

Solution Methods for Microeconomic Dynamic Stochastic Optimization Problems

2026-03-11

Christopher D. Carroll¹

Note: The GitHub repo [SolvingMicroDSOPs](#) associated with this document contains python code that produces all results, from scratch, except for the last section on indirect inference. The numerical results have been confirmed by showing that the answers that the raw python produces correspond to the answers produced by tools available in the [Econ-ARK](#) toolkit, more specifically those in the [HARK](#) which has full [documentation](#). The MSM results at the end have have been superseded by tools in the [EstimatingMicroDSOPs](#) repo.

Abstract

These notes describe tools for solving microeconomic dynamic stochastic optimization problems, and show how to use those tools for efficiently estimating a standard life cycle consumption/saving model using microeconomic data. No attempt is made at a systematic overview of the many possible technical choices; instead, I present a specific set of methods that have proven useful in my own work (and explain why other popular methods, such as value function iteration, are a bad idea). Paired with these notes is Python code that solves the problems described in the text.

Keywords Dynamic Stochastic Optimization, Method of Simulated Moments, Structural Estimation, Indirect Inference

JEL codes E21, F41

Powered by [Econ-ARK](#)

PDF: <https://github.com/lloracc/SolvingMicroDSOPs/blob/master/SolvingMicroDSOPs.pdf>
Slides: <https://github.com/lloracc/SolvingMicroDSOPs/blob/master/SolvingMicroDSOPs-Slides.pdf>
Web: <https://lloracc.github.io/SolvingMicroDSOPs>
Code: <https://github.com/lloracc/SolvingMicroDSOPs/tree/master/Code>
Archive: <https://github.com/lloracc/SolvingMicroDSOPs>
(Contains LaTeX code for this document and software producing figures and results)

¹Carroll: Department of Economics, Johns Hopkins University, Baltimore, MD, ccarroll@jhu.edu

The notes were originally written for my Advanced Topics in Macroeconomic Theory class at Johns Hopkins University; instructors elsewhere are welcome to use them for teaching purposes. Relative to earlier drafts, this version incorporates several improvements related to new results in the paper “[Theoretical Foundations of Buffer Stock Saving](#)” (especially tools for approximating the consumption and value functions). Like the last major draft, it also builds on material in “The Method of Endogenous Gridpoints for Solving Dynamic Stochastic Optimization Problems” published in *Economics Letters*, available at <http://www.econ2.jhu.edu/people/ccarroll/EndogenousArchive.zip>, and by including sample code for a method of simulated moments estimation of the life cycle model *a la* [Gourinchas and Parker \(2002\)](#) and [Cagetti \(2003\)](#). Background derivations, notation, and related subjects are treated in my class notes for first year macro, available at <http://www.econ2.jhu.edu/people/ccarroll/public/lecturenotes/consumption>. I am grateful to several generations of graduate students in helping me to refine these notes, to Marc Chan for help in updating the text and software to be consistent with [Carroll \(2006\)](#), to Kiichi Tokunaka for drafting the section on structural estimation, to Damiano Sandri for exceptionally insightful help in revising and updating the method of simulated moments estimation section, and to Weifeng Wu and Metin Uyanik for revising to be consistent with the ‘method of moderation’ and other improvements. All errors are my own. This document can be cited as [Carroll \(2023a\)](#) in the references.

Contents

1	Introduction	1
2	The Problem	1
3	Normalization	2
4	Notation	4
4.1	Periods, Stages, Perches	4
4.2	Perches	5
4.3	Builders and Connectors	6
4.4	The Pile: Definition and Construction	8
4.4.1	Formal Definition	8
4.4.2	The Consumption-Only Pile	9
4.4.3	Homogeneous and Heterogeneous Piles	9
4.4.4	Backward Construction	9
5	The Usual Theory, and a Bit More Notation	10
6	Solving the Next-to-Last Period	11
6.1	A Direct Expression for c_{T-1}	11
6.2	Discretizing the Distribution	11
6.3	The Approximate Consumption and Value Functions	12
6.4	An Interpolated Consumption Function	13
6.5	Interpolating Expectations	13
6.6	Value Function versus First Order Condition	18
6.7	Transformation	21
6.8	The Natural Borrowing Constraint and the a_{T-1} Lower Bound	23
6.9	The Method of Endogenous Gridpoints ('EGM')	24
6.10	Improving the a Grid	27
6.11	Program Structure	31
6.12	Results	31
7	The Infinite Horizon	31
7.1	Convergence	33
8	The Method of Moderation	33
9	Multiple Control Variables and Modularity	34
9.1	The Joint Optimization Problem	35
9.2	Decomposing Into Modular Stages	36
9.2.1	The Discounting Stage (disc)	37
9.2.2	The Standalone Portfolio Problem	38

Contents

9.2.3	The portable Stage	38
9.2.4	Separating Shocks from Choices	39
9.2.5	Three Period Types	41
9.2.6	Connectors for Each Period Type	42
9.2.7	Numerical Solution	43
9.3	Implementation	43
9.4	Results With Multiple Controls	43
10	Structural Estimation	46
10.1	Life Cycle Model	46
10.2	Estimation	48
10.2.1	An Aside to Computing Sensitivity Measures	52
10.3	Results	52
11	Conclusion	56
A	SCF Data	57

1 Introduction

These notes provide a gentle-as-possible introduction to a particular set of solution tools for the canonical consumption-saving/portfolio allocation problem for a consumer facing uninsurable idiosyncratic risk to nonfinancial income (e.g., labor or transfer income), first without and then with optimal portfolio choice,¹ with detailed intuitive discussion of various mathematical and computational techniques that, together, accelerate the solution by many orders of magnitude. The problem is solved with and without liquidity constraints, and the infinite horizon solution is the limit of the finite horizon solution. After the basic consumption/saving problem with a deterministic interest rate is described and solved, an extension with portfolio choice between a riskless and a risky asset is also solved. Finally, a simple example shows how to use these methods (via the statistical ‘method of simulated moments’ (MSM for short)) to estimate structural parameters like the coefficient of relative risk aversion (*a la* Gourinchas and Parker (2002) and Cagetti (2003)).

2 The Problem

The usual analysis of dynamic stochastic programming problems packs a great many events (intertemporal choice, stochastic shocks, intertemporal returns, income growth, the taking of expectations, time discounting, and more) into a complex decision in which the agent makes an optimal choice simultaneously taking all these elements into account. For the dissection here, we will be careful to break down everything that happens into distinct operations so that each element can be scrutinized and understood in isolation.

We are interested in the behavior of a consumer who begins period t with a certain amount of ‘capital’

$$k_t \mathbf{U} \tag{1}$$

which immediately earns a return factor R_t . Simultaneously, the consumer receives noncapital income y_t , which is the product of ‘permanent income’ p_t and a transitory shock θ_t :

$$y_t = p_t \theta_t \mathbf{U} \tag{2}$$

whose expectation is 1 (that is, before realization of the transitory shock, the consumer’s expectation is that actual income will on average be equal to permanent income p_t).

The combination of the capital return and income defines the consumer’s ‘market resources’ (sometimes called ‘cash-on-hand,’ following Deaton (1992)):

$$m_t = k_t R_t + y_t, \mathbf{U} \tag{3}$$

¹See Merton (1969) and Samuelson (1969) for a solution to the problem of a consumer whose only risk is rate-of-return risk on a financial asset; the combined case (both financial and nonfinancial risk) is solved below, and much more closely resembles the case with only nonfinancial risk than it does the case with only financial risk.

3 Normalization

available to be spent on consumption $\mathbf{c}_t$ for a consumer subject to a liquidity constraint that requires $\mathbf{c}_t \leq \mathbf{m}_t$ (though we are not imposing such a constraint yet—see subsection 6.8). Finally we define

$$\{\text{eq:aLvl}\} \quad \mathbf{a}_t = \mathbf{m}_t - \mathbf{c}_t \mathbf{U} \quad (4)$$

mnemonically as ‘assets-after-all-actions-are-accomplished.’

The consumer’s goal is to maximize discounted utility from consumption over the rest of a lifetime ending at date T :

$$\{\text{eq:MaxProb}\} \quad \max \mathbb{E}_t \left[\sum_{n=0}^{T-t} \beta^n u(\mathbf{c}_{t+n}) \right] \cdot \mathbf{U} \quad (5)$$

Income evolves according to:

$$\{\text{eq:permincgrow}\} \quad \begin{aligned} \mathbf{p}_{t+1} &= \mathcal{G}_{t+1} \mathbf{p}_t && \text{-- permanent labor income dynamics} \\ \log \theta_{t+n} &\sim \mathcal{N}(-\text{std}_\theta^2/2, \text{std}_\theta^2) && \text{-- lognormal transitory shocks } \forall n > 0. \mathbf{U} \end{aligned} \quad (6)$$

Equation (6) indicates that we are allowing for a predictable average profile of income growth over the lifetime $\{\mathcal{G}\}_0^T$ (to capture typical career wage paths, pension arrangements, etc).² Finally, the utility function is of the Constant Relative Risk Aversion (CRRA) form, $u(\bullet) = \bullet^{1-\rho}/(1-\rho)$.

It is well known that this problem can be rewritten in recursive (Bellman) form:

$$\{\text{eq:vrecurse}\} \quad v_t(\mathbf{m}_t, \mathbf{p}_t) = \max_{\mathbf{c}} u(\mathbf{c}) + \beta \mathbb{E}_t[v_{t+1}(\mathbf{m}_{t+1}, \mathbf{p}_{t+1})] \mathbf{U} \quad (7)$$

subject to the Dynamic Budget Constraint (DBC) defined by equation (3), and to the dynamic process for income defined in (6) and to a transition equation that defines next period’s initial capital as this period’s end-of-period assets:

$$\{\text{eq:transitionstate}\} \quad \mathbf{k}_{t+1} = \mathbf{a}_t \cdot \mathbf{U} \quad (8)$$

3 Normalization

{sec:normalization}

The single most powerful method for speeding the solution of such models is to redefine the problem in a way that reduces the number of state variables (if at all possible). In the consumption context, the obvious idea is to see whether the problem can be rewritten in terms of the ratio of various variables to permanent noncapital (‘labor’) income $\mathbf{p}_t$ (henceforth for brevity, ‘permanent income.’)

In the last period of life T , there is no future value, $\mathbf{v}_{T+1} = 0$ (boldface $\mathbf{v}$ denotes the value function in *levels*; the nonbold normalized counterpart v is defined below), so the

²For simplicity, this equation assumes no shocks to permanent income (though they are trivial to add). Empirically, such shocks are large and must be incorporated into any serious model, but they clutter the exposition without adding much intuition; they are introduced in the final section when we match the model to data. For a full treatment including permanent shocks, see [Carroll \(2023b\)](#).

3 Normalization

optimal plan is to consume everything:

$$\mathbf{v}_T(\mathbf{m}_T, \mathbf{p}_T) = \frac{\mathbf{m}_T^{1-\rho}}{1-\rho} \cdot \mathbf{U} \quad (9)$$

Now define nonbold variables as the bold variable divided by the level of permanent income in the same period, so that, for example, $m_T = \mathbf{m}_T / \mathbf{p}_T$; and define $v_T(m_T) = u(m_T)$.³ For our CRRA utility function, $u(xy) = x^{1-\rho}u(y)$, so (9) can be rewritten as

$$\begin{aligned} \mathbf{v}_T(\mathbf{m}_T, \mathbf{p}_T) &= \mathbf{p}_T^{1-\rho} \frac{m_T^{1-\rho}}{1-\rho} \\ &= \mathbf{p}_{T-1}^{1-\rho} \mathcal{G}_T^{1-\rho} v_T(m_T) \cdot \mathbf{U} \end{aligned} \quad (10)$$

Because we are dividing $t+1$ level variables by $\mathbf{p}_{t+1} = \mathcal{G}_{t+1}\mathbf{p}_t$, a normalized return factor emerges:

$$\mathcal{R}_{t+1} \equiv \mathbf{R} / \mathcal{G}_{t+1} \cdot \mathbf{U} \quad (11)$$

(We treat $\mathbf{R}$ as time-invariant and drop the period subscript that appeared in (3).)

Now define a new optimization problem:

$$\begin{aligned} v_t(m_t) &= \max_c u(c) + \beta \mathbb{E}_t[\mathcal{G}_{t+1}^{1-\rho} v_{t+1}(m_{t+1})] \\ \text{s.t.} \\ a_t &= m_t - c_t \\ k_{t+1} &= a_t \\ m_{t+1} &= \mathcal{R}_{t+1} k_{t+1} + \theta_{t+1} \cdot \mathbf{U} \end{aligned} \quad (12)$$

Then it is easy to see that for $t = T-1$, we can write boldface (nonnormalized) $\mathbf{v}$ as a function of v (normalized value) and permanent income:

$$\mathbf{v}_t(\mathbf{m}_t, \mathbf{p}_t) = \mathbf{p}_t^{1-\rho} v_t(m_t), \mathbf{U} \quad (13)$$

and so on back to all earlier periods (by backward induction: if the factorization holds at $t+1$, substituting into the Bellman equation at t and using the homogeneity of CRRA utility yields the same factorization at t). Hence, if we solve the problem (12) which has only a single state variable m_t , we can obtain the levels of the value function from (13), and of consumption and all other variables from the corresponding permanent-income-normalized solution objects by multiplying each by $\mathbf{p}_t$, e.g.

$$c_t(\mathbf{m}_t, \mathbf{p}_t) = \mathbf{p}_t c_t(\overbrace{\mathbf{m}_t / \mathbf{p}_t}^{m_t}).$$

³Nonbold value is bold value divided by $\mathbf{p}^{1-\rho}$ rather than $\mathbf{p}$.

We have thus reduced the problem from two continuous state variables to one (and thereby enormously simplified its solution).⁴

For future reference it will be useful to write (12) in the traditional way, by substituting a_t and k_{t+1} into m_{t+1} :

$$v_t(m_t) = \max_c u(c) + \beta \mathbb{E}_t[\mathcal{G}_{t+1}^{1-\rho} v_{t+1}(\overbrace{(m_t - c)(R/\mathcal{G}_{t+1}) + \theta_{t+1}}^{m_{t+1}})]. \quad (14)$$

4 Notation

4.1 Periods, Stages, Perches

The problem so far assumes that the agent has only one decision. But agents often have multiple choices per period—for example, a consumption decision, a labor supply choice, and a choice of what proportion ς of capital k to invest in a risky vehicle. We identify each stage type in two ways: by a **short-name** (a textual name, given when the stage is first introduced) and by a **control-name** (the stage’s control variable, if any). For example, a labor supply stage might have short-name **labor** and control-name ℓ ; a consumption stage has control-name c . A stage list that constitutes a period may therefore be written by short-name, e.g. `[labor, cons-noshocks, disc]`, or by control-name, e.g. `[\ell, c, \beta]`. A modeler might want to explore whether the order in which the stages are solved makes any difference, either to the substantive results or to aspects of the computational solution like speed and accuracy; with this scheme they do so merely by changing the order in which the stages are listed in the specification of the period.

If, as in section 2, we hard-wire into the solution code for each stage an assumption that its successor stage will be something in particular (say, the consumption stage assumes that the portfolio choice is next), then if we want to change the order of the stages (say, labor supply after consumption, followed by portfolio choice), we will need to re-hard-wire each of the stages to know new things about its new successor (for example, the specifics of the distribution of the rate of return on the risky asset must be known by whatever stage precedes the portfolio choice stage).

The cardinal insight of Bellman (1957) is that *everything that matters* for the solution to the current problem is encoded in a ‘continuation-value function.’

Using that insight, we describe here a framework for isolating the stage problems within a period from each other, and the period from its successors or predecessors in any other period. The advantage of this isolation is that each stage problem becomes a self-contained *module*: Its internal logic—the computation it performs on value functions—is defined independently of where it sits in the sequence of stages.

⁴In the modular framework of section 4, normalization can be viewed as a transformation on the stage description itself: each stage’s state space, transition functions, and value functions are rewritten from level variables $(\mathbf{m}, \mathbf{p})$ to the permanent-income-normalized counterparts (m) , reducing the dimensionality of every stage uniformly rather than applying a one-off simplification to the composite Bellman equation.

Modularity is valuable because it makes exploring such alternative model structures *cheap*. Using control-name indexing (e.g., the consumption stage by c), after considering the stage-order $[\ell, c, \varsigma]$, the modeler can reorder the stages to consider, say, the order $[\ell, \varsigma, c]$ *without rewriting any of the code that solves each individual stage*.⁵ What must change are the *transitions*—the mappings that connect the end of one stage to the beginning of the next—which must be rewired to reflect the new ordering and its implied information structure. The stage-level code itself remains untouched.

4.2 Perches

The key is to distinguish, within each stage’s Bellman problem, three viewpoints or ‘perches’ (we use that word to emphasize that the perch does not *do* anything: It is merely a collection of mathematical and computational functions and objects).

1. **Arrival:** Incoming state variables (e.g., k) are known, but any shocks associated with the stage have not been realized and decision(s) have not yet been made
2. **Decision:** The agent solves the decision problem for the period
3. **Continuation:** After all decisions have been made, their consequences are measured by evaluation of the continuing-value function at the values of the ‘outgoing’ state variables (sometimes called ‘post-state’ variables)

This framework is deliberately silent about *when* shocks (if any) occur within a stage. Shocks may be realized between the Arrival and Decision perches, or between the Decision and Continuation perches—the choice is part of the stage’s specification, not a framework-level constraint. In a consumption problem, the usual assumption is that income shocks have been realized before the spending decision is made, so that the consumer knows their resources when they decide how much to spend; here, shocks fall between Arrival and Decision ($k \xrightarrow{\text{shocks}} m \xrightarrow{\text{decision}} a$). But in a portfolio choice problem, the portfolio share decision must be made *before* the return shock is realized; there, shocks fall between Decision and Continuation ($k \xrightarrow{\text{decision}} (k, \varsigma) \xrightarrow{\text{shocks}} \tilde{m}$). The perch structure accommodates both patterns: the Arrival-to-Decision relationship encodes pre-decision uncertainty, while the Decision-to-Continuation relationship encodes post-decision uncertainty. A stage with no shocks at all (such as **cons-noshocks** or **disc**) simply sets $v_{\prec} = v_{\sim}$ or $v_{\sim} = f(v_{\prec})$ with no intervening expectation.

This **cons-with-shocks** stage corresponds to the consumption problem defined above.

The table illustrates notation we can use when analyzing the problem from a context ‘inside’ a particular stage of a specific period. We require that no variable can have more than one meaning or interpretation inside a period, and we prohibit any reference to values of any variables or functions or other model objects from outside the stage (or period). This is why we use different letters, k and a , to represent liquid resources

⁵The beginning-of-stage and end-of-stage value functions for each stage must be defined over compatible state variables, so that the output of any stage can serve as the input to any other; see the discussion in section 9.

{perches}

Table 1 Perches within the cons-with-shocks stage

Perch	Indicator	State	value functions	Explanation
Arrival	$\prec$	k	$v_{\prec} = \mathbb{E}_{\prec}[v_{\sim}]$	value at entry to stage
Decision(s)	$\sim$	m	$v_{\sim} = \max_c u(c) + v_{\succ}$	value of stage-decision
Continuation	$\succ$	a	$v_{\succ}$	value at exit

before and after the consumption decision, even if ultimately this period’s continuation value of a will transmute into the next period’s initial k . (Both k and a are “k-type” variables—investable capital before returns are realized; the distinction from “m-type” variables like m , which represent spendable resources after returns, is formalized below.)

In contrast, items like value functions v or expectations operators $\mathbb{E}$ have different meanings at different perches; we capture this using a subscript like $\prec$. The fact that all functions in a perch depend on the same state variables (shown in the second column) allows us to write those functions without specifying their arguments.

4.3 Builders and Connectors

{subsubstrabuilders}

Modularity requires that objects inside a period have no direct access to objects from any other period. This means that we must endow a stage or a period, at the time of its creation, with its end-of-stage-or-period value function $v_{\succ}$.

For example, in a model in which every period contains only the single-stage consumption problem above, the continuation value function for the last (and only) stage at t will need to be ‘connected’ to the arrival value function in $(t+1)$, which of necessity requires us to use t -related notation. Concretely, if we designate the end-of-*period* value function as $v_{\succ(t)}$ (which is defined as the continuation value function from the last stage in the period), we use the notation

{eq:trns-single-prd}

$$v_{\succ(t)} := \beta v_{\prec(t+1)}, \quad \mathbf{U} \quad (15)$$

to describe what the builder does when constructing the predecessor to period $t + 1$. The use of the ‘:=’ signals creation: the left-hand side is *brought into existence* by the builder.

Expectation operators across perches. The subscript on an expectation operator $\mathbb{E}$ indicates the information set at that perch: $\mathbb{E}_{\prec}$ conditions on the Arrival state but not on any shocks realized between Arrival and Decision. For adjacent perches at a period boundary— $\mathbb{E}_{\succ(t)}$ (Continuation of period t) and $\mathbb{E}_{\prec(t+1)}$ (Arrival of period $t + 1$)—the information sets are identical, so the two operators are mathematically interchangeable; the notational distinction reflects viewpoint (looking backward from t ’s exit versus forward from $(t + 1)$ ’s entry).

Tying two adjacent stages or periods together also requires that we define a Connector, $\mathcal{C}$, which specifies the relationship between the continuation-perch state variable(s) of the predecessor to the arrival-perch state variable(s) of the successor. Again concretely, for two successive periods each of which contains only a single consumption stage like the one described above, the Connector would look like:

$$\mathcal{C}(a \leftrightarrow k). \quad (16)$$

State-variable types. Every state variable in the framework belongs to exactly one of two *types*:

- **k-type** (capital): investable assets *before* returns and income are realized. The defining characteristic is that a k-type variable must pass through a returns-and-shocks process before it becomes spendable. Examples: k (beginning-of-period capital) and a (end-of-period assets after consumption, awaiting next period's returns).
- **m-type** (market resources): spendable resources *after* returns and income shocks have been realized—what Deaton (1992) calls “cash-on-hand.” The defining characteristic is that an m-type variable is immediately available for spending or allocation decisions. Examples: m (market resources at the Decision perch) and $\tilde{m}$ (post-shock market resources within a period).

