Dynamic Programming II

Efficient Algorithms for the Household Problem

Sebastian Dyrda, University of Toronto

ECO2101 Lecture 3 | March 9, 2026

Roadmap for Lectures 2–3

Two lectures, two throughline applications:

Lecture 2 (last week): **A firm with stochastic productivity**

- Fixed-point theory, contractions, Markov dynamics, optimal stopping
- Threshold policies, monotonicity, continuation values
- Sargent & Stachurski, *Dynamic Programming*, Chapters 1–4

→ Lectures 4–7: Hopenhayn (1992), Khan–Thomas (2008)

Lecture 3 (today): **A household with stochastic income**

- MDPs, VFI / HPI / OPI, endogenous gridpoints, envelope conditions
- Gumbel / extreme-value shocks, discrete-continuous choice
- Sargent & Stachurski, Chapter 5; Carroll (2006); Maliar & Maliar (2013)

→ Lectures 8–10: HANK and sequence-space Jacobians

Roadmap

The Household Problem

Faster Iterations: HPI and OPI

Euler equation methods with interpolation: PFI, EGM, ECM

Extreme Value Shocks and the Gumbel Max Trick

Summary and Forward Links

Roadmap

The Household Problem

Faster Iterations: HPI and OPI

Euler equation methods with interpolation: PFI, EGM, ECM

Extreme Value Shocks and the Gumbel Max Trick

Summary and Forward Links

The Household Problem — Primitives

Primitives

- Finite asset grid $\mathcal{A} \subset \mathbb{R}$ with borrowing limit $\underline{a} \geq 0$
- Labor income $e \in \mathcal{E} = \{e_1, \ldots, e_n\}$; $(e_t)_{t \geq 0}$ is Q-Markov on $\mathcal{E}$
- CRRA preferences: $u(c) = c^{1-\gamma}/(1-\gamma)$, $\gamma > 0$, $\gamma \neq 1$ (log when $\gamma = 1$)
- Gross interest rate $R = 1 + r$, wage w; both taken as given

Bellman equation

$$V(a, e) \;=\; \max_{a' \in \Gamma(a,e)} \Big\{ u\big(Ra + we - a'\big) \;+\; \beta \sum_{e'} V(a', e')\, Q(e, e') \Big\}$$

$$\Gamma(a, e) \;=\; \big\{ a' \geq \underline{a} : Ra + we - a' \geq 0 \big\}$$

From Lecture 2. This is a Markov Decision Process. The Bellman operator T is a contraction of modulus β on $\mathbb{R}^{\mathcal{A} \times \mathcal{E}}$ under $\| \cdot \|_\infty$. The unique fixed point V^* exists; any V^*-greedy policy σ^* is optimal.

The Income Process — Tauchen Discretization

Specification. AR(1) in logs:

$$\ln e_{t+1} \;=\; \rho\, \ln e_t \;+\; \varepsilon_{t+1}, \qquad \varepsilon_t \overset{iid}{\sim} \mathcal{N}(0, \sigma_\varepsilon^2)$$

Tauchen (1986). Approximate by a Markov chain on n evenly-spaced grid points $\{e_1, \ldots, e_n\}$:

$$Q(e_i, e_j) \;=\; \Phi\!\left(\frac{e_j + \delta/2 - \rho\, e_i}{\sigma_\varepsilon}\right) \;-\; \Phi\!\left(\frac{e_j - \delta/2 - \rho\, e_i}{\sigma_\varepsilon}\right)$$

where δ is the grid spacing and Φ is the standard-normal CDF (with boundary corrections).

Key properties: $Q \geq 0$, $Q\mathbf{1} = \mathbf{1}$. When $\rho \geq 0$, Q is *monotone increasing* — income persistence is preserved (Exercise 3.2.2 in S&S).

Tauchen Discretization — Implementation

Input: AR(1) process $\ln e_{t+1} = \rho \ln e_t + \varepsilon_{t+1},\ \varepsilon_t \overset{\text{iid}}{\sim} N(0, \sigma_\varepsilon^2)$.
Choose n grid points and width parameter m (typically $m = 3$).

Step 1: Build the grid. The stationary std. dev. is $\sigma_x = \sigma_\varepsilon / \sqrt{1-\rho^2}$. Set:

$$x_1 = -m\,\sigma_x, \qquad x_n = m\,\sigma_x, \qquad \delta = \frac{x_n - x_1}{n-1}$$

and $x_i = x_1 + (i-1)\delta$ for $i = 1, \ldots, n$. The income grid is $E = \{e^{x_1}, \ldots, e^{x_n}\}$.

Step 2: Build the transition matrix Q. Let Φ denote the standard normal CDF and set $F(\cdot) = \Phi(\cdot/\sigma_\varepsilon)$. For each $i, j \in \{1, \ldots, n\}$:

$$Q(x_i, x_j) = \begin{cases} F(x_1 - \rho\, x_i + \delta/2) & \text{if } j = 1 \\ 1 - F(x_n - \rho\, x_i - \delta/2) & \text{if } j = n \\ F(x_j - \rho\, x_i + \delta/2) - F(x_j - \rho\, x_i - \delta/2) & \text{otherwise} \end{cases}$$

The boundary rules ensure $\sum_j Q(x_i, x_j) = 1$ for every i.
Output: income vector $E \in \mathbb{R}^n$ and $n \times n$ stochastic matrix Q.

