} u(g_c(y_1(z); \mathbf{v_n^e}(z))) \\ \vdots \\ u(g_c(y_{I_k}(z); \mathbf{v_n^e}(z))) \end{bmatrix},$$

and

$$\tilde{\mathbf{P}}_{\mathbf{n}} := \begin{bmatrix} \mathbf{\Pi}_{\mathbf{1},\cdot} \otimes \mathbf{P}_{\mathbf{n}}(z_1) \\ \mathbf{\Pi}_{\mathbf{2},\cdot} \otimes \mathbf{P}_{\mathbf{n}}(z_2) \\ \vdots \\ \mathbf{\Pi}_{\mathbf{N_z},\cdot} \otimes \mathbf{P}_{\mathbf{n}}(z_{N_z}) \end{bmatrix} \text{ with } \mathbf{P}_n(z) = \begin{bmatrix} \mathbf{p}(y_1(z); \mathbf{v}^{\mathbf{e}}_{\mathbf{n}}(z))^\top \\ \vdots \\ \mathbf{p}(y_{I_k}(z); \mathbf{v}^{\mathbf{e}}_{\mathbf{n}}(z))^\top \end{bmatrix}$$

for all $z \in \{z_1, \ldots, z_{N_z}\}$, and $\mathbf{\Pi}_{\mathbf{i},\cdot}$ denotes the i-th row of the matrix $\mathbf{\Pi}$ for all $i = 1, \ldots, N_z$.

Convergence can be accelerated using PFI. With the stacked vector notation, PFI solves $\mathbf{w}_{\mathbf{n}} = (\mathbf{I} - \beta\tilde{\mathbf{P}}_{\mathbf{n}})^{-1}\mathbf{u}_{\mathbf{n}}$ to obtain $\mathbf{w}_{\mathbf{n}}$ and then, for each z, concavifies $\mathbf{w}_{\mathbf{n}}(z)$ to obtain $\mathbf{v}_{n+1}(z)$.

7.2 Incomplete-market, heterogeneous-agents model

Consider the following household's problem in Huggett (1993) model in a stationary equilibrium:

$$v(a, z) = \max_{c, a' \leq a_{max}} u(c) + \beta \sum_{z' \in Z} \pi_{z,z'} v(a', z')$$

subject to the budget constraint,

$$c + a' \leq (1 + r)a + z,$$

and the borrowing constraint,

$$a' \geq a_{min}.$$

The household faces a constant real interest rate, r, and, in each period, chooses its consumption, c, and the next period asset holding, a', after receiving an exogenous endowment, z. The endowment z evolves according to a finite-state Markov chain with the state space Z. Let z_{min} denote the smallest element of Z, and I assume that $z_{min} > 0$. The household's preference discount factor β satisfies $0 \leq \beta < 1$. The real interest rate, r, is assumed to be strictly positive, $r > 0$, and to satisfy $\beta(1 + r) < 1$. I assume that borrowing is allowed, $a_{min} < 0$, but that the borrowing limit is tight in that $a_{min} > -z_{min}/r$. It is also assumed that $a_{max} > 0$.

Because the borrowing limit is tight and $r > 0$, it is feasible to set $a' = a$ at any current asset level $a \geq a_{min}$: with $a' = a$, the maximal possible consumption level is $(1+r)a - a + z = ra + z \geq ra_{min} + z = r(a_{min} + z_{min}/r) > 0$. The value of following such a policy can be used as the initial condition for VFI and PFI.

For a given real interest rate, this income fluctuation problem fits the previous stochastic growth model framework by defining $a = k$ and $F(k, z) = (1+r)k + z$. Hence, the VFI and the PFI algorithms approximately obtain $\mathbf{v}(r) = \mathbf{u}(\mathbf{v}(r)) + \beta\tilde{\mathbf{P}}(\mathbf{v}(r))\mathbf{v}(r)$.