Well-typed connectors. A Connector is a pure rename: it asserts that two variables from adjacent stages or periods are different names for the same economic object. A Connector is *well-typed* if and only if the two variables it connects share the same type. Thus:

- $\mathcal{C}(a \leftrightarrow k)$ is well-typed: both are k-type.
- $\mathcal{C}(\tilde{m} \leftrightarrow m)$ is well-typed: both are m-type.
- $\mathcal{C}(m \leftrightarrow k)$ would be *illegal*: m is m-type and k is k-type. Market resources and investable capital are not merely different names for the same quantity—converting between them requires a substantive transformation (the realization of returns and income). Attempting such a connector is a modeling error: it would silently skip the returns process, treating post-return resources as if they were pre-return capital.

The type constraint prevents a common class of modeling errors in which a modeler inadvertently connects a post-return variable to a pre-return slot (or vice versa), producing economically nonsensical dynamics. By requiring type agreement, the framework ensures that every transition between k-type and m-type passes through an explicit returns-and-shocks stage.

Annotating stage tables with types. When reading the perch tables in this document and in section 9, the arrival and continuation state types can be read off directly:

- **cons-with-shocks:** arrival k (k-type), continuation a (k-type).
- **cons-noshocks:** arrival m (m-type), continuation a (k-type).
- **portable:** arrival k (k-type), continuation $\tilde{m}$ (m-type).
- **disc:** arrival and continuation inherit the type of the predecessor's exit state (passthrough).

A within-period Connector between two stages must respect the same type constraint: the predecessor stage's continuation type must match the successor stage's arrival type.

Namespace uniqueness and the $\sim$ decoration. A strict rule of the framework is that **each variable name refers to exactly one value within a given period**. This *namespace uniqueness constraint* ensures that the mapping between mathematical symbols and computational variables (e.g., Python names) is well-defined: if a symbol appears in an equation, there is no ambiguity about which quantity it denotes.

The constraint has a practical consequence when multiple stages within the same period produce variables of the same type. For example, in the portable-consnoshocks period (section 9.2.5), the **portable** stage produces post-shock market resources, and the **cons-noshocks** stage receives market resources as its Arrival state. Both quantities are m-type, but they arise at different points in the period and generally take different values. To preserve namespace uniqueness, the earlier occurrence receives a decoration: we write $\tilde{m}$ for the **portable** stage's output and m for the **cons-noshocks** stage's input. The within-period Connector $\mathcal{C}(\tilde{m} \leftrightarrow m)$ then renames one to the other.

More generally, when a stage's continuation state has the same type as another stage's arrival state within the same period, the later or secondary occurrence may receive a decoration ($\hat{\cdot}$, etc.) to maintain uniqueness. The decoration signals that the variable is of the same economic type as its undecorated counterpart but occupies a distinct position in the period's computation. This convention makes it straightforward to translate the mathematical notation into code: each decorated symbol maps to a unique Python variable name (e.g., `mNrmCheck` for $\tilde{m}$, `mNrm` for m), and no variable is overwritten or reused within a single period's scope.

4.4 The Pile: Definition and Construction

4.4.1 Formal Definition

The pile P is an ordered sequence of solved periods interleaved with the Connectors that join them. Formally, a pile with N periods is:

$$P = \{ p_1, \mathcal{C}_{1,2}, p_2, \mathcal{C}_{2,3}, \dots, \mathcal{C}_{N-1,N}, p_N \}, \quad (17)$$

where each $\mathbf{p}_t$ is a solved period (containing its stage list, value functions, and policy functions) and each $\mathcal{C}_{t,t+1}$ is a between-period Connector satisfying the type constraint from section 4.3: the predecessor's exit state and the successor's entry state must share the same type (k-type or m-type).

The pile is **well-formed** if and only if every adjacent $(\mathbf{p}_t, \mathcal{C}_{t,t+1}, \mathbf{p}_{t+1})$ triple satisfies:

1. **Type consistency.** The continuation-state type of $\mathbf{p}_t$'s last stage matches the arrival-state type of $\mathbf{p}_{t+1}$'s first stage, and the Connector is a valid rename between them.
2. **Value-function compatibility.** The Connector and builder together produce a continuation value function for $\mathbf{p}_t$ from the arrival value function of $\mathbf{p}_{t+1}$.

4.4.2 The Consumption-Only Pile

For the consumption-only problem of sections 2–3, each period contains the stage list [shocks-only, $\mathcal{C}(\tilde{m} \leftrightarrow m)$, cons-noshocks, disc] (or, in the single-stage formulation, just [cons-with-shocks, disc]). The between-period Connector is $\mathcal{C}(a \leftrightarrow k)$ (both k-type). A three-period consumption-only pile is therefore:

$$\mathbf{P}_{\text{cons}} = \{ \mathbf{p}_{T-2}, \mathcal{C}(a \leftrightarrow k), \mathbf{p}_{T-1}, \mathcal{C}(a \leftrightarrow k), \mathbf{p}_T \}.$$

4.4.3 Homogeneous and Heterogeneous Piles

A pile is **homogeneous** if every period has the same stage list, the same parameter values, and the same between-period Connector. The infinite-horizon consumption problem (section 7) converges to a homogeneous pile: every period is identical, and the converged solution is a fixed point of the backward builder $\mathbf{B}_{\text{prd}}^{\leftarrow}$.

A pile is **heterogeneous** if periods differ in their stage lists, parameter values, or Connectors. A life cycle model is typically heterogeneous: mortality risk, income profiles, and available choices (e.g., whether portfolio optimization is active) may vary by age. For example, a life cycle model might use [portable($\bar{\epsilon} = \perp$), $\mathcal{C}$, cons-noshocks, disc] for working-age periods (with active portfolio choice) and [shocks-only, $\mathcal{C}$, cons-noshocks, disc] for retirement periods (no portfolio choice).

4.4.4 Backward Construction

We call the structure in which accumulating periods are stored the pile $\mathbf{P}$. Once backward induction is complete, the pile holds the full definition of the solved problem.

The process of building the pile is straightforward. We start from the terminal period (section 3: $v_T(m) = u(m)$, consume everything), so initially $\mathbf{P} = \{\mathbf{p}_T\}$. We denote the backward builder as $\mathbf{B}_{\text{prd}}^{\leftarrow}$. Applying $\mathbf{B}_{\text{prd}}^{\leftarrow}$ to the existing $\mathbf{P}$ creates a new Connector and then the new period's solution, yielding $\mathbf{P} = \{\mathbf{p}_{T-1}, \mathcal{C}_{T-1,T}, \mathbf{p}_T\}$. Each subsequent application of $\mathbf{B}_{\text{prd}}^{\leftarrow}$ prepends one more $(\mathbf{p}_t, \mathcal{C}_{t,t+1})$ pair, growing the pile from right to left. Section 6 details the construction of $\mathbf{p}_{T-1}$; with multiple stages or control variables the structure generalizes as in section 9.

5 The Usual Theory, and a Bit More Notation

For reference and to illustrate our new notation, we will now derive the Euler equation and other standard results for the problem described above. Since we can write value as of the end of the consumption stage as a function of a_t :

$$v_{\succ(t)}(a_t) = \beta \mathbb{E}_{\prec(t+1)} [\mathcal{G}_{t+1}^{1-\rho} v_{\sim(t+1)}(\overbrace{(\mathbf{R}/\mathcal{G}_{t+1})a_t + \theta_{t+1}}^{m_{t+1}})], \quad (18)$$

Derivative notation convention. Throughout this document, a superscript ∂ on a function means its derivative with respect to the relevant state variable at that perch: $v_{\succ}^{\partial}(a) \equiv dv_{\succ}/da$, $v_{\sim}^{\partial}(m) \equiv dv_{\sim}/dm$. So, the first order condition for (14) with respect to a_t ⁶ is

$$\begin{aligned} u^{\partial}(m_t - a_t) &= v_{\succ(t)}^{\partial}(a_t) = \mathbb{E}_{\prec(t+1)} [\beta \mathcal{R}_{t+1} \mathcal{G}_{t+1}^{1-\rho} v_{\sim(t+1)}^{\partial}(m_{t+1})] \\ &= \mathbb{E}_{\prec(t+1)} [\beta \mathcal{R} \quad \mathcal{G}_{t+1}^{1-\rho} v_{\sim(t+1)}^{\partial}(m_{t+1})] \mathbf{U} \end{aligned} \quad (19)$$

which illustrates the derivative convention (e.g. v^{∂} is the derivative of v with respect to its argument).⁷ For functions of more than one argument, we append the variable name: $v^{\partial x}$ denotes the partial derivative of v with respect to x .

Because the **Envelope** theorem tells us that

$$v_{\sim(t)}^{\partial}(m_t) = \mathbb{E}_{\prec(t+1)} [\beta \mathcal{R} \mathcal{G}_{t+1}^{1-\rho} v_{\sim(t+1)}^{\partial}(m_{t+1})] \mathbf{U} \quad (20)$$

we can substitute the LHS of (20) for the RHS of (19) to get

$$u^{\partial}(c_t) = v_{\sim(t)}^{\partial}(m_t) \mathbf{U} \quad (21)$$

and rolling forward one period,

$$u^{\partial}(c_{t+1}) = v_{\sim(t+1)}^{\partial}(a_t \mathcal{R}_{t+1} + \theta_{t+1}) \mathbf{U} \quad (22)$$

so that substituting the LHS in equation (19) finally gives us the Euler equation for consumption:

$$u^{\partial}(c_t) = \mathbb{E}_{\succ(t)} [\beta \mathcal{R} \mathcal{G}_{t+1}^{1-\rho} u^{\partial}(c_{t+1})] \cdot \mathbf{U} \quad (23)$$

The derivation above used period-qualified subscripts (e.g., $v_{\succ(t)}$, $v_{\sim(t+1)}^{\partial}$) because the Euler equation relates objects across periods. We can now restate the problem (14) using the simpler within-stage notation, which drops the period qualifier:

$$v_{\sim}(m) = \max_c \quad u(c) + v_{\succ}(m - c) \mathbf{U} \quad (24)$$

⁶Since $c = m - a$, maximizing over c is equivalent to maximizing over a . Differentiating $u(m - a) + v_{\succ}(a)$ with respect to a gives $-u^{\partial}(c) + v_{\succ}^{\partial}(a) = 0$, which rearranges to $u^{\partial}(c) = v_{\succ}^{\partial}(a)$.

⁷The superscript ∂ plays the same role as the more common prime notation (v'), which we avoid because in dynamic programming contexts the prime conventionally denotes the *next-period* value of a variable (e.g., m' for next-period market resources), creating potential ambiguity.

whose first order condition with respect to c is

$$\{eq:upEqbetaOp\} \quad u^\partial(c) = v_\prec^\partial(m - c) \textcolor{red}{U} \quad (25)$$

which is mathematically equivalent to the usual Euler equation for consumption.

We will revert to this formulation when we reach subsection 6.9.

6 Solving the Next-to-Last Period

{ec:solving-the-next}

This section examines the second-to-last period in detail, illustrating a number of powerful techniques for speeding and improving its solution; in doing so it illustrates how the pile P (subsection 4.4) is built backward. We set $\mathcal{G}_t = 1$ here to reduce clutter.

6.1 A Direct Expression for c_{T-1}

direct-substitution}

If $t = T$, the second-to-last-period decision-perch problem is

$$\{eq:vCntrPrdTm1\} \quad v_{\sim(T-1)}(m) = \max_c u(c) + v_{\succ(t-1)}(\overbrace{m - c}^a) \textcolor{red}{U} \quad (26)$$

Using (0) $t = T$; (1) $v_{\sim(T)}(m) = u(m)$; (2) the definition of $u(m)$; and (3) the definition of the expectations operator,

$$\{eq:NumDefInt\} \quad v_{\prec(t)}(a) = \int_0^\infty \frac{(a\mathcal{R}_t + \vartheta)^{1-\rho}}{1-\rho} d\mathcal{F}(\vartheta) \textcolor{red}{U}, \quad (27)$$

where $\mathcal{F}(\theta)$ is the cumulative distribution function for θ , this maximization problem implicitly defines a ‘period-and-stage-local function’ c that yields optimal consumption in period $t - 1$ for any specific numerical level of resources like $m = 1.7$. The explicit statement of the problem is

$$\{eq:cFuncTm1\} \quad c_{\sim(T-1)}(m) = \arg \max_c u(c) + \beta \int_0^\infty \frac{((m - c)\mathcal{R}_t + \vartheta)^{1-\rho}}{1-\rho} d\mathcal{F}(\vartheta). \textcolor{red}{U} \quad (28)$$

But because there is no general analytical solution, for any given m we must use numerical tools to find the c that maximizes the expression. This is excruciatingly slow: for every candidate c , a definite integral over $(0, \infty)$ must be calculated numerically, and optimization is itself a costly operation, so the combination of the two is a double-whammy for slowdown.

6.2 Discretizing the Distribution

Our first speedup trick is therefore to construct a discrete approximation to the lognormal distribution that can be used in place of numerical integration. That is, we want to

approximate the expectation over θ of a function $g(\theta)$ by calculating its value at set of n_θ points θ_i , each of which has an associated probability weight w_i :

$$\begin{aligned}\mathbb{E}[g(\theta)] &= \int_{\underline{\theta}}^{\bar{\theta}} g(\vartheta) d\mathcal{F}(\vartheta) \\ &\approx \sum_{\theta=1}^n w_i g(\theta_i)\end{aligned}$$

(because adding n weighted values to each other is enormously faster than general-purpose numerical integration).

Such a procedure is called a ‘quadrature’ method of integration; [Tanaka and Toda \(2013\)](#) survey a number of options, but for our purposes we choose the one which is easiest to understand: An ‘equiprobable’ approximation (that is, one where each of the values of θ_i has an equal probability, equal to $1/n_\theta$).

We calculate such an n -point approximation as follows.

Define a set of points from $\sharp_0$ to $\sharp_{n_\theta}$ on the $[0, 1]$ interval as the elements of the set $\sharp = \{0, 1/n, 2/n, \dots, 1\}$.⁸ Call the inverse of the θ distribution $\mathcal{F}^{-1}$, and define the points $\sharp_i^{-1} = \mathcal{F}^{-1}(\sharp_i)$. Then the conditional mean of θ in each of the intervals numbered 1 to n is:

$$\theta_i \equiv \mathbb{E}[\theta | \sharp_{i-1}^{-1} \leq \theta < \sharp_i^{-1}] = \int_{\sharp_{i-1}^{-1}}^{\sharp_i^{-1}} \vartheta d\mathcal{F}(\vartheta), \text{ U} \quad (29)$$

and when the integral is evaluated numerically for each i the result is a set of values of θ that correspond to the mean value in each of the n intervals.

The method is illustrated in Figure 1. The solid continuous curve represents the “true” CDF $\mathcal{F}(\theta)$ for a lognormal distribution such that $\mathbb{E}[\theta] = 1$, $\sigma_\theta = 0.1$. The short vertical line segments represent the n_θ equiprobable values of θ_i which are used to approximate this distribution.⁹

The following notebook snippet constructs these points.

We now substitute our approximation (30) for $v_{\succ(t-1)}(a)$ in (26) which is simply the sum of n_θ numbers and is therefore easy to calculate (compared to the full-fledged numerical integration (27) that it replaces).

{eq:vDiscrete}

$$v_{\succ(t-1)}(a) = \beta \left(\frac{1}{n_\theta} \right) \sum_{i=1}^{n_\theta} \frac{(\mathcal{R}_t a + \theta_i)^{1-\rho}}{1-\rho} \text{ U} \quad (30)$$

6.3 The Approximate Consumption and Value Functions

Given any particular value of m , a numerical maximization tool can now find the c that solves (26) in a reasonable amount of time.

⁸These points define intervals that constitute a partition of the domain of $\mathcal{F}$.

⁹More sophisticated approximation methods exist (e.g. Gauss-Hermite quadrature; see [Kopecky and Suen \(2010\)](#) for a discussion of other alternatives), but the method described here is easy to understand, quick to calculate, and has additional advantages briefly described in the discussion of simulation below.

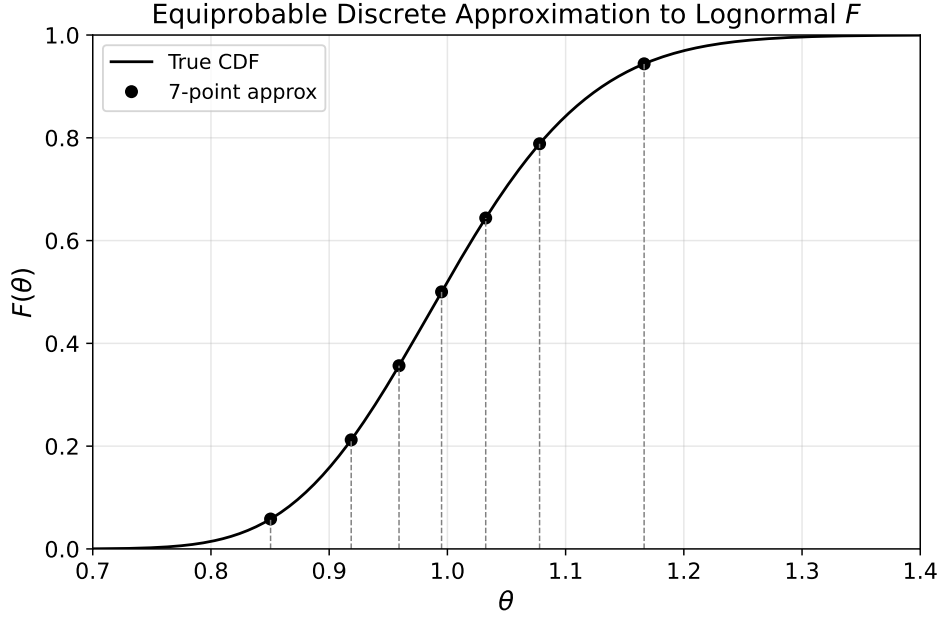


Figure 1 Equiprobable Discrete Approximation to Lognormal Distribution $\mathcal{F}$

The notebook responsible for computing an estimated consumption function begins in “Solving the Model by Value Function Maximization,” where a vector of possible values of market resources m is created. In these notes we use $\mathbf{m}$ for such a vector (e.g. $\mathbf{m}[1]$ the first entry, $\mathbf{m}[-1]$ the last). For illustration we take the grid to be the first five nonnegative integers, $\{0, 1, 2, 3, 4\}$.

6.4 An Interpolated Consumption Function

This is accomplished in “An Interpolated Consumption Function,” which generates an interpolating function that we designate $\hat{c}_{\sim(T-1)}(m)$.

Figures 2 and 3 show plots of the constructed $\hat{c}_{T-1}$ and $\hat{v}_{T-1}$. While the $\hat{c}_{T-1}$ function looks very smooth, the fact that the $\hat{v}_{T-1}$ function is a set of line segments is very evident. This figure provides the beginning of the intuition for why trying to approximate the value function directly is a bad idea (in this context).¹⁰

6.5 Interpolating Expectations

Piecewise linear ‘spline’ interpolation as described above works well for generating a good approximation to the true optimal consumption function. However, there is a clear inefficiency in the program: Since it uses equation (26), for every value of m the

¹⁰For some problems, especially ones with discrete choices, value function approximation is unavoidable; nevertheless, even in such problems, the techniques sketched below can be very useful across much of the range over which the problem is defined.