Why VFI Is Slow — Two Distinct Problems

Value function iteration applies T repeatedly: $V_{k+1} = TV_k$.

Problem 1: Slow convergence when $\beta \approx 1$.
The contraction bound gives $\|V_k - V^*\|_\infty \leq \beta^k \|V_0 - V^*\|_\infty$. To reduce the error by a factor κ:

$$k \;\geq\; \frac{\ln \kappa}{-\ln \beta}$$

With $\beta = 0.98$ and $\kappa = 10^3$: $\quad k \geq 6.91/0.020 \approx$ 342 iterations.

Problem 2: The inner optimization is expensive on a discrete grid.
For each of the $n_a \times n_e$ states, computing $(TV)(a_i, e_j)$ requires evaluating the objective at all n_a candidate savings values:

$$\max_{k=1,\ldots,n_a} \left\{ u(Ra_i + we_j - a'_k) + \beta \textstyle\sum_{e'} V(a'_k, e')\, Q(e_j, e') \right\}$$

Cost per sweep: $O(n_a^2 \cdot n_e)$.

Roadmap

The Household Problem

Faster Iterations: HPI and OPI

Euler equation methods with interpolation: PFI, EGM, ECM

Extreme Value Shocks and the Gumbel Max Trick

Summary and Forward Links

Howard Policy Iteration (HPI)

Key idea. Instead of applying the Bellman operator once,

$$V_{k+1} = TV_k,$$

fix the current policy σ_k, compute its *exact* value V_{σ_k}, and only then improve the policy.

Important practical restriction. In the household problem, we do *not* search over all policies in an abstract set Σ. We restrict attention to **grid policies**

$$\sigma : A \times E \to A,$$

i.e. for each state (a_i, e_j), the policy selects one of the grid points

$$a' \in \{a_1, \ldots, a_{n_a}\}.$$

HPI Algorithm

HPI Algorithm (finite-state MDP; S&S Algorithm 5.3)

1. Start from any grid policy σ_0; set $k = 0$.
2. **Policy evaluation:**
$$V_{\sigma_k} = (I - \beta P_{\sigma_k})^{-1} r_{\sigma_k}.$$
3. **Policy improvement:** for each state (a_i, e_j),
$$\sigma_{k+1}(a_i, e_j) \in \arg\max_{a' \in A} \left\{ u(Ra_i + we_j - a') + \beta \sum_{e'} V_{\sigma_k}(a', e') Q(e_j, e') \right\}.$$
4. If $\sigma_{k+1} = \sigma_k$, stop. Otherwise $k \leftarrow k + 1$ and repeat.

Here P_σ and r_σ are the transition matrix and reward vector induced by the grid policy σ. The evaluation step solves a linear system of size $n_a n_e \times n_a n_e$.

Howard Policy Iteration (HPI)

Why faster than VFI? VFI repeatedly performs

$$V_{k+1} = TV_k,$$

so it improves continuation values only one Bellman step at a time. HPI instead computes the *full value* of the current policy before updating it.

Theory. For finite-state MDPs, HPI returns an *exact* optimal policy in a finite number of steps. Moreover, locally it has Newton-type speed:

$$\|V_{k+1} - V_k\| \leq N\|V_k - V_{k-1}\|^2,$$

so convergence is **quadratic** rather than linear.

HPI: why cheaper than brute force?

Why policy improvement is cheaper than brute force here. Under standard assumptions in the savings problem (CRRA utility, linear budget set, concavity of the Bellman objective),

- $V(a, e)$ is concave in a,
- the savings policy $a'(a, e)$ is increasing in a,
- and typically also increasing in e.

Hence, in the improvement step:

- we do *not* need to search over all n_a candidate savings choices at every state;
- concavity implies a **single-peaked** objective in a', so binary / bracketing search is possible;
- monotonicity implies the optimizer at a_{i+1} lies weakly to the right of the optimizer at a_i, so we can use the previous maximizer as the lower bound of the next search.

Practical implication. HPI reduces the number of outer iterations dramatically, and concavity / monotonicity reduce the cost of each policy-improvement sweep. With $\beta \approx 0.98$, one often sees about $15{-}20$ outer HPI steps versus about 340 VFI iterations.

S&S §5.1.4.2–5.1.4.3; Puterman (2005), Thm. 6.4.8

Optimistic Policy Iteration (OPI)

Three algorithms differ only in how the current policy is evaluated before the next policy improvement.

Value Function Iteration (VFI)

$$V_{k+1} = TV_k.$$

Howard Policy Iteration (HPI)

$$V_{\sigma_k} = (I - \beta P_{\sigma_k})^{-1} r_{\sigma_k}.$$

This corresponds to *exact* policy evaluation.