The matrix $\tilde{\mathbf{P}}(\mathbf{v}(r))$ is a sparse, probability transition matrix for joint state $(a, z) \in A_{grid} \times Z$. Its stationary distribution, g, satisfies

$$\mathbf{g}(r) = \tilde{\mathbf{P}}(\mathbf{v}(r))^\top \mathbf{g}(r).$$

The joint probability of a pair (a_{i_a}, z_{i_z}) given r, $g(a_{i_a}, z_{i_z}; r)$, is stored as the $(I_k \times (i_z - 1) + i_a)$-th element in the $\mathbf{g}(r)$ vector. The $\tilde{\mathbf{P}}$ matrix, which is used to solve an individual's optimization problem, is also used to calculate a stationary distribution of individual states. As Rendahl (2022) points out, this property is the same as in the continuous-time method in Achdou et al. (2021). An invariant distribution of $\tilde{\mathbf{P}}(r)$ can be computed instantly by a "dirty trick" described in Achdou et al. (2021) or by repeatedly applying a transition matrix $\tilde{\mathbf{P}}(r)$ from some initial distribution until convergence is achieved.

The aggregate savings given the real interest rate, $A(r)$, can be computed as

$$A(r) = \mathbf{g}(r)^\top \begin{bmatrix} \mathbf{a} \\ \vdots \\ \mathbf{a} \end{bmatrix},$$

where $\mathbf{a} = (a_1, a_2, \ldots, a_{I_k})^\top$ is a vector of exogenous grid points and the rightmost vector stacks $\mathbf{a}$ for N_z times. In the Huggett (1993) model, the aggregate savings needs be zero in an equilibrium. To find an equilibrium real interest rate, one needs to set up an outer loop over r that implements, e.g., bisection method to find a zero of $A(r)$.

Overall, when solving a heterogeneous-agents model, this paper's method and the continuous-time method differ only in how they solve the individual optimization problem. The outer loop for the real interest rate and computation of a stationary distribution can be set identical, regardless of the model being a discrete- or a continuous-time one. Given that this paper's discrete-time method can solve an individual problem as fast as the continuous-time method, it is expected that they perform equally well in computing an equilibrium.

For the continuous-time method, I use Ben Moll's code (`http://benjaminmoll.com/wp-content/uploads/2020/06/huggett_equilibrium_iterate.m`). As before, all display and plotting commands are disabled. In his code, the preference discount rate is set to 0.05, the relative risk aversion to 2, the borrowing limit k_{min} to -0.15, and the asset upper-bound k_{max} to 5. The endowment shock is a two-state process with $z_1 = 0.2$, $z_2 = 0.1$, and the state switches with a Poisson arrival rate of 1.2.

I use the same parameter values except for the discrete-time transition probability matrix. This change is necessary. In the continuous-time model, within a small time interval, dt, the probability of state switching is $1.2dt$. If one interprets a model with $dt = 1$ as a discrete-time model, it exceeds 1, implying that it is inappropriate for probability. I choose the transition probabilities so that the outer loop for the real rate converges in the same number of steps for discrete- and continuous-time models for $I_k = 1,000$, to facilitate speed comparison. This happens when I set $\pi_{1,1} = \pi_{2,2} = 0.8$ and $\pi_{1,2} = \pi_{2,1} = 0.2$.

I use the same algorithms as in Moll's code to compute a stationary distribution given a real interest rate ("dirty trick") and to compute an equilibrium interest rate (the bisection method).

With 1,000 grid points, the average computational time under PFI and MPFI are similar to the continuous-time method, with PFI being slightly faster than the continuous-time method. Given that the number of outer-loop steps is the same across all methods, these results indicate that the present paper's method is an efficient and robust tool

Table 6: Computational time for the Huggett model: discrete-time PFI, MPFI and continuous-time implicit method

	$I_k = 10{,}000$			$I_k = 1{,}000$		
	Discrete PFI	Discrete MPFI	Continuous	Discrete PFI	Discrete MPFI	Continuous
Total time (sec.)	0.8109	0.6569	0.8187	4.7071×10^{-2}	7.9725×10^{-2}	$5.7952 \times 10^{-}$
Outer loop steps	10	10	12	11	11	11

in computation for heterogeneous-agents general equilibrium models. With 10,000 grid points, MPFI becomes faster than PFI as before, and the continuous-time method is comparable to PFI.