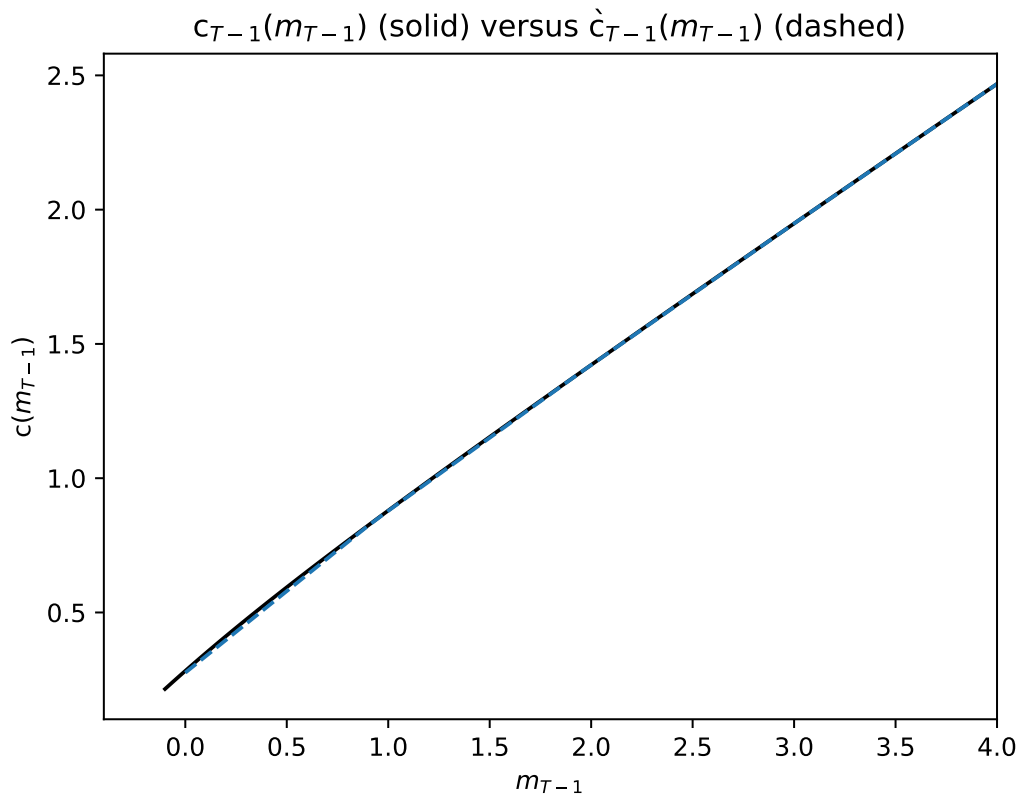


Figure 2 $c_{T-1}(m)$ (solid) versus $\hat{c}_{T-1}(m)$ (dashed)

{fig:PlotcTm1Simple}

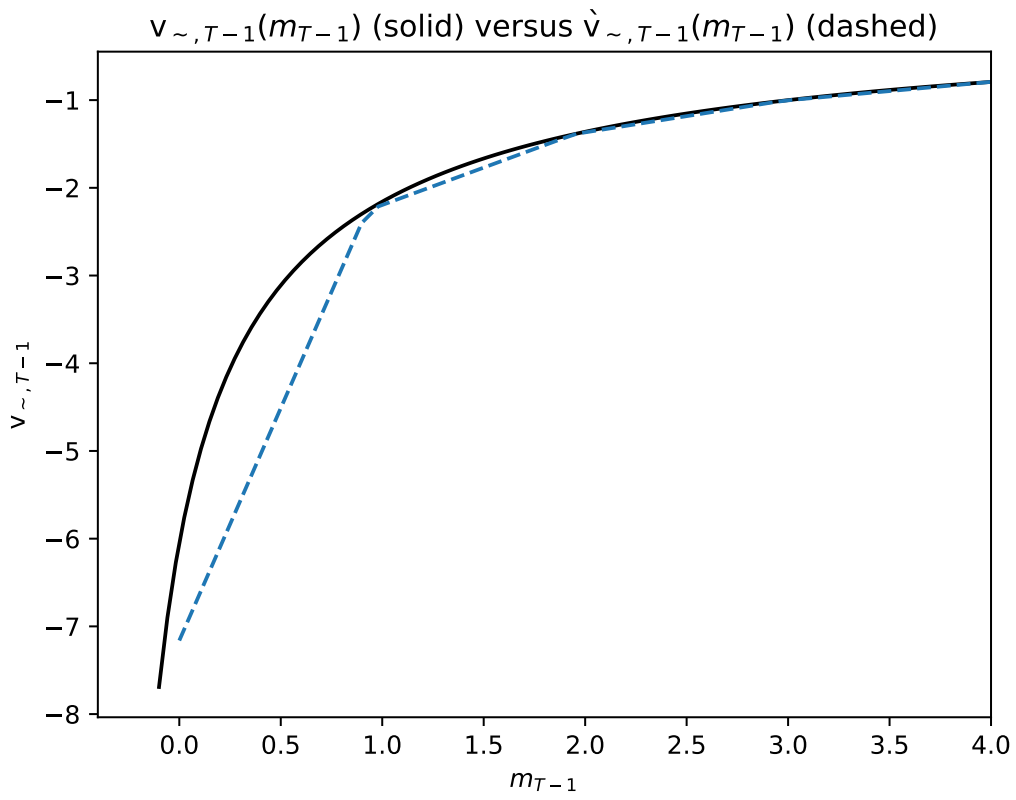


Figure 3 v_{T-1} (solid) versus $\dot{v}_{T-1}(m)$ (dashed)

{fig:PlotVTm1Simple}

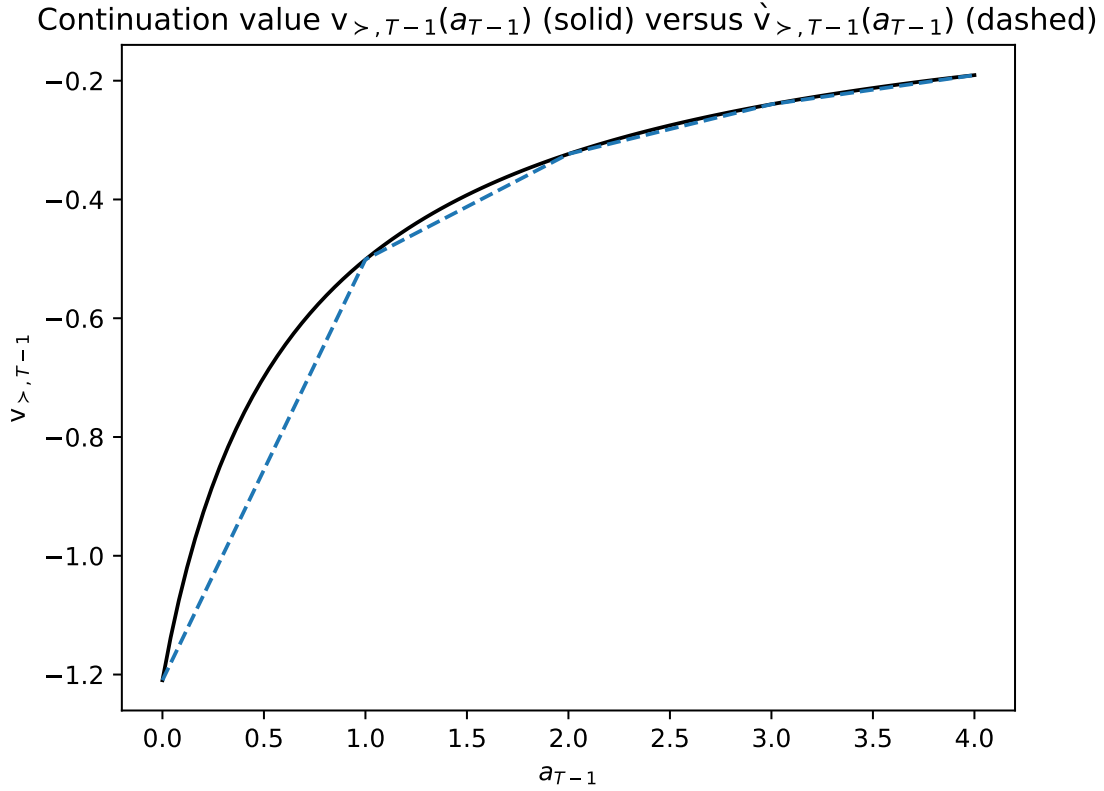


Figure 4 End-Of-Period Value $v_{>(T-1)}(a_{T-1})$ (solid) versus $\hat{v}_{>(T-1)}(a_{T-1})$ (dashed)

program must calculate the utility consequences of various possible choices of c (and therefore a_{T-1}) as it searches for the best choice.

For any given index j in $\mathbf{m}[j]$, as it searches for the corresponding optimal a , the algorithm will end up calculating $v_{>(t-1)}(\tilde{a})$ for many $\tilde{a}$ values close to the optimal a_{T-1} . Indeed, even when searching for the optimal a for a *different* m (say $\mathbf{m}[k]$ for $k \neq j$) the search process might compute $v_{>(t-1)}(a)$ for an a close to the correct optimal a for $\mathbf{m}[j]$. But if that difficult computation does not correspond to the exact solution to the $\mathbf{m}[k]$ problem, it is discarded.

The notebook section “Interpolating Expectations,” now interpolates the expected value of *ending* the period with a given amount of assets.¹¹

Figure 4 compares the true value function to the approximation produced by following the interpolation procedure; the approximated and exact functions are of course identical at the gridpoints of $\mathbf{a}$ and they appear reasonably close except in the region below $m = 1$.

Nevertheless, the consumption rule obtained when the approximating $\hat{v}_{>(T-1)}(a_{T-1})$ is used instead of $v_{>(T-1)}(a_{T-1})$ is surprisingly bad, as shown in figure 5. For example, when m goes from 2 to 3, $\hat{c}_{T-1}$ goes from about 1 to about 2, yet when m goes from 3 to

¹¹What we are doing here is closely related to ‘the method of parameterized expectations’ of [den Haan and Marcet \(1990\)](#); the only difference is that our method is essentially a nonparametric version.

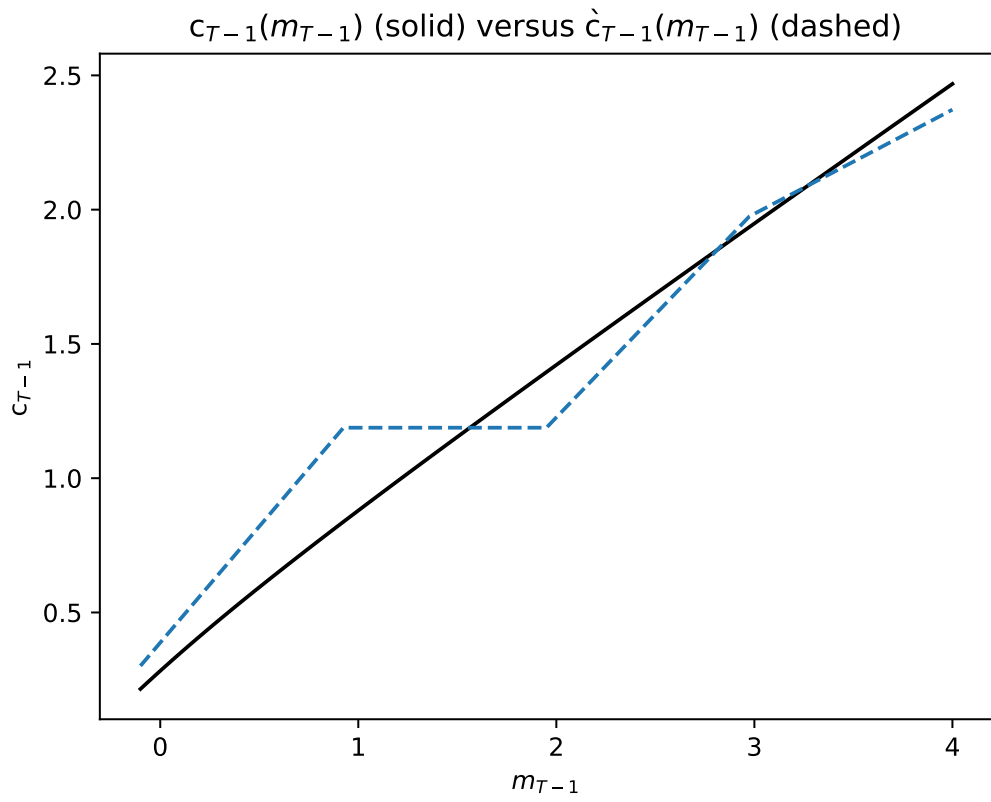


Figure 5 $c_{T-1}(m)$ (solid) versus $\hat{c}_{T-1}(m)$ (dashed)

g:PlotComparecTm1AB}

4, $\hat{c}_{T-1}$ goes from about 2 to about 2.05. The function fails even to be concave, which is distressing because Carroll and Kimball (1996) prove that the correct consumption function is strictly concave in a wide class of problems that includes this one.

6.6 Value Function versus First Order Condition

Loosely speaking, our difficulty reflects the fact that the consumption choice is governed by the *marginal* value function, not by the *level* of the value function (which is the object that we approximated). To understand this point, recall that a quadratic utility function exhibits risk aversion because with a stochastic c ,

$$\mathbb{E}[-(c - \phi)^2] < -(\mathbb{E}[c] - \phi)^2 \quad \text{U} \quad (31)$$

(where ϕ is the ‘bliss point’ which is assumed always to exceed feasible c). However, unlike the CRRA utility function, with quadratic utility the consumption/saving *behavior* of consumers is unaffected by risk since behavior is determined by the first order condition, which depends on *marginal* utility, and when utility is quadratic, marginal utility is unaffected by risk:

$$\mathbb{E}[-2(c - \phi)] = -2(\mathbb{E}[c] - \phi). \quad \text{U} \quad (32)$$

Intuitively, if one’s goal is to accurately capture choices that are governed by marginal value, numerical techniques that approximate the *marginal* value function will yield a more accurate approximation to optimal behavior than techniques that approximate the *level* of the value function.

The first order condition of the maximization problem in period $T - 1$ is:

$$\begin{aligned} u^c(c) &= \beta \mathbb{E}_{\gamma(T-1)}[R u^c(c_T)] \\ c^{-\rho} &= R\beta \left(\frac{1}{n_\theta} \right) \sum_{i=1}^{n_\theta} (R(m - c) + \theta_i)^{-\rho}. \quad \text{U} \end{aligned} \quad (33)$$

The downward-sloping curve in Figure 6 shows the value of $c^{-\rho}$ for our baseline parameter values for $0 \leq c \leq 4$ (the horizontal axis). The solid upward-sloping curve shows the value of the RHS of (33) as a function of c under the assumption that $m = 3$. Optimal consumption given $m = 3$ is the c at which the two curves intersect—just below $c = 2$. The dashed curve shows the same for $m = 4$; its intersection with $u^c(c)$ is slightly below $c = 2.5$, so increasing m from 3 to 4 raises optimal consumption by about 0.5.

Now consider the derivative of $\hat{v}_{\gamma(T-1)}$. Because the function is piecewise linear, its derivative $\hat{v}_{\gamma(T-1)}^\partial(a_{T-1})$ is a step function: constant between adjacent gridpoints, with jumps at each gridpoint.

The solid-line step function in Figure 6 depicts the actual value of $\hat{v}_{\gamma(T-1)}^\partial(3 - c)$. When we attempt to find optimal values of c given m using $\hat{v}_{\gamma(T-1)}^\partial(a_{T-1})$, the numerical optimization routine will return the c for which $u^c(c) = \hat{v}_{\gamma(T-1)}^\partial(m - c)$. Thus, for $m = 3$ the program will return the value of c for which the downward-sloping $u^c(c)$ curve intersects with the $\hat{v}_{\gamma(T-1)}^\partial(3 - c)$; as the diagram shows, this value is exactly

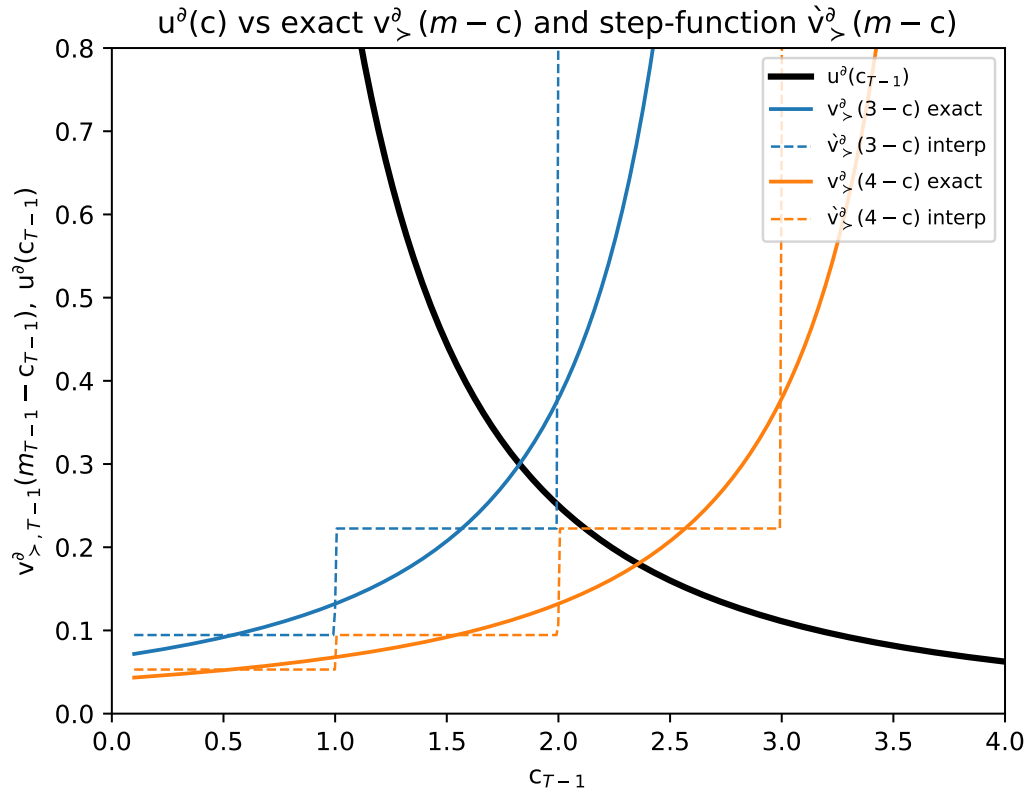


Figure 6 $u^c(c)$ versus $v_{>(T-1)}^\delta(3-c)$, $v_{>(T-1)}^\delta(4-c)$, $\hat{v}_{>(T-1)}^\delta(3-c)$, $\hat{v}_{>(T-1)}^\delta(4-c)$

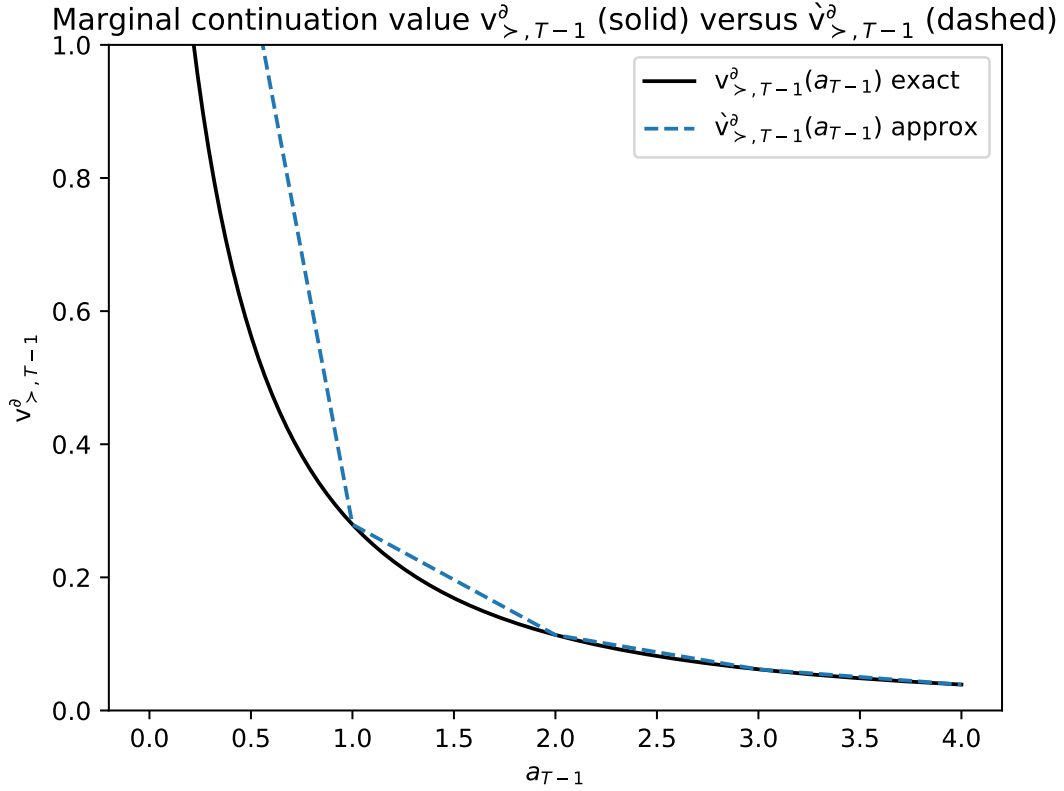


Figure 7 $v_{>(T-1)}^{\partial}(a_{T-1})$ versus $\hat{v}_{>(T-1)}^{\partial}(a_{T-1})$

equal to 2. Similarly, if we ask the routine to find the optimal c for $m = 4$, it finds the point of intersection of $u^c(c)$ with $\hat{v}_{>(T-1)}^{\partial}(4 - c)$; and as the diagram shows, this intersection is only slightly above 2. Hence, this figure illustrates why the numerical consumption function plotted earlier returned values very close to $c = 2$ for both $m = 3$ and $m = 4$.

We would obviously obtain much better estimates of the point of intersection between $u^c(c)$ and $v_{>(T-1)}^{\partial}(m - c)$ if our estimate of $\hat{v}_{>(T-1)}^{\partial}$ were not a step function. In fact, we already know how to construct linear interpolations to functions, so the obvious next step is to construct a linear interpolating approximation to the *expected marginal value of end-of-period assets function* at the points in $\mathbf{a}$:

$$v_{>(T-1)}^{\partial}(\mathbf{a}) = \beta R \left(\frac{1}{n_{\theta}} \right) \sum_{i=1}^{n_{\theta}} (\mathcal{R}_T \mathbf{a} + \theta_i)^{-\rho} \mathbf{u} \quad (34)$$

yielding $\mathbf{v}_{>(T-1)}^{\partial}$ (the vector of expected end-of-period- $(T - 1)$ marginal values of assets corresponding to $\mathbf{aVec}$), and construct $\hat{v}_{>(T-1)}^{\partial}(a_{T-1})$ as the linear interpolating function that fits this set of points.