Optimistic Policy Iteration (OPI)
Replace the exact solve with repeated applications of the policy operator

$$T_\sigma V = r_\sigma + \beta P_\sigma V.$$

S&S §5.1.4.4

Optimistic Policy Iteration (OPI): Algorithm

OPI Algorithm

1. Start with V_0 and policy σ_0.
2. **Partial policy evaluation**

   $$V_{k+1} = T_{\sigma_k}^m V_k,$$

   where m is a fixed integer.
3. **Policy improvement**

   $$\sigma_{k+1}(x) \in \arg\max_a \{r(x,a) + \beta E[V_{k+1}(x')|x,a]\}.$$

4. If $\sigma_{k+1} = \sigma_k$, stop.

Interpretation:

$$m = 1 \Rightarrow \text{VFI}, \qquad m \to \infty \Rightarrow \text{HPI}.$$

Thus OPI interpolates between the two algorithms. S&S §5.1.4.4

Comparison of Dynamic Programming Algorithms

All algorithms compute the unique fixed point V^* of the Bellman operator.

Method	Policy evaluation	Iteration map	Convergence rate
VFI	none	$V_{k+1} = TV_k$	linear
OPI	partial (m steps)	$V_{k+1} = T_{\sigma_k}^m V_k$	superlinear
HPI	exact	Newton step	quadratic

Interpretation

- VFI: contraction mapping with modulus β

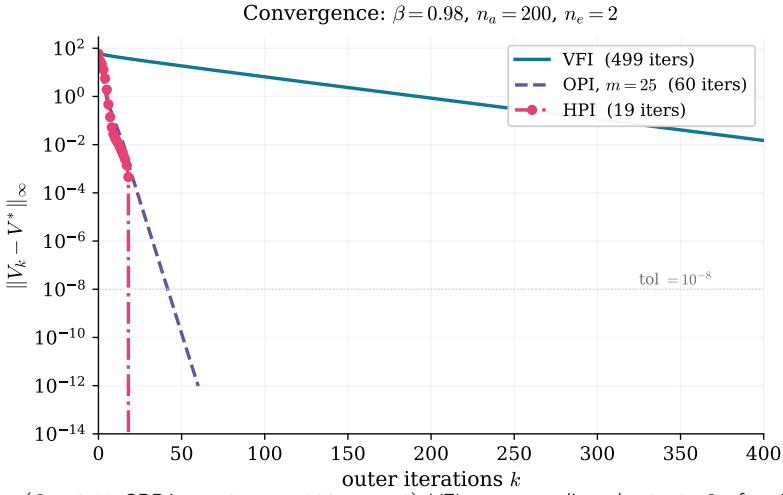
$$\|V_{k+1} - V^*\| \leq \beta \|V_k - V^*\|$$

- HPI: equivalent to Newton's method applied to

$$F(V) = V - TV$$

- OPI: *inexact Newton iteration*

Convergence: VFI vs. OPI vs. HPI



Household savings ($\beta = 0.98$, CRRA $\gamma = 2$, $n_a = 200$, $n_e = 2$). VFI converges linearly at rate β: after 400 sweeps the error is still $\sim 10^{-2}$. HPI reaches machine precision in 19 steps (quadratic). OPI ($m = 25$) splits the difference at 60 outer iterations.

From Grid to Interpolation

So far: iteration on a discrete grid

VFI, HPI, OPI all restrict the policy to grid points:

$$\sigma : \mathcal{A} \times E \to \mathcal{A}$$

Now: interpolation off the grid

Allow $a' \in [\underline{a}, \bar{a}]$, not just $a' \in \mathcal{A}$. Work directly with the Euler equation and approximate the policy function between grid points.

Three methods: PFI with interpolation, EGM, ECM.
All avoid the $O(n_a^2)$ grid search; EGM and ECM also avoid root-finding.

Roadmap

The Household Problem

Faster Iterations: HPI and OPI

Euler equation methods with interpolation: PFI, EGM, ECM

Extreme Value Shocks and the Gumbel Max Trick

Summary and Forward Links

Solving the Income Fluctuation Problem

Bellman equation:

$$\begin{aligned} V(a, e_j) &= \max_{\{c,\, a'\}} \; u(c) + \beta \sum_{e' \in E} Q(e'|e_j)\, V(a', e') \\ &\quad \text{s.t.} \\ c + a' &\leq Ra + we_j \\ a' &\geq \underline{a} \end{aligned}$$

The Euler equation, once we substitute in the budget constraint, reads:

$$u_c\big(Ra + we_j - a'\big) - \beta R \sum_{e' \in E} Q(e'|e_j)\, u_c\big(Ra' + we' - a''\big) \geq 0$$

where the inequality is strict (equality when the constraint does not bind; strict inequality when $a' = \underline{a}$). We look for the policy function $a'(a, e_j)$.