However, this does not imply that the continuous-time method is slower than this paper's method. In the continuous-time method, one can adjust another parameter, Δ, which is the inverse of the discretized time interval dt, to improve speed. The above numbers are obtained under $\Delta = 1{,}000$, and they can be improved by using larger Δ. For example, with $\Delta = 10{,}000$, the average time becomes 4.8187×10^{-2} seconds for $I_k = 1{,}000$ and 0.6993 seconds for $I_k = 10{,}000$; with $\Delta = 50{,}000$, the average time becomes 4.4271×10^{-2} seconds for $I_k = 1{,}000$ and 0.6863 seconds for $I_k = 10{,}000$.

Overall, the results suggest that the discrete-time method proposed in the present paper is a useful, comparably fast alternative for a continuous-time method. Although I focus on computation of a stationary equilibrium here, the sequence-space Jacobian method (Auclert et al., 2021) can be applied to compute a dynamic response to aggregate shocks.

8 Extension: endogenous labor supply

In this section I extend the previous income fluctuation problem to incorporate another intra-temporal choice: endogenous labor supply. I assume that the utility is separable between consumption and labor supply, so that the Euler equation can be analytically solved for the current consumption, taking the derivative of the next-period value function.[12]

Denoting the labor supply by h, the household's problem is now given by

$$\max_{c,h,a'\leq a_{max}} u(c) - q(h) + \beta\hat{V}(a';\mathbf{v}^{\mathbf{e}}(z))$$

subject to the budget constraint,

$$c + a' \leq (1+r)a + wzh,$$

and the borrowing constraint,

$$a' \geq a_{min}.$$

The function q denotes the disutility from labor, and I assume that its derivative, q', has an analytical inverse. As before, I assume $\mathbf{v}^{\mathbf{e}}(z)$ is such that the continuation value function, $\hat{V}(\cdot;\mathbf{v})$, is concave.

8.1 A one-step approach

The first approach is a one-step one, which computes the policy functions for consumption, savings, and labor supply simultaneously.

Fix an arbitrary $z \in Z$. From the first-order condition, $q'(h) = wzu'(c)$ holds, implying $h = (q')^{-1}(wzu'(c))$. Because both w and z are exogenous, the above equation relates the optimal consumption and labor.

Consider when it is optimal to choose $a' \in (a_i, a_{i+1})$ for $i \in \{1, 2, \ldots, I_k - 1\}$. For the optimal savings to be in the interval (a_i, a_{i+1}), the current consumption must satisfy the Euler equation, implying $u'(c) = \lambda_i$ and hence $c = c_i^*$. From the intratemporal first-order condition, the optimal labor supply is pinned down at $h = h_i^* := (q')^{-1}(wzu'(c_i^*))$. Hence, if the beginning-of-period cash-on-hand, $y^{beg} := (1+r)a$, is in an interval $\mathcal{I}_i :=$

[12]Barillas and Fernandez-Villaverde (2007) extend the standard EGM to incorporate endogenous labor supply with non-separable utility.

$(c_i^* - wzh_i^* + a_i, c_i^* - wzh_i^* + a_{i+1})$, we have

$$(c, h, a') = (c_i^*, h_i^*, y^{beg} + wzh_i^* - c_i^*).$$

On an $\mathcal{I}$ interval, a' is linear in y^{beg} and both c and h are constant.