The results are shown in Figure 7. The linear interpolating approximation looks

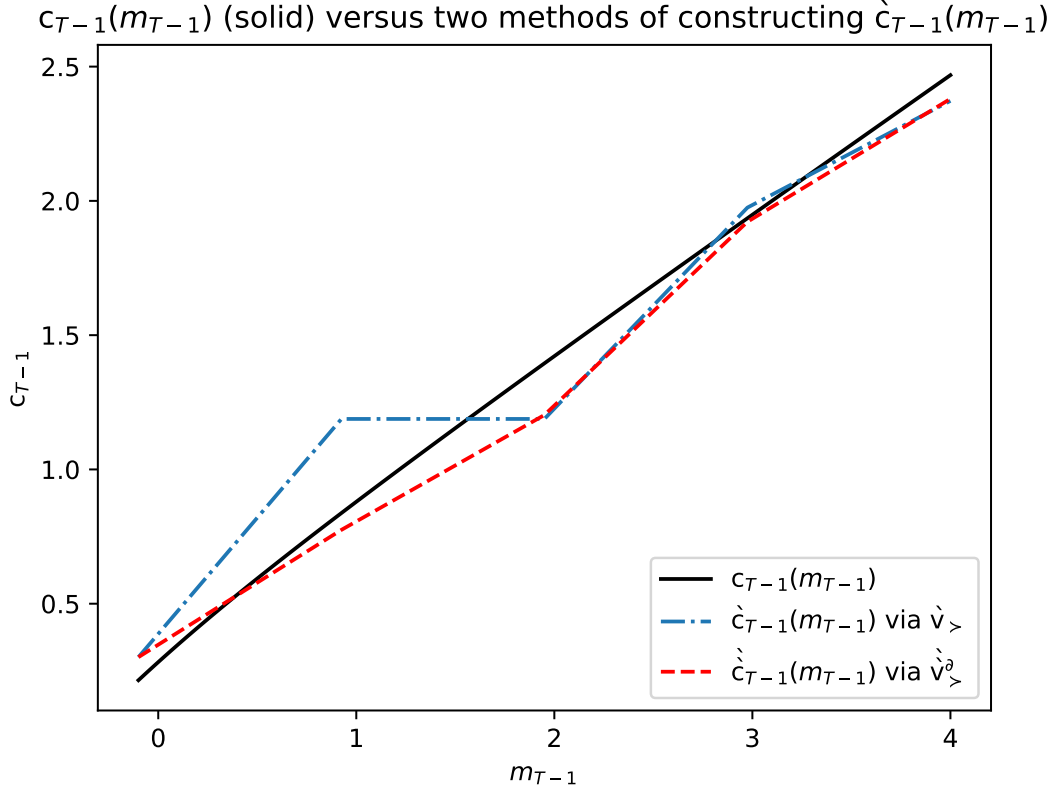


Figure 8 $c_{T-1}(m)$ (solid) Versus Two Methods for Constructing $\hat{c}_{T-1}(m)$

roughly as good (or bad) for the *marginal* value function as it was for the level of the value function. However, Figure 8 shows that the new consumption function (long dashes) is a considerably better approximation of the true consumption function (solid) than was the consumption function obtained by approximating the level of the value function (short dashes).

6.7 Transformation

Even the new-and-improved consumption function diverges notably from the true solution, especially at lower values of m . That is because the linear interpolation does an increasingly poor job of capturing the nonlinearity of $v_{>(T-1)}^{\partial}$ at lower and lower levels of a .

This is where we unveil our next trick. To understand the logic, start by considering the case where $\mathcal{R}_T = \beta = \mathcal{G}_T = 1$ and there is no uncertainty (that is, we know for sure that income next period will be $\theta_T = 1$). The final Euler equation (recall that we are still assuming that $t = T$) is then:

$$c_{T-1}^{-\rho} = c_T^{-\rho} \cdot \mathbf{U} \quad (35)$$

In the case we are now considering with no uncertainty and no liquidity constraints, the optimizing consumer does not care whether a unit of income is scheduled to be received in the future period T or the current period $T - 1$; there is perfect certainty that the income will be received, so the consumer treats its PDV as equivalent to a unit of current wealth. Total resources available at the point when the consumption decision is made is therefore comprised of two types: current market resources m and ‘human wealth’ (the PDV of future income) of $h_{T-1} = 1$ (because it is the value of human wealth as of the end of the period, there is only one more period of income of 1 left).

{eq: vPLin}

$$v_{\sim(T-1)}^{\partial}(m) = \left(\frac{m+1}{2} \right)^{-\rho} \cdot \textcolor{red}{U} \quad (36)$$

Of course, this is a highly nonlinear function. However, if we raise both sides of (36) to the power $(-1/\rho)$ the result is a linear function:

$$[v_{\sim(T-1)}^{\partial}(m)]^{-1/\rho} = \frac{m+1}{2} \cdot \textcolor{red}{U} \quad (37)$$

This is a specific example of a general phenomenon: A theoretical literature discussed in [Carroll and Kimball \(1996\)](#) establishes that under perfect certainty, if the period-by-period marginal utility function is of the form $c_t^{-\rho}$, the marginal value function will be of the form $(\gamma m_t + \zeta)^{-\rho}$ for some constants $\{\gamma, \zeta\}$. This means that if we were solving the perfect foresight problem numerically, we could always calculate a numerically exact (because linear) interpolation.

The key insight is that much of the nonlinearity in v^{∂} comes from raising to the power $-\rho$. By inverting that operation (raising to $-1/\rho$), we can ‘unwind’ it, and the remaining nonlinearity is much smaller. Specifically, applying the foregoing insights to the end-of-period value function $v_{\sim(T-1)}^{\partial}(a)$, we can define an ‘inverse marginal value’ function

{eq: cGoth}

$$\Lambda_{\succ}^{\partial}(a) \equiv (v_{\succ}^{\partial}(a))^{-1/\rho} \cdot \textcolor{red}{U} \quad (38)$$

which would be linear in the perfect foresight case.¹² We then construct a piecewise-linear interpolating approximation to the Λ_t^{∂} function, $\hat{\Lambda}_{\succ}^{\partial}(a_t)$, and for any a that falls in the range $\{\mathbf{a}[1], \mathbf{a}[-1]\}$ we obtain our approximation of marginal value from:

$$\hat{v}_{\sim}^{\partial}(a) = [\hat{\Lambda}^{\partial}(a)]^{-\rho} \cdot \textcolor{red}{U} \quad (39)$$

The most interesting thing about all of this, though, is that the Λ_t^{∂} function has another interpretation. Recall our point in (25) that $u^c(c_t) = v_{\succ}^{\partial}(m_t - c_t)$. Since with CRRA utility $u^c(c) = c^{-\rho}$, this can be rewritten and inverted

$$\begin{aligned} (c_{\succ}(a))^{-\rho} &= v_{\succ}^{\partial}(a) \\ c_{\succ}(a) &= (v_{\succ}^{\partial}(a))^{-1/\rho} \cdot \textcolor{red}{U} \end{aligned} \quad (40)$$

¹²There is a corresponding inverse for the value function: $\Lambda_{\succ}(a_t) = ((1 - \rho)v_{\succ})^{1/(1-\rho)}$, and for the marginal marginal value function etc.

This gives Λ^∂ a concrete interpretation: for any ending a , it reveals how much the agent must *have consumed* to (optimally) reach that a . We will therefore henceforth refer to it as the ‘consumed function.’

{eq:consumedfn}

$$\dot{c}_{\succ}(a) \equiv \dot{\lambda}_{\succ}^\partial(a_t) \cdot \mathbf{U} \quad (41)$$

Thus, for example, for period $T-1$ our procedure is to calculate the vector of $\mathbf{c}$ points on the consumed function:

{eq:consumedfnvecs}

$$\mathbf{c} = c_{\succ(T-1)}(\mathbf{a}) \cdot \mathbf{U} \quad (42)$$

with the idea that we will construct an approximation of the consumed function $\dot{c}_{\succ(T-1)}(a)$ as the interpolating function connecting these $\{\mathbf{a}, \mathbf{c}\}$ points.

6.8 The Natural Borrowing Constraint and the a_{T-1} Lower Bound

iqConstrSelfImposed

This is the appropriate moment to ask an awkward question: How should an interpolated, approximated ‘consumed’ function like $\dot{c}_{\succ(T-1)}(a_{T-1})$ be extrapolated to return an estimated ‘consumed’ amount when evaluated at an a_{T-1} outside the range spanned by $\{\mathbf{a}[1], \dots, \mathbf{a}[n]\}$?

For most canned piecewise-linear interpolation tools like `scipy.interpolate`, when the ‘interpolating’ function is evaluated at a point outside the provided range, the algorithm extrapolates under the assumption that the slope of the function remains constant beyond its measured boundaries (that is, the slope is assumed to be equal to the slope of nearest piecewise segment *within* the interpolated range); for example, if the bottommost gridpoint is $a_1 = \mathbf{a}[1]$ and the corresponding consumed level is $c_1 = c_{\succ(T-1)}(a_1)$ we could calculate the ‘marginal propensity to have consumed’ $\varkappa_1 = \dot{c}_{\succ(T-1)}^\partial(a_1)$ and construct the approximation as the linear extrapolation below $\mathbf{a}[1]$ from:

{eq:ExtrapLin}

$$\dot{c}_{\succ(T-1)}(a) \equiv c_1 + (a - a_1)\varkappa_1 \cdot \mathbf{U} \quad (43)$$

To see that this will lead us into difficulties, consider what happens to the true (not approximated) $v_{\succ(T-1)}^\partial(a_{T-1})$ as a_{T-1} approaches a quantity we will call the ‘natural borrowing constraint’: $\underline{a}_{T-1} = -\underline{\theta}\mathcal{R}_T^{-1}$. From (34) we have

$$\lim_{a \downarrow \underline{a}_{T-1}} v_{\succ(T-1)}^\partial(a) = \lim_{a \downarrow \underline{a}_{T-1}} \beta R \left(\frac{1}{n_\theta} \right) \sum_{i=1}^{n_\theta} (a\mathcal{R}_t + \theta_i)^{-\rho} \cdot \mathbf{U} \quad (44)$$

But since $\underline{\theta} = \theta_1$, exactly at $a = \underline{a}_{T-1}$ the first term in the summation would be $(-\underline{\theta} + \theta_1)^{-\rho} = 1/0^\rho$ which is infinity. The reason is simple: $-\underline{a}_{T-1}$ is the PDV, as of $T-1$, of the *minimum possible realization of income* in T ($\mathcal{R}_T \underline{a}_{T-1} = -\theta_1$). Thus, if the consumer borrows an amount greater than or equal to $\underline{\theta}\mathcal{R}_T^{-1}$ (that is, if the consumer ends $T-1$ with $a_{T-1} \leq -\underline{\theta}\mathcal{R}_T^{-1}$) and then draws the worst possible income shock in period T , they will have to consume zero in period T , which yields $-\infty$ utility and $+\infty$ marginal utility.

As [Zeldes \(1989\)](#) first noticed, this means that the consumer faces a ‘self-imposed’ (or, as above, ‘natural’) borrowing constraint (which springs from the precautionary motive): They will never borrow an amount greater than or equal to $\underline{\theta}\mathcal{R}_T^{-1}$ (that is, assets will never reach the lower bound of $\underline{a}_{T-1}$). The constraint is ‘self-imposed’ in the precise sense that if the utility function were different (say, Constant Absolute Risk Aversion), the consumer might be willing to borrow more than $\underline{\theta}\mathcal{R}_T^{-1}$ because a choice of zero or negative consumption in period T would yield some finite amount of utility.¹³

This self-imposed constraint cannot be captured well when the $v_{>(T-1)}^\partial$ function is approximated by a piecewise linear function like $\hat{v}_{>(T-1)}^\partial$, because it is impossible for the linear extrapolation below $\underline{a}$ to correctly predict $v_{>(T-1)}^\partial(\underline{a}_{T-1}) = \infty$.

So, the marginal value of saving approaches infinity as $a \downarrow \underline{a}_{T-1} = -\underline{\theta}\mathcal{R}_T^{-1}$. But this implies that $\lim_{a \downarrow \underline{a}_{T-1}} c_{>(T-1)}(a) = (v_{>(T-1)}^\partial(a))^{-1/\rho} = 0$; that is, as a approaches its ‘natural borrowing constraint’ minimum possible value, the corresponding amount of worst-case c must approach *its* lower bound: zero.

The upshot is a realization that all we need to do to address these problems is to prepend each of the $\mathbf{a}_{T-1}$ and $\mathbf{c}_{T-1}$ from (42) with an extra point so that the first element in the mapping that produces our interpolation function is $\{\underline{a}_{T-1}, 0\}$. This is done in section “The Self-Imposed ‘Natural’ Borrowing Constraint and the a_{T-1} Lower Bound” of the notebook.

Figure 9 shows the result. The solid line calculates the exact numerical value of the consumed function $c_{>(T-1)}(a)$ while the dashed line is the linear interpolating approximation $\hat{c}_{>(T-1)}(a)$. This figure illustrates the value of the transformation: The true function is close to linear, and so the linear approximation is almost indistinguishable from the true function except at the very lowest values of a .

Figure 10 similarly shows that when we generate $\hat{v}_{>(T-1)}^\partial(a)$ using our augmented $[\hat{c}_{>(T-1)}(a)]^{-\rho}$ (dashed line) we obtain a *much* closer approximation to the true marginal value function $v_{>(T-1)}^\partial(a)$ (solid line) than we obtained in the previous exercise which did not do the transformation (Figure 7).

6.9 The Method of Endogenous Gridpoints (‘EGM’)

The solution procedure above for finding $c_{T-1}(m)$ still requires us, for each point in $\mathbf{m}_{T-1}$, to use a numerical rootfinding algorithm to search for the value of c that solves $u^c(c) = v_{>(T-1)}^\partial(m - c)$. Though sections 6.7 and 6.8 developed a highly efficient and accurate procedure to calculate $\hat{v}_{>(T-1)}^\partial$, those approximations do nothing to eliminate the need for using a rootfinding operation for calculating, for an arbitrary m , the optimal c . And rootfinding is a notoriously computation-intensive (that is, slow!) operation.

Fortunately, it turns out that there is a way to completely skip this slow rootfinding step. The method can be understood by noting that we have already calculated, for a set of arbitrary values of $\mathbf{a} = \mathbf{a}_{T-1}$, the corresponding $\mathbf{c}$ values for which this $\mathbf{a}$ is optimal.

¹³Though it is very unclear what a proper economic interpretation of negative consumption might be – this is an important reason why CARA utility, like quadratic utility, is increasingly not used for serious quantitative work, though it is still useful for teaching purposes.

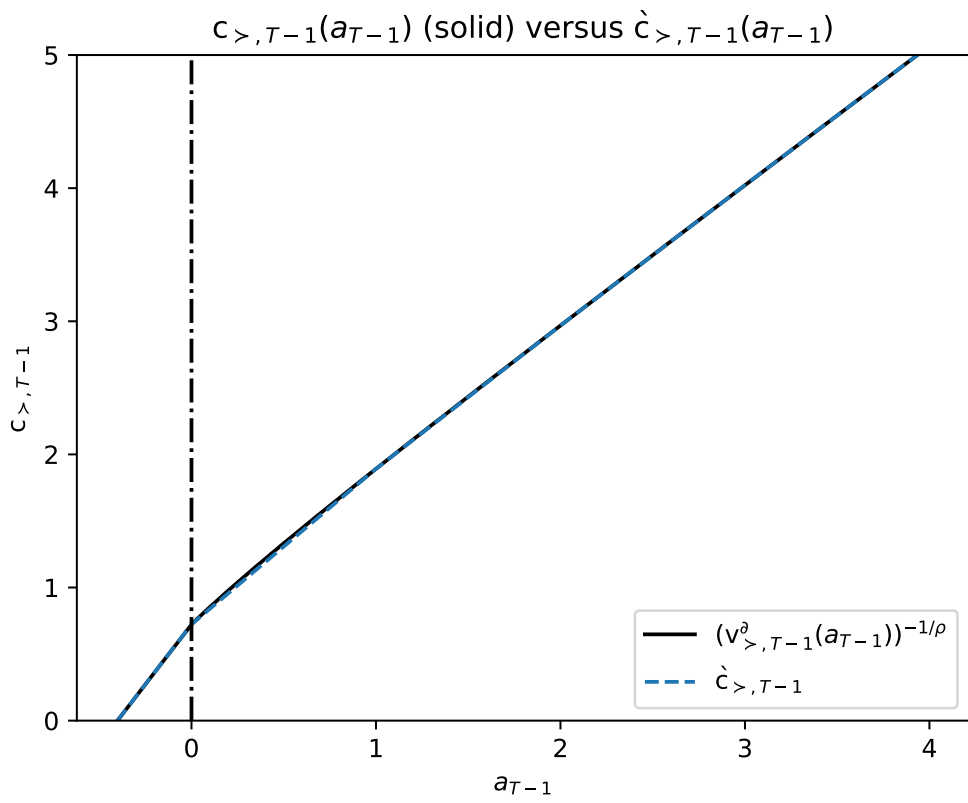


Figure 9 True $\Lambda_{>(T-1)}^\partial(a)$ vs its approximation $\hat{\Lambda}_{>(T-1)}^\partial(a)$

fig:GothVInvVSGothC}

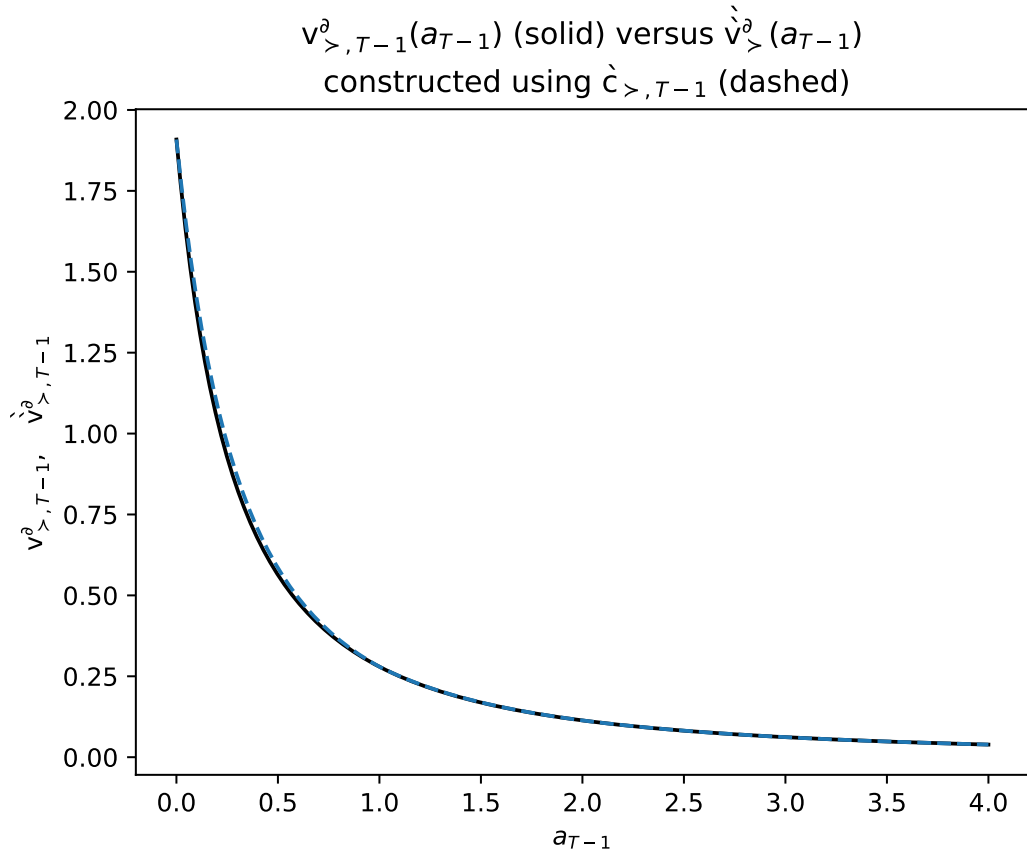


Figure 10 True $v_{>(T-1)}^\partial(a)$ vs. $\hat{v}_{>(T-1)}^\partial(a)$ Constructed Using $\hat{c}_{>(T-1)}(a)$

fig:GothVWSGothCInv}

But with mutually consistent values of $\mathbf{c}_{T-1}$ and $\mathbf{a}_{T-1}$ (consistent, in the sense that they are the unique optimal values that correspond to the solution to the problem), we can obtain the $\mathbf{m}_{T-1}$ vector that corresponds to both of them from

$$\mathbf{m}_{T-1} = \mathbf{c}_{T-1} + \mathbf{a}_{T-1} \cdot \mathbf{U} \quad (45)$$

These m gridpoints are “endogenous” in contrast to the usual solution method of specifying some *ex-ante* (exogenous) grid of values of $\mathbf{m}$ and then using a rootfinding routine to locate the corresponding optimal consumption vector $\mathbf{c}$.

This routine is performed in the “Endogenous Gridpoints” section of the notebook. First, the `endOfPrd.cCntn_Tm1` function is called for each of the pre-specified values of end-of-period assets stored in `aVec`. These values of consumption and assets are used to produce the list of endogenous gridpoints, stored in the object `mVec_egm`. With the $\mathbf{c}$ values in hand, the notebook can generate a set of $\mathbf{m}_{T-1}$ and $\mathbf{c}_{T-1}$ pairs that can be interpolated between in order to yield $\hat{c}_{\sim(T-1)}(m)$ at virtually zero computational cost!¹⁴

One might worry about whether the $\{m, c\}$ points obtained in this way will provide a good representation of the consumption function as a whole, but in practice there are good reasons why they work well (basically, this procedure generates a set of gridpoints that is naturally dense right around the parts of the function with the greatest non-linearity). Figure 11 plots the actual consumption function c_{T-1} and the approximated consumption function $\hat{c}_{T-1}$ derived by the method of endogenous grid points. Compared to the approximate consumption functions illustrated in Figure 8, $\hat{c}_{T-1}$ is quite close to the actual consumption function.

6.10 Improving the a Grid

Thus far, we have arbitrarily used a gridpoints of $\{0., 1., 2., 3., 4.\}$ (augmented in the last subsection by $\underline{a}_{T-1}$). But it has been obvious from the figures that the approximated $\hat{c}_{\sim(T-1)}$ function tends to be farthest from its true value at low values of a . Combining this with our insight that $\underline{a}_{T-1}$ is a lower bound, we are now in position to define a more deliberate method for constructing gridpoints for a – a method that yields values that are more densely spaced at low values of a where the function is more nonlinear.