Deriving the Euler Equation

The first-order conditions are:

$$\begin{aligned} u_c(c) - \lambda &= 0 \\ \beta \sum_{e' \in E} Q(e'|e_j)\, V_a(a', e') - \lambda + \mu &= 0 \\ \mu\,(a' - \underline{a}) &= 0, \qquad \mu \geq 0 \end{aligned}$$

Envelope condition: $V_a(a, e_j) = R\lambda = R\,u_c(c)$. Applying the same condition at (a', e') gives $V_a(a', e') = R\,u_c(c')$. Substituting:

$$\beta R \sum_{e' \in E} Q(e'|e_j)\, u_c(c') - u_c(c) + \mu = 0$$

Using the budget constraint $c = Ra + we_j - a'$ and $c' = Ra' + we' - a''$:

$$u_c(Ra + we_j - a') - \beta R \sum_{e' \in E} Q(e'|e_j)\, u_c(Ra' + we' - a'') = \mu \geq 0$$

Linear Interpolation

All three methods require evaluating a function f at points $\tilde{a}$ that lie *between* grid points. Suppose we know f on the grid $\{a_0, a_1, \ldots, a_m\}$.

Piecewise linear interpolation. For $\tilde{a} \in [a_i,\ a_{i+1}]$, define:

$$\hat{f}(\tilde{a}) \;=\; f(a_i) \;+\; (\tilde{a} - a_i) \cdot \underbrace{\frac{f(a_{i+1}) - f(a_i)}{a_{i+1} - a_i}}_{\text{slope on } [a_i,\, a_{i+1}]}$$

This connects adjacent grid values with straight-line segments. It is:

- continuous on $[a_0, a_m]$
- exact at the grid points: $\hat{f}(a_i) = f(a_i)$
- first-order accurate: error is $O\big(\Delta a^2\big)$ where $\Delta a = \max_i (a_{i+1} - a_i)$

Remark. Higher-order alternatives (cubic splines, Chebyshev polynomials, shape-preserving interpolants) can be used in place of linear interpolation throughout. The algorithms that follow work the same way — only the interpolation step changes.

Policy Function Iteration (PFI) with Interpolation

1. Construct a grid $\{a_0, a_1, \ldots, a_m\}$ with $a_0 = \underline{a}$.
2. Guess a policy $\widehat{a}^0(a_i, e_j)$ for each grid point.
3. For each (a_i, e_j), check whether the borrowing constraint binds:

$$u_c(Ra_i + we_j - \underline{a}) - \beta R \sum_{e' \in E} Q(e'|e_j)\, u_c\big(R\underline{a} + we' - \widehat{a}^0(\underline{a}, e')\big) > 0$$

   If yes: set $a'(a_i, e_j) = \underline{a}$ and move to the next grid point.
4. Otherwise, use a nonlinear solver to find a^* satisfying

$$u_c(Ra_i + we_j - a^*) = \beta R \sum_{e' \in E} Q(e'|e_j)\, u_c\big(Ra^* + we' - \widehat{a}^0(a^*, e')\big)$$

   Evaluate $\widehat{a}^0$ off the grid via piecewise linear interpolation. Set $a'(a_i, e_j) = a^*$.

PFI with Interpolation — Convergence

5. Check convergence: declare convergence at iteration n when

$$\max_{i,j} \left| a'_n(a_i, e_j) - \widehat{a}^{\,n}(a_i, e_j) \right| < \varepsilon$$

6. If not converged, update: $\widehat{a}^{\,n+1}(a_i, e_j) = a'_n(a_i, e_j)$ and repeat.

Bottleneck. Step 4 requires a nonlinear equation solver at every interior grid point. We now discuss two methods that avoid root-finding entirely.

Endogenous Grid Method (EGM)

Key idea. Grid over a' (next-period assets) instead of a. Given the Euler equation, consumption and current assets can be recovered analytically — no nonlinear solver needed.

Requirements: policy function weakly monotonic in the state.

Euler equation (at equality):

$$u_c\big(c(a, e_j)\big) = \beta R \sum_{e' \in E} Q(e'|e_j)\, u_c\big(c(a', e')\big)$$

We iterate on a guess $\widehat{c}^0(a, e)$ until convergence.

Endogenous Grid Method (EGM): Setup

Goal: compute policy $c(a, e)$.

Iteration object

$$\widehat{c}^{\,k}(a, e)$$

Grids

Current assets (policy grid):

$$A = \{a_1, \ldots, a_{N_a}\}$$

- Next-period assets (control grid):

$$A' = A$$

- Endogenous grid generated by EGM:

$$a^{EGM}(a', e)$$

EGM Algorithm

1. **Initialization**

   Guess consumption on the current asset grid

   $$\hat{c}^k(a_i, e_j)$$

   (using the same grid for evaluating a'). Example initial guess $\widehat{c}^0(a_i, e_j) = ra_i + we_j$

2. **Euler RHS**

   $$B(a_i', e_j) = \beta R \sum_{e'} Q(e'|e_j)\, u_c(\widehat{c}^0(a_i', e'))$$

3. **Invert marginal utility**

   $$\widetilde{c}^0(a_i', e_j) = u_c^{-1}(B(a_i', e_j))$$

   CRRA: $\widetilde{c}^0(a_i', e_j) = [B(a_i', e_j)]^{-1/\gamma}$.