Consider when it is optimal to choose $a' = a_i$ for $i \in \{2, 3, \ldots, I_k - 1\}$. As in the model with exogenous labor supply, the Euler equation implies $\lambda_i \leq u'(c) \leq \lambda_{i-1}$, and it follows that $c_{i-1}^* \leq c \leq c_i^*$. Hence, if the beginning-of-period cash-on-hand, y^{beg}, is in an interval $\mathcal{J}_i := (c_{i-1}^* - wzh_{i-1}^* + a_i, c_i^* - wzh_i^* + a_i)$, we have

$$(c, h, a') = \left(c, (q')^{-1}(wzu'(c)), a_i\right),$$

where c solves a nonlinear equation,

$$c - wz(q')^{-1}(wzu'(c)) = y^{beg} - a_i.$$

On $\mathcal{J}$ intervals, c and h are non-linear in y^{beg}, whereas a' is constant. This property also holds true for the boundary cases, $a' = a_1$ and $a' = a_{I_k}$.

Note that the policy function for savings, a', remains piecewise linear, implying that the evaluation of the continuation value reduces to a vector multiplication:

$$\hat{V}(g_{a'}(y_i(z); \mathbf{v^e}(z)); \mathbf{v^e}(z)) = \mathbf{p}(y_i(z); \mathbf{v^e}(z))^\top \mathbf{v^e}(z). \tag{4}$$

Therefore, PFI is again equivalent to solving a large yet sparse system of linear equations, which can be executed very quickly.

Is it costly to evaluate the policy function for consumption and labor on $\mathcal{J}$ intervals, because it requires solving a nonlinear equation? Not significantly. Notice that equation (4) is independent of the continuation value function, $\mathbf{v^e}(z)$. Hence, for any combination of an arbitrary grid point for the beginning-of-period cash on hand, y_j^{beg}, and an arbitrary grid point for the next-period savings, a_i, one can pre-compute $c_{i,j}$ that satisfies equation

(4). Therefore, for each $i \in \{1, 2, \ldots, I_k - 1\}$, one just needs to find a set of j's such that $c^*_{i-1} \leq c_{i,j} \leq c^*_i$.[13]

8.2 A two-step approach

The second approach is to solve an approximate problem in two steps. It is based on the observation that the household's optimization problem can be written as

$$\max_h -q(h) + m(y; \mathbf{v}^{\mathbf{e}}(z))$$

subject to the budget constraint $y \leq (1 + r)a + wzh$. This is because the $\mathcal{P}(y; \mathbf{v}^{\mathbf{e}}(z))$-problem can be interpreted as the household's problem after she has chosen her labor supply and, as a result, labor earnings.

This reformulated problem can be approximated by

$$\max_h -q(h) + \hat{V}(y; \mathbf{m})$$

subject to the budget constraint, where $\mathbf{m} = (m(y_1; \mathbf{v}^{\mathbf{e}}(z)), \ldots, m(y_{I_y}; \mathbf{v}^{\mathbf{e}}(z)))^\top$ is the vector of values of $m(.; \mathbf{v}^{\mathbf{e}}(z))$ on a cash-on-hand grid. The cash-on-hand grid can be either exogenous or endogenous. Because the function m is not in general piecewise linear, this problem is only an approximation of the original problem.

Observe that this approximate problem has the same structure as the problem solved in Proposition 1 after a change of variables.[14] Hence, its solution can be analytically obtained and is a piecewise linear function.

The two-step approach therefore eliminates the need to solve nonlinear equation, and the policy function can be evaluated with piecewise linear interpolation. Because the problem it solves is only an approximation of that the one-step approach solves, the resulting policy function is less accurate than that of the one-step approach.

[13]For $i = 1$, one needs to find a set of j's such that $c_{i,j} \leq c^*_i$, and for $i = I_k$, a set of j's such that $c^*_{I_k - 1} \leq c_{i,j}$.