A pragmatic choice that works well is to find the values such that (1) the last value *exceeds the lower bound* by the same amount $\bar{a}$ as our original maximum gridpoint (in our case, 4.); (2) we have the same number of gridpoints as before; and (3) the *multi-exponential growth rate* (that is, $e^{e^{e^{\dots}}}$ for some number of exponentiations n – our default is 3) from each point to the next point is constant (instead of, as previously, imposing constancy of the absolute gap between points).

Section “Improve the *agrid*” begins by defining a function which takes as arguments the specifications of an initial grid of assets and returns the new grid incorporating the multi-exponential approach outlined above.

Notice that the graphs depicted in Figures 12 and 13 are notably closer to their respective truths than the corresponding figures that used the original grid.

¹⁴This is the essential point of [Carroll \(2006\)](#).

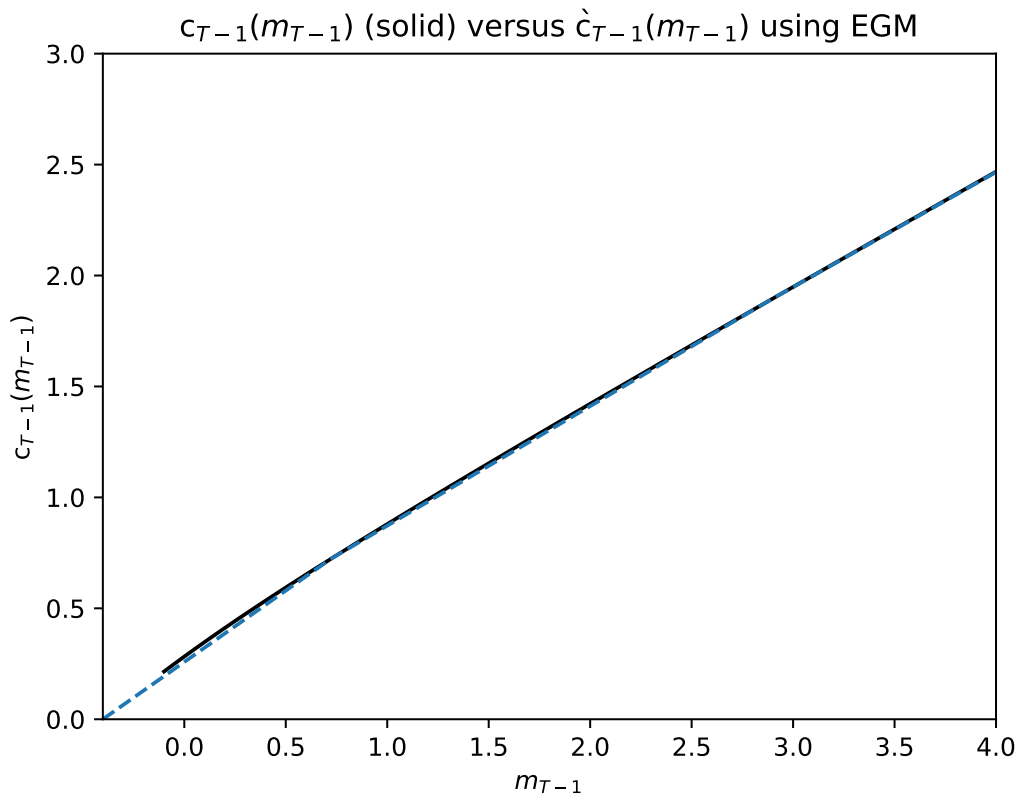


Figure 11 $c_{T-1}(m)$ (solid) versus $\hat{c}_{T-1}(m)$ (dashed)

{fig:ComparecTm1AD}

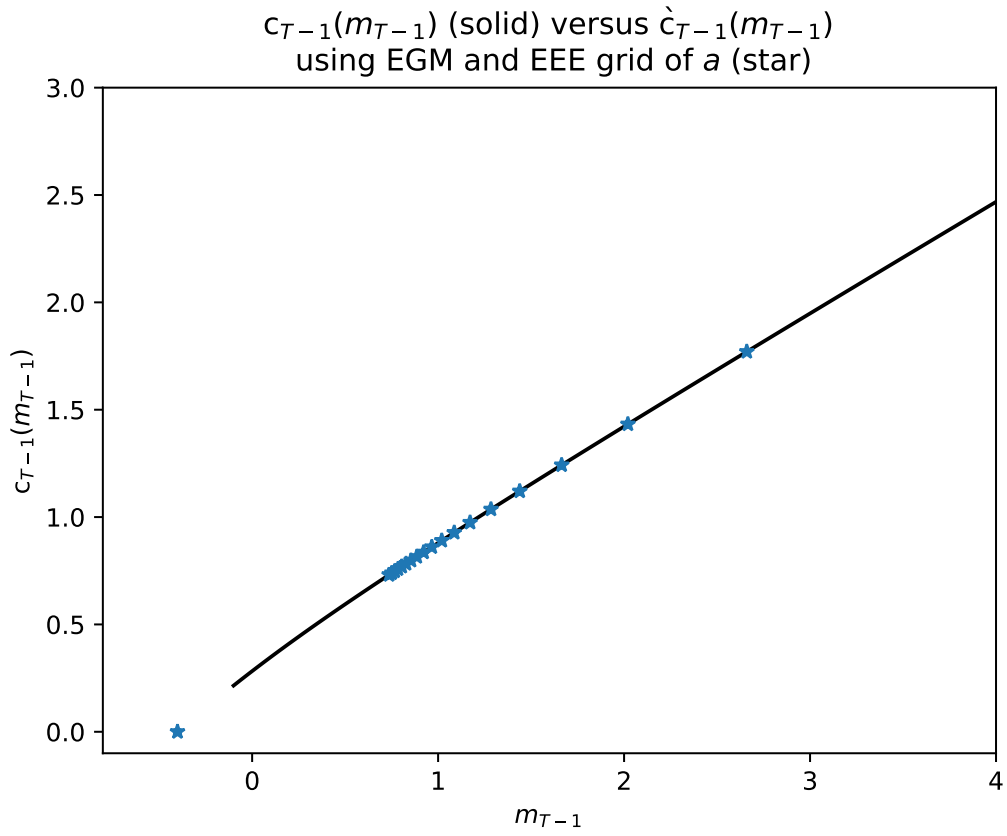


Figure 12 $c_{\succ(T-1)}(a)$ versus $\dot{c}_{\succ(T-1)}(a)$, Multi-Exponential aVec

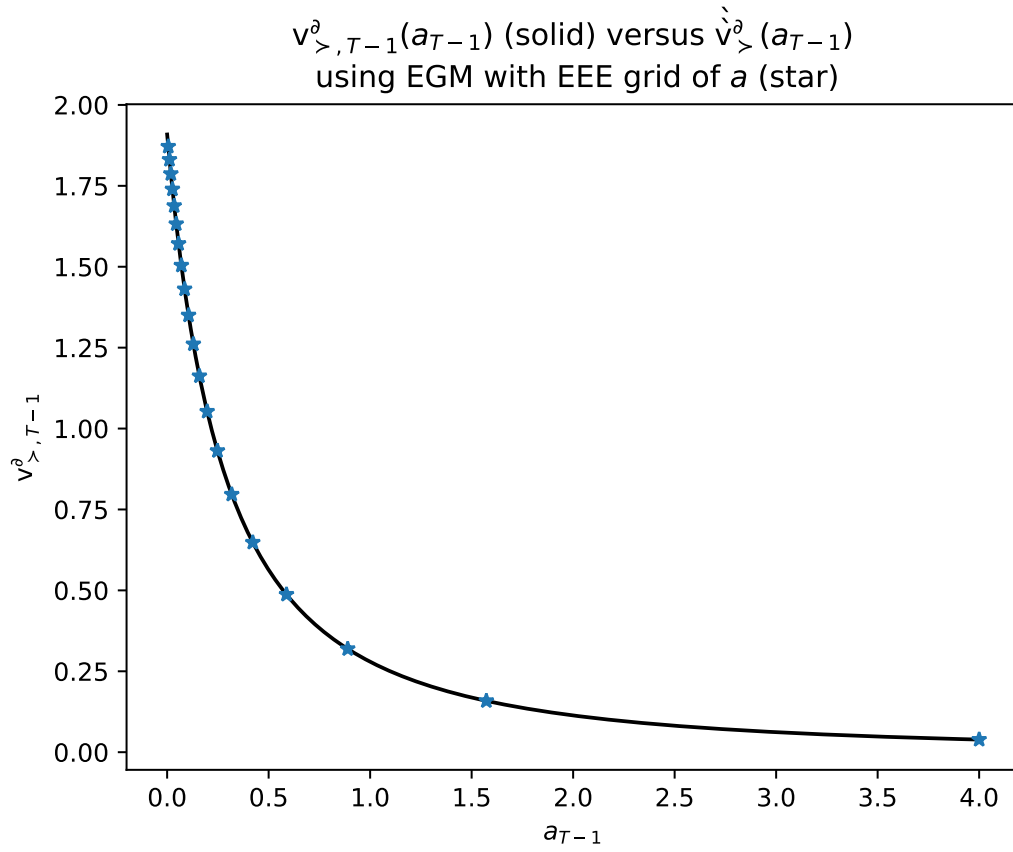


Figure 13 $v_{>(T-1)}^{\partial}(a)$ vs. $\dot{v}_{>(T-1)}^{\partial}(a)$, Multi-Exponential aVec

6.11 Program Structure

In section “Solve for $c_t(m)$ in Multiple Periods,” the natural and artificial borrowing constraints are combined with the endogenous gridpoints method to approximate the optimal consumption function for a specific period. Then, this function is used to compute the approximated consumption in the previous period, and this process is repeated for some specified number of periods.

The essential structure of the program is a loop that iteratively solves for consumption functions by working backward from an assumed final period, using the dictionary `cFunc_life` to store the interpolated consumption functions up to the beginning period. Consumption in a given period is utilized to determine the endogenous gridpoints for the preceding period. This is the sense in which the computation of optimal consumption is done recursively.

In the terminology of section 4.3, each iteration of this backward loop is an invocation of the backward builder $B_{\text{prd}}^{\leftarrow}$: it creates the Connector (inserted into P between the new and existing period solutions), uses it and the already-solved next period to perform the creation of $v_{\succ(t)}$, and solves for the optimal consumption rule. The dictionary `cFunc_life` is the computational embodiment of P —the growing structure of solved periods and Connectors between them assembled by backward induction.

For a realistic life cycle problem, it would also be necessary at a minimum to calibrate a nonconstant path of expected income growth over the lifetime that matches the empirical profile; allowing for such a calibration is the reason we have included the $\{\mathcal{G}\}_t^T$ vector in our computational specification of the problem.

6.12 Results

The code creates the relevant $\hat{c}_t(m)$ functions for any period in the horizon, at the given values of m . Figure 14 shows $\hat{c}_{T-n}(m)$ for $n = \{20, 15, 10, 5, 1\}$. At least one feature of this figure is encouraging: the consumption functions converge as the horizon extends, something that Carroll (2023b) shows must be true under certain parametric conditions that are satisfied by the baseline parameter values being used here.

The construction of $B_{\text{prd}}^{\leftarrow}$ in this single-stage case uses the builders and connectors of subsection 4.3.

7 The Infinite Horizon

All of the solution methods presented so far have involved period-by-period iteration from an assumed last period of life, as is appropriate for life cycle problems. However, if the parameter values for the problem satisfy certain conditions (detailed in Carroll (2023b)), the consumption rules (and the rest of the problem) will converge to a fixed rule as the horizon (remaining lifetime) gets large, as illustrated in Figure 14. Furthermore, Deaton (1991), Carroll (1992; 1997) and others have argued that the ‘buffer-stock’ saving behavior that emerges under some further restrictions on parameter values is a good approximation of the behavior of typical consumers over much of the lifetime. Methods

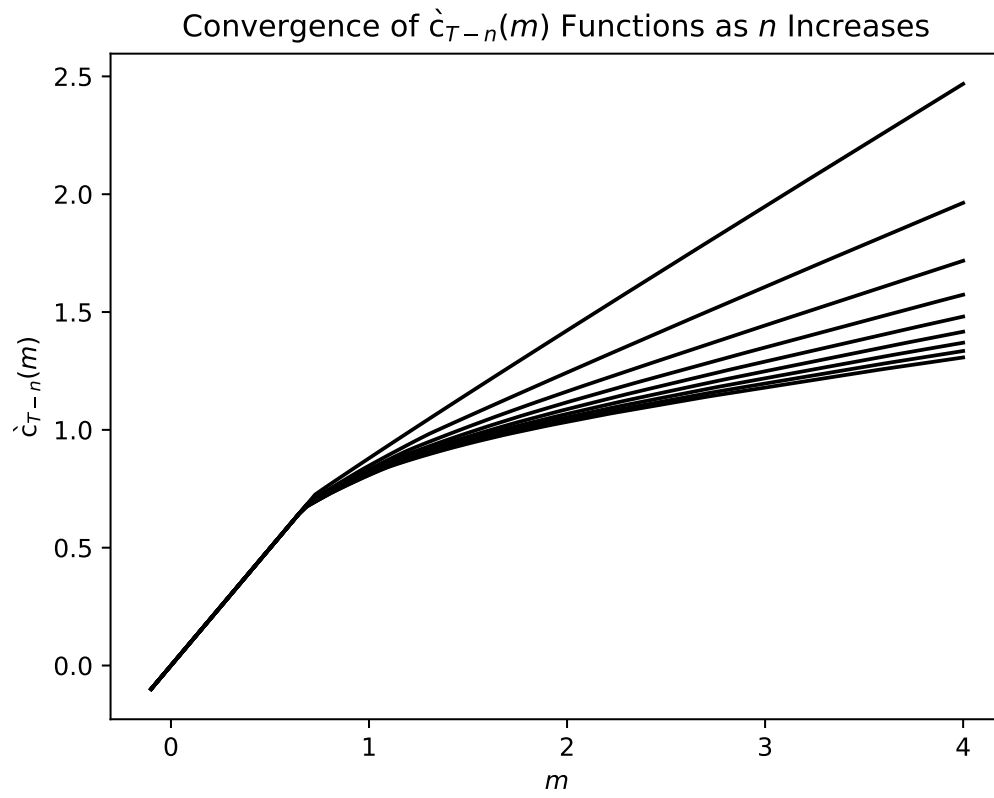


Figure 14 Converging $\dot{c}_{T-n}(m)$ Functions as n Increases

{PlotCFuncsConverge}

for finding the converged functions are therefore of interest, and are dealt with in this section.

Of course, the simplest such method is to solve the problem as specified above for a large number of periods. This is feasible, but there are much faster methods.

7.1 Convergence

In solving an infinite-horizon problem, it is necessary to have some metric that determines when to stop because a solution that is ‘good enough’ has been found.

A natural metric is defined by the unique ‘target’ level of wealth that [Carroll \(2023b\)](#) proves will exist in problems of this kind **under certain conditions**: The $\hat{m}$ such that

$$\mathbb{E}_t[m_{t+1}/m_t] = 1 \text{ if } m_t = \hat{m} \quad (46)$$

where the accent is meant to signify that this is the value that other m ’s ‘point to.’

Given a consumption rule $c(m)$ it is straightforward to find the corresponding $\hat{m}$. So for our problem, a solution is declared to have converged if the following criterion is met: $|\hat{m}_{t+1} - \hat{m}_t| < \epsilon$, where ϵ is a very small number and defines our degree of convergence tolerance.

Similar criteria can obviously be specified for other problems. However, it is always wise to plot successive function differences and to experiment a bit with convergence criteria to verify that the function has converged for all practical purposes.

8 The Method of Moderation

The endogenous gridpoints method constructs a consumption function $\bar{c}_t$ by interpolating a finite set of (m, c) pairs. A practical problem arises when the approximation must be evaluated at values of m outside the grid: naïve linear extrapolation can predict consumption so high that precautionary saving turns negative, an economically impossible result.

A solution, developed in [Carroll, Lujan, Chipeniuk, Tokuoka, and Wu \(2026\)](#), exploits the fact that the true consumption function is bounded between two analytically computable perfect-foresight rules.

- **Optimist.** An agent who is certain that future income shocks will always equal their mean has human wealth $h_{\succ}$ and consumes

$$\bar{c}_t(m) = (m + h_{\succ}) \underline{\kappa}_t, \quad (47)$$

where $\underline{\kappa}_t$ is the minimal marginal propensity to consume of the corresponding perfect-foresight problem.

- **Pessimist.** An agent who assumes the worst possible income realisation in every future period has lower human wealth and a consumption floor at

$$\underline{c}_t(m) = (m - \underline{m}_t) \underline{\kappa}_t, \quad (48)$$

where $\underline{m}_t$ is the natural borrowing constraint.

- **Realist.** The true solution satisfies $\underline{c}_t(m) < c_t(m) < \bar{c}_t(m)$ for all m .

Because the realist's consumption always lies between these bounds, one can define a *moderation ratio*

$$\omega_t(m) \equiv \frac{c_t(m) - \underline{c}_t(m)}{\bar{c}_t(m) - \underline{c}_t(m)} \in (0, 1), \quad (49)$$

which measures how close the realist is to the optimist ($\omega = 1$) versus the pessimist ($\omega = 0$). The key insight is that the logit of this ratio, $\log(\omega/(1-\omega))$, is nearly linear in m and therefore easy to interpolate accurately. Consumption is recovered via the inverse transformation:

$$c_t(m) = \underline{c}_t(m) + \frac{1}{1 + e^{-\chi_t(m)}} (\bar{c}_t(m) - \underline{c}_t(m)), \quad (50)$$

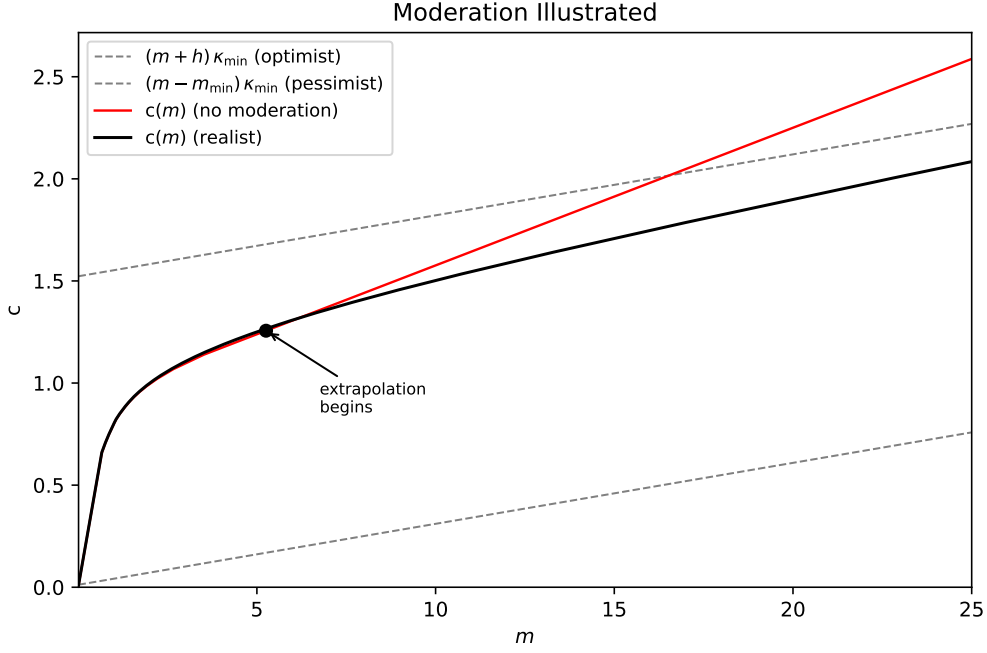
where $\chi_t \equiv \log(\omega_t/(1-\omega_t))$ is the logit-transformed moderation ratio. Since the recovered c is automatically sandwiched between the bounds, the extrapolation problem vanishes.

Figure 15 illustrates the idea: the realist consumption function (middle curve) is bounded above by the optimist and below by the pessimist.

The full treatment in [Carroll, Lujan, Chipeniuk, Tokuoka, and Wu \(2026\)](#) extends the method with tighter upper bounds, applies an analogous transformation to the value function, and shows how to handle stochastic returns and Hermite interpolation refinements.

9 Multiple Control Variables and Modularity

We now consider problems with multiple control variables. Section 9.1 presents the joint consumption-and-portfolio optimization as a canonical example: we derive the simultaneous first-order conditions and show that direct numerical solution of the multidimensional problem is computationally expensive—but that decomposing it into a sequence of single-control problems is much faster. Section 9.2 develops the stage-based machinery for such decompositions, building on the modular notation from section 4: the discounting stage (**disc**), the standalone portfolio optimization (section 9.2.2), and the general-purpose **portable** returns stage that unifies portfolio choice and shock realization. Section 9.2.5 combines these building blocks into three canonical period types.


 Figure 15 Moderation Illustrated: $\underline{c}_{t-1} < \dot{c}_{t-1} < \bar{c}_{t-1}$

9.1 The Joint Optimization Problem

Our canonical example of a multi-control problem is the case where the agent has both a consumption choice and a decision about ς (archaic Greek **stigma**): the share of assets in the risky asset. Given a risky-asset return factor $\mathbf{R}$ and a riskless return factor $\mathbf{R}$, the realized portfolio return is

$$\mathfrak{R}(\varsigma) = \mathbf{R} + (\mathbf{R} - \mathbf{R})\varsigma. \quad (51)$$

Written as a single combined decision (substituting the budget constraint):

$$v_t(m) = \max_{\{c, \varsigma\}} u(c) + \mathbb{E}_{\sim}[\beta v_{t+1}(\overbrace{(m - c)\mathfrak{R}(\varsigma) + \theta}^{m_{t+1}})] \quad (52)$$

Whether the stochastic variables $\mathbf{R}$ and θ are revealed at the end of the current period or the start of the next does not affect this equation. The first-order conditions for this joint problem are:

$$\begin{aligned} u^{\partial}(c) &= v_{\varsigma}^{\partial a}(m - c, \varsigma) \\ 0 &= v_{\varsigma}^{\partial \varsigma}(m - c, \varsigma) \end{aligned} \quad (53)$$

Direct simultaneous solution of these two conditions is computationally expensive; the remainder of this section develops modular stage-based machinery that decomposes the problem into a sequence of simpler single-control optimizations.