EGM Algorithm (continued)

4. **Endogenous current assets**. From $c + a' = Ra + we$

$$a_i^{EGM}(e_j) = \frac{\widetilde{c}^0(a_i', e_j) + a_i' - we_j}{R}$$

5. **Interpolate to original grid**. EGM generates policy points $(a_n^{EGM}(e_j),\ \widetilde{c}^0(a_n', e_j))$.
   $\rightarrow$ If $a_i > a_0^{EGM}(e_j)$ (constraint does not bind). Find n s.t. $a_n^{EGM}(e_j) \leq a_i \leq a_{n+1}^{EGM}(e_j)$

$$\widehat{c}^1(a_i, e_j) = \widetilde{c}^0(a_n', e_j) + \frac{a_i - a_n^{EGM}(e_j)}{a_{n+1}^{EGM}(e_j) - a_n^{EGM}(e_j)} \left(\widetilde{c}^0(a_{n+1}', e_j) - \widetilde{c}^0(a_n', e_j) \right)$$

   $\rightarrow$ If $a_i < a_0^{EGM}(e_j)$ (constraint binds):

$$\widehat{c}^1(a_i, e_j) = Ra_i + we_j - \underline{a}$$

6. **Convergence**. Else update and repeat.

$$\max_{i,j} |\widehat{c}^1(a_i, e_j) - \widehat{c}^0(a_i, e_j)| < \varepsilon$$

Envelope Condition Method (ECM)

Key idea. Use the envelope condition $V_a(a, e_j) = R\, u_c(c)$ to extract the policy from the derivative of the value function — again without a nonlinear solver.

1. Construct two grids on a: $\mathcal{A}^V$ (for the value function) and $\mathcal{A}^D$ (for derivatives). Represent $\widehat{V}^0(a, e_j) = \sum_k w^0_{kj} B_k(a)$ with piecewise linear splines B_k.
2. Compute $\widehat{V}'_0(a_i, e_j) = \sum_k w^0_{kj} B'_k(a_i)$ on $\mathcal{A}^D$ (need a separate grid because the spline is not differentiable at its own knots).
3. Recover the policy from the envelope condition:

$$\widehat{a}^0(a_i, e_j) = Ra_i + we_j - u_c^{-1}\left(\frac{\widehat{V}'_0(a_i, e_j)}{R}\right)$$

ECM: Value Function Update and Convergence

4. Update the value function on $\mathcal{A}^V$:

$$\widehat{V}^{\,1}(a_k, e_j) = u\big(Ra_k + we_j - \widehat{a}^{\,0}(a_k, e_j)\big) + \beta \sum_{e' \in E} Q(e'|e_j)\, \widehat{V}^{\,0}\big(\widehat{a}^{\,0}(a_k, e_j),\, e'\big)$$

   Use interpolation to evaluate $\widehat{a}^{\,0}$ on $\mathcal{A}^V$ (different from $\mathcal{A}^D$).
5. Enforce the borrowing constraint: if $\widehat{a}^{\,0}(a_k, e_j) < \underline{a}$, set $\widehat{a}^{\,0}(a_k, e_j) = \underline{a}$.
6. Stop if

$$\max_{k,j} \left| \widehat{V}^{\,1}(a_k, e_j) - \widehat{V}^{\,0}(a_k, e_j) \right| < \varepsilon$$

Extensions to Richer Problems

Multiple continuous states. Two-asset household problems (liquid + illiquid) with convex adjustment costs. Nested EGM applies one endogenous-grid inversion per asset dimension.

- Auclert, Bardóczy, Rognlie & Straub (2021, *Ecma*), Appendix E

Non-convexities and kinks. When the value function is non-smooth or non-concave (e.g. fixed adjustment costs), the standard EGM can produce non-monotone endogenous grids. Requires detecting and discarding dominated segments ("upper envelope" step).

- Fella (2014, *RED*); Druedahl & Jørgensen (2017, *JEDC*)

Discrete-continuous choice. Agent makes both a discrete decision $d \in \{0, 1\}$ (e.g. rebalance or not) and a continuous savings choice. Run EGM separately per discrete branch, combine via upper envelope or Gumbel smoothing.

- Iskhakov, Jørgensen, Rust & Schjerning (2017, *QE*) — DCEGM

We turn to the discrete-choice smoothing idea next.

Roadmap

The Household Problem

Faster Iterations: HPI and OPI

Euler equation methods with interpolation: PFI, EGM, ECM

Extreme Value Shocks and the Gumbel Max Trick

Summary and Forward Links

Discrete Choices — The Problem

Augment the household problem with a discrete choice $d \in D = \{0, 1, \ldots, |D| - 1\}$. The value function becomes:

$$V(a, e) \;=\; \max_{d \in D} \; V^d(a, e)$$

where each branch V^d solves a continuous savings problem whose payoff depends on d.

Example: extensive-margin labor supply. The household chooses whether to work ($d = 1$) or stay home ($d = 0$):

$$V^0(a, e) = \max_{a'} \left\{ u(Ra + h - a') + \beta \textstyle\sum_{e'} V(a', e')\, Q(e, e') \right\}$$

$$V^1(a, e) = \max_{a'} \left\{ u(Ra + we - a') - \phi + \beta \textstyle\sum_{e'} V(a', e')\, Q(e, e') \right\}$$

where h is home production and ϕ is a disutility of work. No new state variable is needed — the two branches differ only in the flow payoff.