[14]This can be seen by changing variables as follows: $d = -wzh$ and $\tilde{u}(d) = -q(-d/(wz))$. Then the objective function is $\tilde{u}(d) + \beta\hat{V}(y; \mathbf{m}/\beta)$ and the constraint is $d + y \leq (1 + r)a$.

9 Conclusion

In this paper, I developed a fast, accurate, and stable solution method for income fluctuation problems with a single asset. The method combines the standard endogenous grid method and value function iteration with piecewise linear interpolation and obtains an analytical solution under the condition that the value function is a concave, piecewise-linear function. The availability of the analytical solution eliminates errors in optimization and in policy function evaluation. Piecewise linearity of the analytical policy function makes the optimization step of the value iteration algorithm computationally inexpensive. Policy function iteration algorithms further accelerate convergence, and the speed is comparable to the two well-known fast methods: the standard endogenous grid method and the continuous-time method. In addition, the proposed method is stable, with guaranteed convergence — a particularly attractive property when solving a model repeatedly under varying parameter values, such as in structural estimation or in computing general equilibrium prices. In terms of accuracy, I have shown that a simple modification reduces the Euler equation error to a level comparable to that of the standard endogenous grid method. These properties make the method a viable alternative for solving dynamic programming problems in applied macroeconomic models.

A Implementation of Step 1 in Algorithm 1

The endogenous grid points $\{y_1^{end}, \ldots, y_{2I_k}^{end}\}$ are given by:

$$\begin{aligned}
y_1^{end} &= k_1, \\
y_{2i}^{end} &= c_i^* + k_i, \ \text{for } i = 1, \ldots, I_k - 1, \\
y_{2i+1}^{end} &= c_i^* + k_{i+1}, \ \text{for } i = 1, \ldots, I_k - 1, \\
y_{2I_k}^{end} &= y_{max},
\end{aligned}$$

where y_{max} is a constant that is strictly greater than both $c_{I_k-1}^* + k_{I_k}$ and the largest *exogenous* grid point, y_{I_y}. In theory, y_{max} can be arbitrarily large, but in practice it is better to set to a number that is close to $\max\{c^*_{I_k-1}+k_{I_k}, y_{I_y}\}$.

The policy function for consumption takes on the following values on the endogenous grid:

$$\begin{aligned} g_c(y^{end}_1) &= 0, \\ g_c(y^{end}_{2i}) &= g_c(y^{end}_{2i+1}) = c^*_i, \text{ for } i = 1, \dots, I_k - 1, \\ g_c(y^{end}_{2I_k}) &= y_{max} - k_{i_k}. \end{aligned}$$

The policy function for savings on the endogenous grid can be obtained by calculating $g_k(y^{end}_i) = y^{end}_i - g_c(y^{end}_i)$ for all i.

B Standard EGM algorithm

Algorithm 4 (Standard EGM algorithm) *1. Initialization:*

(a) Fix the policy tolerance level, $tol_c > 0$, and the capital grid, $k_{grid} = \{k_1, k_2, \dots, k_{I_k}\}$ with $k_1 = k_{min}$ and $k_{I_k} = k_{max}$.

(b) Pre-compute $y_{grid} = \{y_1, y_2, \dots, y_{I_k}\}$, with $y_i = F(k_i)$ for $i = 1, \dots, I_k$.

(c) Set an initial condition for the value function derivative estimate, $\mathbf{Dv}_0$, so that $Dv_{0,i} = u'(F(k_i) - k_i)(F'(k_i) - 1)/(1-\beta)$ for $i = 1, \dots, I_k$.

2. EGM: for $n = 0, \dots,$ taking $\mathbf{Dv}_n$ as given,

(a) Compute, for all $i = 1, \dots, I_k$, $c^{**}_i := (u')^{-1}(\beta Dv_{n,i})$ and $y^{end}_i = c^{**}_i + k_i$.