9.2 Decomposing Into Modular Stages

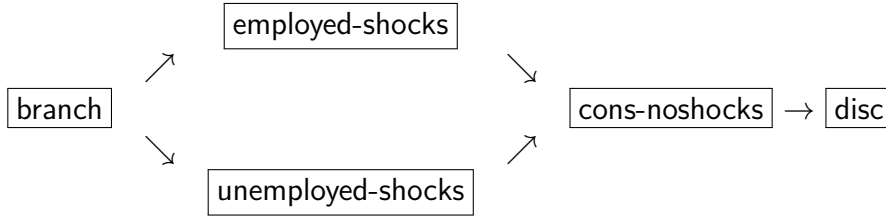
The single-stage-per-period formulation from sections 2–6 is equivalent to a sequence of simpler stages, each with a single job. We decompose it in that way so that adding portfolio choice later requires no change to the consumption-stage code—only the stage list and the Connectors between stages change (see section 4). The decomposition yields the same value functions, consumption function, and Euler equation.

Stage graphs: from lists to directed acyclic graphs. The stage lists presented above—e.g., [shocks-only, C, cons-noshocks, disc]—are *linear*: each stage has exactly one predecessor and one successor. More generally, the stages within a period may form a *directed acyclic graph* (DAG), in which a single predecessor stage fans out into multiple successor branches that later reconverge. Each branch represents a distinct contingency or regime that the agent may face, and the Arrival value function of the branching stage is a probability-weighted combination of the Arrival values from the branches.

Example: employment branches (expository). Consider a consumer who, at the start of each period, is either *employed* or *temporarily unemployed*, with probabilities p_{emp} and $1 - p_{\text{emp}}$ respectively. The two labor-market states imply different income processes:

- **Employed branch.** The consumer receives ordinary labor income θ , so market resources are $m^{\text{emp}} = \mathcal{R}k + \theta$.
- **Unemployed branch.** The consumer receives unemployment insurance benefits μ (a fixed fraction of permanent income), so market resources are $m^{\text{unemp}} = \mathcal{R}k + \mu$.

After the branch-specific income is realized, both branches feed into the *same* shock-free consumption stage (cons-noshocks), which chooses c given m . The DAG for a single period is:



The branching stage has no optimization; it merely routes the agent into one of the two income-realization stages with the appropriate probability. The Arrival value function at the branch point is:

$$v_{\text{branch}, \prec}(k) = p_{\text{emp}} v_{\text{emp}, \prec}(k) + (1 - p_{\text{emp}}) v_{\text{unemp}, \prec}(k).$$

Each branch is itself a valid stage (with its own Arrival, Decision, and Continuation perches); the only new element is the probability-weighted fan-out at the branch point and the fan-in (via a shared Connector) before the consumption stage. This example

is purely expository—no code implementation is provided—but it illustrates how the DAG generalization accommodates heterogeneous regimes within a single period without duplicating the consumption-stage logic.

9.2.1 The Discounting Stage (*disc*)

The simplest decomposition isolates discounting into a standalone stage that applies for the entire period. We call it the **disc** stage (control-name β). From equation (15), $v_{\succ(t)} := \beta v_{\prec(t+1)}$; the factor β was implicitly inside the backward builder $B_{\text{prd}}^{\leftarrow}$ that created $v_{\succ}$. Instead, henceforth at the end of every period, we put the stub **disc** stage which has no choice, no shocks, and a trivial decision value function $v_{\sim} = \beta \cdot v_{\succ}$, and we set the discount factor to 1 for every prior stage in the period (leaving all the discounting to the **disc** stage).¹⁵

Table 2 Discounting Stage (β) Perches

Perch	Indicator	State	value functions	Explanation
Arrival	$\prec$	•	$v_{\prec} = v_{\sim}$	no shocks
Decision(s)	$\sim$	•	$v_{\sim} = \beta v_{\succ}$	apply β
Continuation	$\succ$	•	$v_{\succ}$	value at exit

n.b.: • is a generic passthrough state whose type (k-type or m-type) is inherited from the predecessor stage’s Continuation state. For example, when **disc** follows **cons-noshocks**, • stands for a (k-type); when it follows **portable**, • stands for $\tilde{m}$ (m-type).

Table 3 Rosetta Stone: **disc** Stage Variables

Math Symbol	L ^A T _E X Macro	Python Variable	Description
•	—	state	Passthrough state (inherits type from predecessor)
β	<code>\DiscFac</code>	DiscFac	Time-preference discount factor
$v_{\beta,\sim}$	<code>\vFunc_{\DiscFac,\dcsn}</code>	vDcsn	Decision value: $\beta \cdot v_{\succ}$ (no optimization)
$v_{\beta,\succ}$	<code>\vFunc_{\DiscFac,\cntn}</code>	vCntn	Continuation value function (value at exit)
$v_{\beta,\prec}$	<code>\vFunc_{\DiscFac,\arvl}</code>	vArvl	Arrival value function ($= v_{\sim}$; no shocks)

The two-stage period has stage list [**cons-with-shocks**, **disc**] (more compactly, $[c, \beta]$). The convention that every non-**disc** stage is undiscounted and every period ends with **disc** will mean we do not have to rearrange discounting as we rearrange stages within the period.

¹⁵Equations in sections 2–5, which predate the modular decomposition, show β inside the continuation value of the consumption problem (e.g., (12)). That formulation is equivalent: the β that appears there is the same discount factor, housed inside a composite continuation value rather than in a separate stage. The modular decomposition separates what those equations combine.

9.2.2 The Standalone Portfolio Problem

Consider the standalone problem of an ‘investor’ choosing the portfolio share ς . The state variable at the start is k (capital available for investment); the Decision chooses ς ; stochastic shocks $(\mathbf{R}, \theta)$ are then realized, yielding the Continuation state $\tilde{m} = \mathfrak{R}(\varsigma)k + \theta$. The portfolio share must be chosen *before* the return shock $\mathbf{R}$ is realized. We write $\tilde{m}$ rather than m for the post-shock state because a strict rule of our framework is the prohibition of multiple timings of a given variable within a period. Note that $\tilde{m}$ is m-type (market resources after shocks), matching m ; the $\tilde{}$ decoration distinguishes the two timings while preserving the type (see section 4.3).

The first-order condition with respect to ς is:

$$0 = \mathbb{E}_{\sim} [(\mathbf{R} - \mathbf{R})v_{\succ}^{\partial}(\tilde{m})] \quad \mathbf{U} \quad (54)$$

where $v_{\succ}^{\partial}$ denotes the derivative of the continuation value function with respect to $\tilde{m}$.

The Decision equation yields the portfolio share function:

$$\varsigma(k) = \arg \max_{\varsigma} \mathbb{E}_{\sim} [v_{\succ}((\mathbf{R} + (\mathbf{R} - \mathbf{R})\varsigma)k + \theta)], \quad \mathbf{U} \quad (55)$$

9.2.3 The portable Stage

The portfolio optimization problem above can be generalized into a single configurable *returns-and-shocks* stage by making the portfolio share an *optional parameter* rather than a control variable that must always be optimized. We call this the **portable** stage (“*portfolio-able*”): it is “able” to perform portfolio optimization or not, depending on a parameter $\bar{\varsigma}$.

The stage’s Arrival state includes the investable capital k and an optional portfolio share $\bar{\varsigma} \in [0, 1] \cup \{\perp\}$, where $\perp$ means “absent—optimize.” The behavior of the stage depends on $\bar{\varsigma}$:

- **Optimize mode** ($\bar{\varsigma} = \perp$): The stage solves the portfolio optimization problem from section 9.2.2, choosing ς optimally via (55) with first-order condition (54).
- **Fixed-share mode** ($\bar{\varsigma} \in [0, 1]$): The portfolio share is prescribed; no optimization occurs. The value function is $v_{\prec}(k, \bar{\varsigma}) = \mathbb{E}_{\prec}[v_{\succ}(\mathfrak{R}(\bar{\varsigma})k + \theta)]$.
- **No-risky-investment mode** ($\bar{\varsigma} = 0$): A special case of fixed-share mode. All assets earn the riskless return: $\tilde{m} = \mathbf{R}k + \theta$. The risky return $\mathbf{R}$ is irrelevant. The value function reduces to $v_{\prec}(k, 0) = \mathbb{E}_{\prec}[v_{\succ}(\mathbf{R}k + \theta)]$.

The perches of this stage are:

When $\bar{\varsigma} = \perp$, the Decision value function is $v_{\sim} = \max_{\varsigma} \mathbb{E}_{\sim}[v_{\succ}]$. When $\bar{\varsigma} \in [0, 1]$, it is $v_{\sim} = \mathbb{E}[v_{\succ}]$ with ς fixed at $\bar{\varsigma}$.

Table 4 portable Stage Perches

Perch	Indicator	State	value functions	Explanation
Arrival	$\prec$	$(k, \bar{\varsigma})$	$v_{\prec} = v_{\sim}$	no pre-decision shocks
Decision(s)	$\sim$	$(k, \bar{\varsigma})$	depends on $\bar{\varsigma}$ (see above)	choose ς or use $\bar{\varsigma}$
Continuation	$\succ$	$\tilde{m}$	$v_{\succ}$	post-shock value

Table 5 Rosetta Stone: portable Stage Variables

Math Symbol	LaTeX Macro	Python Variable	Description
k	<code>\kNrm</code>	<code>kNrm</code>	Normalized capital (arrival state, k-type)
ς	<code>\shr</code>	<code>Shr</code>	Portfolio share in risky asset (control or
$\bar{\varsigma}$	<code>\bar{\shr}</code>	<code>ShrFixed</code>	Prescribed portfolio share ($\perp$ = optimize)
$\tilde{m}$	<code>\check{mNrm}</code>	<code>mNrmCheck</code>	Post-shock market resources (continuation)
$\mathfrak{R}$	<code>\Rport</code>	<code>Rport</code>	Portfolio-weighted return factor: $\mathbf{R} + (\mathbf{R} - \mathbf{R}_f)\mathfrak{R}$
$\mathbf{R}$	<code>\Risky</code>	<code>Risky</code>	Stochastic risky-asset return factor
$\mathbf{R}_f$	<code>\Rfree</code>	<code>Rfree</code>	Risk-free return factor
θ	<code>\tranShkEmp</code>	<code>tranShkEmp</code>	Transitory income shock
$\mathcal{R}$	<code>\RNrmByG</code>	<code>RNrmByG</code>	Normalized return factor $\mathbf{R}/\mathcal{G}$ (used when $\mathfrak{R} = \perp$)
$v_{\varsigma, \sim}$	<code>\vFunc_{\shr, \dcsn}</code>	<code>vDcsn</code>	Decision value: $\max_{\varsigma} \mathbb{E}[v_{\succ}]$ or $\mathbb{E}[v_{\succ}]$ if $\varsigma = \bar{\varsigma}$
$v_{\varsigma, \succ}$	<code>\vFunc_{\shr, \cntn}</code>	<code>vCntn</code>	Continuation value function (post-shock)
$v_{\varsigma, \prec}$	<code>\vFunc_{\shr, \arvl}</code>	<code>vArvl</code>	Arrival value function ($= v_{\sim}$; no pre-decision)

Argument notation. When a stage can be constructed with a fixed parameter, we write the value of that argument in parentheses. Thus $\varsigma(0)$ denotes the **portable** stage with $\bar{\varsigma} = 0$ (the **shocks-only** configuration described below), and $\varsigma(\perp)$ denotes **portable** in optimize mode.

9.2.4 Separating Shocks from Choices

The **cons-with-shocks** stage in section 4.2 combines the $k \rightarrow m$ shock realization and the $m \rightarrow a$ consumption decision. We split these into two stages.

The shocks-only stage (portable with $\bar{\varsigma} = 0$). Calling the **portable** stage with $\bar{\varsigma} = 0$ produces a stage whose only function is to draw the shocks. We call this configuration **shocks-only**. It handles the transition from capital k to (post-shock) market resources $\tilde{m}$: with $\varsigma = 0$, the transition is $\tilde{m} = \mathcal{R}k + \theta$ —precisely the $k \rightarrow \tilde{m}$ mapping from the original problem (the next stage receives this as m). No optimization occurs: The Arrival value function takes the expectation over the shocks: $v_{\prec}(k) = \mathbb{E}_{\prec}[v_{\succ}(\mathcal{R}k + \theta)]$.

Table 6 shocks-only (portable with $\bar{\varsigma} = 0$) Stage Perches

Perch	Indicator	State	value functions	Explanation
Arrival	$\prec$	k	$v_{\prec} = \mathbb{E}_{\prec}[v_{\succ}]$	pre-shock value
Decision(s)	$\sim$	k	(none)	no choice
Continuation	$\succ$	$\check{m}$	$v_{\succ}$	post-shock value

The Continuation state $\check{m} = \mathcal{R}k + \theta$ is fully determined (non-stochastic) once the shocks are realized.

The shock-free consumption stage (cons-noshocks). With shocks in the preceding stage, the consumption stage has Arrival state m and no shocks between Arrival and Decision, so $v_{\prec} = v_{\sim}$. We call this the **cons-noshocks** stage (control-name c):

Table 7 cons-noshocks Stage Perches

Perch	Indicator	State	value functions	Explanation
Arrival	$\prec$	m	$v_{\prec} = v_{\sim}$	no shocks; identity
Decision(s)	$\sim$	m	$v_{\sim}(m) = \max_c u(c) + v_{\succ}(m - c)$	choose consumption
Continuation	$\succ$	a	$v_{\succ}$	value at exit

Table 8 Rosetta Stone: cons-noshocks Stage Variables

Math Symbol	L ^A T _E X Macro	Python Variable	Description
m	<code>\mNrm</code>	<code>mNrm</code>	Normalized market resources (arrival s
c	<code>\cNrm</code>	<code>cNrm</code>	Normalized consumption (control vari
a	<code>\aNrm</code>	<code>aNrm</code>	Normalized end-of-stage assets (contin
$v_{c,\sim}$	<code>\vFunc_{\{cFunc,dcsn\}}</code>	<code>vDcsn</code>	Decision value function: $\max_c u(c) + v$
$v_{c,\succ}$	<code>\vFunc_{\{cFunc,cntn\}}</code>	<code>vCntn</code>	Continuation value function at stage e
$v_{c,\prec}$	<code>\vFunc_{\{cFunc,arvl\}}</code>	<code>vArvl</code>	Arrival value function ($= v_{\sim}$; no pre-d
u	<code>\uFunc</code>	<code>uFunc</code>	CRRA utility function
ρ	<code>\CRRA</code>	<code>CRRA</code>	Coefficient of relative risk aversion

The Decision equation for this stage defines the consumption function:

$$c_{c,\sim}(m) = \arg \max_c u(c) + v_{c,\succ}(\underbrace{m - c}_a) \cdot \mathbf{U} \quad (56)$$

The Decision is unchanged from the single-stage formulation. But this stage is a ‘stub’: It could not be the only stage in a period because its continuation state a is of a different type than its arrival state m . A full period would require it to be paired with another stage that produced the transition from a to m .

The three-stage period. The consumption-only period from sections 2–6 is defined by the stage list [shocks-only, $\mathcal{C}(\tilde{m} \leftrightarrow m)$, cons-noshocks, disc], or equivalently $[\varsigma(0), \mathcal{C}(\tilde{m} \leftrightarrow m), c, \beta]$ in control-name form (recall $\varsigma(0)$ denotes **portable** with $\bar{\varsigma} = 0$; see section 9.2.3). Explicitly:

Element	Transition	Action
shocks-only	$k \rightarrow \tilde{m}$	shocks realize (no choice)
$\mathcal{C}(\tilde{m} \leftrightarrow m)$	$\tilde{m} \rightarrow m$	rename
cons-noshocks	$m \rightarrow a$	choose c
disc		apply β

This is functionally identical to the single-stage formulation. Each stage resolves its stochastic content *internally*; the value at a stage’s exit is therefore non-stochastic.

Multi-stage notation. Once our Pile $\mathbf{P}$ has accumulated multiple periods and stages, we address any perch-specific object (like a value function) by comma-separated subscripts, ordered from outermost to innermost—period, stage, perch:

$$v_{t,\varsigma,\succ}(a).$$

We drop the period when considering a stage from a context inside a period ($v_{\varsigma,\succ}(a)$).

9.2.5 Three Period Types

Using the **portable** stage (with $\bar{\varsigma} = 0$ or $\perp$), the **cons-noshocks** stage, and the **disc** stage at the end of every period, we can construct three kinds of periods. A period type is defined by its stage list (section 4), written in square brackets:

Table 9 Three Period Types

Period type	Transition	Description
shocksonly–consnoshocks	$k \rightarrow a$	No portfolio choice
portable–consnoshocks	$k \rightarrow a$	Beginning-of-period returns
consnoshocks–portable	$m \rightarrow \tilde{m}$	End-of-period returns

We now describe these three variants in detail. We present the consnoshocks–portable (end-of-period returns) variant first, because it makes the need for the $\tilde{m}$ notation most transparent.

bsec:ending-returns}

The consnoshocks–portable period (end-of-period returns). If the portfolio share choice is made and stochastic shocks are realized at the *end* of the period, the shock-free consumption stage comes first and the ς stage follows: stage list by control-name $[c, \mathcal{C}(a \leftrightarrow k), \varsigma, \beta]$. The flow is:

Element	Transition	Action
cons-noshocks	$m \rightarrow a$	choose c
$\mathcal{C}(a \leftrightarrow k)$	$a \rightarrow k$	rename
portable	$k \rightarrow \tilde{m}$	choose ς , shocks realize
disc		apply β

The within-period Connector $\mathcal{C}(a \leftrightarrow k)$ relabels the consumption stage's output as the ς stage's input; the remaining Connectors are catalogued in section 9.2.6. The reason we used $\tilde{m}$ rather than m for the post-portfolio state now becomes evident: this prevents two different values of m from coexisting within the same period. The $\sim$ decoration is appropriate because $\tilde{m}$ remains m-type—it is market resources, differing from m only in timing.

c:beginning-returns}

The portable–consnoshocks period (beginning-of-period returns). The beginning-returns problem places the ς stage *before* the shock-free consumption stage: stage list by control-name $[\varsigma, \mathcal{C}(\tilde{m} \leftrightarrow m), c, \beta]$. The first-order condition for the portfolio-choice stage is (54).

The flow through a single period is:

Element	Transition	Action
portable	$k \rightarrow \tilde{m}$	choose ς , shocks realize
$\mathcal{C}(\tilde{m} \leftrightarrow m)$	$\tilde{m} \rightarrow m$	rename
cons-noshocks	$m \rightarrow a$	choose c
disc		apply β

Since all stochastic shocks are realized *inside* the ς stage, the value $\tilde{m}$ that exits it is fully determined. The within-period Connector $\mathcal{C}(\tilde{m} \leftrightarrow m)$ therefore simply relabels $\tilde{m}$ as m for the consumption stage (see section 9.2.6 for the full catalogue).

The shockonly–consnoshocks period. This is the three-stage decomposition from above: $[\varsigma(0), \mathcal{C}(\tilde{m} \leftrightarrow m), c, \beta]$. A life cycle model in which portfolio choice is available at some ages but not others is now trivially constructed: the modeler simply sets $\bar{\varsigma} = \perp$ for ages with active portfolio choice and $\bar{\varsigma} = 0$ for ages without.

9.2.6 Connectors for Each Period Type

subsubsec:connectors}

To construct a Pile $\mathbf{P}$ that repeats instances of a given period type, the backward builder $\mathbf{B}_{\text{prd}}^{\leftarrow}$ must know the *between-period* Connector that ties the last stage of one period to the first stage of the next (see section 4.3). Within-period Connectors are already specified in the period tables above.

period type	Between-period Connector
shocksonly–consnoshocks	$\mathcal{C}(a \leftrightarrow k)$
portable–consnoshocks	$\mathcal{C}(a \leftrightarrow k)$
consnoshocks–portable	$\mathcal{C}(\tilde{m} \leftrightarrow m)$

Each of these Connectors is a pure rename that respects the state-variable type constraint from section 4.3: a and k are both k -type (capital before returns), while $\tilde{m}$ and m are both m -type (market resources after returns). The connector is determined by the predecessor’s exit state and the successor’s first-stage arrival state. For portable–consnoshocks, the period ends after disc (passthrough), so the exit state is a ; the next period’s portable stage arrives with k , hence $\mathcal{C}(a \leftrightarrow k)$. For shocksonly–consnoshocks the same logic gives $\mathcal{C}(a \leftrightarrow k)$. For consnoshocks–portable, the period exits with $\tilde{m}$; the Connector relabels it as m for the next period’s cons-noshocks stage.