Discrete Choices — The Problem (continued)

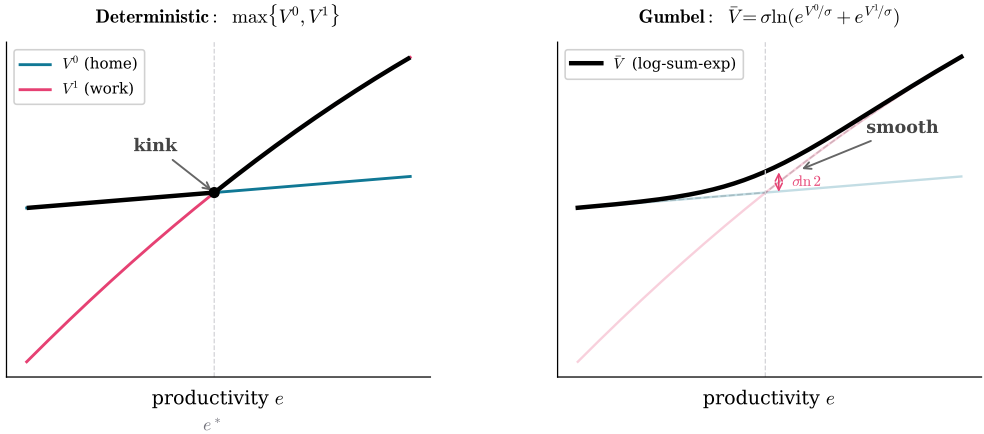
Two problems the discrete choice creates:

1. Discontinuous policies. The decision rule $d^*(a, e) = \mathbf{1}\{V^1(a, e) \geq V^0(a, e)\}$ is a step function. On a discrete grid, its first-order response to any shock is zero unless a grid point sits exactly at the threshold.

2. Kinks propagate backward. The discontinuity in the policy at t creates a kink in V_t. Through the continuation value, this kink propagates into $V_{t-1}, V_{t-2}, \ldots$ EGM and ECM require a differentiable continuation value; the kink breaks both methods.

Solution. Add iid preference shocks ε^d to each discrete option. With the right distributional assumption, the continuation value becomes smooth and policies become smooth choice *probabilities*.

The Kink and Its Cure — Illustration



Left: $\max\{V^0, V^1\}$ has a kink at the switching threshold a^*. The kink propagates backward through the continuation value and breaks EGM/ECM.

Right: The log-sum-exp rounds off the kink. $\bar{V}$ is everywhere differentiable ($\bar{V} \geq \max\{V^0, V^1\}$ everywhere, with gap $= \sigma \ln 2$ at the crossing). Fast algorithms apply.

The Augmented Problem

Each discrete option d receives an iid taste shock ε^d with scale parameter $\sigma > 0$:

$$V(a, e, \boldsymbol{\varepsilon}) \;=\; \max_{d \in D} \left\{ V^d(a, e) + \sigma \varepsilon^d \right\}$$

The **ex-ante value function** integrates out the shocks:

$$\bar{V}(a, e) \;=\; \mathbb{E}_{\boldsymbol{\varepsilon}} \left[\max_{d \in D} \left\{ V^d(a, e) + \sigma \varepsilon^d \right\} \right]$$

We need a distribution for ε^d such that $\bar{V}$ and the choice probabilities $p^d(a, e)$ have **closed forms**, $\bar{V}$ is **smooth**, and $\sigma \to 0$ recovers the deterministic problem. The Gumbel distribution delivers all three.

Why Add These Shocks?

1. Computational (primary). The shocks smooth out the kinks created by the discrete choice.

- $\bar{V}$ is differentiable $\Rightarrow$ EGM, ECM, and perturbation methods apply to each branch.
- Transition matrix $\mathbf{\Lambda}$ is everywhere positive $\Rightarrow$ sequence-space Jacobians are well-defined.
- Without smoothing, none of the fast algorithms from the previous section work.

2. Substantive (secondary). Without taste shocks, the model predicts that *all* households in state (a, e) make identical choices. In the data they do not — reflecting health, family structure, information, or other dimensions the model omits. The shocks turn decision rules into decision probabilities that can be matched to empirical choice frequencies.

Caveat. This interpretation should not be pushed too far. The iid Gumbel assumption imposes a specific functional form (logit, IIA) chosen for tractability, not because it is a credible model of the missing heterogeneity.

The Gumbel Distribution

Definition. $Z \sim G(\mu, \sigma)$ has CDF $F(z) = \exp\!\big(-e^{-(z-\mu)/\sigma}\big)$, with location $\mu \in \mathbb{R}$ and scale $\sigma > 0$.

$\mathbb{E}[Z] = \mu + \sigma\gamma_{\mathrm{EM}}$, where $\gamma_{\mathrm{EM}} \approx 0.5772$ is the Euler–Mascheroni constant. $\mathrm{Var}(Z) = \pi^2\sigma^2/6$.