(b) Obtain the policy functions for consumption and capital, $g_{c,n}(y_i)$ and $g_{k',n}(y_i)$, by linearly interpolating data $\{y^{end}_j, c^{**}_j\}$ and $\{y^{end}_j, k_j\}$, respectively.

(c) Update the value function derivative estimate as $Dv_{n+1,i} = u'(g_{c,n}(y_i))F'(k_i)$ for all $i = 1, \dots, I_k$.

(d) Stop if $\max_i |g_{c,n}(y_i) - g_{c,n-1}(y_i)| < tol_c$. Go back to 2.1 otherwise.

References

Achdou, Yves, Jiequn Han, Jean-Michel Lasry, Pierre-Louis Lions, and Benjamin Moll (2021): "Income and Wealth Distribution in Macroeconomics: A Continuous-Time Approach," *The Review of Economic Studies*, 89, 45–86.

Aiyagari, S. Rao (1994): "Uninsured Idiosyncratic Risk and Aggregate Saving," *The Quarterly Journal of Economics*, 109, 659–684.

Auclert, Adrien, Bence Bardóczy, Matthew Rognlie, and Ludwig Straub (2021): "Using the Sequence-Space Jacobian to Solve and Estimate Heterogeneous-Agent Models," *Econometrica*, 89, 2375–2408.

Barillas, Francisco and Jesus Fernandez-Villaverde (2007): "A generalization of the endogenous grid method," *Journal of Economic Dynamics and Control*, 31, 2698–2712.

Bellman, Richard (1957): *Dynamic Programming*, Princeton, NJ: Princeton University Press, dover (2003) paperback edition.

Cai, Yongyang and Kenneth L. Judd (2012): "Shape-preserving dynamic programming," *Mathematical Methods of Operations Research*, 77, 407–421.

——— (2014): "Chapter 8 - Advances in Numerical Dynamic Programming and New Applications," in *Handbook of Computational Economics Vol. 3*, ed. by K. Schmedders and K. L. Judd, Elsevier, vol. 3 of *Handbook of Computational Economics*, 479–516.

Carroll, Christopher D. (2006): "The method of endogenous gridpoints for solving dynamic stochastic optimization problems," *Economics Letters*, 91, 312–320.

Fukushima, Kenichi and Yuichiro Waki (2013): "A polyhedral approximation approach to concave numerical dynamic programming," *Journal of Economic Dynamics and Control*, 37, 2322–2335.

Howard, Ronald A. (1960): *Dynamic Programming and Markov Process*, Cambridge, MA.: The MIT Press.

Huggett, Mark (1993): "The risk-free rate in heterogeneous-agent incomplete-insurance economies," *Journal of Economic Dynamics and Control*, 17, 953–969.

Judd, Kenneth L. (1998): *Numerical Methods in Economics*, Cambridge, MA.: The MIT Press.

Judd, Kenneth L. and Andrew Solnick (1994): "Numerical Dynamic Programming with Shape-Preserving Splines," *mimeo.*

Rendahl, Pontus (2022): "Continuous vs. discrete time: Some computational insights," *Journal of Economic Dynamics and Control*, 144.

Santos, Manuel S. and John Rust (2004): "Convergence Properties of Policy Iteration," *SIAM Journal on Control and Optimization*, 42, 2094–2115.

Santos, Manuel S. and Jesus Vigo-Aguiar (1998): "Analysis of a Numerical Dynamic Programming Algorithm Applied to Economic Models," *Econometrica*, 66, 409–426.

Schumaker, Larry I. (1983): "On Shape Preserving Quadratic Spline Interpolation," *SIAM Journal on Numerical Analysis*, 20, 854–864.

Stachurski, John (2008): "Continuous State Dynamic Programming via Nonexpansive Approximation," *Computational Economics*, 31, 141–160.

Young, Eric R. (2010): "Solving the incomplete markets model with aggregate uncertainty using the Krusell-Smith algorithm and non-stochastic simulations," *Journal of Economic Dynamics and Control*, 34, 36–41.