9.2.7 Numerical Solution

Following the sequential approach outlined in section 9.1, we solve the portfolio stage numerically for the optimal ς at a vector of $\mathbf{k}$ and construct an approximated optimal portfolio share function $\zeta(k)$ as the interpolating function among the members of the $\{\mathbf{k}, \varsigma\}$ mapping. Having done this, we calculate a vector of values and marginal values at that grid:

$$\begin{aligned} \mathbf{v} &= v_{\varsigma, \prec}(\mathbf{k}) \\ \mathbf{v}^\partial &= v_{\varsigma, \prec}^\partial(\mathbf{k}) \cdot \mathbf{U} \end{aligned} \quad (57)$$

With the $\mathbf{v}^\partial$ approximation in hand, we construct our approximation to the consumption function using *exactly the same EGM procedure* that we used in solving the problem *without* a portfolio choice (see (38)):

$$\mathbf{c} \equiv (\mathbf{v}^\partial)^{-1/\rho}, \mathbf{U} \quad (58)$$

which, following the procedure in subsection 6.9, yields an approximated consumption function $\hat{c}_t(m)$.

9.3 Implementation

Following the discussion from section 9.1, to provide a numerical solution to the problem with multiple control variables, we must define expressions that capture the expected marginal value of end-of-period assets with respect to the level of assets and the share invested in risky assets. This is addressed in “Multiple Control Variables.”

9.4 Results With Multiple Controls

Figure 16 plots the $t-1$ consumption function generated by the program; qualitatively it does not look much different from the consumption functions generated by the program without portfolio choice.

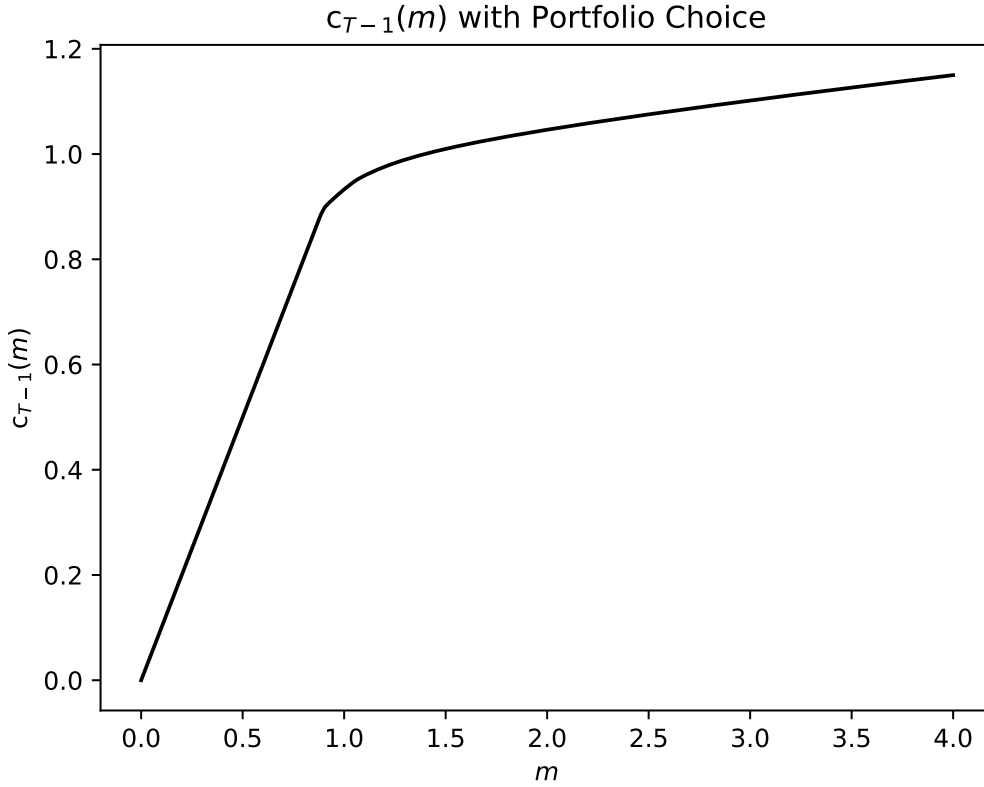


Figure 16 $c_{T-1}(m)$ With Portfolio Choice

fig:PlotctMultContr}

But Figure 17 which plots the optimal portfolio share as a function of the level of assets, exhibits several interesting features. First, even with a coefficient of relative risk aversion of 6, an equity premium of only 4 percent, and an annual standard deviation in equity returns of 15 percent, the optimal choice at values of a_t less than about 2 is for the agent to invest a proportion 1 (100 percent) of the portfolio in stocks (instead of the safe bank account with riskless return R). Second, the proportion of the portfolio kept in stocks is *declining* in the level of wealth - i.e., the poor should hold all of their meager assets in stocks, while the rich should be cautious, holding more of their wealth in safe bank deposits and less in stocks. This seemingly bizarre (and highly counterfactual – see Carroll (2002)) prediction reflects the nature of the risks the consumer faces. Those consumers who are poor in measured financial wealth will likely derive a high proportion of future consumption from their labor income. Since by assumption labor income risk is uncorrelated with rate-of-return risk, the covariance between their future consumption and future stock returns is relatively low. By contrast, persons with relatively large wealth will be paying for a large proportion of future consumption out of that wealth, and hence if they invest too much of it in stocks their consumption will have a high covariance with stock returns. Consequently, they reduce that correlation by holding some of their wealth in the riskless form.

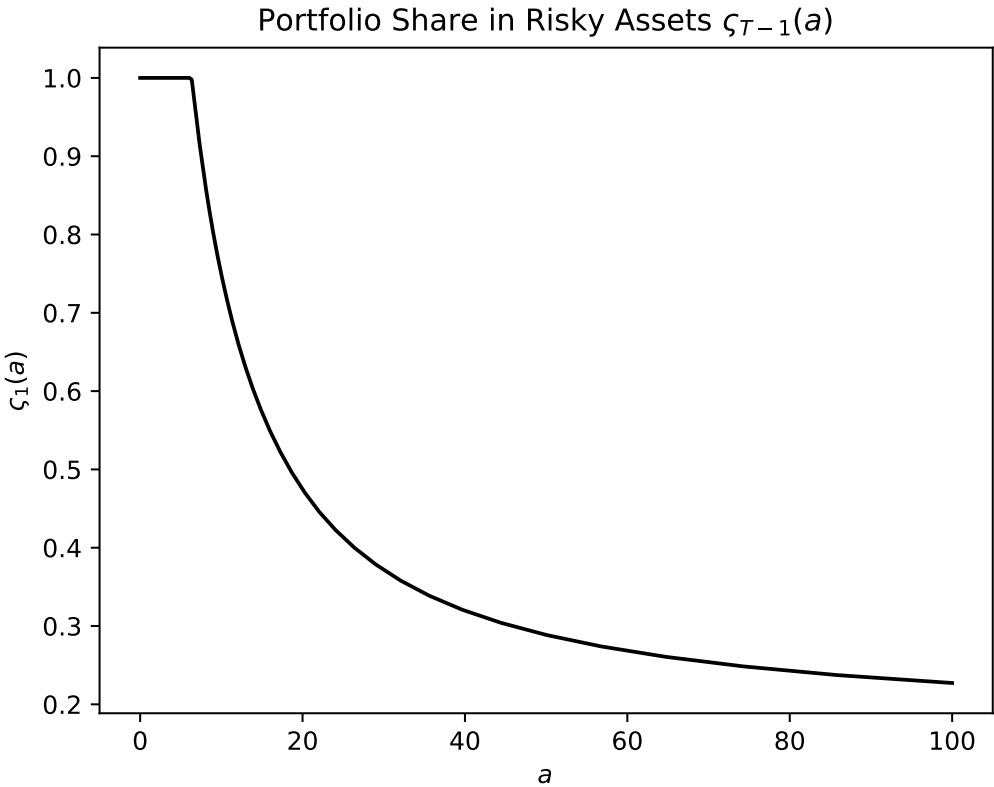


Figure 17 Portfolio Share in Risky Assets $\zeta_{T-1}(a)$

PlotRiskyShareOfat}

10 Structural Estimation

This section describes how to use the methods developed above to structurally estimate a life-cycle consumption model, following closely the work of [Cagetti \(2003\)](#).¹⁶ The key idea of structural estimation is to look for the parameter values (for the time preference rate, relative risk aversion, or other parameters) which lead to the best possible match between simulated and empirical moments.

10.1 Life Cycle Model

Realistic calibration of a life cycle model needs to take into account a few things that we omitted from the bare-bones model described above. For example, the whole point of the life cycle model is that life is finite, so we need to include a realistic treatment of life expectancy; this is done easily enough, by assuming that utility accrues only if you live, so effectively the rising mortality rate with age is treated as an extra reason for discounting the future. Similarly, we may want to capture the demographic evolution of the household (e.g., arrival and departure of kids). A common way to handle that, too, is by modifying the discount factor (arrival of a kid might increase the total utility of the household by, say, 0.2, so if the ‘pure’ rate of time preference were 1.0 the ‘household-size-adjusted’ discount factor might be 1.2. We therefore modify the model presented above to allow age-varying discount factors that capture both mortality and family-size changes (we just adopt the factors used by [Cagetti \(2003\)](#) directly), with the probability of remaining alive between t and $t + n$ captured by $\mathcal{L}$ and with $\hat{\beta}$ now reflecting all the age-varying discount factor adjustments (mortality, family-size, etc). Using $\beth$ (the Hebrew cognate of β) for the ‘pure’ time preference factor, the value function for the revised problem is

$$v_{\sim(t)}(\mathbf{p}_t, \mathbf{m}_t) = \max_{\{\mathbf{c}\}_t^T} u(\mathbf{c}_t) + \mathbb{E}_{\succ(t)} \left[\sum_{n=1}^{T-t} \beth^n \mathcal{L}_t^{t+n} \hat{\beta}_t^{t+n} u(\mathbf{c}_{t+n}) \right] \quad \mathbf{u} \quad (59)$$

subject to the constraints

$$\begin{aligned} \mathbf{a}_t &= \mathbf{m}_t - \mathbf{c}_t \\ \mathbf{p}_{t+1} &= \mathcal{G}_{t+1} \mathbf{p}_t \psi_{t+1} \\ \mathbf{y}_{t+1} &= \mathbf{p}_{t+1} \theta_{t+1} \\ \mathbf{m}_{t+1} &= R \mathbf{a}_t + \mathbf{y}_{t+1} \end{aligned}$$

where

$$\begin{aligned} \mathcal{L}_t^{t+n} &: \text{probability to survive until age } t+n \text{ given alive at age } t \\ \hat{\beta}_t^{t+n} &: \text{age-varying discount factor between ages } t \text{ and } t+n \\ \psi_t &: \text{mean-one shock to permanent income} \\ \beth &: \text{time-invariant ‘pure’ discount factor} \end{aligned}$$

¹⁶Similar structural estimation exercises have been also performed by [Palumbo \(1999\)](#) and [Gourinchas and Parker \(2002\)](#).

and all the other variables are defined as in section 2.

Households start life at age $s = 25$ and live with probability 1 until retirement ($s = 65$). Thereafter the survival probability shrinks every year and agents are dead by $s = 91$ as assumed by Cagetti.

Transitory and permanent shocks are distributed as follows:

$$\Xi_s = \begin{cases} 0 & \text{with probability } q > 0 \\ \theta_s/q & \text{with probability } (1 - q), \text{ where } \log \theta_s \sim \mathcal{N}(-\sigma_\theta^2/2, \sigma_\theta^2) \end{cases} \quad (60)$$

$$\log \psi_s \sim \mathcal{N}(-\sigma_\psi^2/2, \sigma_\psi^2) \mathbf{U}$$

where q is the probability of unemployment (and unemployment shocks are turned off after retirement).

The parameter values for the shocks are taken from Carroll (1992), $q = 0.5/100$, $\sigma_\theta = 0.1$, and $\sigma_\psi = 0.1$.¹⁷ The income growth profile $\mathcal{G}_t$ is from Carroll (1997) and the values of $\mathcal{L}_t$ and $\hat{\beta}_t$ are obtained from Cagetti (2003) (Figure 18).¹⁸ The interest rate is assumed to equal 1.03. The model parameters are included in Table 10.

Table 10 Parameter Values

σ_θ	0.1	Carroll (1992)
σ_ψ	0.1	Carroll (1992)
q	0.005	Carroll (1992)
$\mathcal{G}_s$	figure 18	Carroll (1997)
$\hat{\beta}_s, \mathcal{L}_s$	figure 18	Cagetti (2003)
R	1.03	Cagetti (2003)

The structural estimation of the parameters $\beth$ and ρ is carried out using the procedure specified in the following section, which is then implemented in the `StructEstimation.py` file. This file consists of two main components. The first section defines the objects required to execute the structural estimation procedure, while the second section executes the procedure and various optional experiments with their corresponding commands. The next section elaborates on the procedure and its accompanying code implementation in greater detail.

¹⁷Note that $\sigma_\theta = 0.1$ is smaller than the estimate for college graduates estimated in Carroll and Samwick (1997) ($= 0.197 = \sqrt{0.039}$) which is used by Cagetti (2003). The reason for this choice is that Carroll and Samwick (1997) themselves argue that their estimate of σ_θ is almost certainly increased by measurement error.

¹⁸The income growth profile is the one used by Carroll for operatives. Cagetti computes the time-varying discount factor by educational groups using the methodology proposed by Attanasio et al. (1999) and the survival probabilities from the 1995 Life Tables (National Center for Health Statistics 1998).

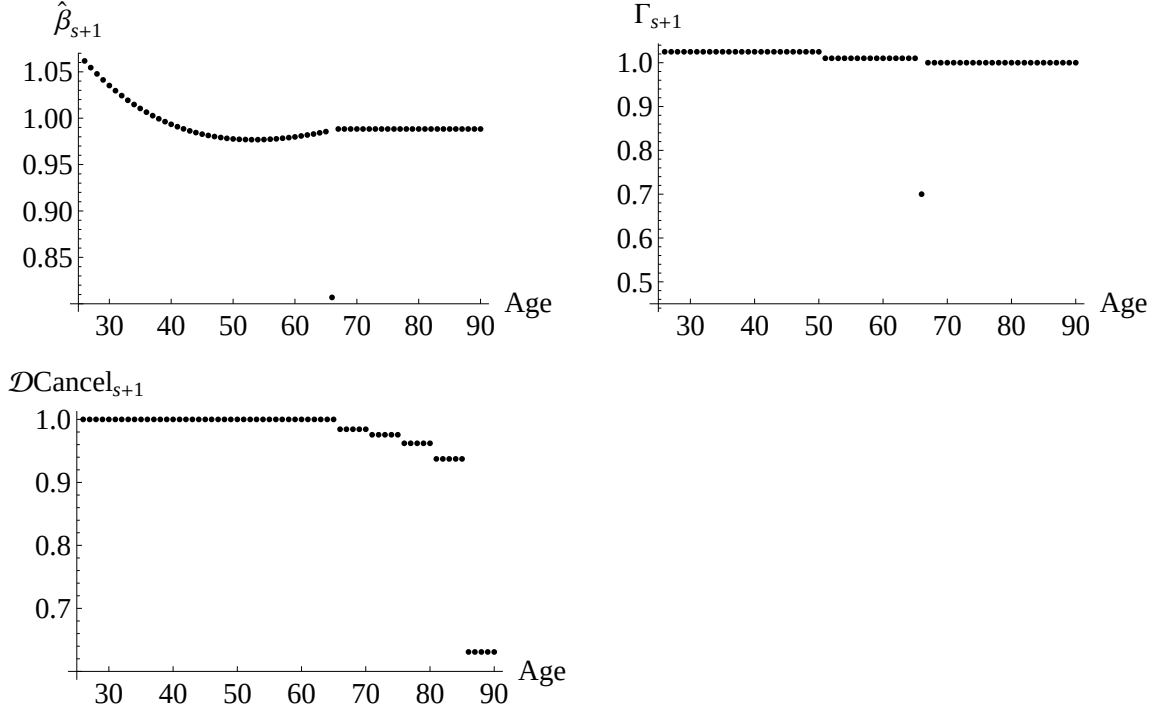


Figure 18 Time Varying Parameters

10.2 Estimation

When economists say that they are performing “structural estimation” of a model like this, they mean that they have devised a formal procedure for searching for values for the parameters β and ρ at which some measure of the model’s outcome (like “median wealth by age”) is as close as possible to an empirical measure of the same thing. Here, we choose to match the median of the wealth to permanent income ratio across 7 age groups, from age 26 – 30 up to 56 – 60.¹⁹ The choice of matching the medians rather than the means is motivated by the fact that the wealth distribution is much more concentrated at the top than the model is capable of explaining using a single set of parameter values. This means that in practice one must pick some portion of the population who one wants to match well; since the model has little hope of capturing the behavior of Bill Gates, but might conceivably match the behavior of Homer Simpson, we choose to match medians rather than means.

As explained in section 3, it is convenient to work with the normalized version of the

¹⁹Cagetti (2003) matches wealth levels rather than wealth to income ratios. We believe it is more appropriate to match ratios both because the ratios are the state variable in the theory and because empirical moments for ratios of wealth to income are not influenced by the method used to remove the effects of inflation and productivity growth.

model which can be written in Bellman form as:

$$\begin{aligned}
 v_{\sim(t)}(m_t) &= \max_{c_t} \quad u(c_t) + \beta_{t+1} \mathbb{E}_t[(\psi_{t+1} \mathcal{G}_{t+1})^{1-\rho} v_{\sim(t+1)}(m_{t+1})] \\
 &\text{s.t.} \\
 a_t &= m_t - c_t \\
 m_{t+1} &= a_t \underbrace{\left(\frac{R}{\psi_{t+1} \mathcal{G}_{t+1}} \right)}_{\equiv \mathcal{R}_{t+1}} + \theta_{t+1}
 \end{aligned}$$

with the first order condition:

$$u^c(c_t) = \beta_{t+1} \mathbb{E}_t [u^c(\psi_{t+1} \mathcal{G}_{t+1} c_{t+1} (a_t \mathcal{R}_{t+1} + \theta_{t+1}))] \cdot U \quad (61)$$

The first substantive perch in this estimation procedure is to solve for the consumption functions at each age. We need to discretize the shock distribution and solve for the policy functions by backward induction using equation (61) following the procedure in sections 6 and ‘Recursion.’ The latter routine is slightly complicated by the fact that we are considering a life-cycle model and therefore the growth rate of permanent income, the probability of death, the time-varying discount factor and the distribution of shocks will be different across the years. We thus must ensure that at each backward iteration the right parameter values are used.

Correspondingly, the first part of the `StructEstimation.py` file begins by defining the agent type by inheriting from the baseline agent type `IndShockConsumerType`, with the modification to include time-varying discount factors. Next, an instance of this “life-cycle” consumer is created for the estimation procedure. The number of periods for the life cycle of a given agent is set and, following Cagett, (2003), we initialize the wealth to income ratio of agents at age 25 by randomly assigning the equal probability values to 0.17, 0.50 and 0.83. In particular, we consider a population of agents at age 25 and follow their consumption and wealth accumulation dynamics as they reach the age of 60, using the appropriate age-specific consumption functions and the age-varying parameters. The simulated medians are obtained by taking the medians of the wealth to income ratio of the 7 age groups.

To complete the creation of the consumer type needed for the simulation, a history of shocks is drawn for each agent across all periods by invoking the `make_shock_history` function. This involves discretizing the shock distribution for as many points as the number of agents we want to simulate and then randomly permuting this shock vector as many times as we need to simulate the model for. In this way, we obtain a time varying shock for each agent. This is much more time efficient than drawing at each time from the shock distribution a shock for each agent, and also ensures a stable distribution of shocks across the simulation periods even for a small number of agents. (Similarly, in order to speed up the process, at each backward iteration we compute the consumption function and other variables as a vector at once.)

With the age-varying consumption functions derived from the life-cycle agent, we

can proceed to generate simulated data and compute the corresponding medians. Estimating the model involves comparing these simulated medians with empirical medians, measuring the model’s success by calculating the difference between the two. However, before performing the necessary steps of solving and simulating the model to generate simulated moments, it’s important to note a difficulty in producing the target moments using the available data.

Specifically, defining ξ as the set of parameters to be estimated (in the current case $\xi = \{\rho, \beth\}$), we could search for the parameter values which solve

{eq:naivePowell}

$$\min_{\xi} \sum_{\tau=1}^7 |\mathfrak{w}^{\tau} - \mathbf{s}^{\tau}(\xi)| \quad \text{U} \quad (62)$$

where $\mathfrak{w}^{\tau}$ and $\mathbf{s}^{\tau}$ are respectively the empirical and simulated medians of the wealth to permanent income ratio for age group τ . A drawback of proceeding in this way is that it treats the empirically estimated medians as though they reflected perfect measurements of the truth. Imagine, however, that one of the age groups happened to have (in the consumer survey) four times as many data observations as another age group; then we would expect the median to be more precisely estimated for the age group with more observations; yet (62) assigns equal importance to a deviation between the model and the data for all age groups.

We can get around this problem (and a variety of others) by instead minimizing a slightly more complex object:

{eq:StructEstim}

$$\min_{\xi} \sum_i^N \omega_i |\mathfrak{w}_i^{\tau} - \mathbf{s}^{\tau}(\xi)| \quad \text{U} \quad (63)$$

where ω_i is the weight of household i in the entire population,²⁰ and $\mathfrak{w}_i^{\tau}$ is the empirical wealth to permanent income ratio of household i whose head belongs to age group τ . ω_i is needed because unequal weight is assigned to each observation in the Survey of Consumer Finances (SCF). The absolute value is used since the formula is based on the fact that the median is the value that minimizes the sum of the absolute deviations from itself.

With this in mind, we turn our attention to the computation of the weighted median wealth target moments for each age cohort using this data from the 2004 Survey of Consumer Finances on household wealth. The objects necessary to accomplish this task are `weighted_median` and `get_targeted_moments`. The actual data are taken from several waves of the SCF and the medians and means for each age category are plotted in figure 19. More details on the SCF data are included in appendix A.