Key property: stability of the maximum (S&S Lemma 5.3.2; Mukoyama, 2019).
Let $\varepsilon_1, \ldots, \varepsilon_n \overset{\text{iid}}{\sim} G(0, \sigma)$ and $V_1, \ldots, V_n \in \mathbb{R}$. Then:

$$\max_{1 \le i \le n} \,(V_i + \varepsilon_i) \;\sim\; G\!\left(\sigma \ln \sum_{i=1}^{n} e^{V_i/\sigma},\;\; \sigma\right)$$

Taking expectations:

$$\boxed{\mathbb{E}\left[\max_{1 \le i \le n} \,(V_i + \varepsilon_i)\right] \;=\; \sigma \ln \sum_{i=1}^{n} e^{V_i/\sigma}}$$

(up to the additive constant $\sigma\gamma_{\mathrm{EM}}$, which cancels in differences).
This is the **log-sum-exp** formula: a smooth, everywhere-differentiable approximation to $\max$.

Choice Probabilities and the $\sigma \to 0$ Limit

Choice probabilities (multinomial logit). Differentiating $\bar{V}$ with respect to V_i:

$$p_i \;\equiv\; \Pr\big(i = \arg\max_d \{V_d + \varepsilon_d\}\big) \;=\; \frac{e^{V_i/\sigma}}{\sum_{j=1}^{n} e^{V_j/\sigma}}$$

Limiting behavior (Mukoyama, 2019, Result 3):

- As $\sigma \to 0$: $p_i \to \mathbf{1}\{V_i > V_j \;\forall\, j \neq i\}$ and $\bar{V} \to \max_i V_i$.
  Recovers the deterministic (step-function) decision rule.
- As $\sigma \to \infty$: $p_i \to 1/n$ for all i. All options equally likely.

In practice, σ controls the **degree of smoothing**: small σ keeps the model close to the deterministic benchmark while ensuring differentiability.

Smoothing the Bellman Operator

Setup. Finite action set $\mathcal{A} = \{a_1, \ldots, a_k\}$. Each action returns a reward plus an iid Gumbel shock:

$$r(e', \varepsilon', a') \;=\; r(e', a') + \sigma\varepsilon'(a'), \qquad \varepsilon'(a') \overset{\text{iid}}{\sim} G(0,1)$$

(Equivalently, $\sigma\varepsilon'(a') \sim G(0,\sigma)$ as before; factoring out σ simplifies the operator.)

Define the **expected value function**: $g(e,a) := \sum_{e'} \int v(e', \varepsilon')\, Q(e, e')\, \varphi(\varepsilon')\, d\varepsilon'$

Expected-value Bellman operator (S&S eq. 5.36). Applying the log-sum-exp formula:

$$(Rg)(e,a) \;=\; \sum_{e'} \sigma \ln\left[\sum_{a'} \exp\left(\frac{r(e',a') + \beta\, g(e',a')}{\sigma}\right)\right] Q(e,e')$$

R has **no max operator** — it is smooth and differentiable in g everywhere.

Proposition 5.3.3 (S&S). R is a contraction of modulus β on $\mathbb{R}^{\mathcal{G}}$. The unique fixed point g^* recovers the optimal policy via $\sigma^*(e,\varepsilon) \in \arg\max_{a'} \{r(e,a') + \sigma\varepsilon(a') + \beta g^*(e,a')\}$.

Implied transition matrix. Under Gumbel shocks, every action is chosen with positive probability. The induced transition matrix $\mathbf{\Lambda}$ is everywhere positive $\Rightarrow$ Jacobians are well-defined.

Back to the Household Problem

Apply the Gumbel smoothing to our labor-supply example. Recall the two branches V^0 (home) and V^1 (work), each solving a continuous savings problem. The ex-ante value is:

$$\bar{V}(a,e) \;=\; \sigma \ln\!\big(e^{V^0(a,e)/\sigma} + e^{V^1(a,e)/\sigma}\big)$$

Each branch $d \in \{0,1\}$ now uses $\bar{V}$ as its continuation value:

$$V^d(a,e) \;=\; \max_{a' \geq \underline{a}} \Big\{ u\big(c^d(a,e,a')\big) + \beta \sum_{e' \in E} Q(e'|e)\, \bar{V}(a',e') \Big\}$$

where $c^0 = Ra + h - a'$ and $c^1 = Ra + we - a' - \phi$.

What has changed relative to the deterministic problem?

- The $\max$ over d is replaced by the smooth log-sum-exp $\Rightarrow$ $\bar{V}$ is differentiable in a.
- The continuation value in each branch is $\bar{V}$, not $\max\{V^0, V^1\}$ $\Rightarrow$ no kinks propagate.
- The Euler equation within each branch d has the standard form — only the continuation value has changed.

EGM with Gumbel Shocks

Since $\bar{V}(a', e')$ is smooth in a', we can run EGM **separately on each branch** d.