We now turn our attention to the two key functions in this section of the code file. The first, `simulate_moments`, executes the solving (`solve`) and simulation (`simulation`)

²⁰The Survey of Consumer Finances includes many more high-wealth households than exist in the population as a whole; therefore if one wants to produce population-representative statistics, one must be careful to weight each observation by the factor that reflects its “true” weight in the population.

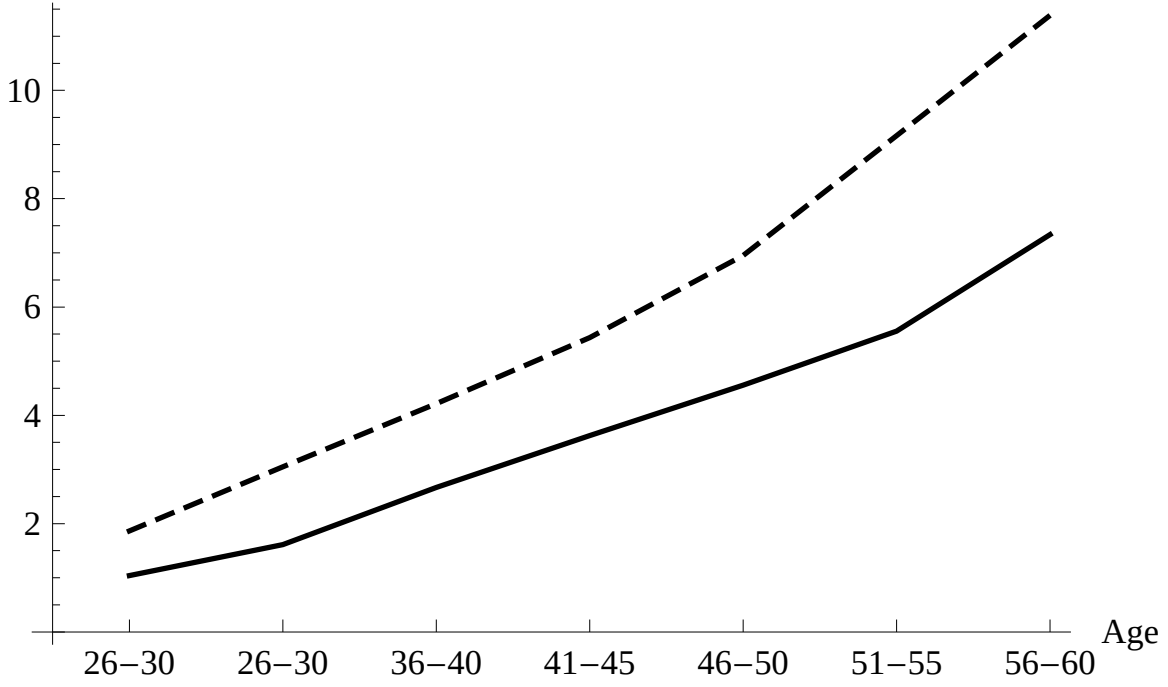


Figure 19 Wealth to Permanent Income Ratios from SCF (means (dashed) and medians (solid))

steps for the defined life-cycle agent. Subsequently, the function uses the agents' tracked levels of wealth based on their optimal consumption behavior to compute and store the simulated median wealth to income ratio for each age cohort. The second function, `smmObjectiveFxn`, calls the `simulate_moments` function to create the objective function described in (63), which is necessary to perform the SMM estimation.

Thus, for a given pair of the parameters to be estimated, the single call to the function `smmObjectiveFxn` executes the following:

1. solves for the consumption functions for the life-cycle agent
2. simulates the data and computes the simulated medians
3. returns the value of equation (63)

We delegate the task of finding the coefficients that minimize the `smmObjectiveFxn` function to the `minimize_nelder_mead` function, which is defined elsewhere and called in the second part of this file. This task can be quite slow and rather problematic if the `smmObjectiveFxn` function has very flat regions or sharp features. It is thus wise to verify the accuracy of the solution, for example by experimenting with a variety of alternative starting values for the parameter search.

The final object defined in this first part of the `StructEstimation.py` file is `calculateStandardErrorsByBootstrap`. As the name suggests, the purpose of this function is to compute the standard errors by bootstrap.²¹ This involves:

²¹For a treatment of the advantages of the bootstrap see Horowitz (2001)

1. drawing new shocks for the simulation
2. drawing a random sample (with replacement) of actual data from the SCF
3. obtaining new estimates for ρ and Ξ

We repeat the above procedure several times (**Bootstrap**) and take the standard deviation for each of the estimated parameters across the various bootstrap iterations.

10.2.1 An Aside to Computing Sensitivity Measures

A common drawback in commonly used structural estimation procedures is a lack of transparency in its estimates. As [Andrews, Gentzkow, and Shapiro \(2017\)](#) notes, a researcher employing such structural empirical methods may be interested in how alternative assumptions (such as misspecification or measurement bias in the data) would “change the moments of the data that the estimator uses as inputs, and how changes in these moments affect the estimates.” The authors provide a measure of sensitivity for given estimator that makes it easy to map the effects of different assumptions on the moments into predictable bias in the estimates for non-linear models.

In the language of [Andrews, Gentzkow, and Shapiro \(2017\)](#), section 10 is aimed at providing an estimator $\xi = \{\rho, \Xi\}$ that has some true value ξ_0 by assumption. Under the assumption a_0 of the researcher, the empirical targets computed from the SCF is measured accurately. These moments of the data are precisely what determine our estimate $\hat{\xi}$, which minimizes (63). Under alternative assumptions a , such that a given cohort is mismeasured in the survey, a different estimate is computed. Using the plug-in estimate provided by the authors, we can see quantitatively how our estimate changes under these alternative assumptions a which correspond to mismeasurement in the median wealth to income ratio for a given age cohort.

10.3 Results

The second part of the file `StructEstimation.py` defines a function `main` which produces our ρ and Ξ estimates with standard errors using 10,000 simulated agents by setting the positional arguments `estimate_model` and `compute_standard_errors` to `true`.²² Results are reported in Table 11.²³

The literature on consumption and saving behavior over the lifecycle in the presence of labor income uncertainty²⁴ warns us to be careful in disentangling the effect of time preference and risk aversion when describing the optimal behavior of households in this setting. Since the precautionary saving motive dominates in the early stages of life,

²²The procedure is: First we calculate the ρ and Ξ estimates as the minimizer of equation (63) using the actual SCF data. Then, we apply the **Bootstrap** function several times to obtain the standard error of our estimates.

²³Differently from Cagetti (2003) who estimates a different set of parameters for college graduates, high school graduates and high school dropouts, we perform the structural estimation on the full population.

²⁴For example, see [Gourinchas and Parker \(2002\)](#) for an exposition of this.

Table 11 Estimation Results

{tab:EstResults}

ρ	β
3.69	0.88
(0.047)	(0.002)

the coefficient of relative risk aversion (as well as expected labor income growth) has a larger effect on optimal consumption and saving behavior through their magnitude relative to the interest rate. Over time, life-cycle considerations (such as saving for retirement) become more important and the time preference factor plays a larger role in determining optimal behavior for this cohort.

Using the positional argument `compute_sensitivity`, Figure 20 provides a plot of the plug-in estimate of the sensitivity measure described in 10.2.1. As you can see from the figure the inverse relationship between ρ and β over the life-cycle is retained by the sensitivity measure. Specifically, under the alternative assumption that *a particular cohort is mismeasured in the SCF dataset*, we see that the y-axis suggests that our estimate of ρ and β change in a predictable way.

Suppose that there are not enough observations of the oldest cohort of households in the sample. Suppose further that the researcher predicts that adding more observations of these households to correct this mismeasurement would correspond to a higher median wealth to income ratio for this cohort. In this case, our estimate of the time preference factor should increase: the behavior of these older households is driven by their time preference, so a higher value of β is required to match the affected wealth to income targets under this alternative assumption. Since risk aversion is less important in explaining the behavior of this cohort, a lower value of ρ is required to match the affected empirical moments.

To recap, the sensitivity measure not only matches our intuition about the inverse relationship between ρ and β over the life-cycle, but provides a quantitative estimate of what would happen to our estimates of these parameters under the alternative assumption that the data is mismeasured in some way.

By setting the positional argument `make_contour_plot` to true, Figure 21 shows the contour plot of the `smmObjectiveFxn` function and the parameter estimates. The contour plot shows equally spaced isoquants of the `smmObjectiveFxn` function, i.e. the pairs of ρ and β which lead to the same deviations between simulated and empirical medians (equivalent values of equation (63)). Interestingly, there is a large rather flat region; or, more formally speaking, there exists a broad set of parameter pairs which leads to similar simulated wealth to income ratios. Intuitively, the flatter and larger is this region, the harder it is for the structural estimation procedure to precisely identify the parameters.

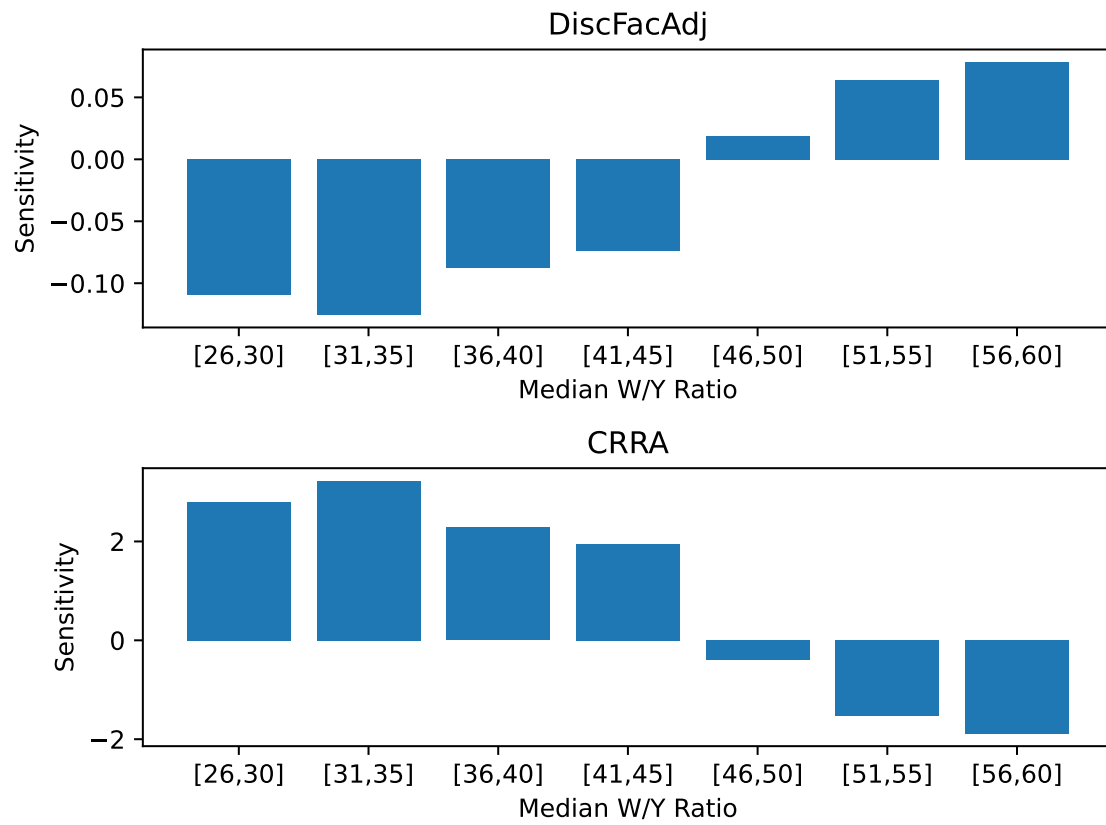


Figure 20 Sensitivity of Estimates $\{\rho, \gamma\}$ regarding Alternative Mismeasurement Assumptions.

tSensitivityMeasure}

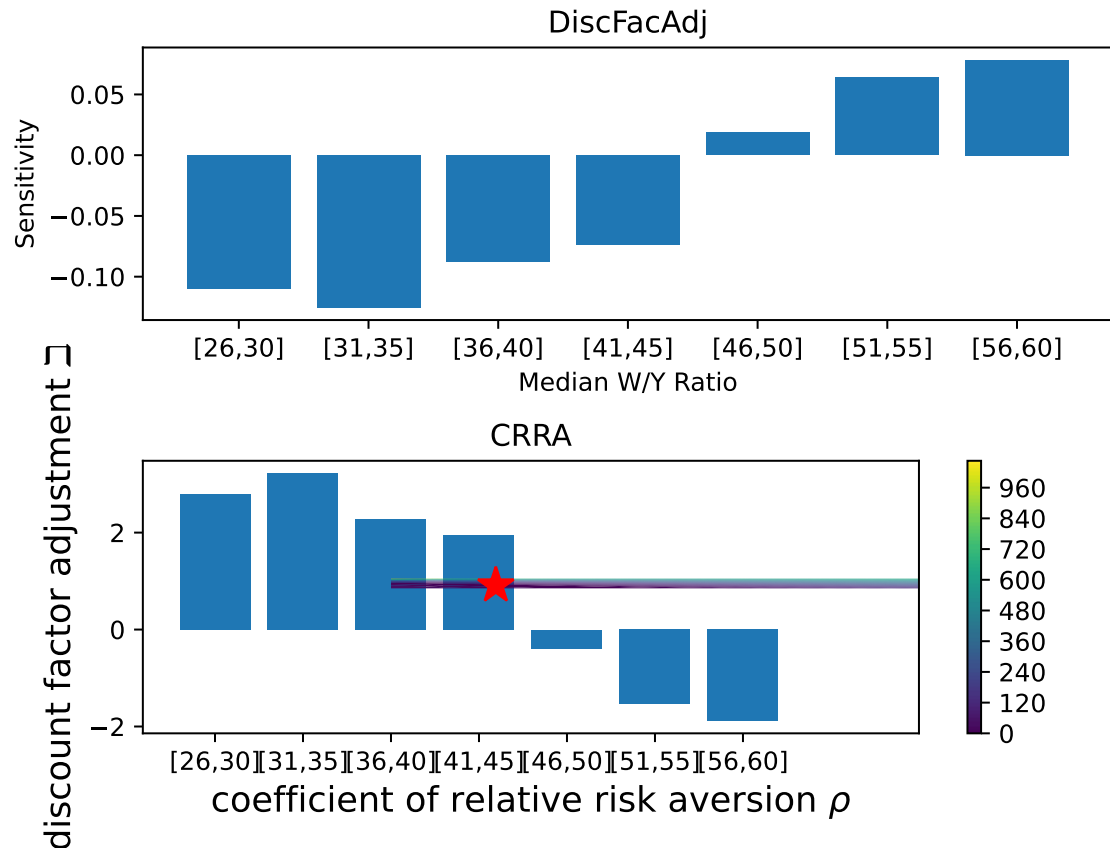


Figure 21 Contour Plot (larger values are shown lighter) with $\{\rho, \gamma\}$ Estimates (red dot).

ContourMedianStrEst}

11 Conclusion

Many choices can be made for solving microeconomic dynamic stochastic optimization problems. The set of techniques, and associated code, described in these notes represents an approach that I have found to be powerful, flexible, and efficient, but other problems may require other techniques. For a much broader treatment of many of the issues considered here, see Judd ([1998](#)).

Appendices

A SCF Data

{app:scf-data}

Data used in the estimation is constructed using the SCF 1992, 1995, 1998, 2001 and 2004 waves. The definition of wealth is net worth including housing wealth, but excluding pensions and social securities. The data set contains only households whose heads are aged 26-60 and excludes singles, following Cagetti (2003).²⁵ Furthermore, the data set contains only households whose heads are college graduates. The total sample size is 4,774.

In the waves between 1995 and 2004 of the SCF, levels of *normal* income are reported. The question in the questionnaire is "About what would your income have been if it had been a normal year?" We consider the level of normal income as corresponding to the model's theoretical object P , permanent noncapital income. Levels of normal income are not reported in the 1992 wave. Instead, in this wave there is a variable which reports whether the level of income is normal or not. Regarding the 1992 wave, only observations which report that the level of income is normal are used, and the levels of income of remaining observations in the 1992 wave are interpreted as the levels of permanent income.

Normal income levels in the SCF are before-tax figures. These before-tax permanent income figures must be rescaled so that the median of the rescaled permanent income of each age group matches the median of each age group's income which is assumed in the simulation. This rescaled permanent income is interpreted as after-tax permanent income. Rescaling is crucial since in the estimation empirical profiles are matched with simulated ones which are generated using after-tax permanent income (remember the income process assumed in the main text). Wealth / permanent income ratio is computed by dividing the level of wealth by the level of (after-tax) permanent income, and this ratio is used for the estimation.²⁶

²⁵Cagetti (2003) argues that younger households should be dropped since educational choice is not modeled. Also, he drops singles, since they include a large number of single mothers whose saving behavior is influenced by welfare.

²⁶Please refer to the archive code for details of how these after-tax measures of P are constructed.

References

- ANDREWS, ISAIAH, MATTHEW GENTZKOW, AND JESSE M SHAPIRO (2017): “Measuring the sensitivity of parameter estimates to estimation moments,” *The Quarterly Journal of Economics*, 132(4), 1553–1592.
- ATTANASIO, O.P., J. BANKS, C. MEGHIR, AND G. WEBER (1999): “Humps and Bumps in Lifetime Consumption,” *Journal of Business and Economic Statistics*, 17(1), 22–35.
- BELLMAN, RICHARD (1957): *Dynamic Programming*. Princeton University Press, Princeton, NJ.
- CAGETTI, MARCO (2003): “Wealth Accumulation Over the Life Cycle and Precautionary Savings,” *Journal of Business and Economic Statistics*, 21(3), 339–353.
- CARROLL, CHRISTOPHER D. (1992): “The Buffer-Stock Theory of Saving: Some Macroeconomic Evidence,” *Brookings Papers on Economic Activity*, 1992(2), 61–156.
- (1997): “Buffer Stock Saving and the Life Cycle/Permanent Income Hypothesis,” *Quarterly Journal of Economics*, CXII(1), 1–56.
- (2002): “Portfolios of the Rich,” in *Household Portfolios: Theory and Evidence*. MIT Press, Cambridge, MA, <https://www.econ2.jhu.edu/people/ccarroll/richportfolios.pdf>.
- (2006): “The Method of Endogenous Gridpoints for Solving Dynamic Stochastic Optimization Problems,” *Economics Letters*, 91(3), 312–320, <https://www.econ2.jhu.edu/people/ccarroll/EndogenousGridpoints.pdf>.
- (2023a): “Solving Microeconomic Dynamic Stochastic Optimization Problems,” *Econ-ARK REMARK*.
- (2023b): “Theoretical Foundations of Buffer Stock Saving,” *Revise and Resubmit, Quantitative Economics*.
- CARROLL, CHRISTOPHER D., AND MILES S. KIMBALL (1996): “On the Concavity of the Consumption Function,” *Econometrica*, 64(4), 981–992, <https://www.econ2.jhu.edu/people/ccarroll/concavity.pdf>.
- CARROLL, CHRISTOPHER D., ALAN LUJAN, KARSTEN CHIPENIUK, KIICHI TOKUOKA, AND WEIFENG WU (2026): “The Method of Moderation,” *Manuscript, Johns Hopkins University*, REMARK replication package; paper and reproducibility materials at <https://github.com/econ-ark/method-of-moderation>.
- CARROLL, CHRISTOPHER D., AND ANDREW A. SAMWICK (1997): “The Nature of Precautionary Wealth,” *Journal of Monetary Economics*, 40(1), 41–71.

- DEATON, ANGUS S. (1991): "Saving and Liquidity Constraints," *Econometrica*, 59, 1221–1248, <https://www.jstor.org/stable/2938366>.
- (1992): *Understanding Consumption*. Oxford University Press, New York.
- DEN HAAN, WOUTER J, AND ALBERT MARCET (1990): "Solving the Stochastic Growth Model by Parameterizing Expectations," *Journal of Business and Economic Statistics*, 8(1), 31–34, Available at <http://ideas.repec.org/a/bes/jnlbes/v8y1990i1p31-34.html>.
- GOURINCHAS, PIERRE-OLIVIER, AND JONATHAN PARKER (2002): "Consumption Over the Life Cycle," *Econometrica*, 70(1), 47–89.
- HOROWITZ, JOEL L. (2001): "The Bootstrap," in *Handbook of Econometrics*, ed. by James J. Heckman, and Edward Leamer, vol. 5. Elsevier/North Holland.
- JUDD, KENNETH L. (1998): *Numerical Methods in Economics*. The MIT Press, Cambridge, Massachusetts.
- KOPECKY, KAREN A., AND RICHARD M.H. SUEN (2010): "Finite State Markov-Chain Approximations To Highly Persistent Processes," *Review of Economic Dynamics*, 13(3), 701–714, <http://www.karenkopecky.net/RouwenhorstPaper.pdf>.
- MERTON, ROBERT C. (1969): "Lifetime Portfolio Selection under Uncertainty: The Continuous Time Case," *Review of Economics and Statistics*, 51, 247–257.
- PALUMBO, MICHAEL G (1999): "Uncertain Medical Expenses and Precautionary Saving Near the End of the Life Cycle," *Review of Economic Studies*, 66(2), 395–421, Available at <http://ideas.repec.org/a/bla/restud/v66y1999i2p395-421.html>.
- SAMUELSON, PAUL A. (1969): "Lifetime Portfolio Selection by Dynamic Stochastic Programming," *Review of Economics and Statistics*, 51, 239–46.
- TANAKA, KEN'ICHIRO, AND ALEXIS AKIRA TODA (2013): "Discrete approximations of continuous distributions by maximum entropy," *Economics letters*, 118(3), 445–450.
- ZELDES, STEPHEN P. (1989): "Optimal Consumption with Stochastic Income: Deviations from Certainty Equivalence," *Quarterly Journal of Economics*, 104(2), 275–298.