Fix branch d and income e_j. The Euler equation (at equality) reads:

$$u_c(c^d) \;=\; \beta R \sum_{e' \in E} Q(e'|e_j)\, \bar{V}_a(a', e')$$

where $\bar{V}_a$ is well-defined because the log-sum-exp is differentiable.

Algorithm (for each iteration):

1. For each (a'_i, e_j) on the grid, compute $\bar{V}_a(a'_i, e')$ from the current guess and form the RHS of the Euler equation.
2. Invert u_c to get $\widetilde{c}^d(a'_i, e_j)$ analytically (same as standard EGM).
3. Recover the endogenous grid: $a^*_i = (\widetilde{c}^d + a'_i - \text{income}^d(e_j))/R$.
4. Interpolate back onto the fixed grid; handle the borrowing constraint as before.
5. After solving both branches, update: $\bar{V}(a, e) = \sigma \ln(e^{V^0/\sigma} + e^{V^1/\sigma})$.

Cost per iteration: $O(|D| \cdot n_a n_e)$ — linear in the number of discrete options.

Discrete-Continuous Choice — DCEGM

DCEGM Algorithm (Iskhakov, Jørgensen, Rust & Schjerning, 2017):

1. For each discrete branch d, run EGM using the smooth log-sum-exp continuation value. The Gumbel integration renders this differentiable in a'.
2. Combine branches: $\bar{V}(a, e) = \sigma \ln(e^{V^0/\sigma} + e^{V^1/\sigma})$.
3. Iterate until convergence.

Key properties:

- Smooth continuation value $\Rightarrow$ EGM applies to each branch without modification.
- Choice probabilities $p^d(a, e)$ are differentiable $\Rightarrow$ sequence-space Jacobians exist.
- Cost: $O(|D| \cdot n_a n_e)$ per iteration.
- As $\sigma \to 0$: recovers the deterministic upper-envelope approach of Fella (2014).

Roadmap

The Household Problem

Faster Iterations: HPI and OPI

Euler equation methods with interpolation: PFI, EGM, ECM

Extreme Value Shocks and the Gumbel Max Trick

Summary and Forward Links

Summary: Algorithms for the Household Problem

Method	Key idea	Rate	Cost/sweep	Reference
VFI	Successive approx. on T	$O(\beta^k)$	$O(n_a^2 n_e)$	S&S §5.1.4
HPI	Newton on T; exact V_σ	Quadratic	$O((n_a n_e)^3)$	S&S §5.1.4
OPI	Approx. V_σ with T_σ^m	Tunable	$O(m\, n_a n_e)$	S&S §5.1.4
PFI + interp.	Euler eq. + root-finding	Outer loop	$O(n_a n_e)$	—
EGM	Euler eq. + endogenous grid	Outer loop	$O(n_a n_e)$	Carroll (2006)
ECM	Envelope cond. on V_a	Outer loop	$O(n_a n_e)$	M&M (2013)
Gumbel	Log-sum-exp; logit probs.	Outer loop	$O(\lvert D\rvert\, n_a n_e)$	S&S §5.3
DCEGM	EGM per branch + log-sum-exp	Outer loop	$O(\lvert D\rvert\, n_a n_e)$	Iskhakov+ (2017)

Top block: iteration on a discrete grid (policy $\sigma : \mathcal{A} \times E \to \mathcal{A}$).
Middle block: interpolation off the grid (continuous a', no grid search). Typically $10{-}30$ outer iterations.
Bottom block: discrete-continuous choice with smoothed Bellman operator.

Key Takeaways

1. Speed comes from exploiting structure.

- Concavity and monotonicity $\Rightarrow$ reduce the grid search in VFI/HPI.
- The Euler equation $\Rightarrow$ replace optimization with a functional equation.
- The endogenous grid trick $\Rightarrow$ eliminate root-finding entirely.

2. Interpolation is the bridge. Moving from grid policies to continuous policies via piecewise linear (or higher-order) interpolation is what makes EGM and ECM possible. The grid is a tool for representing the function, not a constraint on the solution.

3. Discrete choices need smoothing. Deterministic discrete choices create kinks that break fast algorithms and linearization. Gumbel taste shocks restore differentiability at a controlled cost — the scale parameter σ governs the trade-off between smoothness and proximity to the deterministic model.

Forward Links

Lecture 4: Hopenhayn (1992). Stationary industry equilibrium with entry, exit, and idiosyncratic productivity. Firm problem solved with the same DP toolkit.

Lecture 5: Buera & Shin (2011). Entrepreneurs face collateral constraints that limit borrowing to a fraction of wealth $\Rightarrow$ misallocation of capital across firms. Illustrates recursive competitive equilibrium (RCE) with a non-trivial stationary distribution. (See also Moll, 2014)

Lecture 6: Khan & Thomas (2008). Lumpy investment with fixed adjustment costs — firm-level non-convexities interact with aggregate shocks.

Lectures 7–9: HANK and sequence-space Jacobians.

- Auclert, Bardóczy, Rognlie & Straub (2021): the household block is solved with EGM; the Jacobian of aggregate consumption with respect to prices is computed by differentiating through the policy and distribution.